ESCUELA TÉCNICA SUPERIOR DE INGENIEROS

INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO TÉCNICO INDUSTRIAL ELÉCTRICO

Título del proyecto:

"DISPOSITIVO ROTATORIO PORTÁTIL INDEPENDIENTE DE LA RED ELÉCTRICA PARA EL ACOPLE DE HILOS DE FIBRA ÓPTICA"

MEMORIA

Alumno: Asier Miranda Herrero

Tutor: Dr. Carlos Ruiz Zamarreño

Pamplona, a 23 de abril de 2013

INDICE

INDICE DE FIGURAS

1. Introducción y objetivos

1.1. Introducción

El presente proyecto trata sobre el diseño y realización de un dispositivo rotatorio de pequeño tamaño, bajo coste, consumo reducido y con un sistema para poder acoplarle uno o varios hilos de fibra óptica. Además, debe ser diseñado de manera estanca para poder ser introducido en una máquina de recubrimientos por Sputtering, evaporación térmica, etc., y para la realización de tratamientos. Estos tratamientos de recubrimientos en fibras ópticas son bastante complejos, ya que al no tratarse de sustratos planos, conseguir una deposición uniforme a lo largo de la fibra es complicado. Una solución es pulir la fibra creando un sustrato plano, y otra es crear la deposición en tres giros de 120º. El dispositivo anteriormente utilizado al que va a sustituir este proyecto, consigue deposiciones uniformes en un solo paso para la fabricación de nuevos refractómetros de fibra óptica basados en recubrimientos metálicos mediante la aplicación de la técnica de Sputtering. Este dispositivo consta de un motor de continua, un potenciómetro para regular la velocidad, una pila de botón de tres voltios como sistema de alimentación y un interruptor como sistema de marcha (Figura 1).

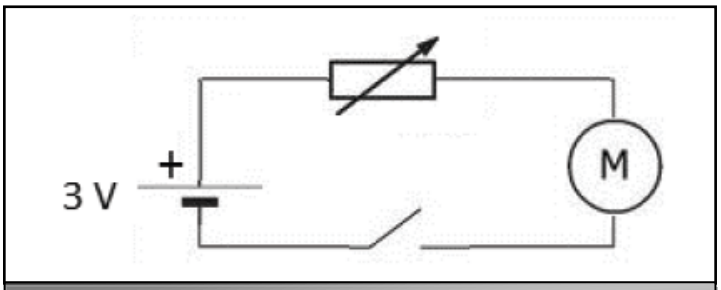


Figura 1: Esquema del anterior sistema

Los inconvenientes de este dispositivo son la poca duración de la pila, la velocidad de giro muy elevado del motor, el sistema de sujeción de las fibras al dispositivo y que sólo permite colocar una fibra por cada tratamiento (Figura 2).

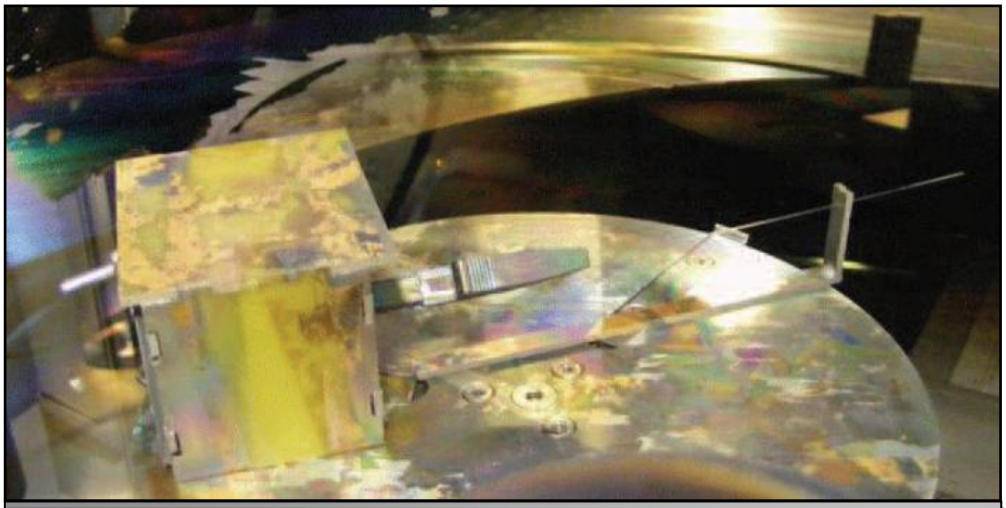


Figura 2: Dispositivo utilizado anteriormente

Debido a estos inconvenientes es necesario diseñar un nuevo sistema. La idea de este proyecto es idear un dispositivo que elimine los inconvenientes del anterior sistema y, si es posible, se le añada alguna opción más, ya sea en el aspecto de información sobre la batería y/o regulación automática de la velocidad, además de la posibilidad de poder reprogramar el dispositivo cuando se necesite o cuando se quiera variar algún aspecto en concreto del funcionamiento.

1.2. Objetivos

Los objetivos del proyecto es cumplir con las condiciones más importantes que debe tener el dispositivo: que sea independiente de la red eléctrica, con bajo consumo, que el giro del motor sea a revoluciones muy bajas y que se le puedan añadir varias fibras ópticas para que sean tratadas a la vez. Lo ideal sería conseguir todo esto con un bajo coste, pero aunar estas condiciones en un solo dispositivo no es tarea fácil, así que puede que aumente el gasto para conseguir alguno de estos objetivos.

Además, como se ha comentado antes, se le intentará añadir alguna opción más que mejore el uso del dispositivo y lo haga más versátil sin que se eleve en exceso el coste.

2. Programas y materiales utilizados

Para el desarrollo del proyecto se han necesitado diferentes materiales y programas, tanto para la parte de diseño como para la parte de construcción del dispositivo. En este apartado se van a presentar los diferentes elementos y se van a explicar a grandes rasgos sus características.

2.1 PICDEM 2 PLUS, ICD2 Y MPLAB IDE

Estos tres elementos permiten diseñar el programa que controlará el funcionamiento del microcontrolador y de todo el dispositivo, así como realizar pruebas de funcionamiento y simulaciones.

2.1.1. PICDEM 2 PLUS

La tarjeta PICDEM 2 PLUS se ha utilizado para el diseño del programa y las pruebas del microcontrolador. Se ha elegido ésta y no otra porque estaba disponible en el laboratorio y por su facilidad de manejo, ya que se conocía parte de su funcionamiento.

La tarjeta consta de tres zócalos para diferentes tipos de controladores (18, 28 y 40 pines), cuatro leds de estado más uno de indicación de encendido, tres pulsadores, un zumbador, conexión para alimentador de red, conexión para pila de 9 voltios, conector para cable RJ11, conector RS-232 y una pantalla LCD.

Figura 3: Placa PICDEM 2PLUS [1]

2.1.2. ICD 2 Y MPLAB

El MPLAB es un software de programación de microcontroladores totalmente gratuito, que permite crear programas, simularlos y volcarlos al propio microcontrolador. Es el más utilizado por programadores de PICs. También incluye el ICD2, que es un debugger o depurador, y que se conecta al ordenador con USB y a la tarjeta PICDEM por medio de un cable RJ-11, para poder trabajar como si el programa estuviera volcado en el microcontrolador, pero con la seguridad de que si hay algún fallo éste no resultará dañado y para comprobar estados de los registros del micro. Se utilizaron tanto en la simulación del programa como en la programación del micro.

Dentro del programa MPLAB se pueden utilizar diferentes compiladores. En nuestro caso hemos elegido el Hi-Tech Compiler. El compilador es un programa que va a incluir las librerías que son necesarias para el funcionamiento del programa. El Hi-Tech no tiene casi librerías, pero para nuestro uso es más que suficiente, ya que el objetivo es realizar un programa que haga lo necesario. En el caso de haber elegido otro, como por ejemplo MikroC, (que tiene muchas librerías) y haber necesitado modificarlas, no hubiéramos podido.

Figura 4: Debugger ICD2 [2]

2.2 Osciloscopio

Otro elemento bastante importante ha sido el osciloscopio. En este caso se ha empleado uno de los osciloscopios digitales del laboratorio, en concreto el modelo *DSO3062A* de la empresa ***Agilent Technologies.*** Nos ha permitido ver las señales PWM que salían del micro, llegaban al driver o salían del encoder, y ver si se correspondía con los valores de frecuencias, periodos y tensiones que queríamos nosotros. Ha sido una parte fundamental en el desarrollo del proyecto.

Figura 5: Osciloscopio

2.3 Fuente de alimentación

Todo proyecto relacionado con la electrónica necesita una fuente de alimentación. En nuestro proyecto se ha utilizado una de las del laboratorio, el modelo *FAC 662B*, de ***PROMAX Electronics.*** Se ha usado para alimentar a 5 voltios todo el circuito y para simular los 9 voltios de la pila. También, para hacernos una idea de la corriente que demandaban los motores y el posible consumo del dispositivo, así como para realizar alguna prueba para el Convertidor A/D.

Figura 6: Fuente de alimentación

2.4. DesignSpark PCB

Para realizar la placa del dispositivo se ha empleado el software *DesignSpark PCB*, que es un programa de diseño de esquemas y placas PCB gratuito de la empresa ***RS Components.*** Con una gran comunidad detrás, ofrece la posibilidad de descargarse esquemas y librerías, además de poder crear nuevos componentes y librerías y luego poder compartirlas. Herramienta esencial en la realización del proyecto. En el Anexo XX se explica con detalle el su funcionamiento, ya que si no se ha visto ningún programa similar su iniciación puede resultar algo costosa.

Figura 7: Logo de DesignSpark [37]

2.5. PROTEUS VSM

El entorno de diseño electrónico *Proteus VSM,* perteneciente a la empresa ***Labcenter Electronics***, permite el diseño de circuitos electrónicos, su simulación y el diseño de PCBs. De todas las posibilidades que ofrece, solamente se utilizó la parte de simulación, en la que la gran ventaja es que se puede ejecutar código de microcontrolador. Este programa nos posibilitó realizar pruebas previas de funcionamiento del microcontrolador y un driver.

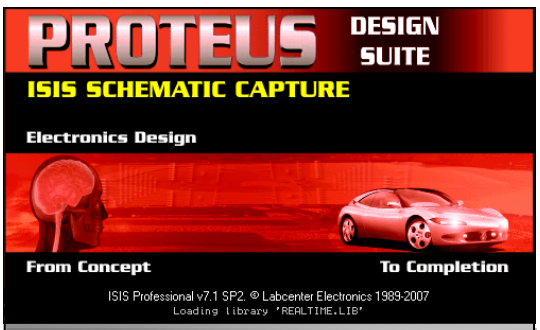


Figura 8: Logo de Proteus [36]

3. Elección de componentes

A la hora de elegir los componentes se tienen que tener en cuenta los diferentes objetivos del dispositivo:

-Pequeño tamaño-

El disco en el que se va a situar el dispositivo tiene 140mm de diámetro (Figura 9), por lo que el dispositivo no debe superar esa medida de largo. El ancho lo marcará la cantidad de motores que vayamos a poner. Un aspecto que limita mucho la construcción del dispositivo va a ser la altura que hay desde el disco base hasta los dos platos más pequeños que se encuentran a su lado. En principio, el dispositivo debe pasar por debajo de estos dos discos, así que hay que hacerlo de menor tamaño que esa altura (Figura 10).

Figura 9: Disco donde colocar el dispositivo

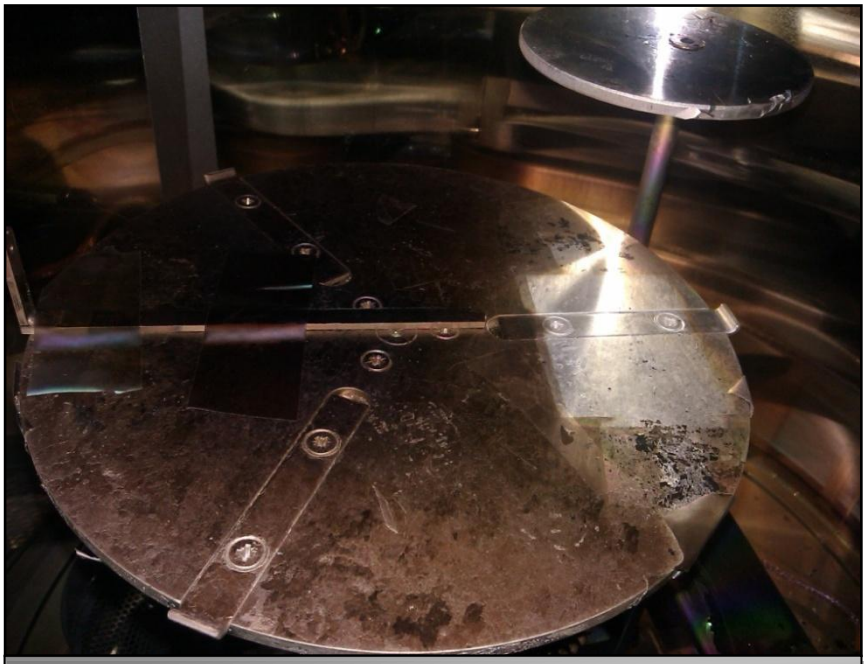


Figura 10: Altura disponible

-Bajo consumo-

Cada tratamiento dura unos 10-12 minutos. Por tanto, la batería/pila debe tener la capacidad suficiente para aguantar varias veces ese tiempo y que se puedan hacer varios tratamientos con la misma batería.

-Bajas revoluciones-

Las bajas revoluciones de giro del motor son necesarias porque las fibras ópticas son muy ligeras y si el dispositivo gira muy rápido, la fibra se deforma y la deposición no se realiza correctamente.

Para conseguir estas revoluciones se han pensado en varias opciones. Una de ellas era utilizar un Stepper o motor paso a paso, cuyo giro no se realiza de manera continua si no por pasos, es decir, giran un determinado número de grados. Su control se realiza polarizando sus bobinas de una manera adecuada para que giren correctamente. Este método se descartó porque nos interesaba un movimiento giratorio continuo y no por pasos, y para eso teníamos los servomotores, a priori más fáciles de controlar. Otra opción era la de utilizar servomotores, que son motores de continua a los que se les ha añadido la electrónica necesaria para su control, y que se controlan por medio de PWM. Otra posibilidad era utilizar motores DC y controlar su velocidad con un PWM. Y la última opción que se pensó se basaba en un motor de continua, pero acoplándole una reductora para disminuir sus revoluciones.

En el apartado ***3.2 Motor*** se explicará qué opción se ha elegido y por qué.

-Varias fibras-

Es necesario un sistema que permita incluir varias fibras para poder hacer varios tratamientos simultáneamente con el objetivo de obtener recubrimientos similares y poder estudiar mejor su efecto, y a la vez ahorrar tiempo y costes. En principio, esto implica dos posibilidades: incluir varios motores a los que cada uno se le acople un hilo de fibra, o un solo motor con un sistema que se le pueda acoplar varias fibras. La primera opción tiene el inconveniente de que aumentará el consumo al poner varios motores, y la segunda opción tiene el problema de que no es fácil ingeniar algo para colocar más de un hilo, además de que a la hora de realizar el tratamiento puede provocar problemas en la deposición. Probablemente, se elegirá la opción de varios motores.

3.1 Microcontrolador

Para controlar el dispositivo se ha pensado en utilizar un microntrolador. Un microcontrolador es un circuito integrado que está formado por las tres unidades funcionales de un ordenador: microprocesador, memoria y periféricos de entrada y salida [44].

La forma en la que funciona un microcontrolador se determina por el programa almacenado en su memoria. Este programa se puede diseñar y escribir en diferentes lenguajes de programación y tras una compilación, se descarga en la memoria interna del microcontrolador en lenguaje ejecutable. Esto, unido a su alta flexibilidad, hace que los microcontroladores se

empleen en multitud de aplicaciones: automatización, robótica, domótica, medicina, aeronáutica, automoción, telecomunicaciones, etc.

Las principales características de los microcontroladores son [45]:

-Microprocesador: normalmente de 8 bits, pero existen versiones de 4, 32 y hasta 64 bits con arquitectura Harvard, con memoria/bus de datos separada de la memoria/bus de instrucciones de programa.

-Memoria de Programa: puede ser una memoria ROM (Read Only Memory), EPROM (Electrically Programable ROM), EEPROM (Electrically Erasable/Programable ROM) o Flash. Es la encargada de almacenar el código del programa que ejecutará el microprocesador.

-Memoria de Datos: es una memoria RAM (Random Access Memory) que típicamente puede ser de 1, 2, 4, 8, 16 o 32 kilobytes.

-Generador de Reloj interno: cristal de cuarzo que produce unos impulsos con una determinada frecuencia y genera una señal oscilante. Esta frecuencia suele ir desde 31kHz a 40 MHz.

-Interfaz de Entrada/Salida: puertos paralelos, seriales (UARTs, Universal Asynchronous Receiver/Transmitter), I2C (Inter-Integrated Circuit), Interfaces de periféricos seriales (SPIs, Serial Peripheral Interfaces), Red de Área de Controladores (CAN, Controller Area Network), USB (Universal Serial Bus), etc.

-Otras opciones-

-Convertidores Analógicos-Digitales (A/D, analog-to-digital): para convertir un nivel de voltaje en un cierto pin a un valor digital manipulable por el programa del microcontrolador. Estos convertidores A/D suelen tener una resolución típica de 10 bits, aunque existen versiones de 12,16 o 32 bits.

-Moduladores por Ancho de Pulso (PWM, Pulse Width Modulation): para generar ondas cuadradas de frecuencia fija pero con ancho de pulso variable. Aunque cualquier salida digital del microcontrolador puede ser programada para hacer esta función mediante el uso de interrupciones y temporizadores, muchos microcontroladores incluyen algunas salidas especialmente dedicadas a este efecto, lo cual simplifica su uso.

3.1.1. Características necesarias

Uno de los componentes más importantes a elegir en el proyecto es el microcontrolador. Es el que nos va a permitir realizar múltiples tareas, como controlar la velocidad del motor, iluminar una secuencia de leds, reducir los componentes de la placa final, etc.

Debido a la cantidad de modelos y familias que hay la tarea no es fácil, pero a la vez, esta variedad facilitará hacernos con uno que se adapte perfectamente a nuestras necesidades.

Vamos a describir las funciones que debería tener nuestro microcontrolador.

La primera de todas es que tenga puertos de entrada y salida de varios bits, porque cabe la posibilidad de incluir varios pulsadores y leds indicadores. Según esto, los micros de gama baja no nos valen, así que tendremos que decidirnos por uno de gama media o de gama alta.

Dentro de estas gamas nos fijaremos en las familias de microcontroladores de *Microchip* **12X** y **16X**, que son las más habituales.

Para controlar el motor nos sabemos si lo vamos a hacer de forma digital o por medio de un PWM, así que se reduce la elección a la familia **16X**, que es la que tiene esta última opción.

Otra función bastante importante es que incluya un oscilador interno de diferentes frecuencias (4MHz, 8MHz…) para así evitar incluir más elementos en la placa. Este oscilador debe ser estable y fiable. Además, tiene que poder ofrecer alguna opción más, que aunque ahora no vayamos a utilizar, si podamos hacerlo en un futuro en alguna aplicación, como por ejemplo, que incluya un convertidor analógico-digital. Otro aspecto no menos importante es el del tamaño del micro. Como nuestro dispositivo debe ser muy reducido, el micro debe tener todas estas posibilidades en un encapsulado lo más pequeño posible, de alrededor de **18-20** pines.

3.1.2. Elección del microcontrolador

Atendiendo a todas estas opciones se redujo la lista a cuatro opciones. En esta tabla (Figura 11) se resumen las principales características que interesan para nuestro proyecto y si las tienen o no [3].

Características	***PIC16F84A***	***PIC16F628A***	***PIC16F87***	***PIC16F88***
Puertos de varios bits	**Sí**	**Sí**	**Sí**	**Sí**
PWM	*No*	**Sí**	**Sí**	**Sí**
Oscilador interno	*No*	**Sí**	**Sí**	**Sí**
Convertidor A/D	*No*	*No*	*No*	**Sí**
Encapsulado reducido	**Sí**	**Sí**	**Sí**	**Sí**
Interrupciones	**Sí**	**Sí**	**Sí**	**Sí**
Temporizador	**Sí**	**Sí**	**Sí**	**Sí**

Figura 11: Tabla comparativa de microcontroladores

Como vemos en la tabla, el único que cumple con todas las características es el ***PIC16F88***. Aunque no se haya mencionado, el precio siempre es un aspecto a tener en cuenta, pero en este caso, por la poca diferencia entre ellos y el hecho de que se van adquirir pocas unidades, no se ha tenido muy presente. Si habláramos de miles de unidades sí que cobra más importancia este tema ya que afecta a los costes totales de fabricación del dispositivo.

Los datos de los precios se han obtenido de la página web de microchip [3] a fecha de 10 de abril de 2013.

PIC16F84A → 3,11 euros

PIC16F628A → 1,41 euros

PIC16F87 → 2,06 euros

PIC16F88 → 2,20 euros

Como se ve, las diferencias son mínimas y para nuestro dispositivo no tiene relevancia, por lo que es mejor que tenga más posibilidades aunque se eleve algo el coste.

Además de lo mencionado, el 16F88 es, por así decirlo, el sustituto del PIC16F84A, un micro muy popular y del que se pueden encontrar numerosos manuales y ejemplos por la red en formato físico y que pueden facilitar mucho su manejo. Ambos son prácticamente iguales, pero el segundo cuenta con más opciones (más memoria, oscilador interno, PWM,..). Así pues, el micro elegido será el ***PIC16F88***. Tras la elección del microcontrolador se ha empezado a revisar su datasheet y a continuación se van a explicar sus características más importantes.

3.1.3. Características principales del PIC16F88

El PIC16F88 es el microcontrolador elegido para realizar el proyecto por las razones anteriormente comentadas. Se trata de un microcontrolador de la gama media de 8 bits de la empresa ***Microchip Technology***. Se ha elegido el encapsulado de 18-Pin PDIP por tamaño y comodidad de montaje.

Sus características más importantes son:

-Osciladores-

- Tres modos de osciladores de cristal: LP, XT y HS hasta 20MHz.

-Dos modos externos RC.

-Un modo externo de hasta 20MHz.

-Oscilador interno de 8 frecuencias: 31Khz, 125kHz, 250kHz, 500kHz, 1MHz, 2MHz, 4MHz y 8MHz.

-Periféricos-

-Módulo CCP/PWM:

-CCP: Modo captura y/o comparación de 16 bits.

-PWM: Modo Modulación de anchura de impulsos de 10 bits.

-Convertidor A/D: Analógico/Digital de 10 bits y 7 canales.

-SSP: puerto serie síncrono con SPI (Maestro/esclavo) y SPI2 (esclavo).

-USART de 9 bits.

-Comparador analógico dual.

-Características de bajo consumo.

-Algunas características especiales-

-Durabilidad de la memoria EEPROM durante 40 años.

-WatchDog Timer (WDT) programable el periodo de 1ms a 268 s.

-Tensión de operación entre 2.0 y 5.0 voltios.

-Algunas funciones no mencionadas-

-Interrupción externa.

-Interrupción por cambio en el PORTB.

-Timer1 Reloj/oscilador de bajo consumo.

-MCLR (Master clear) puede configurarse como entrada.

-16 pines de entrada/salida.

En la siguiente tabla se resumen las características básicas de microcontrolador y se compara con el PIC16F87, con el que coincide en todo menos en el convertidor A/D.

Device	Program Memory		Data Memory		I/O Pins	10-bit A/D (ch)	CCP (PWM)	AUSART	Comparators	SSP	Timers 8/16-bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)							
PIC16F87	7168	4096	368	256	16	N/A	1	Y	2	Y	2/1
PIC16F88	7168	4096	368	256	16	1	1	Y	2	Y	2/1

Figura 12: Resumen características del 16F88/16F87

-Organización de la memoria-

En nuestro caso, dado que el microcontrolador se va a programar en lenguaje C no es tan necesaria la información de la memoria, pero sí vamos a explicar algunos detalles de su distribución.

La memoria disponible en el dispositivo:

Device	Program Flash	Data Memory	Data EEPROM
PIC16F87/88	4K x 14	368 x 8	256 x 8

Figura 13: Memoria disponible

1.-Memoria de programa de 4K (FLASH) organizada en 2 páginas (accesible mediante PCLATH).

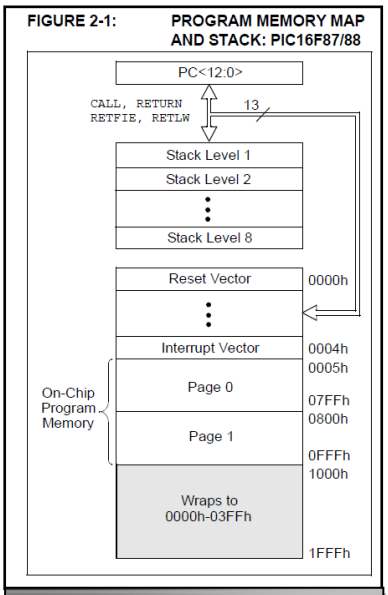


Figura 14: Memoria FLASH

2.- Cuatro bancos de memoria de datos (RAM) organizada en registros, accesibles mediante *RP1/RP2* (status).

RP1:RP0	Bank
00	0
01	1
10	2
11	3

Figura 15: Bancos de memoria

-Mapa de los registros disponibles-

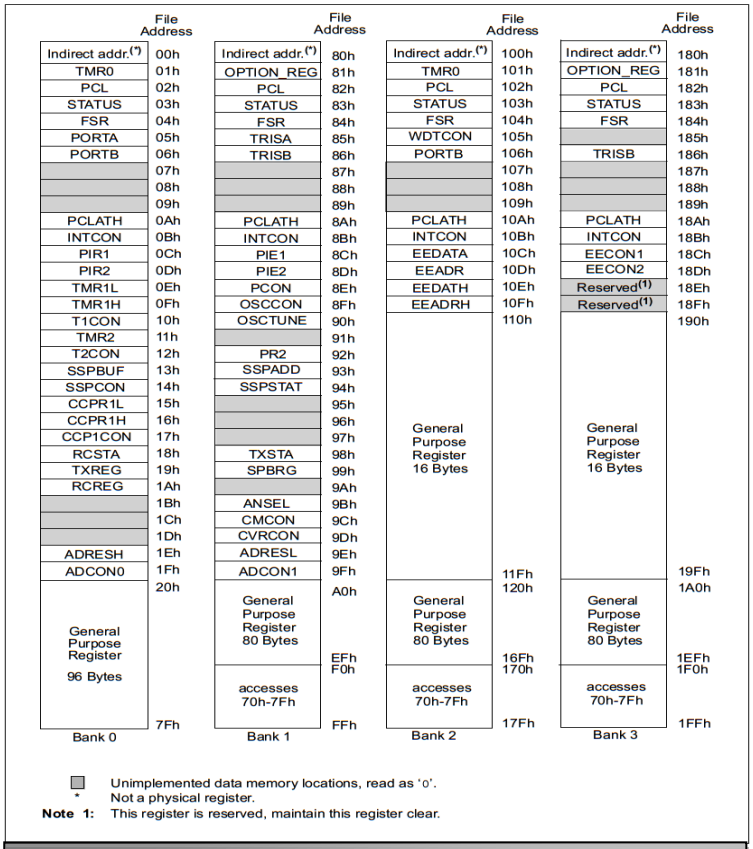


Figura 16: Mapa de registros

Los registros utilizados en nuestro proyecto para configurar el micro serán explicados más adelante en el apartado ***5.1. Programa y registros utilizados***.

3.2. Motor

Un motor eléctrico es una máquina que transforma la energía eléctrica en energía mecánica, y generalmente en movimiento rotativo. Existen diferentes tipos de motores eléctricos: de corriente continua, de corriente alterna, paso a paso, servomotores, etc. Para elegir nuestro motor, por las características de nuestro proyecto, se han reducido las opciones a servomotores y motores de corriente continua. Los de alterna no se utilizan en sistemas tan pequeños y los paso a paso se utilizan para controlar la posición, lo cual no nos interesa en este proyecto.

Los motores de continua y los servomotores son dos elementos muy utilizados en el campo de la electrónica. Los dos tienen partes en común y diferencias, ventajas e inconvenientes según para la aplicación que se utilicen.

Vamos a exponer las principales características de cada uno de ellos, siempre referido a nuestra aplicación. Y finalmente se explicará cuál se ha elegido y por qué.

3.2.1. Motores de continua

Los motores de continua están formados por un estator, el cual da soporte mecánico a la máquina y posee una cavidad en el centro de éste, y un rotor al que le llega la corriente procedente del estator a través de las escobillas. Proporciona un par de giro proporcional a la corriente inyectada en su armadura. Tienen velocidades de giro muy altas, y por esta razón depende de para qué aplicación se utilice es necesario utilizar cajas reductoras para adaptar la velocidad a un rango usable.

Una forma muy habitual de controlar los motores de continua es por medio del PWM (P*ulse Width Modulation)* o modulación de anchura de impulsos. Es un sistema en el que se modifica el ciclo de trabajo de la señal que le introducimos al motor.

De esta manera cuanto mayor es el ciclo de trabajo más tensión le llegará al motor, ya que lo que ve el motor es una tensión media y girará más rápido. Si queremos que el motor gire más lento hay que reducir el ciclo de trabajo, y como consecuencia se reducirá la tensión media.

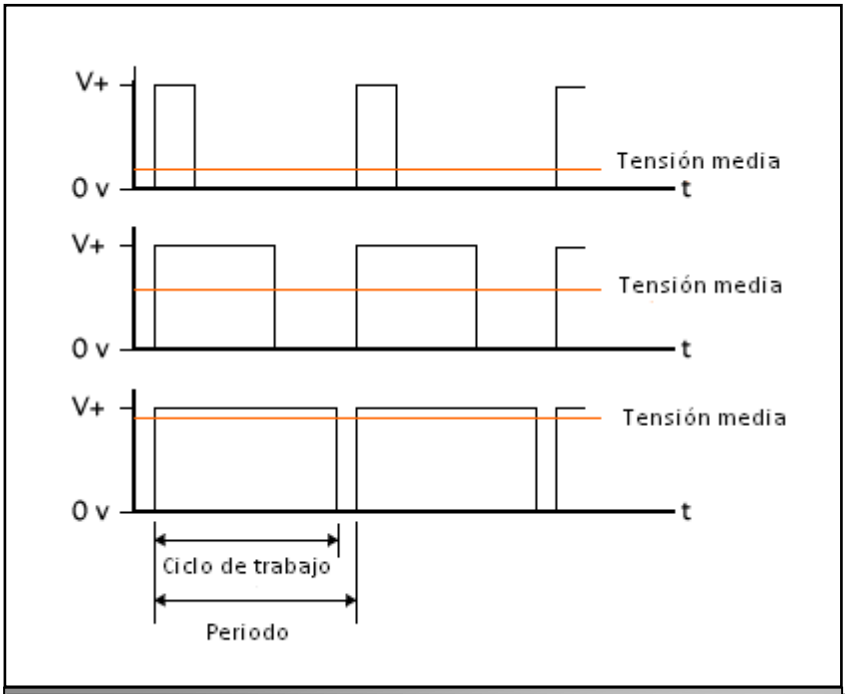


Figura 17: Gráfica PWM [4]

Fijándonos en nuestra aplicación, el principal problema de los motores de continua son sus altas velocidades, ya que estos están diseñados para girar normalmente entre 2000 y 15000 rpm., y para nuestro proyecto es necesario que gire a revoluciones muy bajas, cuanto más bajas mejor, ya que la fibra óptica es muy ligera y a revoluciones altas puede deformarse, llegar a combarse e incluso romperse. Esto se puede solucionar incluyendo en el motor una reductora. Las hay de muchas relaciones (5:1, 10:1, 50:1, 250:1,…) y también vienen motores con la reductora ya incorporada.

Debido al problema de consumo que podemos tener en el dispositivo, si se reduce la tensión de salida sin nosotros habérselo ordenado al micro, la velocidad de giro también bajará, cosa que no queremos. Para solucionarlo caben varias posibilidades. Una de ellas es estar comprobando la tensión de salida constantemente, cosa poco eficiente para el propio consumo del micro. Otra opción es utilizar las interrupciones de las que disponemos. La idea es utilizar un ***timer*** que se desborde cada cierto tiempo y genere una interrupción; cuando el micro acuda a atender la interrupción compruebe la velocidad a la que gira el motor y si es menor de la deseada, que aumente el ciclo de trabajo hasta llegar a la velocidad adecuada.

Para ello es necesario incluir un encoder al circuito o adquirir un motor con encoder ya incorporado. Por lo tanto si elegimos esta opción debe constar de las siguientes partes: motor de continua, reductora y encoder.

3.2.2. Servomotores

Los servomotores son motores de continua que se han modificado para que puedan ser controlados tanto en velocidad como en posición [40].

Las partes principales que forman un servo son el motor de continua, un juego de engranajes para la reducción de la velocidad, un potenciómetro colocado sobre el eje de salida que se utiliza para saber la posición del servo, y una placa de control. Los hay de dos tipos: analógicos y digitales.

En construcción sólo difieren en la placa de control (más sofisticada en los digitales), los demás elementos son iguales. Solamente en el funcionamiento y en su control es en lo que difieren entre sí. Otro aspecto común para los dos es que los dos tipos constan de tres cables: rojo (alimentación entre 4 y 6 voltios), negro (masa) y amarillo o blanco (pulsos de señal para el control).

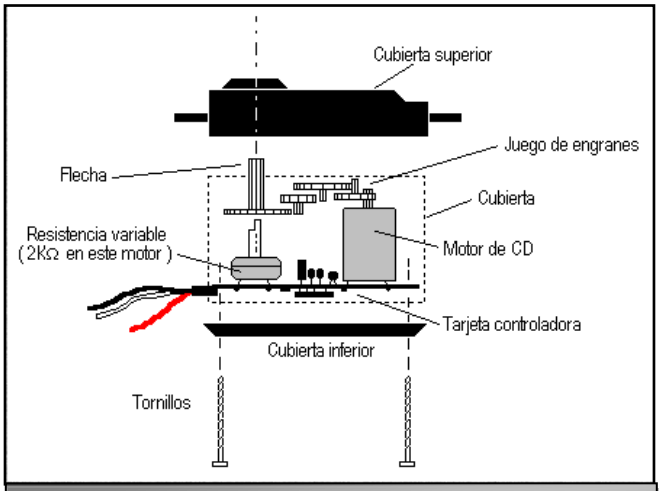


Figura 18: Partes de un servomotor [6]

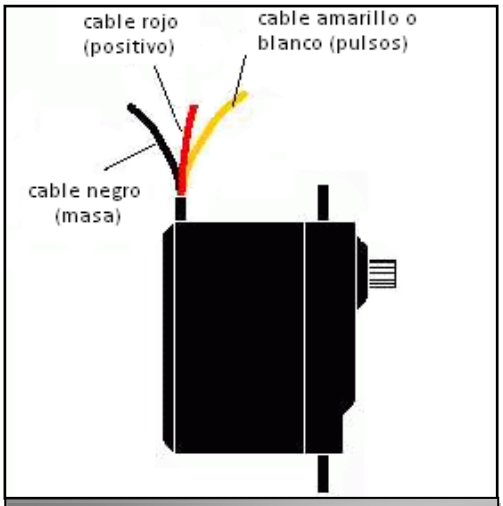


Figura 19: Cables del servo [5]

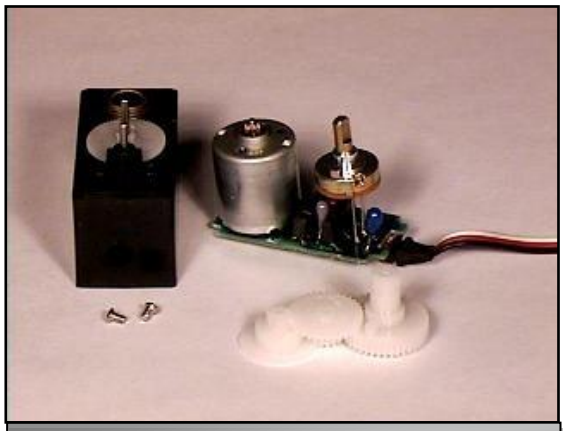


Figura 20: Servo real [5]

Se van a explicar un poco más en profundidad los servomotores analógicos y digitales y su control.

3.2.2.1. Servomotores analógicos

El servo analógico funciona de la siguiente manera [41]: la electrónica del servo coloca al motor en cada posición dependiendo del tiempo en que el pulso que le inyectamos permanece en alto. Si el tiempo que dura el pulso en estado alto es de exactamente 1,5 milisegundos entonces el servo se coloca en el centro de su recorrido, si el pulso dura exactamente 0,5 milisegundos el servo retrocede desde el punto medio unos 90º y se coloca en su extremo izquierdo y si, por último, el pulso dura exactamente 2,5 milisegundos el servo avanza desde el punto medio unos 90º y se coloca en su extremo derecho. Estos valores de ciclo de trabajo son orientativos, ya que dependen del fabricante del servo. En la siguiente tabla se pueden ver algunos ejemplos:

Ancho de pulso (msg)				
Fabricante	***mín***	***neutral***	***máx***	***Frecuencia (Hz)***
Futaba	0.9	1.5	2.1	50
Hitech	0.9	1.5	2.1	50
Graupner/Jr	0.8	1.5	2.2	50
Multiplex	1.05	1.6	2.15	40

Figura 21: Tabla ejemplos servos [7]

Los pulsos están separados unos de otros 20 milisegundos, que es exactamente lo mismo que decir que se envían con una frecuencia de 50 Hz, ya que 50 Hz son 50 pulsos por segundo y por lo tanto 1000 milisegundos (que tiene un segundo) dividido entre 50 son exactamente 20 milisegundos (**f = 1 / t**).

En la siguiente figura se ve la posición que ocuparía el servo según el ciclo de trabajo del PWM.

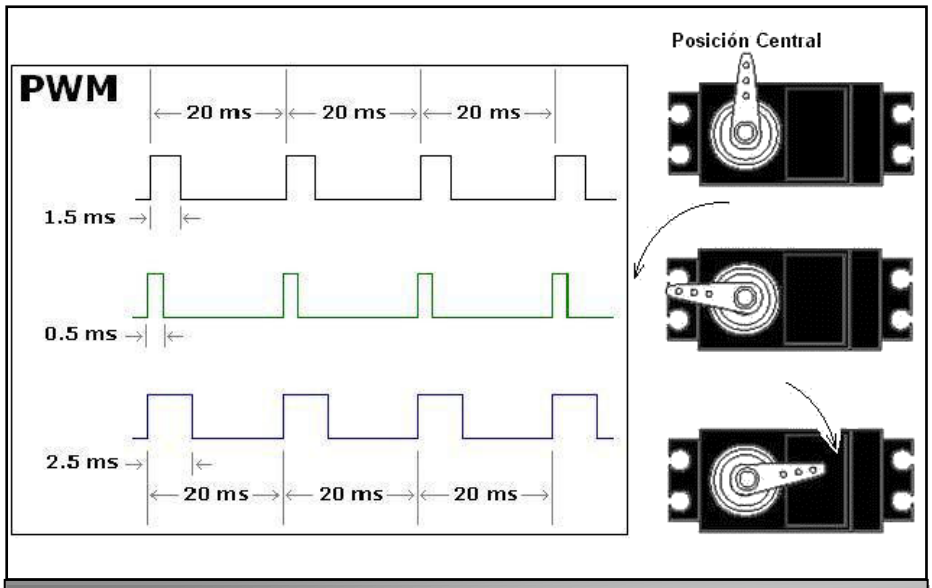


Figura 22: Posición servo según ciclo de trabajo [5] [8]

El control de estos pulsos se puede hacer con el PWM del microcontrolador o con interrupciones internas, aunque puede ser más recomendable con interrupciones para controlar exactamente el tiempo que está en alto el impulso (el tiempo que dura el ciclo de trabajo). Con los temporizadores del microcontrolador y conociendo la frecuencia de funcionamiento del PIC es bastante sencillo calcular cuándo queremos que se desborde el temporizador produciéndose la interrupción.

3.2.2.2. Servomotores digitales

Como ya se ha comentado, los servos digitales [41] difieren en su construcción respecto a los analógicos solamente en la placa de control, a la que se le ha agregado un microcontrolador que se encarga de analizar la señal, procesarla y controlar el motor.

La diferencia más grande de rendimiento está en la velocidad a la que reacciona el servo a un cambio en la señal. En un mismo lapso de tiempo, el servo digital puede recibir cinco o seis veces más pulsos de control que un analógico. Como resultado la respuesta del servo a un cambio en la orden de posición es mucho más veloz.

Este aumento en la velocidad de los pulsos también se traduce en mejoras en el rendimiento electromecánico del motor (mayor velocidad y fuerza). Esto es posible debido a que el micro es capaz de modificar el ciclo de trabajo en función de unos parámetros de funcionamiento interno.

En segundo lugar es capaz de aumentar la frecuencia de trabajo, si con un servo estándar teníamos 50 ciclos por segundo ahora podremos tener hasta 300 ciclos por segundo con lo cual la duración del periodo baja hasta los 1/300 = 3,33 milisegundos. Lógicamente al disminuir el periodo proporcionalmente también disminuirá el ancho de pulso manejable, pero el ciclo de trabajo permanecerá constante, con lo cual conseguimos enviar pulsos mucho más estrechos pero con más frecuencia.

Debido a las características constructivas y de funcionamiento de cualquier motor eléctrico se da la circunstancia de que es precisamente esta situación en la que se obtiene un mayor rendimiento del mismo, ya que con frecuencias muy altas no se descarga la bobina equivalente creada por el inducido del motor y los picos de corriente son menores, es por tanto más efectivo, en general en un motor el rendimiento es proporcional a la frecuencia de trabajo.

Con este aumento de potencia no sólo se consigue aumentar la velocidad de respuesta, sino una aceleración/deceleración mucho más rápida y suave, mayor resolución en el posicionamiento y un mayor par, dicho aumento de par se ve reflejado tanto en funcionamiento estático como dinámico, es decir, cuando el servo está detenido en una posición, la fuerza que hay que ejercer sobre el brazo del mismo para conseguir que gire es muy superior a la de un servo estándar, asimismo el par de giro suministrado cuando está realizando un desplazamiento es tres veces superior al de un servo estándar.

En la siguiente figura se puede ver una comparación de la frecuencia de los pulsos entre un servo standard o analógico y uno digital.

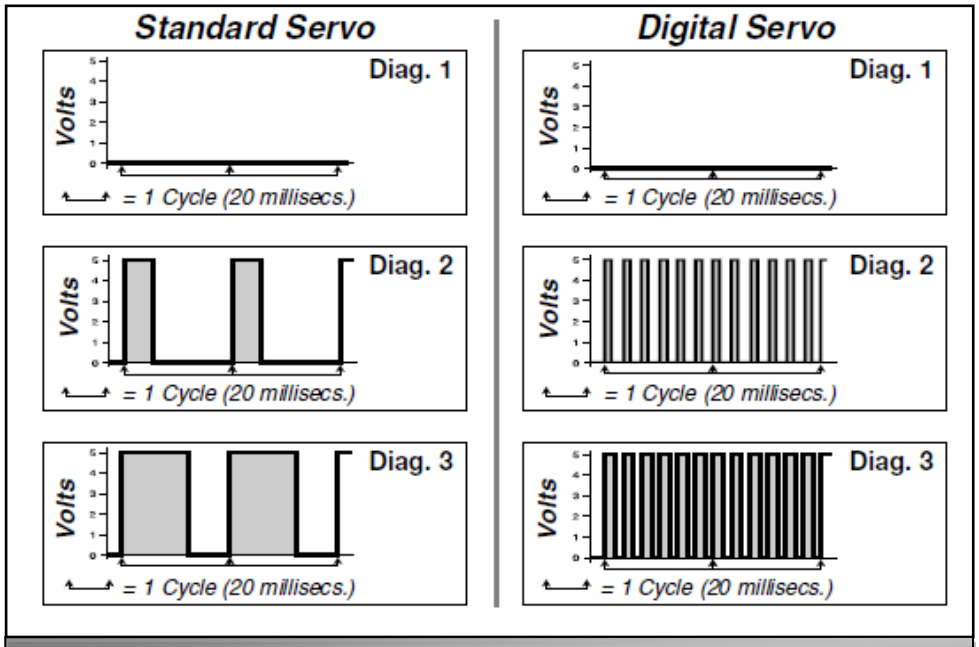


Figura 23: Diferencias ciclo de trabajo entre servo digital y analógico [9]

Pero no todo son ventajas en los servos digitales. El consumo es algo mayor que en los analógicos debido al circuito de la placa de control y a los ajustes más continuados que producen un gasto mayor de energía, y también un mayor desgaste del motor.

También al mantener un ciclo de trabajo idéntico al de un servo estándar pero aumentar la frecuencia lógicamente también aumenta el consumo, dicho aumento de consumo sería aproximadamente un 60% superior aun servo estándar de "similares" prestaciones. Es por ello que dependiendo del número de servos que vayamos a instalar y las características de consumo de los mismos tendremos que tener cuidado de vigilar la carga de las baterías [41].

El eje del servo tiene una rotación limitada de 180º, pero se pueden encontrar servos diseñados para que giren 360º y tengan una rotación continua como un motor.

3.2.3. Elección del motor

Una vez explicado las características de los motores de continua y de los servomotores, se van a presentar algunos dispositivos de los dos tipos (servos y de continua) que se han encontrado en diferentes catálogos con sus características, atendiendo a los diferentes apartados que nos interesan para así poder decidirnos mejor.

3.2.3.1. Características de los posibles motores

-Posibles motores-

Servomotores

Power HD Micro Servo HD-1800A [10]→

Figura 24: Servo HD-1800A

-*Power HD Micro Servo HD-1160A* [11]→

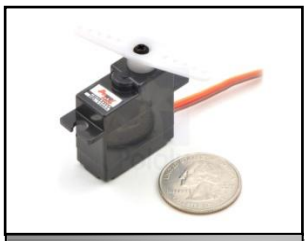


Figura 25: Servo HD-1160A

-*Power HD Micro Servo digital DS65HB* [12]→

Figura 26: Servo DS65HB

Motores DC

-*Micro metal DC (con reductora)* [13] →

Figura 27: Micro metal

-*Precision Microdrives 24mm DC Motor* [14] →

Figura 28: Precision Microdrives Motor

-*Mechtex standard DC motor DC25* [15]→

Figura 29: Mechtex DC25

-*Mitsumi DC M10E-2Series* [16] →

Figura 30: Mitsumi DC M10E-2

-CONSUMO-

Los consumos en los servos tienden a ser mayores que en los motores de continua pudiendo ser debido a la electrónica que incorpora el propio servo, que por otra parte le hace ser mucho más preciso en posicionamiento y velocidad.

Servomotores

-*Power HD Micro Servo HD-1800A* →	4,8V: **120mA** (sin carga)
	6,0V: **140mA** (sin carga)
-*Power HD Micro Servo HD-1160A* →	4,8V: **160mA** (sin carga)
	6,0V**: 180mA** (sin carga)
-*Power HD Micro Servo digital DS65HB* →	4,8V**: 200mA** (sin carga)
	6,0V: **220mA** (sin carga**)**

Motores DC

-*Micro metal DC (con reductora)* →	6,0V: **50mA** (360mA máx)
-*Precision Microdrives 24mm DC Motor* →	2V: **60 mA** (sin carga)
-*Mechtex standard DC motor DC25* →	6V: **60 mA** (sin carga)
-*Mitsumi DC M10E-2Series* →	3,3 V: **40mA** (sin carga)

Según los datos de consumo los motores de continua son mejor opción que los servomotores, factor bastante determinante ya que es difícil reducir ese consumo variando el diseño del dispositivo.

-DIMENSIONES-

Para nuestra aplicación el tamaño debe ser lo más pequeño posible y que mantenga las características técnicas necesarias de consumo y control.

Servomotores

-Power HD Micro Servo HD-1800A → Largo: **31,6mm**
Ancho: **11,56mm**
Alto: **24mm**

-Power HD Micro Servo HD-1160A → Largo: **40mm**
Ancho: **13,20mm**
Alto: **29,6mm**

-Power HD Micro Servo digital DS65HB → Largo: **30mm**
Ancho: **11,5mm**
Alto: **20mm**

Motores DC

-Micro metal DC (con reductora) → Largo: **24mm**
Ancho: **12mm**
Alto: **10mm**

-Precision Microdrives 24mm DC Motor → Diámetro: **24mm**
Ancho: **8mm**

-Mechtex standard DC motor DC25 → Diámetro: **24mm**
Alto: **20,1mm**

-Mitsumi DC M10E-2 Series → Largo: **16,2mm**
Ancho: **10mm**
Alto: **8mm**

En el aspecto del tamaño también es claramente mejor opción el motor de continua, aspecto que no se puede modificar.

-VELOCIDAD-

Como se ha explicado anteriormente, los servomotores se pueden diseñar para que giren a bajas revoluciones y, a priori, es mejor opción que los motores de continua. Pero a estos últimos se le pueden acoplar reductoras que permiten reducir su velocidad, además hay muchos ratios de reducciones a elegir en el mercado. Veamos las diferentes velocidades de los motores.

Servomotores

-Power HD Micro Servo HD-1800A →	4,8V: **0,1 segundos/60º**
	6,0V: **0,08 segundos/60º**
-Power HD Micro Servo HD-1160A →	4,8V: **0,12 segundos/60º**
	6,0V**: 0,11 segundos/60º**
-Power HD Micro Servo digital DS65HB →	4,8V**: 0,1 segundos/60º**
	6,0V: **0,08 segundos/60º**

Motores DC

-*Micro metal DC (con reductora)* →	3 a 9V: **45 rpm** (sin carga)
-*Precision Microdrives 24mm DC Motor* →	2 V: **5.300 rpm** (sin carga)
-*Mechtex standard DC motor DC25* →	6V: **600 a 7.700 rpm** (sin carga)
-*Mitsumi DC M10E-2Series* →	3,3 V: **11.200 rpm** (sin carga)

En principio la mejor opción atendiendo a la velocidad de giro sería la de los servomotores, pero hay que tener en cuenta que los motores de continua no llevan incorporada la reductora.

-PRECIO-

El precio de los servos y de los motores son bastante parecidos, alrededor de 10 euros, a esto hay que añadir los gastos de envío que depende de la pagina web elegida.

Servos

-Power HD Micro Servo HD-1800A → **9,95 euros**

-Power HD Micro Servo HD-1160A → **9,95 euros**

-Power HD Micro Servo digital DS65HB → **9,45 euros**

Motor DC

-Micro metal DC (con reductora) → **13,20 euros**

-Precision Microdrives 24mm DC Motor → **10,20 euros**

-Mechtex standard DC motor DC25 → No disponible

-Mitsumi DC M10E-2Series → No disponible

-RESUMEN

Para una mejor visualización de las características de los motores se ha realizado la siguiente tabla resumen:

Modelos	Consumo		Dimensiones (mm)			Velocidad		Precio
	V	*I*	*Largo*	*Ancho*	*Alto*	*Tensión*	*Velocidad*	*Euros*
Servo 1800A	4,8 V	120 mA	31,6	11,56	24	4,8 V	0,1s/60º	9,95
	6,0 V	140 mA				6,0 V	0,08s/60º	
Servo 1160A	4,8 V	160 mA	40	13,20	29,6	4,8 V	0,12s/60º	9,95
	6,0 V	180 mA				6,0 V	0,11s/60º	
Servo DS65HB	4,8 V	200 mA	30	11,5	20	4,8 V	0,1s/60º	9,45
	6,0 V	200 mA				6,0 V	0,08s/60º	
Micro Metal DC	6,0 V	50 mA	24	12	10	3 a 9 V	45rpm	13,20
Precision Microdrives	2,0 V	60 mA	Diámetro: 24		8	2,0 V	5300rpm	10,20
Mechtex DC25	6,0 V	60 mA	Diámetro: 24		20,1	6,0 V	600-7700rpm	No disp.
Mitsumi DC M10	3,3 V	40 mA	16,2	10	8	3,3V	11200rpm	No disp.

Figura 31: Tabla resumen características de los motores y servomotores

Las ventajas que tienen los servos respecto a los motores de continua es el poder controlar la velocidad de giro muy fácilmente y hacerlo girar a velocidades bastante bajas, que para nuestra aplicación es muy importante. Por el contrario fallan en el tema del consumo, que es claramente mayor que en los motores de continua, y en el tamaño.

La principal ventaja del motor de continua es el menor consumo respecto a los servos y también sale ganador en el apartado de tamaño. Por otra parte, el principal inconveniente es que gire a muy pocas revoluciones ya que los que cumplen los requisitos de consumo y tamaño están diseñados para que giren entre 2000 y 15000 rpm.

Una buena opción podría ser utilizar un motor de corriente continua y acoplarle un reductor y así tendríamos bajo consumo y bajas velocidades.

3.2.3.2. Elección del motor final

Fijándonos en el consumo, el motor de continua tiene mejores datos y el problema de sus altas revoluciones se puede solucionar con una reductora. Así que se decidió que la mejor opción es la de un motor de continua con una reductora y además añadirle un encoder para poder hacer lecturas de velocidad.

La idea era adquirir el motor "Micro Metal DC con reductora" y añadirle un encoder, pero buscando información acerca del motor se encontró que el fabricante (TTMotor) tenía en su catálogo el mismo motor pero con un encoder ya acoplado, y aun precio prácticamente igual (10 euros). Así pues nos pusimos en contacto con el fabricante y se consiguió comprar ese motor. De esta manera se ahorra espacio para la placa y dinero evitando comprar más componentes

El motor elegido fue el ***GM12-N20VA-EN*** de la empresa **TTMotor**, en concreto se cogieron las versiones con las reductoras 298:1 y 100:1. Se han escogido estos valores de reducción porque eran los más adecuados, ya que en el caso de 50:1 era poca reducción y en el caso de 1000:1 era excesiva [24].

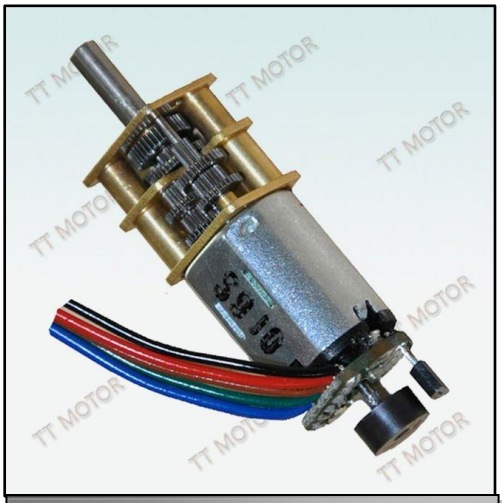


Figura 32: Motor GM12-N20VA [24]

-Características del motor elegido-

Modelo		Tensión		Sin carga		Con carga				Parada	
	Reducción	Rango operación	Nominal	Velocidad	Corriente	Velocidad	Corriente	Par	Salida	Par	Corriente
			V	rpm	A	rpm	A	N. m	W	N. m	A
GM12-N20VA-EN	50:1	2.0-5.0	5.0	285	0.040	240	0.15	0.007	0.35	0.038	0.52
	100:1	2.0-5.0	5.0	145	0.040	120	0.15	0.015	0.35	0.076	0.52
	298:1	2.0-5.0	5.0	50	0.040	40	0.15	0.045	0.35	0.200	0.52
	1000:1	2.0-5.0	5.0	14	0.040	12	0.15	0.100	0.35	0.300	0.52

Figura 33: Tabla de las características del motor elegido GM12-N20VA

3.3. Sistema de alimentación

La forma de alimentar el dispositivo también es una de las partes más importantes del dispositivo, ya que al ser independiente de la red eléctrica había que conseguir bastante duración de la batería para que se puedan realizar varios procesos y así evitar aumento de costes reponiéndola constantemente, y había que conseguirlo en un espacio muy reducido.

Primero se pensó en la posibilidad de colocar varias pilas AAA en serie y ya que tienen entre 900 y 1200mAh de capacidad y sumando sus tensiones se pueden tener unos 5 V, que es lo mínimo para que funcione todo el sistema, pero se descartó porque al ser 1,5 voltios, hay que poner varias y suponía un problema de espacio.

También se pensó en baterías portátiles pero también daban problemas de espacio.

Finalmente se optó por realizar unas pruebas con pilas de 9V, que tienen unos 200-250mAh de capacidad y añadirle al circuito un regulador de tensión para pasar a 5V y que alimente a todos los elementos del dispositivo. En las pruebas se comprobó que con una sola pila de este tipo y, teniendo en cuenta que cada proceso dura unos 12 minutos, se podían hacer 10 procesos, lo que equivale a 30 fibras ópticas tratadas. De esta manera se solucionó el problema de la alimentación y además, en lugar de utilizar pilas alcalinas convencionales se decidió utilizar pilas recargables que tienen prácticamente el mismo rendimiento y así reducir los costes. Se adquirieron pilas de 9V de Ni-MH (Niquel-Metal Hidruro) de la empresa ***Ansmann*** en la página de ***RS Components.***

Figura 34: Pila de 9V [25]

3.4 Otros componentes

En este apartado se van a explicar otros componentes que se han incorporado, unos para que tenga un funcionamiento básico y consiga los objetivos principales y otros para que además el dispositivo tenga unas características adicionales al funcionamiento básico.

3.4.1. Driver L293D

La salida PWM del microcontrolador puede sacar como máximo 20mA. Debido a esto no se puede conectar directamente esta salida a la entrada del motor, ya que no es corriente suficiente para excitar las bobinas del mismo. Para solucionar este problema se pensó en incluir un driver que permitiera aumentar esta corriente. Se pensó en driver **L293D** [51].

Se ha elegido este integrado porque se vio en una aplicación parecida a la nuestra y con muy buen resultado. A continuación se va explicar más a fondo cómo es este driver y las posibilidades que presenta.

El **L293D** es un integrado con cuatro circuitos internos para manejar motores de pequeña potencia y con la capacidad de controlar corriente hasta 600 mA y picos de 1,2 A en cada circuito y una tensión entre 4,5 V a 36 V.

En esta imagen se presenta una vista general del integrado. Con las patillas de habilitación (pines 1 y 9) entradas de señal 1A, 2A, 3A y 4A y alimentación **Vcc1**, **Vcc2** y las de masa pines 4,5, 12 y 13.

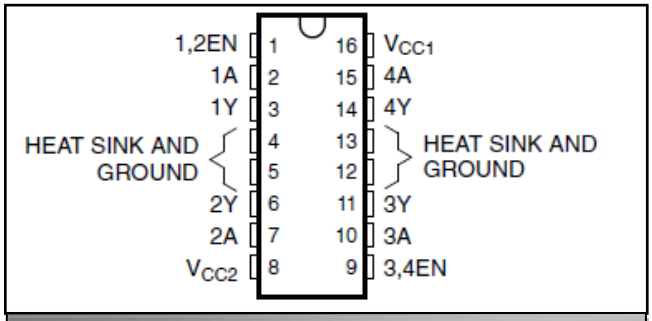


Figura 35: Vista general del L293D

En estas imágenes se presentan los diagramas de pines y el simplificado.

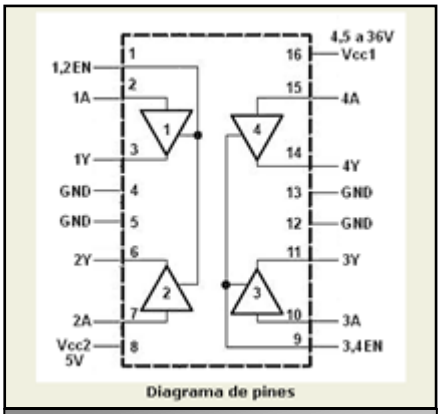


Figura 36: Diagrama de pines del L293D

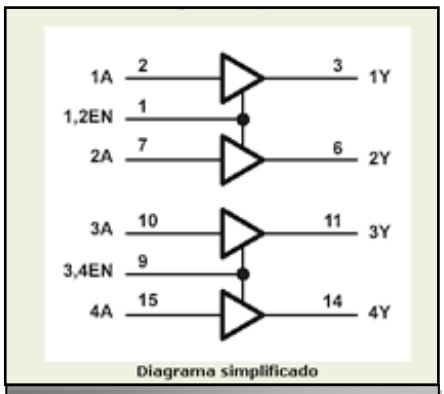


Figura 37: Diagrama simplificado del L293D

Las salidas tienen un diseño que permite el manejo directo de cargas inductivas tales como relés, solenoides, motores de corriente continua y motores por pasos, ya que incorpora internamente los diodos de protección de contracorriente para cargas inductivas.

Las entradas son compatibles con niveles de lógica TTL. Para lograr esto, incluso cuando se manejen motores de voltajes no compatibles con los niveles TTL, el chip tiene pines de alimentación separadas para la lógica (**Vcc2**, que debe ser de 5V) y para la alimentación de la carga (**Vcc1**, que puede ser entre 4,5V y 36V). En nuestro caso las dos estarán a 5V. Los circuitos de salida se pueden habilitar en pares por medio de una señal TTL. Los circuitos de manejo de potencia 1 y 2 se habilitan con la señal **1,2EN** y los circuitos 3 y 4 con la señal **3,4EN**.

Las entradas de habilitación permiten controlar con facilidad el circuito, lo que facilita la regulación de velocidad de los motores por medio de un PWM.

Las salidas actúan cuando su correspondiente señal de habilitación está en alto. En estas condiciones, las salidas están activas y su nivel varía en relación con las entradas. Cuando la señal de habilitación del par de circuitos de manejo está en bajo, las salidas están desconectadas y en un estado de alta impedancia.

En la tabla de funcionamiento que sigue se puede observar los niveles TTL que corresponden a cada situación de trabajo:

TABLA DE FUNCIONAMIENTO
(para cada uno de los circuitos)

ENTRADAS †		SALIDA
A	EN	Y
H	H	H
L	H	L
X	L	Z

H = nivel alto L = nivel bajo X = irrelevante
Z = alta impedancia EN = habilitación
† en el modo de corte por protección térmica, las salidas estarán en el estado de alta impedancia, sin que afecte el estado de las entradas.

Figura 38: Tabla funcionamiento

Según cómo se configure tenemos la posibilidad de controlar cuatro motores dos a dos con sentido de giro único, o dos motores pudiendo variar el sentido de giro.

Con cada pin de habilitación (pines 1 y 9) controlamos dos salidas.

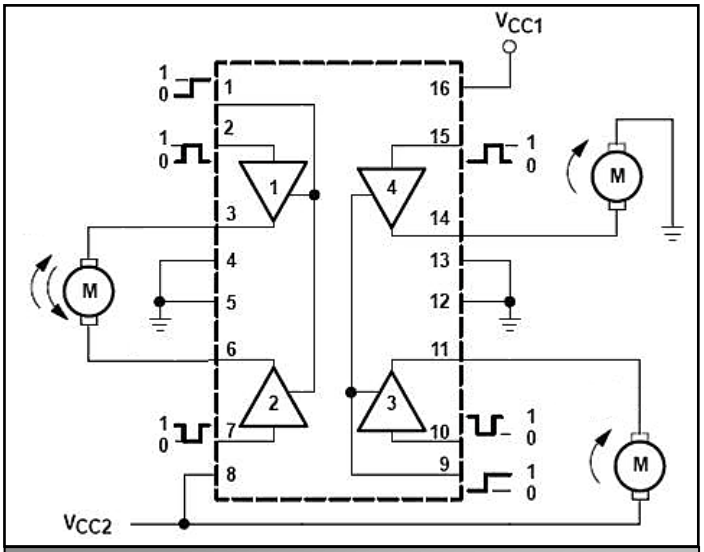


Figura 39: Configuración L293D

La tensión **Vcc1** es para las entradas lógicas TTL, y será de una tensión de 5V. La **Vcc2** normalmente separada del la Vcc1, aunque en nuestro caso será la misma que la Vcc1, puede ir desde 4,5 V hasta 36 V, En nuestro caso será de 5 V, ya que todo el dispositivo estará alimentado a esta tensión, y así la tensión comprendida que le llegará al motor estará entre 0 y 5V.

Por los pines 2, 7, 10 y 15 introduciremos la señal PWM que viene del microcontrolador, y por los pines 3, 6, 11 y 14 obtendremos la señal que mandaremos a una entrada del motor.

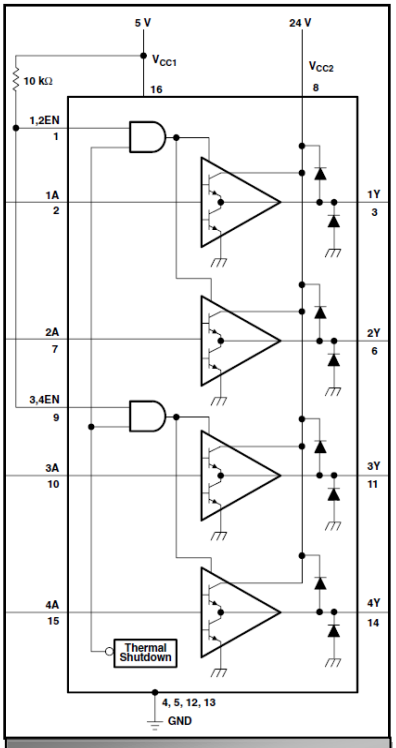


Figura 40: Diagrama interno del L293D

3.4.2. SN74LS125AN

La idea de incluir el encoder en el motor era para poder utilizar los pulsos que nos proporcione e introducírselos al microcontrolador para que éste realice una serie de cálculos modifique el PWM si es necesario y pueda regular la velocidad de giro.

Al realizar las pruebas (que se explicarán en el apartado *4.2. Pruebas del encoder*) se vio como el encoder no proporcionaba una señal clara que el micro pudiera leer, así que se tuvo que incluir un acondicionador de señal. El objetivo de este acondicionador es adaptar la señal de entrada a niveles TTL utilizando para ello la tecnología Trigger-Schmitt o disparador Schmitt.

Después de las pruebas y del diseño de la placa se escogió el **SN74LS125AN**. Este integrado es un bus buffer de 14 pines con 4 salidas. En nuestro caso solamente nos hace falta una entrada y una salida, pero no se ha encontrado un dispositivo con menos de cuatro salidas y 14 pines.

A continuación se va a explicar el integrado y su funcionamiento.

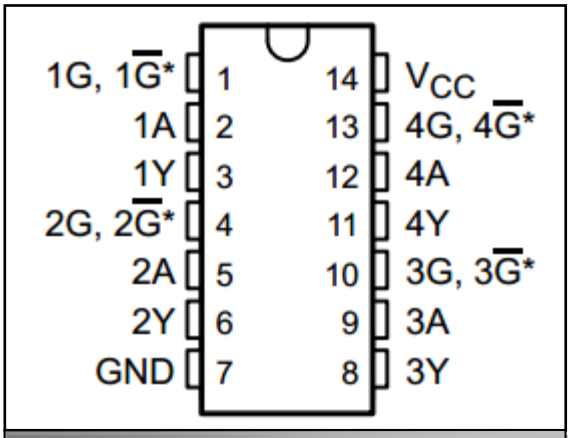


Figura 41: Vista general del 74LS125AN

Este integrado utiliza la tecnología Trigger- Schmitt o también llamada disparador Schmitt cuya tecnología usa la histéresis para encuadrar la señal de entrada, es decir toma ciertos valores de voltaje para un "0" lógico y otros para un "1"; cuando tengamos el nivel de voltaje alrededor de 3.5 v lo tomará como un "1" lógico y a la salida tendremos un "1" lógico; para voltajes en la entrada menores a 0.5v lo tomara como un "0" y a la salida tendremos un "0". Se usa para prevenir los ruidos que podrían dar cambios de estado erróneos y ser malinterpretados por el micro y para adaptar la señal a niveles TTL del microcontrolador.

Imagen descriptiva de la tecnología Trigger-Schmitt:

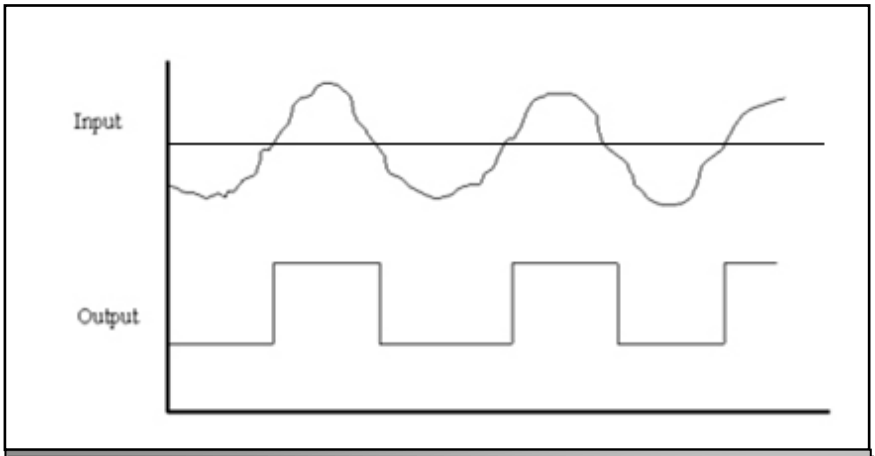


Figura 42: Gráfica funcionamiento Trigger-Schmitt [26]

3.4.3. Regulador LM2940CS-5.0

El sistema de alimentación del dispositivo consta de una pila de 9V y un regulador de tensión para pasar de esos 9 voltios a los 5 que necesitan los integrados para funcionar. El regulador es el **LM2940CS-5.0**. Se ha emulado el circuito de alimentación de la placa PICDEM 2 PLUS, ya que también es capaz de funcionar con una pila de 9V. Observando su hoja de características se puede realizar fácilmente este circuito, que consta de un diodo, dos condensadores y el propio regulador, como se aprecia en la Figura 43.

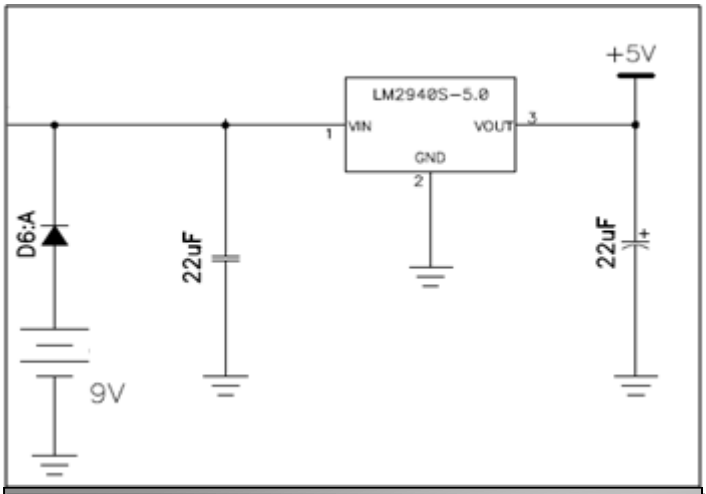


Figura 43: Esquema del regulador LM2940-5.0

Si en la entrada hay 5V o más, en la salida el regulador proporcionará 5V. Si en la entrada hay menos de 5V, en la salida obtendremos la misma tensión de entrada, menos 0,07 voltios, que es lo que cae en el regulador. También el límite de tensión de entrada es de 35 V. En nuestro caso como van a ser 9V no hay ningún problema, pero sí que habrá que incluir un radiador o utilizar un sistema de montaje que permita disipar el calor. Como se aprecia en el esquema, el es conveniente colocar un diodo rectificador después de la alimentación de 9V. En nuestro casi hemos escogido el **1N4007** (Figura 45) porque estaba en el laboratorio, pero uno similar preparado para menos tensión también puede valer sin problema.

También el regulador necesita condensadores a su entrada y a su salida. Estos condensadores son de tipo electrolítico y deben ser de un tamaño concreto, ya que para poder realizar posteriormente la carcasa que va a proteger la placa y otros elementos no debe superar una altura máxima. Los que se han colocado son unos de 220 uF y 50V, de 14mm de altura como el de la Figura 44, de la marca ***Panasonic*** y adquiridos en ***RS Components.***

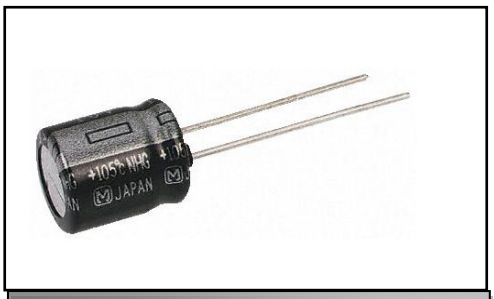


Figura 44: Condensador para el regulador [27]

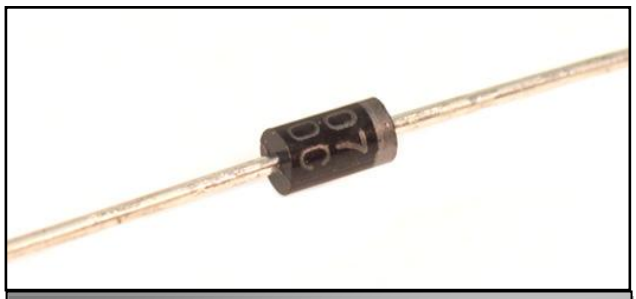


Figura 45: Diodo rectificador 1N4007 [47]

3.4.4. Conectores

Hasta ahora se han explicado componentes que se necesitaban para el funcionamiento básico del dispositivo. En este apartado se van a explicar componentes que añaden al dispositivo funcionalidades extra, por decirlo de alguna manera. Sin ellos el dispositivo cumpliría con los objetivos marcados de la aplicación, pero añadiéndolos, nos proporciona más posibilidades.

3.4.4.1. Conector RJ11

Se ha incluido en el dispositivo una clavija **RJ11** con la finalidad de poder programar le microcontrolador sin tener la necesidad de sacarlo de la placa y llevarlo a la tarjeta PICDEM 2 PLUS. Con el ICD2 y el software MPLAB y conectándolo directamente a esta conexión se puede programar de nuevo y así tener la posibilidad de añadirle alguna función más, depurar el programa o realizar alguna reparación. En concreto se eligió el de la empresa ***Wurth Elektronik***, adquirido a través de ***RS Components***.

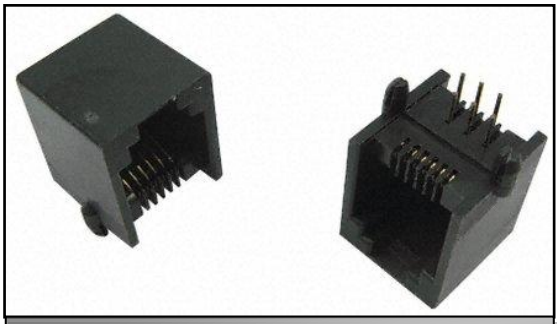


Figura 46: Conector RJ11 [28]

3.4.4.2. Switch de tres posiciones

El dispositivo va a tener tres motores, pero puede que el usuario no desee o no necesite que funcionen todos a la vez. Por esta razón se ha incluido un switch de tres posiciones que permita conectar y desconectar los motores cuando se quiera. En un principio se pensó en colocar el switch de perfil bajo como el de la Figura 47 (pero de tres posiciones), pero por motivos de colocación y forma final del dispositivo, se decidió colocar uno de ángulo recto. Se eligió el **206-3RAST** del fabricante ***CTS Electronic Components*** (Figura 48). El primero se utilizó para realizar las pruebas, y el segundo para incluirlo en la placa.

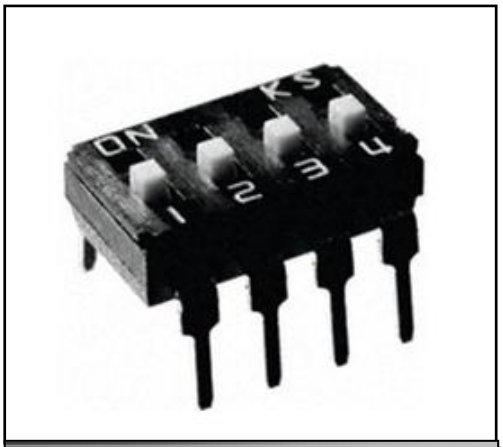


Figura 47: Switch de pruebas [29]

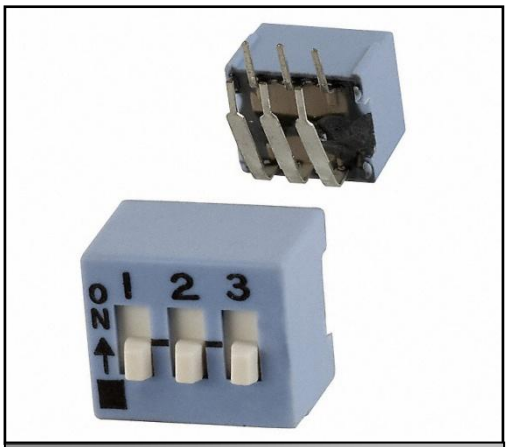


Figura 48: Switch ángulo recto [30]

3.4.4.3. Pulsador

Como se explicará en el apartado de *Programa y registros*, hace falta que el micro reciba interrupciones externas y a partir de ahí actúe. Para generar esta interrupción se va a utilizar un pulsador. En principio se pensó en uno similar al de la placa PICDEM 2 PLUS, pero por el diseño del dispositivo se decidió incluir uno de ángulo recto. En concreto se eligió el MJTP1236A (Figura 49) de la empresa ***Apem***. Una vez finalizado el diseño del dispositivo y por motivos de mejor manejo del mismo, se sustituyó por el **MJTP1236B** (Figura 50), que tiene el pulsador algo más largo y facilita la activación en el tamaño tan reducido del que disponemos.

Figura 49: Pulsador provisional [31]

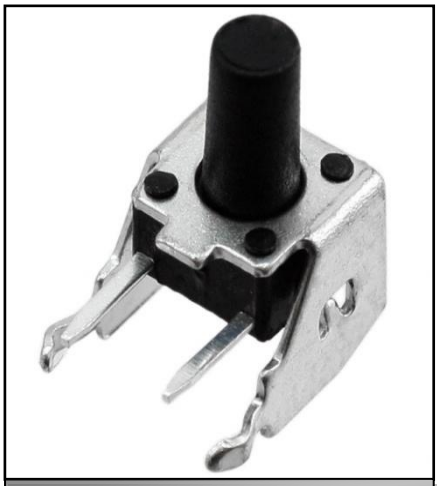


Figura 50: Pulsador definitivo [32]

3.4.4.4. Interruptor ON/OFF

Para elegir el interruptor de encendido primero se pensó en uno de tipo rocker, similar al de la Figura 51, pero se acabó descartando por problemas de espacio y por modificaciones en el diseño del dispositivo.

Figura 51: Interruptor tipo rocker [33]

Buscando en catálogos de diferentes distribuidores y sobre todo atendiendo al diseño del dispositivo (forma y tamaño), se decidió intentar adquirir un interruptor deslizante de pequeño tamaño. Finalmente se adquirió el **25339NA** de la empresa ***Apem*** (Figura 52).

Figura 52: Interruptor de encendido definitivo [34]

4. Pruebas realizadas

En este capítulo se van a explicar las diferentes pruebas que se han realizado para construir el dispositivo. A raíz de hacer estas pruebas se han incluido elementos que no estaban pensados introducir en el diseño inicial, se han quitado algunos que parecían fijos y se ha ido incluyendo mejoras dentro del mismo dispositivo.

4.1. Prueba del PWM

Para controlar la velocidad de giro del motor se decidió que un buen sistema era el de utilizar el PWM que puede generar el microcontrolador y mandar esta señal al motor.

Como se ha explicado en apartados anteriores, el PWM (*Pulse Width Modulation)* o modulación de anchura de impulsos es un sistema en el que se modifica el ciclo de trabajo de la señal que le introducimos al motor. De esta manera cuanto mayor es el ciclo de trabajo más tensión le llegará al motor, ya que lo que ve el motor es una tensión media y girará más rápido. Si queremos que el motor gire más lento hay que reducir el ciclo de trabajo, y como consecuencia se reducirá la tensión media.

Vamos a explicar qué registros hay que modificar en el microcontrolador para poder generar correctamente el PWM [49]. Para ello se ha realizado un programa de prueba.

Primero se estudió detenidamente la hoja de especificaciones del micro para ver qué registro es el que hace referencia al PWM. El registro es el ***CCP1CON***, y para seleccionar el modo PWM los bits 2 y 3 del registro deben estar puestos a uno (***CCP1M<3:2>***). Para seleccionar la frecuencia del PWM hay que darle un valor a ***PR2*** y para establecer el ciclo de trabajo deseado hay que darle valor a ***CCPR1L***.

REGISTER 9-1: CCP1CON: CAPTURE/COMPARE/PWM CONTROL REGISTER 1 (ADDRESS 17h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **CCP1X:CCP1Y:** PWM Least Significant bits

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPR1L.

bit 3-0 **CCP1M<3:0>:** CCP1 Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCP1 module)
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, set output on match (CCP1IF bit is set)
1001 = Compare mode, clear output on match (CCP1IF bit is set)
1010 = Compare mode, generate software interrupt on match (CCP1IF bit is set, CCP1 pin is unaffected)
1011 = Compare mode, trigger special event (CCP1IF bit is set, CCP1 pin is unaffected); CCP1 resets TMR1 and starts an A/D conversion (if A/D module is enabled)
11xx = PWM mode

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Figura 53: Registro CCP1CON

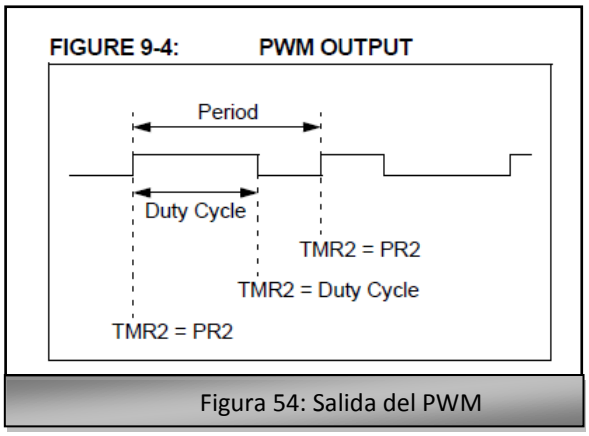


Figura 54: Salida del PWM

Para seleccionar la frecuencia del PWM hay que variar el valor de ***PR2***. Pero, ¿cómo saber qué valor poner y a qué frecuencia corresponde? Es muy sencillo con la ayuda de la siguiente fórmula:

EQUATION 9-1:

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot TOSC \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period].

Figura 55: Ecuación cálculo del periodo del PWM

Se puede cambiar la frecuencia del PWM con esta sencilla fórmula donde ***"Tosc"*** es el periodo de oscilación del oscilador interno, en nuestro caso como la frecuencia que vamos a poner es de 8MHz, el Tosc será 1/8MHz. ***"PWM Period"*** es el periodo que deseamos para el PWM, y ***"TMR2 Prescale Value"*** es el valor de prescaler de ***TIMER2*** (1:1, 1:4 ó 1:16). Esto nos permitirá hacer el PWM con la frecuencia que deseemos.

Por ejemplo si queremos una frecuencia para el PWM de 8KHz teniendo ***Tosc***=8MHz y un prescaler de 1:1 tendríamos que poner:

$$[1/8KHz]/[4*(1/8MHz)*1]=PR2+1$$

Haciendo esta operación nos sale un PR2+1=250. Por lo tanto hay que poner el **PR2** con un valor de **249** en hexadecimal, que es **F9**.

Así que el ciclo de trabajo establecido en ***CCPR1L*** tiene que ir comprendido entre 0 y 249 (en hexadecimal).

Para establecer la frecuencia del oscilador interno hay que configurar el registro ***OSCCON***.

REGISTER 4-2: OSCCON: OSCILLATOR CONTROL REGISTER (ADDRESS 8Fh)

U-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0
—	IRCF2	IRCF1	IRCF0	OSTS(1)	IOFS	SCS1	SCS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6-4 **IRCF<2:0>:** Internal RC Oscillator Frequency Select bits
000 = 31.25 kHz
001 = 125 kHz
010 = 250 kHz
011 = 500 kHz
100 = 1 MHz
101 = 2 MHz
110 = 4 MHz
111 = 8 MHz

bit 3 **OSTS:** Oscillator Start-up Time-out Status bit(1)
1 = Device is running from the primary system clock
0 = Device is running from T1OSC or INTRC as a secondary system clock

Note 1: Bit resets to '0' with Two-Speed Start-up mode and LP, XT or HS selected as the oscillator mode.

bit 2 **IOFS:** INTOSC Frequency Stable bit
1 = Frequency is stable
0 = Frequency is not stable

bit 1-0 **SCS<1:0>:** Oscillator Mode Select bits
00 = Oscillator mode defined by FOSC<2:0>
01 = T1OSC is used for system clock
10 = Internal RC is used for system clock
11 = Reserved

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Figura 56: Registro OSCCON

En nuestro caso vamos a poner el oscilador para que funcione a 8MHz. Tenemos que configurar los bit 6 al 4, poniéndolos todos a uno. También se configuran los bits 0 y 1 de manera que seleccionamos *"T1OSC usado como sistema de reloj"*, es decir poniendo un "01" en el ***SCS<1:0>***. Los demás bits en principio no hacen falta configurarlos.

Otro apartado a modificar es el **TIMER2**, que también afecta en la fórmula del periodo del PWM. Para configurar el prescaler del PWM con el **TIMER2**, hay que configurar el bit 0 y bit 1 del registro poniéndolos a cero los dos, porque es un prescaler de 1:1, y para activarlo hay que poner el bit 2 a uno. Los bits 3 a 7 no se configuran.

REGISTER 8-1: T2CON: TIMER2 CONTROL REGISTER (ADDRESS 12h)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6-3 **TOUTPS<3:0>:** Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
0010 = 1:3 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON:** Timer2 On bit
1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Figura 57: Registro del Timer2

Otro apartado a configurar es por qué pin queremos que salga la señal de PWM, a elegir entre **RB0** y **RB3**. En nuestro caso seleccionamos la salida **RB3**. Esto lo hacemos en la línea de configuración inicial poniendo ***"CCPMX_RB3"*** y lo hacemos porque se va a utilizar el pin **RB0** para introducir una interrupción externa.

La elección de la frecuencia del PWM de 8kHz no se ha hecho a la ligera. Para escoger esta frecuencia hay que tener presente una serie de aspectos:

1) A mayor frecuencia, mayor ruido en el sistema.

2) La frecuencia que es capaz de oír el ser humano está entre 20Hz y 20kHz, así que frecuencias en este rango puede provocar un ruido molesto, y depende de qué aplicaciones no debe oírse.

3) Frecuencias más altas permiten regular velocidades más bajas.

La frecuencia adecuada para cada caso depende del motor utilizado y de otros elementos del circuito, y en casi todos los casos, a menos que indique el fabricante del motor esa frecuencia, hay que realizar pruebas para ver cuál es la mejor elección.

Se han hecho una serie de pruebas con diferentes frecuencias para elegir la mejor para el motor. Se han escogido las frecuencias de 2kHz, 4kHz, 8kHz, 10kHz, 13kHz, 15kHz, 18kHz, 20kHz y 25kHz. Primero se ha comprobado si con un ciclo de trabajo del 10% el motor arrancaba, y no se ha dado el caso con ninguna frecuencia [53]. Si hubiera arrancado con

alguna de las frecuencias habría sido una señal de que esa es la frecuencia adecuada para el motor. Luego se ha probado con cada frecuencia y los diferentes ciclos de trabajo de los que va disponer el dispositivo para ver si producían ruido y cómo respondían a las diferentes variaciones de velocidad, con estos resultados:

- 2kHz → Dos primeras velocidades muy audibles, pero buena respuesta.
- 4kHz → Primera velocidad muy audible, pero buena respuesta.
- 8kHz → Prácticamente inaudible y buena respuesta.
- 10kHz → Algo audible, más que en el caso de 8kHz y buena respuesta.
- 13kHz → Dos primeras velocidades muy audibles.
- 15kHz → Dos primeras velocidades muy audibles.
- 18kHz → No audible pero mala respuesta, a altas velocidades.
- 20kHz → No audible pero mala respuesta a altas velocidades.
- 25kHz → No audible pero mala respuesta a altas velocidades.

A la vista de los resultados se ha escogido la frecuencia de 8kHz porque tiene buena respuesta y, aunque produce algo de ruido, no es muy apreciable, y además el dispositivo va a estar situado dentro de una máquina y el posible ruido que se pueda generar no va a ser audible. Evidentemente, es mejor que se produzca algo de ruido y funcione correctamente, a que sea muy silencioso pero sea inestable.

4.1.1. Montaje provisional

Una vez configurado los parámetros del PWM sólo nos queda poner diferentes ciclos de trabajo y hacer retardos de unos pocos segundos para ver si reacciona bien el motor. Para ello se ha utilizado el siguiente código presentado en la Figura 58.

```
//*Prueba de PWM*//
#include<htc.h>                                  //Incluimos librería del micro a usar
__CONFIG(WRT_OFF & WDTE_OFF & PWRTE_OFF & FOSC_INTOSCIO & LVP_OFF & CCPMX_RB3);
#define _XTAL_FREQ 8000000                       //Oscilador Interno de 8MHZ
void main(void){

TRISB=0x01;                                      //Puerto B configurado como salida
TRISA=0x10;                                      //Puerto RA4 como entrada
int i;

//Comparadores desactivados
 //CMCON=0x07;
    GIE=0;                                       //interrupciones globales desactivadas
    PEIE=0;                                      //Desactiva interrupciones por periféricos
    ANSEL=0;                                     //Todas las entradas digitales
    SCS1=0;                                      //T1OSC usado como sistema de reloj
    SCS0=1;
    IRCF2=1;                                     //Configura oscilador interno a 8MHz
    IRCF1=1;
    IRCF0=1;
    CCP1M3=1;                                    //Configurar el pic en modo PWM
    CCP1M2=1;
    CCP1M1=0;
    CCP1M0=0;
                                        //Poner el puerto B a cero
    T2CONbits.T2CKPS1=0;                         //Prescaler 1:4
    T2CONbits.T2CKPS0=1;
    T2CONbits.TMR2ON=1;                          //Activamos Timer2
    PORTBbits.RB1=1;

                if(RA4==0){                      //Condición de pulsar RA4
                    PR2=0x7C;                    //Periodo de trabajo a 4KHz
                    //CCPR1L=0x4A;               //Ciclo de trabajo del 60%

                    //__delay_ms(8000);
                    //CCPR1L=0x3E;               //Ciclo de trabajo del 50%

                    //__delay_ms(8000);
                    //CCPR1L=0x32;               //Ciclo de trabajo del 40%

                    //__delay_ms(8000);
                    //CCPR1L=0x25;               //Ciclo de trabajo del 30%

                    //__delay_ms(8000);
                    CCPR1L=0x19;                 //Ciclo de trabajo del 20%

                }
                    else if(RB0==0)
```

Figura 58: Programa de prueba del PWM

La primera opción es conectar la salida RB3 del micro a un terminal del motor y el otro a GND.

Primero se conecta el osciloscopio al terminal del micro y se observa que realiza bien el PWM y los porcentajes del ciclo de trabajo que se han asignado se corresponden con lo que aparece en el osciloscopio. La Figura 59 hace es con un ciclo de trabajo del 30% y la Figura 60 con un ciclo de trabajo del 50%:

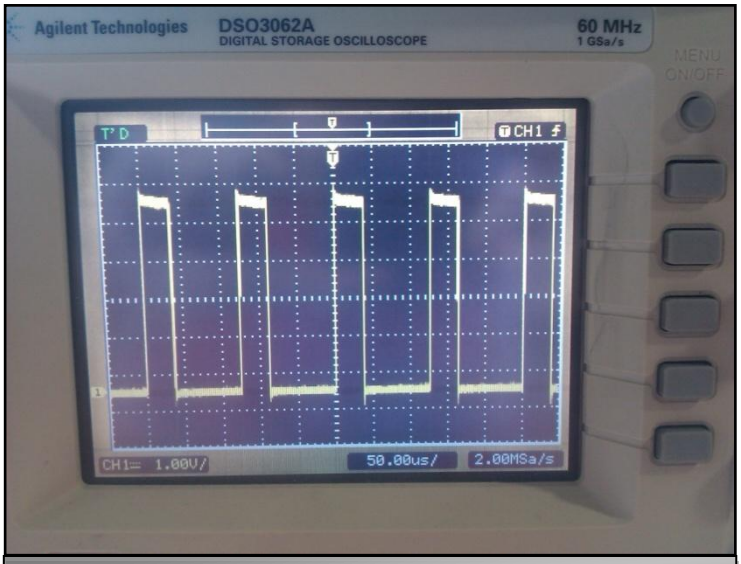


Figura 59: Imagen pulsos del PWM (1)

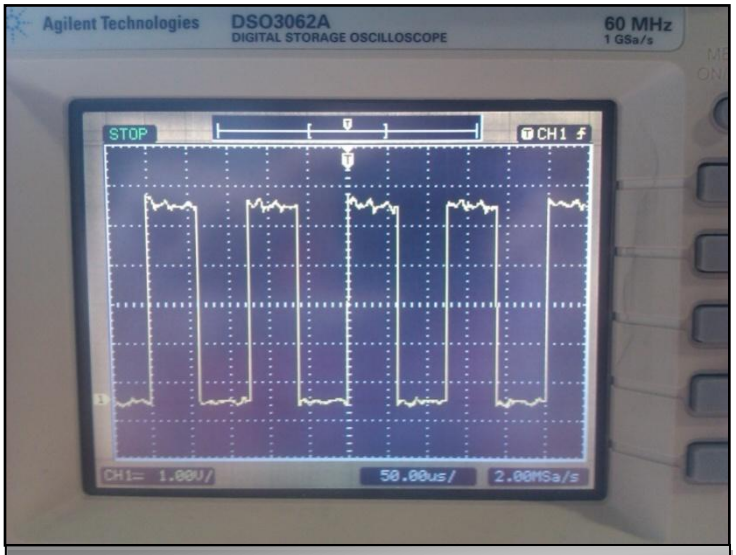


Figura 60: Imagen pulsos del PWM (2)

Pero, al conectar el terminal a la entrada del motor, comprobamos que el motor no gira. Esto puede ser debido a que como las salidas del micro dan alrededor de **20-25mA**, esta corriente no sea suficiente para mover el motor aunque pongamos el ciclo de trabajo cerca del 100%. Hay que buscar otra forma de controlar el motor con PWM.

Lo podemos conseguir añadiendo algún elemento entre la salida del micro y la entrada del motor. Se va a utilizar el esquema de la Figura 61.

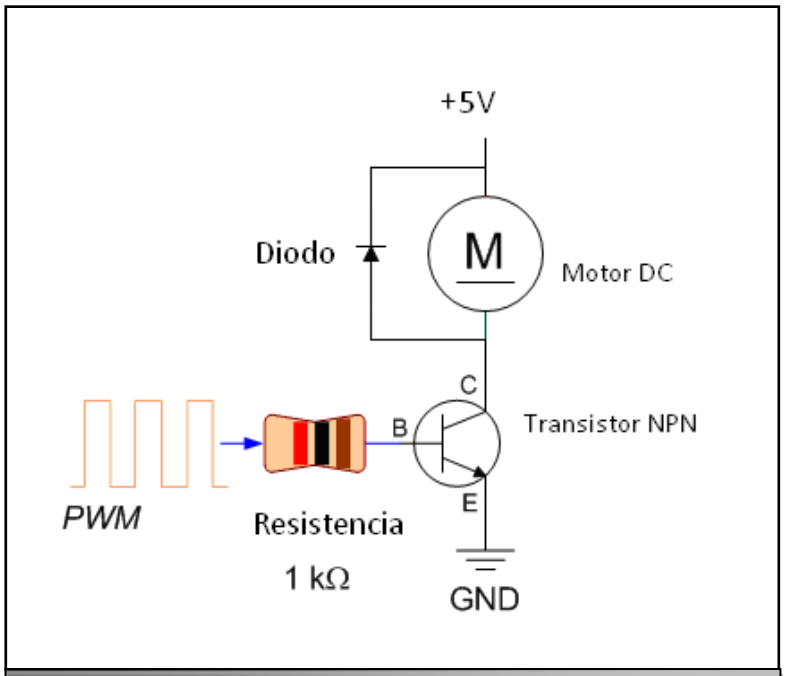


Figura 61: Esquema provisional para el PWM [35]

El funcionamiento del circuito es sencillo [35], variando el ciclo de trabajo del PWM controlamos que le llegue más o menos corriente a la base del transistor, y éste conducirá más o menos y hará que le llegue más o menos tensión al motor y por tanto gire a más o menos revoluciones. El diodo se coloca como elemento de protección para los picos de corriente que se pueden dar al arrancar el motor.

Con este circuito provisional el motor funciona correctamente y va aumentando de velocidad según se lo ordena el micro. El proceso de diseño del PWM ha terminado.

Este circuito se ha realizado para comprobar el funcionamiento del PWM del micro, pero no va a ser el circuito final. Para el proyecto vamos a usar un driver para poder mandar la señal del micro a varios motores y así poder realizar el tratamiento a varias fibras a la vez. Para ello vamos a utilizar el driver **L293D**.

4.1.2. Montaje con el driver L293D

En el apartado ***3.4.1. Driver L293D*** se ha explicado más detalladamente cómo funciona este driver y las posibilidades que tiene. Ahora sólo nos interesa que le introducimos una señal PWM y el driver lo aumenta según unos valores de tensión de referencia que se le introducen.

La introducción de este driver en el circuito es para poder incluir varios motores, así con una sola señal PWM se pueden añadir hasta cuatro motores diferentes. De otra manera habría que reproducir el circuito del montaje provisional tantas veces como motores se vayan a colocar. Como nuestro dispositivo no va a superar la cantidad de cuatro motores porque no disponemos de espacio suficiente, con un solo driver nos vale.

Antes de pedir el componente y realizar pruebas reales se pensó que era buena idea utilizar un simulador para tener más seguridad de si va a funcionar correctamente o no. Para ello se utilizó el programa **Proteus**.

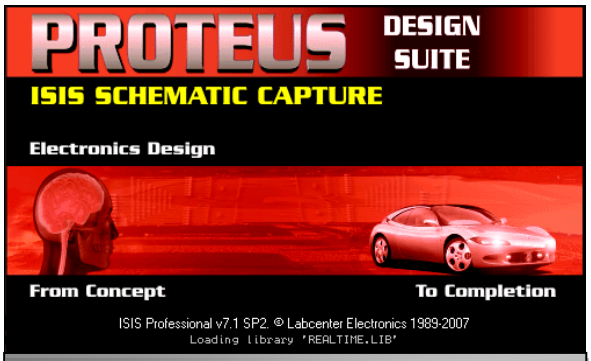


Figura 62: Logo de Proteus [36]

Con este software se puede incluir el código del programa y asociarlo al micro y así saber si funciona de manera adecuada [43]. El esquema utilizado es el siguiente:

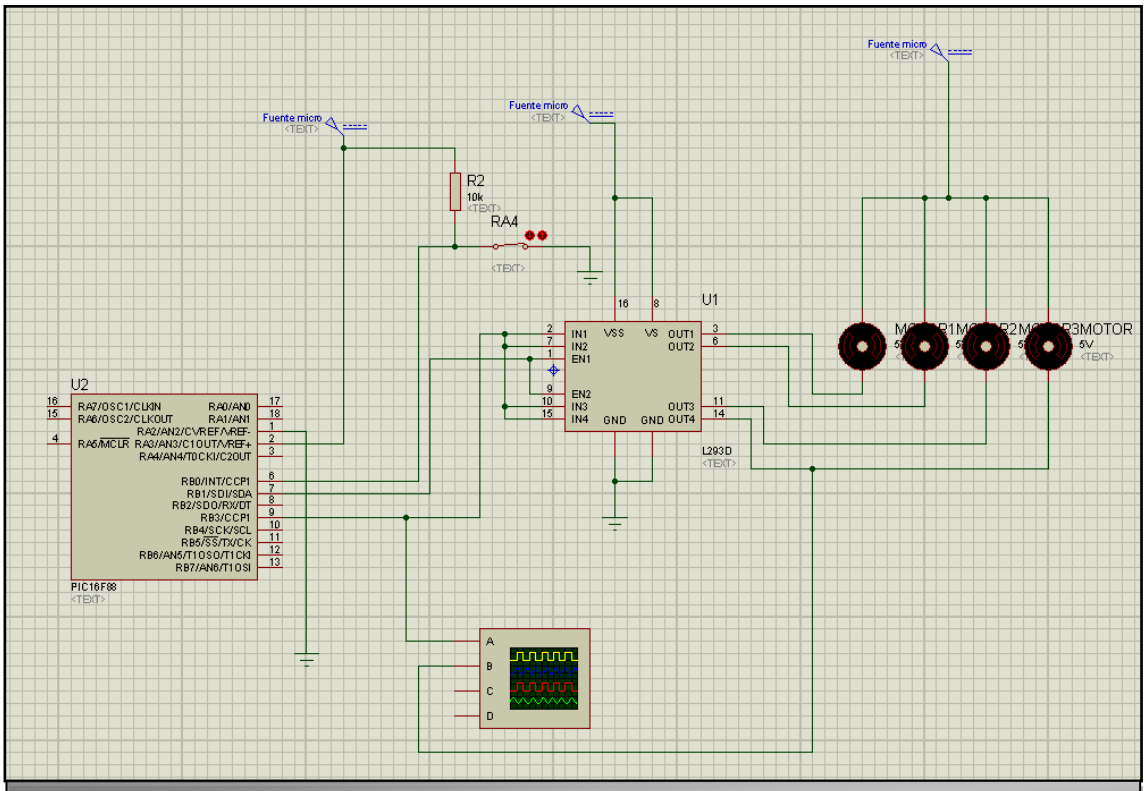


Figura 63: Esquema para la simulación del driver L293D en Proteus

Una vez que tenemos el esquema, hay que asociar al micro el código que queremos que ejecute. Para ello debemos hacer doble clic en el símbolo del PIC y aparecerá la siguiente pestaña:

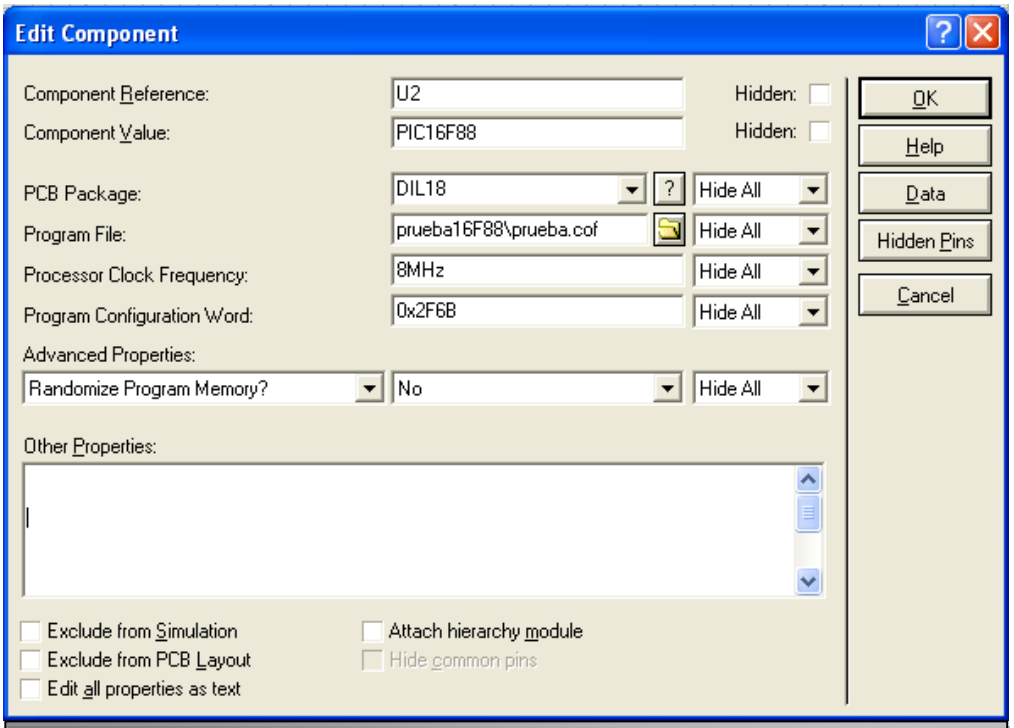


Figura 64: Ventana de edición del PIC en Proteus

Nos dirigimos a la pestaña de ***Program File*** y pinchamos en el símbolo de carpeta que aparece y se despliega una ventana. En esta ventana hay que buscar la ruta de nuestro programa y seleccionar el archivo .**cof** o **.hex** asociado a nuestro código. En nuestro caso elegimos el **.cof**. En esta misma ventana también podemos modificar la frecuencia del reloj a la que trabaja el micro.

Una vez asociado el código ya podemos simularlo para ver cómo responde. Para simular nos tenemos que dirigir a la parte de abajo a la izquierda de la pantalla, donde nos encontramos símbolos como el play, stop y pause típicos.

Figura 65: Controles de simulación de Proteus

Pinchamos en play y vemos, según vamos activando y desactivando el interruptor, que la pantalla del osciloscopio va variando el ciclo de trabajo, mientras a la vez los motores van girando (Figura 66).

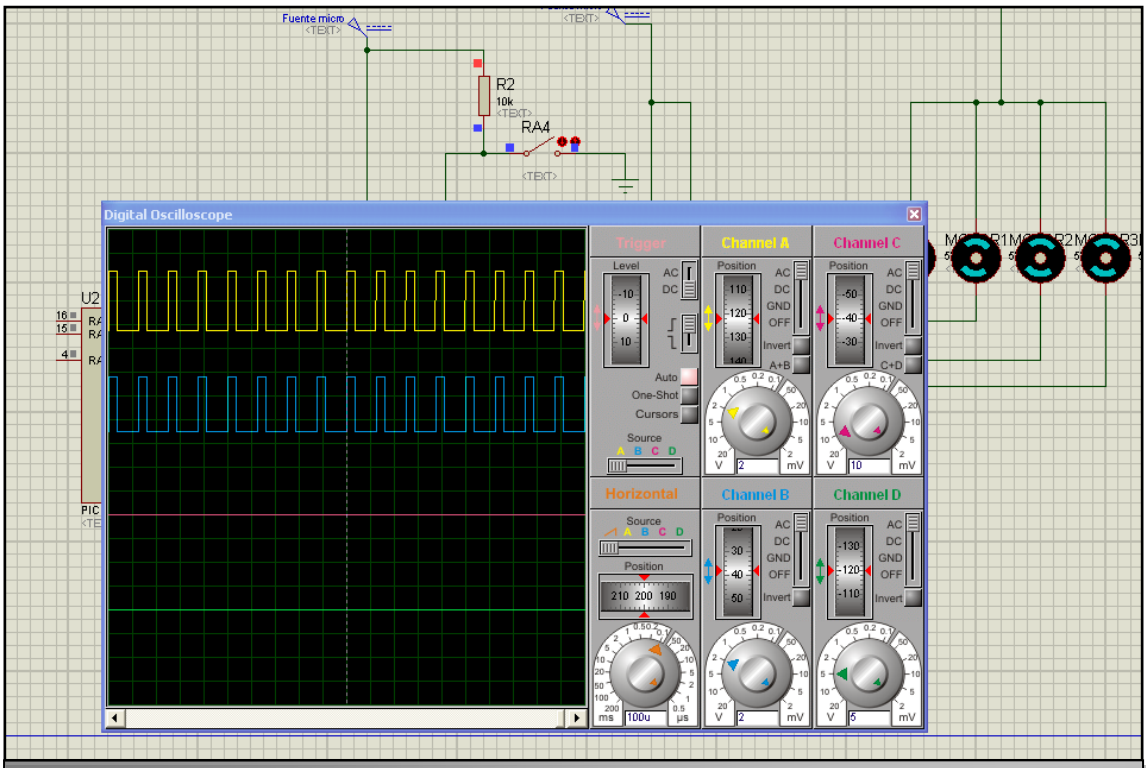


Figura 66: Simulación del programa en Proteus

Hay que decir que para que los cambios del estado del interruptor surtan efecto, hay que pausar y arrancar la simulación cada vez que se haga un cambio en el interruptor, es decir, si queremos pasar de interruptor cerrado a abierto, hay que primero abrir el interruptor, luego pausar la simulación, y acto seguido reanudar la simulación.

Como en el osciloscopio aparecen correctamente los cambios de ciclo de trabajo y los motores giran podemos afirmar que el driver funciona correctamente y que puede ser una buena opción para incluir varios motores en el dispositivo.

De esta manera hemos ahorrado tiempo evitándonos todo el montaje externo, y dinero también ya que si se llega a pedir y a montar en la placa y después no funciona, se habría malgastado ese dinero.

Una vez simulado se procede a pedir el componente y a realizar el montaje en la placa de pruebas. Se conecta el osciloscopio y se aprecia que la anchura del ciclo de trabajo varía según le previsto y que los motores funcionan correctamente.

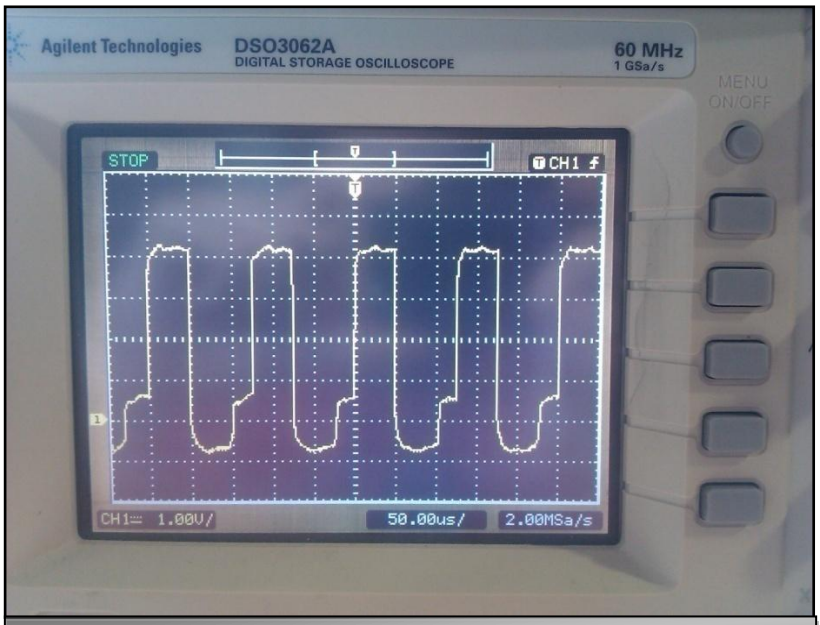


Figura 67: Pulsos a la salida del driver

4.2.Pruebas del encoder

El objetivo de incluir un encoder en el circuito viene por la intención de poder controlar la velocidad del motor. La idea es mandar la señal del encoder que lleva incorporado el motor al microcontrolador, que el programa detecte los pulsos y pueda calcular el periodo de los pulsos, lo compare con unos valores de referencia y que modifique el ciclo de trabajo en función de ese periodo.

4.2.1. Señal inicial del encoder

Para conseguir esto primero hay que ver el tipo de señal que manda el encoder y ver si el microcontrolador es capaz de procesarla. Es bastante probable que esta señal no sea legible, y haya que incluir algún sistema que la adapte a las condiciones del microcontrolador.

Para realizar la prueba ya se está utilizando el driver L293D y también las condiciones del PWM explicadas anteriormente.

Se introducen los pulsos del PWM, el motor gira y se visualiza la señal del encoder (Figura 68).

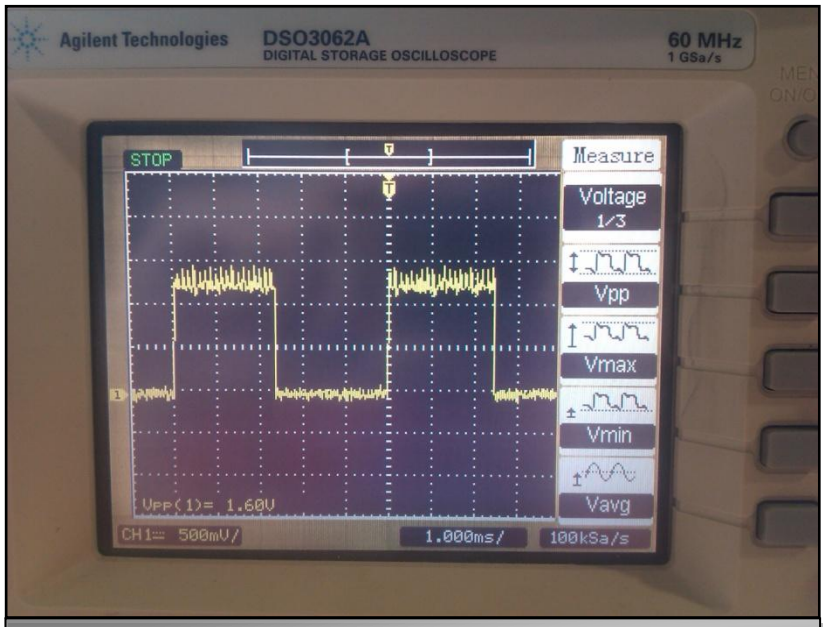


Figura 68: Pulsos del encoder sin tratar

Como se puede apreciar la señal tiene mucho ruido y una tensión máxima de 1,6 Voltios, insuficiente, ya que el micro necesita valores TTL, así que hay que adaptarla.

4.2.2. Inclusión del acondicionador de señal

Para ello se va a utilizar un acondicionador de señal de tecnología TTL, que utiliza la tecnología de Trigger Schmitt o disparador Schmitt para adaptar la señal.

En un principio se pensó en utilizar el 74LS540, ya que se había visto en algún circuito parecido, pero en el laboratorio sólo se encontraba el 74LS244N, que es muy similar, así que se empezó a realizar las pruebas con este último. Para poder usarlo hay que configurar el pin 1, ya que este pin habilita las cuatro primeras entradas, y como tiene un inversor en su entrada es necesario ponerlo a cero. Como es lógico hay que alimentarlo por el pin 20 y llevarlo a masa con la pin 10. Ahora sólo queda introducir la señal del encoder al pin 2 y visualizar la señal de salida en el pin 18.

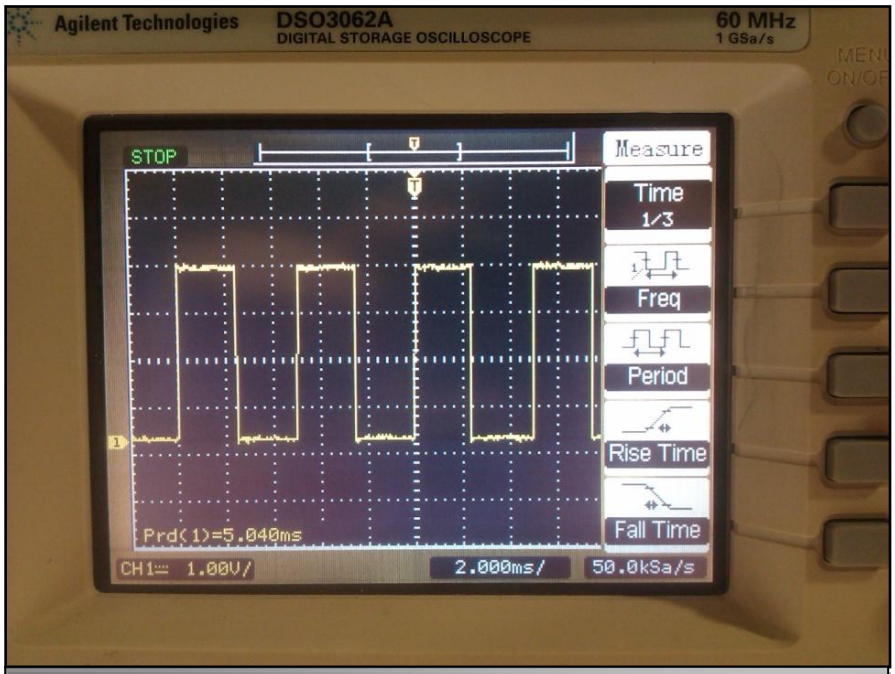


Figura 69: Pulsos encoder con el 244N a 30% del ciclo de trabajo

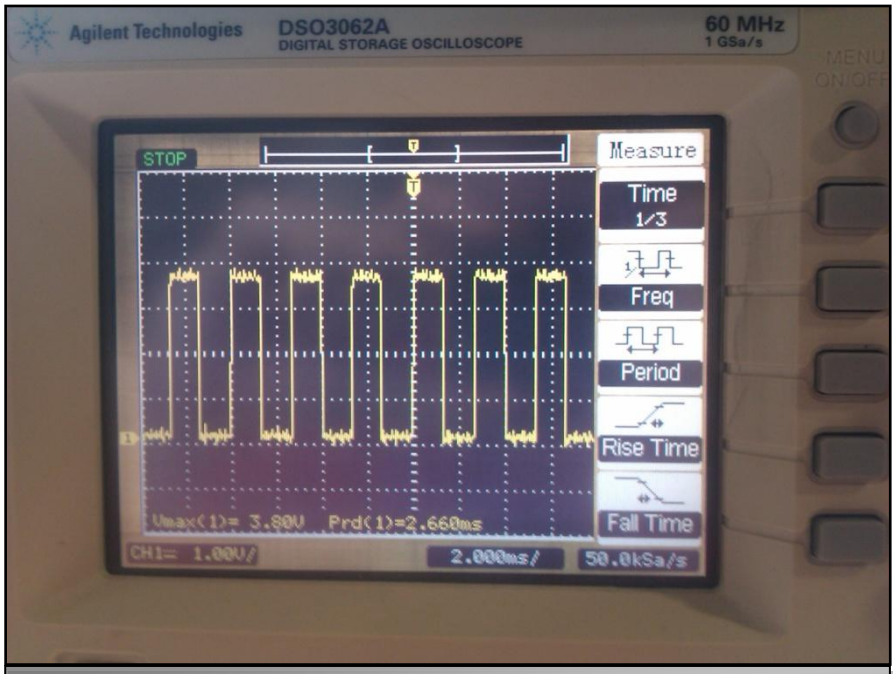


Figura 70: Pulsos encoder con el 244N a 70% del ciclo de trabajo

Como se aprecia la señal es limpia y está en niveles TTL, en principio esta señal sí es legible por el microcontrolador y ya se pueden empezar a realizar pruebas de ciclos de trabajo y de periodos de esta señal.

Una vez conseguido el 74LS540 se realizó el mismo procedimiento que con el 74LS244N, y se comprobó que este último daba una señal más limpia y estable. Así que se iba a sustituir el LS540 por el LS244N.

Posteriormente, a medida que se iba avanzando con el diseño de la placa PCB se vio que este integrado tiene veinte pines, y que solamente se utilizan cuatro de ellos. Esto supone una pérdida de espacio para diseñar la placa. Así que se buscó un integrado que cumpliera con la misma función pero que tuviera menos pines. Se encontró el 74LS125A, que es de 14 pines. Se realizaron las mismas pruebas y se comprobó que el resultado era el mismo que en el caso de 74LS244N, por lo tanto el acondicionador de señal definitivo sería el ***74LS125A.***

4.2.3. Pruebas ciclo de trabajo/periodo

Una vez que la señal del encoder es limpia y en niveles que entiende le PIC, se van a realizar una serie de pruebas para ver la relación que tiene el ciclo de trabajo del PWM con el periodo que se obtiene en el encoder. Esto se realiza para poder establecer una serie de valores con los que el micro pueda comparar, para luego modificar el ciclo de trabajo si fuera necesario. Para ello se ha realizado una tabla y una gráfica donde se aprecia cómo varían estos dos valores.

Los datos finales se han obtenido con la configuración del driver y el 74LS125A.

Ciclo de trabajo	Periodo (valor medio)(ms)	Periodo (valor mín.) (ms)	Periodo (valor máx.) (ms)
30%	8,1	7,86	8,34
40%	5,2	5,06	5,34
50%	3,88	3,76	4,01
60%	3,11	3,01	3,21
70%	2,54	2,52	2,56

Figura 71: Tabla de ciclos de trabajo sus periodos

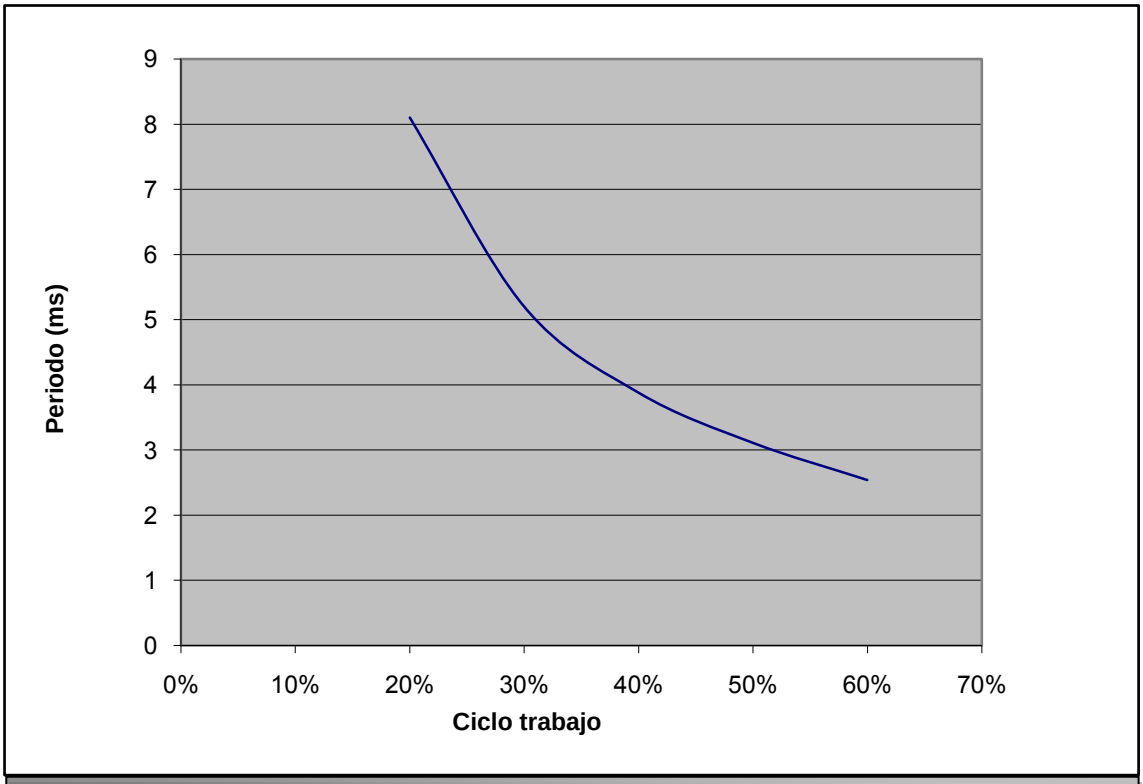


Figura 72: Gráfica Ciclo de trabajo/Periodo

Una vez obtenidos los datos de los periodos a los que gira el motor, se diseñó un programa para hacer lo siguiente. El micro reciba los pulsos, los cuente, haga una serie de operaciones para saber el periodo y lo compare con un array que contenga los valores de la tabla, y en función de si es mayor o menor a estos valores, que aumente o disminuya el ciclo de trabajo.

Para conseguir esto se han utilizado los otros dos Timer que tiene disponible el micro, el **Timer0** y el **Timer1**, que no se están utilizando[50].

El **Timer0** se va a utilizar para contar los pulsos que vienen del encoder, y cuando se desborde se producirá una interrupción. En esa interrupción se utilizará el **Timer1** para ver el tiempo que ha pasado desde que ha llegado el primer pulso hasta que se ha desbordado el **Timer0**, y con ese valor de **Timer1**, dividirlo entre el número de pulsos que ha contado el **Timer0** y así obtener el periodo. Una vez obtenido el periodo, se compara con el establecido en el array y, si es menor, quiere decir que el motor gira más rápido de lo que debería, y entonces bajaría el ciclo de trabajo; si el periodo es mayor, el motor está girando más despacio y entonces habría que aumentar el ciclo de trabajo.

Para realizar esto hay que configurar los Timer de una determinada manera:

-Timer0-

Para configurar el Timer0 hay que ir a los registros ***OPTION_REG*** (Figura 73) e ***INTCON*** (Figura 74) y modificar el valor de sus bits.

REGISTER 6-1: OPTION_REG: OPTION CONTROL REGISTER (ADDRESS 81h, 181h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$:** PORTB Pull-up Enable bit
bit 6 **INTEDG:** Interrupt Edge Select bit
bit 5 **T0CS**: TMR0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKO)
bit 4 **T0SE**: TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin
bit 3 **PSA**: Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module
bit 2-0 **PS<2:0>:** Prescaler Rate Select bits

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

Legend:
R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
-n = Value at POR '1' = Bit is set '0' = Bit is cleared x = Bit is unknown

Note: To avoid an unintended device Reset, the instruction sequence shown in the *"PICmicro® Mid-Range MCU Family Reference Manual"* (DS33023) must be executed when changing the prescaler assignment from Timer0 to the WDT. This sequence must be followed even if the WDT is disabled.

Figura 73: Registro OPTION_REG (TImer0)

Los bits 6 y 7 no se configuran. El bit 5 (T0CS) se pone a uno, ya que queremos que cuente a través de la entrada T0CKI, que es el pin 3 del micro. El bit 4 (T0SE) es para elegir entre que cuente por flanco de subida o de bajada, en este caso es indiferente uno u otro, así que elegimos por flanco de subida.

EL bit 3 (PSA) es para elegir qué prescaler queremos asignarle al Timer, a elegir entre el del propio Timer0 o el del WDT (WatchDog Timer). Elegimos el WDT porque queremos un prescaler de 1:1 y solo es posible con el WDT. En los bits <2:0> los ponemos a cero para elegir este prescaler de 1:1.

Otro registro a configurar es el ***INTCON***, que hace referencia a las interrupciones. La que afecta al Timer0 es la ***TMR0IE*** (bit 5), que es la que habilita la interrupción por este Timer. Se pone a uno para habilitarla.

También hay que inicializar el valor del **Timer0** [50]. La idea es inicializarlo a un valor para que primero cuente una serie de pulsos y cuando se desborde, darle el valor 216 para que solamente cuente 40 pulsos. Inicialmente se le da el valor 100.

REGISTER 2-3: INTCON: INTERRUPT CONTROL REGISTER (ADDRESS 0Bh, 8Bh, 10Bh, 18Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
bit 7							bit 0

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **PEIE:** Peripheral Interrupt Enable bit
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INT0IE:** RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit
A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition and allow flag bit RBIF to be cleared.
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

Figura 74: Registro INTCON (Timer0)

-Timer1-

Para configurar el Timer1 hay que ir al registro ***T1CON*** (Figura 75) y modificar sus bits.

REGISTER 7-1: T1CON: TIMER1 CONTROL REGISTER (ADDRESS 10h)

U-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6 **T1RUN**: Timer1 System Clock Status bit
1 = System clock is derived from Timer1 oscillator
0 = System clock is derived from another source

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable Control bit
1 = Oscillator is enabled
0 = Oscillator is shut off (the oscillator inverter is turned off to eliminate power drain)

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Control bit
TMR1CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
TMR1CS = 0:
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS:** Timer1 Clock Source Select bit
1 = External clock from pin RB6/AN5(1)/PGC/T1OSO/T1CKI (on the rising edge)
0 = Internal clock (Fosc/4)
Note 1: Available on PIC16F88 devices only.

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

Figura 75: Registro T1CON (TImer1)

El bit 7 no se configura. El bit 6 lo ponemos a cero. Los bit 5 y 4 les asignamos les damos valor uno para establecer un prescaler de 1:8. De esta manera y con el ciclo de reloj de 8MHz podemos contar hasta un tiempo máximo de 0,26 segundos. Este dato se obtiene de la siguiente operación:

$$Tiempo = 4*1/8MHz*Prescaler*65535$$

El valor de 65535 es el máximo que puede contar el timer1 ya que es un registro de 16 bits. Y el prescaler es de 1:8.

El bit 3 se pone a cero para deshabilitar el oscilador del Timer1, ya que vamos a utilizar el oscilador interno del micro, y de esta manera eliminamos consumo innecesario. El bit 1 es para seleccionar precisamente esto, que cuente a través de del oscilador interno. Nos hemos saltado el bit 2 porque configurando el bit 1 con cero, el micro ignora lo que haya en el bit 2.

El bit 0 es para iniciar y parar el Timer1. También hay que inicializarlo a un valor, en este caso el Timer1 valdrá inicialmente cero.

Después de configurar los dos timer hay que escribir el programa que permita realizar la lectura del encoder.

```
//*Prueba pulsador*//
#include<htc.h>                          //Incluimos librería del micro a usar
__CONFIG(WRT_OFF & WDTE_OFF & PWRTE_OFF & FOSC_INTOSCIO & LVP_OFF & CCPMX_RB3 & BOREN_OFF);     //Salida del PWM por RB3
#define _XTAL_FREQ 8000000               //Oscilador Interno de 8MHZ.necesario definirlo así para los delays

unsigned char cont=0;
unsigned int contador=0;
unsigned char aux=0;
unsigned int periodo=0;
unsigned int tiempo=0;
unsigned int Tabla_per[10]={6553, 5900, 5100, 4600, 4400, 4000, 3800, 3300, 3800, 3300};
unsigned char velocidad[6]={0x25, 0x32, 0x3E, 0x4A, 0x4A, 0x00};            //Definimos el array de ciclos de trabajo
unsigned int conversor=0;
void main(void){
    TRISA=0x7E;                              //Puerto A configurado como entrada menos RA0 y RA7 como salidas
    TRISB=0xF7;                              //Puerto B configurado como entrada menos RB3 como salida


    GIE=1;                               //Interrupciones globales activadas
    PEIE=1;                              //Activa interrupciones por periféricos
    RBIE=0;                              //Desactiva interrupciones en RB<7:4>
    ANSEL=0;                             //Todas las entradas digitales
    SCS1=0;                              //T1OSC usado como sistema de reloj
    SCS0=1;
    IRCF2=1;                             //Configura oscilador interno a 8MHz
    IRCF1=1;
    IRCF0=1;
    CCP1M3=1;                            //Configurar el pic en modo PWM
    CCP1M2=1;
    CCP1M1=0;
    CCP1M0=0;
    CCPR1L=0x00;                             //Inicializamos ciclo de trabajo a cero para que al
                                             //encender esté parado
//TIMER0
    T0CS=1;                              //TIMER0 en modo contador
    T0SE=0;                              //Por flanco de subida
    PSA=1;                               //Prescaler asignado a WTD
    OPTION_REGbits.PS2=0;                //Prescaler de 1:1
    OPTION_REGbits.PS1=0;
    OPTION_REGbits.PS0=0;
    TMR0IE=1;                            //Habilitamos interrupcion en TIMER0
    TMR0=100;                            //Inicializar TMR0 a 100
```

Figura 76: Programa prueba del encoder (Parte 1)

```
//TIMER1
    TMR1CS=0;                        //Cuenta a través del reloj interno (Fosc/4)
    TMR1ON=0;                        //TIMER1 inhabilitado, se habilita después
    T1RUN=0;                         //System clock is derived from another source
    T1OSCEN=0;                       //Bit de habilitación del Timer1
    T1CONbits.T1CKPS1=1;             //Prescaler de 1:8
    T1CONbits.T1CKPS0=1;
    TMR1H=0x00;                      //Inicializamos a cero los registros del TIMER1
    TMR1L=0x00;                      //del TIMER1
    TMR1IE=0;                        //Deshabilitamos interrupción en TIMER1


//TIMER2
    T2CONbits.T2CKPS1=0;             //Prescaler 1:4
    T2CONbits.T2CKPS0=1;
    T2CONbits.TMR2ON=1;              //Activamos Timer2

    INTEDG=0;                        //Interrupcion por flanco de bajada en RB0
    INT0IE=1;                        //Habilitar interrupcion en RB0
    PR2=0x7C;                        //Frecuencia del PWM a 4KHz



while(1){
        CLRWDT();
        __delay_ms(50);}
}
static void interrupt isr(void){
__delay_ms(50);                          //Retardo para evitar rebotes en el pulsador.
if(INT0IF==1){
            INT0IF=0;                                   //Condicion de que el cont sea menor que 5
            CCPR1L = velocidad[cont];                   //Igualamos el ciclo de trabajo con el valor del array
            cont++;                                     //Incrementamos cont
            if(cont==6){                                //Si el cont es igual a 6 es que ha pasado por todas las velocidades
            cont=0;}                                    //y tiene que volver a empezar
}
```

Figura 77: Programa prueba del encoder (Parte 2)

```
if(TMR0IF==1){
            TMR0IF=0;                       //Poner a cero la bandera de interrupción
            if (aux==0){                    //Condicion de que entre al bucle
                aux=1;                      //Poner aux a uno
                TMR0=216;                   //Inicializar Timer0 a 216 para que cuente 40 pulsos
                TMR1ON=1;                   //Iniciar el Timer1
                }
            else{
                aux=0;                      //Poner aux a 1
                TMR1ON=0;                   //Parar el Timer1
                TMR0=0;                     //Dar valor cero al Timer0
                contador=TMR1L;             //Pasar al contador la parte baja del Timer1
                contador+=(TMR1H<<8);       //Pasar al contador la parte alta del Timer1
                tiempo=contador/250;        //Obtener el tiempo transcurrido en contar los 40 pulsos
                periodo=tiempo*25;          //Obtener el periodo de los ciclos
   switch(cont)
                {
                    case 1: { if (periodo>Tabla_per[0]){CCPR1L++;}      //Si el periodo es mayor que el del array
                         else if (periodo<Tabla_per[1]){CCPR1L--;}      //que aumente el ciclo de trabajo.
                         else{}                                         //Si es menor, que lo disminuya
                         break;
                }
                    case 2: {if (periodo>Tabla_per[2]){CCPR1L++;}
                         else if (periodo<Tabla_per[3]){CCPR1L--;}
                         else{}
                         break;
                    }
                    case 3: {if (periodo>Tabla_per[4]){CCPR1L++;}
                         else if (periodo<Tabla_per[5]){CCPR1L--;}
                         else{}
                         break;
                    }
                    case 4:  {if (periodo>Tabla_per[6]){CCPR1L++;}
                         else if (periodo<Tabla_per[7]){CCPR1L--;}
                         else{}
                         break;
                    }
                    case 5: {if (periodo>Tabla_per[8]){CCPR1L++;}
                         else if (periodo<Tabla_per[9]){CCPR1L--;}
                         else{}
                         break;
                         default:;
                    }

                }
```

Figura 78: Programa prueba del encoder (Parte 3)

La parte en la que se le da al pulsador y el motor actúa se explicará más adelante en el apartado de ***5. Programa y registros utilizados***.

El programa realiza lo siguiente. Inicialmente el **Timer0** tiene el valor 100. Cuando le lleguen 116 pulsos se desbordará y se producirá una interrupción, entonces el micro se dirigirá a la interrupción para tratarla. Primero ponemos a cero la bandera de interrupción. Utilizamos un contador auxiliar (*aux*) para saber si es la primera vez que se desborda o para saber si es la vez en que ha contado los cuarenta pulsos. Si es la primera vez (*aux=0*) entra en el ***if***, pone aux a uno para que la siguiente vez que se desborde vaya al ***else***, ponemos el Timer0 con valor 216, activamos el **Timer1** y sale del ***if***. El **Timer1** se activa en ese momento y no antes para que los dos temporizadores empiecen a la vez y que los cuarenta pulsos se correspondan con el tiempo transcurrido en el Timer1.

Una vez que ha contado los cuarenta pulsos, se produce otra interrupción y ahora entra en el ***else***. Ponemos *aux* a cero, el Timer0 también y paramos el Timer1. Utilizamos una variable (contador) para guardar el valor del Timer1. Primero guardamos la parte baja y luego la parte alta.

Ahora hacemos la siguiente operación para obtener el tiempo transcurrido y el periodo.

Tiempo=4*Prescaler*contador/8MHz => contador/250

Periodo=Tiempo*1000000/40 =>Tiempo*25

Cuando ya se tienen los valores del periodo, se realiza un switch para que vaya comparando el cada caso del "cont" (la velocidad en la que se encuentra el motor), el periodo que tiene con el que corresponda y que lo aumente o los disminuya según sea necesario. Otras veces no disminuye ni aumenta el ciclo de trabajo.

Cuando se prueba el programa se observa que el motor sí que actúa y que modifica el ciclo de trabajo, pero no lo hace correctamente. Para comprobarlo se varía el ciclo de trabajo y se pone uno mayor que el que le corresponde. Haciendo esto vemos como sí que reduce el ciclo de trabajo, pero no lo hace correctamente ya que no lo deja de hacer cuando debería, si no que lo disminuye hasta que prácticamente se para. Para ver qué es lo que puede estar pasando se utiliza la opción Watch que da el programa MPLAB en la que se puede ver el valor de las variables y de los registros utilizados en el programa conforme se está ejecutando el programa. Con esta herramienta se observa que el valor que se espera en el contador no es el que se obtiene.

Con esto podemos deducir o que el micro no cuenta bien los pulsos, o que no realiza bien la operación al contar los pulsos. La causa de esto puede ser la prioridad que da el micro a las interrupciones que se generan. En este micro no existe la posibilidad por software de darle prioridad a unas interrupciones frente a otras, así que, posiblemente, al producirse la interrupción externa afecte a la otra interrupción y por eso no cuente bien los pulsos.

Debido a estos problemas y a la falta de tiempo, la opción de comprobar la velocidad se va a descartar por el momento.

4.3. Duración de la batería

Como el dispositivo tenía que ser independiente de la red eléctrica la parte de la elección de la batería era un aspecto que había que decidirlo correctamente. Debido al tamaño del que se disponía se decidió que una buena opción era la de utilizar una pila de 9V como batería, ya que la placa PICDEM2PLUS puede funcionar con este mismo sistema. Pero para saber si era el sistema definitivo había que realizar una serie de pruebas.

Primero se conectó la pila en el montaje de la placa de pruebas y se comprobó con un solo motor si reaccionaba igual que conectado a la fuente de alimentación. A su vez se comprobó el consumo de corriente, que con un solo motor era de unos 80-90mA. Posteriormente se conectaron los otros dos motores que iba a llevar el dispositivo y el consumo aumentó hasta 130-140mA. Con estos valores de corriente y teniendo en cuenta que la pila tiene una

capacidad de 270mAh, sale una duración de la batería de unas dos horas. Se dejaron girando los tres motores a la velocidad dos para ver cuánto duraba en realidad la batería y ver si se correspondía con los cálculos y se observó que efectivamente sí que estaba funcionando correctamente ese tiempo, así que la el sistema de la pila de 9V era una buena opción a falta de una última prueba.

Esta prueba consistía en introducir la placa de pruebas con los tres motores girando, en la máquina donde iba a ir introducido el dispositivo. Esto se hizo para comprobar si el proceso de vacío que realiza la máquina afectaba de alguna manera a la pila y al sistema en general. Ningún elemento del circuito se vio afectado y el sistema se mantuvo funcionando las dos horas previstas.

Como ya se ha dicho en el apartado de **3.3. *Sistema de alimentación***, con esta duración se pueden realizar unos 10-12 tratamientos, que equivalen a más de 30 fibras tratadas. Una muy buena cantidad para una sola carga de la pila.

4.4. Prueba Convertidor A/D

Un pequeño problema que puede tener el dispositivo es que cuando la pila baje de 5V, en la salida del regulador habrá menos de estos 5V y por lo tanto los motores giren a unas revoluciones más bajas de lo que debería. Para evitar esto se había pensado utilizar el convertidor analógico-digital que tiene el microntrolador para medir esta tensión y si es menor que un determinado valor, por ejemplo 4,7V, que ilumine un led como aviso de poca batería e indicación de que hay que sustituir la misma.

La idea es introducir la tensión en un pin del micro, que mida esta tensión y la convierta a un valor digital, y que lo compare con un valor que vamos a establecer y si es mayor o igual a 4,7V que no ilumine el led, y si es menor que habilite una salida para que ilumine el led.

Para poder utilizar el convertidor hay que modificar los registros *ANSEL, ADCON1 Y ADCON2*.

-Registro ***ANSEL***

Este registro nos permite configurar las entradas y salidas como analógicas o como digitales. Como vamos a elegir el pin **RA1** como la entrada para medir la tensión, hay que poner a uno el bit 1, que hace referencia a este pin.

REGISTER 12-1: ANSEL: ANALOG SELECT REGISTER (ADDRESS 9Bh) PIC16F88 DEVICES ONLY

U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	ANS6	ANS5	ANS4	ANS3	ANS2	ANS1	ANS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6-0 **ANS<6:0>:** Analog Input Select bits
Bits select input function on corresponding AN<6:0> pins.
1 = Analog I/O(1,2)
0 = Digital I/O

Note 1: Setting a pin to an analog input disables the digital input buffer. The corresponding TRIS bit should be set to input mode when using pins as analog inputs. Only AN2 is an analog I/O, all other ANx pins are analog inputs.

2: See the block diagrams for the analog I/O pins to see how ANSEL interacts with the CHS bits of the ADCON0 register.

Figura 79: Registro ANSEL

-Registro ***ADCON0***

REGISTER 12-2: ADCON0: A/D CONTROL REGISTER (ADDRESS 1Fh) PIC16F88 DEVICES ONLY

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7-6 **ADCS<1:0>:** A/D Conversion Clock Select bits
If ADCS2 = 0:
00 = Fosc/2
01 = Fosc/8
10 = Fosc/32
11 = FRC (clock derived from the internal A/D module RC oscillator)
If ADCS2 = 1:
00 = Fosc/4
01 = Fosc/16
10 = Fosc/64
11 = FRC (clock derived from the internal A/D module RC oscillator)

bit 5-3 **CHS<2:0>:** Analog Channel Select bits
000 = Channel 0 (RA0/AN0)
001 = Channel 1 (RA1/AN1)
010 = Channel 2 (RA2/AN2)
011 = Channel 3 (RA3/AN3)
100 = Channel 4 (RA4/AN4)
101 = Channel 5 (RB6/AN5)
110 = Channel 6 (RB7/AN6)

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
If ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion)
0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

bit 1 **Unimplemented:** Read as '0'

bit 0 **ADON:** A/D On bit
1 = A/D converter module is operating
0 = A/D converter module is shut off and consumes no operating current

Figura 80: Registro ADCON0

En los bits 7 y 6 del registro ***ADCON0*** nos permite configurar el tiempo de conversión. Para ello hay que tener en cuenta la frecuencia a la que hemos configurado el oscilador interno, en

nuestro caso 8MHz. Según el datasheet el tiempo no debe estar comprendido entre 1,6 y 6,4us. Según esto debimos elegir el caso en que la *Fosc/16*, ya que salen 2us. Además cumplimos con esta tabla:

TABLE 12-1: TAD vs. MAXIMUM DEVICE OPERATING FREQUENCIES – STANDARD DEVICES (C)

AD Clock Source (TAD)			Maximum Device Frequency
Operation	ADCS<2>	ADCS<1:0>	Max.
2 Tosc	0	00	1.25 MHz
4 Tosc	1	00	2.5 MHz
8 Tosc	0	01	5 MHz
16 Tosc	1	01	10 MHz
32 Tosc	0	10	20 MHz
64 Tosc	1	10	20 MHz
RC(1,2,3)	x	11	(Note 1)

Figura 81: Tabla comparación Periodo convertidor/Frecuencia máxima del micro

Por lo tanto tendríamos que configurar ***ADCS<2:0>*** = **101**. Los bits 5-3 son para elegir el canal por el que vamos a introducir la señal analógica, en este caso como es **RA1** hay que elegir el Channel1 configurando ***CHS<2:0>*** = **001**. EL bit 2 y el bit 0 son para iniciar la conversión poniendo los dos a uno. El bit 1 no se configura.

-Registro ***ADCON1***

REGISTER 12-3: ADCON1: A/D CONTROL REGISTER 1 (ADDRESS 9Fh) PIC16F88 DEVICES ONLY

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
ADFM	ADCS2	VCFG1	VCFG0	—	—	—	—
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified. Six Most Significant bits of ADRESH are read as '0'.
0 = Left justified. Six Least Significant bits of ADRESL are read as '0'.

bit 6 **ADCS2:** A/D Clock Divide by 2 Select bit
1 = A/D clock source is divided by 2 when system clock is used
0 = Disabled

bit 5-4 **VCFG<1:0>:** A/D Voltage Reference Configuration bits

Logic State	VREF+	VREF-
00	AVDD	AVSS
01	AVDD	VREF-
10	VREF+	AVSS
11	VREF+	VREF-

Note: The ANSEL bits for AN3 and AN2 inputs must be configured as analog inputs for the VREF+ and VREF- external pins to be used.

bit 3-0 **Unimplemented:** Read as '0'

Figura 82: Registro ADCON1

Con el registro ***ADCON1*** terminamos de configurar el convertidor A/D. Con el bit 7 elegimos la opción de justificar el registro *ADRES* a la izquierda o a la derecha. EL registro *ADRES* es donde se guarda el valor de la conversión. Es un registro dividido en dos de 8bits, ***ADRESL*** y ***ADRESH***, parte alta y baja respectivamente. Pero en realidad sólo se pueden se utilizan 10 bits,

justificado a la derecha o a la izquierda. En nuestro caso lo justificamos a la derecha, así que este bit estará puesto a uno.

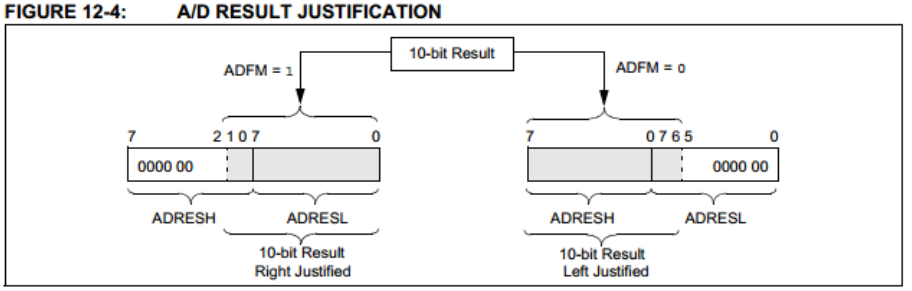


Figura 83: Justificación Convertidor A/D

El bit 6 estará a uno, ya que según el registro ***ADCON0*** si queremos un tiempo de *Fosc/16* ***ADCS2*** debe valer uno. Los bit 5 y 4 son para establecer los valores de referencia superior (Vref+) e inferior (Vref-) del convertidor. Los ponemos los dos a cero ya que vamos a tomar como referencia los valores de alimentación que le llegan al micro. También se puede coger otra referencia e introducirla por las entradas **RA2** y **RA3**, eso sí, configuradas como entradas analógicas.

Una vez configurados los registros del convertidor hay que realizar las pruebas para ver si está todo bien configurado. Una de las cosas que hay que hacer previamente es establecer el valor a partir del cual vamos a iluminar el led. Nosotros queremos que avise cuando la tensión sea menor que 4,7 voltios. Como el registro donde se guarda el valor de la conversión es de 10 bits, y nuestro rango va de 0 a 5V, entonces cada bit será: 5/1024=4,88 mV. Como queremos que avise con 4,7 voltios, que equivale a un valor de: 4,7V/4,88mV=**963**. Por lo tanto el programa tendrá que mirar que el registro valga menos de **963** y cuando sea así habilite una salida y se ilumine el led.

Como el convertidor tiene la posibilidad de que el terminar la conversión se genere una interrupción, vamos a utilizar esta opción.. Lo que debe hacer el programa es leer el pin de entrada, convertirlo a un valor numérico, y entonces se generará una interrupción, en esa interrupción se comparará con 963, si es menor se habilitará una salida para encender el led, y si es mayor no hará nada.

Cuando el programa esté completo se generará un retardo para que cada un determinado tiempo mire la tensión pero para estas pruebas se va a hacer para que lo mire nada más ejecutarse el programa

Se escribe el programa y queda así:

-Parte de configuración del convertidor-

```
void main(void){
    TRISA=0x7E;                         //Puerto A configurado como entrada menos RA0 y RA7 como salidas
    TRISB=0xF7;                         //Puerto B configurado como entrada menos RB3 como salida


    GIE=1;                              //Interrupciones globales activadas
    PEIE=1;                             //Activa interrupciones por periféricos
    RBIE=0;                             //Desactiva interrupciones en RB<7:4>
    ANSEL=0;                            //Todas las entradas digitales

//Configuración convertidor
    ANSELbits.ANS1=1;                   //RA1 como entrada analógica
    PORTAbits.RA0=0;                    //inicializar a cero RA0
    PORTAbits.RA7=0;
    ADCS2=1;                            //Establecer el tiempo de conversión en Tosc*16->2us
    ADCS1=0;
    ADCS0=1;
    CHS2=0;                             //Elegimos RA1 como la entrada del conversor
    CHS1=0;
    CHS0=1;
    ADFM=1;                             //Resultado justificado a la derecha
    ADIE=1;                             //Activar interrupción cuando el conversor termine la conversión
    VCFG1=0;                            //Vref+ = Vdd
    VCFG0=0;                            //Vref- = Vss
    ADCS2=0;                            //Deshabilitado A/D clock/2
    ADRESH=0;                           //Inicializar a cero
    ADRESL=0;
    ADON=1;                             //Se activa el conversor
    ADCON0bits.GO=1;                    //Bit del estado de la converión
```

Figura 84: Programa prueba del convertidor (Parte 1)

-Parte de tratamiento de la interrupción-

```
while(1){
        CLRWDT();
        __delay_ms(50);}
}
static void interrupt isr(void){
__delay_ms(50);                                 //Retardo para evitar rebotes en el pulsador.
if(INT0IF==1){
            INT0IF=0;                               //Condicion de que el cont sea menor que 5
            CCPR1L = velocidad[cont];               //Igualamos el ciclo de trabajo con el valor del array
            cont++;                                 //Incrementamos cont
            if(cont==6){                            //Si el cont es igual a 6 es que ha pasado por todas las velocidades
            cont=0;}                                //y tiene que volver a empezar
}
else if(ADIF==1){
                ADIF=0;                             //Poner a cero la bandera de interrupción
                conversor=ADRESL;                   //Coger la parte baja del valor de la conversión
                conversor+=(ADRESH<<8);             //Coger la parte alta del valor de la conversión
                if(conversor<963){                  //Condición de que el valor sea menor que 963
                PORTAbits.RA0=1;                    //Si el valor es menor habilita la salida.
                }
}
}
```

Figura 85: Programa prueba del convertidor (Parte 2)

Para realizar las pruebas, con la ayuda de la fuente de alimentación, se van a introducir valores superiores e inferiores a 4,7 V en el pin **RA1** y comprobar con la herramienta "Watch" de MPLAB el valor de la conversión. Se hacen las pruebas y se observa cómo hace bien la conversión y el led luce el valor es menor, y no lo hace cuando es mayor.

La siguiente prueba consiste en conectar la alimentación al pin de entrada **RA1** y reducir la alimentación de todo el dispositivo, ya que es lo que tiene que mirar cuando ya esté funcionando completamente.

Se realiza la operación y se comprueba que no hace bien la conversión y que no luce nunca el led. Esto es debido a que como tenemos como ***Vref+*** la alimentación del dispositivo, y al reducir esta alimentación también se reduce esta referencia y en consecuencia el mocrocontrolador no realiza la conversión como si la ***Vref+*** fueran 5V y los valores no se corresponden. Haciéndolo así nunca va a lucir el led. Una opción sería poner un divisor de tensión y tomar ahí la ***Vref+,*** pero nos encontraríamos con el mismo problema, ya que todo depende de esa alimentación. Definitivamente no podemos utilizar el convertidor analógico-digital para mirar la carga de la batería.

5. Programa y registros del micro

En ese apartado se va a explicar el programa definitivo que va a controlar el dispositivo. Primero se van a explicar los registros utilizados y posteriormente se expondrá el funcionamiento del programa.

5.1 Registros utilizados

Los registros utilizados han sido el INTCON, OSCCON, CCP1CON y el T2CON [54] [55].

5.1.1. Registro INTCON

REGISTER 2-3: INTCON: INTERRUPT CONTROL REGISTER (ADDRESS 0Bh, 8Bh, 10Bh, 18Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
bit 7							bit 0

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **PEIE:** Peripheral Interrupt Enable bit
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INT0IE:** RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit
A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition and allow flag bit RBIF to be cleared.
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

Figura 86: Registro INTCON

Este registro es utilizado para habilitar o deshabilitar las interrupciones en el micro así como las banderas de estado de estas interrupciones que permite el micro. En nuestro programa como vamos a utilizar interrupciones externas (por periféricos) tenemos que poner a uno el bit 7 (GIE) que hace referencia a las interrupciones globales. Al bit 6 (PEIE) también se le da el valor uno porque habilita las interrupciones por periféricos. El bit 5 (TMR0IE) habilita la interrupción por desbordamiento en el Timer0, el cual no nos interesa y no lo configuramos. EL bit 4 (INT0IE) habilita la interrupción externa en **RB0/INT**, así que lo ponemos a uno.

El bit 3 (RBEIE) lo ponemos a uno para deshabilitar la interrupción por el puerto B. Esto lo hacemos para evitar posibles interrupciones no deseadas por los posibles ruidos creados en el circuito. El bit 2 (TMR0IF) hace referencia a la bandera de la interrupción creada por el desbordamiento del **Timer0**. Los bits 1 y 0 (INT0IF y RBIF) hacen referencia a la bandera de interrupción en el pin **RB0/INT** y en el puerto B (PORTB), respectivamente. En nuestro programa sólo trataremos la **INT0IF**.

5.1.2. Registro OSCCON

REGISTER 4-2: OSCCON: OSCILLATOR CONTROL REGISTER (ADDRESS 8Fh)

U-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0
—	IRCF2	IRCF1	IRCF0	OSTS(1)	IOFS	SCS1	SCS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6-4 **IRCF<2:0>:** Internal RC Oscillator Frequency Select bits
000 = 31.25 kHz
001 = 125 kHz
010 = 250 kHz
011 = 500 kHz
100 = 1 MHz
101 = 2 MHz
110 = 4 MHz
111 = 8 MHz

bit 3 **OSTS:** Oscillator Start-up Time-out Status bit(1)
1 = Device is running from the primary system clock
0 = Device is running from T1OSC or INTRC as a secondary system clock

Note 1: Bit resets to '0' with Two-Speed Start-up mode and LP, XT or HS selected as the oscillator mode.

bit 2 **IOFS:** INTOSC Frequency Stable bit
1 = Frequency is stable
0 = Frequency is not stable

bit 1-0 **SCS<1:0>:** Oscillator Mode Select bits
00 = Oscillator mode defined by FOSC<2:0>
01 = T1OSC is used for system clock
10 = Internal RC is used for system clock
11 = Reserved

Figura 87: Registro OSCCON

Este registro nos permite configurar parámetros del oscilador interno. El bit 7 no se configura. Los bits 6-4 nos permiten elegir entre las diferentes frecuencias del oscilador interno que tiene disponibles el micro. En este caso vamos a usar la de 8MHz así que tendremos que poner IRCF<2:0> = 111. Los bits 1-0 son para configurar el modo del oscilador, en este caso lo configuramos como "01".

5.1.3. Registro CCP1CON

Con este registro elegimos entre el modo captura, el modo comparador o el modo PWM.

REGISTER 9-1: CCP1CON: CAPTURE/COMPARE/PWM CONTROL REGISTER 1 (ADDRESS 17h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **CCP1X:CCP1Y:** PWM Least Significant bits

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPR1L.

bit 3-0 **CCP1M<3:0>:** CCP1 Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCP1 module)
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, set output on match (CCP1IF bit is set)
1001 = Compare mode, clear output on match (CCP1IF bit is set)
1010 = Compare mode, generate software interrupt on match (CCP1IF bit is set, CCP1 pin is unaffected)
1011 = Compare mode, trigger special event (CCP1IF bit is set, CCP1 pin is unaffected); CCP1 resets TMR1 and starts an A/D conversion (if A/D module is enabled)
11xx = PWM mode

Figura 88: Registro CCP1CON

Como en nuestro caso vamos a utilizar un PWM para controlar la velocidad de los motores debemos configurar los bits 3-0 como "11xx". Los bit 5-4 se utilizan para dar el ciclo de trabajo deseado al PWM. Este registro se explica más detalladamente en el apartado *4.1 Prueba del PWM.*

5.1.4. Registro T2CON

REGISTER 8-1: T2CON: TIMER2 CONTROL REGISTER (ADDRESS 12h)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6-3 **TOUTPS<3:0>:** Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
0010 = 1:3 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON:** Timer2 On bit
1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Figura 89: Registro T2Con (Timer2)

Este registro nos permite configurar el Timer2. Tal y como se explica en el apartado ***4.1. Prueba del PWM***, el Timer2 lo usamos para variar la frecuencia del PWM. Para configurar el prescaler del PWM con el Timer2, hay que configurar el bit 0 y bit 1 del registro poniéndolos a cero los dos, porque es un prescaler de 1:1, y para activarlo hay que poner el bit 2 a uno. Los bits 3 a 7 no se configuran.

Un registro que también se modifica es el PR2, que hace referencia al periodo del Timer2 o a lo que es lo mismo a la frecuencia del PWM.

5.2. Explicación del programa

Una vez explicados los registros utilizados se va a comentar el programa que se ha utilizado para controlar la velocidad de los motores. Es un programa corto y sencillo pero que es suficiente para nuestra aplicación.

El programa va a realizar la siguiente operación. Nada más encender el dispositivo, el motor estará parado. Cuando activemos el pulsador el motor empezará a girar con un ciclo de trabajo del 30%. Con la siguiente pulsación aumentará el ciclo hasta el 40%, así sucesivamente hasta que pulsemos una sexta vez, en ese momento el motor se parará. SI volvemos a pulsar el motor empezará a girar con un ciclo del 30% otra vez. La velocidad mínima del dispositivo será del 30% y la máxima del 70%.

-Programa principal-

```
//*Programa final*//
#include<htc.h>                                  //Incluimos librería del micro a usar
__CONFIG(WRT_OFF & WDTE_OFF & PWRTE_OFF & FOSC_INTOSCIO & LVP_OFF & CCPMX_RB3 & BOREN_OFF);     //Salida del PWM por RB3
#define _XTAL_FREQ 8000000                       //Oscilador Interno de 8MHZ.Necesario definirlo así para los delays

unsigned char cont=0;
unsigned char velocidad[6]={0x4B, 0x64, 0x7C, 0x95, 0xAE, 0x00};    //Definimos el array de ciclos de trabajo
void main(void){

    TRISA=0xFF;                                  //Puerto A configurado como entrada
    TRISB=0xF7;                                  //Puerto B configurado como entrada menos RB3 como salida
    GIE=1;                                       //Interrupciones globales activadas
    PEIE=1;                                      //Activa interrupciones por periféricos
    RBIE=0;                                      //Desactiva interrupciones en RB<7:4>
    INTEDG=0;                                    //Interrupcion por flanco de bajada en RB0
    INT0IE=1;                                    //Habilitar interrupcion en RB0
    ANSEL=0;                                     //Todas las entradas digitales
    SCS1=0;                                      //T1OSC usado como sistema de reloj
    SCS0=1;
    IRCF2=1;                                     //Configura oscilador interno a 8MHz
    IRCF1=1;
    IRCF0=1;

//PWM
    CCP1M3=1;                                    //Configurar el pic en modo PWM
    CCP1M2=1;
    CCP1M1=0;
    CCP1M0=0;
    CCPR1L=0x00;                                 //Inicializamos ciclo de trabajo a cero para que al
                                                 //encenderlo el motor esté parado

//TIMER2
    T2CONbits.T2CKPS1=0;                         //Prescaler 1:1
    T2CONbits.T2CKPS0=0;
    T2CONbits.TMR2ON=1;                          //Activamos Timer2
    PR2=0xF9;                                    //Frecuencia del PWM a 8KHz

while(1){
        CLRWDT();
        __delay_ms(50);}
}
```

Figura 90: Programa final (Parte 1: Programa principal)

Se va a explicar las partes más destacadas del programa. En la primera línea cabe destacar que se han hecho tres modificaciones en el registro de configuración. Una es la de incluir **FOSC_INTOSCIO** para indicar que el micro utilice el oscilador interno. Otra es la de **CCPMX_RB3**, que es para indicar al micro que la salida del PWM la haga por el pin RB3 y no por el RB0, que es como lo hace por defecto.

En la siguiente línea ***#define _XTAL_FREQ 8000000*** se establece la frecuencia del oscilador interno para poder utilizar los ***delays***. Con esta línea no quiere decir que hemos establecido el oscilador interno a 8MHz, esto se hace más adelante, si no que es una frecuencia para que el micro pueda realizar bien los retardos.

En la siguiente línea declaramos la variable “cont” que nos sirve para indicar a la velocidad que tiene que ir el micro y se inicializa a cero.

Luego declaramos el array donde van a ir las diferentes velocidades a las que van a girar los motores.

Ya en el programa principal se configuran los puertos A y B como salidas o entradas según corresponda y se configuran los diferentes registros explicados anteriormente como habilitar interrupciones, configurar el PIC en modo PWM, etc.

Finalmente se activa el Timer2, se le da el valor F9 al PR2 y se pone un while para que el programa no haga nada hasta que se produzca una interrupción. También se limpia el Watchdog para evitar interrupciones.

Hasta aquí llega el programa principal. Ahora se va a explicar la parte del tratamiento de la interrupción.

-Tratamiento de la interrupción-

```
//Tratamiento de la interrución
static void interrupt isr(void){
__delay_ms(50);                                  //Retardo para evitar rebotes en el pulsador.
if(INT0IF==1){
            INT0IF=0;                            //Poner a cero la bandera de interrupción
            CCPR1L = velocidad[cont];            //Igualamos el ciclo de trabajo con el valor del array
            cont++;                              //Incrementamos cont
            if(cont==6){                         //Si el cont es igual a 6, es que ha pasado por todas las velocidades
            cont=0;}                             //y tiene que volver a empezar
}
}
```

Figura 91: Programa final (Parte 2: Tratamiento de la interrupción)

Cuando se produce una interrupción (se le ha dado al pulsador), el programa tiene que tratarla. Lo primero es incluir un delay de 50 milisegundos para evitar rebotes en el pulsador y que haga que el programa no funcione correctamente. Si no lo incluimos se producen saltos de velocidad.

Luego se utiliza la bandera de interrupción ***INT0IF*** para entrar a un ***if***. Dentro de este bucle ponemos esta bandera a cero, y le damos al registro ***CCPR1L*** (que es el que determina la anchura del PWM) el valor del array de velocidad en la posición cero, es decir, la primera velocidad. Para ello utilizamos la variable “cont”. En la siguiente línea aumentamos esta variable en una unidad para que la siguiente vez que entremos al bucle, pase a la siguiente posición del array y así aumente a la siguiente velocidad. Antes de salir del primer bucle ponemos un ***if*** para que si el valor de esta variable “cont” vale 6, que quiere decir que ya ha pasado por todas las velocidades y está en la última posición del array (el motor está parado), ponga a cero la variable "cont" y así la siguiente vez que se produzca la interrupción vaya a la primera posición y el motor arranque con la primera velocidad del 30%. Si el valor de “cont” es menor que 6 entonces no hace nada y pasa a la siguiente velocidad que le corresponde.

6. Diseño de la placa

Para diseñar la placa se ha utilizado el software gratuito DesignSpark PCB [48]. En el ***Anexo 1*** se explica más detalladamente el funcionamiento y las posibilidades que tiene. En este apartado se va a intentar resumir el proceso que se ha realizado hasta completar la placa definitiva.

Figura 92: Logo DesignSpark [37]

6.1. Diseño del esquema

Para poder diseñar la placa primero hay que realizar el esquema en el programa. Para ello se ha buscado en las librerías del programa los componentes que se iban a utilizar. Muchos de ellos estaban incluidos en las librerías pero otros tuvieron que ser buscados en el catálogo de RS e incluidos en las librerías y otros tuvieron que ser creados desde cero.

Para entender el esquema se va a dividir en dos, la parte de alimentación y la parte de control.

-Parte de la alimentación-

A la hora de realizar la parte de la alimentación se escogió como modelo la parte de la alimentación de la placa PICDEM2 PLUS. La parte de alimentación consta de una regleta para la conexión de los cables de la pila, el interruptor de encendido, un diodo, el regulador de tensión y los condensadores electrolíticos a la entrada y la salida del regulador. La parte de masa del regulador se une a una conexión de 0V y la de alimentación a una de 5V.

Todos los terminales del resto de componentes del circuito que estén unidos a una conexión de 5V estarán unidos a la salida del regulador en el circuito real. Lo mismo pasa con la conexión de 0V.

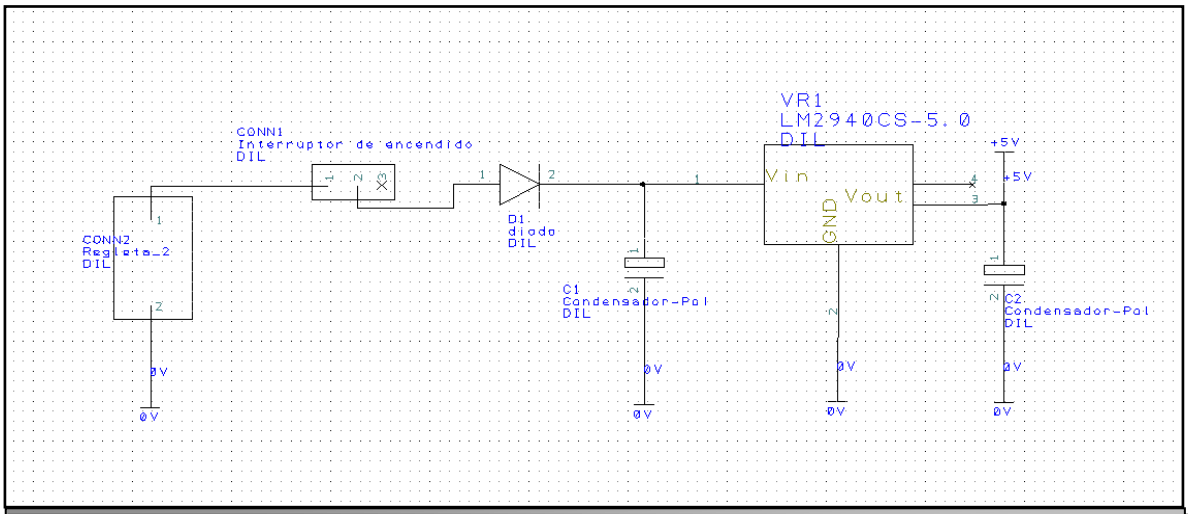


Figura 93: Esquema (Parte 1: Alimentación)

-Parte de control-

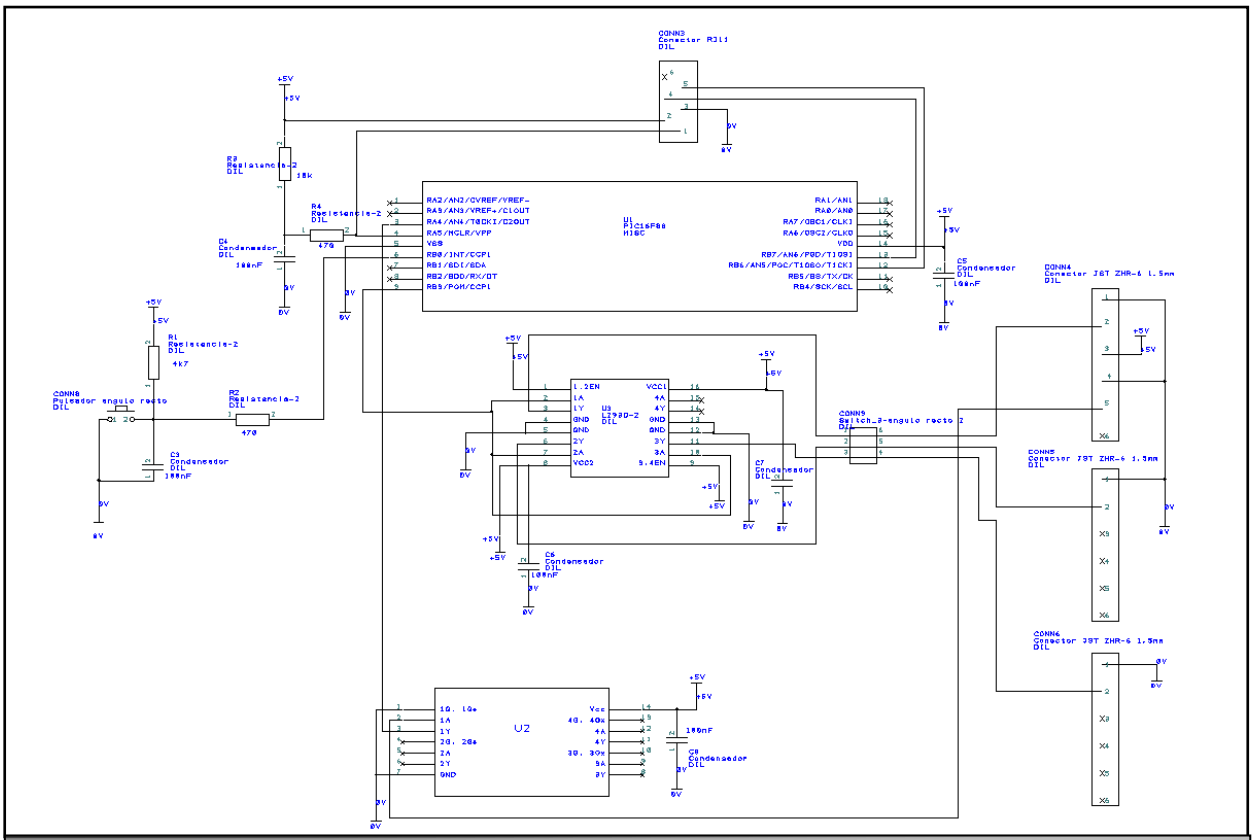


Figura 94: Esquema (Parte 2: Control)

La parte de control contiene muchos más elementos. Destacan los tres integrados que son el PIC, el driver y el acondicionador de señal. También está el pulsador con sus resistencias y condensadores necesarios para su correcto funcionamiento, el conector RJ11, el switch de tres posiciones, los conectores para los motores y los condensadores de desacople para la alimentación de los integrados.

6.2. Diseño de los componentes

Para diseñar los componentes que no estaban en las librerías hay que tener en cuenta que hay que crearlos tanto para el esquema como para el diseño del PCB. Para realizar el símbolo de esquema no se tuvo en cuenta el tamaño del componente, pero sí evidentemente el numero de pines o terminales y la forma en la que estaban dispuestos.

Para realizar los símbolos de PCB hay que considerar el tamaño y forma del mismo, y también el tamaño, forma y posición exacta de los pines del componente. Esto es así porque los componentes tienen que entrar exactamente en los orificios creados en la placa. Si un terminal es por ejemplo un milímetro más ancho que el orificio no se podrá colocar bien en la placa. Y al contario, si el orificio es demasiado grande respecto al terminal, puede haber problemas para soldar el componente. Para ello se utilizaron las hojas de características de cada componente y un calibre como herramienta de medida.

Los elementos que se tuvieron que crear fueron el conector RJ11, el interruptor de encendido, el pulsador, el switch y los conectores de los motores. Además, el resto de componentes que sí estaban en las librerías se tuvieron que modificar para facilitar su soldadura como por ejemplo los integrados y condensadores.

6.3. Pruebas de la placa

Una vez diseñado los componentes necesarios se comenzaron a hacer pruebas del enrutado de la placa para ver si los componentes estaban bien diseñados.

Estas pruebas consistían en imprimir en papel la cara top y la cara bottom de la placa, y comprobar si los elementos coincidían con el tamaño que tenían en realidad. Para ello simplemente había que ir al menú del programa y seleccionar la opción de imprimir, y el programa te creaba un archivo **.pdf** a tamaño real de lo que seleccionases, ya sea la cara top, la bottom, serigrafías, etc. El resultado era el siguiente:

De esta manera se pudo comprobar además de que los componentes estaban bien diseñados, el tamaño real que iba a tener la placa (Figura 95).

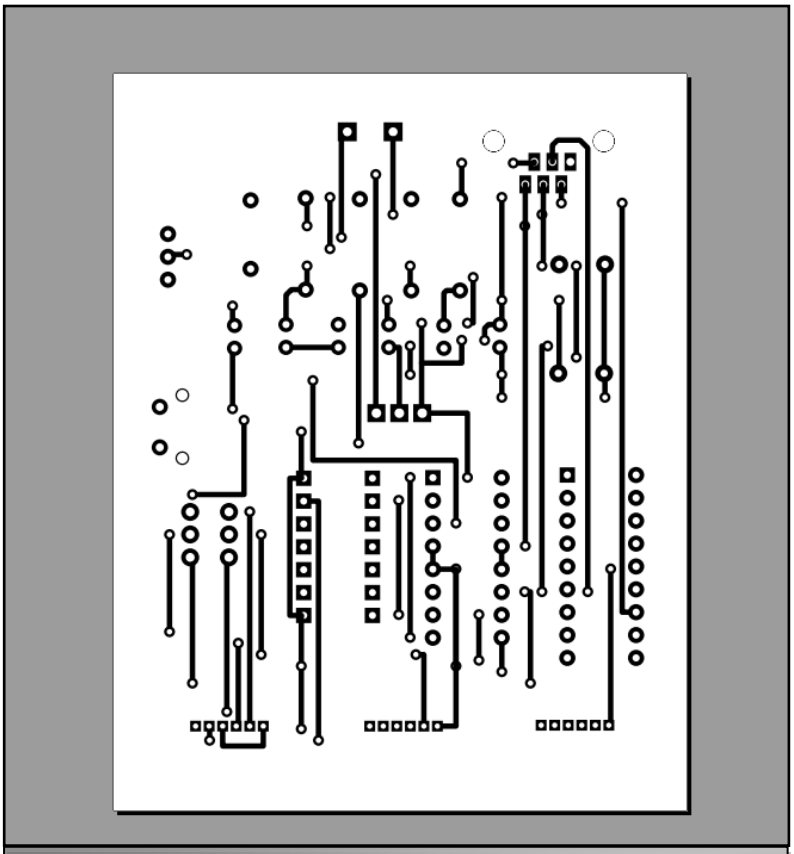


Figura 95: Prueba de la placa en papel

6.4. Conceptos a tener en cuenta

Una vez diseñados todos los componentes se procedió a realizar el enrutado de la placa. Para realizar este proceso correctamente hay que tener en cuenta una serie de aspectos:

- Anchura de las pistas: según la corriente que vaya a pasar por el circuito se necesitará más o menos anchura en la pista. En este caso como van a pasar 180mA como máximo, se hicieron las pistas de 0,6mm de ancho, más que suficiente para esa corriente.

-Número de vías: las vías son orificios en la placa que comunican la cara top con la bottom. Como el proceso de fabricación de la placa se realiza en la propia universidad, estas vías no tienen cobre en el propio orificio, es decir hay que hacer mediante soldadura que se unan la cara top y la bottom. Por esta razón es necesario reducir el número de vías lo máximo posible, ya que ese proceso de soldadura es algo complicado, así que cuantas menos veces haya que realizarlo mejor. Si la construcción de la placa se realizase en una empresa especializada, el rellenado de cobre de las vías lo realiza la propia empresa y no es necesario hacerle ninguna soldadura, y reducir el número de vías no es tan necesario. Aunque se construya la placa en una empresa, durante el diseño hay que estar atento ya que el programa hace vías que no son necesarias para el correcto funcionamiento del circuito, y éstas hay que eliminarlas.

-Pistas en cara top y bottom: al colocar algunos componentes como por ejemplo el conector RJ11, quedan pegados a la propia placa, por ello hay que hacer que las pistas que le llegan, sean de la capa bottom, porque si llegaran de la top no se podría soldar. Esto sucede en este conector RJ11, en los conectores de los motores, en interruptor de encendido, en la regleta, en el switch y en menor medida en los integrados, ya que estos según la pista que sea, se puede poner por la capa top.

Otro aspecto relacionado con las pistas es que hay que evitar en la medida de lo posible los ángulos rectos al realizar cambios de dirección.

-Colocación de los componentes: hay componentes que deben estar colocados en una posición en concreta debido al diseño del dispositivo, como son el conector RJ11, el switch y el interruptor de encendido, que deben estar en un lateral. Pero otros como las resistencias, condensadores e integrados pueden ir en cualquier punto, así que hay que pensar en qué posición hay que colocarlos para que el enrutado sea lo más sencillo posible y que cumplan las condiciones anteriormente mencionadas y que a la vez queden bien visualmente en el conjunto de la placa.

-Capa de masa: cuando ya se han hecho las pistas adecuadamente, se han reducido el número de vías y se han colocado los componentes, antes de proceder a la fabricación de la placa, hay colocar una capa de masa (tanto en la capa top como la bottom) para darle estabilidad al circuito. Esta capa se extiende por toda la superficie de la placa y es necesaria para el correcto funcionamiento del circuito.

6.5. Primer prototipo

Después de explicar los conceptos para realizar la placa, se hizo un primer prototipo. El resultado en el diseño quedó con en la Figura 96.

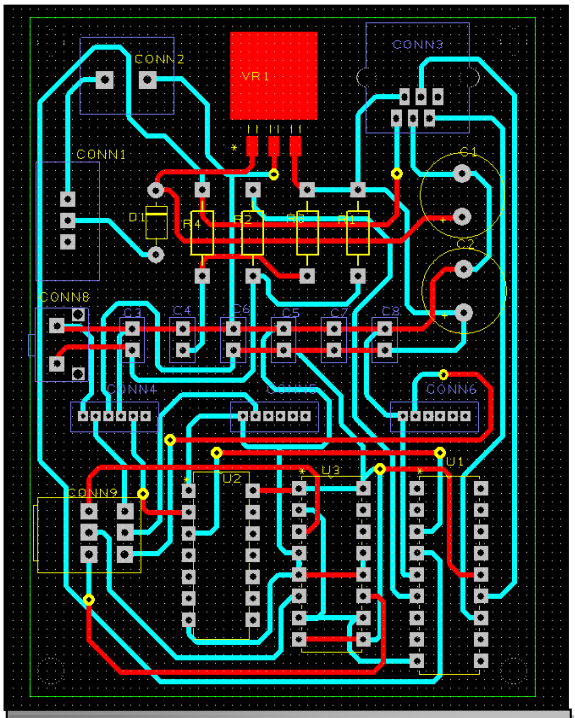


Figura 96: Prototipo sin capas de masa

Después se le añadió las capas de masa:

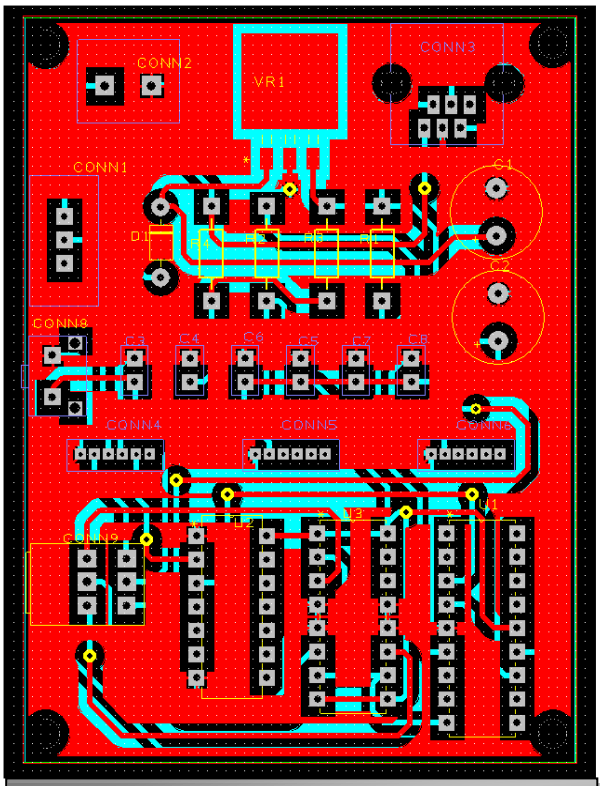


Figura 97: Prototipo con capas de masa

Después se activa la vista en 3D para visualizar mejor el posible resultado final:

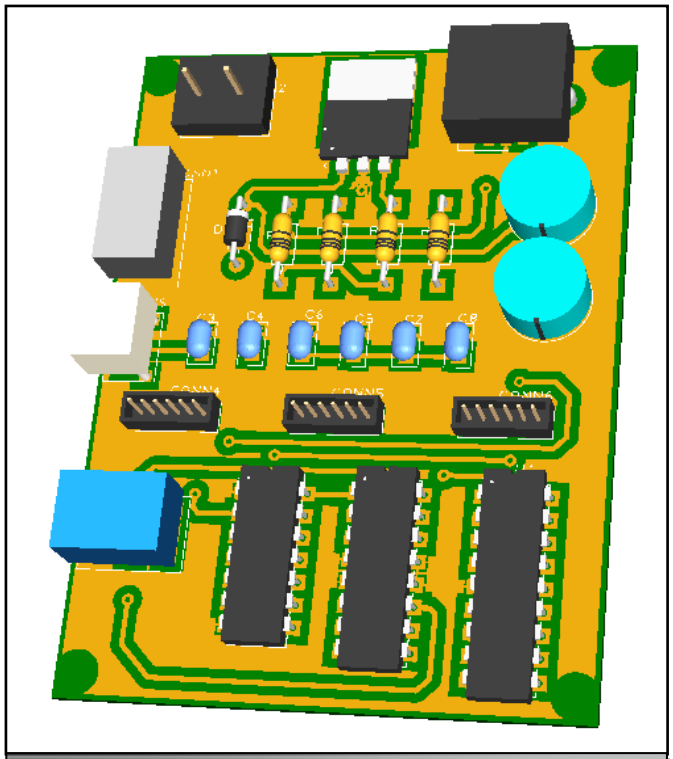


Figura 98: Prototipo Imagen en 3D

Finalmente la placa con los componentes soldados Figura 99 y 100.

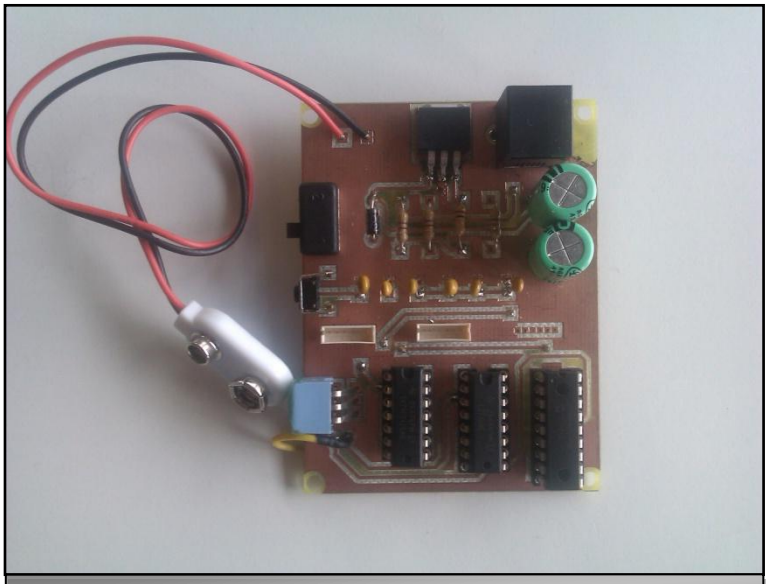


Figura 99: Prototipo Imagen real cara top

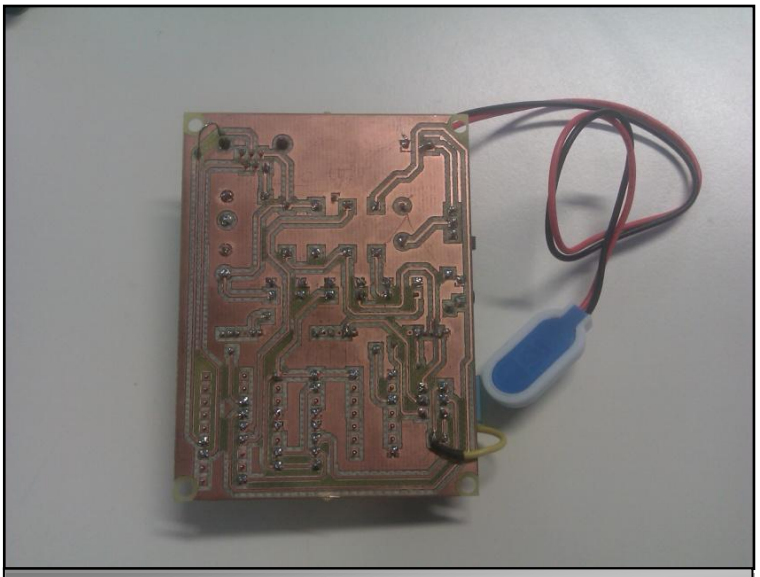


Figura 100: Prototipo Imagen real cara bottom

En este primer prototipo surgieron una serie de problemas que han provocado que no llegara a ser la placa final. En primer lugar no se tuvo en cuenta que dos pines de los conectores de los motores tenían diferente forma y a la hora de colocarlos en la placa era prácticamente imposible, aunque se pudo colocar al menos uno para comprobar el correcto funcionamiento del circuito. Otro problema fue al soldar una vía, que hubo que solucionar colocando un cable que uniera los dos puntos. Además, durante la fabricación desapareció un tramo de una pista posiblemente porque no había espacio suficiente entre la misma y un orificio para el conector RJ11 y hubo que utilizar un cable para solucionarlo. También la regleta no se pudo colocar, pero se solucionó soldando los cables del conector de la pila. Una vez reparados los errores, se

comprobó que todo funcionaba correctamente, tanto el motor como la comunicación con el MPLAB. Este prototipo sirvió para comprobar que el esquema era correcto y para descubrir pequeños fallos que no surgen hasta que no se hace un prototipo real.

6.6. Placa final

Para evitar que se repitieran los errores surgidos en el primer prototipo, se modificaron la forma de los conectores de los motores, el tamaño de los orificios para la regleta y se separó la pista que había dado problemas. Además, debido a cambios en el diseño que se iba a hacer en el dispositivo, hubo que recolocar el conector RJ11, que pasó de estar en la parte superior de la placa, a estar en el lateral. Como consecuencia hubo que mover el switch, el pulsador, el interruptor de encendido y los conectores para los motores.

Al realizar estos cambios el diseño de la placa quedó como en las Figura 101, 102 y 103.

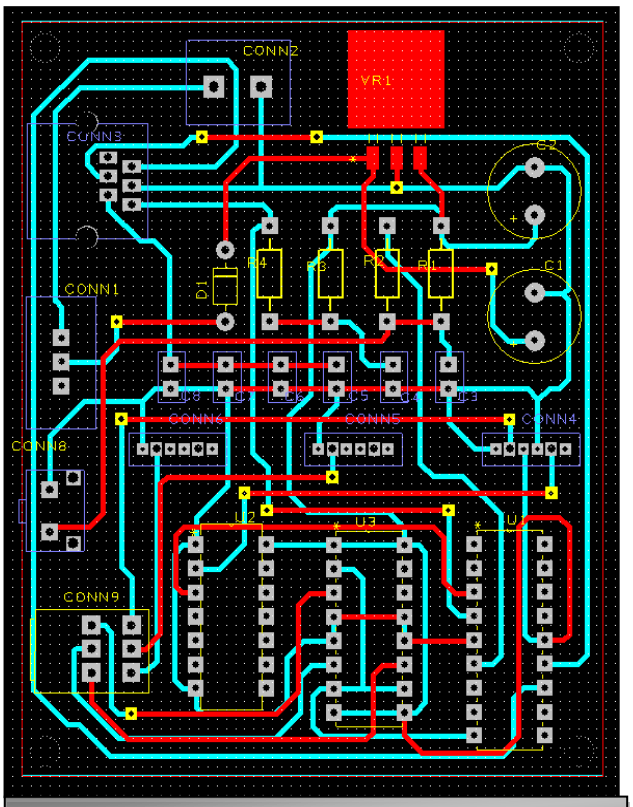


Figura 101: Placa final sin capas de masa

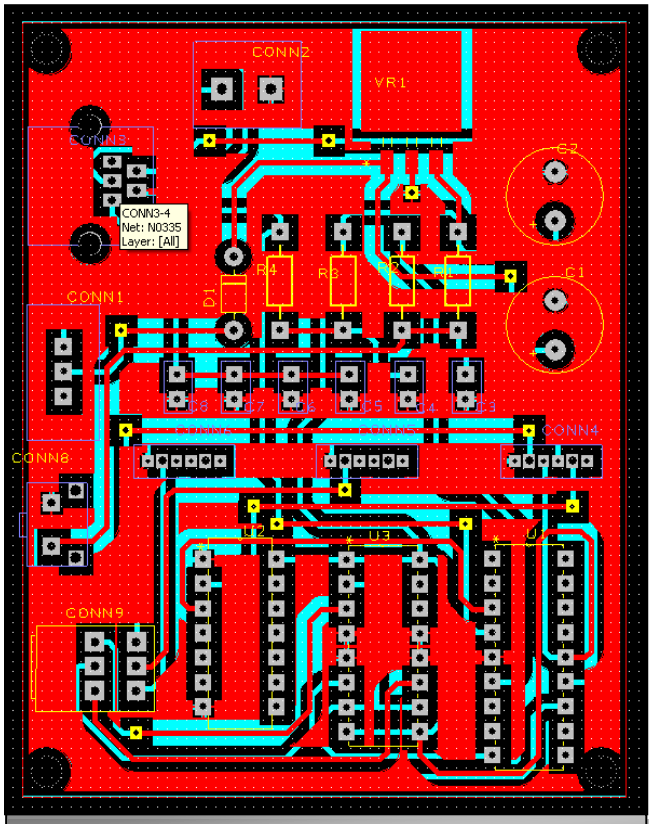


Figura 102: Placa final con capas de masa

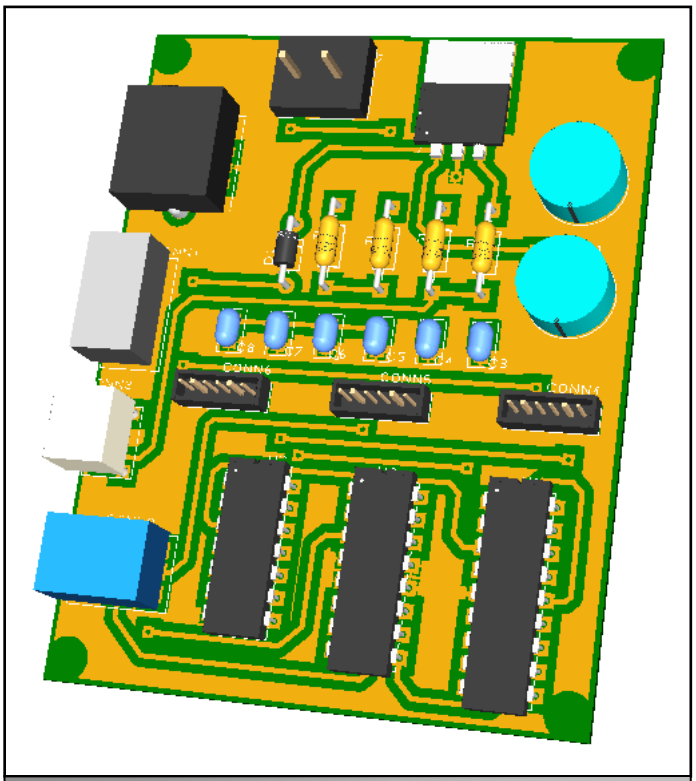


Figura 103: Placa final Imagen 3D

Y finalmente la placa con los componentes:

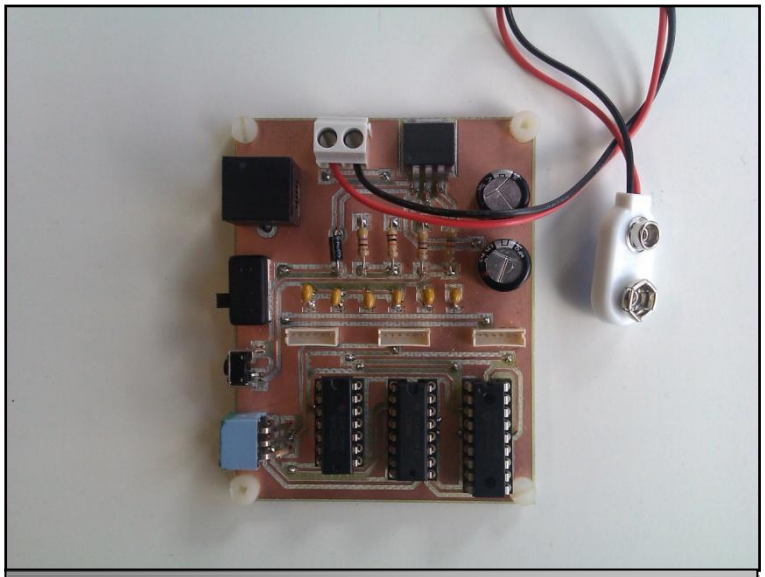


Figura 104: Placa final Imagen real cara top

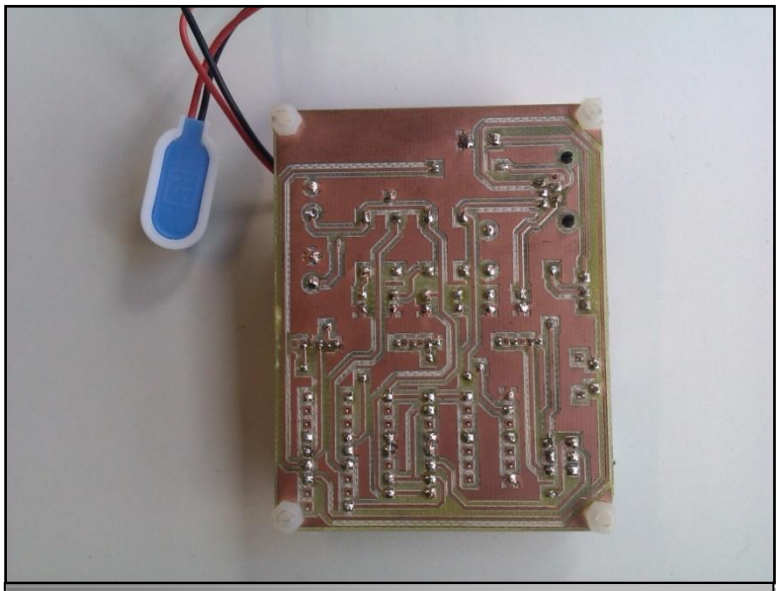


Figura 105: Placa final imagen real cara bottom

Hasta aquí viene el diseño y construcción de la placa. En el apartado siguiente se va a explicar el proceso de construcción del dispositivo.

7. Construcción del dispositivo

Cuando se diseñó la placa, evidentemente ya se tenía una idea de cómo podría ser el dispositivo final. Pero hasta que no comienza su construcción no se sabe realmente cómo va a ser.

7.1. Condiciones

Para construir el dispositivo hay que tener en cuenta una serie de condiciones que debe tener.

-Se tiene que poder colocar las fibras ópticas de una manera sencilla pero que a la vez las mantenga bien aseguradas durante los tratamientos.

-Se tiene que poder acceder fácilmente a los pulsadores y los interruptores.

-Tiene que entrar en el disco sobre el que va a ir colocado, salvar los dos discos superiores y no tocar a ningún otro elemento de la máquina.

-La pila tiene que ser de fácil sustitución.

-Sería muy interesante idear un sistema para poder abrir la carcasa que constituye el dispositivo por si hay que hacer alguna reparación del mismo, o sustituir la placa por otra, pero a la vez tiene que ser un sistema estanco para que no le afecten los tratamientos.

7.2. Boceto

Lo primero que se hizo fue realizar un simple boceto en tres dimensiones de la forma de la carcasa con la ayuda del programa Google SketchUp. El boceto es el de la Figura 106.

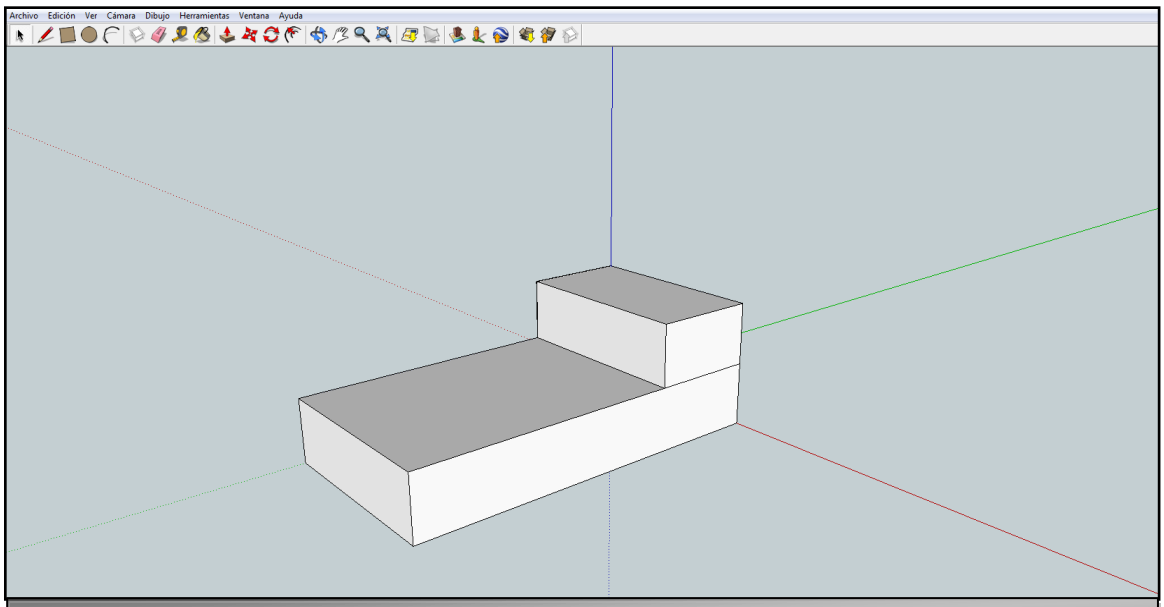


Figura 106: Prototipo carcasa en SketchUp

Las medidas son: 130mm de largo, 65mm de ancho, 22mm en la parte central y 39mm en el otro extremo.

En las Figuras 107 y 108 se aprecia con más detalle la forma que va a tener la carcasa del dispositivo. En la figura 107 también se ve dónde van a ir colocados los tres motores. En la Figura 108 vemos los huecos que se dejan para colocar el conector RJ11, el switch, pulsador y el interruptor.

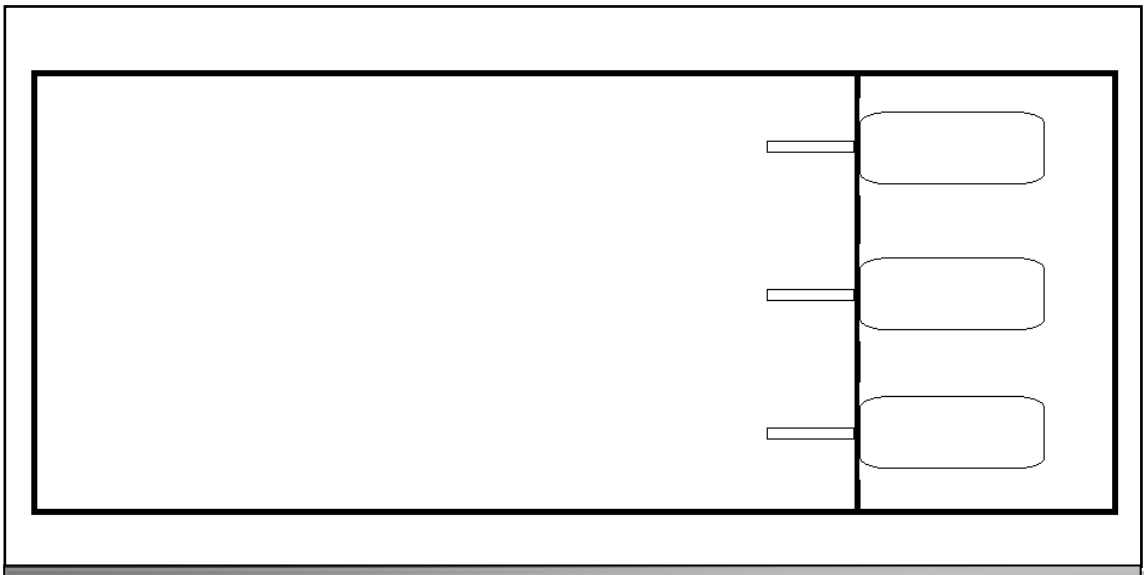


Figura 107: Prototipo carcasa vista superior

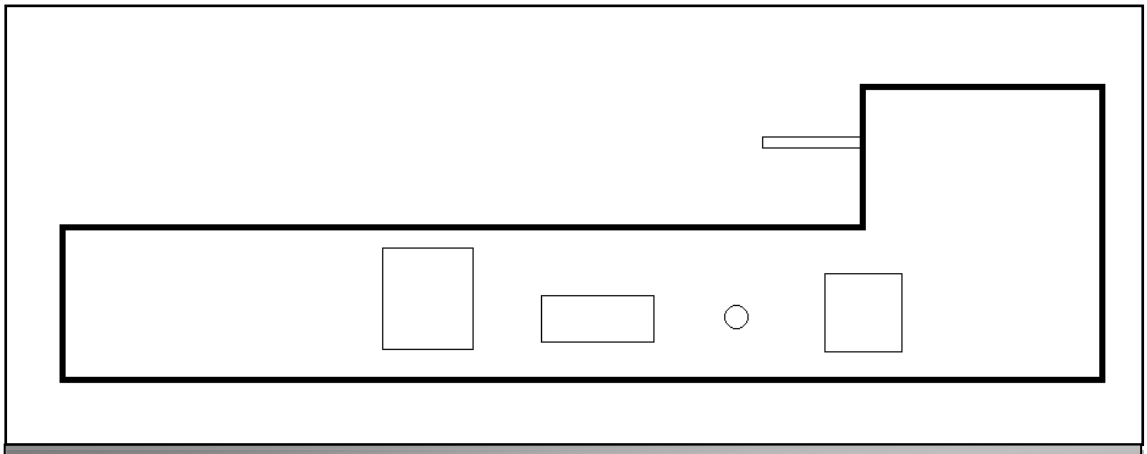


Figura 108: Prototipo carcasa vista la lateral

7.3. Materiales

Después se escogió el material con el que se iba a construir. Primero se buscó algo en el laboratorio que pudiera servir, para ahorrar en presupuesto, pero no se encontró nada que pudiera ser adaptado a las medidas que se necesitaban. Finalmente se encontró en los grandes

almacenes de bricolaje de ***Leroy Merlin*** una placa de aluminio de 0,5mm de grosor. El aluminio es un buen material para esta aplicación, ya que se puede doblar y cortar con facilidad y a la vez es lo suficientemente rígido y estable. Se escogió una placa de 50x25cm por si era necesario construir más de una carcasa.

Figura 109: Imagen Placa Aluminio

Al mismo tiempo que la placa se adquirió un adhesivo para poder pegar las diferentes caras de la carcasa. Se escogió el adhesivo epoxi bicomponente **Araldit** de pegado rápido de la marca ***Ceys*** de 15ml+15ml.

Figura 110: Adhesivo Arladit

Para poder sujetar las fibras se pensó utilizar unas pinzas de tipo cocodrilo, similares a las de los cables para los osciloscopios, pero sin dientes y de menor tamaño.

Figura 111: Pinzas BU-34C

Estas pinzas se van a colocar en los ejes de los motores, y para sujetar la fibra por el otro extremo, simplemente se va a realizar tres oficios en una placa de aluminio y la fibra se apoya sobre estos orificios. De esta manera tiene libertad para girar y no hay riesgo de que se salga. A la hora de escoger estas pinzas y de hacer la carcasa también se tuvo en cuenta que la abertura de la pinza no tocara en la superficie de la carcasa al girar, ni tampoco con la pinza que tenga a su lado.

7.4. Construcción de la carcasa

Después de decidir cómo se iban a asegurar las fibras, había que idear un sistema para poder abrir la carcasa para sustituir la pila y para poder hacer modificaciones en la placa en caso de que fuera necesario.

Se pensó en poder abrir un lateral y utilizar unas gomas con adhesivo que se utilizan por ejemplo en colocación de ventanas, para que a la vez que se encajan en el lateral, hagan presión sobre la carcasa y así se consigue esa estanqueidad y colocando una pestaña se pueda abrir aplicando un poco de fuerza.

Partiendo de esa idea se empezó a cortar y a dar forma a las diferentes partes de la carcasa.

Primero se cortó a medida el lateral donde iban a ir el pulsador, el interruptor, el switch y el conector RJ11. Se dejó una pestaña para poder colocar la parte superior.

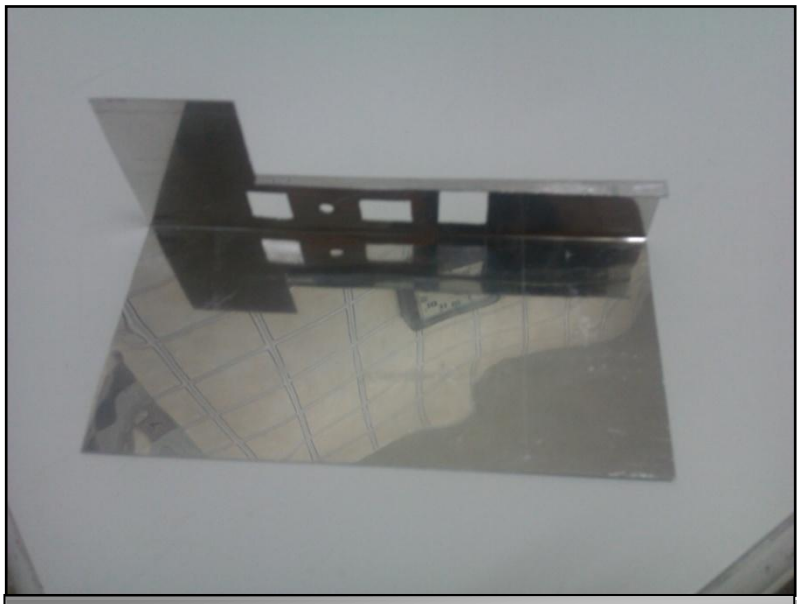


Figura 112: parte inferior carcasa (1)

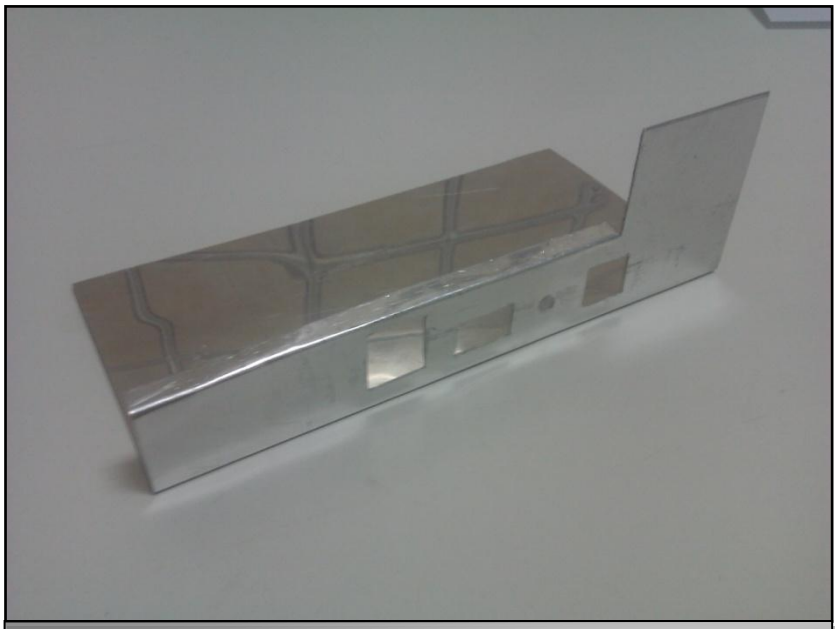


Figura 113: Parte inferior carcasa (2)

Después de realizar esta parte, se procedió a construir la parte superior, que cubriría la placa y parte de los motores. En este parte también se dejó alguna pestaña y se realizaron los orificios por donde se iban a introducir los ejes de los motores (Figuras 114 y 115).

Figura 114: Parte superior carcasa (1)

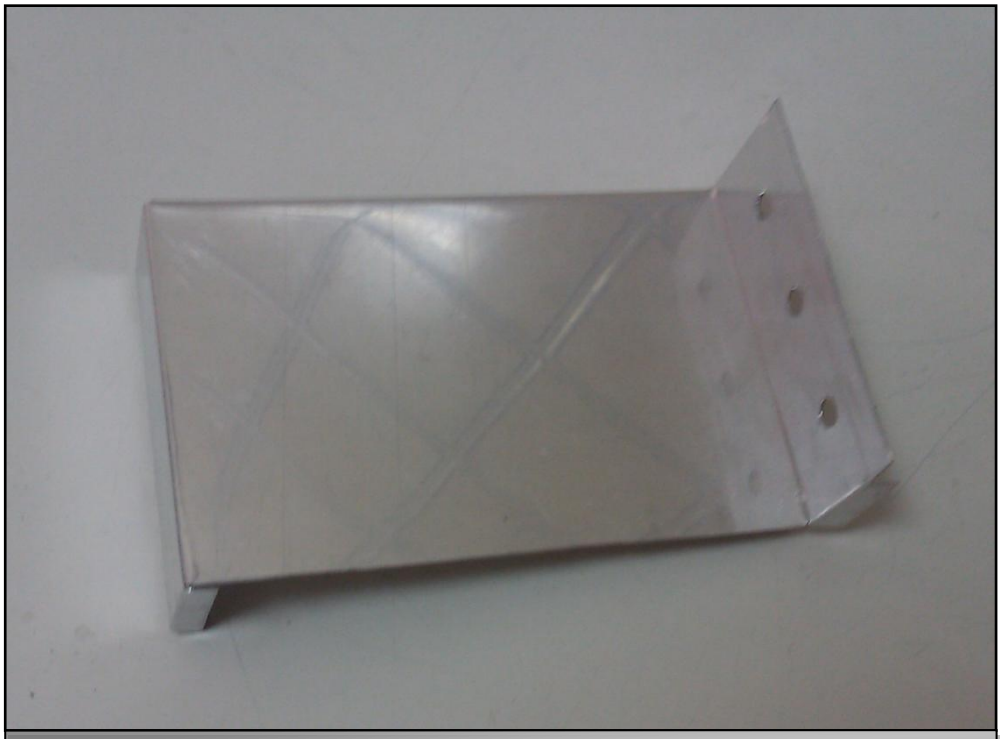


Figura 115: Parte superior carcasa (2)

Después de realizar esta parte se hizo la pieza que iba a cubrir los motores por la parte de arriba y que iba a cerrar el otro extremo del dispositivo (Figuras 116 y 117).

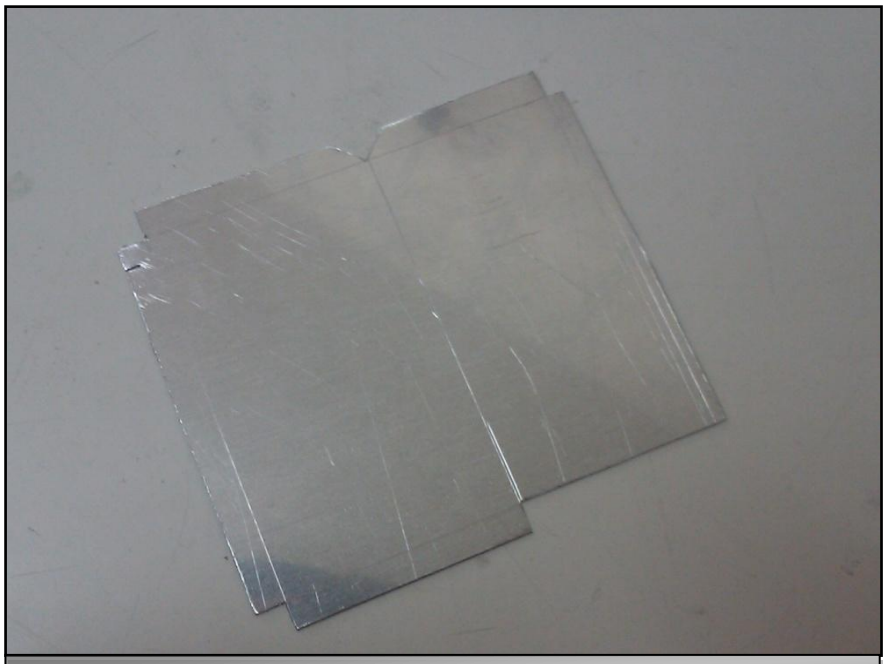


Figura 116: Parte extremo carcasa (1)

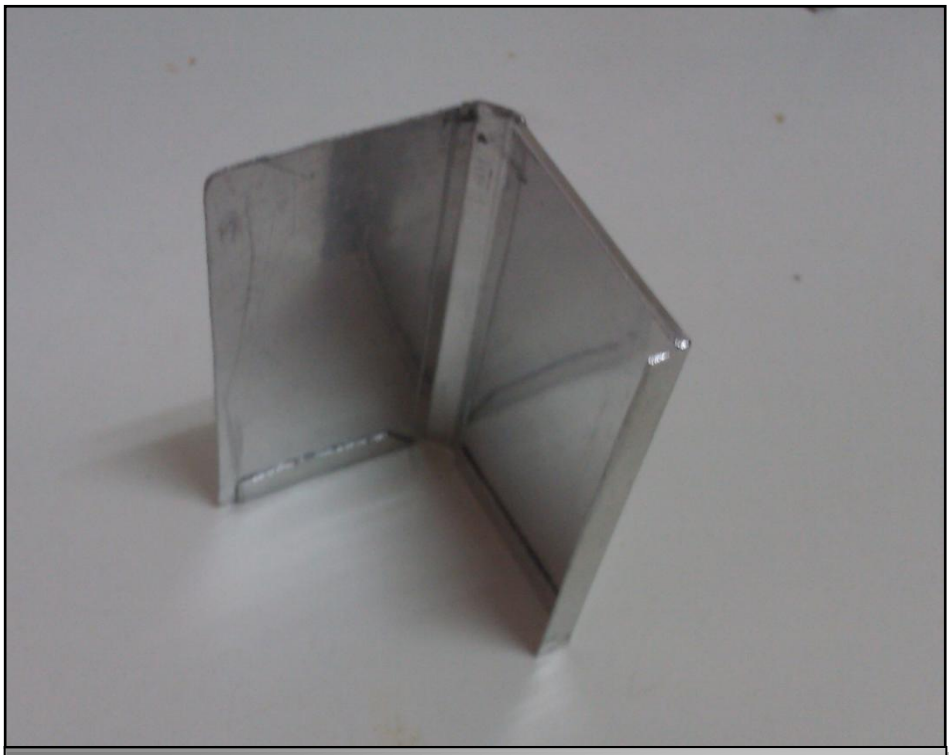


Figura 117: Parte extremo carcasa (2)

Hechas estas tres piezas, se comprobó si encajaban bien montándolas con cinta aislante de dos caras se vio que si estaba bien construida salvo por pequeños retoques en las pestañas.

Una vez montada no dirigimos a la máquina de Sputtering para comprobar si entraba bien en el disco y si tocaba con algún elemento de la propia máquina. Con el disco no hubo problema, pero sí que teníamos dificultades con el tema de la altura, ya que los dos discos más pequeños tocaban con la parte más alta de la carcasa, aunque fuera por unos 2-3 milímetros, pero lo suficiente como para afectar al funcionamiento.

Para solucionar este problema había dos opciones, intentar bajar la altura de la carcasa, cosa algo difícil ya que estaba en la medida justa para que entrara la placa, los motores y que hubiera suficiente espacio para las pinzas. Otra opción era intentar elevar estos dos discos y así poder salvar esa altura. Pero para realizar esto había que tener en cuenta que los discos no se pueden subir todo lo que se quiera, si no que deben guardar una distancia con la tapa superior de la máquina, porque si tocan con esta parte se produce un cortocircuito.

Finalmente se eligió esta opción pero fijándonos en que se elevara lo justo para que entrara el dispositivo y a la vez hubiera la suficiente distancia con la tapa. Se consiguió sin mayores problemas.

Una vez solucionado el problema de la altura, se procedió a realizar el pegado definitivo de las tres piezas, utilizando el adhesivo **Araldit**. Primero se unieron los tres motores a la pieza que tiene los tres orificios. Se fueron pegando uno a uno manteniendo presión unos diez minutos hasta que fraguara el adhesivo y esperando alrededor de una hora entre un motor y otro, para que el pegado fuera consistente, , tal como se recomendaba en las instrucciones.

Una vez pegados los tres motores se procedió a la unión de esta pieza con la base. De la misma manera se mantuvo la presión unos 10 minutos. El resultado se puede ver en las Figuras 118 y 119.

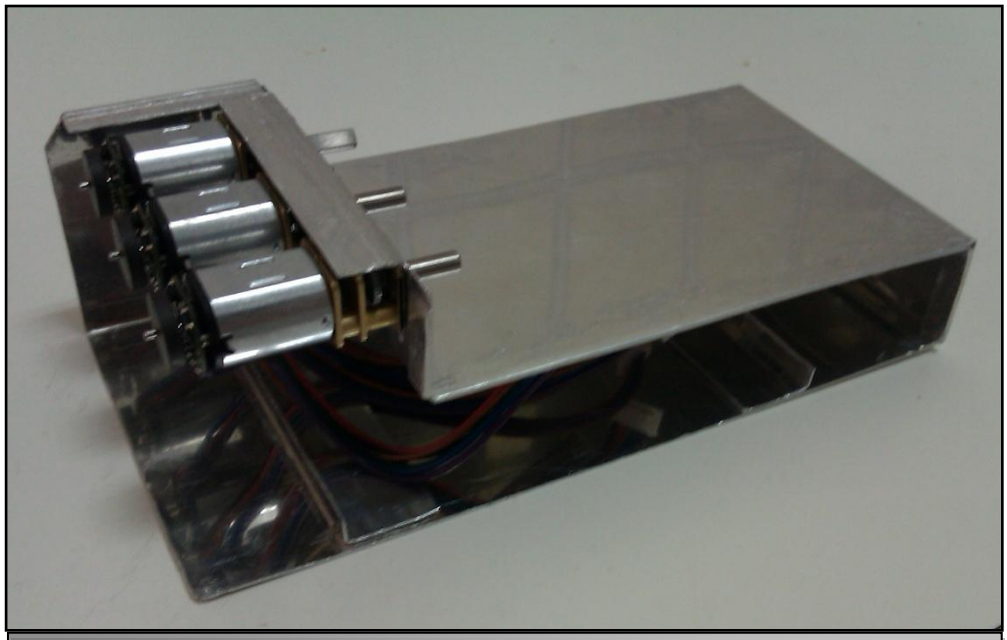


Figura 118: Motores colocados (1)

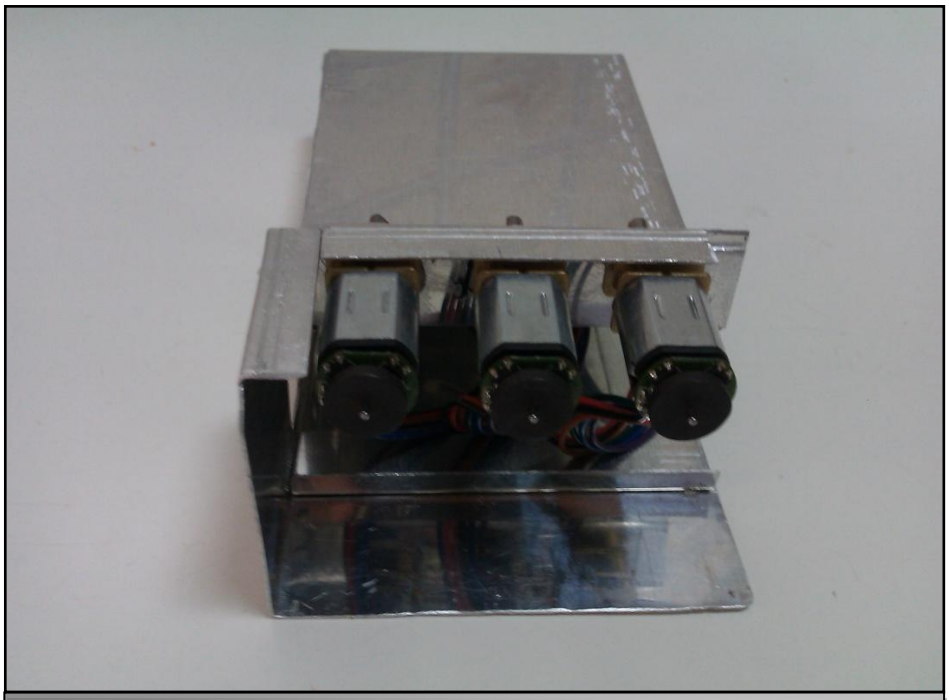


Figura 119: Motores colocados (2)

Después de esperar una hora para que la unión fuera consistente, se realizó la unión de la parte en la que se cubre el otro extremo de la carcasa, como se ve en la Figura 120. En esta figura se puede apreciar también los dos raíles que se colocaron dentro para hacer de guías a la hora de colocar la placa.

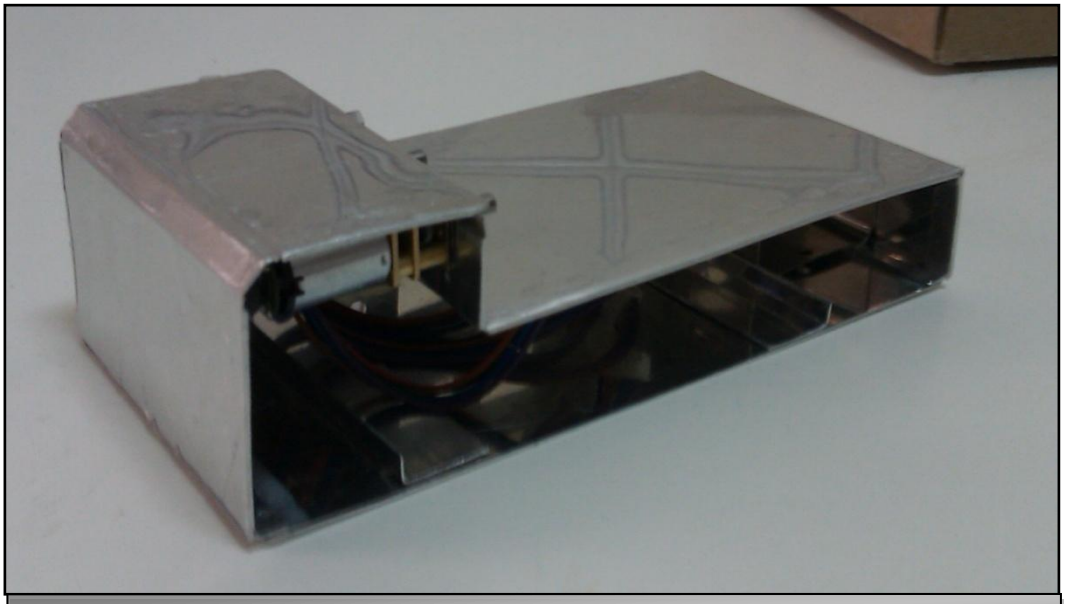


Figura 120: Carcasa del extremo colocada

Después de tener esta parte de la carcasa, se pensó que antes que avanzar en cómo se iban a colocar las fibras, era buen momento para comprobar cómo encajaba la placa y la pila dentro

de ella, Y ver si había algún problema para poder sustituir la pila con facilidad. Para ello se utilizó este soporte para anclar la pila dentro de la carcasa (Figura 121).

Figura 121: Pieza para la pila

Después de colocar estas piezas para la sujeción de la placa y la pila, se probó que encajaba todo bien y también que los motores se podían conectar sin dificultad a la placa.

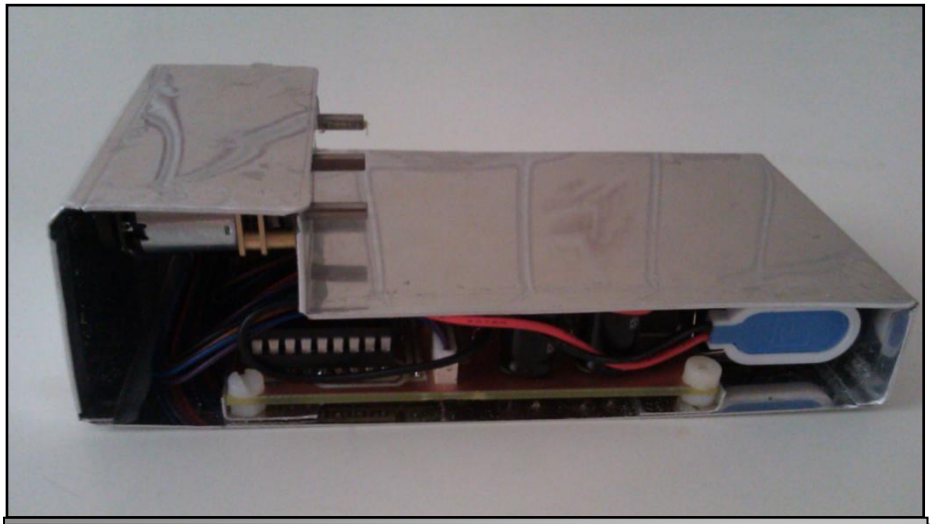


Figura 122: Placa y pila en interior carcasa

Ahora es buen momento para hacer la tapa que cerraría la carcasa. Como se ha explicado anteriormente se pensó en utilizar una goma colocada alrededor de la tapa por medio de pestañas, y que hubiera que hacer presión para encajarla en la carcasa (Figura 23). De esta manera se podría quitar y poner la tapa sin mucha dificultad y a la vez mantener bastante nivel

de estanqueidad. Después de tomar medidas y realizar un par de pruebas se consiguió que encajara bastante bien. El resultado se puede apreciar en las Figuras 123 y 124.

Figura 123: Tapa de la carcasa (1)

Figura 124: Tapa de la carcasa (2)

Una vez hecha la parte de la carcasa queda terminar el dispositivo añadiendo a los ejes de los motores las pinzas y en el otro extremo el sistema para sujetar las fibras.

Primero se colocan las pinzas con el sistema de pegado provisional con la cinta aislante a dos caras. Después se toman medidas para hacer la chapa en el otro extremo para que apoyen las fibras. Los orificios realizados para este objetivo son de 2mm de diámetro.

Figura 125: Placa apoyo fibras provisional

Posiblemente haya que aumentar este diámetro, ya que el propio giro del eje y la colocación de la pinza hacen que al girar, la fibra haga una circunferencia imaginaria del diámetro del eje del motor. Después de ver cómo giran los motores con las pinzas se estudiará el aumentar el diámetro de estos orificios. También se pega a la carcasa de manera provisional, tal y como queda en la Figura 126.

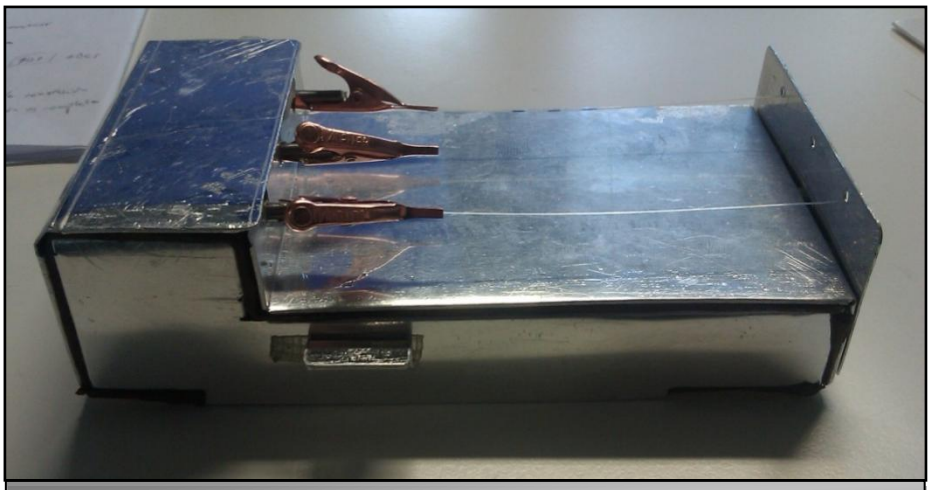


Figura 126: Dispositivo provisional

Para comprobar si el dispositivo se estaba construyendo bien, se realizó un simulacro de tratamiento, con las pinzas colocadas y con fibras ópticas. Se introdujo en la máquina y se hizo un tratamiento pero sin que depositara nada de material. La Figura 127 es una captura de un video del simulacro.

Figura 127: Dispositivo dentro de la máquina

Después de realizar esta prueba se observó que el giro no se realizaba bien porque los orificios eran de poco diámetro y la fibra se combaba, así que se aumentó su diámetro hasta 4mm.

Figura 128: Placa apoyo fibras definitiva

Además se modificó el sistema de agarre de las pinzas con los ejes del motor para poder sacarlas y colocar la fibra más cómodamente y sin forzar el eje del motor.

Finalmente el dispositivo ha quedado como se aprecia en las Figuras 129 y 130.

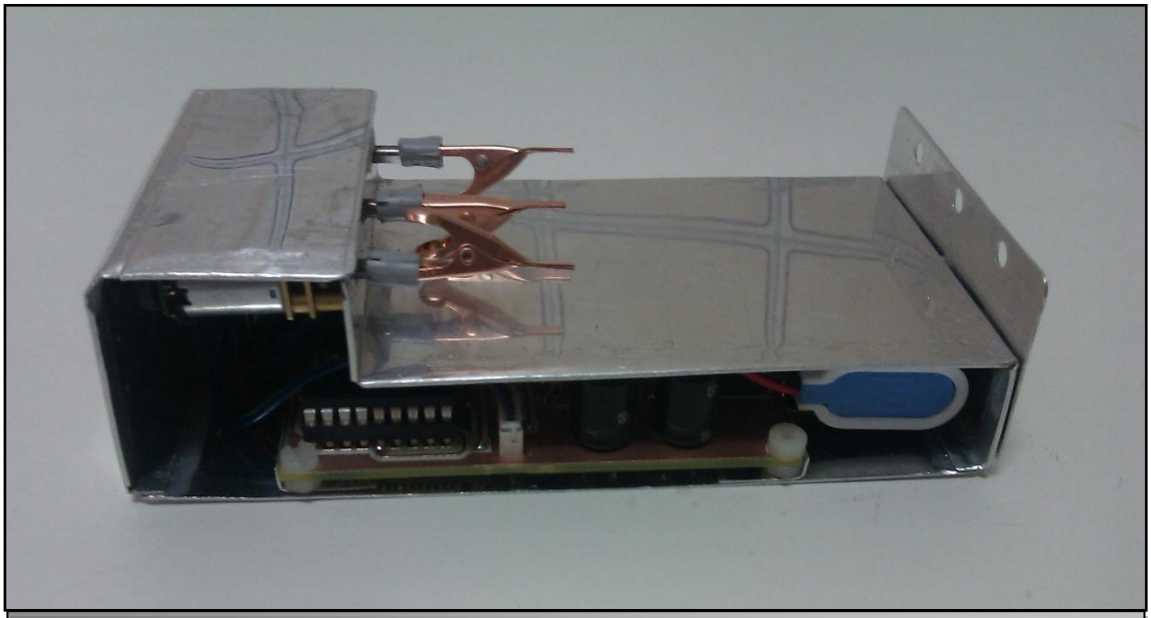


Figura 129: Dispositivo final (1)

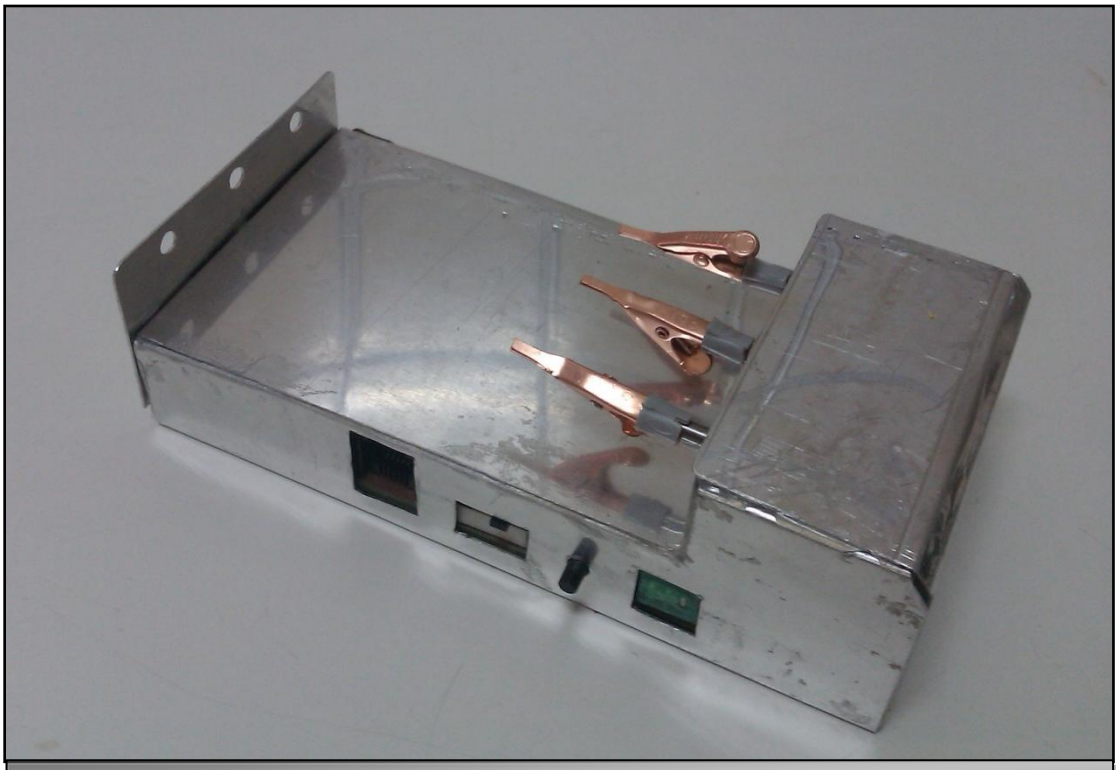


Figura 130: Dispositivo final (2)

8. Futuras mejoras

El dispositivo construido cumple con los requisitos necesarios y con alguna funcionalidad añadida, pero a lo largo de la realización del proyecto han ido surgiendo opciones para hacer el dispositivo mejor, aunque algunas finalmente no se hayan completado.

En este apartado se van a comentar esas opciones que no se han incluido pero que más adelante con más tiempo, dedicación y algunas modificaciones del diseño original si se podrían llegar a incluir.

8.1. Regulación de la velocidad

Ya se ha comentado y explicado la opción de incluir un sistema para que el micro pueda comprobar la velocidad de giro de los motores y modificarla automáticamente. Aunque finalmente no se incluyó en el dispositivo final, la placa tiene varios elementos que se incluyeron en el diseño final por este motivo (encoder, 74LS125A), así que en principio solamente modificando algo de la programación y sustituyendo si es necesario el microcontrolador (poner uno con prioridad de interrupciones), se podría incluir esta posibilidad en el dispositivo. Es la opción más interesante de incluir y la más sencilla ya que en un principio no habría que modificar el diseño de la placa ni del dispositivo.

8.2. Comprobación de la batería

Otra opción interesante de incluir sería una en la que de alguna manera tuviéramos información acerca del estado de la batería. Inicialmente se pensó en utilizar el Convertidor analógico-digital del que disponía el microcontrolador, pero finalmente no se pudo utilizar, como se ha explicado en el apartado ***4.4. Prueba del convertidor A/D***.

Dado que no podíamos utilizar el convertidor, se pensó en diseñar un circuito sencillo que nos diera esta información y añadirlo al diseño de la placa. Buscando por internet se encontró este circuito, Figura 131[46]:

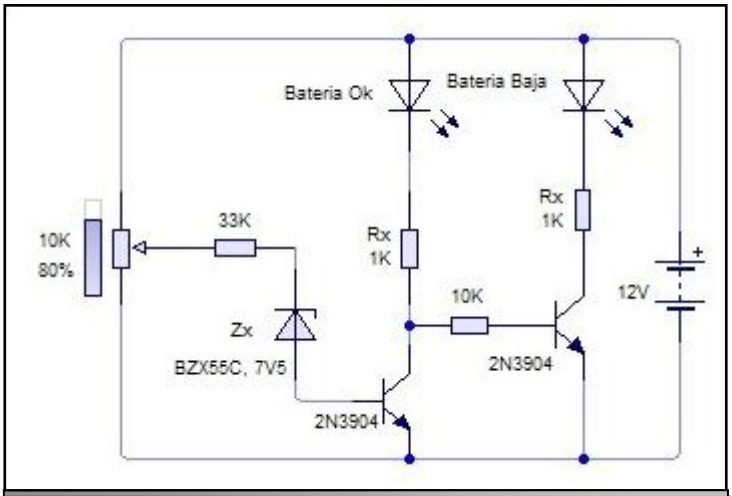


Figura 131: Circuito medidor pila

El funcionamiento del circuito se basa en hacer circular parte de la corriente de la batería bajo prueba, por el circuito del indicador. Un potenciómetro de ajuste de 10K sirve para regular la sensibilidad del aparato. La corriente atraviesa una resistencia y un diodo zener (Zx) para luego llegar a la base del primer transistor. De acuerdo a su intensidad, bastará o no para encender el primer led (verde), que es el que indica que la batería bajo prueba está cargada. El segundo led (rojo), comandado por el segundo transistor, cumple la función opuesta: si se enciende, nuestra batería debe ser recargada [52].

Es un circuito sencillo que puede que no ocupe mucho espacio en la placa. Se realizó una prueba en la placa protoboard y se comprobó que no funcionaba del todo bien, si que variaba la intensidad de los leds, pero estaban los dos permanentemente encendidos. Esto podía ser debido a que los operacionales no son los mismos que en el circuito y el diodo zener tampoco. Por falta de tiempo no se pudieron hacer más pruebas y se tuvo que aparcar esta opción para poder seguir avanzando con el proyecto y con los objetivos marcados.

Con más tiempo, consiguiendo los componentes adecuados y haciendo algunas pruebas más, es bastante probable que se pudiera conseguir y por lo menos tener un indicador de que la pila está a punto de agotarse, y sustituirla por otra antes de empezar a realizar otro tratamiento.

8.3. Modificación de la tapa

Como se ha explicado en el apartado ***7. Construcción del dispositivo***, para poder sustituir la pila y realizar posibles arreglos o modificaciones se ha incluido en el diseño del dispositivo una tapa en uno de los laterales. Esta tapa consta de un sistema de gomas que se colocan por la cara interna de la carcasa. Dado que han quedado unos mínimos huecos con este sistema, sería buena idea realizar otra tapa igual, pero de manera que las gomas encajen por la cara externa de la carcasa, de esta manera nos aseguramos que toda la sustancia que precipita sobre el dispositivo quede en la parte externa y no afecte en ningún momento a algún elemento interno.

9. Presupuesto

Para el diseño y elaboración del presente proyecto se han tenido en cuenta los honorarios del ingeniero y los elementos utilizados para construir el dispositivo.

9.1. Coste de mano de obra

Este coste se obtiene de la suma del salario del ingeniero y las obligaciones sociales. Para ello, se ha consultado la página web de la seguridad social:

http://www.seg-social.es/Internet_1/Trabajadores/CotizacionRecaudaci10777/Basesytiposdecotiza36537/index.htm

9.1.1. Salario

Para calcular el salario del ingeniero, se observa que pertenece al grupo 2 de cotización, con una base mínima de 867,00 €/mes y una base máxima de 3.262,50 €/mes.

Grupo de Cotización	Categorías Profesiones	Bases mínimas euros/mes	Bases máximas euros /mes
1	Ingenieros y Licenciados.Personal de alta dirección no incluido en el artículo 1.3.c) del Estatuto de los Trabajadores	1.045,20	3.262,50
2	Ingenieros Técnicos, Peritos y Ayudantes Titulados	867,00	3.262,50
3	Jefes Administrativos y de Taller	754,20	3.262,50
4	Ayudantes no Titulados	748,20	3.262,50
5	Oficiales Administrativos	748,20	3.262,50
6	Subalternos	748,20	3.262,50
7	Auxiliares Administrativos	748,20	3.262,50
		Bases mínimas euros/día	Bases máximas euros /día
8	Oficiales de primera y segunda	24,94	108,75
9	Oficiales de tercera y Especialistas	24,94	108,75
10	Peones	24,94	108,75
11	Trabajadores menores de dieciocho años, cualquiera que sea su categoría profesional	24,94	108,75

Figura 132: Bases de cotización contingencias comunes de la seguridad social

Se ha elegido la media de estas dos bases, siendo esta 2.064,75 €/mes. O lo que es lo mismo, 66,60 €/día.

	DIAS TRABAJADOS	SALARIO POR DIA	SUELDO TOTAL
INGENIERO	140	66,60 €/día	9.324€

Figura 133: Tabla salario

9.1.2. Obligaciones sociales

	EMPRESA	TRABAJADORES	TOTAL
Contingencias comunes	23,6	4,7	28,3 %
Desempleo (tipo general)	5,5	1,55	7,05 %
Fondo Garantía Salarial (FOGASA)	0,2	-	0,2 %
Formación profesional	0,6	0,10	0,7 %
Accidentes	-	-	0,99 %
TOTAL	**37,24 %**		

Figura 134: Tabla obligaciones sociales

$$Obligaciones\ sociales = 37{,}24\% \cdot SALARIO\ INGENIERO$$

$$Obligaciones\ sociales = 37{,}24\% \cdot 9324€ = 3.472{,}25\ €$$

9.1.3. Coste total mano de obra

$$COSTE\ TOTAL = SALARIO\ INGENIERO + OBLIGACIONES\ SOCIALES$$

$$COSTE\ TOTAL = 9324\ € + 3.472{,}25\ € = \mathbf{12.796{,}25}\ €$$

Por lo tanto, el coste total de la mano de obra asciende a DOCE MIL SETECIENTOS NOVENTA Y SEIS euros CON VEINTICINCO céntimos de euro.

9.2. Coste de materiales

Todos los materiales se han conseguido en **RS Components y Mouser Electronics** , a excepción de la placa y el adhesivo que se adquirieron en **Leroy Merlin**.

Nombre	Tipo	Fabricante	Cantidad	Precio Unitario	Total
Conector RJ11	RJ11	Wurth Elecktronik	1	0,76 €	0,76 €
Conector Motor	JST ZHR-6 1,5mm	JST	3	0,12 €	0,36
Interruptor ON/OFF	25339NA	Apem	1	3,38 €	3,38 €
Regulador tensión	LM2940CA-5.0	Texas Instruments	1	1,70 €	1,70 €
Adaptador señal	SN74LS125AN	Texas Instruments	1	0,81 €	0,81 €
Driver	L293D	Texas Instruments	1	2,84 €	2,84 €
Microcontrolador	PIC16F88	Microchip	1	2,5 €	2,5 €
Conector pila	Conector "I" Style	KeyStone	1	0,77 €	0,77 €
Pila 9V	Recargable NI MH	ANNSMANN	2	14,44 €	28,88 €
Switch	206-RAST	CTS Electrocomponents	1	0,96 €	0,96 €
Pulsador	MJTP1236B	Apem	1	0,17 €	0,17 €
Motor	GM12-N20VA-EN	TTMotor	3	10 €	30 €
Diodo	1N4007	Genérico	1	0,02 €	0,02 €
Resistencias	R1:4k7 R2:470 R3:10K R4:470	Genérico	4	0,01 €	0,04 €
Condensador cerámico	100nF 50V	Genérico	6	0,14 €	0,84 €
Condensador electrolítico	220uF 50V	Panasonic	2	0,51 €	1,02
Placa aluminio	Bruto Pulido fino 0,5mm	Leroy Merlin	1	6,05 €	6,05 €
Adhesivo	Araldit rápido 15+15	Ceys	1	7,30 €	7,30 €
Pinza	BU-34C	Mueller	3	0,61 €	1,83 €
TOTAL					**90,23 €**

Figura 135: Tabla coste de materiales

9.3. Gastos generales

Son los concernientes al equipamiento empleado para medidas y pruebas en concepto de amortización de los equipos y gastos de energía. Se han estimado como el 10% de los apartados 8.1. y 8.2.

Coste mano de obra: 12.796,25 * 0,10 = **1279,62 €**

Coste materiales : 90,23 * 0,10 = **9,02 €**

Total Gastos generales = 1288,64 €

9.4. Importe total del proyecto

El coste total del proyecto es el siguiente:

Coste mano de obra.....12.796,25 €

Coste materiales.....90,63 €

Costes generales.....1.288,64 €

Subtotal.....14.175,52 €

I.V.A (21%).....2.976,85 €

TOTAL.....17.152,37 €

El importe total del proyecto realizado asciende a un total de DIECISIETE MIL CIENTO CINCUENTA Y DOS euros y TREINTA Y SIETE céntimos de euro.

10. Conclusiones

El objetivo con el que se inició este proyecto fue con la idea de realizar un proyecto de algo distinto a lo que se suele realizar en la carrera de ITI Eléctrica (instalación eléctrica de viviendas o naves industriales), y relacionado con los microcontroladores y con la electrónica analógica para aprender más acerca de estos dos temas, que son los que más me llaman de mi carrera. Personalmente creo que se han conseguido bastante bien los objetivos marcados y he podido aprender bastante sobre estos temas.

Para conseguirlo se han tratado varias cuestiones:

- Documentación: se han tenido que leer numerosas hojas de especificaciones de los componentes tanto en castellano como en inglés.

- Diseños electrónicos: Se ha aprendido a manejar totalmente un programa de creación de placas PCB desde cero.

- Programación de microcontroladores: se ha aprendido a programar mejor un micro y a aprovechar más las posibilidades del software MPLAB.

- Solución de problemas: se ha tenido que encontrar soluciones a los problemas que han ido surgiendo y que no estaban previstos.

BIBLIOGRAFÍA

RECURSOS ELECTRÓNICOS

[1] Imagen PICDEM 2PLUS

http://www.tme.eu/es/details/picdem2-plus/juegos-de-arranque-para-circ-microchip/microchip-technology/dm163022/#

[2] Imagen ICD2

http://www.automation-drive.com/icd-microchip

[3] PRECIO DEL MICROCONTROLADOR (TABLA)

http://www.microchip.com/ParamChartSearch/chart.aspx?branchID=1024&mid=10&lang=en&pageId=74

[4] IMAGEN FUNCIONAMIENTO PWM

http://meteoyelectronica.blogspot.com.es/2010/12/pwm-una-manera-sencilla-de-controlar-un.html

[5] IMAGENES SERVOS 1

http://robots-argentina.com.ar/MotorServo_basico.htm

[6] IMAGEN DE LOS SERVOS 2

http://www.info-ab.uclm.es/labelec/solar/electronica/elementos/servomotor.htm

[7] EXPLICACIÓN SERVOS

http://usuarios.multimania.es/koldolo/servos.htm

[8] IMAGEN FUNCIONAMIENTO SERVO ANALÓGICO

http://picmania.garcia-cuervo.net/picc2.php

[9] IMAGEN FUNCIONAMIENTO SERVO DIGITAL 2

http://www.futaba-rc.com/servos/digitalservos.pdf

IMAGENES DE LOS MOTORES

[10] http://www.goodluckbuy.com/power-hd-high-speed-micro-analog-servo-8g-hd-1800a-.htmL

[11] http://www.electan.com/power-mini-servo-hd1160a-p-2972-en.html

[12] http://articulo.mercadolibre.com.ar/MLA-431286517-micro-servo-digital-power-hd-ds65hb-65g-15kg-007sec-_JM

[13] http://www.bricogeek.com/shop/113-motor-micro-metal-dc-con-reductora-10-1.html

[14] https://catalog.precisionmicrodrives.com/order-parts/products/dc-motor

[15] http://www.directindustry.es/prod/motion-drivetronics-pvt-ltd/motores-electricos-dc-miniatura-55237-397383.html

[16] http://www.mitsumi.co.jp/latest/Catalog/compo/motorav/m10e2series_photo.html

DATOS DE LOS MOTORES

[17] http://www.pololu.com/catalog/product/1049

[18] http://www.pololu.com/catalog/product/1046

[19] http://www.pololu.com/catalog/product/1051/faqs

[20] http://www.pololu.com/catalog/product/1094

[21] https://catalog.precisionmicrodrives.com/order-parts/product/124-002-24mm-dc-motor-8mm-type

[22] http://www.mechtex.com/geared_motors.html

[23] http://www.mitsumi.co.jp/latest/Catalog/compo/motorav/m10e2series_e.html

[24] DATOS MOTOR ELEGIDO E IMAGEN

http://www.ttmotor.com/Product.asp?ID=848

[25] IMAGEN PILA

http://es.rs-online.com/web/p/pilas-recargables-9-voltios/7033524/

[26] IMAGEN DESCRIPTIVA TRIGGER SCHMIDTT

http://www.acroname.com/robotics/info/ideas/omnispeed/omnispeed.html

[27] IMAGEN CONDENSADOR

http://es.rs-online.com/web/p/condensadores-de-aluminio/3654290/

[28] IMAGEN CONECTOR RJ11

http://es.rs-online.com/web/p/conectores-rj11/7350282/

IMAGENES SWITCH

[29] http://es.rs-online.com/web/p/interruptores-dip-y-sip/7023322/

[30] http://www.digikey.com/product-detail/en/206-3RAST/CT2063RAST-ND/92915

IMAGEN PULSADOR

[31]http://es.mouser.com/ProductDetail/Apem/MJTP1236A/?qs=Uv2m1HzDAyqrSXyUnCBIpg==

[32] http://www.digikey.com/product-detail/en/MJTP1236B/679-2446-ND/1798056

IMAGENES INTERRUPTORES

[33] http://www.abcelectronica.net/productos/interruptores/rocker/

[34] http://elcodis.com/parts/2481607/25339NLD_p3.html

[35] IMAGEN ESQUEMA PROVISIONAL

http://www.robotroom.com/PWM5.html

[36] IMAGEN PROTEUS

http://elektronisc.blogspot.com.es/2012/08/progarmas.html

[37] IMAGEN DESIGNSPARK

http://mx.rsdelivers.com/campaigns/designsparkpcb/designsparkpcb.aspx

[38] IMAGEN ARALDIT

http://perfumerialavirgen.es/index.php/ceys/ceys-araldit-rapido-5-minutos-1.html

[39] IMAGEN PINZA

http://www.digikey.com/product-detail/en/BU-34/314-1017-ND/304581

[40] EXPLICACIÓN SERVOS

http://robots-argentina.com.ar/MotorServo_basico.htm

[41] SERVOS DIGITALES Y ANALOGICOS

http://usuarios.multimania.es/koldolo/servos.htm

[42] PÁGINA DE MICROCHIP

http://www.microchip.com/ParamChartSearch/chart.aspx?branchID=1024&mid=10&lang=en&pageId=74

[43] MANUAL BÁSICO PROTEUS

http://es.scribd.com/doc/25217993/Manual-basico-PROTEUS-ISIS

EXPLICACIÓN MICROCONTROLADORES

[44] http://perso.wanadoo.es/pictob/micropic.htm

[45] http://www.slideshare.net/andresmg/la-familia-de-microcontroladores-pic-3361548

[46] CIRCUITO FUTURAS MEJORAS

http://www.neoteo.com/construya-un-indicador-del-estado-de-carga-de-sus

[47] IMAGEN DIODO

http://baudaeletronica.blogspot.com.es/2011/09/1n4007.html

[48] EXPLICACIÓN BASICA DESIGNSPARK PCB

http://www.neoteo.com/designspark-pcb-facil-gratis

[49] PWM DEL MICROCONTROLADOR

http://es.scribd.com/doc/45642711/17/CONTROL-DE-VELOCIDAD-DE-UN-MOTOR-DC-CON-PWM

[50] INTERRUPCIONES Y TEMPORIZACIONES

http://www.slideshare.net/lmzurita/microcontroladores-ii-en-c-tema-2

[51] FUNCIONAMIENTO DRIVER L293D

http://robots-argentina.com.ar/MotorCC_L293D.htm

[52] CIRCUITO MEDIDOR DE BATERÍA

http://www.electronicafacil.net/circuitos/Indicador-de-estado-para-baterias.html

[53] FRECUECNCIA PWM

http://www.todopic.com.ar/foros/index.php?topic=19866.0

RECURSOS BIBLIOGRÁFICOS

[54] **MICROCONTROLADORES PIC. DISEÑO PRÁCTICO DE APLICACIONES 1ª parte. PIC12F508 y PIC16F84A. Lenguajes Ensamblador, C y PBASIC.** Autores: José Mª Angulo Usategui, Ignacio Angulo Martínez y Aritza Etxebarría Ruiz. 4ªEdición. McGrawHill.

[55**] Microcontroladores PIC. Teoría y Práctica. El PIC 16F88X y el Laboratorio USB-PIC's School** Mikel Etxebarria Isuskiza. Creaciones CopyRight.

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS

INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO TÉCNICO INDUSTRIAL ELÉCTRICO

Título del proyecto:

"DISPOSITIVO ROTATORIO PORTÁTIL INDEPENDIENTE DE LA RED ELÉCTRICA PARA EL ACOPLE DE HILOS DE FIBRA ÓPTICA"

ANEXO I: “MANUAL DESIGNSPARK PCB”

Alumno: Asier Miranda Herrero

Tutor: Dr. Carlos Ruiz Zamarreño

Pamplona, a 23 de abril de 2013

ANEXO I: MANUAL DESIGNSPARK PCB

ÍNDICE

INDICE DE FIGURAS

1. Introducción

DesignSpark PCB es un software profesional de diseño de placas PCB (Printed Circuit Board) gratuito. La mayoría de este tipo de software es de pago y de difícil acceso para estudiantes y personas que está empezando a realizar sus primeros diseños de placas electrónicas. Las versiones gratuitas de estos programas de pago tienen limitaciones de funciones y de cantidad de componentes haciendo su uso inviable para la mayoría de los circuitos típicos. DesignSpark PCB no tiene ninguna limitación, es 100% funcional y además le acompaña una comunidad de usuarios donde poder compartir esquemas y componentes. Es una de las mejores opciones, (si no la mejor) de realizar placas PCB.

1.1. Instalación, registro y activación

Para disponer de este software lo primero que hay que hacer es visitar la página web oficial: www.designspark.com, descargarse el fichero de instalación y ejecutarlo.

Figura 2: Página oficial de DesignSpark

La instalación es como la de cualquier otro programa, seguir el asistente de instalación y aceptar términos y condiciones. Una vez instalado, la primera vez que se ejecute el programa, dará un código de licencia y pedirá que el usuario se registre y active el producto. Esto es bastante importante ya que solamente con el registro gratuito se puede acceder al foro y compartir con otros usuarios esquemas y librerías de componentes.

La activación es muy sencilla: una vez registrado en la página oficial del programa, proceso que no dura más de dos minutos, se copia el código de licencia que se ha proporcionado previamente y se tendrá que introducir en la cuenta de usuario de DesignSpark PCB y éste mandará al instante un código de activación al correo electrónico. Una vez introducido este código en el programa ya se tiene 100% funcional el software y se podrá utilizar el programa y también acceder a numerosas librerías y esquemas disponibles en el foro de la página. Esta es una de las grandes ventajas de este programa.

Se pueden realizar 6 activaciones del programa con una sola cuenta de usuario.

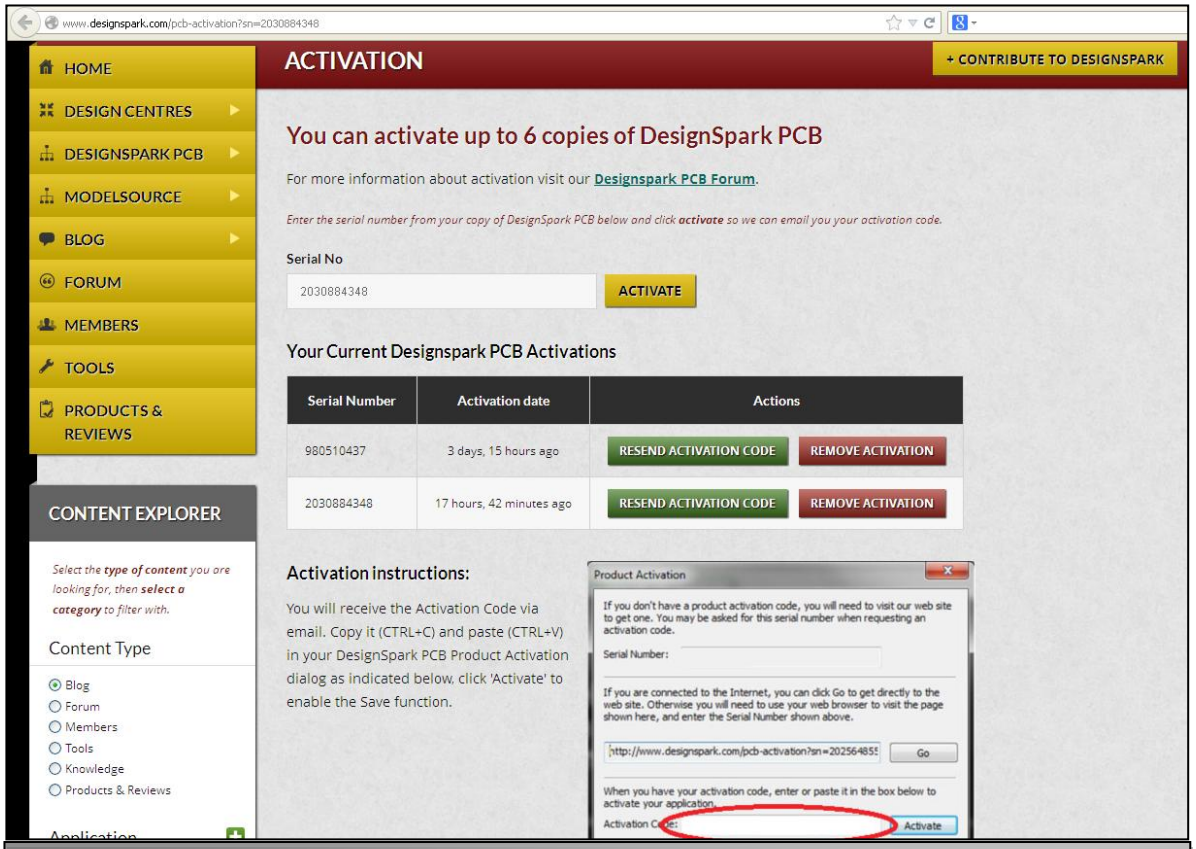


Figura 2: Página de activación [1]

1.2. Primeros pasos

Una vez hecho todo el proceso de activación, la primera pantalla que saldrá es la de un ejemplo de diseño de una placa. Se cierra el ejemplo y se irá a la barra de menú, ***File -> Nuevo***, y en la pantalla aparecerán diferentes opciones y seleccionará la de ***Schematic Design*** como se puede ver en la Figura 3, se pulsa en *OK* y se obtiene el escritorio donde se va a realizar el esquema (Figura 4).

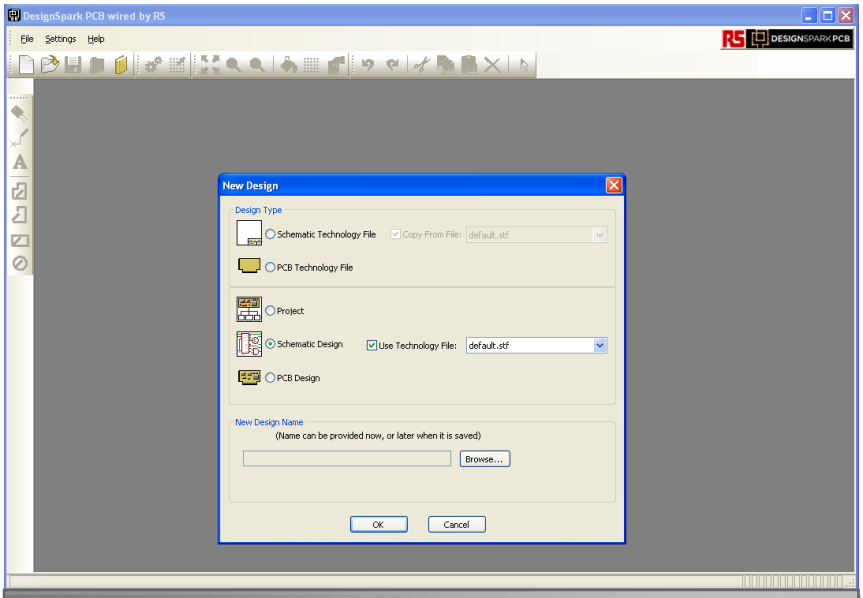


Figura 3: Ventana de nuevo diseño

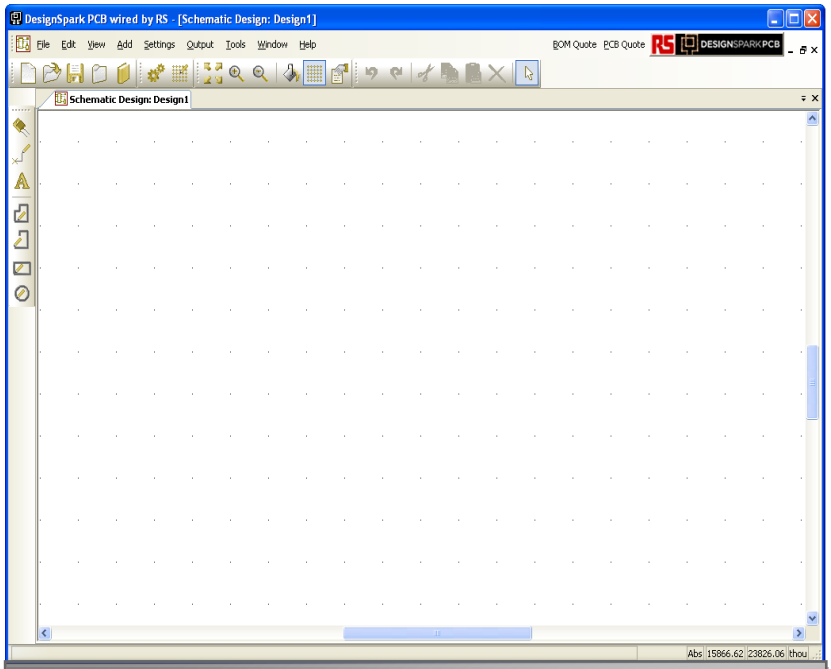


Figura 4: Escritorio para realizar el nuevo esquema

Una vez que se tiene el escritorio de trabajo se van a presentar las dos barras de herramientas que se utilizan.

En la parte superior se tiene la Barra de Herramientas General (Figura 5).

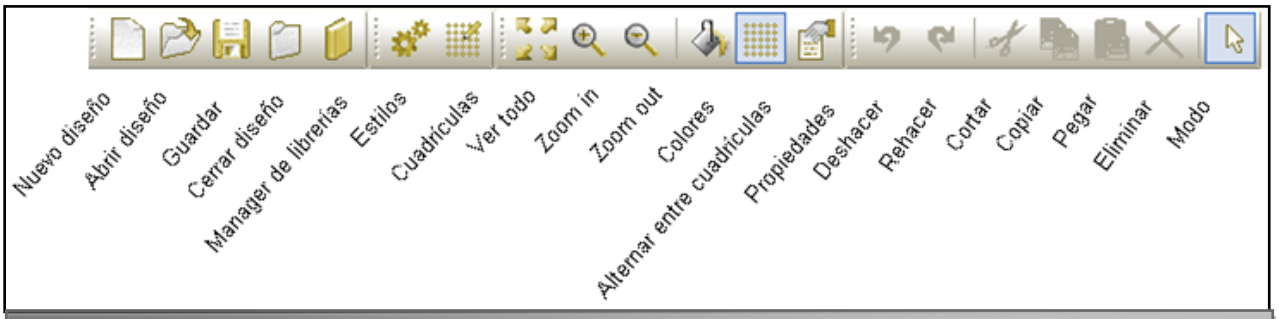


Figura 5: Barra de Herramientas General

En el lateral izquierdo se ve la Barra de Herramientas Lateral (Figura 6).

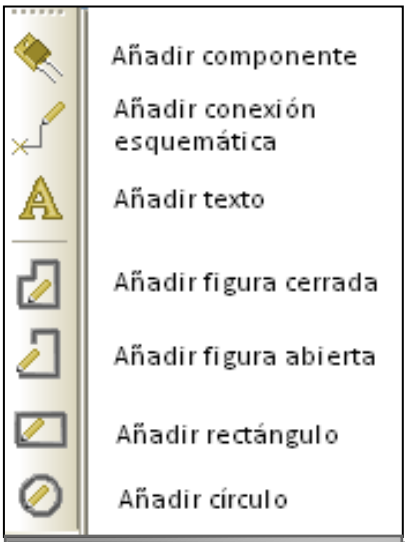


Figura 6: Barra lateral

2. Añadir componentes

2.1. Activar librerías

Antes de añadir ningún componente hay que activar todas las librerías que se tienen disponibles. Para ello hay que ir a la barra de tareas superior pinchar en el icono de

Manager de librerías y aparecerá la siguiente ventana:

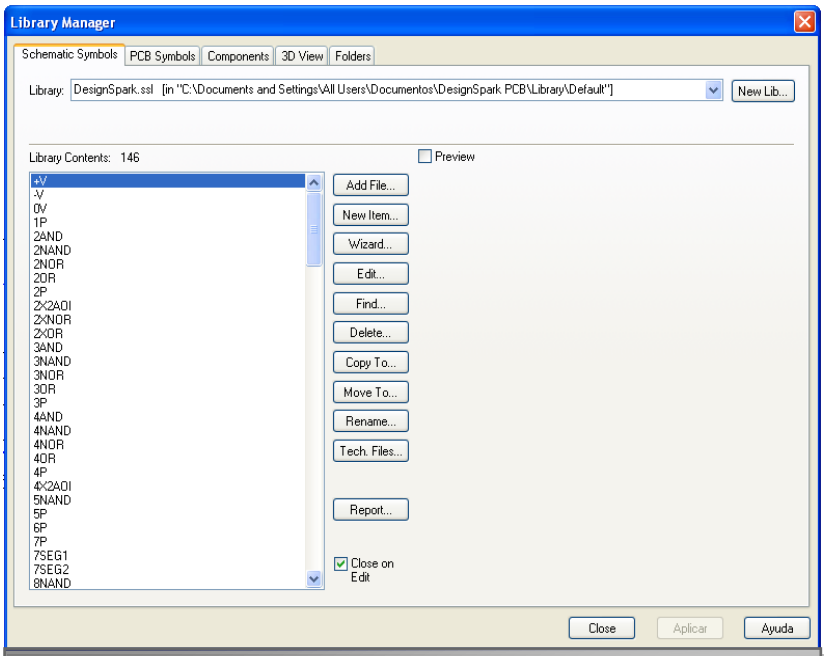


Figura 7: Imagen inicial del Manager de Librerías

Ahora se selecciona la pestaña ***Folders***:

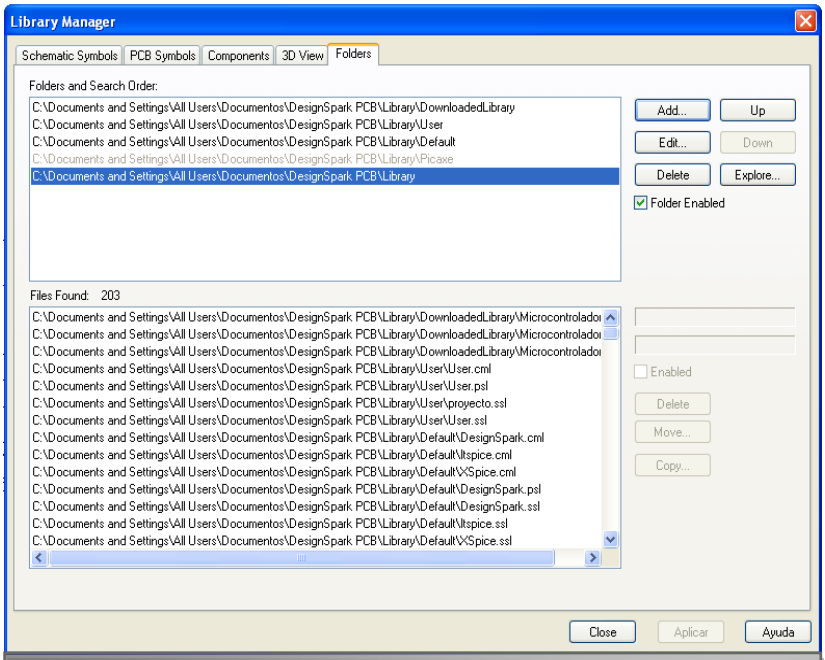


Figura 8: Barra de Herramientas General

Para disponer de las librerías hay que ir al cuadro superior y activar las librerías que se deseen con la pestaña de la derecha ***Folder Enabled***. En concreto hay que activar la que sale marcada en la imagen para tener disponibles componentes de empresas como Microchip, Fairchild, Philips, Intel, etc., ya que ésta no sale activada de serie y es la que más componentes tiene. Una vez activadas las librerías ya se puede acceder a un gran número de componentes.

Para añadir uno de estos componentes al esquema se tienen varias opciones que se explican a continuación.

2.2.1. Icono de Barra lateral

Pinchando el icono de  ***Añadir componente*** de la barra de herramientas lateral, aparece un cuadro donde podemos seleccionar entre las diferentes librerías y dentro de éstas buscar el componente que se necesite.

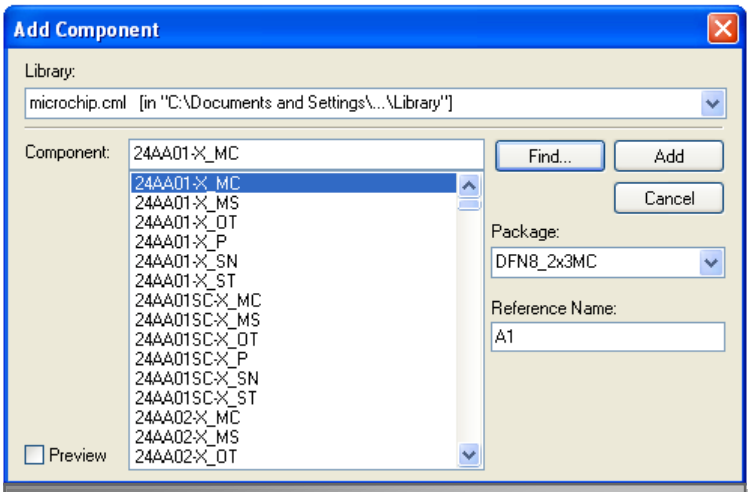


Figura 9: Ventana de añadir componente

Es interesante activar la pestaña ***Preview*** que permitirá tener una vista previa del componente, tanto del símbolo de esquema como del encapsulado PCB.

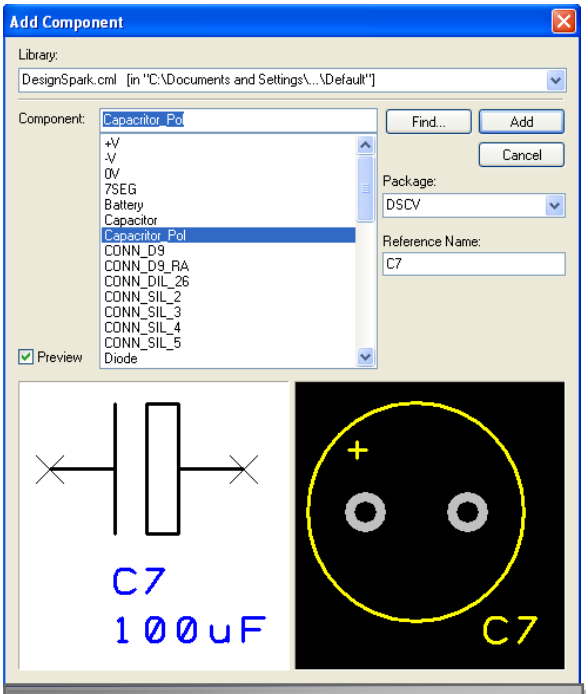


Figura 10: Componente + previsualizado

Si se quiere un componente muy concreto, existe la posibilidad de buscarlo pinchando en ***Find*** y se verá otro cuadro con diferentes criterios como el nombre, número de pins… De esta forma, se encuentra el componente.

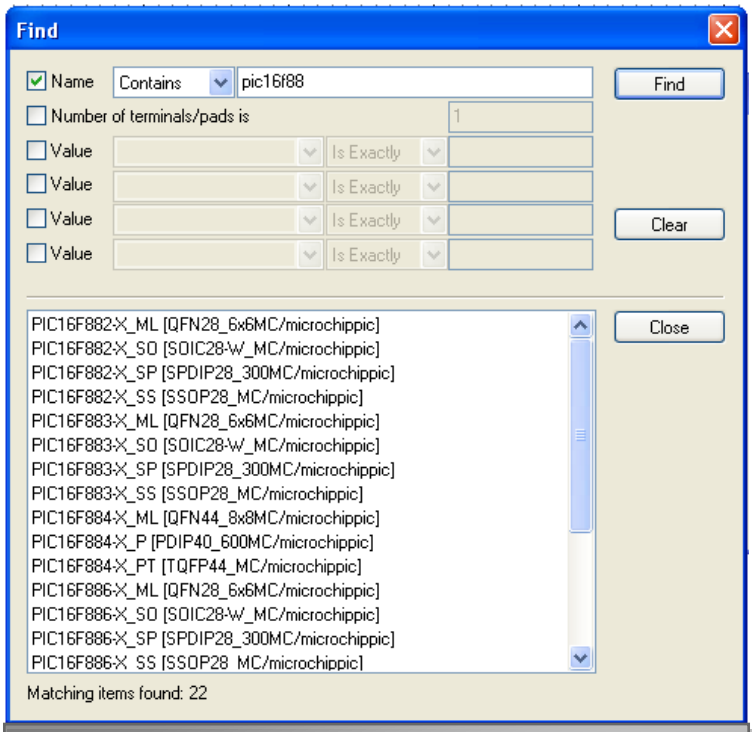


Figura 11: Buscar componente

Una vez encontrado el componente se clica dos veces encima de él y ya está disponible para colocarlo dentro del esquema donde se quiera.

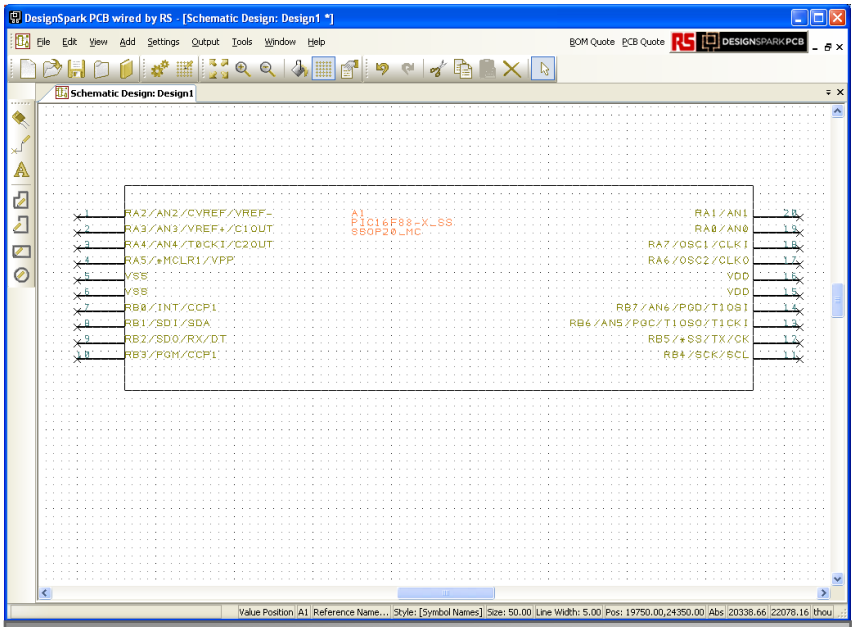


Figura 12: Componente añadido al esquema

2.2.2. Interaction Bar

La segunda opción de añadir componentes es pinchando en ***View -> Interaction Bar*** de la barra de menú. Se añadirá una barra de herramientas a la derecha de la pantalla (Figuras 13 y 14).

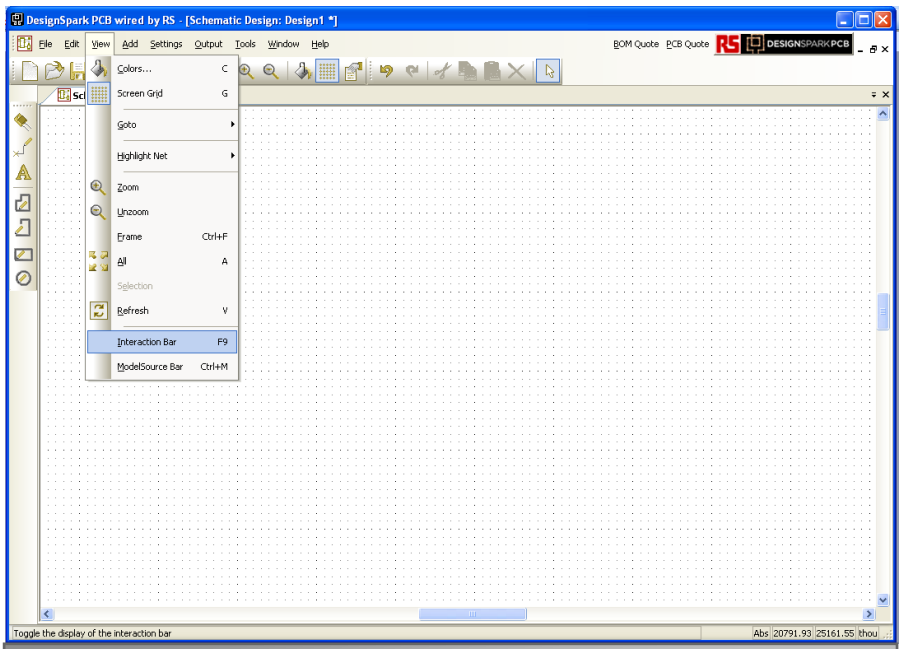


Figura 13: Añadir "Interaction Bar"

Se clica abajo a la derecha de esta barra, donde pone ***Add Component***.

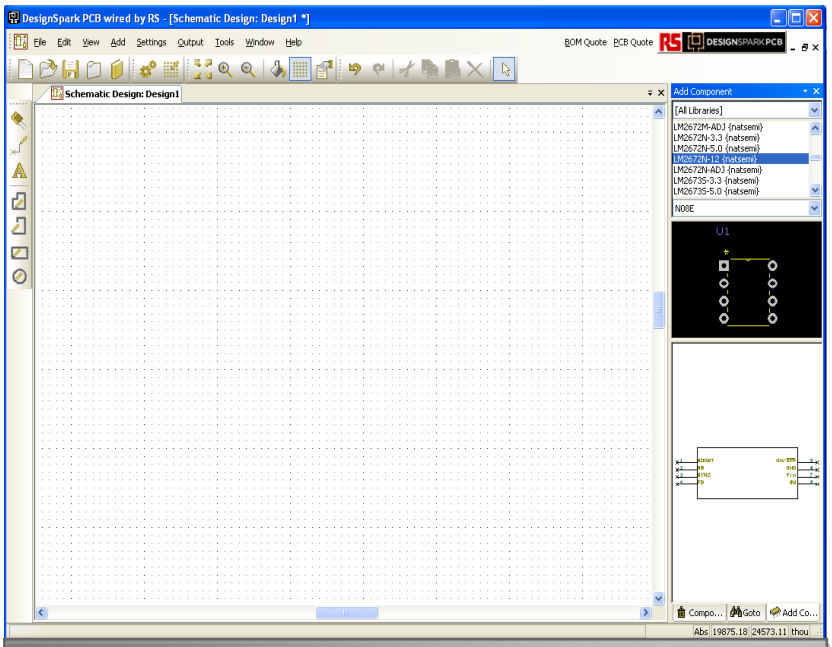


Figura 14: "Interaction Bar" añadida

En esta ***Interaction Bar*** se puede seleccionar la librería y el componente y dará una vista previa del aspecto del componente en el esquema, el formato y la forma del encapsulado para la posterior realización de la placa. En muy poco espacio da casi toda la información necesaria acerca del componente.

Al igual que antes, si se pincha dos veces ya se tiene disponible para colocarlo en el esquema.

2.2.3. Modelsource Bar

La tercera opción se consigue pinchando en ***View -> ModelSource Bar***, también de la barra de menú.

Con esta opción se puede buscar el componente dentro del catálogo de **RS Components**, empresa de distribución de todo tipo de componentes eléctricos y electrónicos, y la que ha creado este software de diseño de placas.

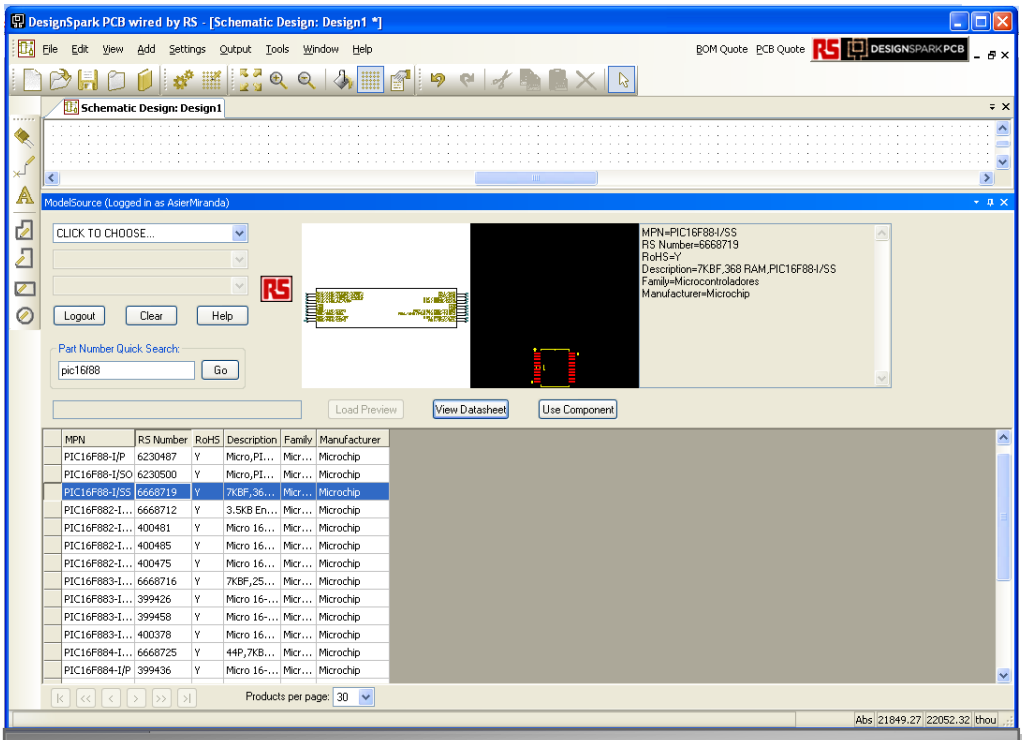


Figura 15: Ventana con "Modelsource Bar"

Además, se puede visualizar la forma del componente en el esquema y su encapsulado pinchando en ***Load Preview***. También aparece a la izquierda del encapsulado (si está disponible), información adicional como la referencia de RS (por si se quiere adquirir físicamente), la familia de la librería a la que pertenece el componente y la empresa que fabrica el mismo.

Una vez encontrado el elemento deseado no hay más que darle a ***Use Component*** y automáticamente creará una librería donde se encontrará el componente. Ahora se selecciona el icono de ***Añadir componente*** en la barra lateral izquierda, se busca la librería que ha creado anteriormente y ya se puede colocar el elemento en el esquema.

Dentro de esta opción también se puede buscar el componente a la inversa, es decir, si ya se sabe el componente deseado y se tiene localizado en la página de RS, con meter la referencia en el cuadro de búsqueda aparecerá el componente; si está disponible también aparecerá su encapsulado y símbolo del esquema.

2.3. Editar parámetros de un componente

Una vez colocado el componente se puede editar realizando doble clic sobre él y aparecerá un cuadro con pestañas para poder editar diferentes aspectos como por ejemplo, qué se quiere que se vea en el esquema, la posición del componente y su inclinación, entre otras cosas.

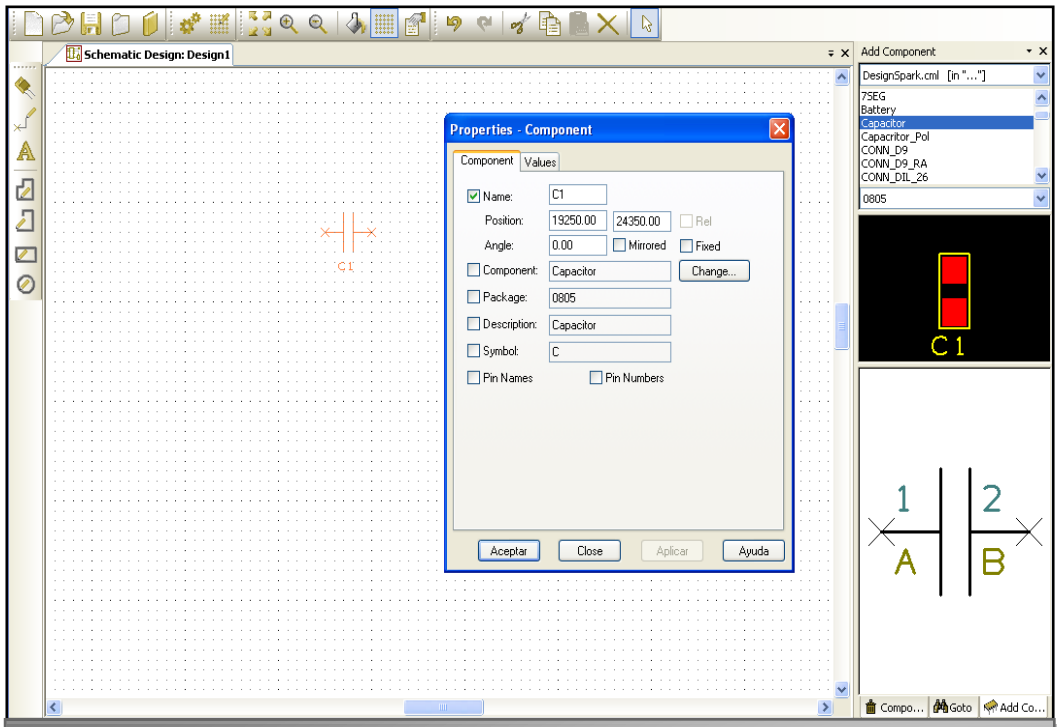


Figura 16: Modificar propiedades del componente

También se puede poner el valor del componente, por ejemplo en este condensador, para indicar que sea de 220uF, hay que ir a la pestaña ***Values***, pinchar en ***Add***, y escribir un nombre y el valor de 220uF. Luego seleccionarlo para que se pueda ver en el esquema (Figuras 17 y 18).

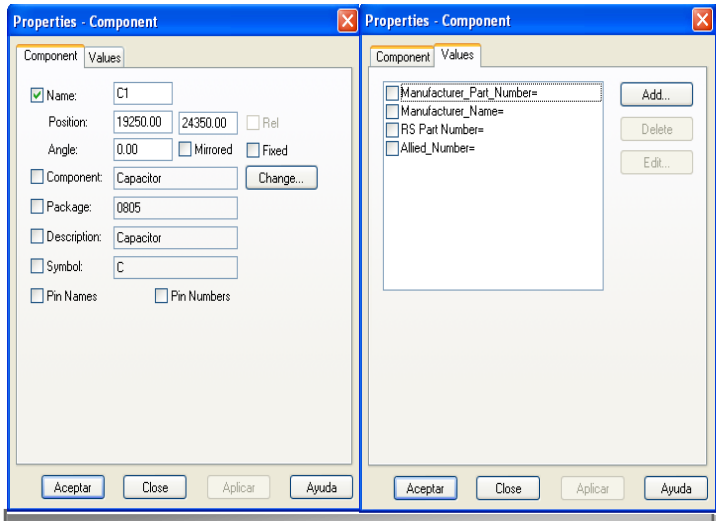


Figura 17: Modificando propiedades

En la pestaña ***Values*** se escribe el valor del componente y se selecciona el recuadro de la izquierda para que se vea en el esquema.

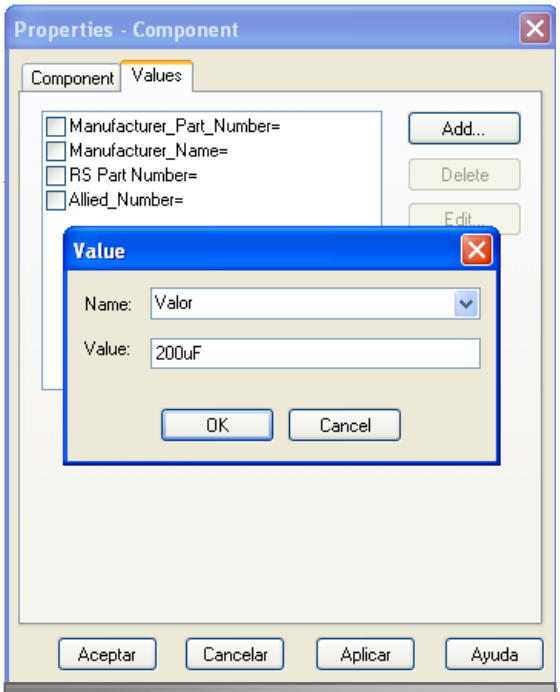


Figura 18: Añadir propiedades

3. Realizar conexiones entre componentes

Para unir los componentes entre sí, DesignSpark PCB posee una herramienta muy sencilla de usar que actúa como un lápiz dibujando las conexiones. Para lograrlo, hay que unir las cruces que se encuentran en los extremos de conexión de cada componente. El lápiz se sitúa en la Barra de Herramientas Lateral izquierda. Para usarlo sólo hay que hacer clic sobre el icono y llevar la conexión pulsando en cada "corte" o "giro" que se quiera darle a la vía. Una vez que se llega al terminal del componente destino, se vuelve a pulsar y la conexión se genera de manera automática.

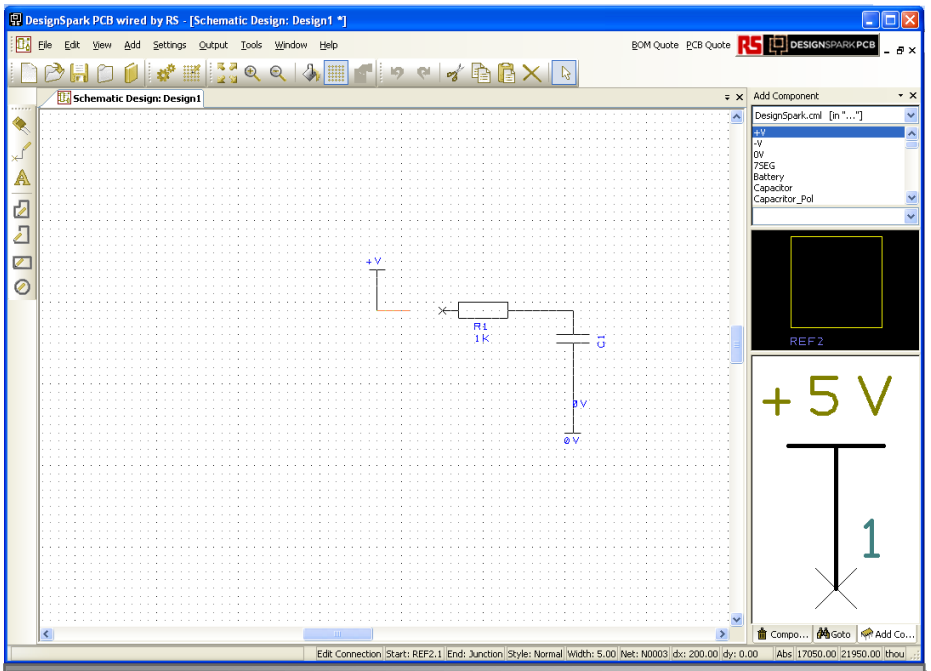


Figura 19: Conexionado de dos componentes

4. Creación y edición de componentes

En ocasiones se necesitarán componentes para un diseño de una placa pero que no se encuentra en las librerías. Otras veces se tendrá disponible para colocarlo en el esquema pero no estará disponible el encapsulado. Y en otras ocasiones no existirá un componente en las librerías pero sí un encapsulado igual o muy parecido. Por estas razones en este aparatado se va a indicar cómo crear componentes nuevos (tanto en esquema como para PCB), editar algunos ya existentes y cómo incluirlos en las librerías para poder tenerlos siempre disponibles.

4.1. Creación de un componente nuevo

La creación de un nuevo componente se divide en tres pasos: la creación del símbolo del esquema, la del símbolo en PCB y la unión de estos dos para que formen parte del mismo componente.

4.1.1. Creación del símbolo de esquema

Primero hay que acceder al ***Manager de librerías*** pinchando en el icono de la barra de tareas general y aparecerá una ventana similar a la de la Figura 20.

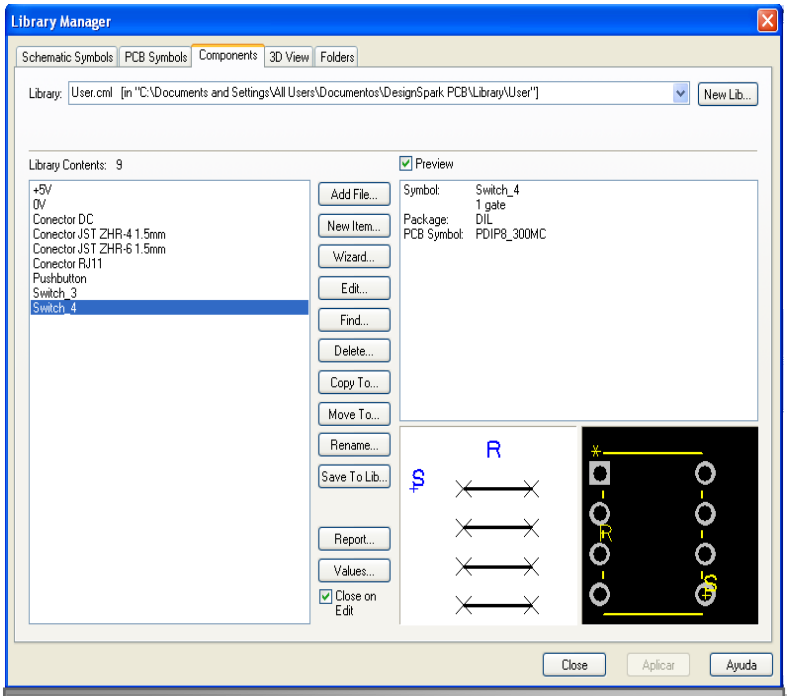


Figura 20: Manager de Librerías

Antes de todo se va a generar una librería para incluir ahí el ejemplo que se va a crear. Dentro del manager se selecciona la pestaña ***Schematic Symbols,*** se pincha arriba a la derecha donde pone ***New Lib…***, y ponemos el nombre que queramos a la librería, en este caso la llamamos "ejemplo" (Figura 21).

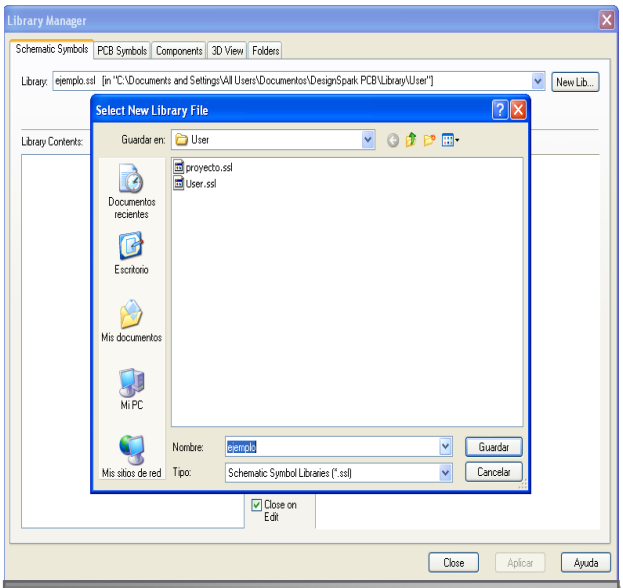


Figura 21: Crear nueva librería

Este proceso habrá que hacerlo para los dos pasos siguientes (símbolo PCB y creación final del componente). Se hace para guardar los símbolos en esas carpetas y tenerlos disponibles para la creación o edición de componentes si se desea.

Una vez creada la librería, hay dos opciones de crear un nuevo elemento, pinchando ***New Item…,*** o pinchando en ***Wizard***. La primera opción es para crear el componente desde cero manualmente, y la segunda es un asistente que ayudará y guiará al usuario en la creación. Como en este caso se va a crear un elemento sencillo, se elegirá la opción de hacerlo de forma manual. Para componentes más complicados o con muchos pines se recomienda el uso del asistente, ya que simplifica mucho el trabajo.

Una vez seleccionada la opción ***New Item...*** se obtendrá una ventana parecida a la del esquema general, sólo variará la barra de tareas lateral.

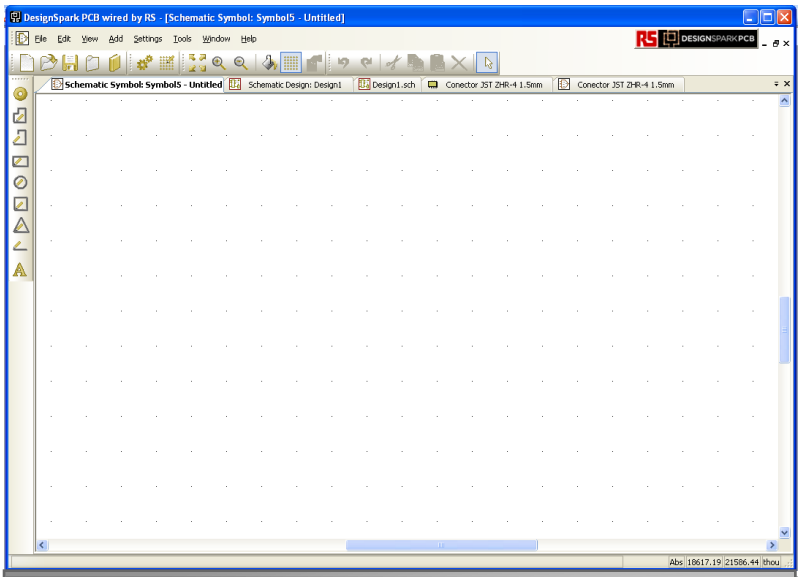


Figura 22: Pantalla de creación de símbolo de esquema

Barra de tareas lateral (Figura 23).

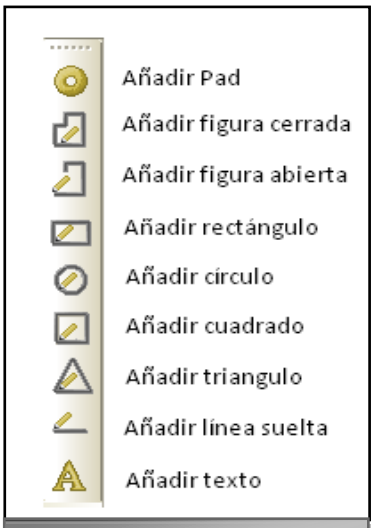


Figura 23: Barra de tareas lateral

Antes de empezar a dibujar, hay que establecer una serie de parámetros. Yendo a ***Settings->Units***, seleccionamos la opción *"mm"* con por ejemplo una décima de precisión:

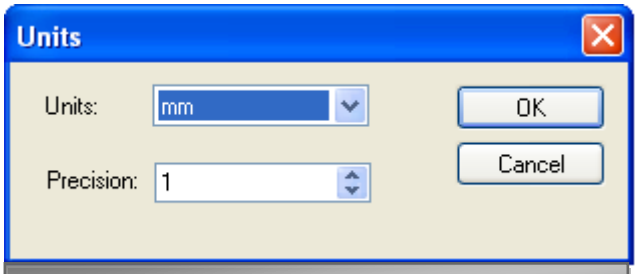


Figura 24: Editar unidades

Este parámetro es para que lo que se mida esté en milímetros. La opción ***Precision*** permite elegir el número de decimales que va a dar esta medida. La posición del ratón dentro del esquema se indica en la parte inferior derecha de la ventana. De esta manera se puede saber la distancia entre dos puntos.

Otro parámetro a configurar es el de *Grids*,  en la Barra de tareas General.

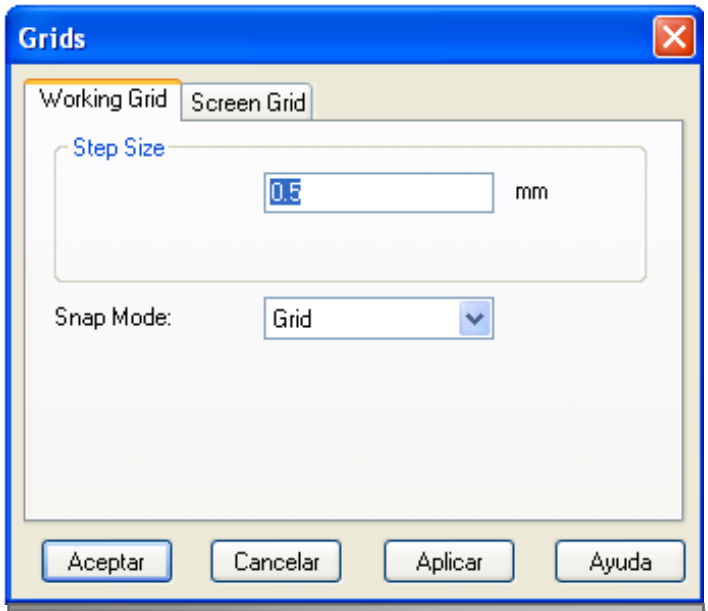


Figura 25: Editar "Grids"

Este parámetro modifica el escritorio donde se va a dibujar el elemento. En este caso cada vez que se esté dibujando una línea recta, al mover el ratón, se avanzarán 0,5mm. Esto se hace para conseguir mayor precisión a la hora de dibujar, que aunque en este apartado no es muy importante en el dibujo del símbolo PCB , que se explicará más adelante, es esencial, ya que requiere mucha mayor precisión debido a las dimensiones tan pequeñas de los componentes.

Ahora ya se puede empezar a dibujar el componente con la ayuda de la barra de tareas lateral. En este caso se va a dibujar un elemento muy sencillo, una resistencia.

En la Barra de Tareas General seleccionamos para realizar una figura cerrada. Pero antes hay que cambiar el estilo de línea a normal pulsando **<S>** mientras se dibuja la primera línea, se selecciona la opción ***Normal*** y se pulsa ***OK***.

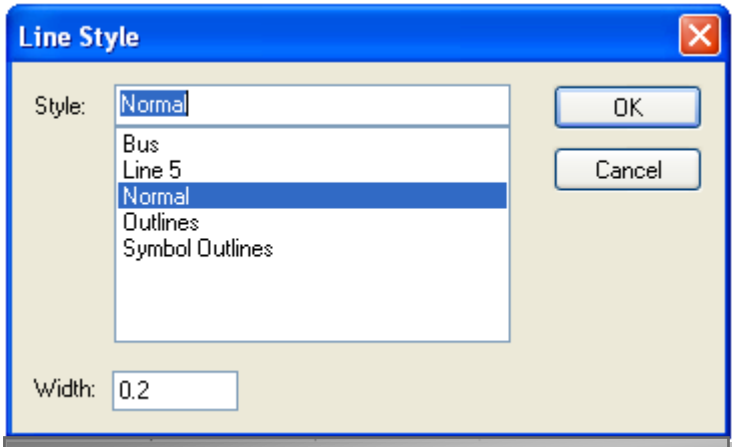


Figura 26: Editar estilo de línea

Ahora se procede a realizar uno de los símbolos típicos de resistencias.

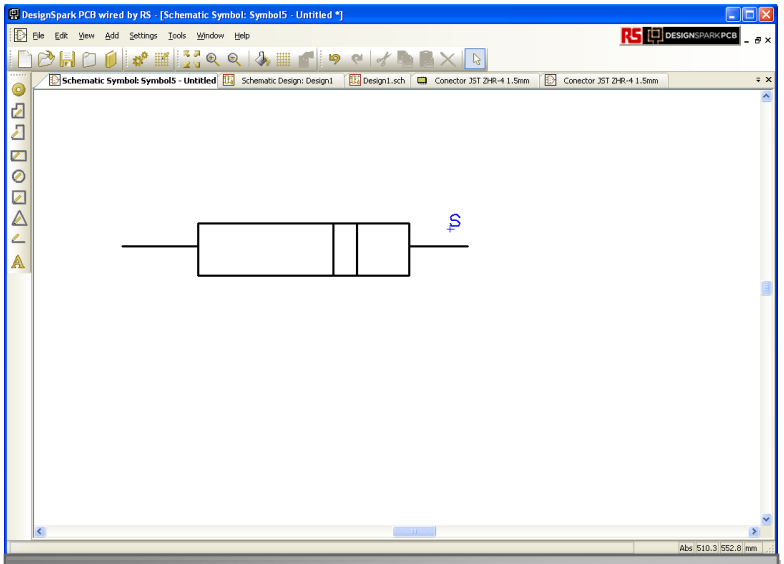


Figura 27: Símbolo dibujado

Hay que añadir los pines de conexión pinchando en ***Add Pad***.

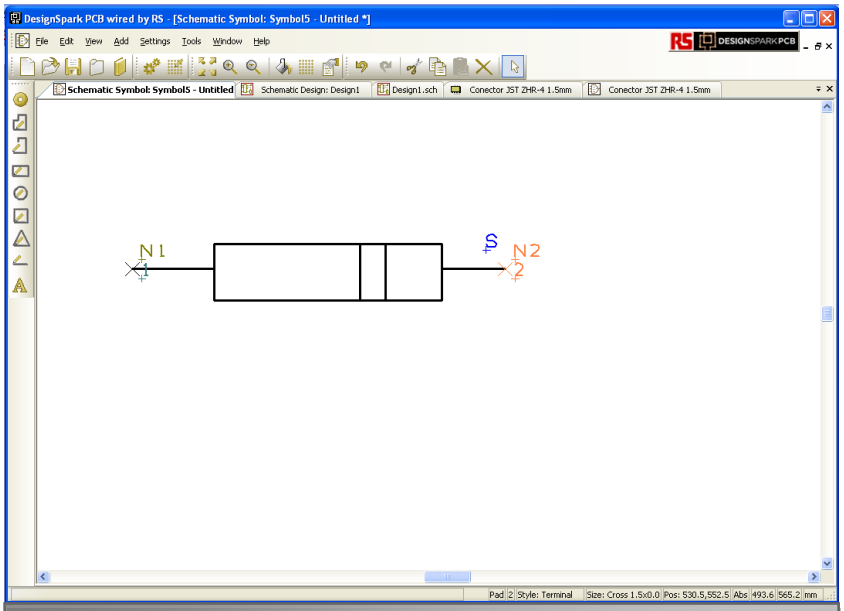


Figura 28: Símbolo con "Pads"

Una opción interesante es ir a la barra de menú e ir a ***Add->Reference Origin***. Con esta opción se añade un origen de referencia y permite mover el nombre del componente donde se desee, de lo contrario el programa lo pondrá encima del componente automáticamente y no se podrá mover, algo que visualmente no queda bien y puede molestar en el diseño del esquema, así que añadimos un origen de referencia:

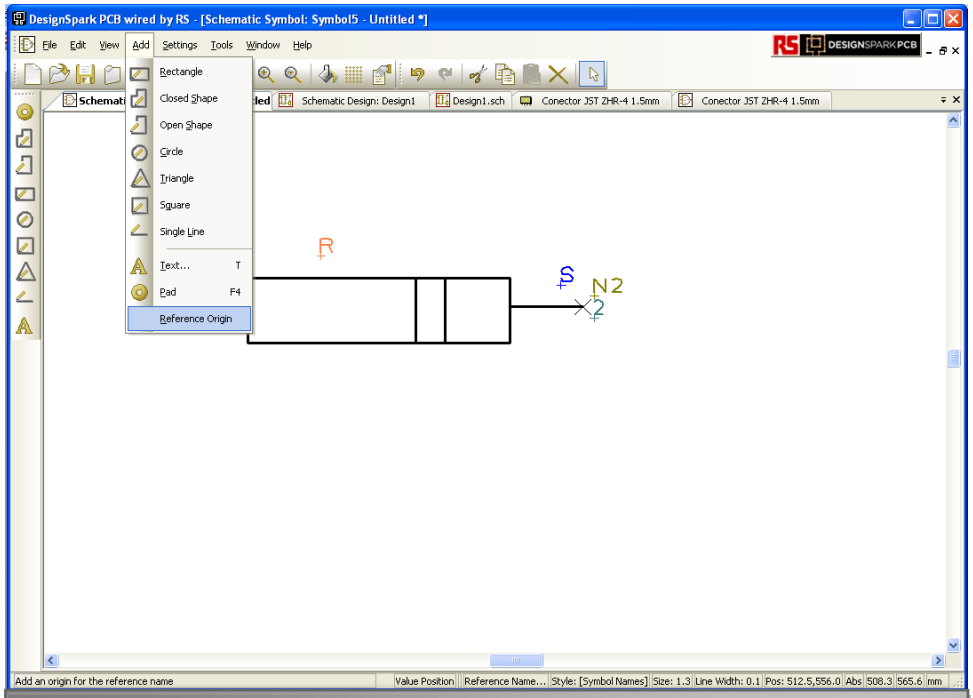


Figura 29: Añadir "Reference Origin"

Ahora ya podemos proceder a guardar el símbolo pinchando en el icono de guardado con el nombre "Resistencia" en nuestra librería *ejemplo.*

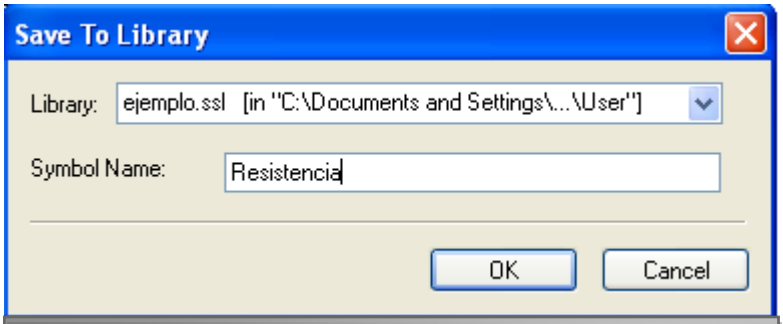


Figura 30: Guardar símbolo creado

4.1.2. Creación del símbolo PCB

Se accede al manager de librerías y en la pestaña *PCB Symbols* y se crea una librería con el mismo nombre que anteriormente, *ejemplo.* Y se selecciona la opción ***New Item***. Como en el paso anterior existe el asistente que es muy útil cuando el componente tiene muchos pines pero que ahora se va a descartar.

Aparece la siguiente pantalla:

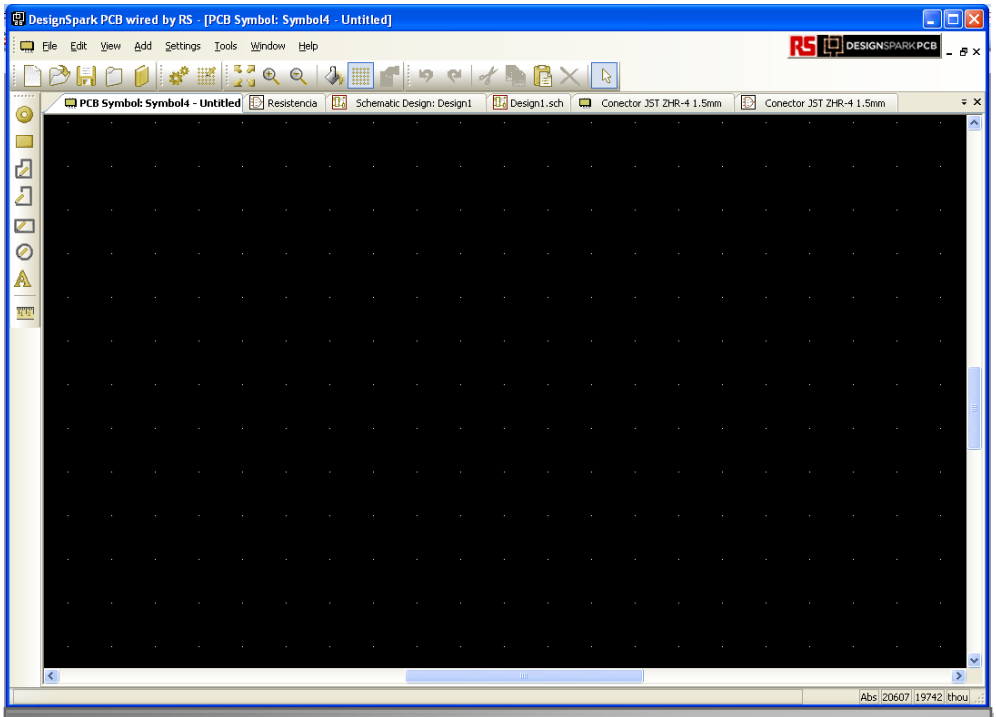


Figura 31: Pantalla inicial de creación símbolo de PCB

De la misma manera que en el paso 4.1, se configuran los dos parámetros, ***Units*** y ***Grids***.

La barra de tareas que se va a utilizar es similar pero difiere un poco de la del paso 4.1:

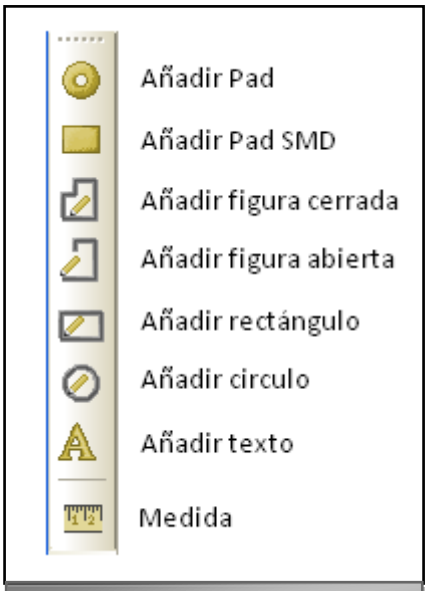


Figura 32: Barra lateral

-Añadir Pad SMD: es un tipo de conexión para los pines sin perforar la placa

-Medida: nos da la distancia entre dos elementos. Si son circulares, también nos da la distancia entre sus centros. Muy útil para comprobar medidas.

Los demás elementos son como en la Edición del símbolo de esquema.

Con ayuda de los iconos de esta barra se procede a realizar el dibujo del encapsulado de la resistencia teniendo en cuenta las medidas de una resistencia real, cuya parte cerámica mide unos 9mm de largo por 3mm de ancho. Se puede comprobar las medidas con la herramienta ***Medida*** de la barra lateral.

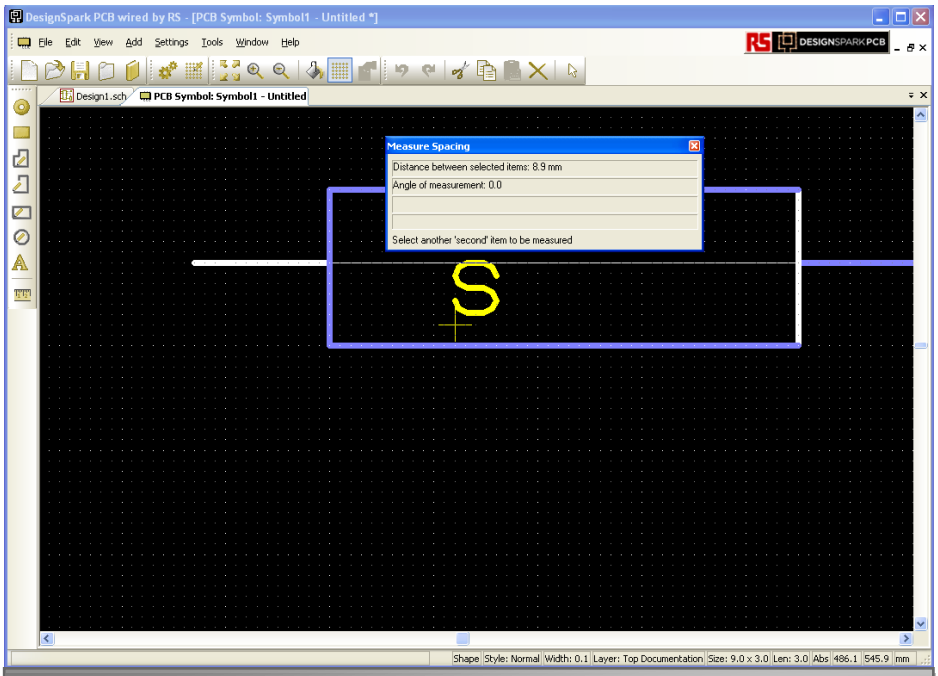


Figura 33: Comprobando medidas

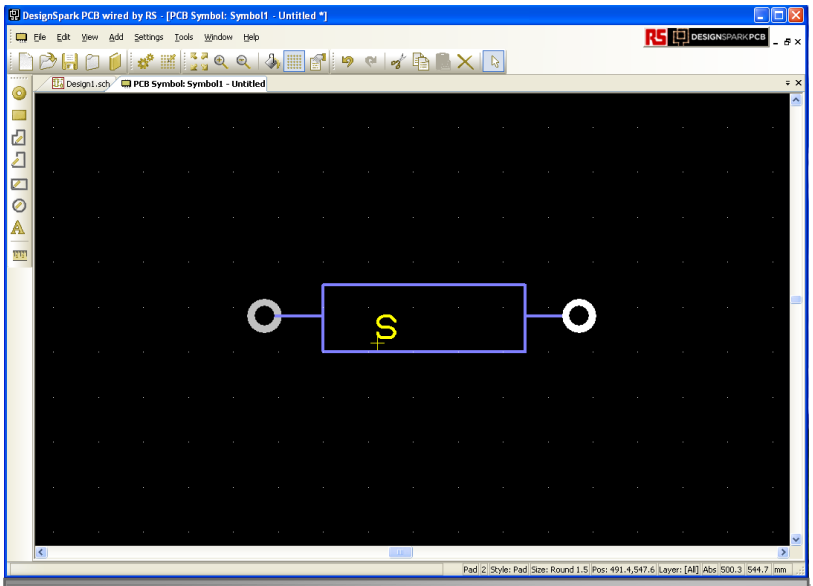


Figura 34: Símbolo con "Pads" añadidos

Se colocan los Pads en los extremos de la resistencia, como en el caso del símbolo del esquema.

Ahora se van a editar estos Pad para que coincidan con las patillas de la resistencia real, las cuales tienen un diámetro de algo menos de 0,5 mm. Por lo tanto hay que poner un orificio de 0,6 mm en la placa para que no haya problemas a la hora de introducir la resistencia. También hay que configurar el diámetro del cobre que va a rodear al orificio, se elige una medida 1,5mm.

Haciendo doble clic sobre el Pad aparece la siguiente ventana y se modifican los valores de ***Hole size*** y ***Width***, que hacen referencia al orificio y al cobre que le rodea, respectivamente.

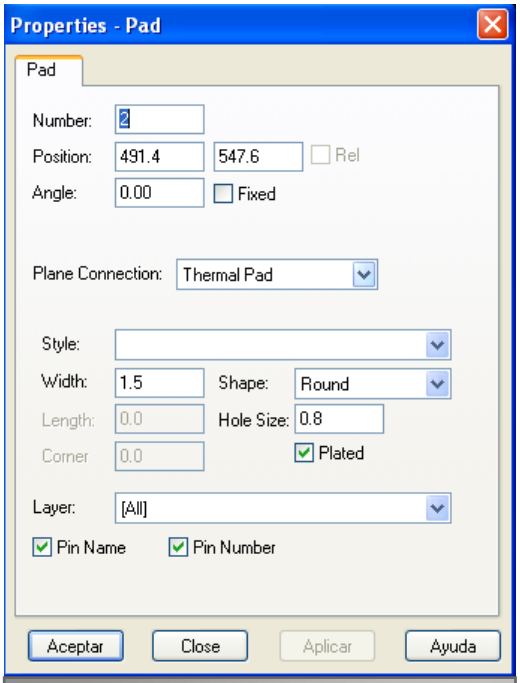


Figura 35: Propiedades el Pad

Se coloca el ***Reference Origin*** y ya se puede guardar el símbolo con el nombre "Resistencia" en la librería *ejemplo.*

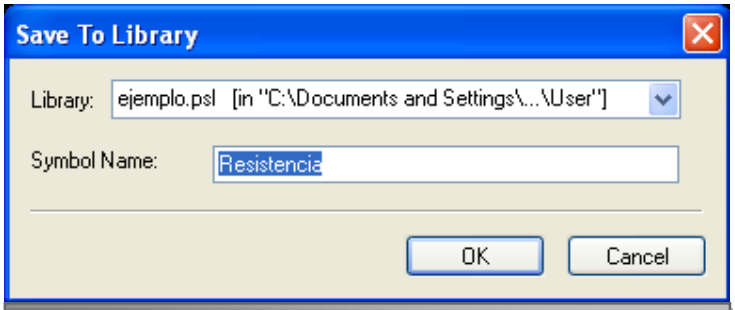


Figura 36: Guardar símbolo de PCB

En este momento ya están creados los símbolos del esquema y PCB de la resistencia.

Ahora se va a crear el nuevo componente y utilizar estos dos símbolos.

4.1.3. Creación componente

En el ***Manager de librerías***, se selecciona la pestaña ***Components*** y se crea la librería correspondiente *ejemplo.* Ahora se selecciona la opción ***New Item***. Como en los pasos anteriores está el asistente de creación que aunque en este caso no difieren mucho las dos opciones de creación entre sí, vamos a seguir con la que hemos utilizado hasta ahora: ***New Item***. Se obtiene la siguiente ventana.

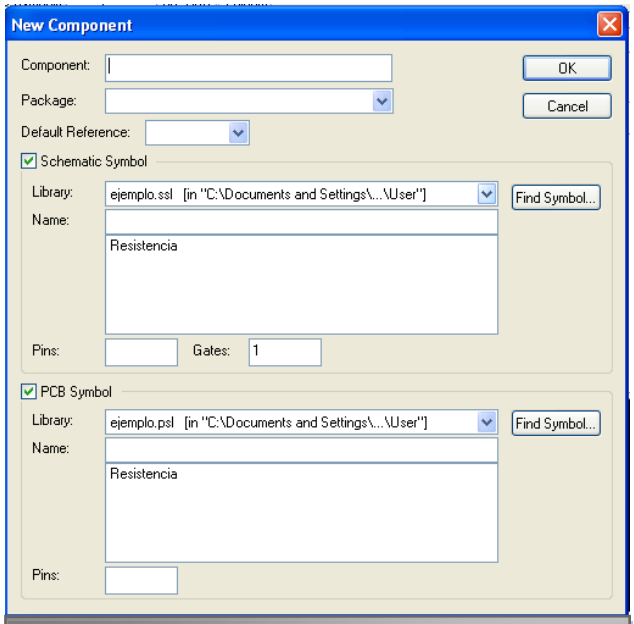


Figura 37: Ventana de nuevo componente

Se modifican las opciones de la ventana para que quede como en la imagen, es decir, se relacionan los dos símbolos creados antes con el nuevo componente.

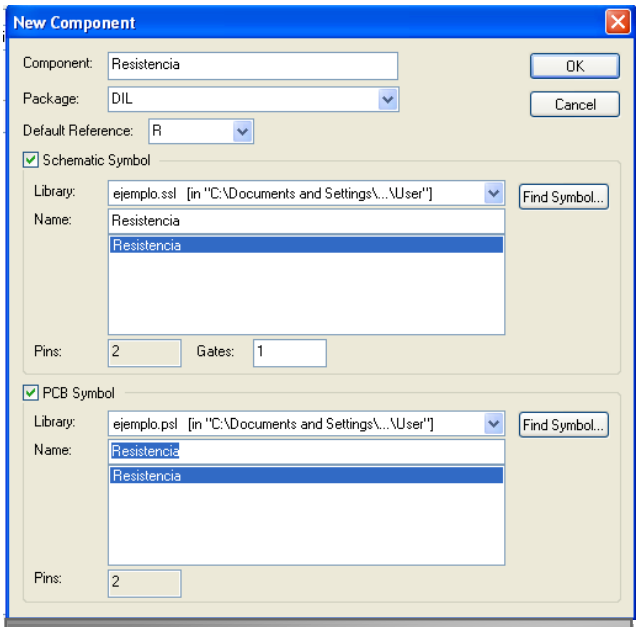


Figura 38: Propiedades del componente

-Component: nombre del componente.

-Package: tipo de encapsulado.

-Default Reference: referencia para la búsqueda del componente.

Se seleccionan los símbolos de esquema y PCB que han sido creados. Se pincha en OK y aparece la siguiente pantalla:

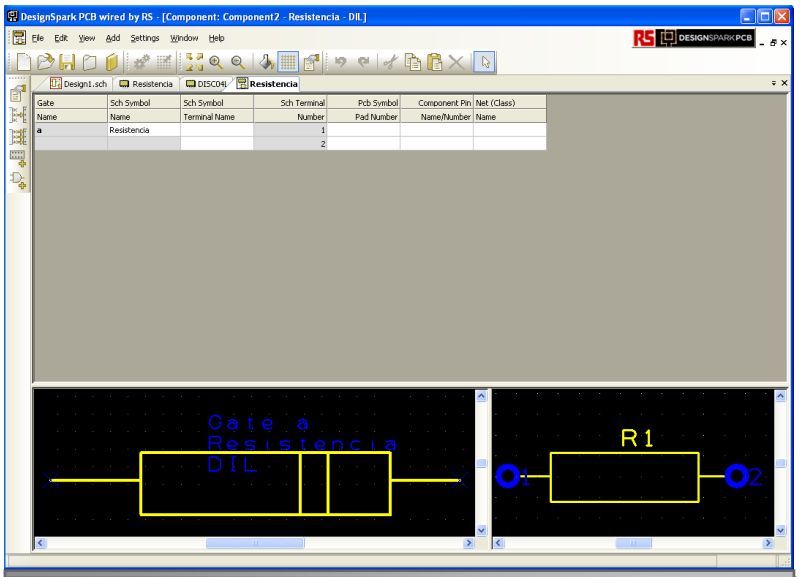


Figura 39: Vista previa del componente

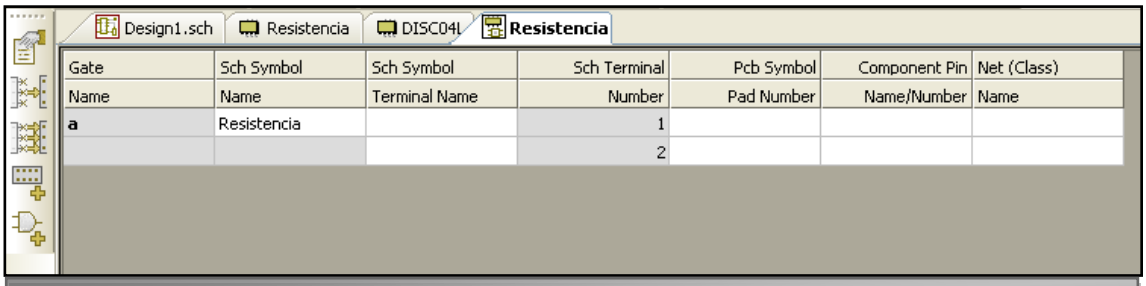


Figura 40: Edición de los Pads

Se utiliza la siguiente barra de tareas para terminar de editar el componente:

Figura 41: Barra lateral

Ahora se deben asignar los pines para que se correspondan los del esquema con los del símbolo de PCB. Se puede hacer manualmente con el icono ***Asignar pins***, o automáticamente con la opción ***Asignar pins uno a uno***. En este caso se elige la segunda opción ya que es más rápida. La ventana debe quedar como la imagen.

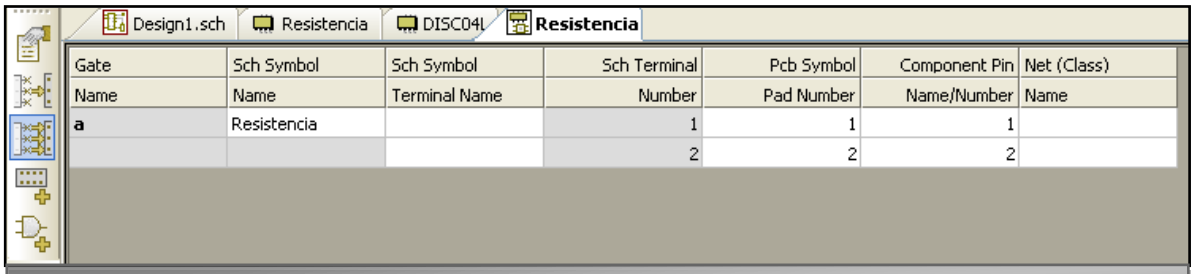


Figura 42: Pads asignados a cada símbolo

El pin 1 del esquema se corresponderá con el 1 del símbolo PCB, y el pin 2 con el 2 del PCB. Si se desea cambiar, hay que ir al icono de ***Asignar pins*** y realizar las modificaciones.

Una vez hecho esto ya se puede guardar el componente con el nombre “Resistencia” en la librería *ejemplo,* y se tendrá disponible para ponerlo en el esquema y en el diseño de la placa.

A la hora de añadirlo debe quedar así:

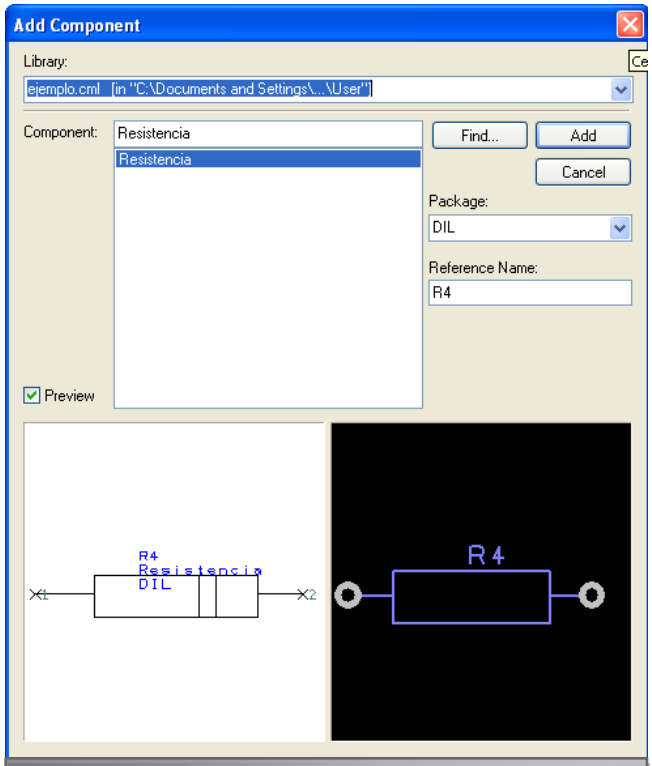


Figura 43: Nuevo componente

4.2. Edición de componentes

A menudo, los proyectos necesitarán componentes que no existen dentro de las librerías, pero sí un símbolo de esquema o un símbolo PCB muy parecido o igual al del componente. Para poder utilizarlos es necesario editar estos símbolos y crear el componente. Para explicar cómo hacerlo se va a realizar un ejemplo suponiendo que se necesita un diodo zener y no está disponible en la librería.

4.2.1. Edición símbolo de esquema

Se accede al ***Mánager de Librerías***, en la pestaña de ***Schematic Symbols***. Como el componente es un diodo zener y se conoce su símbolo, se modificará el símbolo de un diodo LED, ya que es muy similar. Hay que ir a la librería ***DesignSpark*** y buscamos el símbolo de ***LED***.

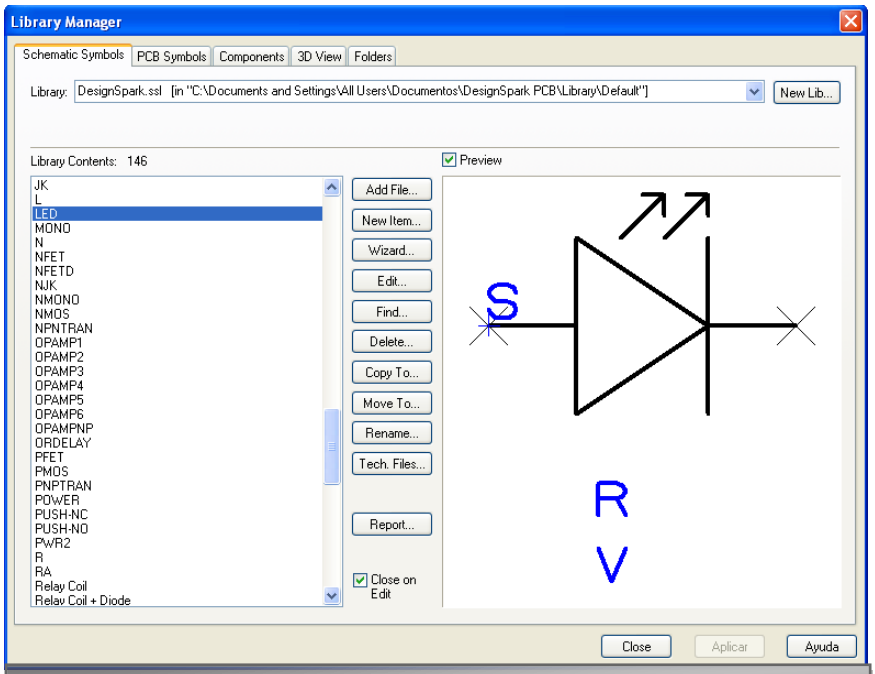


Figura 44: Símbolo "LED"

Una vez aquí se pincha en ***Edit…*** y aparece la ventana como en la Figura 45 para editar el componente.

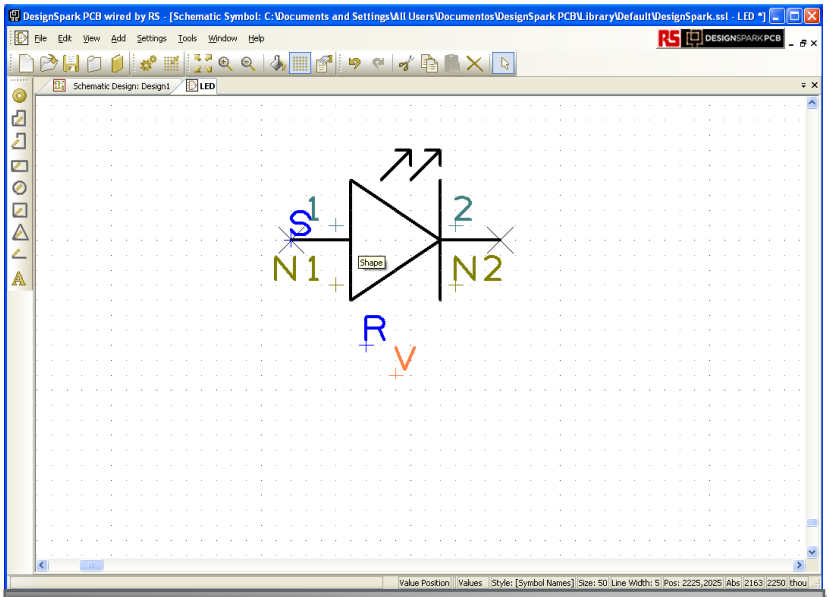


Figura 45: Aspecto del componente a editar

Se modifica para que quede como en la Figura 46.

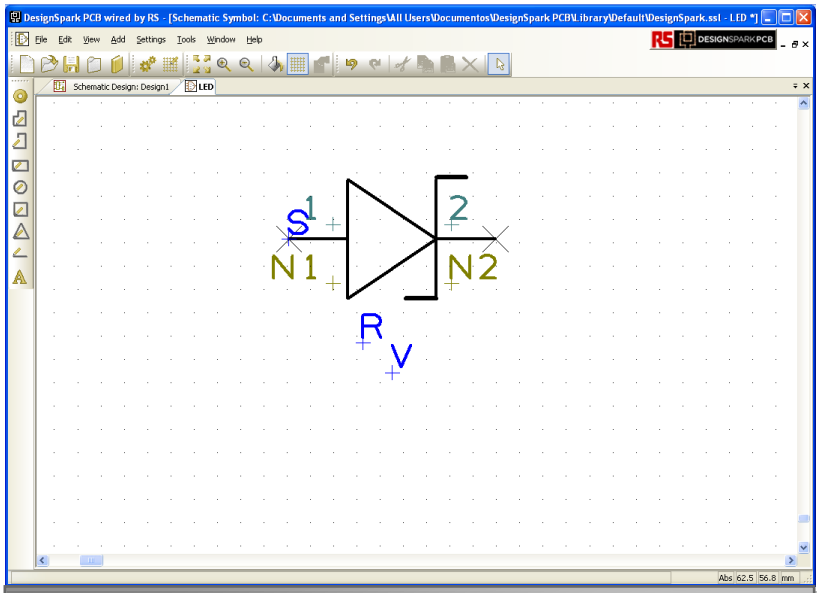


Figura 46: Componente modificado

Una vez modificado se guarda en la librería "*ejemplo*" con el nombre "Diodo zener".

4.2.2. Edición símbolo de PCB

Se accede al ***Manager de Librerías***, la pestaña de ***PCB Symbols*** se localiza el símbolo del diodo, en este caso "DIOD04".

Como el encapsulado del diodo y el zener son iguales, se va a conseguir el símbolo de forma diferente al paso 4.1. Se va a copiar directamente a la carpeta ejemplo. Es la forma más rápida de conseguir el símbolo si éste es igual al del componente. Se puede hacer igualmente para el caso en que coincida el símbolo del esquema.

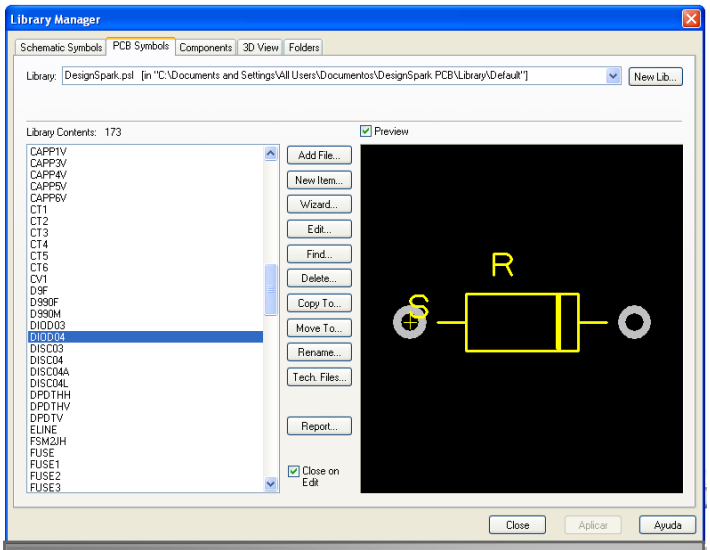


Figura 47: Símbolo PCB a copiar

Se pincha en la pestaña ***Copy To…***

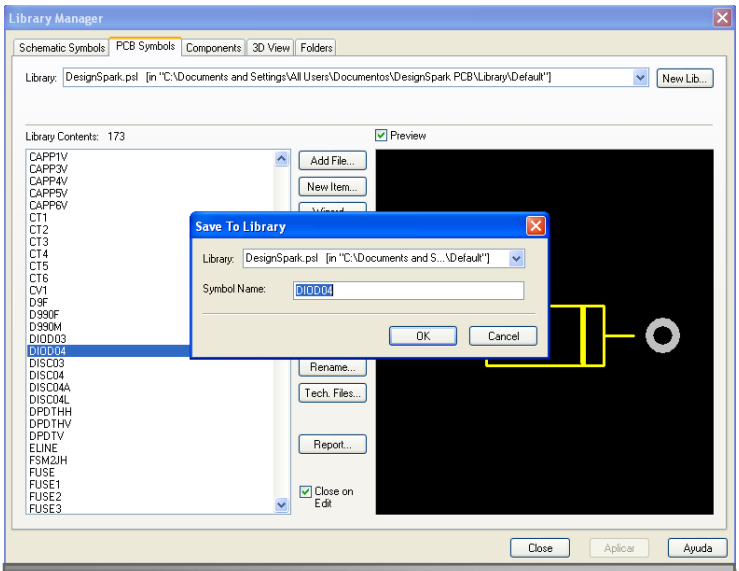


Figura 48: Copiar componente

Se selecciona la librería *ejemplo* y se guarda con el nombre "Diodo zener". Clic en **OK** y ya se tiene el símbolo guardado.

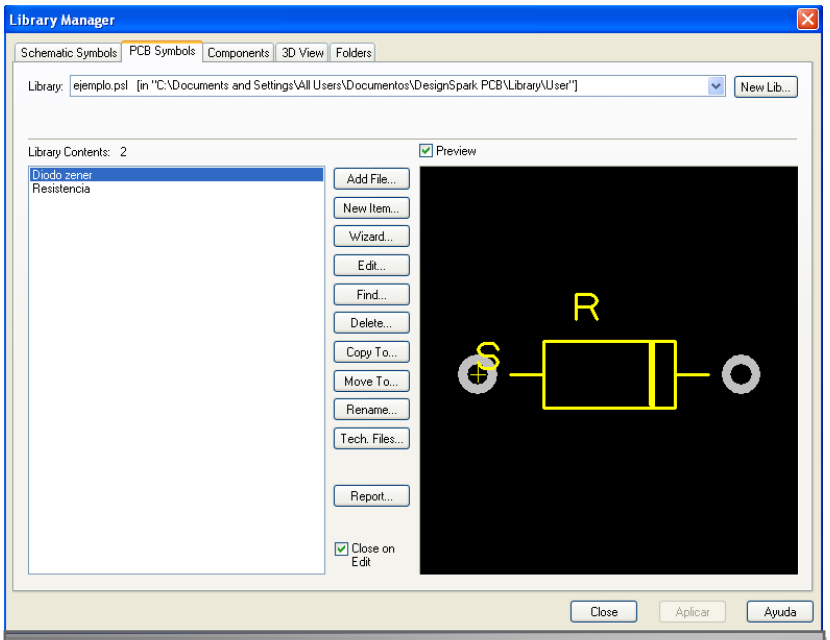


Figura 49: Componente copiado

4.2.3 Creación del componente

Ahora sólo queda crear el componente en sí, tal y como se ha hecho en el paso ***4.1.3. Creación componente***. Pestaña ***Components*** del ***Manager de Librerías***, seleccionar la librería *ejemplo*, pinchar en ***New Item…***, configurar los parámetros como en la Figura 50 y darle a ***OK***.

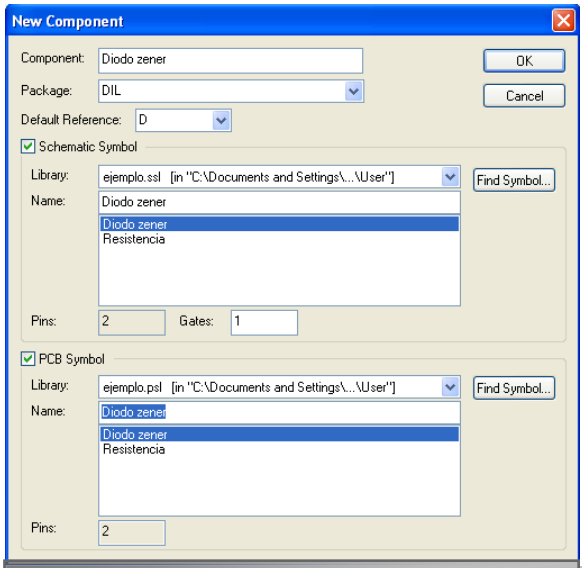


Figura 50: Parámetros componente

Sólo queda asignar los pads de los dos símbolos y guardar el componente en la librería *ejemplo* con el nombre “Diodo zener”.

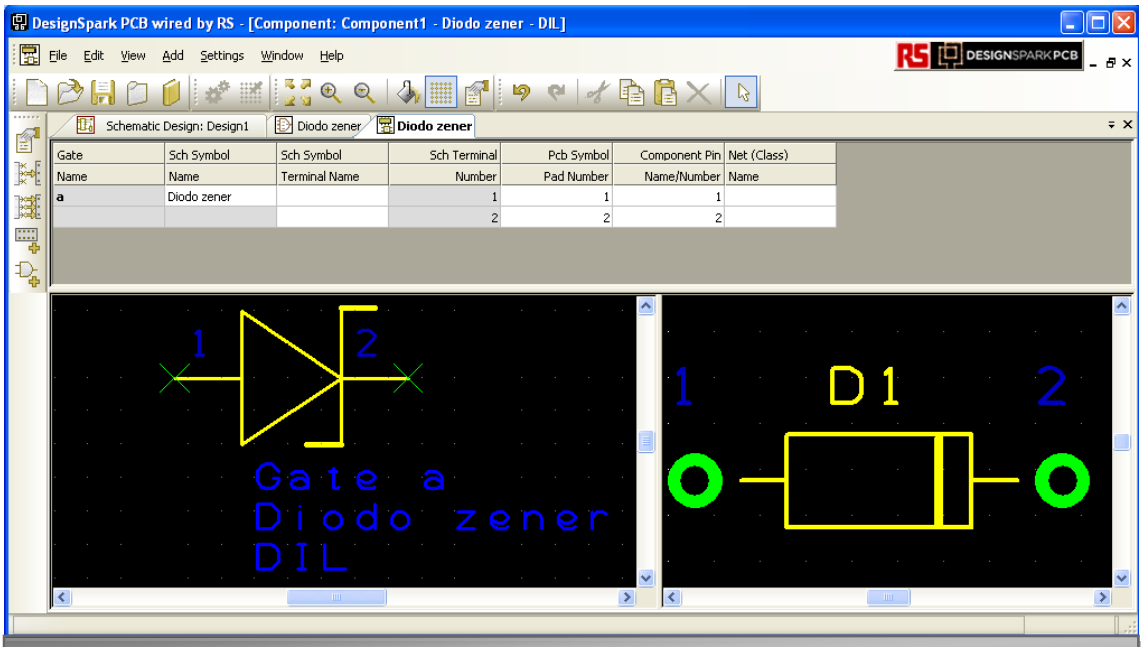


Figura 51: Asignación de pads

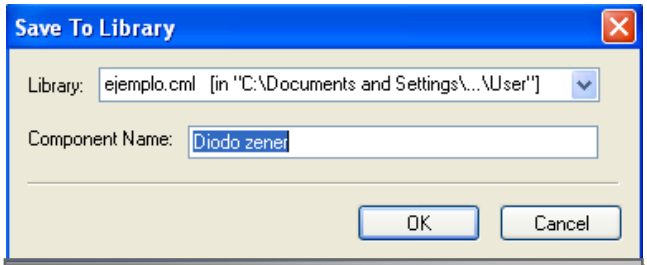


Figura 52: Guardar componente

De esta manera se pueden crear numerosos componentes, y de forma muy rápida, ya que aunque hay muchos de ellos que no están en las librerías, tienen símbolos muy similares o iguales a otros que sí están, y haciendo unos pocos retoques se pueden conseguir los símbolos que queramos.

En el siguiente apartado se va a explicar cómo pasar del esquema diseñado a una placa PCB de circuito impreso, todos los pasos a seguir y las diferentes opciones que presenta el programa.

5. Diseño de la placa PCB

Una vez que se sabe colocar componentes, crearlos y editarlos, se va a explicar la parte para la que se ha creado este programa, pasar de un esquema electrónico a una placa real. Para ello se va a realizar un ejemplo sencillo con pocos componentes. El ejemplo va a constar de una pila cuadrada de 9V con un LM340T-5 para pasar la tensión a 5V y que éste ilumine un LED rojo cuando le demos a un pulsador. Todos los componentes se encuentran en la librería DesignSpark salvo el del LM340T-5 que habrá que crearlo. Su símbolo de PCB coincide con el TO220 de la librería DesignSpark y su símbolo de esquema se consigue fácilmente editando el símbolo "SBLOCK" de esta misma librería. De esta manera se practica lo explicado en apartados anteriores.

Una vez puestos los componentes y editados sus valores debería quedar el esquema similar al de la Figura 52.

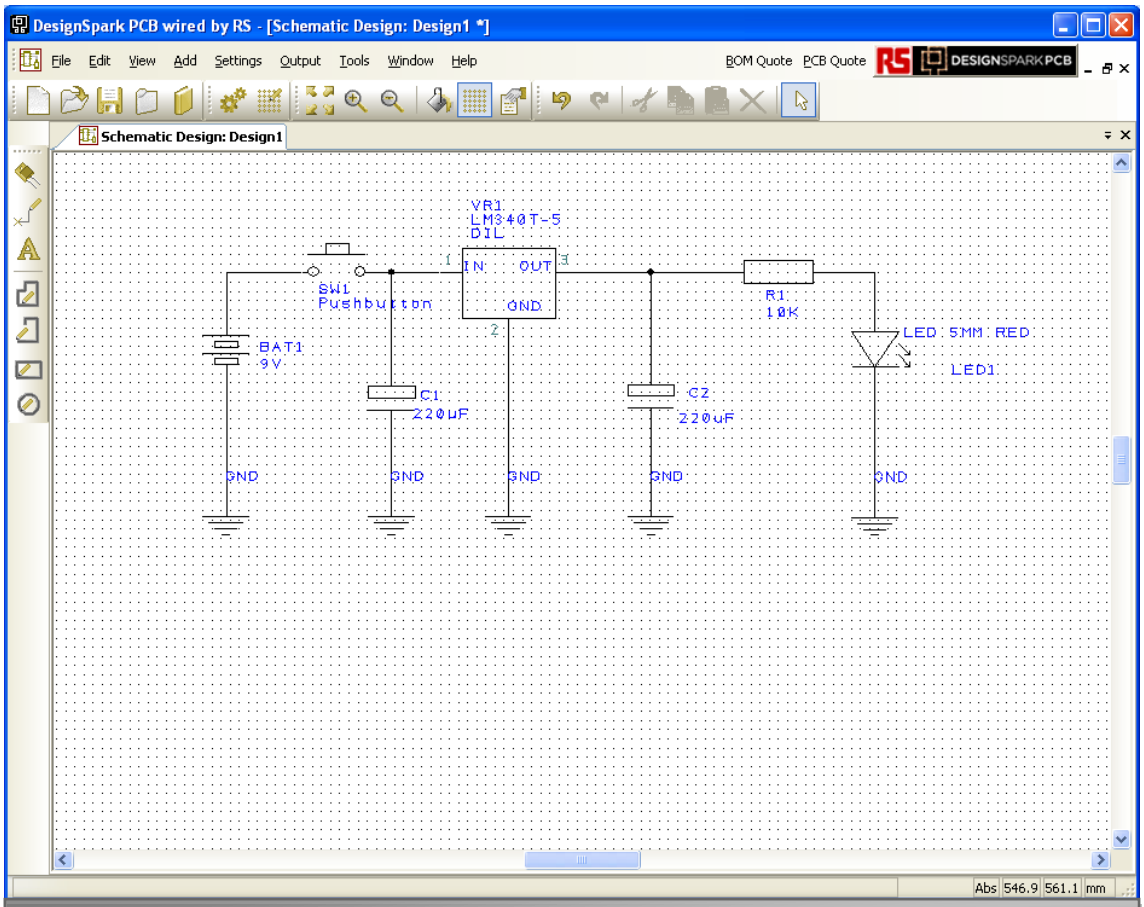


Figura 53: Esquema ejemplo

5.1. Crear archivo PCB

Como este es el esquema final, se puede proceder a pasarlo a PCB. Para ello en la *Barra de Menú* se accede a ***Tools -> Translate to PCB…*** y aparecerá el asistente de creación de PCB que guiará por la creación de la placa.

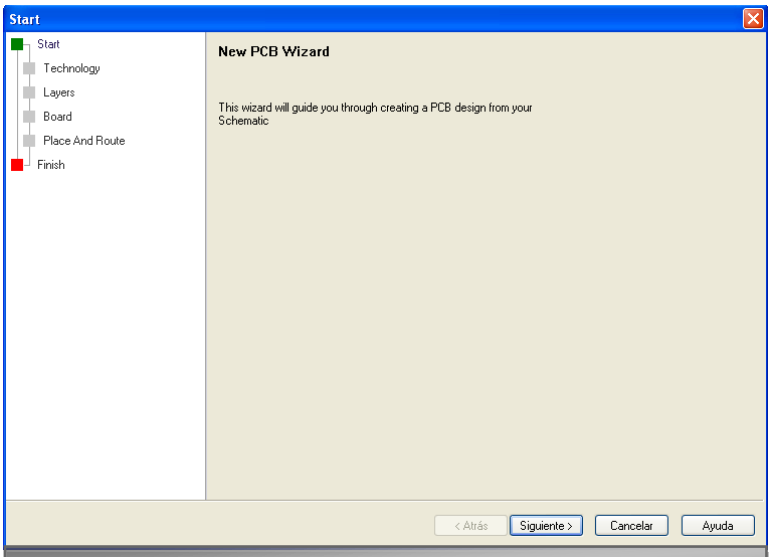


Figura 54: Primera imagen del asistente

Clic en ***Siguiente*** y se tendrá la opción de seleccionar en qué unidad se va a determinar el tamaño de la placa (Figura 54). Se seleccionan milímetros. Además se mantiene la opción que viene por defecto ***Use Default Technology*** y clic en ***Siguiente***.

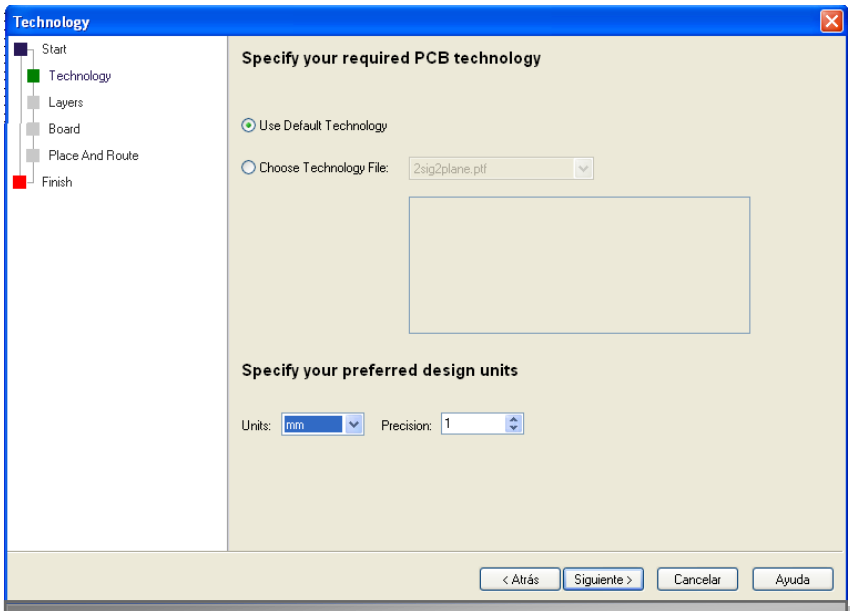


Figura 55: Selección de unidades

Ahora aparece la parte en que se configuran las capas de las que va a constar la placa (Figura 55).

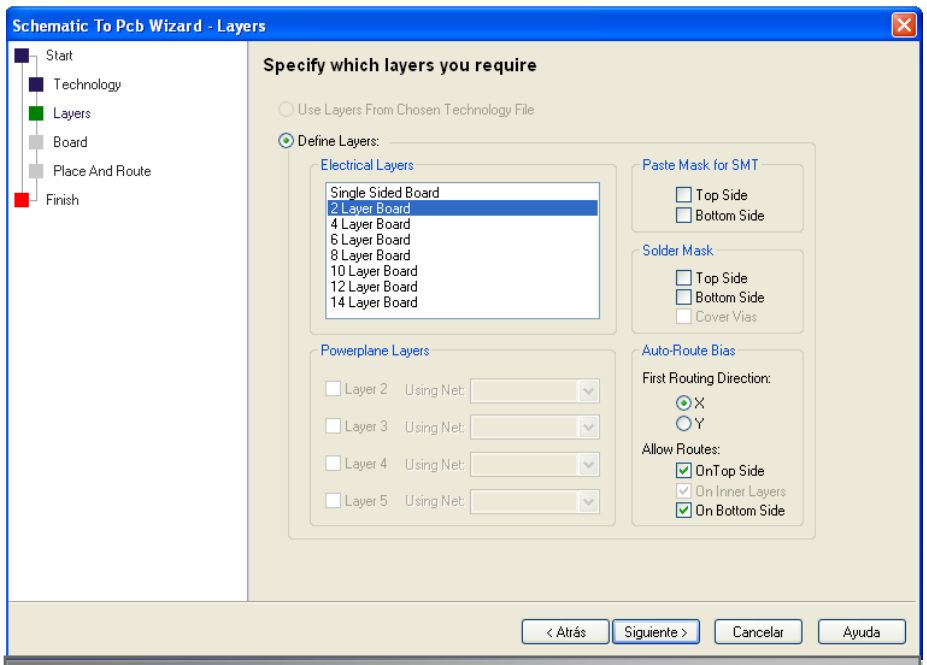


Figura 56: Selección de capas de la placa

Cada parte de la placa que tiene vías o pistas de cobre se le llama capa. Hay que distinguir el término capa con el de cara o lado. Las placas sólo tienen dos caras pero pueden tener una capa, dos o más, según se diseña la misma.

Dentro de esas dos caras se puede hacer que sólo una de las dos (la superior) tenga cobre, que la se llamará de simple cara o que tengan las dos, que se le llamará de doble cara.

¿Cómo se consigue tener varias capas en una sola placa?

La respuesta está en la tecnología de fabricación del PCB. Por ejemplo, se puede depositar una capa de circuito impreso, luego sobre ella depositar una capa aislante completa, luego otra capa de cobre (las vías únicamente), luego aislante, luego cobre y así hasta formar una pila de capas intermedias que llevarán conexión a múltiples lugares dentro de un PCB [2].

El poner varias capas en una sola placa se utiliza cuando se tienen muchos componentes en relación con el espacio total de la placa, ya que si sólo hay pistas en una sola cara es imposible realizar todas las conexiones. Este concepto se irá aclarando mejor a lo largo de este tutorial.

Una vez explicado esto, se van a configurar las capas del ejemplo. Como el esquema es muy sencillo se va a seleccionar una sola capa con la opción ***Single Sided Board*** (simple cara) del cuadro de ***Electrical Layers.***

Respecto a las opciones de máscaras que hay a la derecha, se dejan sin marcar, ya que hacen referencia a las máscaras que se aplican a las soldaduras. Por ejemplo, “Solder Mask” hace referencia a la pintura de color que impide que la soldadura se derrame por toda la vía y para proteger al cobre del PCB de la corrosión y el paso del tiempo [2].

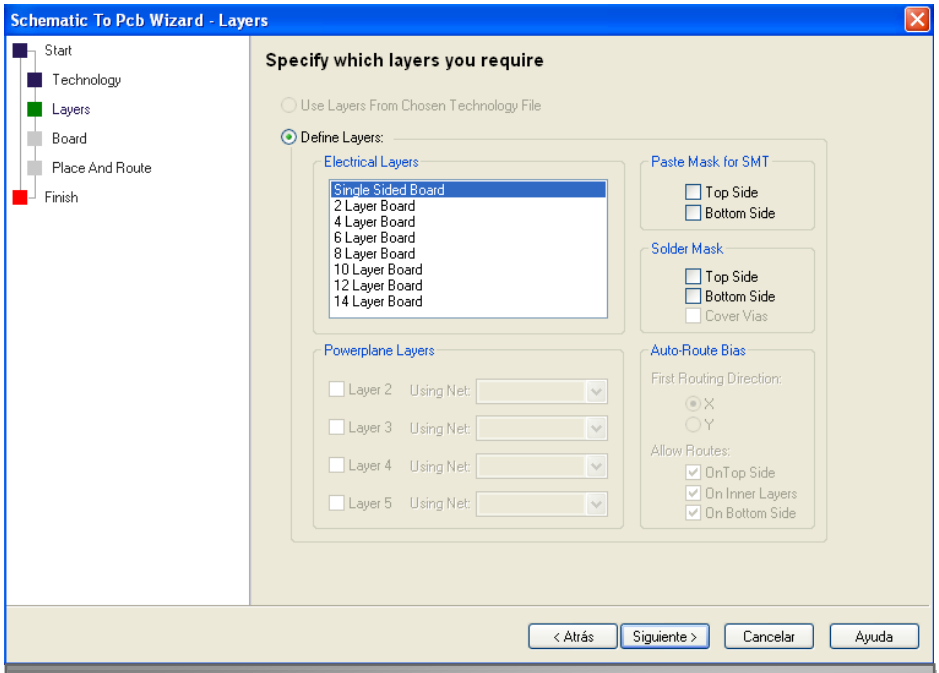


Figura 57: Selección de capas de la placa

Se pulsa en ***Siguiente*** y se obtiene la ventana donde configurar las medidas físicas de la placa. Dado los pocos componentes del circuito, con unas medidas de 50mm de ancho por 70mm de largo es suficiente.

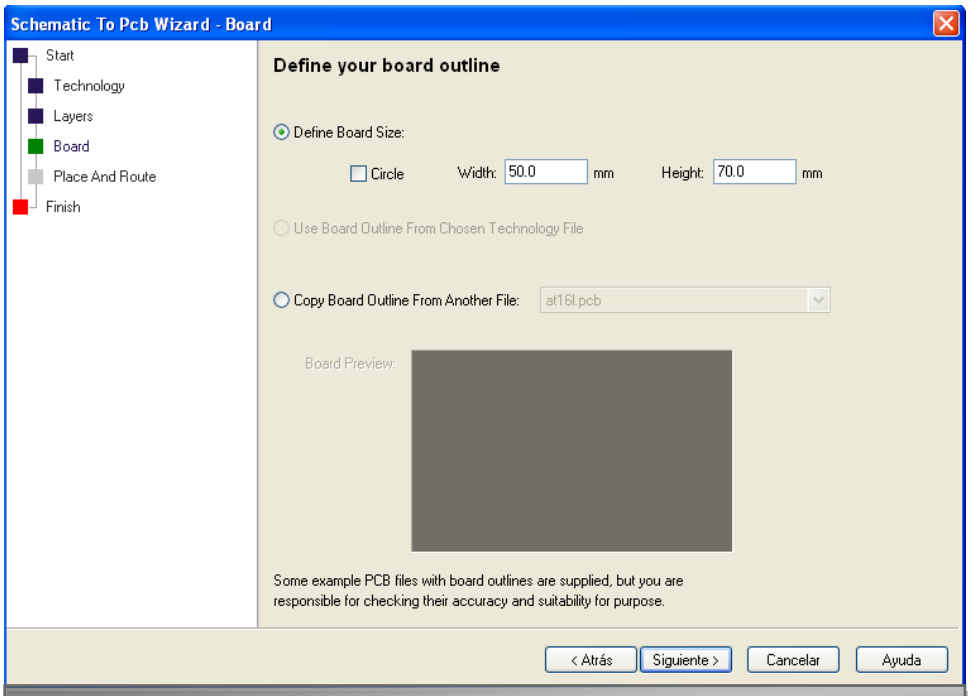


Figura 58: Medidas de la placa

En la siguiente ventana da la opción de situar los componentes fuera de la placa (***Arrange Outside the Board***) o de situarlos automáticamente (***Automatically Position Inside the Board***). Se elige la que viene por defecto de situarlos fuera de la placa y ya se situarán más adelante.

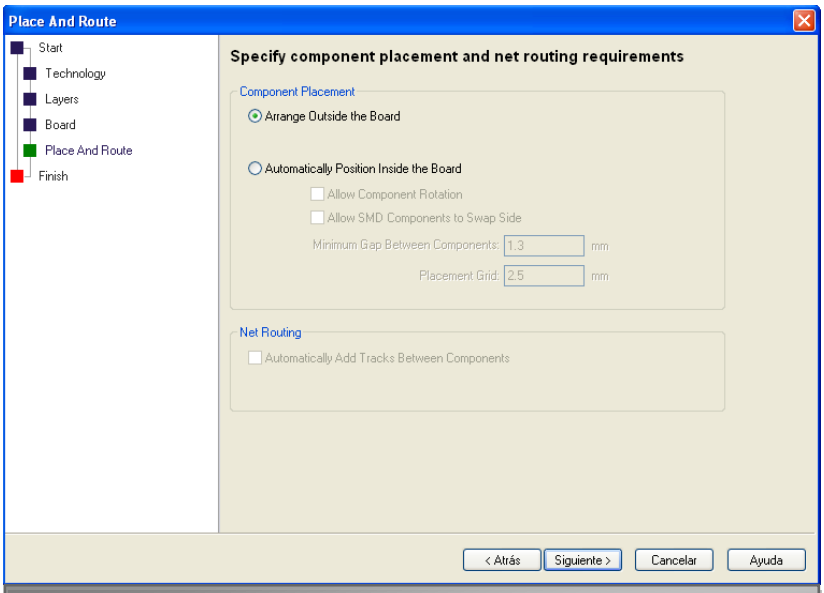


Figura 59: Asignar componentes fuera de la placa

Clic en siguiente y en esta pantalla permite seleccionar donde se quiere guardar y con qué nombre el archivo **.pcb** que contiene la placa. En principio lo guarda con el mismo nombre con el que se ha guardado el esquema, es la mejor forma, así se tienen los dos archivos localizados. Se selecciona el directorio y clic en ***Finalizar***.

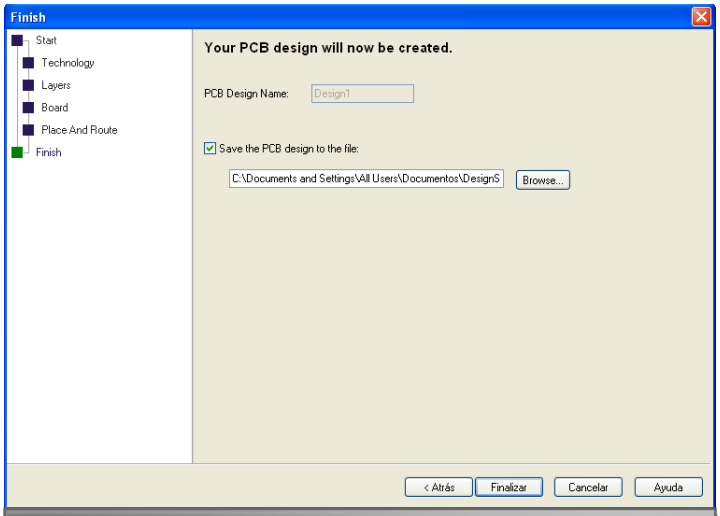


Figura 60: Guardar archivo .PCB

5.2. Edición de la placa

En este apartado se va a explicar cómo colocar los componentes en su sitio, realizar las pistas y todos los aspectos necesarios para configurar correctamente la placa pcb.

5.2.1. Disposición de los componentes

Una vez terminado con el asistente de creación aparecerá una pantalla como la Figura 60. El recuadro verde es el tamaño que se le ha asignado a la placa, y situados fuera de ella todos los componentes.

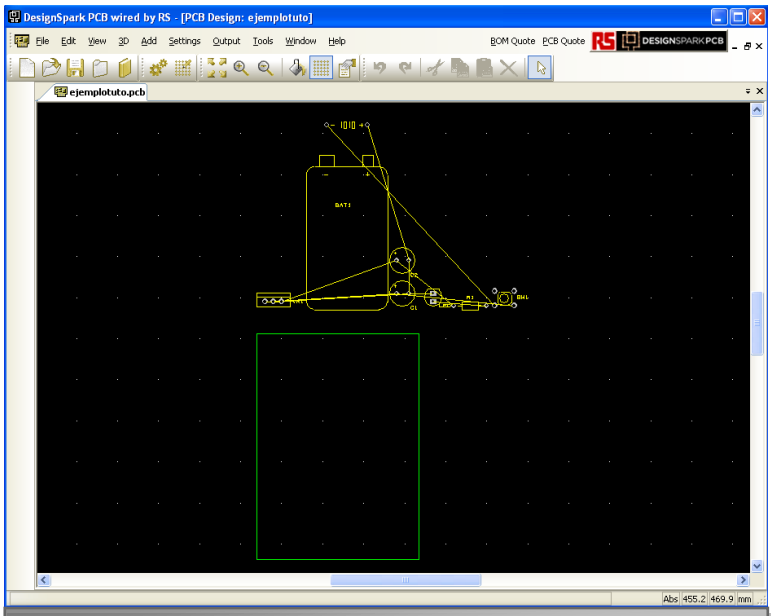


Figura 61: Vista del escritorio de creación de PCBs

Ahora se procede a colocar los componentes dentro de la placa de manera que se crucen las menos vías posibles y que esté todo en orden y más o menos alineado. Se puede colocar la batería a la izquierda, el pulsador y el LED arriba a la derecha y los demás componentes en el espacio restante. Quedaría más o menos como en la Figura 61.

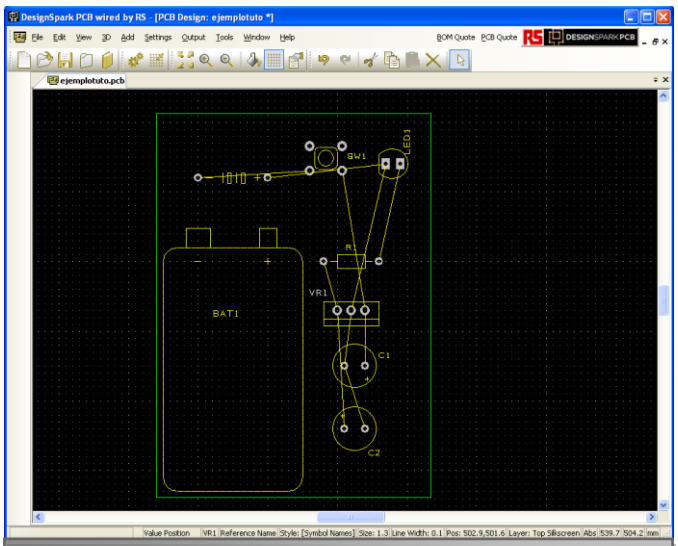


Figura 62: Componentes situados en la placa

5.2.2. Creación de pistas

Después de colocar los componentes ya se puede proceder a que el programa haga las pistas él sólo. En la Barra de Menú, en ***Tools -> Auto Route Nets -> All Nets***. Se desplegará un cuadro con numerosas opciones. En principio se dejan todas como están por defecto.

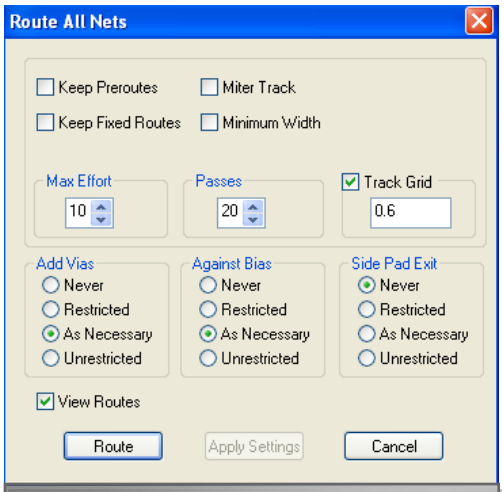


Figura 63: Opciones de la placa

Clic en ***Route*** y se ve cómo va realizando las pistas el programa solo. Cuando termina el proceso aparece el cuadro de la Figura 63. Esto quiere decir que ha podido realizar todas las pistas sin problemas.

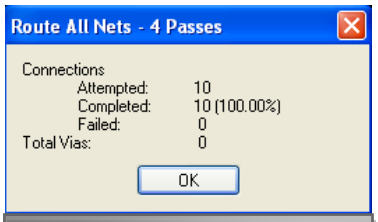


Figura 64: Pistas OK

Clic en **OK** y debería quedar el circuito similar al de la Figura 64.

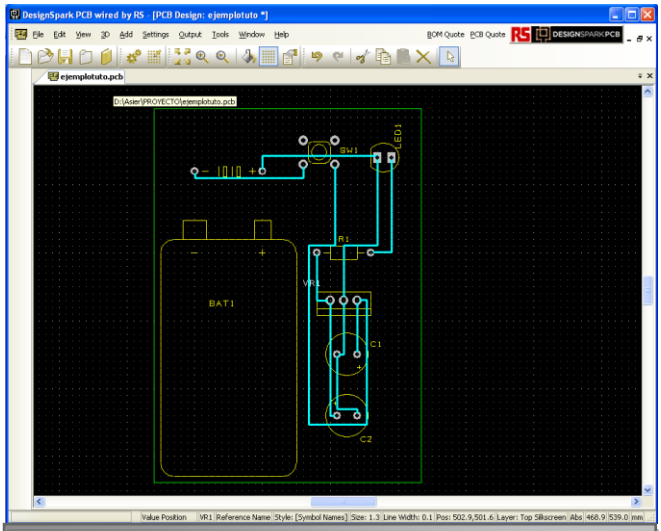


Figura 65: Pistas ruteadas

Si por alguna razón, cuando se han creado las pistas puede que alguna conexión que no se haya hecho correctamente, para saber cuál es hay que ir a ***Output -> Reports*** y seleccionar ***Net Completion***. Esto abrirá un documento de texto donde indicará las conexiones que no se han podido realizar, para que se tenga la posibilidad de realizarlas manualmente. Además se indicará datos de dónde se ha guardado el archivo, fecha, hora, etc.

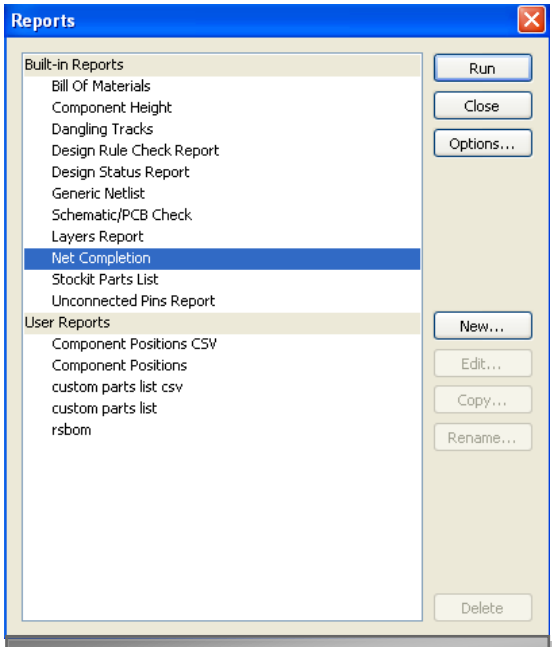


Figura 66: Ventana de *Reports*

En este caso como se han realizado todas las conexiones correctamente no informará de ningún error, como se puede ver en la Figura 66.

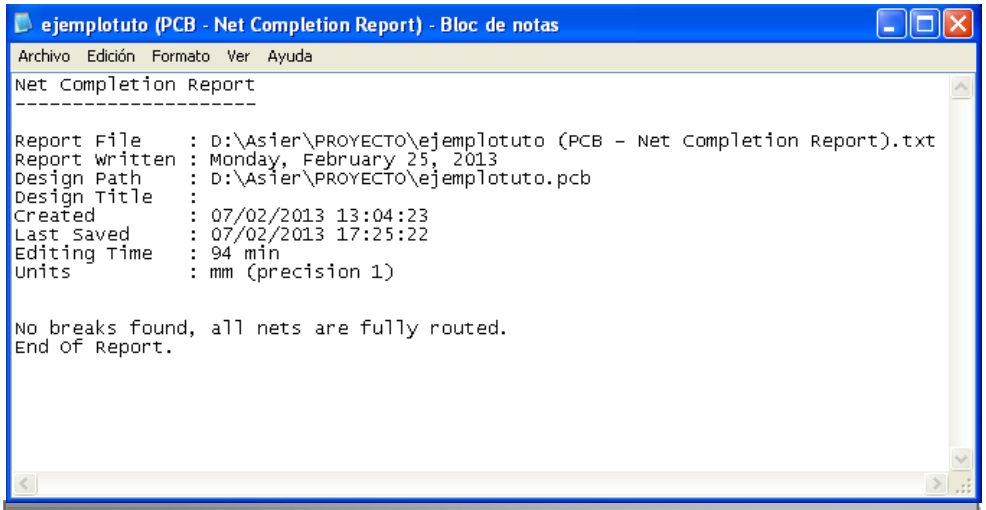


Figura 67: Resultados de ***Net Completion***

5.2.3. Edición de una pista

Al realizar las pistas automáticamente el programa las realiza con una anchura de 0,4 mm. Esto es así porque el programa tiene unas condiciones establecidas por defecto y entre ellas está la de anchura de pista. Para cambiar la esta anchura de pista, simplemente hay que pinchar con el botón derecho encima de una pista e ir a propiedades y ahí cambiarle la anchura, por ejemplo a 0,8 mm. La anchura de pista es bastante importante en el diseño, ya que según la corriente que pase necesitaremos más o menos anchura. En este caso con 0,7-0,8mm es más que suficiente.

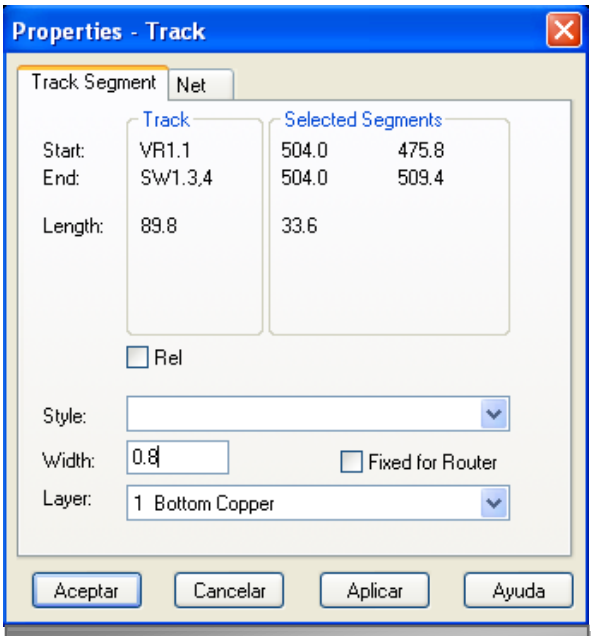


Figura 68: Propiedades de Track

En la Figura 68 se aprecia cómo ha cambiado la pista seleccionada.

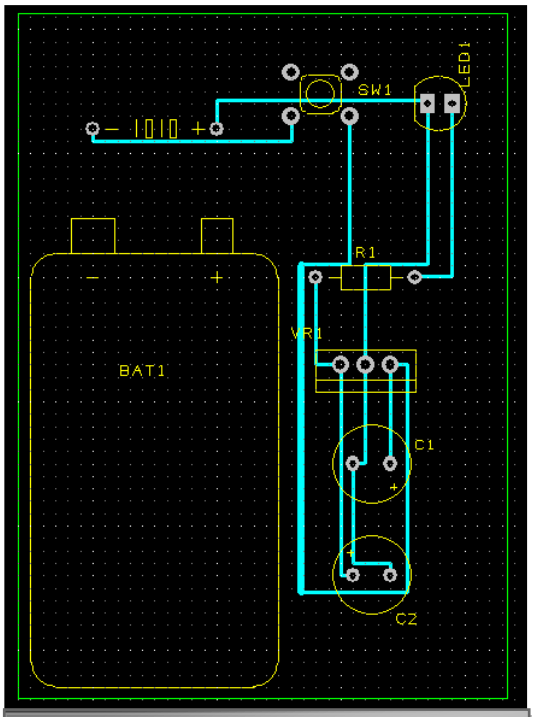


Figura 69: Anchura de pista

5.2.4. Edición de todas las pistas

Esto sirve para cambiar la anchura de una sola pista. Pero lo interesante es poder realizarlas todas a la vez, sin necesidad de ir una por una, con la anchura que se quiera cuando le demos a ***Autorute Nets***. Para ello hay que crear un nuevo tipo de ***Track***, asociarla a una ***Net*** y finalmente establecerlo como valor por defecto. Para conseguir esto hay que ir a la Barra de Menú y dirigirse a ***Design Technology***.

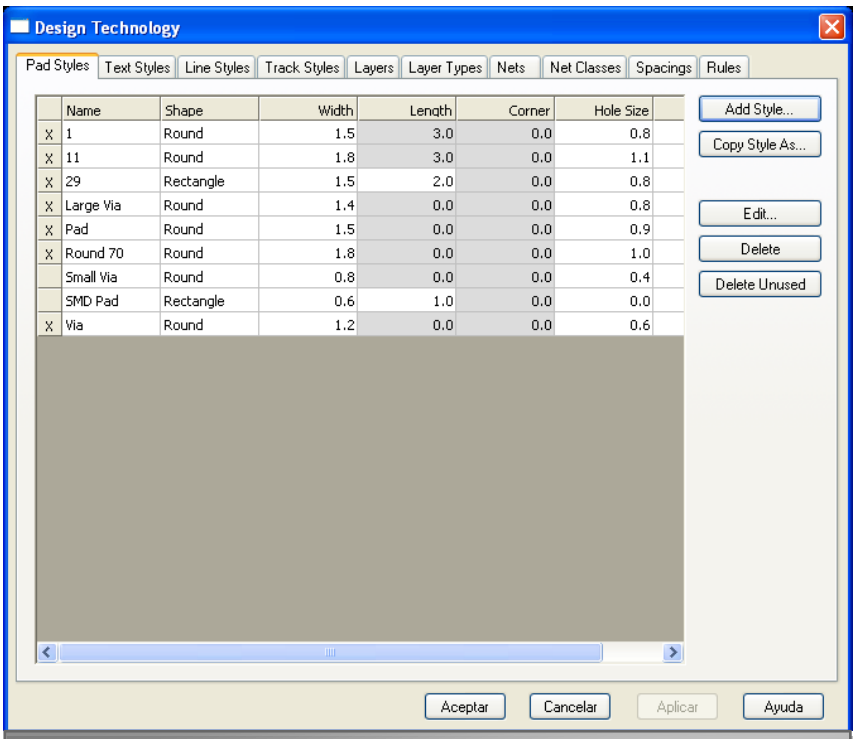


Figura 70: Ventana de Design Technology

Dentro de este cuadro hay que ir a ***Track Styles*** y pinchar en ***Add Style:***

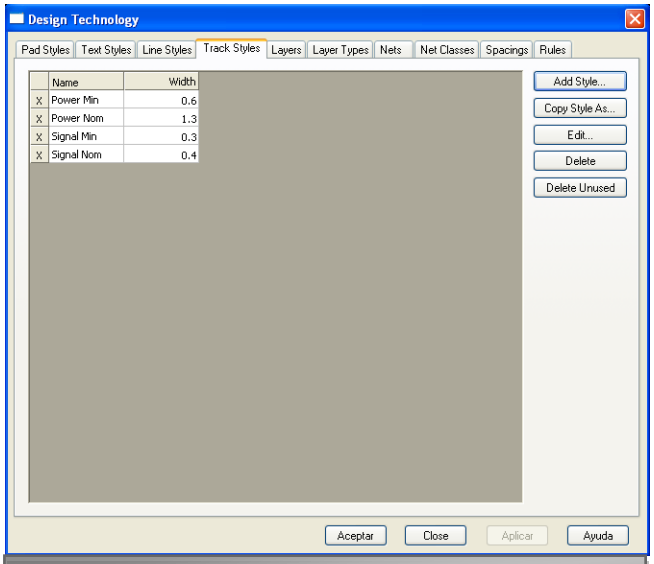


Figura 71: Pestaña de Track Styles

Dentro de ***Add Style*** se le asigna un nombre al tipo de ***Track*** y la anchura que se desee, por ejemplo 0,7mm. Se pincha ***OK*** y ya aparece en la lista de tracks.

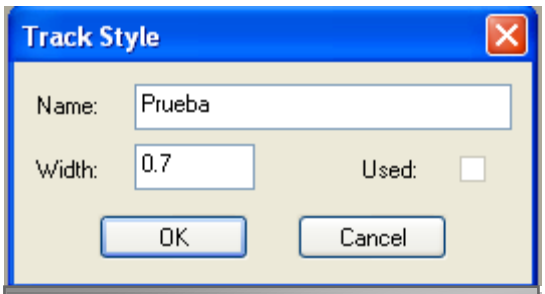


Figura 72: Datos de Track a crear

Ahora hay que asignarle un tipo de Net. Dentro del mismo cuadro de ***Design Technology***, hay que dirigirse a la pestaña ***Nets Classes*** y añadir un tipo de Net pinchando en ***Add…,*** y aquí hay que asignarle un nombre a la net y asignarle los tipos de track. Se observa cómo se puede elegir entre un mínimo y un nominal en el estilo de pista. Esta es una opción interesante en el momento de hacer el enrutado, ya que se le está ordenando al programa que intente enrutar con la distancia nominal, pero si no es posible, que emplee la mínima.

En este caso se elige como nombre "Prueba Net" y los tracks se escogen los de "Prueba", previamente configurados. Ahora se selecciona ***OK*** y se ve como se añade a la lista.

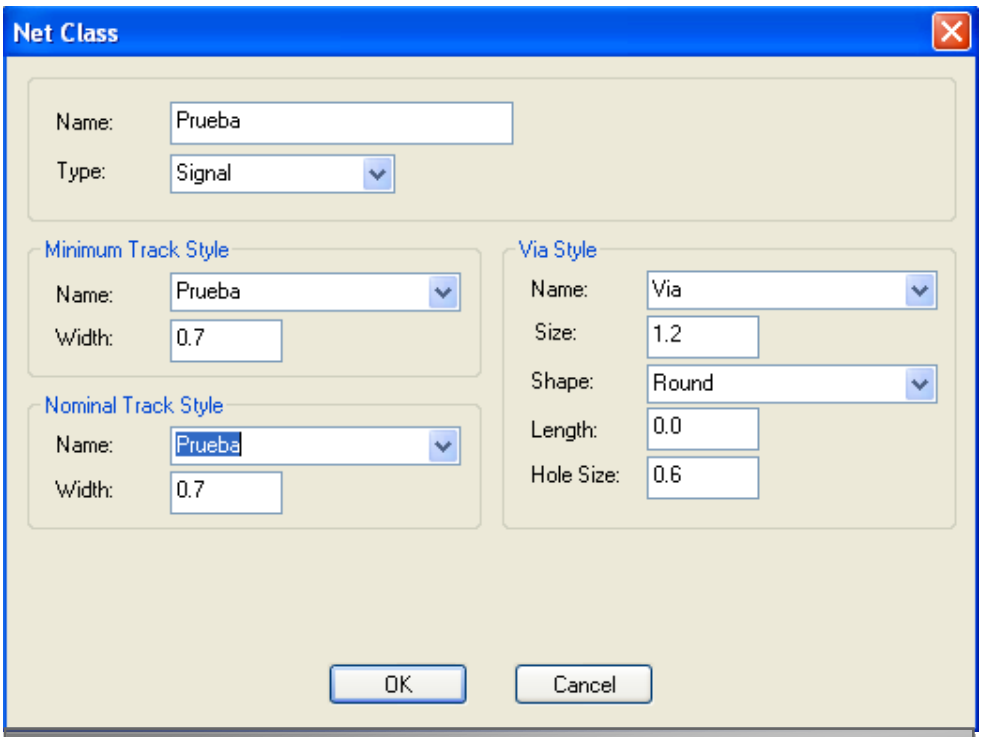


Figura 73: Condiciones de la Net Class a crear

El tipo de vía se mantiene como está por defecto. Como es lógico también se puede crear un tipo de vía al gusto del usuario, exactamente igual que como en las pistas pero en la pestaña de ***Pad Styles***. Pero esta opción no se aplica ahora ya que no es necesario en este ejemplo.

Ahora hay que dirigirse a la pestaña de ***Nets*** y seleccionar la opción de la derecha de ***ShowAuto-Named Nets*** para que aparezcan todas las nets.

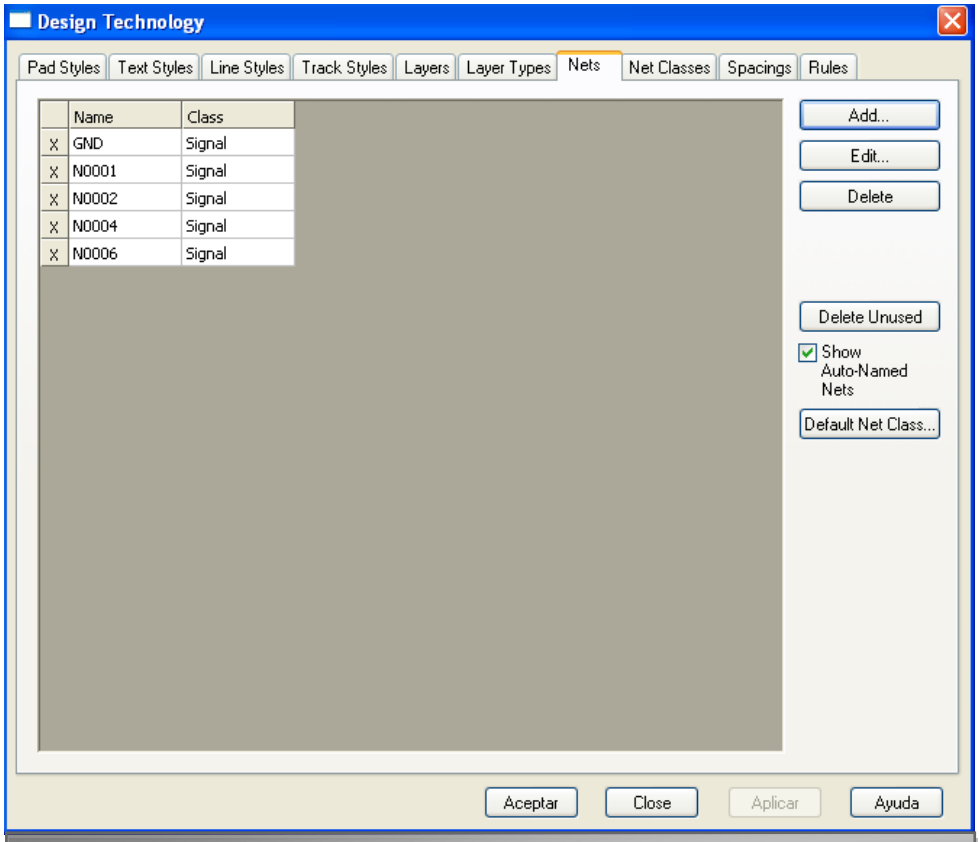


Figura 74: Pestaña de Nets

Ahora se elige una net cualquiera y se pincha en ***Edit…*** y en ***Net Class*** se selecciona la que se ha creado anteriormente.

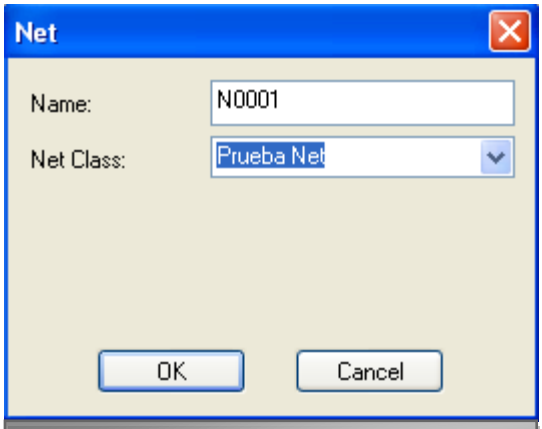


Figura 75: Asignar Net Class

Una vez que se ha pinchado en ***OK*** se observa como esa Net ha cambiado. Para que afecte a todas hay que pinchar con el botón derecho encima de cualquier net y clicar en ***Apply To Column***, y se cambiarán todas las nets.

Clic en ***Aceptar*** y ahora sólo queda cambiar los valores por defecto. En ***Settings -> Defaults*** hay que dirigirse a la pestaña de ***Tracks*** y ahí modificar la ***Track Style.***

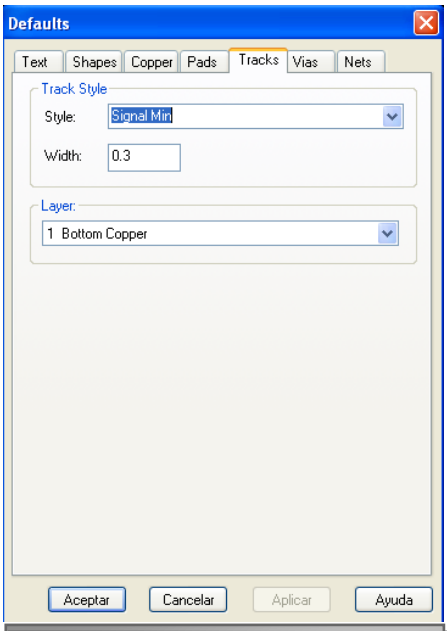


Figura 76: Track por defecto

En este cuadro se selecciona en la pestaña de ***Tracks*** la track creada "Prueba" y se aceptan los cambios.

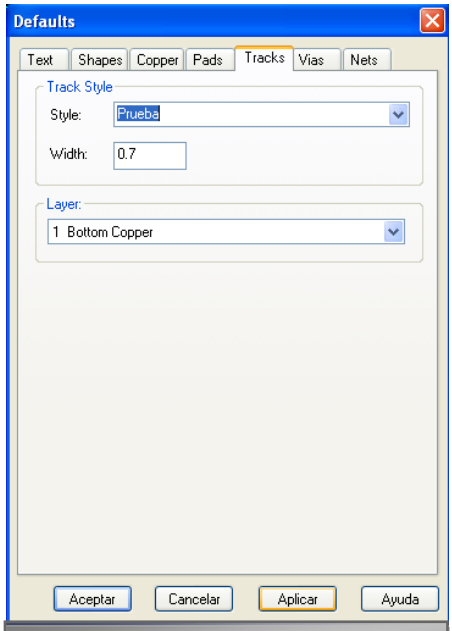


Figura 77: Track "Prueba"

Ahora sólo queda el último paso, rutear las pistas de nuevo. Se hace como en el apartado de ***Autorute Nets***, y ahora sí el programa realiza las pistas a la anchura de 0,7 como se quería.

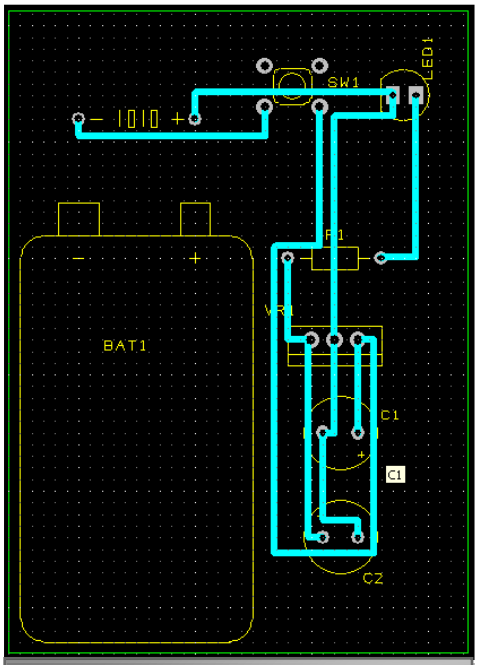


Figura 78: Pistas a 0,7 mm

5.2.5. Pistas con ángulos rectos

Como se aprecia en la Figura 77 las pistas tienen ángulos rectos, algo que es conveniente evitar. Como siempre se puede hacer manualmente y automáticamente. De manera manual, simplemente con hacer doble clic en la esquina de la pista y mover el ratón ya se puede modificar este ángulo.

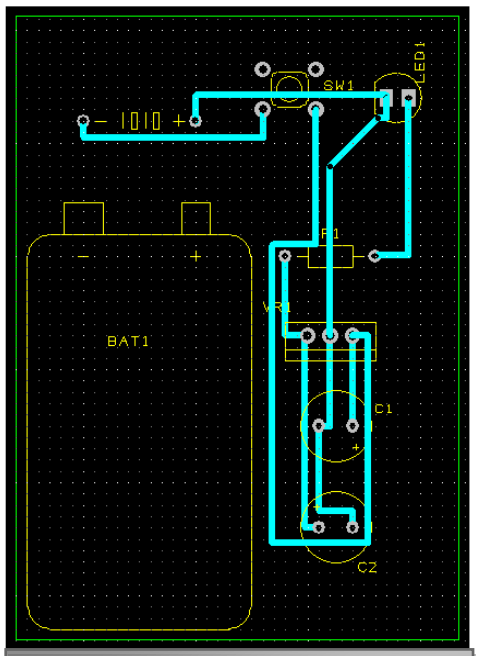


Figura 79: Ángulo modificado

Como en otras ocasiones, lo interesante y cómodo es que el programa lo realice de manera automática. Para ello hay que dirigirse al menú de ***Tools -> Auto Route Nets -> All Nets*** y seleccionar ***Miter track*** y dentro de ésta activar la opción ***Any angle***, y dejar los demás valores por defecto (Figura 79).

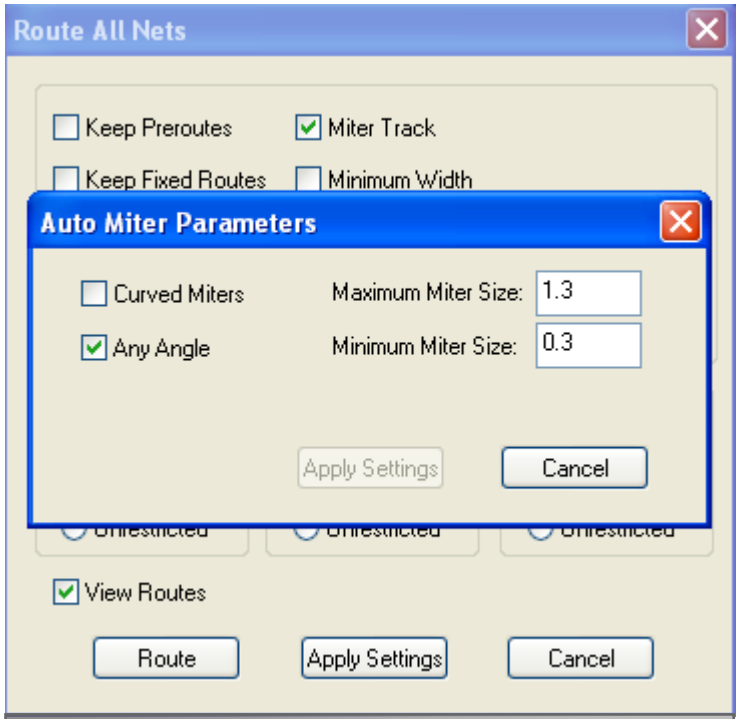


Figura 80: Activar ruteado con ángulos

Se aplican los cambios (***Apply Settings***), se rutea de nuevo y se obtiene el siguiente resultado donde se puede apreciar como no hay ninguna pista con ángulo recto. Igualmente se puede modificar el ángulo al gusto del usuario haciendo doble clic sobre el mismo ángulo.

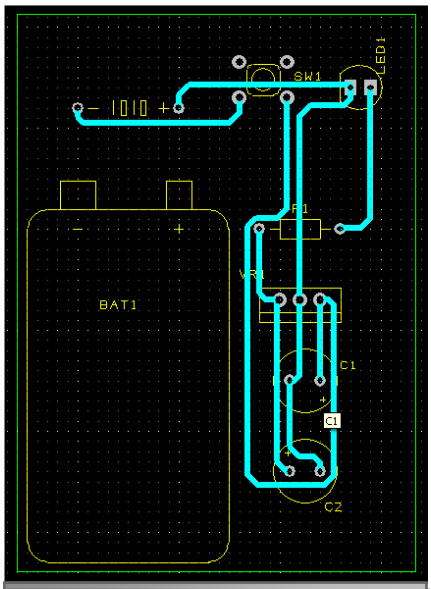


Figura 81: Placa sin ángulos rectos

5.2.6. Edición de las vías

En este caso como la placa era a simple cara y había pocas pistas, el programa no ha realizado ninguna vía. Una vía es un pad que comunica la cara superior (top) con la inferior (bottom).

Si realizamos una placa a doble cara y con muchas pistas el programa realizará las vías que crea oportuno, muchas veces de manera innecesaria. Para evitar que realice más vías de las estrictamente necesarias, hay que limitárselo en el menú de ***Tools -> Auto Route Nets -> All Nets.*** En el apartado de ***Add Vias*** hay que seleccionar la opción de ***Restricted***. Así el programa hará el ruteo de las pistas de manera que realice menos vías. Si se realiza algún ejemplo que se necesiten vías se puede apreciar cómo disminuye el número de vías considerablemente en la placa.

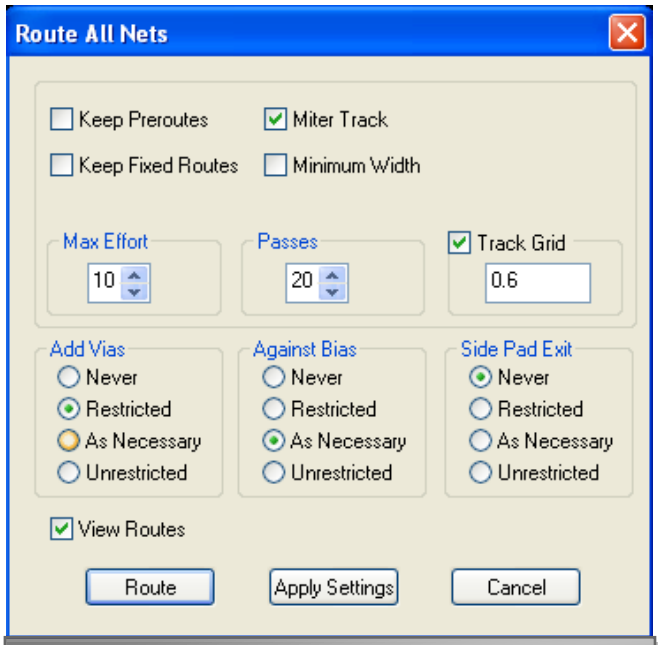


Figura 82: Restringir vías

5.2.7. Vista en 3D

Ahora que ya están realizadas las pistas es un buen momento para utilizar una opción bastante curiosa que brinda el programa. La vista 3D. En la barra de menú donde pone 3D, se sigue la ruta ***3D => 3D View*** y aparece una vista en 3D de la placa que se está realizando.

Es una opción bastante visual, pero también tiene su utilidad ya que permite al usuario hacerse una idea de cómo van a quedar los componentes al final del diseño y modificarlo si lo cree necesario.

En las figuras 82 y 83 se aprecia esta vista 3D, por la parte delantera (top) y trasera (bottom), respectivamente.

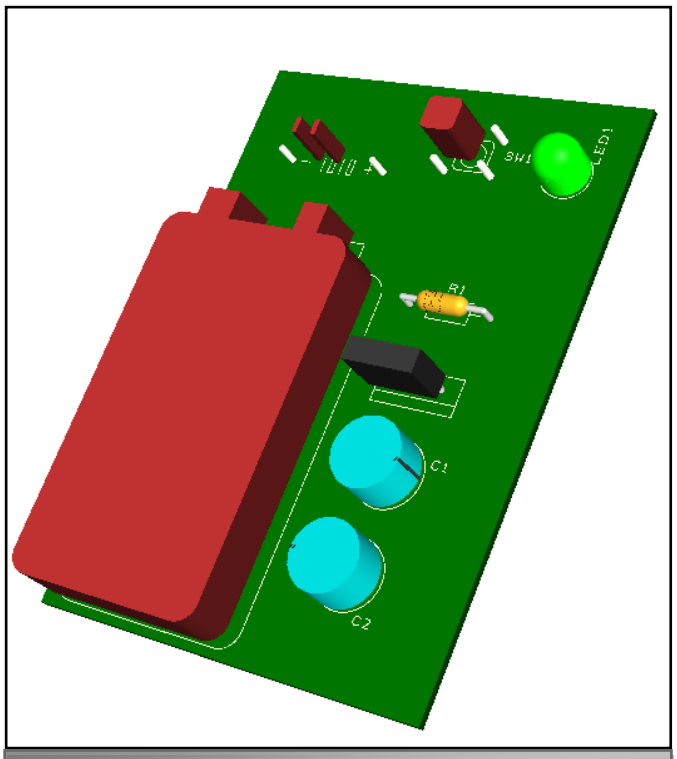


Figura 83: Vista de la cara Top en 3D

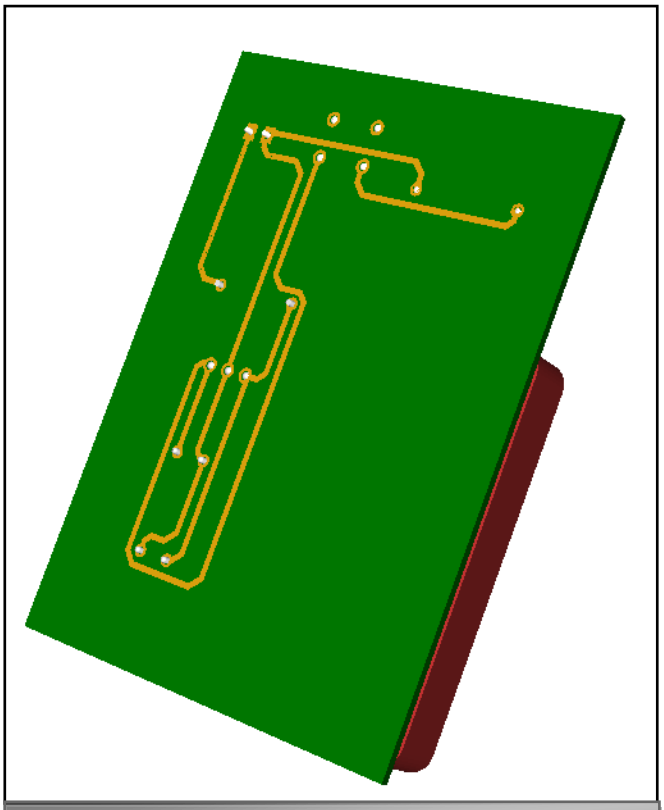


Figura 84: Vista de la cara Bottom en 3D

5.3. Últimos pasos

Aunque la placa está casi completada, antes de dar por terminada la placa hay que realizar una serie de pasos previos que mejorará el funcionamiento del sistema.

5.3.1. Añadir capa de masa

Una vez realizadas las pistas correctamente hay que hacer un último paso antes de tener la placa completamente diseñada. Hay que incluir una capa de masa que ocupe toda la superficie de la placa para estabilizar todo el circuito. Si la placa fuera a doble cara y queremos más estabilidad en el circuito, por ejemplo cuando se traten de señales de alta frecuencia, es recomendable hacer una capa de masa en las dos caras, por arriba y por abajo.

Esto se consigue yendo a la barra de menú en ***Add -> Copper Pour Area -> Rectangle***, porque la placa tiene forma rectangular. Una vez seleccionada hay que realizar un rectángulo del tamaño de la placa justo encima de esta.

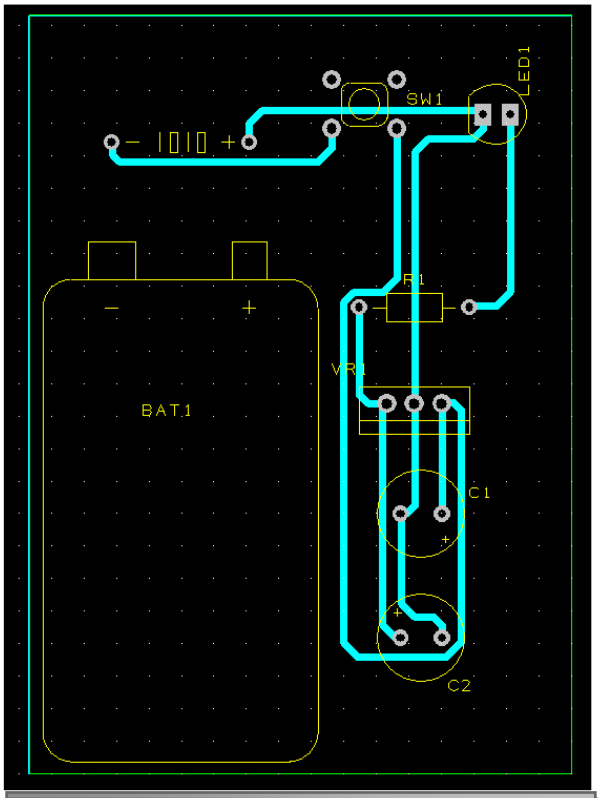


Figura 85: Rectángulo de Copper Pour

Se aprecia cómo ha cambiado el color del contorno de la placa. Ahora se selecciona un segmento de este rectángulo y con el botón derecho se va a ***Pour Copper***.

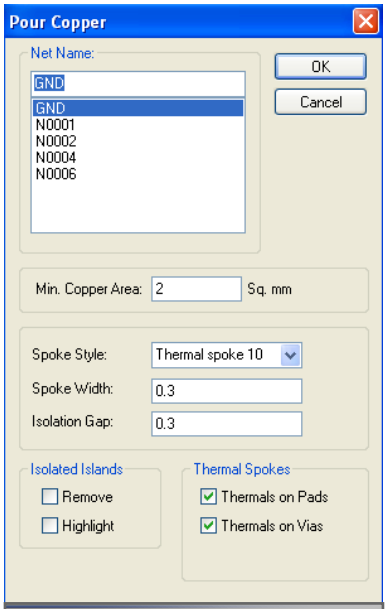


Figura 86: Opciones de Pour Copper

Y en las opciones se selecciona **GND**. Las demás opciones se dejan como están por defecto y pulsa en **OK**. La placa debería quedar como aparece en la Figura 86.

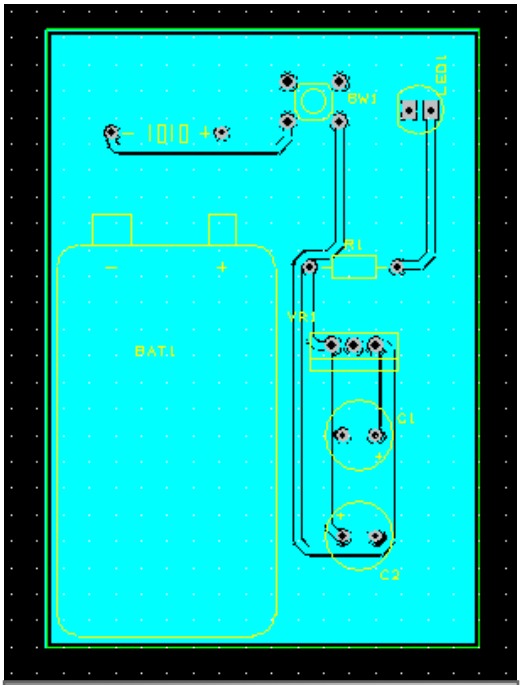


Figura 87: Capa de masa añadida

Se puede apreciar como los terminales que estaban a masa en el circuito ahora están unidos a la capa de masa.

Y en la Vista 3D también se aprecia esta capa.

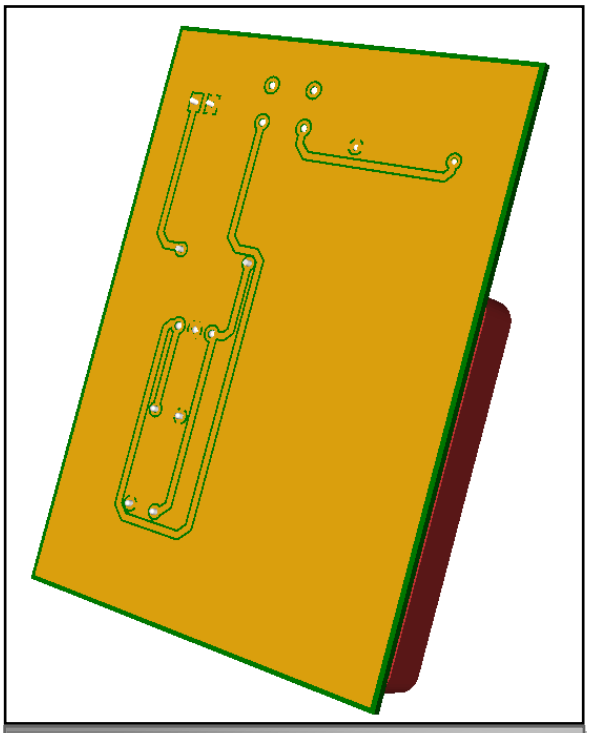
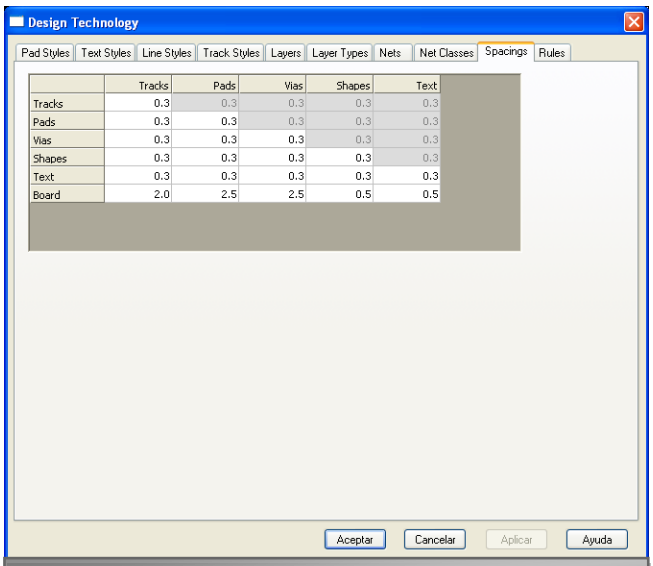


Figura 88: Capa de masa en 3D

5.3.2. Editar la capa de masa

Como en el caso de las pistas, el programa realiza la capa de masa según unas condiciones por defecto que tiene establecidas. La separación entre las pistas, pads y vías está establecida en ***Settings -> Design Technology***, en la pestaña de ***Spacings***.



	Tracks	Pads	Vias	Shapes	Text
Tracks	0.3	0.3	0.3	0.3	0.3
Pads	0.3	0.3	0.3	0.3	0.3
Vias	0.3	0.3	0.3	0.3	0.3
Shapes	0.3	0.3	0.3	0.3	0.3
Text	0.3	0.3	0.3	0.3	0.3
Board	2.0	2.5	2.5	0.5	0.5

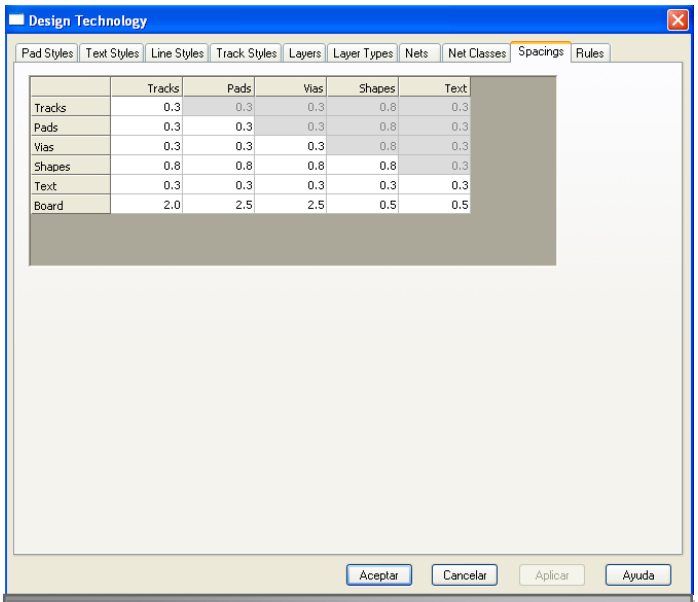
Figura 89: Pestaña Spacings

En la tabla de la Figura 88 aparece una tabla que nos indica el espaciado entre todos los elementos de la placa, por ejemplo entre tracks, entre pads y tracks, entre vías y tracks, etc.

Pero lo que interesa es la fila de ***Shapes***, que hace referencia a esta capa de masa.

Como se aprecia en la Figura 88, el espaciado entre todos los elementos y la capa de masa es de 0,3mm, que puede ser un espacio escaso en según qué diseños. Se puede aumentar este espacio modificando estos parámetros. Esto es muy conveniente para asegurarse de que las soldaduras de los componentes no se unen a la capa de masa del circuito.

Se modifican los parámetros y se pone, por ejemplo, un espaciado de 0,8 mm.



	Tracks	Pads	Vias	Shapes	Text
Tracks	0.3	0.3	0.3	0.8	0.3
Pads	0.3	0.3	0.3	0.8	0.3
Vias	0.3	0.3	0.3	0.8	0.3
Shapes	0.8	0.8	0.8	0.8	0.3
Text	0.3	0.3	0.3	0.3	0.3
Board	2.0	2.5	2.5	0.5	0.5

Figura 90: Espaciado modificado

Una vez modificado el espaciado hay que seleccionar la capa, y con el botón derecho ir a ***Pour Copper*** y seleccionar **GND**. Se puede apreciar cómo ha variado el espaciado (Figura 90).

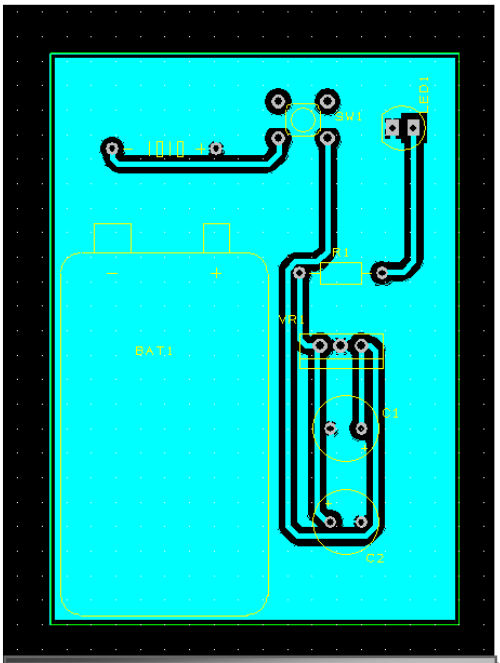


Figura 91: Capa modificada

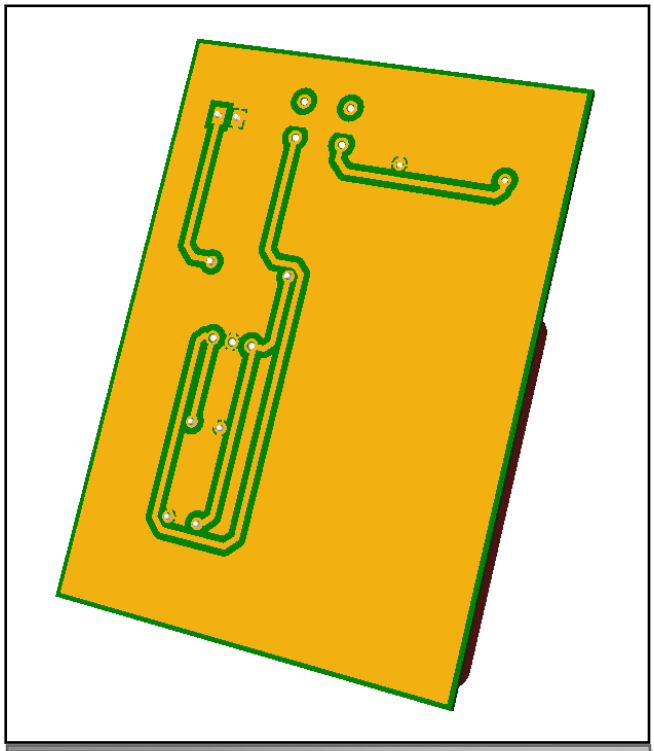


Figura 92: Capa modificada en 3D

Ahora ya sí el diseño de la placa ha finalizado, y solamente en el archivo **.pcb** se tiene toda la información para poder realizar la placa, ya sea a la empresa que nos realice la placa o hacerlo el usuario mismo.

Recursos electrónicos del Anexo I

[1] Página oficial DesignSpark PCB

http://www.designspark.com/page/designspark-pcb-home-page

[2] Tutorial básico DesignSpark PCB

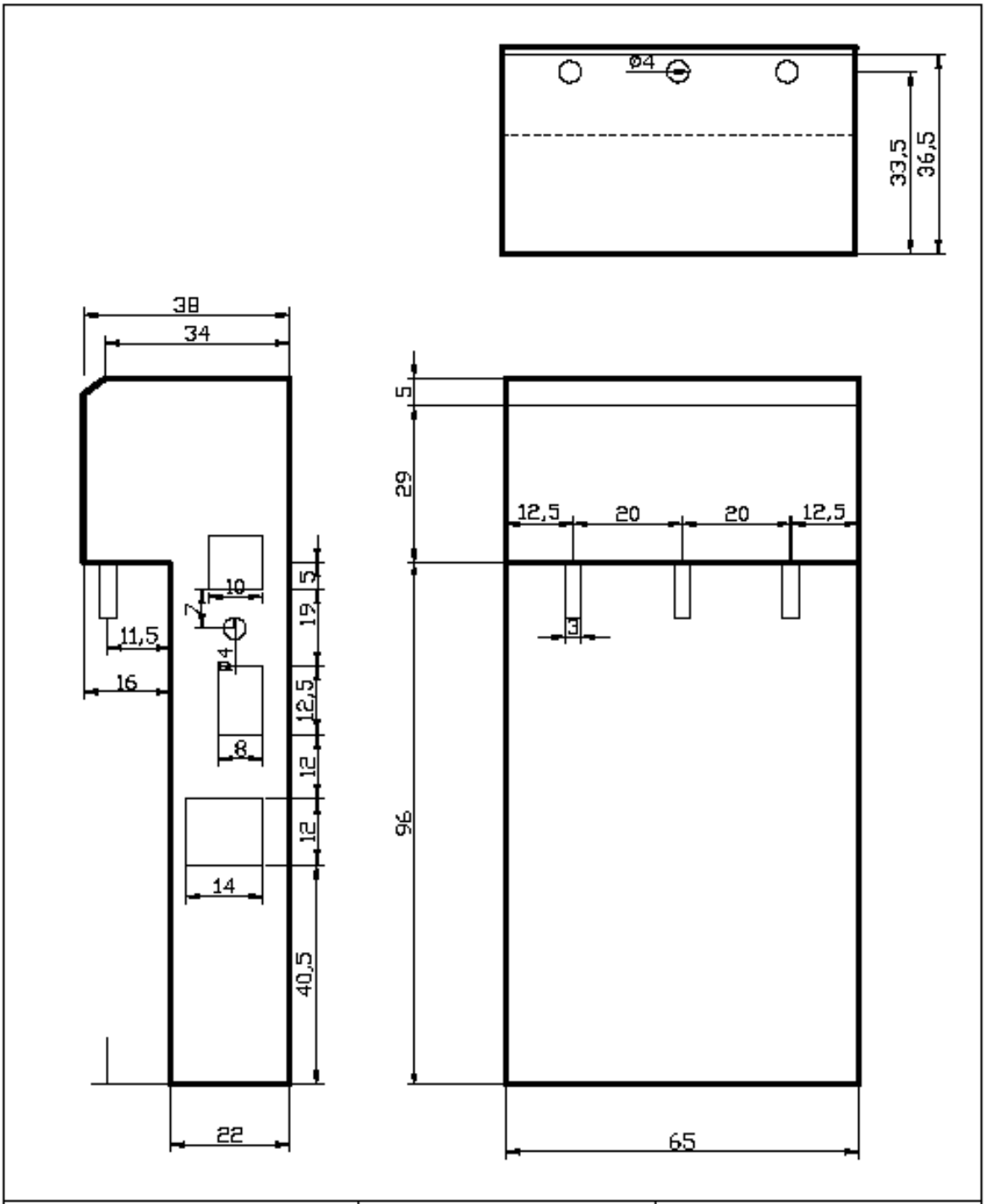
http://www.neoteo.com/designspark-pcb-facil-gratis

ANEXO II: PROGRAMA DEFINITITIVO

```
//*Programa final*//
#include<htc.h>                         //Incluimos librería del micro a usar
__CONFIG(WRT_OFF & WDTE_OFF & PWRTE_OFF & FOSC_INTOSCIO & LVP_OFF &
CCPMX_RB3 & BOREN_OFF);                 //Salida del PWM por RB3
#define _XTAL_FREQ 8000000              //Oscilador Interno de 8MHZ.necesario definirlo así
                                        //para los delays

unsigned char cont=0;
unsigned char velocidad[6]={0x25, 0x32, 0x3E, 0x4A, 0x57, 0x00};
                                        //Definimos el   array de ciclos de trabajo
void main(void){

        TRISA=0xFF;                     //Puerto A configurado como entrada
        TRISB=0xF7;                     //Puerto B configurado como entrada menos
                                        //RB3como salida
        GIE=1;                          //Interrupciones globales activadas
        PEIE=1;                         //Activa interrupciones por periféricos
        RBIE=0;                         //Desactiva interrupciones en RB<7:4>
        ANSEL=0;                        //Todas las entradas digitales
        SCS1=0;                         //T1OSC usado como sistema de reloj
        SCS0=1;
        IRCF2=1;                        //Configura oscilador interno a 8MHz
        IRCF1=1;
        IRCF0=1;
        CCP1M3=1;                       //Configurar el pic en modo PWM
        CCP1M2=1;
        CCP1M1=0;
        CCP1M0=0;
        CCPR1L=0x00;                    //Inicializamos ciclo de trabajo a cero para que al
                                        //encender esté parado
//TIMER0
        T0CS=0;                         //TIMER0 en modo contador
        T0SE=0;                         //Por flanco de subida
        PSA=1;                          //Prescaler asignado a WTD
        OPTION_REGbits.PS2=0;           //Prescaler de 1:1
        OPTION_REGbits.PS1=0;
        OPTION_REGbits.PS0=0;
        TMR0IE=1;                       //Habilitamos interrupción en TIMER0
        TMR0=100;                       //Inicializar TMR0 a 100.

//TIMER2
        T2CONbits.T2CKPS1=0;            //Prescaler 1:4
        T2CONbits.T2CKPS0=1;
        T2CONbits.TMR2ON=1;             //Activamos Timer2
        INTEDG=0;                       //Interrupción por flanco de bajada en RB0
        INT0IE=1;                       //Habilitar interrupción en RB0
        PR2=0x7C;                       //Frecuencia del PWM a 4KHz
```

```
while (1) {
                CLRWDT();
                __delay_ms(50);}
        }
static void interrupt isr (void) {
__delay_ms(50);                         //Retardo para evitar rebotes en el pulsador
if (INT0IF==1){
                INT0IF=0;               //Poner a cero la bandera de interrupción
                CCPR1L = velocidad[cont];   //Igualamos el ciclo de trabajo con el valor del
                                        //array
                        cont++;         //Incrementamos cont
                        if (cont==6){   //Si el cont es igual a 6, es que ha pasado por
                                        //todas las velocidades y tiene que volver a
                                        //empezar
                        cont=0;         //Poner cont a cero
                                }

                }
}
```

ANEXO III: PLANO DE LA CARCASA



Universidad Pública de Navarra / Nafarroako Unibertsitate Publikoa	E.T.S.I.I.T. / INGENIERO TECNICO INDUSTRIAL E.	DEPARTAMENTO: DEPARTAMENTO DE INGENIERIA ELECTRICA Y ELECTRONICA
PROYECTO: DISPOSITIVO ROTATORIO PORTATIL PARA FIBRA OPTICA		REALIZADO: MIRANDA HERRERO, ASIER FIRMA:
PLANO: ALZADO, PLANTA Y PERFIL DE LA CARCASA	FECHA: 23/4/2013	ESCALA: 1:1 / N° PLANO: 1

PIC16F87/88
Data Sheet

18/20/28-Pin Enhanced Flash Microcontrollers with nanoWatt Technology

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Trademarks

Printed on recycled paper.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip received ISO/TS-16949:2002 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company's quality system processes and procedures are for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

PIC16F87/88

18/20/28-Pin Enhanced Flash MCUs with nanoWatt Technology

Low-Power Features:

- Power-Managed modes:
  - Primary Run: RC oscillator, 76 µA, 1 MHz, 2V
  - RC_RUN: 7 µA, 31.25 kHz, 2V
  - SEC_RUN: 9 µA, 32 kHz, 2V
  - Sleep: 0.1 µA, 2V
- Timer1 Oscillator: 1.8 µA, 32 kHz, 2V
- Watchdog Timer: 2.2 µA, 2V
- Two-Speed Oscillator Start-up

Oscillators:

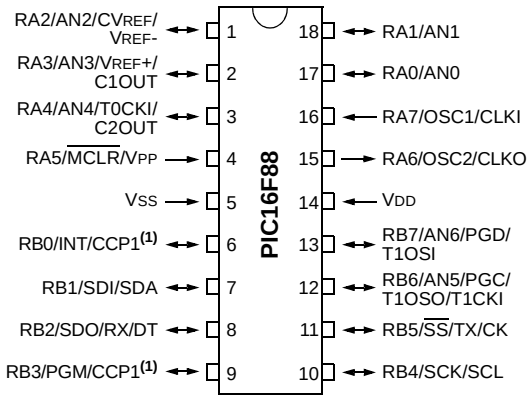
- Three Crystal modes:
  - LP, XT, HS: up to 20 MHz
- Two External RC modes
- One External Clock mode:
  - ECIO: up to 20 MHz
- Internal oscillator block:
  - 8 user selectable frequencies: 31 kHz, 125 kHz, 250 kHz, 500 kHz, 1 MHz, 2 MHz, 4 MHz, 8 MHz

Peripheral Features:

- Capture, Compare, PWM (CCP) module:
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- 10-bit, 7-channel Analog-to-Digital Converter
- Synchronous Serial Port (SSP) with SPI™ (Master/Slave) and I^2C™ (Slave)
- Addressable Universal Synchronous Asynchronous Receiver Transmitter (AUSART/SCI) with 9-bit address detection:
  - RS-232 operation using internal oscillator (no external crystal required)
- Dual Analog Comparator module:
  - Programmable on-chip voltage reference
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Pin Diagram

18-Pin PDIP, SOIC



Note 1: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

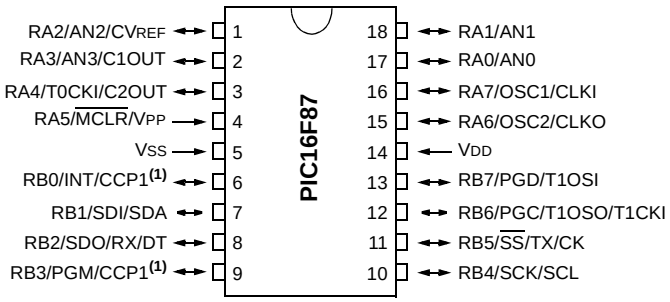
Special Microcontroller Features:

- 100,000 erase/write cycles Enhanced Flash program memory typical
- 1,000,000 typical erase/write cycles EEPROM data memory typical
- EEPROM Data Retention: > 40 years
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Processor read/write access to program memory
- Low-Voltage Programming
- In-Circuit Debugging via two pins
- Extended Watchdog Timer (WDT):
  - Programmable period from 1 ms to 268s
- Wide operating voltage range: 2.0V to 5.5V

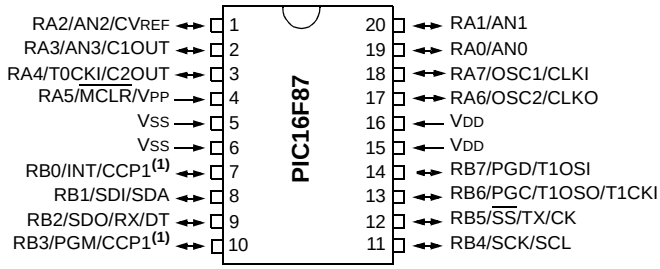
Device	Program Memory		Data Memory		I/O Pins	10-bit A/D (ch)	CCP (PWM)	AUSART	Comparators	SSP	Timers 8/16-bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)							
PIC16F87	7168	4096	368	256	16	N/A	1	Y	2	Y	2/1
PIC16F88	7168	4096	368	256	16	1	1	Y	2	Y	2/1

Pin Diagrams

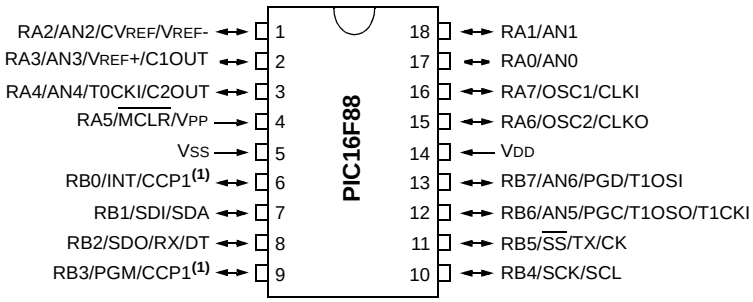
18-Pin PDIP, SOIC



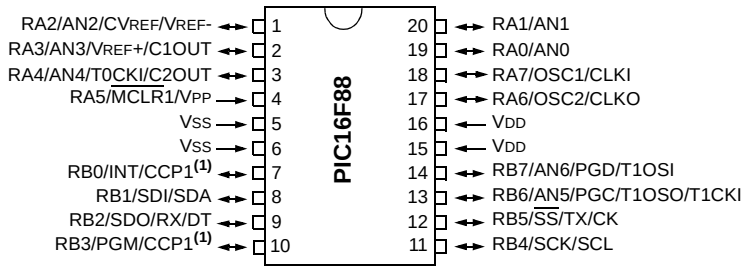
20-Pin SSOP



18-Pin PDIP, SOIC



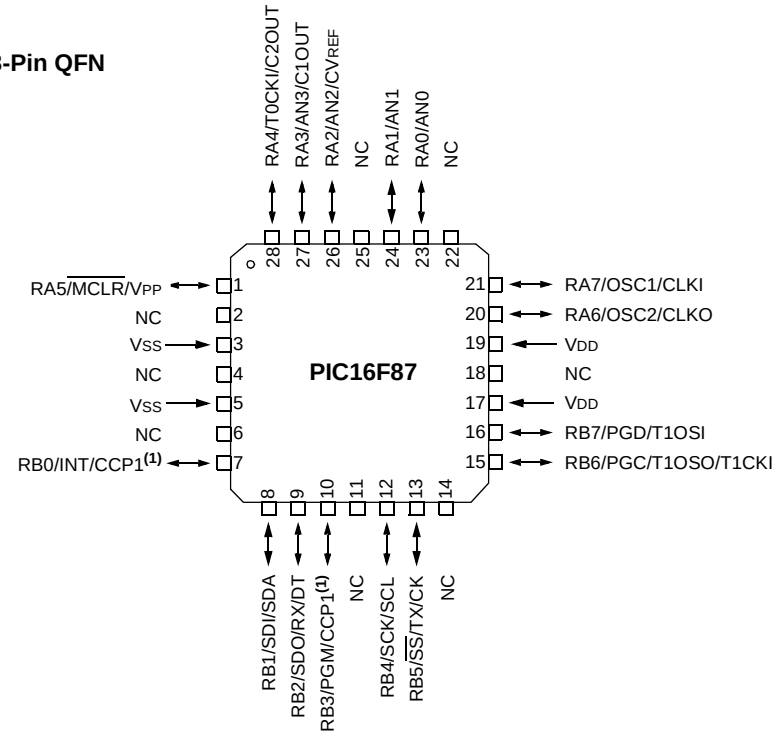
20-Pin SSOP



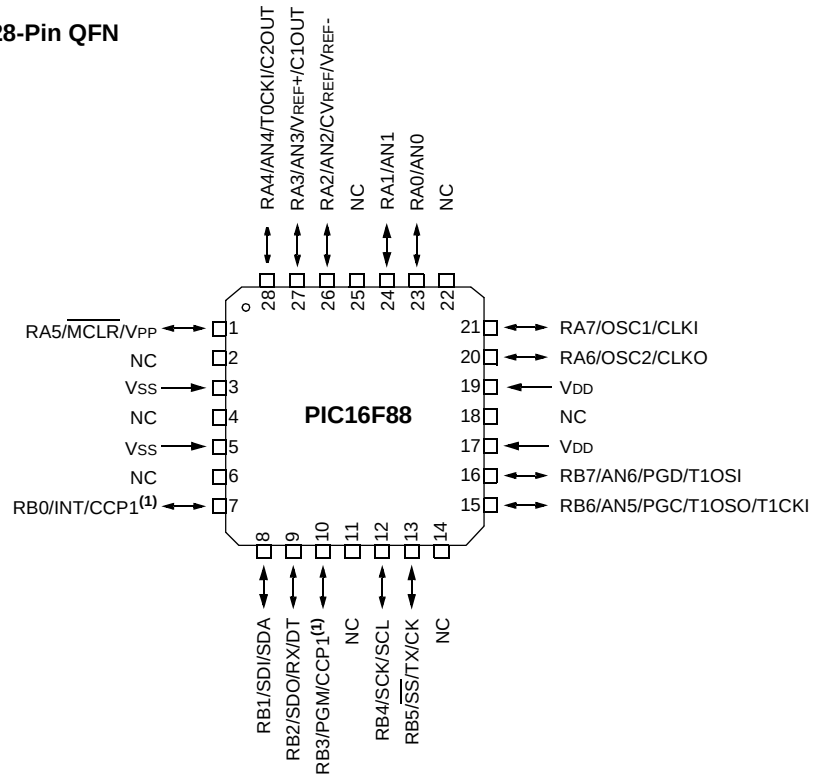
Note 1: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

Pin Diagrams (Cont'd)

28-Pin QFN



28-Pin QFN



Note 1: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

Table of Contents

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com** to receive the most current information on all of our products.

1.0 DEVICE OVERVIEW

This document contains device specific information for the operation of the PIC16F87/88 devices. Additional information may be found in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023) which may be downloaded from the Microchip web site. This Reference Manual should be considered a complementary document to this data sheet and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

The PIC16F87/88 belongs to the Mid-Range family of the PICmicro® devices. Block diagrams of the devices are shown in Figure 1-1 and Figure 1-2. These devices contain features that are new to the PIC16 product line:

- Low-power modes: RC_RUN allows the core and peripherals to be clocked from the INTRC, while SEC_RUN allows the core and peripherals to be clocked from the low-power Timer1. Refer to **Section 4.7 "Power-Managed Modes"** for further details.
- Internal RC oscillator with eight selectable frequencies, including 31.25 kHz, 125 kHz, 250 kHz, 500 kHz, 1 MHz, 2 MHz, 4 MHz and 8 MHz. The INTRC can be configured as a primary or secondary clock source. Refer to **Section 4.5 "Internal Oscillator Block"** for further details.
- The Timer1 module current consumption has been greatly reduced from 20 µA (previous PIC16 devices) to 1.8 µA typical (32 kHz at 2V), which is ideal for real-time clock applications. Refer to **Section 7.0 "Timer1 Module"** for further details.
- Extended Watchdog Timer (WDT) that can have a programmable period from 1 ms to 268s. The WDT has its own 16-bit prescaler. Refer to **Section 15.12 "Watchdog Timer (WDT)"** for further details.
- Two-Speed Start-up: When the oscillator is configured for LP, XT or HS Oscillator mode, this feature will clock the device from the INTRC while the oscillator is warming up. This, in turn, will enable almost immediate code execution. Refer to **Section 15.12.3 "Two-Speed Clock Start-up Mode"** for further details.
- Fail-Safe Clock Monitor: This feature will allow the device to continue operation if the primary or secondary clock source fails by switching over to the INTRC.
- The A/D module has a new register for PIC16 devices named ANSEL. This register allows easier configuration of analog or digital I/O pins.

TABLE 1-1: AVAILABLE MEMORY IN PIC16F87/88 DEVICES

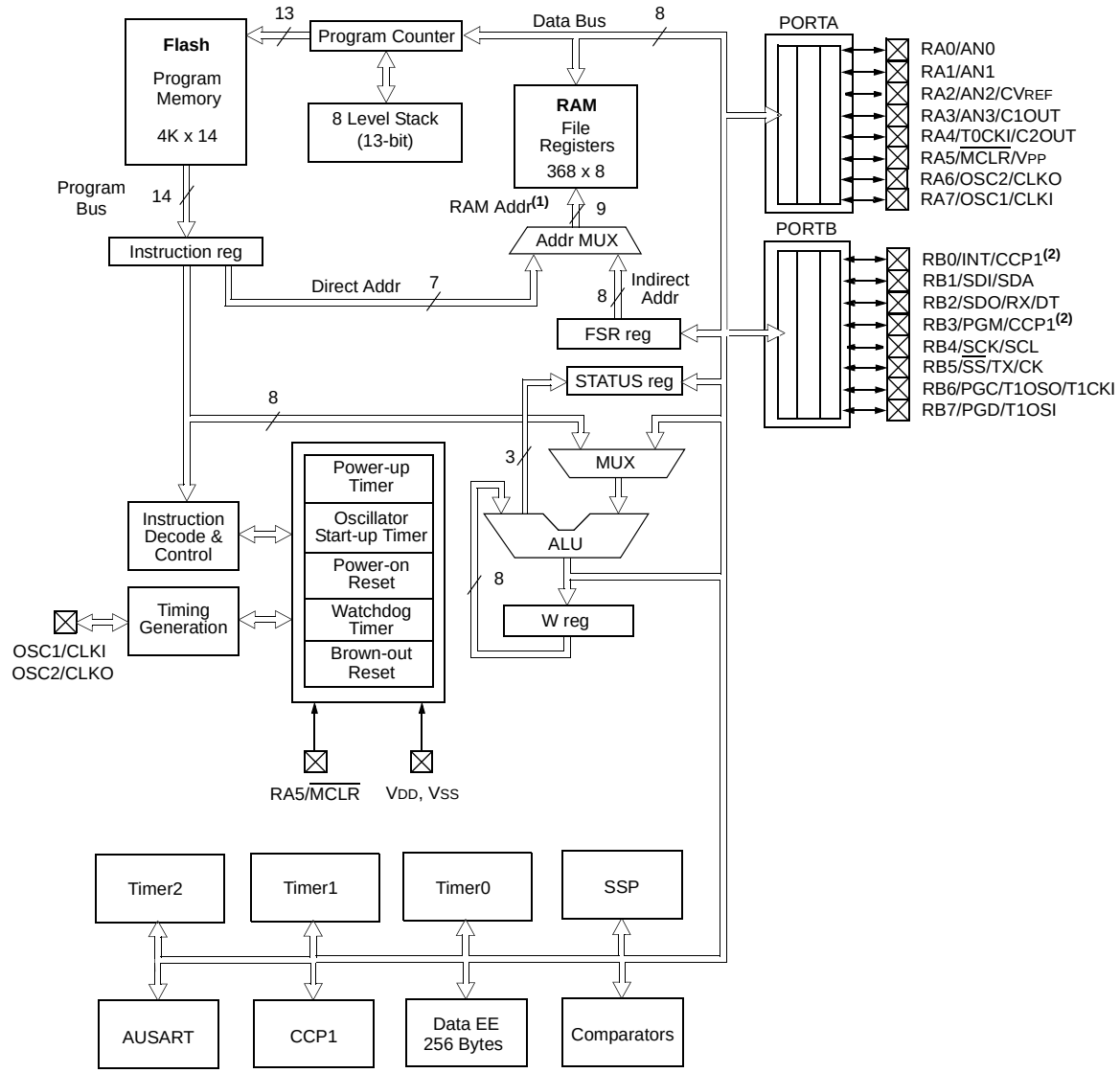
Device	Program Flash	Data Memory	Data EEPROM
PIC16F87/88	4K x 14	368 x 8	256 x 8

There are 16 I/O pins that are user configurable on a pin-to-pin basis. Some pins are multiplexed with other device functions. These functions include:

- External Interrupt
- Change on PORTB Interrupt
- Timer0 Clock Input
- Low-Power Timer1 Clock/Oscillator
- Capture/Compare/PWM
- 10-bit, 7-channel A/D Converter (PIC16F88 only)
- SPI™/I^{2}C™
- Two Analog Comparators
- AUSART
- $\overline{\text{MCLR}}$ (RA5) can be configured as an input

Table 1-2 details the pinout of the devices with descriptions and details for each pin.

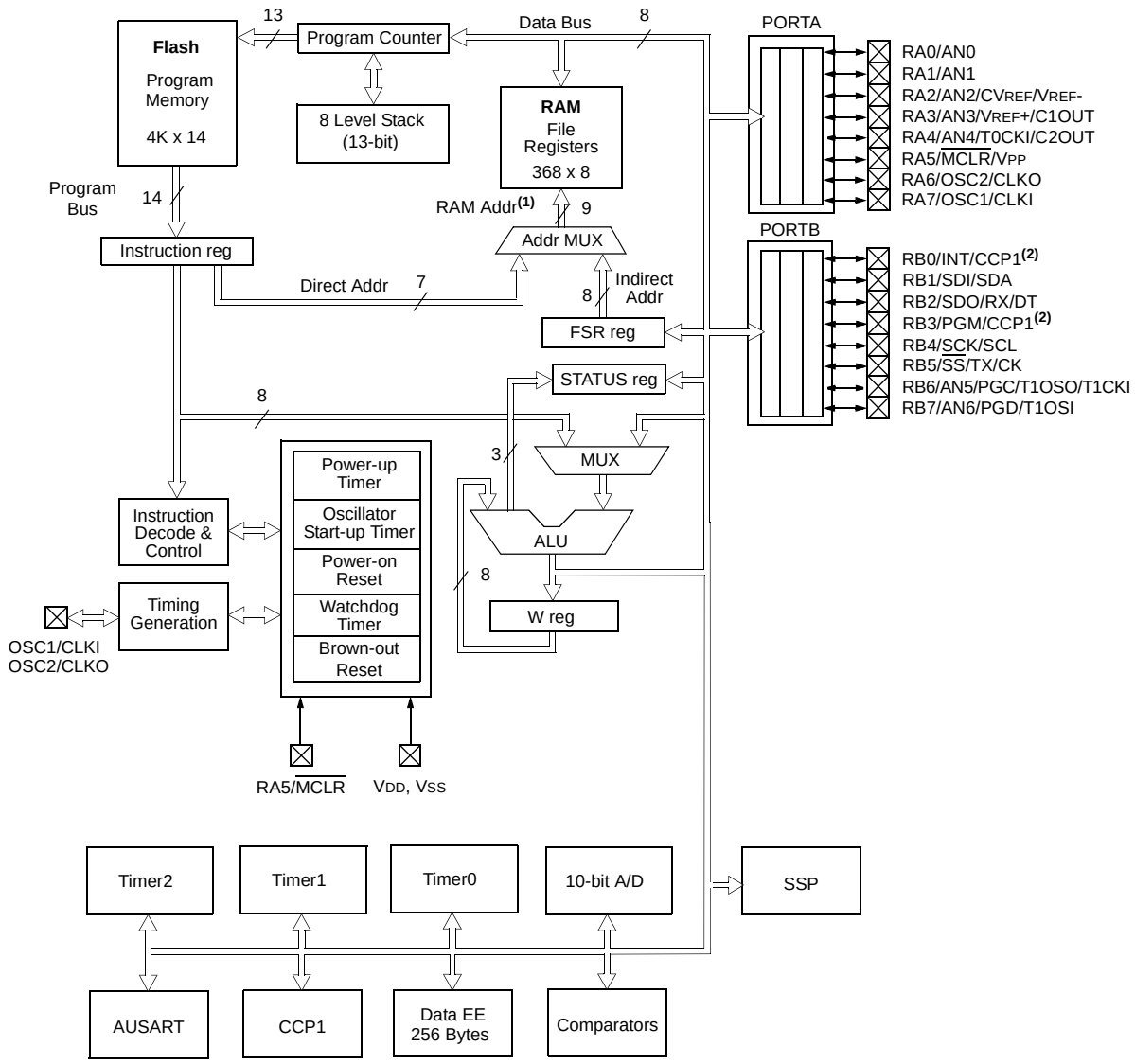
FIGURE 1-1: PIC16F87 DEVICE BLOCK DIAGRAM



Note 1: Higher order bits are from the STATUS register.

2: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

FIGURE 1-2: PIC16F88 DEVICE BLOCK DIAGRAM



Note 1: Higher order bits are from the STATUS register.

2: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

TABLE 1-2: PIC16F87/88 PINOUT DESCRIPTION

Pin Name	PDIP/ SOIC Pin#	SSOP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
						PORTA is a bidirectional I/O port.
RA0/AN0	17	19	23			
RA0				I/O	TTL	Bidirectional I/O pin.
AN0				I	Analog	Analog input channel 0.
RA1/AN1	18	20	24			
RA1				I/O	TTL	Bidirectional I/O pin.
AN1				I	Analog	Analog input channel 1.
RA2/AN2/CVREF/VREF-	1	1	26			
RA2				I/O	TTL	Bidirectional I/O pin.
AN2				I	Analog	Analog input channel 2.
CVREF				O		Comparator VREF output.
VREF-(4)				I	Analog	A/D reference voltage (Low) input.
RA3/AN3/VREF+/C1OUT	2	2	27			
RA3				I/O	TTL	Bidirectional I/O pin.
AN3				I	Analog	Analog input channel 3.
VREF+(4)				I	Analog	A/D reference voltage (High) input.
C1OUT				O		Comparator 1 output.
RA4/AN4/T0CKI/C2OUT	3	3	28			
RA4				I/O	ST	Bidirectional I/O pin.
AN4(4)				I	Analog	Analog input channel 4.
T0CKI				I	ST	Clock input to the TMR0 timer/counter.
C2OUT				O		Comparator 2 output.
RA5/$\overline{\text{MCLR}}$/VPP	4	4	1			
RA5				I	ST	Input pin.
$\overline{\text{MCLR}}$				I	ST	Master Clear (Reset). Input/programming voltage input. This pin is an active-low Reset to the device.
VPP				P	–	Programming voltage input.
RA6/OSC2/CLKO	15	17	20			
RA6				I/O	ST	Bidirectional I/O pin.
OSC2				O	–	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO				O	–	In RC mode, this pin outputs CLKO signal which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA7/OSC1/CLKI	16	18	21			
RA7				I/O	ST	Bidirectional I/O pin.
OSC1				I	ST/CMOS(3)	Oscillator crystal input.
CLKI				I	–	External clock source input.

Legend: I = Input O = Output I/O = Input/Output P = Power
– = Not used TTL = TTL Input ST = Schmitt Trigger Input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.
4: PIC16F88 devices only.
5: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

TABLE 1-2: PIC16F87/88 PINOUT DESCRIPTION (CONTINUED)

Pin Name	PDIP/ SOIC Pin#	SSOP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
						PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT/CCP1(5)	6	7	7			
RB0				I/O	TTL	Bidirectional I/O pin.
INT				I	ST(1)	External interrupt pin.
CCP1				I/O	ST	Capture input, Compare output, PWM output.
RB1/SDI/SDA	7	8	8			
RB1				I/O	TTL	Bidirectional I/O pin.
SDI				I	ST	SPI™ data in.
SDA				I/O	ST	I^2C™ data.
RB2/SDO/RX/DT	8	9	9			
RB2				I/O	TTL	Bidirectional I/O pin.
SDO				O	ST	SPI data out.
RX				I		AUSART asynchronous receive.
DT				I/O		AUSART synchronous detect.
RB3/PGM/CCP1(5)	9	10	10			
RB3				I/O	TTL	Bidirectional I/O pin.
PGM				I/O	ST	Low-Voltage ICSP™ Programming enable pin.
CCP1				I	ST	Capture input, Compare output, PWM output.
RB4/SCK/SCL	10	11	12			
RB4				I/O	TTL	Bidirectional I/O pin. Interrupt-on-change pin.
SCK				I/O	ST	Synchronous serial clock input/output for SPI.
SCL				I	ST	Synchronous serial clock Input for I^2C.
RB5/$\overline{SS}$/TX/CK	11	12	13			
RB5				I/O	TTL	Bidirectional I/O pin. Interrupt-on-change pin.
$\overline{SS}$				I	TTL	Slave select for SPI in Slave mode.
TX				O		AUSART asynchronous transmit.
CK				I/O		AUSART synchronous clock.
RB6/AN5/PGC/T1OSO/ T1CKI	12	13	15			
RB6				I/O	TTL	Bidirectional I/O pin. Interrupt-on-change pin.
AN5(4)				I		Analog input channel 5.
PGC				I/O	ST(2)	In-Circuit Debugger and programming clock pin.
T1OSO				O	ST	Timer1 oscillator output.
T1CKI				I	ST	Timer1 external clock input.
RB7/AN6/PGD/T1OSI	13	14	16			
RB7				I/O	TTL	Bidirectional I/O pin. Interrupt-on-change pin.
AN6(4)				I		Analog input channel 6.
PGD				I	ST(2)	In-Circuit Debugger and ICSP programming data pin.
T1OSI				I	ST	Timer1 oscillator input.
VSS	5	5, 6	3, 5	P	–	Ground reference for logic and I/O pins.
VDD	14	15, 16	17, 19	P	–	Positive supply for logic and I/O pins.

Legend: I = Input O = Output I/O = Input/Output P = Power
– = Not used TTL = TTL Input ST = Schmitt Trigger Input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.
4: PIC16F88 devices only.
5: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

NOTES:

2.0 MEMORY ORGANIZATION

There are two memory blocks in the PIC16F87/88 devices. These are the program memory and the data memory. Each block has its own bus, so access to each block can occur during the same oscillator cycle.

The data memory can be further broken down into the general purpose RAM and the Special Function Registers (SFRs). The operation of the SFRs that control the “core” are described here. The SFRs used to control the peripheral modules are described in the section discussing each individual peripheral module.

The data memory area also contains the data EEPROM memory. This memory is not directly mapped into the data memory but is indirectly mapped. That is, an indirect address pointer specifies the address of the data EEPROM memory to read/write. The PIC16F87/88 device’s 256 bytes of data EEPROM memory have the address range of 00h-FFh. More details on the EEPROM memory can be found in **Section 3.0 “Data EEPROM and Flash Program Memory”**.

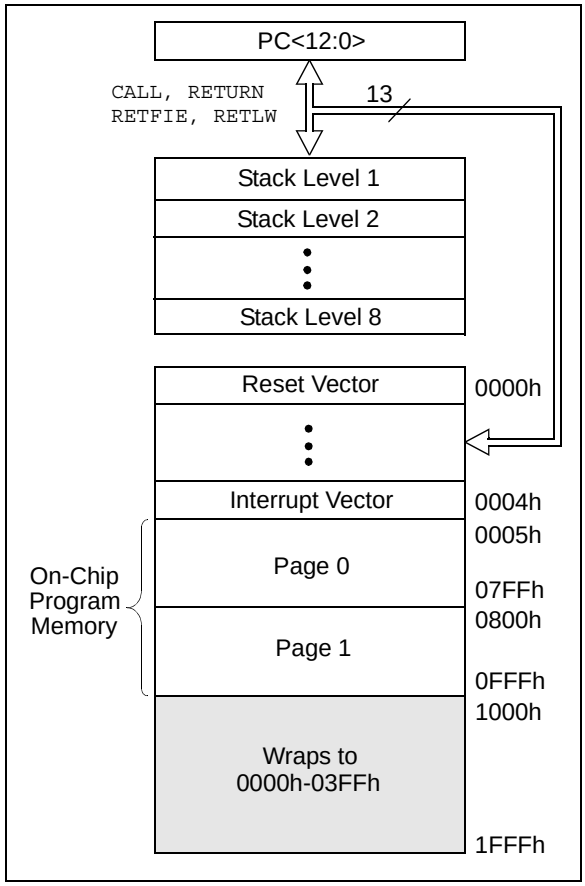
Additional information on device memory may be found in the “*PICmicro® Mid-Range MCU Family Reference Manual*” (DS33023).

2.1 Program Memory Organization

The PIC16F87/88 devices have a 13-bit program counter capable of addressing an 8K x 14 program memory space. For the PIC16F87/88, the first 4K x 14 (0000h-0FFFh) is physically implemented (see Figure 2-1). Accessing a location above the physically implemented address will cause a wraparound. For example, the same instruction will be accessed at locations 020h, 420h, 820h, C20h, 1020h, 1420h, 1820h and 1C20h.

The Reset vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 2-1: PROGRAM MEMORY MAP AND STACK: PIC16F87/88



2.2 Data Memory Organization

The data memory is partitioned into multiple banks that contain the General Purpose Registers and the Special Function Registers. Bits RP1 (STATUS<6>) and RP0 (STATUS<5>) are the bank select bits.

RP1:RP0	Bank
00	0
01	1
10	2
11	3

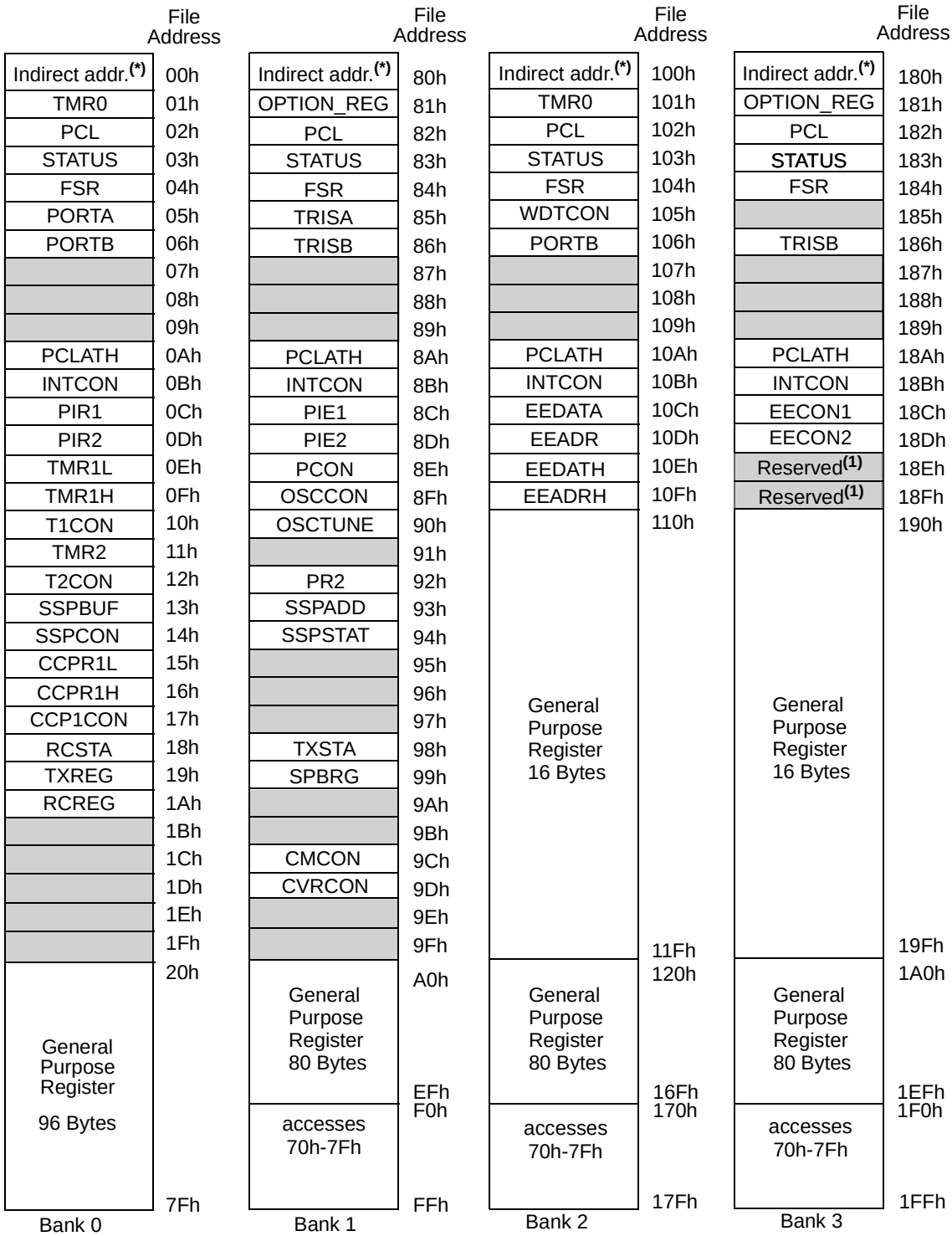
Each bank extends up to 7Fh (128 bytes). The lower locations of each bank are reserved for the Special Function Registers. Above the Special Function Registers are General Purpose Registers, implemented as static RAM. All implemented banks contain SFRs. Some “high use” SFRs from one bank may be mirrored in another bank for code reduction and quicker access (e.g., the STATUS register is in Banks 0-3).

Note: EEPROM data memory description can be found in **Section 3.0 “Data EEPROM and Flash Program Memory”** of this data sheet.

2.2.1 GENERAL PURPOSE REGISTER FILE

The register file can be accessed either directly, or indirectly, through the File Select Register (FSR).

FIGURE 2-2: PIC16F87 REGISTER FILE MAP



Unimplemented data memory locations, read as '0'.

* Not a physical register.

Note 1: This register is reserved, maintain this register clear.

FIGURE 2-3: PIC16F88 REGISTER FILE MAP

Bank 0	File Address	Bank 1	File Address	Bank 2	File Address	Bank 3	File Address
Indirect addr.(*)	00h	Indirect addr.(*)	80h	Indirect addr.(*)	100h	Indirect addr.(*)	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h	WDTCON	105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
	07h		87h		107h		187h
	08h		88h		108h		188h
	09h		89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved(1)	18Eh
TMR1H	0Fh	OSCCON	8Fh	EEADRH	10Fh	Reserved(1)	18Fh
T1CON	10h	OSCTUNE	90h	General Purpose Register 16 Bytes	110h	General Purpose Register 16 Bytes	190h
TMR2	11h		91h				
T2CON	12h	PR2	92h				
SSPBUF	13h	SSPADD	93h				
SSPCON	14h	SSPSTAT	94h				
CCPR1L	15h		95h				
CCPR1H	16h		96h				
CCP1CON	17h		97h				
RCSTA	18h	TXSTA	98h				
TXREG	19h	SPBRG	99h				
RCREG	1Ah		9Ah				
	1Bh	ANSEL	9Bh				
	1Ch	CMCON	9Ch				
	1Dh	CVRCON	9Dh				
ADRESH	1Eh	ADRESL	9Eh				
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
General Purpose Register 96 Bytes	20h	General Purpose Register 80 Bytes	A0h	General Purpose Register 80 Bytes	120h	General Purpose Register 80 Bytes	1A0h
			EFh		16Fh		1EFh
		accesses 70h-7Fh	F0h	accesses 70h-7Fh	170h	accesses 70h-7Fh	1F0h
	7Fh		FFh		17Fh		1FFh

Unimplemented data memory locations, read as '0'.

* Not a physical register.

Note 1: This register is reserved, maintain this register clear.

2.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. A list of these registers is given in Table 2-1.

The Special Function Registers can be classified into two sets: core (CPU) and peripheral. Those registers associated with the core functions are described in detail in this section. Those related to the operation of the peripheral features are described in detail in the peripheral feature section.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page
Bank 0											
00h(2)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	26, 135
01h	TMR0	Timer0 Module Register								xxxx xxxx	69
02h(2)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	
03h(2)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	17
04h(2)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	135
05h	PORTA	PORTA Data Latch when written; PORTA pins when read (PIC16F87) PORTA Data Latch when written; PORTA pins when read (PIC16F88)								xxxx 0000 xxx0 0000	52
06h	PORTB	PORTB Data Latch when written; PORTB pins when read (PIC16F87) PORTB Data Latch when written; PORTB pins when read (PIC16F88)								xxxx xxxx 00xx xxxx	58
07h	—	Unimplemented								—	—
08h	—	Unimplemented								—	—
09h	—	Unimplemented								—	—
0Ah(1,2)	PCLATH	—	—	—	Write Buffer for the Upper 5 bits of the Program Counter					---0 0000	135
0Bh(2)	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	19, 69, 77
0Ch	PIR1	—	ADIF(4)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	21, 77
0Dh	PIR2	OSFIF	CMIF	—	EEIF	—	—	—	—	00-0 ----	23, 34
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	77, 83
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	77, 83
10h	T1CON	—	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	-000 0000	72, 83
11h	TMR2	Timer2 Module Register								0000 0000	80, 85
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	80, 85
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	90, 95
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	89, 95
15h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	83, 85
16h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	83, 85
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	81, 83
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	98, 99
19h	TXREG	AUSART Transmit Data Register								0000 0000	103
1Ah	RCREG	AUSART Receive Data Register								0000 0000	105
1Bh	—	Unimplemented								—	—
1Ch	—	Unimplemented								—	—
1Dh	—	Unimplemented								—	—
1Eh	ADRESH(4)	A/D Result Register High Byte								xxxx xxxx	120
1Fh	ADCON0(4)	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	—	ADON	0000 00-0	114, 120

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for PC<12:8>, whose contents are transferred to the upper byte of the program counter.

2: These registers can be addressed from any bank.

3: RA5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

4: PIC16F88 device only.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page
Bank 1											
80h(2)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	26, 135
81h	OPTION_REG	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	18, 69
82h(2)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	135
83h(2)	STATUS	IRP	RP1	RP0	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	0001 1xxx	17
84h(2)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	135
85h	TRISA	TRISA7	TRISA6	TRISA5(3)	PORTA Data Direction Register (TRISA<4:0>)					1111 1111	52, 126
86h	TRISB	PORTB Data Direction Register								1111 1111	58, 85
87h	—	Unimplemented								—	—
88h	—	Unimplemented								—	—
89h	—	Unimplemented								—	—
8Ah(1,2)	PCLATH	—	—	—	Write Buffer for the Upper 5 bits of the Program Counter					---0 0000	135
8Bh(2)	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	19, 69, 77
8Ch	PIE1	—	ADIE(4)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	20, 80
8Dh	PIE2	OSFIE	CMIE	—	EEIE	—	—	—	—	00-0 ----	22, 34
8Eh	PCON	—	—	—	—	—	—	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	---- --0q	24
8Fh	OSCCON	—	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	-000 0000	40
90h	OSCTUNE	—	—	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	--00 0000	38
91h	—	Unimplemented								—	—
92h	PR2	Timer2 Period Register								1111 1111	80, 85
93h	SSPADD	Synchronous Serial Port (I^2C™ mode) Address Register								0000 0000	95
94h	SSPSTAT	SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF	0000 0000	88, 95
95h	—	Unimplemented								—	—
96h	—	Unimplemented								—	—
97h	—	Unimplemented								—	—
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	97, 99
99h	SPBRG	Baud Rate Generator Register								0000 0000	99, 103
9Ah	—	Unimplemented								—	—
9Bh	ANSEL(4)	—	ANS6	ANS5	ANS4	ANS3	ANS2	ANS1	ANS0	-111 1111	120
9Ch	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	121, 126, 128
9Dh	CVRCON	CVREN	CVROE	CVRR	—	CVR3	CVR2	CVR1	CVR0	000- 0000	126, 128
9Eh	ADRESL(4)	A/D Result Register Low Byte								xxxx xxxx	120
9Fh	ADCON1(4)	ADFM	ADCS2	VCFG1	VCFG0	—	—	—	—	0000 ----	52, 115, 120

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for PC<12:8>, whose contents are transferred to the upper byte of the program counter.

2: These registers can be addressed from any bank.

3: RA5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

4: PIC16F88 device only.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page
Bank 2											
100h(2)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	26, 135
101h	TMR0	Timer0 Module Register								xxxx xxxx	69
102h(2)	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	135
103h(2)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	17
104h(2)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	135
105h	WDTCON	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	SWDTEN	---0 1000	142
106h	PORTB	PORTB Data Latch when written; PORTB pins when read (PIC16F87) PORTB Data Latch when written; PORTB pins when read (PIC16F88)								xxxx xxxx 00xx xxxx	58
107h	—	Unimplemented								—	—
108h	—	Unimplemented								—	—
109h	—	Unimplemented								—	—
10Ah(1,2)	PCLATH	—	—	—	Write Buffer for the Upper 5 bits of the Program Counter					---0 0000	135
10Bh(2)	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	19, 69, 77
10Ch	EEDATA	EEPROM/Flash Data Register Low Byte								xxxx xxxx	34
10Dh	EEADR	EEPROM/Flash Address Register Low Byte								xxxx xxxx	34
10Eh	EEDATH	—	—	EEPROM/Flash Data Register High Byte						--xx xxxx	34
10Fh	EEADRH	—	—	—	—	EEPROM/Flash Address Register High Byte				---- xxxx	34
Bank 3											
180h(2)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	135
181h	OPTION_REG	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	18, 69
182h(2)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	135
183h(2)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	17
184h(2)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	135
185h	—	Unimplemented								—	—
186h	TRISB	PORTB Data Direction Register								1111 1111	58, 83
187h	—	Unimplemented								—	—
188h	—	Unimplemented								—	—
189h	—	Unimplemented								—	—
18Ah(1,2)	PCLATH	—	—	—	Write Buffer for the Upper 5 bits of the Program Counter					---0 0000	135
18Bh(2)	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	19, 69, 77
18Ch	EECON1	EEPGD	—	—	FREE	WRERR	WREN	WR	RD	x--x x000	28, 34
18Dh	EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	34
18Eh	—	Reserved, maintain clear								0000 0000	—
18Fh	—	Reserved, maintain clear								0000 0000	—

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for PC<12:8>, whose contents are transferred to the upper byte of the program counter.

2: These registers can be addressed from any bank.

3: RA5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

4: PIC16F88 device only.

2.2.2.1 STATUS Register

The STATUS register, shown in Register 2-1, contains the arithmetic status of the ALU, the Reset status and the bank select bits for data memory.

The STATUS register can be the destination for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper three bits and set the Z bit. This leaves the STATUS register as '`000u u1uu`' (where `u` = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register because these instructions do not affect the Z, C or DC bits from the STATUS register. For other instructions not affecting any Status bits, see **Section 16.0 "Instruction Set Summary"**.

> **Note:** The C and DC bits operate as a $\overline{borrow}$ and $\overline{digit\ borrow}$ bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

REGISTER 2-1: STATUS: ARITHMETIC STATUS REGISTER (ADDRESS 03h, 83h, 103h, 183h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

bit 7 **IRP:** Register Bank Select bit (used for indirect addressing)
`1` = Bank 2, 3 (100h-1FFh)
`0` = Bank 0, 1 (00h-FFh)

bit 6-5 **RP<1:0>:** Register Bank Select bits (used for direct addressing)
`11` = Bank 3 (180h-1FFh)
`10` = Bank 2 (100h-17Fh)
`01` = Bank 1 (80h-FFh)
`00` = Bank 0 (00h-7Fh)
Each bank is 128 bytes.

bit 4 **$\overline{TO}$:** Time-out bit
`1` = After power-up, `CLRWDT` instruction or `SLEEP` instruction
`0` = A WDT time-out occurred

bit 3 **$\overline{PD}$:** Power-Down bit
`1` = After power-up or by the `CLRWDT` instruction
`0` = By execution of the `SLEEP` instruction

bit 2 **Z:** Zero bit
`1` = The result of an arithmetic or logic operation is zero
`0` = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit carry/$\overline{borrow}$ bit (`ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions)(1)
`1` = A carry-out from the 4th low-order bit of the result occurred
`0` = No carry-out from the 4th low-order bit of the result

bit 0 **C:** Carry/$\overline{borrow}$ bit (`ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions)(1,2)
`1` = A carry-out from the Most Significant bit of the result occurred
`0` = No carry-out from the Most Significant bit of the result occurred

Note 1: For borrow, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand.

2: For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high or low-order bit of the source register.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.2.2.2 OPTION_REG Register

The OPTION_REG register is a readable and writable register that contains various control bits to configure the TMR0 prescaler/WDT postscaler (single assignable register known also as the prescaler), the external INT interrupt, TMR0 and the weak pull-ups on PORTB.

> **Note:** To achieve a 1:1 prescaler assignment for the TMR0 register, assign the prescaler to the Watchdog Timer. Although the prescaler can be assigned to either the WDT or Timer0, but not both, a new divide counter is implemented in the WDT circuit to give multiple WDT time-out selections. This allows TMR0 and WDT to each have their own scaler. Refer to **Section 15.12 "Watchdog Timer (WDT)"** for further details.

REGISTER 2-2: OPTION_REG: OPTION CONTROL REGISTER (ADDRESS 81h, 181h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$**: PORTB Pull-up Enable bit
1 = PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG:** Interrupt Edge Select bit
1 = Interrupt on rising edge of RB0/INT pin
0 = Interrupt on falling edge of RB0/INT pin

bit 5 **T0CS:** TMR0 Clock Source Select bit
1 = Transition on RA4/T0CKI/C2OUT pin
0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE:** TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on RA4/T0CKI/C2OUT pin
0 = Increment on low-to-high transition on RA4/T0CKI/C2OUT pin

bit 3 **PSA:** Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS<2:0>:** Prescaler Rate Select bits

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.2.2.3 INTCON Register

The INTCON register is a readable and writable register that contains various enable and flag bits for the TMR0 register overflow, RB Port change and External RB0/INT pin interrupts.

> **Note:** Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 2-3: INTCON: INTERRUPT CONTROL REGISTER (ADDRESS 0Bh, 8Bh, 10Bh, 18Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
bit 7							bit 0

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **PEIE:** Peripheral Interrupt Enable bit
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INT0IE:** RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit
A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition and allow flag bit RBIF to be cleared.
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

2.2.2.4 PIE1 Register

This register contains the individual enable bits for the peripheral interrupts.

Note:	Bit PEIE (INTCON<6>) must be set to enable any peripheral interrupt.

REGISTER 2-4: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1 (ADDRESS 8Ch)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6 **ADIE:** A/D Converter Interrupt Enable bit(1)
1 = Enabled
0 = Disabled

Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

bit 5 **RCIE:** AUSART Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **TXIE:** AUSART Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **SSPIE:** Synchronous Serial Port (SSP) Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **CCP1IE:** CCP1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **TMR1IE:** TMR1 Overflow Interrupt Enable bit
1 = Enabled
0 = Disabled

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.2.2.5 PIR1 Register

This register contains the individual flag bits for the peripheral interrupts.

> **Note:** Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 2-5: PIR1: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 1 (ADDRESS 0Ch)

U-0	R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0
—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6 **ADIF:** A/D Converter Interrupt Flag bit(1)
1 = The A/D conversion completed (must be cleared in software)
0 = The A/D conversion is not complete

Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

bit 5 **RCIF:** AUSART Receive Interrupt Flag bit
1 = The AUSART receive buffer is full (cleared by reading RCREG)
0 = The AUSART receive buffer is not full

bit 4 **TXIF:** AUSART Transmit Interrupt Flag bit
1 = The AUSART transmit buffer is empty (cleared by writing to TXREG)
0 = The AUSART transmit buffer is full

bit 3 **SSPIF:** Synchronous Serial Port (SSP) Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 2 **CCP1IF:** CCP1 Interrupt Flag bit
Capture mode:
1 = A TMR1 register capture occurred (must be cleared in software)
0 = No TMR1 register capture occurred
Compare mode:
1 = A TMR1 register compare match occurred (must be cleared in software)
0 = No TMR1 register compare match occurred
PWM mode:
Unused in this mode.

bit 1 **TMR2IF:** TMR2 to PR2 Interrupt Flag bit
1 = A TMR2 to PR2 match occurred (must be cleared in software)
0 = No TMR2 to PR2 match occurred

bit 0 **TMR1IF:** TMR1 Overflow Interrupt Flag bit
1 = The TMR1 register overflowed (must be cleared in software)
0 = The TMR1 register did not overflow

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.2.2.6 PIE2 Register

The PIE2 register contains the individual enable bit for the EEPROM write operation interrupt.

REGISTER 2-6: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2 (ADDRESS 8Dh)

R/W-0	R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0
OSFIE	CMIE	—	EEIE	—	—	—	—
bit 7							bit 0

bit 7 **OSFIE:** Oscillator Fail Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **CMIE:** Comparator Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **Unimplemented:** Read as '0'

bit 4 **EEIE:** EEPROM Write Operation Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3-0 **Unimplemented:** Read as '0'

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.2.2.7 PIR2 Register

The PIR2 register contains the flag bit for the EEPROM write operation interrupt.

Note:	Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 2-7: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2 (ADDRESS 0Dh)

R/W-0	R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0
OSFIF	CMIF	—	EEIF	—	—	—	—
bit 7							bit 0

bit 7 **OSFIF:** Oscillator Fail Interrupt Flag bit
1 = System oscillator failed, clock input has changed to INTRC (must be cleared in software)
0 = System clock operating

bit 6 **CMIF:** Comparator Interrupt Flag bit
1 = Comparator input has changed (must be cleared in software)
0 = Comparator input has not changed

bit 5 **Unimplemented:** Read as '0'

bit 4 **EEIF:** EEPROM Write Operation Interrupt Flag bit
1 = The write operation completed (must be cleared in software)
0 = The write operation is not complete or has not been started

bit 3-0 **Unimplemented:** Read as '0'

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

2.2.2.8 PCON Register

Note:	Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit, or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

The Power Control (PCON) register contains a flag bit to allow differentiation between a Power-on Reset (POR), a Brown-out Reset, an external $\overline{\text{MCLR}}$ Reset and WDT Reset.

Note:	$\overline{\text{BOR}}$ is unknown on Power-on Reset. It must then be set by the user and checked on subsequent Resets to see if $\overline{\text{BOR}}$ is clear, indicating a brown-out has occurred. The $\overline{\text{BOR}}$ status bit is a ‘don’t care’ and is not necessarily predictable if the brown-out circuit is disabled (by clearing the BOREN bit in the Configuration Word register).

REGISTER 2-8: PCON: POWER CONTROL REGISTER (ADDRESS 8Eh)

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-x
—	—	—	—	—	—	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
bit 7							bit 0

bit 7-2 **Unimplemented:** Read as ‘0’

bit 1 **$\overline{\text{POR}}$:** Power-on Reset Status bit
1 = No Power-on Reset occurred
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

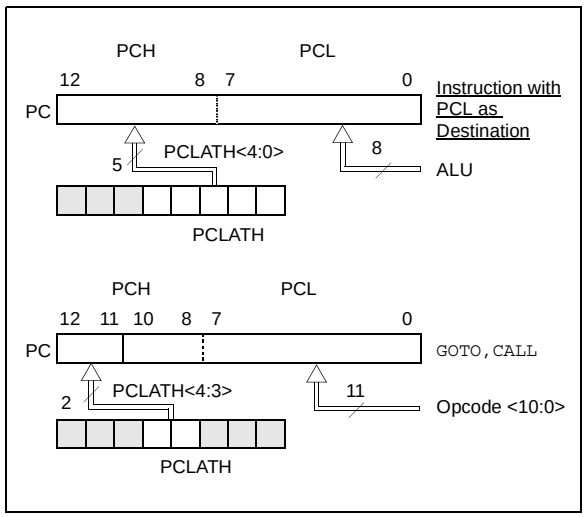
bit 0 **$\overline{\text{BOR}}$:** Brown-out Reset Status bit
1 = No Brown-out Reset occurred
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

2.3 PCL and PCLATH

The Program Counter (PC) is 13 bits wide. The low byte comes from the PCL register which is a readable and writable register. The upper bits (PC<12:8>) are not readable but are indirectly writable through the PCLATH register. On any Reset, the upper bits of the PC will be cleared. Figure 2-4 shows the two situations for the loading of the PC. The upper example in the figure shows how the PC is loaded on a write to PCL (PCLATH<4:0> → PCH). The lower example in the figure shows how the PC is loaded during a `CALL` or `GOTO` instruction (PCLATH<4:3> → PCH).

FIGURE 2-4: LOADING OF PC IN DIFFERENT SITUATIONS



2.3.1 COMPUTED GOTO

A computed `GOTO` is accomplished by adding an offset to the program counter (`ADDWF PCL`). When doing a table read using a computed `GOTO` method, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block). Refer to the application note, *AN556, "Implementing a Table Read"*.

2.3.2 STACK

The PIC16F87/88 family has an 8-level deep x 13-bit wide hardware stack. The stack space is not part of either program or data space and the Stack Pointer is not readable or writable. The PC is PUSHed onto the stack when a `CALL` instruction is executed or an interrupt causes a branch. The stack is POPed in the event of a `RETURN`, `RETLW` or a `RETFIE` instruction execution. PCLATH is not affected by a PUSH or POP operation.

The stack operates as a circular buffer. This means that after the stack has been PUSHed eight times, the ninth push overwrites the value that was stored from the first push. The tenth push overwrites the second push (and so on).

> **Note 1:** There are no status bits to indicate stack overflow or stack underflow conditions.
>
> **2:** There are no instructions/mnemonics called PUSH or POP. These are actions that occur from the execution of the `CALL`, `RETURN`, `RETLW` and `RETFIE` instructions, or the vectoring to an interrupt address.

2.4 Program Memory Paging

All PIC16F87/88 devices are capable of addressing a continuous 8K word block of program memory. The `CALL` and `GOTO` instructions provide only 11 bits of address to allow branching within any 2K program memory page. When doing a `CALL` or `GOTO` instruction, the upper 2 bits of the address are provided by PCLATH<4:3>. When doing a `CALL` or `GOTO` instruction, the user must ensure that the page select bits are programmed so that the desired program memory page is addressed. If a return from a `CALL` instruction (or interrupt) is executed, the entire 13-bit PC is popped off the stack. Therefore, manipulation of the PCLATH<4:3> bits is not required for the `RETURN` instructions (which POPs the address from the stack).

> **Note:** The contents of the PCLATH register are unchanged after a `RETURN` or `RETFIE` instruction is executed. The user must rewrite the contents of the PCLATH register for any subsequent subroutine calls or `GOTO` instructions.

Example 2-1 shows the calling of a subroutine in page 1 of the program memory. This example assumes that PCLATH is saved and restored by the Interrupt Service Routine (if interrupts are used).

EXAMPLE 2-1: CALL OF A SUBROUTINE IN PAGE 1 FROM PAGE 0

```
        ORG 0x500
        BCF PCLATH, 4
        BSF PCLATH, 3 ;Select page 1
                      ;(800h-FFFh)
        CALL SUB1_P1  ;Call subroutine in
        :             ;page 1 (800h-FFFh)
        :
        ORG 0x900     ;page 1 (800h-FFFh)
SUB1_P1
        :             ;called subroutine
                      ;page 1 (800h-FFFh)
        :
        RETURN        ;return to
                      ;Call subroutine
                      ;in page 0
                      ;(000h-7FFh)
```

2.5 Indirect Addressing, INDF and FSR Registers

The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing.

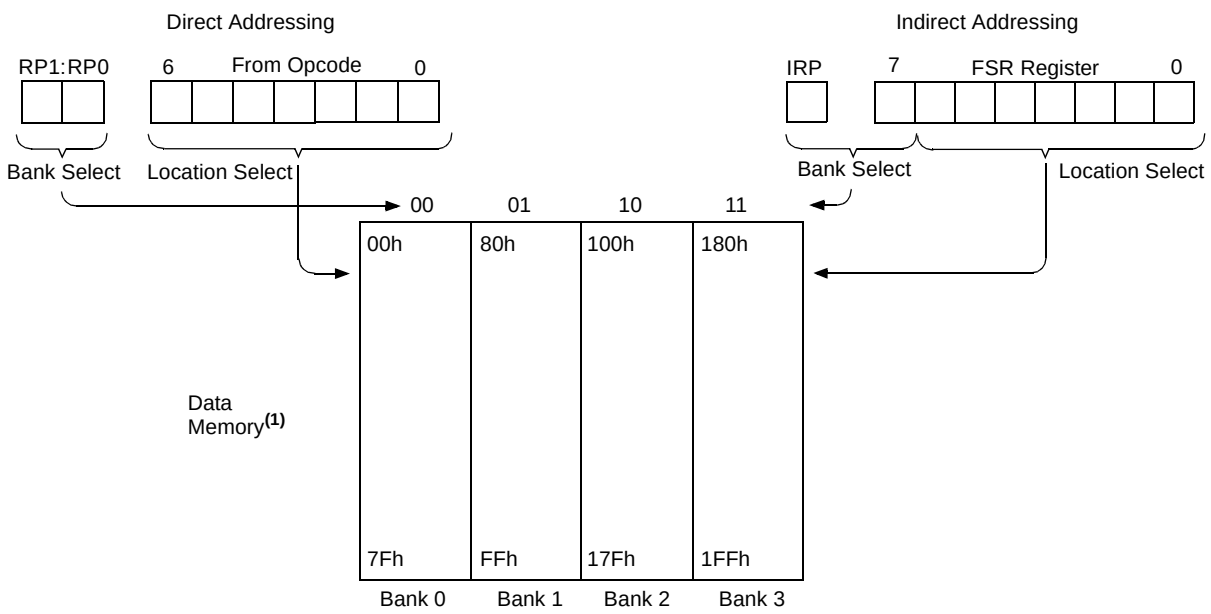
Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses the register pointed to by the File Select Register, FSR. Reading the INDF register itself, indirectly (FSR = 0) will read 00h. Writing to the INDF register indirectly results in a no operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 2-5.

A simple program to clear RAM locations 20h-2Fh using indirect addressing is shown in Example 2-2.

EXAMPLE 2-2: INDIRECT ADDRESSING

```
        MOVLW  0x20    ;initialize pointer
        MOVWF  FSR     ;to RAM
NEXT    CLRF   INDF    ;clear INDF register
        INCF   FSR, F  ;inc pointer
        BTFSS  FSR, 4  ;all done?
        GOTO   NEXT    ;no clear next
CONTINUE
        :              ;yes continue
```

FIGURE 2-5: DIRECT/INDIRECT ADDRESSING



Note 1: For register file map detail, see Figure 2-2 or Figure 2-3.

3.0 DATA EEPROM AND FLASH PROGRAM MEMORY

The data EEPROM and Flash program memory are readable and writable during normal operation (over the full VDD range). This memory is not directly mapped in the register file space. Instead, it is indirectly addressed through the Special Function Registers. There are six SFRs used to read and write this memory:

- EECON1
- EECON2
- EEDATA
- EEDATH
- EEADR
- EEADRH

This section focuses on reading and writing data EEPROM and Flash program memory during normal operation. Refer to the appropriate device programming specification document for serial programming information.

When interfacing the data memory block, EEDATA holds the 8-bit data for read/write and EEADR holds the address of the EEPROM location being accessed. The PIC16F87/88 devices have 256 bytes of data EEPROM with an address range from 00h to 0FFh. When writing to unimplemented locations, the charge pump will be turned off.

When interfacing the program memory block, the EEDATA and EEDATH registers form a two-byte word that holds the 14-bit data for read/write and the EEADR and EEADRH registers form a two-byte word that holds the 13-bit address of the EEPROM location being accessed. The PIC16F87/88 devices have 4K words of program Flash with an address range from 0000h to 0FFFh. Addresses above the range of the respective device will wraparound to the beginning of program memory.

The EEPROM data memory allows single byte read and write. The Flash program memory allows single-word reads and four-word block writes. Program memory writes must first start with a 32-word block erase, then write in 4-word blocks. A byte write in data EEPROM memory automatically erases the location and writes the new data (erase before write).

The write time is controlled by an on-chip timer. The write/erase voltages are generated by an on-chip charge pump, rated to operate over the voltage range of the device for byte or word operations.

When the device is code-protected, the CPU may continue to read and write the data EEPROM memory. Depending on the settings of the write-protect bits, the device may or may not be able to write certain blocks of the program memory; however, reads of the program memory are allowed. When code-protected, the device programmer can no longer access data or program memory; this does NOT inhibit internal reads or writes.

3.1 EEADR and EEADRH

The EEADRH:EEADR register pair can address up to a maximum of 256 bytes of data EEPROM, or up to a maximum of 8K words of program EEPROM. When selecting a data address value, only the LSB of the address is written to the EEADR register. When selecting a program address value, the MSB of the address is written to the EEADRH register and the LSB is written to the EEADR register.

If the device contains less memory than the full address reach of the address register pair, the Most Significant bits of the registers are not implemented. For example, if the device has 128 bytes of data EEPROM, the Most Significant bit of EEADR is not implemented on access to data EEPROM.

3.2 EECON1 and EECON2 Registers

EECON1 is the control register for memory accesses.

Control bit EEPGD determines if the access will be a program or data memory access. When clear, as it is when reset, any subsequent operations will operate on the data memory. When set, any subsequent operations will operate on the program memory.

Control bits, RD and WR, initiate read and write, respectively. These bits cannot be cleared, only set in software. They are cleared in hardware at completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental, premature termination of a write operation.

The WREN bit, when set, will allow a write or erase operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write (or erase) operation is interrupted by a $\overline{\text{MCLR}}$, or a WDT Time-out Reset during normal operation. In these situations, following Reset, the user can check the WRERR bit and rewrite the location. The data and address will be unchanged in the EEDATA and EEADR registers.

Interrupt flag bit, EEIF in the PIR2 register, is set when the write is complete. It must be cleared in software.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the EEPROM write sequence.

REGISTER 3-1: EECON1: EEPROM ACCESS CONTROL REGISTER 1 (ADDRESS 18Ch)

R/W-x	U-0	U-0	R/W-x	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	—	—	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

bit 7 **EEPGD**: Program/Data EEPROM Select bit
1 = Accesses program memory
0 = Accesses data memory

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **FREE:** EEPROM Forced Row Erase bit
1 = Erase the program memory row addressed by EEADRH:EEADR on the next WR command
0 = Perform write only

bit 3 **WRERR:** EEPROM Error Flag bit
1 = A write operation is prematurely terminated (any $\overline{\text{MCLR}}$ or any WDT Reset during normal operation)
0 = The write operation completed

bit 2 **WREN:** EEPROM Write Enable bit
1 = Allows write cycles
0 = Inhibits write to the EEPROM

bit 1 **WR:** Write Control bit
1 = Initiates a write cycle. The bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.
0 = Write cycle to the EEPROM is complete

bit 0 **RD:** Read Control bit
1 = Initiates an EEPROM read, RD is cleared in hardware. The RD bit can only be set (not cleared) in software.
0 = Does not initiate an EEPROM read

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	S = Set only
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

3.3 Reading Data EEPROM Memory

To read a data memory location, the user must write the address to the EEADR register, clear the EEPGD control bit (EECON1<7>) and then set control bit RD (EECON1<0>). The data is available in the very next cycle in the EEDATA register; therefore, it can be read in the next instruction (see Example 3-1). EEDATA will hold this value until another read or until it is written to by the user (during a write operation).

The steps to reading the EEPROM data memory are:

1. Write the address to EEADR. Make sure that the address is not larger than the memory size of the device.
2. Clear the EEPGD bit to point to EEPROM data memory.
3. Set the RD bit to start the read operation.
4. Read the data from the EEDATA register.

EXAMPLE 3-1: DATA EEPROM READ

```
BANKSEL EEADR         ; Select Bank of EEADR
MOVF    ADDR, W       ;
MOVWF   EEADR         ; Data Memory Address
                      ; to read
BANKSEL EECON1        ; Select Bank of EECON1
BCF     EECON1, EEPGD; Point to Data memory
BSF     EECON1, RD    ; EE Read
BANKSEL EEDATA        ; Select Bank of EEDATA
MOVF    EEDATA, W     ; W = EEDATA
```

3.4 Writing to Data EEPROM Memory

To write an EEPROM data location, the user must first write the address to the EEADR register and the data to the EEDATA register. Then, the user must follow a specific write sequence to initiate the write for each byte.

The write will not initiate if the write sequence is not exactly followed (write 55h to EECON2, write AAh to EECON2, then set WR bit) for each byte. We strongly recommend that interrupts be disabled during this code segment (see Example 3-2).

Additionally, the WREN bit in EECON1 must be set to enable write. This mechanism prevents accidental writes to data EEPROM due to errant (unexpected) code execution (i.e., lost programs). The user should keep the WREN bit clear at all times except when updating EEPROM. The WREN bit is not cleared by hardware

After a write sequence has been initiated, clearing the WREN bit will not affect this write cycle. The WR bit will be inhibited from being set unless the WREN bit is set. At the completion of the write cycle, the WR bit is cleared in hardware and the EE Write Complete Interrupt Flag bit (EEIF) is set. The user can either enable this interrupt or poll this bit. EEIF must be cleared by software.

The steps to write to EEPROM data memory are:

1. If step 10 is not implemented, check the WR bit to see if a write is in progress.
2. Write the address to EEADR. Make sure that the address is not larger than the memory size of the device.
3. Write the 8-bit data value to be programmed in the EEDATA register.
4. Clear the EEPGD bit to point to EEPROM data memory.
5. Set the WREN bit to enable program operations.
6. Disable interrupts (if enabled).
7. Execute the special five instruction sequence:

   Write 55h to EECON2 in two steps (first to W, then to EECON2).

   Write AAh to EECON2 in two steps (first to W, then to EECON2).

   Set the WR bit.
8. Enable interrupts (if using interrupts).
9. Clear the WREN bit to disable program operations.
10. At the completion of the write cycle, the WR bit is cleared and the EEIF interrupt flag bit is set (EEIF must be cleared by firmware). If step 1 is not implemented, then firmware should check for EEIF to be set, or WR to clear, to indicate the end of the program cycle.

EXAMPLE 3-2: DATA EEPROM WRITE

```
        BANKSEL EECON1         ; Select Bank of
                               ; EECON1
        BTFSC   EECON1, WR     ; Wait for write
        GOTO    $-1            ; to complete
        BANKSEL EEADR          ; Select Bank of
                               ; EEADR
        MOVF    ADDR, W        ;
        MOVWF   EEADR          ; Data Memory
                               ; Address to write
        MOVF    VALUE, W       ;
        MOVWF   EEDATA         ; Data Memory Value
                               ; to write
        BANKSEL EECON1         ; Select Bank of
                               ; EECON1
        BCF     EECON1, EEPGD  ; Point to DATA
                               ; memory
        BSF     EECON1, WREN   ; Enable writes

        BCF     INTCON, GIE    ; Disable INTs.
Required MOVLW  55h            ;
Sequence MOVWF  EECON2         ; Write 55h
        MOVLW   AAh            ;
        MOVWF   EECON2         ; Write AAh
        BSF     EECON1, WR     ; Set WR bit to
                               ; begin write
        BSF     INTCON, GIE    ; Enable INTs.
        BCF     EECON1, WREN   ; Disable writes
```

3.5 Reading Flash Program Memory

To read a program memory location, the user must write two bytes of the address to the EEADR and EEADRH registers, set the EEPGD control bit (EECON1<7>) and then set control bit RD (EECON1<0>). Once the read control bit is set, the program memory Flash controller will use the second instruction cycle to read the data. This causes the second instruction immediately following the "`BSF EECON1,RD`" instruction to be ignored. The data is available in the very next cycle in the EEDATA and EEDATH registers; therefore, it can be read as two bytes in the following instructions. EEDATA and EEDATH registers will hold this value until another read or until it is written to by the user (during a write operation).

EXAMPLE 3-3: FLASH PROGRAM READ

```
BANKSEL EEADRH        ; Select Bank of EEADRH
MOVF    ADDRH, W      ;
MOVWF   EEADRH        ; MS Byte of Program
                      ; Address to read
MOVF    ADDRL, W      ;
MOVWF   EEADR         ; LS Byte of Program
                      ; Address to read
BANKSEL EECON1        ; Select Bank of EECON1
BSF     EECON1, EEPGD ; Point to PROGRAM
                      ; memory
BSF     EECON1, RD    ; EE Read
                      ;
NOP                   ; Any instructions
                      ; here are ignored as
NOP                   ; program memory is
                      ; read in second cycle
                      ; after BSF EECON1,RD
BANKSEL EEDATA        ; Select Bank of EEDATA
MOVF    EEDATA, W     ; DATAL = EEDATA
MOVWF   DATAL         ;
MOVF    EEDATH, W     ; DATAH = EEDATH
MOVWF   DATAH         ;
```

3.6 Erasing Flash Program Memory

The minimum erase block is 32 words. Only through the use of an external programmer, or through ICSP control, can larger blocks of program memory be bulk erased. Word erase in the Flash array is not supported.

When initiating an erase sequence from the micro-controller itself, a block of 32 words of program memory is erased. The Most Significant 11 bits of the EEADRH:EEADR point to the block being erased. EEADR< 4:0> are ignored.

The EECON1 register commands the erase operation. The EEPGD bit must be set to point to the Flash program memory. The WREN bit must be set to enable write operations. The FREE bit is set to select an erase operation.

For protection, the write initiate sequence for EECON2 must be used.

After the "`BSF EECON1,WR`" instruction, the processor requires two cycles to setup the erase operation. The user must place two `NOP` instructions after the WR bit is set. The processor will halt internal operations for the typical 2 ms, only during the cycle in which the erase takes place. This is not Sleep mode, as the clocks and peripherals will continue to run. After the erase cycle, the processor will resume operation with the third instruction after the EECON1 write instruction.

3.6.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

The sequence of events for erasing a block of internal program memory location is:

1. Load EEADRH:EEADR with address of row being erased.
2. Set EEPGD bit to point to program memory, set WREN bit to enable writes and set FREE bit to enable the erase.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write AAh to EECON2.
6. Set the WR bit. This will begin the row erase cycle.
7. The CPU will stall for duration of the erase.

EXAMPLE 3-4: ERASING A FLASH PROGRAM MEMORY ROW

```
          BANKSEL  EEADRH          ; Select Bank of EEADRH
          MOVF     ADDRH, W        ;
          MOVWF    EEADRH          ; MS Byte of Program Address to Erase
          MOVF     ADDRL, W        ;
          MOVWF    EEADR           ; LS Byte of Program Address to Erase
ERASE_ROW
          BANKSEL  EECON1          ; Select Bank of EECON1
          BSF      EECON1, EEPGD   ; Point to PROGRAM memory
          BSF      EECON1, WREN    ; Enable Write to memory
          BSF      EECON1, FREE    ; Enable Row Erase operation
;
          BCF      INTCON, GIE     ; Disable interrupts (if using)
          MOVLW    55h             ;
          MOVWF    EECON2          ; Write 55h
          MOVLW    AAh             ;
          MOVWF    EECON2          ; Write AAh
          BSF      EECON1, WR      ; Start Erase (CPU stall)
          NOP                      ; Any instructions here are ignored as processor
                                   ; halts to begin Erase sequence
          NOP                      ; processor will stop here and wait for Erase complete
                                   ; after Erase processor continues with 3rd instruction
          BCF      EECON1, FREE    ; Disable Row Erase operation
          BCF      EECON1, WREN    ; Disable writes
          BSF      INTCON, GIE     ; Enable interrupts (if using)
```

3.7 Writing to Flash Program Memory

Flash program memory may only be written to if the destination address is in a segment of memory that is not write-protected, as defined in bits WRT1:WRT0 of the device Configuration Word (Register 15-1). Flash program memory must be written in four-word blocks. A block consists of four words with sequential addresses, with a lower boundary defined by an address, where EEADR<1:0> = `00`. At the same time, all block writes to program memory are done as write-only operations. The program memory must first be erased. The write operation is edge-aligned and cannot occur across boundaries.

To write to the program memory, the data must first be loaded into the buffer registers. There are four 14-bit buffer registers and they are addressed by the low 2 bits of EEADR.

The following sequence of events illustrate how to perform a write to program memory:

- Set the EEPGD and WREN bits in the EECON1 register
- Clear the FREE bit in EECON1
- Write address to EEADRH:EEADR
- Write data to EEDATH:EEDATA
- Write 55 to EECON2
- Write AA to EECON2
- Set WR bit in EECON1

The user must follow the same specific sequence to initiate the write for each word in the program block by writing each program word in sequence (`00`, `01`, `10`, `11`).

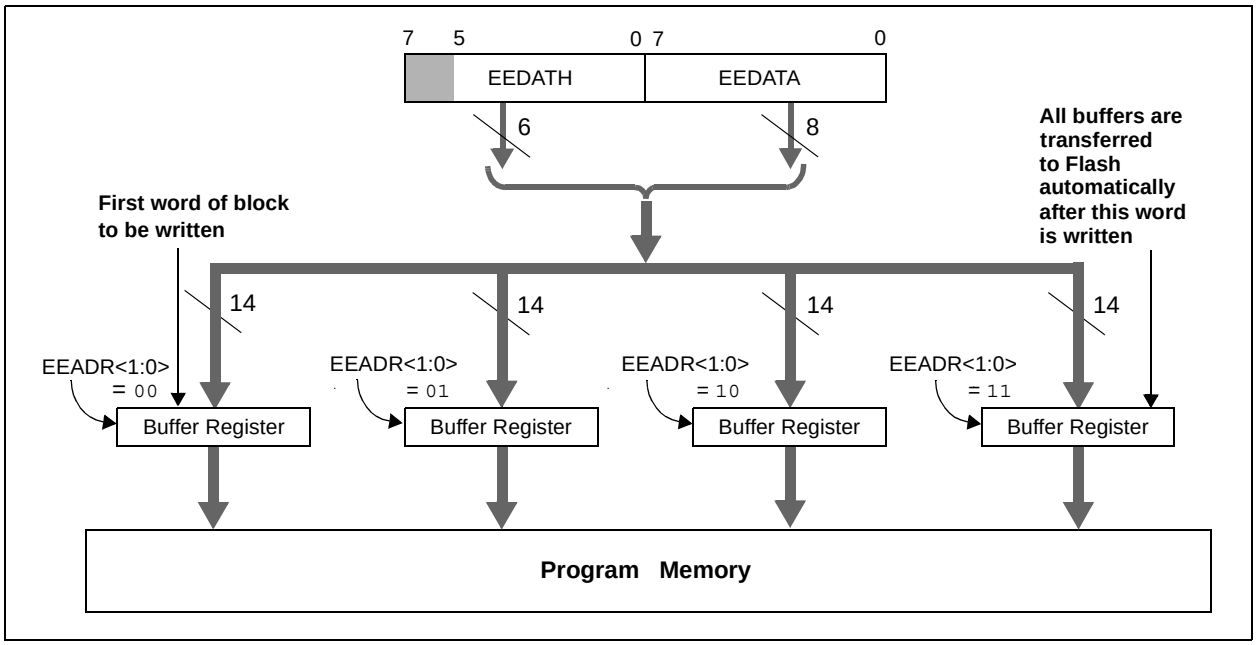
There are 4 buffer register words and all four locations **MUST** be written to with correct data.

After the "`BSF EECON1, WR`" instruction, if EEADR ≠ `xxxxxx11`, then a short write will occur. This short write only transfers the data to the buffer register. The WR bit will be cleared in hardware after 1 cycle.

After the "`BSF EECON1, WR`" instruction, if EEADR = `xxxxxx11`, then a long write will occur. This will simultaneously transfer the data from EEDATH:EEDATA to the buffer registers and begin the write of all four words. The processor will execute the next instruction and then ignore the subsequent instruction. The user should place `NOP` instructions into the second words. The processor will then halt internal operations for typically 2 msec in which the write takes place. This is not Sleep mode, as the clocks and peripherals will continue to run. After the write cycle, the processor will resume operation with the 3rd instruction after the EECON1 write instruction.

After each long write, the 4 buffer registers will be reset to 3FFF.

FIGURE 3-1: BLOCK WRITES TO FLASH PROGRAM MEMORY



An example of the complete four-word write sequence is shown in Example 3-5. The initial address is loaded into the EEADRH:EEADR register pair; the four words of data are loaded using indirect addressing, assuming that a row erase sequence has already been performed.

EXAMPLE 3-5: WRITING TO FLASH PROGRAM MEMORY

```
; This write routine assumes the following:

; 1. The 32 words in the erase block have already been erased.
; 2. A valid starting address (the least significant bits = '00') is loaded into EEADRH:EEADR
; 3. This example is starting at 0x100, this is an application dependent setting.
; 4. The 8 bytes (4 words) of data are loaded, starting at an address in RAM called ARRAY.
; 5. This is an example only, location of data to program is application dependent.
; 6. word_block is located in data memory.

        BANKSEL   EECON1           ;prepare for WRITE procedure
        BSF       EECON1, EEPGD    ;point to program memory
        BSF       EECON1, WREN     ;allow write cycles
        BCF       EECON1, FREE     ;perform write only

        BANKSEL   word_block
        MOVLW     .4
        MOVWF     word_block       ;prepare for 4 words to be written

        BANKSEL   EEADRH           ;Start writing at 0x100
        MOVLW     0x01
        MOVWF     EEADRH           ;load HIGH address
        MOVLW     0x00
        MOVWF     EEADR            ;load LOW address
        BANKSEL   ARRAY
        MOVLW     ARRAY            ;initialize FSR to start of data
        MOVWF     FSR
LOOP
        BANKSEL   EEDATA
        MOVF      INDF, W          ;indirectly load EEDATA
        MOVWF     EEDATA
        INCF      FSR, F           ;increment data pointer
        MOVF      INDF, W          ;indirectly load EEDATH
        MOVWF     EEDATH
        INCF      FSR, F           ;increment data pointer

        BANKSEL   EECON1
        MOVLW     0x55             ;required sequence
        MOVWF     EECON2
        MOVLW     0xAA
        MOVWF     EECON2
        BSF       EECON1, WR       ;set WR bit to begin write
        NOP                        ;instructions here are ignored as processor
        NOP

        BANKSEL   EEADR
        INCF      EEADR, f         ;load next word address
        BANKSEL   word_block
        DECFSZ    word_block, f    ;have 4 words been written?
        GOTO      loop             ;NO, continue with writing

        BANKSEL   EECON1
        BCF       EECON1, WREN     ;YES, 4 words complete, disable writes
        BSF       INTCON,GIE       ;enable interrupts
```

Required Sequence

3.8 Protection Against Spurious Write

There are conditions when the device should not write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built-in. On power-up, WREN is cleared. Also, the Power-up Timer (72 ms duration) prevents an EEPROM write.

The write initiate sequence and the WREN bit together help prevent an accidental write during brown-out, power glitch or software malfunction.

3.9 Operation During Code-Protect

When the data EEPROM is code-protected, the microcontroller can read and write to the EEPROM normally. However, all external access to the EEPROM is disabled. External write access to the program memory is also disabled.

When program memory is code-protected, the microcontroller can read and write to program memory normally, as well as execute instructions. Writes by the device may be selectively inhibited to regions of the memory depending on the setting of bits WRT1:WRT0 of the Configuration Word (see **Section 15.1 "Configuration Bits"** for additional information). External access to the memory is also disabled.

TABLE 3-1: REGISTERS/BITS ASSOCIATED WITH DATA EEPROM AND FLASH PROGRAM MEMORIES

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other Resets
10Ch	EEDATA	EEPROM/Flash Data Register Low Byte								xxxx xxxx	uuuu uuuu
10Dh	EEADR	EEPROM/Flash Address Register Low Byte								xxxx xxxx	uuuu uuuu
10Eh	EEDATH	—	—	EEPROM/Flash Data Register High Byte						--xx xxxx	--uu uuuu
10Fh	EEADRH	—	—	—	—	EEPROM/Flash Address Register High Byte				---- xxxx	---- uuuu
18Ch	EECON1	EEPGD	—	—	FREE	WRERR	WREN	WR	RD	x--x x000	x--x q000
18Dh	EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	---- ----
0Dh	PIR2	OSFIF	CMIF	—	EEIF	—	—	—	—	00-0 ----	00-0 ----
8Dh	PIE2	OSFIE	CMIE	—	EEIE	—	—	—	—	00-0 ----	00-0 ----

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0', q = value depends upon condition.
Shaded cells are not used by data EEPROM or Flash program memory.

4.0 OSCILLATOR CONFIGURATIONS

4.1 Oscillator Types

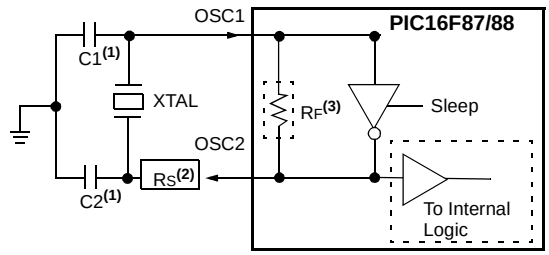
The PIC16F87/88 can be operated in eight different oscillator modes. The user can program three configuration bits (FOSC2:FOSC0) to select one of these eight modes (modes 5-8 are new PIC16 oscillator configurations):

1. LP — Low-Power Crystal
2. XT — Crystal/Resonator
3. HS — High-Speed Crystal/Resonator
4. RC — External Resistor/Capacitor with FOSC/4 output on RA6
5. RCIO — External Resistor/Capacitor with I/O on RA6
6. INTIO1 — Internal Oscillator with FOSC/4 output on RA6 and I/O on RA7
7. INTIO2 — Internal Oscillator with I/O on RA6 and RA7
8. ECIO — External Clock with I/O on RA6

4.2 Crystal Oscillator/Ceramic Resonators

In XT, LP or HS modes, a crystal or ceramic resonator is connected to the OSC1/CLKI and OSC2/CLKO pins to establish oscillation (see Figure 4-1 and Figure 4-2). The PIC16F87/88 oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturer's specifications.

FIGURE 4-1: CRYSTAL OPERATION (HS, XT, OR LP OSCILLATOR CONFIGURATION)



Note1: See Table 4-1 for typical values of C1 and C2.

2: A series resistor (RS) may be required for AT strip cut crystals.

3: RF varies with the crystal chosen (typically between 2 MΩ to 10 MΩ).

TABLE 4-1: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR (FOR DESIGN GUIDANCE ONLY)

Osc Type	Crystal Freq	Typical Capacitor Values Tested:	
		C1	C2
LP	32 kHz	33 pF	33 pF
XT	200 kHz	56 pF	56 pF
	1 MHz	15 pF	15 pF
	4 MHz	15 pF	15 pF
HS	4 MHz	15 pF	15 pF
	8 MHz	15 pF	15 pF
	20 MHz	15 pF	15 pF

Capacitor values are for design guidance only.

These capacitors were tested with the crystals listed below for basic start-up and operation. These values were not optimized.

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.

See the notes following this table for additional information.

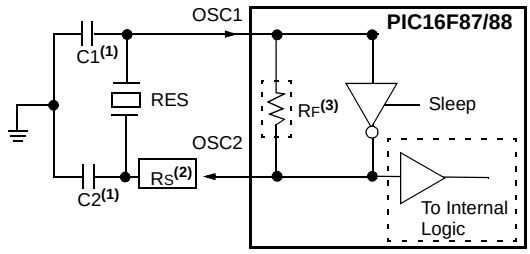
Note 1: Higher capacitance increases the stability of oscillator but also increases the start-up time.

2: Since each crystal has its own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.

3: RS may be required in HS mode, as well as XT mode, to avoid overdriving crystals with low drive level specification.

4: Always verify oscillator performance over the VDD and temperature range that is expected for the application.

FIGURE 4-2: CERAMIC RESONATOR OPERATION (HS OR XT OSC CONFIGURATION)



Note 1: See Table 4-2 for typical values of C1 and C2.

2: A series resistor (R_S) may be required.

3: R_F varies with the resonator chosen (typically between 2 MΩ to 10 MΩ).

TABLE 4-2: CERAMIC RESONATORS (FOR DESIGN GUIDANCE ONLY)

Typical Capacitor Values Used:			
Mode	**Freq**	**OSC1**	**OSC2**
XT	455 kHz 2.0 MHz 4.0 MHz	56 pF 47 pF 33 pF	56 pF 47 pF 33 pF
HS	8.0 MHz 16.0 MHz	27 pF 22 pF	27 pF 22 pF

Capacitor values are for design guidance only.

These capacitors were tested with the resonators listed below for basic start-up and operation. These values were not optimized.

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected V_{DD} and temperature range for the application.

See the notes following this table for additional information.

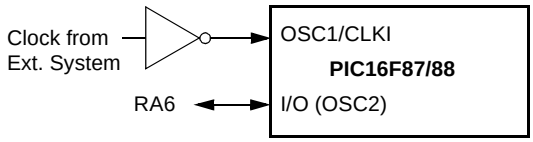
Note: When using resonators with frequencies above 3.5 MHz, the use of HS mode, rather than XT mode, is recommended. HS mode may be used at any V_{DD} for which the controller is rated. If HS is selected, it is possible that the gain of the oscillator will overdrive the resonator. Therefore, a series resistor should be placed between the OSC2 pin and the resonator. As a good starting point, the recommended value of R_S is 330Ω.

4.3 External Clock Input

The ECIO Oscillator mode requires an external clock source to be connected to the OSC1 pin. There is no oscillator start-up time required after a Power-on Reset, or after an exit from Sleep mode.

In the ECIO Oscillator mode, the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6). Figure 4-3 shows the pin connections for the ECIO Oscillator mode.

FIGURE 4-3: EXTERNAL CLOCK INPUT OPERATION (ECIO CONFIGURATION)

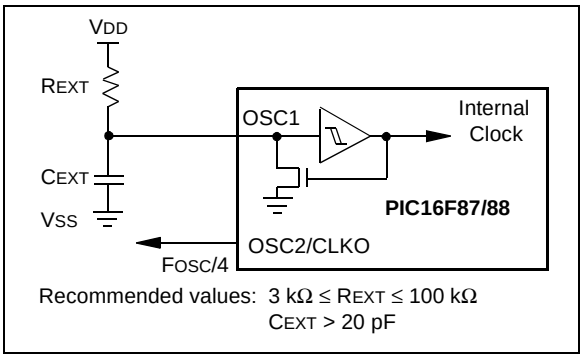


4.4 RC Oscillator

For timing insensitive applications, the “RC” and “RCIO” device options offer additional cost savings. The RC oscillator frequency is a function of the supply voltage, the resistor (REXT) and capacitor (CEXT) values and the operating temperature. In addition to this, the oscillator frequency will vary from unit to unit due to normal manufacturing variation. Furthermore, the difference in lead frame capacitance between package types will also affect the oscillation frequency, especially for low CEXT values. The user also needs to take into account variation due to tolerance of external R and C components used. Figure 4-4 shows how the R/C combination is connected.

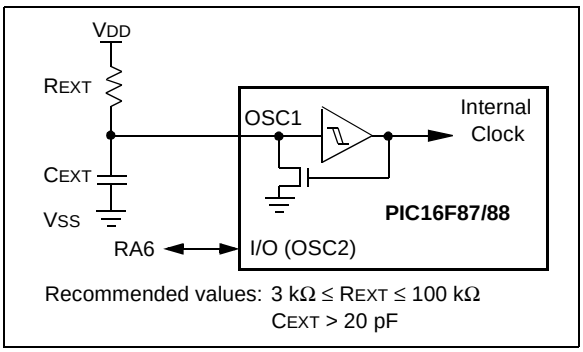
In the RC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic.

FIGURE 4-4: RC OSCILLATOR MODE



The RCIO Oscillator mode (Figure 4-5) functions like the RC mode, except that the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6).

FIGURE 4-5: RCIO OSCILLATOR MODE



4.5 Internal Oscillator Block

The PIC16F87/88 devices include an internal oscillator block which generates two different clock signals; either can be used as the system’s clock source. This can eliminate the need for external oscillator circuits on the OSC1 and/or OSC2 pins.

The main output (INTOSC) is an 8 MHz clock source which can be used to directly drive the system clock. It also drives the INTOSC postscaler which can provide a range of six clock frequencies from 125 kHz to 4 MHz.

The other clock source is the internal RC oscillator (INTRC) which provides a 31.25 kHz (32 µs nominal period) output. The INTRC oscillator is enabled by selecting the INTRC as the system clock source or when any of the following are enabled:

- Power-up Timer
- Watchdog Timer
- Two-Speed Start-up
- Fail-Safe Clock Monitor

These features are discussed in greater detail in **Section 15.0 “Special Features of the CPU”**.

The clock source frequency (INTOSC direct, INTRC direct or INTOSC postscaler) is selected by configuring the IRCF bits of the OSCCON register (page 40).

Note:	Throughout this data sheet, when referring *specifically* to a generic clock source, the term “INTRC” may also be used to refer to the clock modes using the internal oscillator block. This is regardless of whether the actual frequency used is INTOSC (8 MHz), the INTOSC postscaler or INTRC (31.25 kHz).

4.5.1 INTRC MODES

Using the internal oscillator as the clock source can eliminate the need for up to two external oscillator pins, after which it can be used for digital I/O. Two distinct configurations are available:

- In INTIO1 mode, the OSC2 pin outputs FOSC/4, while OSC1 functions as RA7 for digital input and output.
- In INTIO2 mode, OSC1 functions as RA7 and OSC2 functions as RA6, both for digital input and output.

4.5.2 OSCTUNE REGISTER

The internal oscillator's output has been calibrated at the factory but can be adjusted in the application. This is done by writing to the OSCTUNE register (Register 4-1). The tuning sensitivity is constant throughout the tuning range. The OSCTUNE register has a tuning range of ±12.5%.

When the OSCTUNE register is modified, the INTOSC and INTRC frequencies will begin shifting to the new frequency. The INTRC clock will reach the new frequency within 8 clock cycles (approximately 8 * 32 µs = 256 µs); the INTOSC clock will stabilize within 1 ms. Code execution continues during this shift. There is no indication that the shift has occurred. Operation of features that depend on the 31.25 kHz INTRC clock source frequency, such as the WDT, Fail-Safe Clock Monitor and peripherals, will also be affected by the change in frequency.

REGISTER 4-1: OSCTUNE: OSCILLATOR TUNING REGISTER (ADDRESS 90h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-0 **TUN<5:0>:** Frequency Tuning bits

011111 = Maximum frequency
011110 =
•
•
•
000001 =
000000 = Center frequency. Oscillator module is running at the calibrated frequency.
111111 =
•
•
•
100000 = Minimum frequency

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

4.6 Clock Sources and Oscillator Switching

The PIC16F87/88 devices include a feature that allows the system clock source to be switched from the main oscillator to an alternate low-frequency clock source. PIC16F87/88 devices offer three alternate clock sources. When enabled, these give additional options for switching to the various power-managed operating modes.

Essentially, there are three clock sources for these devices:

- Primary oscillators
- Secondary oscillators
- Internal oscillator block (INTRC)

The **primary oscillators** include the External Crystal and Resonator modes, the External RC modes, the External Clock mode and the internal oscillator block. The particular mode is defined on POR by the contents of Configuration Word 1. The details of these modes are covered earlier in this chapter.

The **secondary oscillators** are those external sources not connected to the OSC1 or OSC2 pins. These sources may continue to operate even after the controller is placed in a power-managed mode.

PIC16F87/88 devices offer the Timer1 oscillator as a secondary oscillator. This oscillator continues to run when a `SLEEP` instruction is executed and is often the time base for functions such as a real-time clock.

Most often, a 32.768 kHz watch crystal is connected between the RB6/T1OSO and RB7/T1OSI pins. Like the LP mode oscillator circuit, loading capacitors are also connected from each pin to ground. The Timer1 oscillator is discussed in greater detail in **Section 7.6 “Timer1 Oscillator”**.

In addition to being a primary clock source, the **internal oscillator block** is available as a power-managed mode clock source. The 31.25 kHz INTRC source is also used as the clock source for several special features, such as the WDT, Fail-Safe Clock Monitor, Power-up Timer and Two-Speed Start-up.

The clock sources for the PIC16F87/88 devices are shown in Figure 4-6. See **Section 7.0 “Timer1 Module”** for further details of the Timer1 oscillator. See **Section 15.1 “Configuration Bits”** for Configuration register details.

4.6.1 OSCCON REGISTER

The OSCCON register (Register 4-2) controls several aspects of the system clock’s operation, both in full power operation and in power-managed modes.

The System Clock Select bits, SCS1:SCS0, select the clock source that is used when the device is operating in power-managed modes. When the bits are cleared (SCS<1:0> = `00`), the system clock source comes from the main oscillator that is selected by the FOSC2:FOSC0 configuration bits in Configuration Word 1 register. When the bits are set in any other manner, the system clock source is provided by the Timer1 oscillator (SCS1:SCS0 = `01`) or from the internal oscillator block (SCS1:SCS0 = `10`). After a Reset, SCS<1:0> are always set to ‘`00`’.

Note:	The instruction to immediately follow the modification of SCS<1:0> will have an instruction time (TCY) based on the previous clock source. This should be taken into consideration when developing time dependant code.

The Internal Oscillator Select bits, IRCF2:IRCF0, select the frequency output of the internal oscillator block that is used to drive the system clock. The choices are the INTRC source (31.25 kHz), the INTOSC source (8 MHz) or one of the six frequencies derived from the INTOSC postscaler (125 kHz to 4 MHz). Changing the configuration of these bits has an immediate change on the multiplexor’s frequency output.

The OSTS and IOFS bits indicate the status of the primary oscillator and INTOSC source; these bits are set when their respective oscillators are stable. In particular, OSTS indicates that the Oscillator Start-up Timer has timed out.

4.6.2 CLOCK SWITCHING

Clock switching will occur for the following reasons:

- The FCMEN (CONFIG2<0>) bit is set, the device is running from the primary oscillator and the primary oscillator fails. The clock source will be the internal RC oscillator.
- The FCMEN bit is set, the device is running from the T1OSC and T1OSC fails. The clock source will be the internal RC oscillator.
- Following a wake-up due to a Reset or a POR, when the device is configured for Two-Speed Start-up mode, switching will occur between the INTRC and the system clock defined by the FOSC<2:0> bits.
- A wake-up from Sleep occurs due to an interrupt or WDT wake-up and Two-Speed Start-up is enabled. If the primary clock is XT, HS or LP, the clock will switch between the INTRC and the primary system clock after 1024 clocks (OST) and 8 clocks of the primary oscillator. This is conditional upon the SCS bits being set equal to ‘`00`’.
- SCS bits are modified from their original value.
- IRCF bits are modified from their original value.

Note:	Because the SCS bits are cleared on any Reset, no clock switching will occur on a Reset unless the Two-Speed Start-up is enabled and the primary clock is XT, HS or LP. The device will wait for the primary clock to become stable before execution begins (Two-Speed Start-up disabled).

4.6.3 CLOCK TRANSITION AND WDT

When clock switching is performed, the Watchdog Timer is disabled because the Watchdog ripple counter is used as the Oscillator Start-up Timer.

Note:	The OST is only used when switching to XT, HS and LP Oscillator modes.

Once the clock transition is complete (i.e., new oscillator selection switch has occurred), the Watchdog counter is re-enabled with the counter reset. This allows the user to synchronize the Watchdog Timer to the start of execution at the new clock frequency.

REGISTER 4-2: OSCCON: OSCILLATOR CONTROL REGISTER (ADDRESS 8Fh)

U-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0
—	IRCF2	IRCF1	IRCF0	OSTS(1)	IOFS	SCS1	SCS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6-4 **IRCF<2:0>:** Internal RC Oscillator Frequency Select bits
000 = 31.25 kHz
001 = 125 kHz
010 = 250 kHz
011 = 500 kHz
100 = 1 MHz
101 = 2 MHz
110 = 4 MHz
111 = 8 MHz

bit 3 **OSTS:** Oscillator Start-up Time-out Status bit(1)
1 = Device is running from the primary system clock
0 = Device is running from T1OSC or INTRC as a secondary system clock

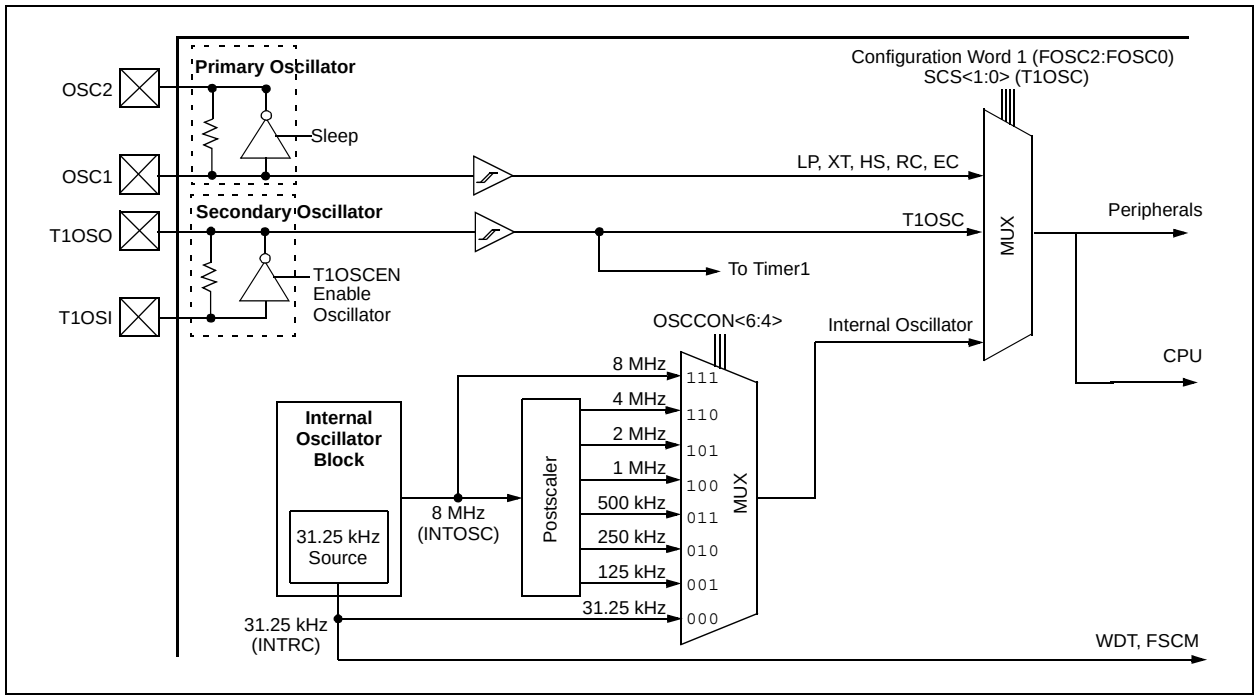
Note 1: Bit resets to '0' with Two-Speed Start-up mode and LP, XT or HS selected as the oscillator mode.

bit 2 **IOFS:** INTOSC Frequency Stable bit
1 = Frequency is stable
0 = Frequency is not stable

bit 1-0 **SCS<1:0>:** Oscillator Mode Select bits
00 = Oscillator mode defined by FOSC<2:0>
01 = T1OSC is used for system clock
10 = Internal RC is used for system clock
11 = Reserved

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

FIGURE 4-6: PIC16F87/88 CLOCK DIAGRAM



4.6.4 MODIFYING THE IRCF BITS

The IRCF bits can be modified at any time regardless of which clock source is currently being used as the system clock. The internal oscillator allows users to change the frequency during run time. This is achieved by modifying the IRCF bits in the OSCCON register. The sequence of events that occur after the IRCF bits are modified is dependent upon the initial value of the IRCF bits before they are modified. If the INTRC (31.25 kHz, IRCF<2:0> = 000) is running and the IRCF bits are modified to any other value than '000', a 4 ms (approx.) clock switch delay is turned on. Code execution continues at a higher than expected frequency while the new frequency stabilizes. Time sensitive code should wait for the IOFS bit in the OSCCON register to become set before continuing. This bit can be monitored to ensure that the frequency is stable before using the system clock in time critical applications.

If the IRCF bits are modified while the internal oscillator is running at any other frequency than INTRC (31.25 kHz, IRCF<2:0> ≠ 000), there is no need for a 4 ms (approx.) clock switch delay. The new INTOSC frequency will be stable immediately after the **eight** falling edges. The IOFS bit will remain set after clock switching occurs.

Note:	Caution must be taken when modifying the IRCF bits using BCF or BSF instructions. It is possible to modify the IRCF bits to a frequency that may be out of the VDD specification range; for example, VDD = 2.0V and IRCF = 111 (8 MHz).

4.6.5 CLOCK TRANSITION SEQUENCE

Following are three different sequences for switching the internal RC oscillator frequency.

- Clock before switch: 31.25 kHz (IRCF<2:0> = 000)

1. IRCF bits are modified to an INTOSC/INTOSC postscaler frequency.
2. The clock switching circuitry waits for a falling edge of the current clock, at which point CLKO is held low.
3. The clock switching circuitry then waits for eight falling edges of requested clock, after which it switches CLKO to this new clock source.
4. The IOFS bit is clear to indicate that the clock is unstable and a 4 ms (approx.) delay is started. Time dependent code should wait for IOFS to become set.
5. Switchover is complete.

- Clock before switch: One of INTOSC/INTOSC postscaler (IRCF<2:0> ≠ 000)

1. IRCF bits are modified to INTRC (IRCF<2:0> = 000).
2. The clock switching circuitry waits for a falling edge of the current clock, at which point CLKO is held low.
3. The clock switching circuitry then waits for eight falling edges of requested clock, after which it switches CLKO to this new clock source.
4. Oscillator switchover is complete.

- Clock before switch: One of INTOSC/INTOSC postscaler (IRCF<2:0> ≠ 000)

1. IRCF bits are modified to a different INTOSC/INTOSC postscaler frequency.
2. The clock switching circuitry waits for a falling edge of the current clock, at which point CLKO is held low.
3. The clock switching circuitry then waits for eight falling edges of requested clock, after which it switches CLKO to this new clock source.
4. The IOFS bit is set.
5. Oscillator switchover is complete.

4.6.6 OSCILLATOR DELAY UPON POWER-UP, WAKE-UP AND CLOCK SWITCHING

Table 4-3 shows the different delays invoked for various clock switching sequences. It also shows the delays invoked for POR and wake-up.

TABLE 4-3: OSCILLATOR DELAY EXAMPLES

Clock Switch		Frequency	Oscillator Delay	Comments
From	**To**			
Sleep/POR	INTRC T1OSC	31.25 kHz 32.768 kHz	CPU Start-up(1)	Following a wake-up from Sleep mode or POR, CPU start-up is invoked to allow the CPU to become ready for code execution.
	INTOSC/ INTOSC Postscaler	125 kHz-8 MHz	4 ms (approx.) and CPU Start-up(1)	
INTRC/Sleep	EC, RC	DC – 20 MHz		
INTRC (31.25 kHz)	EC, RC	DC – 20 MHz		
Sleep	LP, XT, HS	32.768 kHz-20 MHz	1024 Clock Cycles (OST)	Following a change from INTRC, an OST of 1024 cycles must occur.
INTRC (31.25 kHz)	INTOSC/ INTOSC Postscaler	125 kHz-8 MHz	4 ms (approx.)	Refer to **Section 4.6.4 “Modifying the IRCF Bits”** for further details.

Note 1: The 5-10 µs start-up delay is based on a 1 MHz system clock.

4.7 Power-Managed Modes

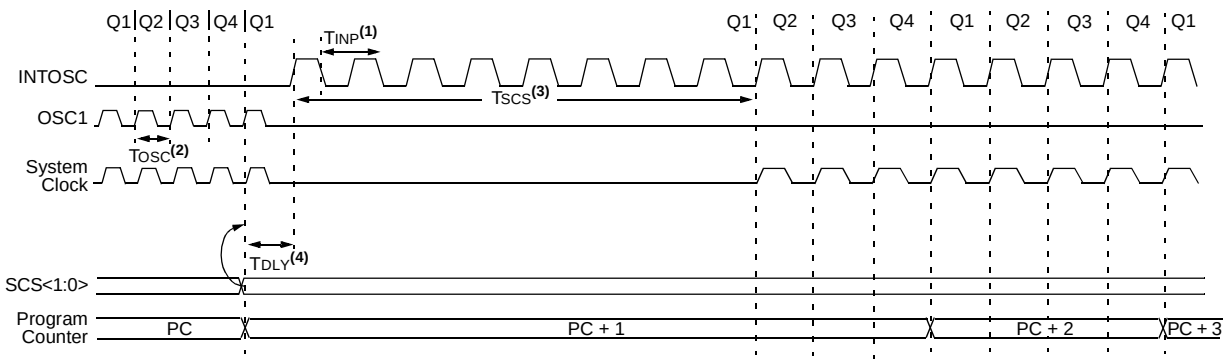
4.7.1 RC_RUN MODE

When SCS bits are configured to run from the INTRC, a clock transition is generated if the system clock is not already using the INTRC. The event will clear the OSTS bit, switch the system clock from the primary system clock (if SCS<1:0> = `00`) determined by the value contained in the configuration bits, or from the T1OSC (if SCS<1:0> = `01`) to the INTRC clock option and shut down the primary system clock to conserve power. Clock switching will not occur if the primary system clock is already configured as INTRC.

If the system clock does not come from the INTRC (31.25 kHz) when the SCS bits are changed and the IRCF bits in the OSCCON register are configured for a frequency other than INTRC, the frequency may not be stable immediately. The IOFS bit (OSCCON<2>) will be set when the INTOSC or postscaler frequency is stable, after 4 ms (approx.).

After a clock switch has been executed, the OSTS bit is cleared, indicating a low-power mode and the device does not run from the primary system clock. The internal Q clocks are held in the Q1 state until eight falling edge clocks are counted on the INTRC oscillator. After the eight clock periods have transpired, the clock input to the Q clocks is released and operation resumes (see Figure 4-7).

FIGURE 4-7: TIMING DIAGRAM FOR XT, HS, LP, EC AND EXTRC TO RC_RUN MODE



Note 1: TINP = 32 µs typical.
2: TOSC = 50 ns minimum.
3: TSCS = 8 TINP.
4: TDLY = 1 TINP.

4.7.2 SEC_RUN MODE

The core and peripherals can be configured to be clocked by T1OSC using a 32.768 kHz crystal. The crystal must be connected to the T1OSO and T1OSI pins. This is the same configuration as the low-power timer circuit (see **Section 7.6 "Timer1 Oscillator"**). When SCS bits are configured to run from T1OSC, a clock transition is generated. It will clear the OSTS bit, switch the system clock from either the primary system clock or INTRC, depending on the value of SCS<1:0> and FOSC<2:0>, to the external low-power Timer1 oscillator input (T1OSC) and shut down the primary system clock to conserve power.

After a clock switch has been executed, the internal Q clocks are held in the Q1 state until eight falling edge clocks are counted on the T1OSC. After the eight clock periods have transpired, the clock input to the Q clocks is released and operation resumes (see Figure 4-8). In addition, T1RUN (In T1CON) is set to indicate that T1OSC is being used as the system clock.

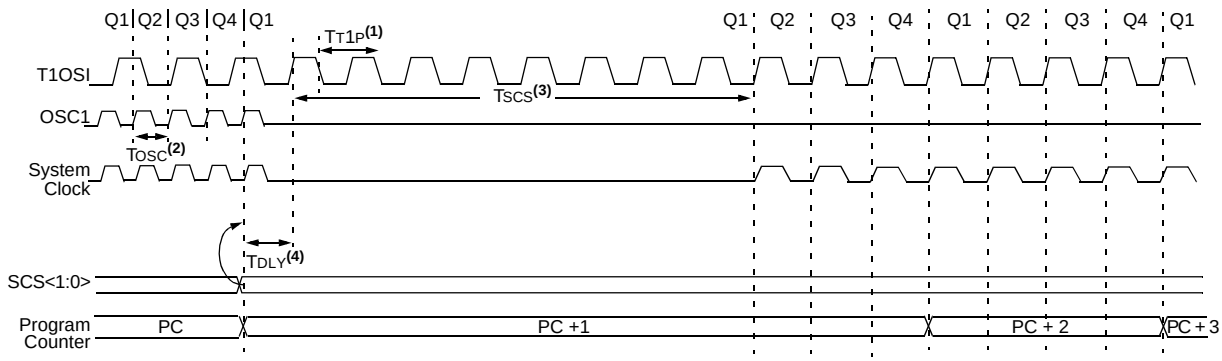
Note 1: The T1OSCEN bit must be enabled and it is the user's responsibility to ensure T1OSC is stable before clock switching to the T1OSC input clock can occur.

2: When T1OSCEN = 0, the following possible effects result.

Original SCS<1:0>	Modified SCS<1:0>	Final SCS<1:0>
00	01	00 – no change
00	11	10 – INTRC
10	11	10 – no change
10	01	00 – Oscillator defined by FOSC<2:0>

A clock switching event will occur if the final state of the SCS bits is different from the original.

FIGURE 4-8: TIMING DIAGRAM FOR SWITCHING TO SEC_RUN MODE



Note 1: TT1P = 30.52 µs.
2: TOSC = 50 ns minimum.
3: TSCS = 8 TT1P
4: TDLY = 1 TT1P.

4.7.3 SEC_RUN/RC_RUN TO PRIMARY CLOCK SOURCE

When switching from a SEC_RUN or RC_RUN mode back to the primary system clock, following a change of SCS<1:0> to '00', the sequence of events that takes place will depend upon the value of the FOSC bits in the Configuration register. If the primary clock source is configured as a crystal (HS, XT or LP), then the transition will take place after 1024 clock cycles. This is necessary because the crystal oscillator has been powered down until the time of the transition. In order to provide the system with a reliable clock when the changeover has occurred, the clock will not be released to the changeover circuit until the 1024 count has expired.

During the oscillator start-up time, the system clock comes from the current system clock. Instruction execution and/or peripheral operation continues using the currently selected oscillator as the CPU clock source, until the necessary clock count has expired, to ensure that the primary system clock is stable.

To know when the OST has expired, the OSTS bit should be monitored. OSTS = 1 indicates that the Oscillator Start-up Timer has timed out and the system clock comes from the primary clock source.

Following the oscillator start-up time, the internal Q clocks are held in the Q1 state until eight falling edge clocks are counted from the primary system clock. The clock input to the Q clocks is then released and operation resumes with the primary system clock determined by the FOSC bits (see Figure 4-10).

When in SEC_RUN mode, the act of clearing the T1OSCEN bit in the T1CON register will cause SCS<0> to be cleared, which causes the SCS<1:0> bits to revert to '00' or '10' depending on what SCS<1> is. Although the T1OSCEN bit was cleared, T1OSC will be enabled and instruction execution will continue until the OST time-out for the main system clock is complete. At that time, the system clock will switch from the T1OSC to the primary clock or the INTRC. Following this, the T1 oscillator will be shut down.

Note:	If the primary system clock is either RC or EC, an internal delay timer (5-10 µs) will suspend operation after exiting Secondary Clock mode to allow the CPU to become ready for code execution.

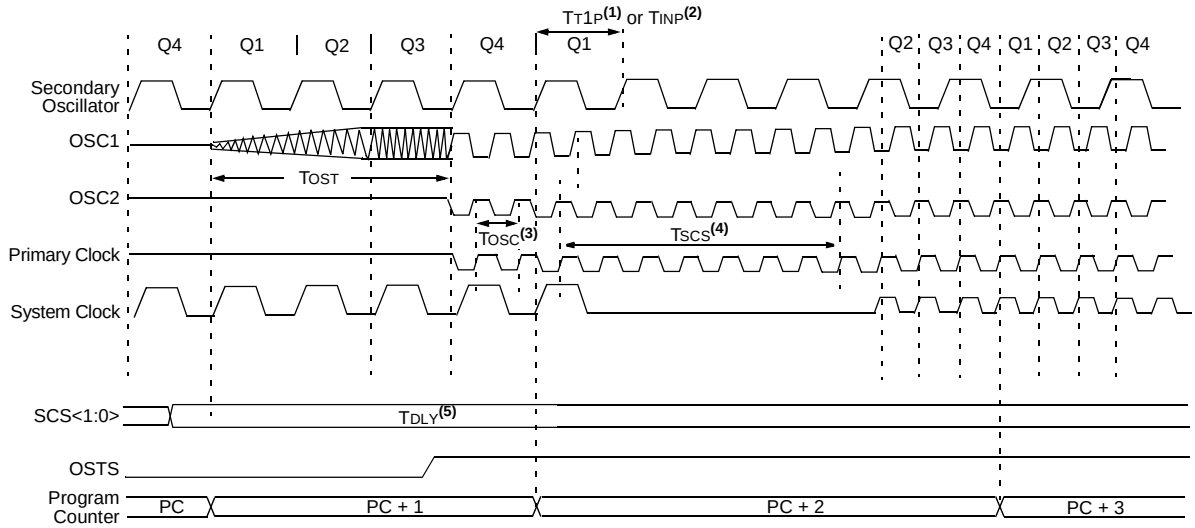
4.7.3.1 Returning to Primary Clock Source Sequence

Changing back to the primary oscillator from SEC_RUN or RC_RUN can be accomplished by either changing SCS<1:0> to '00', or clearing the T1OSCEN bit in the T1CON register (if T1OSC was the secondary clock).

The sequence of events that follows is the same for both modes:

1. If the primary system clock is configured as EC, RC or INTRC, then the OST time-out is skipped. Skip to step 3.
2. If the primary system clock is configured as an external oscillator (HS, XT, LP), then the OST will be active, waiting for 1024 clocks of the primary system clock.
3. On the following Q1, the device holds the system clock in Q1.
4. The device stays in Q1 while eight falling edges of the primary system clock are counted.
5. Once the eight counts transpire, the device begins to run from the primary oscillator.
6. If the secondary clock was INTRC and the primary is not INTRC, the INTRC will be shut down to save current providing that the INTRC is not being used for any other function, such as WDT or Fail-Safe Clock monitoring.
7. If the secondary clock was T1OSC, the T1OSC will continue to run if T1OSCEN is still set; otherwise, the T1 oscillator will be shut down.

FIGURE 4-9: TIMING FOR TRANSITION BETWEEN SEC_RUN/RC_RUN AND PRIMARY CLOCK



Note 1: TT1P = 30.52 µs.
2: TINP = 32 µs typical.
3: TOSC = 50 ns minimum.
4: TSCS = 8 TINP OR 8 TT1P.
5: TDLY = 1 TINP OR 1 TT1P.

4.7.3.2 Returning to Primary Oscillator with a Reset

A Reset will clear SCS<1:0> back to '00'. The sequence for starting the primary oscillator following a Reset is the same for all forms of Reset, including POR. There is no transition sequence from the alternate system clock to the primary system clock on a Reset condition. Instead, the device will reset the state of the OSCCON register and default to the primary system clock. The sequence of events that takes place after this will depend upon the value of the FOSC bits in the Configuration register. If the external oscillator is configured as a crystal (HS, XT or LP), the CPU will be held in the Q1 state until 1024 clock cycles have transpired on the primary clock. This is necessary because the crystal oscillator has been powered down until the time of the transition.

During the oscillator start-up time, instruction execution and/or peripheral operation is suspended.

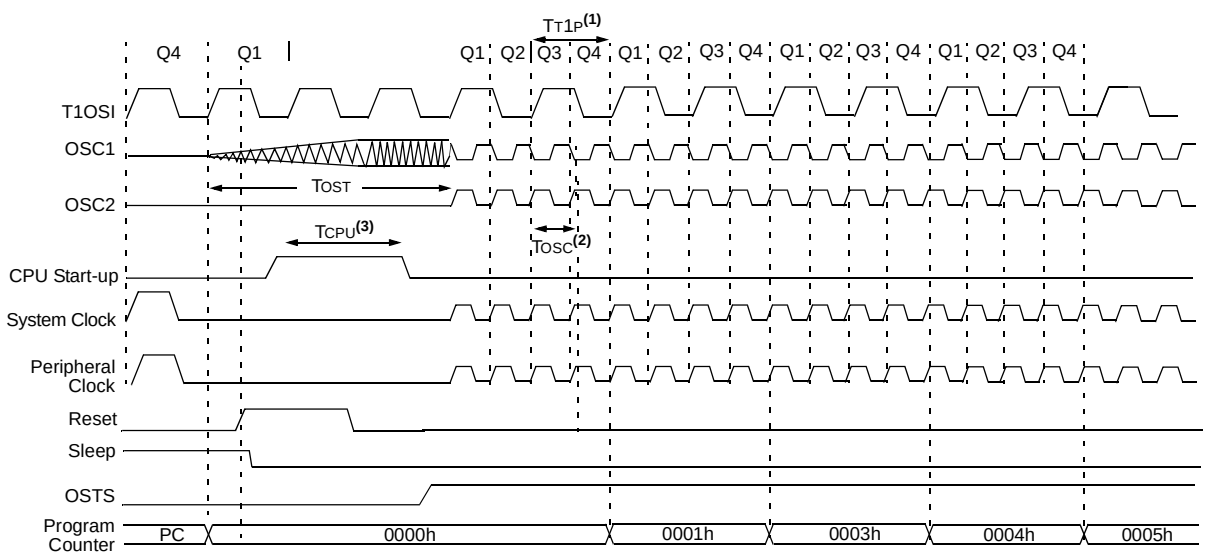
Note:	If Two-Speed Clock Start-up mode is enabled, the INTRC will act as the system clock until the OST timer has timed out.

If the primary system clock is either RC, EC or INTRC, the CPU will begin operating on the first Q1 cycle following the wake-up event. This means that there is no oscillator start-up time required because the primary clock is already stable; however, there is a delay between the wake-up event and the following Q2. An internal delay timer of 5-10 µs will suspend operation after the Reset to allow the CPU to become ready for code execution. The CPU and peripheral clock will be held in the first Q1.

The sequence of events is as follows:

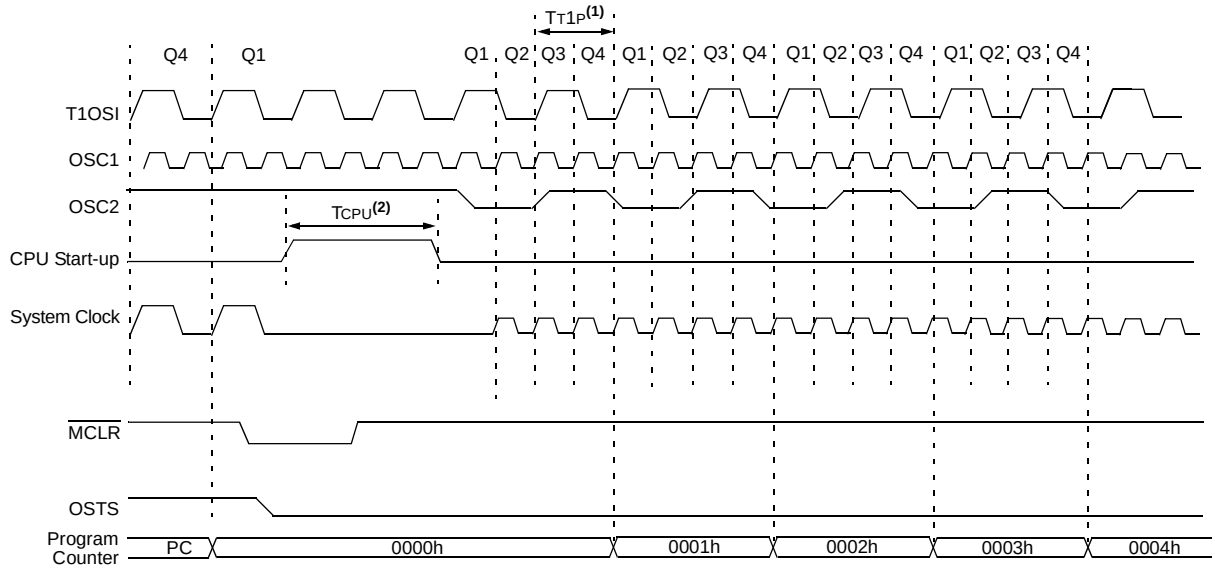
1. A device Reset is asserted from one of many sources (WDT, BOR, $\overline{\text{MCLR}}$, etc.).
2. The device resets and the CPU start-up timer is enabled if in Sleep mode. The device is held in Reset until the CPU start-up time-out is complete.
3. If the primary system clock is configured as an external oscillator (HS, XT, LP), then the OST will be active waiting for 1024 clocks of the primary system clock. While waiting for the OST, the device will be held in Reset. The OST and CPU start-up timers run in parallel.
4. After both the CPU start-up and OST timers have timed out, the device will wait for one additional clock cycle and instruction execution will begin.

FIGURE 4-10: PRIMARY SYSTEM CLOCK AFTER RESET (HS, XT, LP)



Note 1: TT1P = 30.52 µs.
2: TOSC = 50 ns minimum.
3: TCPU = 5-10 µs (1 MHz system clock).

FIGURE 4-11: PRIMARY SYSTEM CLOCK AFTER RESET (EC, RC, INTRC)



Note 1: TT1P = 30.52 µs.
2: TCPU = 5-10 µs (1 MHz system clock).

TABLE 4-4: CLOCK SWITCHING MODES

Current System Clock	SCS Bits <1:0> Modified to:	Delay	OSTS Bit	IOFS Bit	T1RUN Bit	New System Clock	Comments
LP, XT, HS, T1OSC, EC, RC	10 (INTRC) FOSC<2:0> = LP, XT or HS	8 Clocks of INTRC	0	1(1)	0	INTRC or INTOSC or INTOSC Postscaler	The internal RC oscillator frequency is dependant upon the IRCF bits.
LP, XT, HS, INTRC, EC, RC	01 (T1OSC) FOSC<2:0> = LP, XT or HS	8 Clocks of T1OSC	0	N/A	1	T1OSC	T1OSCEN bit must be enabled.
INTRC T1OSC	00 FOSC<2:0> = EC or FOSC<2:0> = RC	8 Clocks of EC or RC	1	N/A	0	EC or RC	
INTRC T1OSC	00 FOSC<2:0> = LP, XT, HS	1024 Clocks (OST) + 8 Clocks of LP, XT, HS	1	N/A	0	LP, XT, HS	During the 1024 clocks, program execution is clocked from the secondary oscillator until the primary oscillator becomes stable.
LP, XT, HS	00 (Due to Reset) LP, XT, HS	1024 Clocks (OST)	1	N/A	0	LP, XT, HS	When a Reset occurs, there is no clock transition sequence. Instruction execution and/or peripheral operation is suspended unless Two-Speed Start-up mode is enabled, after which the INTRC will act as the system clock until the OST timer has expired.

Note 1: If the new clock source is the INTOSC or INTOSC postscaler, then the IOFS bit will be set 4 ms (approx.) after the clock change.

4.7.4 EXITING SLEEP WITH AN INTERRUPT

Any interrupt, such as WDT or INT0, will cause the part to leave the Sleep mode.

The SCS bits are unaffected by a `SLEEP` command and are the same before and after entering and leaving Sleep. The clock source used after an exit from Sleep is determined by the SCS bits.

4.7.4.1 Sequence of Events

If SCS<1:0> = `00`:

1. The device is held in Sleep until the CPU start-up time-out is complete.
2. If the primary system clock is configured as an external oscillator (HS, XT, LP), then the OST will be active waiting for 1024 clocks of the primary system clock. While waiting for the OST, the device will be held in Sleep unless Two-Speed Start-up is enabled. The OST and CPU start-up timers run in parallel. Refer to **Section 15.12.3 "Two-Speed Clock Start-up Mode"** for details on Two-Speed Start-up.
3. After both the CPU start-up and OST timers have timed out, the device will exit Sleep and begin instruction execution with the primary clock defined by the FOSC bits.

If SCS<1:0> = `01` or `10`:

1. The device is held in Sleep until the CPU start-up time-out is complete.
2. After the CPU start-up timer has timed out, the device will exit Sleep and begin instruction execution with the selected oscillator mode.

Note: If a user changes SCS<1:0> just before entering Sleep mode, the system clock used when exiting Sleep mode could be different than the system clock used when entering Sleep mode.

As an example, if SCS<1:0> = `01` and T1OSC is the system clock and the following instructions are executed:

```
BCF       OSCCON, SCS0
SLEEP
```

then a clock change event is executed. If the primary oscillator is XT, LP or HS, the core will continue to run off T1OSC and execute the `SLEEP` command.

When Sleep is exited, the part will resume operation with the primary oscillator after the OST has expired.

5.0 I/O PORTS

Some pins for these I/O ports are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

Additional information on I/O ports may be found in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023).

5.1 PORTA and the TRISA Register

PORTA is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISA. Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin).

Note:	On a Power-on Reset, the pins PORTA<4:0> are configured as analog inputs and read as '0'.

Reading the PORTA register, reads the status of the pins, whereas writing to it, will write to the port latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the port data latch.

Pin RA4 is multiplexed with the Timer0 module clock input. On PIC16F88 devices, it is also multiplexed with an analog input to become the RA4/AN4/T0CKI/C2OUT pin. The RA4/AN4/T0CKI/C2OUT pin is a Schmitt Trigger input and full CMOS output driver.

Pin RA5 is multiplexed with the Master Clear module input. The RA5/$\overline{\text{MCLR}}$/VPP pin is a Schmitt Trigger input.

Pin RA6 is multiplexed with the oscillator module input and external oscillator output. Pin RA7 is multiplexed with the oscillator module input and external oscillator input. Pin RA6/OSC2/CLKO and pin RA7/OSC1/CLKI are Schmitt Trigger inputs and full CMOS output drivers.

Pins RA<1:0> are multiplexed with analog inputs. Pins RA<3:2> are multiplexed with analog inputs and comparator outputs. On PIC16F88 devices, pins RA<3:2> are also multiplexed with the VREF inputs. Pins RA<3:0> have TTL inputs and full CMOS output drivers.

EXAMPLE 5-1: INITIALIZING PORTA

```
BANKSEL PORTA     ; select bank of PORTA
CLRF    PORTA     ; Initialize PORTA by
                  ; clearing output
                  ; data latches
BANKSEL ANSEL     ; Select Bank of ANSEL
MOVLW   0x00      ; Configure all pins
MOVWF   ANSEL     ; as digital inputs

MOVLW   0xFF      ; Value used to
                  ; initialize data
                  ; direction
MOVWF   TRISA     ; Set RA<7:0> as inputs
```

TABLE 5-1: PORTA FUNCTIONS

Name	Bit#	Buffer	Function
RA0/AN0	bit 0	TTL	Input/output or analog input.
RA1/AN1	bit 1	TTL	Input/output or analog input.
RA2/AN2/CVREF/VREF-(2)	bit 2	TTL	Input/output, analog input, VREF- or comparator VREF output.
RA3/AN3/VREF+(2)/C1OUT	bit 3	TTL	Input/output, analog input, VREF+ or comparator output.
RA4/AN4(2)/T0CKI/C2OUT	bit 4	ST	Input/output, analog input, TMR0 external input or comparator output.
RA5/$\overline{\text{MCLR}}$/VPP	bit 5	ST	Input, Master Clear (Reset) or programming voltage input.
RA6/OSC2/CLKO	bit 6	ST	Input/output, connects to crystal or resonator, oscillator output or 1/4 the frequency of OSC1 and denotes the instruction cycle in RC mode.
RA7/OSC1/CLKI	bit 7	ST/CMOS(1)	Input/output, connects to crystal or resonator or oscillator input.

Legend: TTL = TTL input, ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

2: PIC16F88 only.

TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
05h	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxx 0000(1) xxx0 0000(2)	uuuu 0000(1) uuu0 0000(2)
85h	TRISA	TRISA7	TRISA6	TRISA5(3)	PORTA Data Direction Register					1111 1111	1111 1111
9Fh	ADCON1	ADFM	ADCS2	VCFG1	VCFG0	—	—	—	—	0000 ----	0000 ----
9Bh	ANSEL(4)	—	ANS6	ANS5	ANS4	ANS3	ANS2	ANS1	ANS0	-111 1111	-111 1111

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTA.

Note 1: This value applies only to the PIC16F87.

2: This value applies only to the PIC16F88.

3: Pin 5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

4: PIC16F88 device only.

FIGURE 5-1: BLOCK DIAGRAM OF RA0/AN0:RA1/AN1 PINS

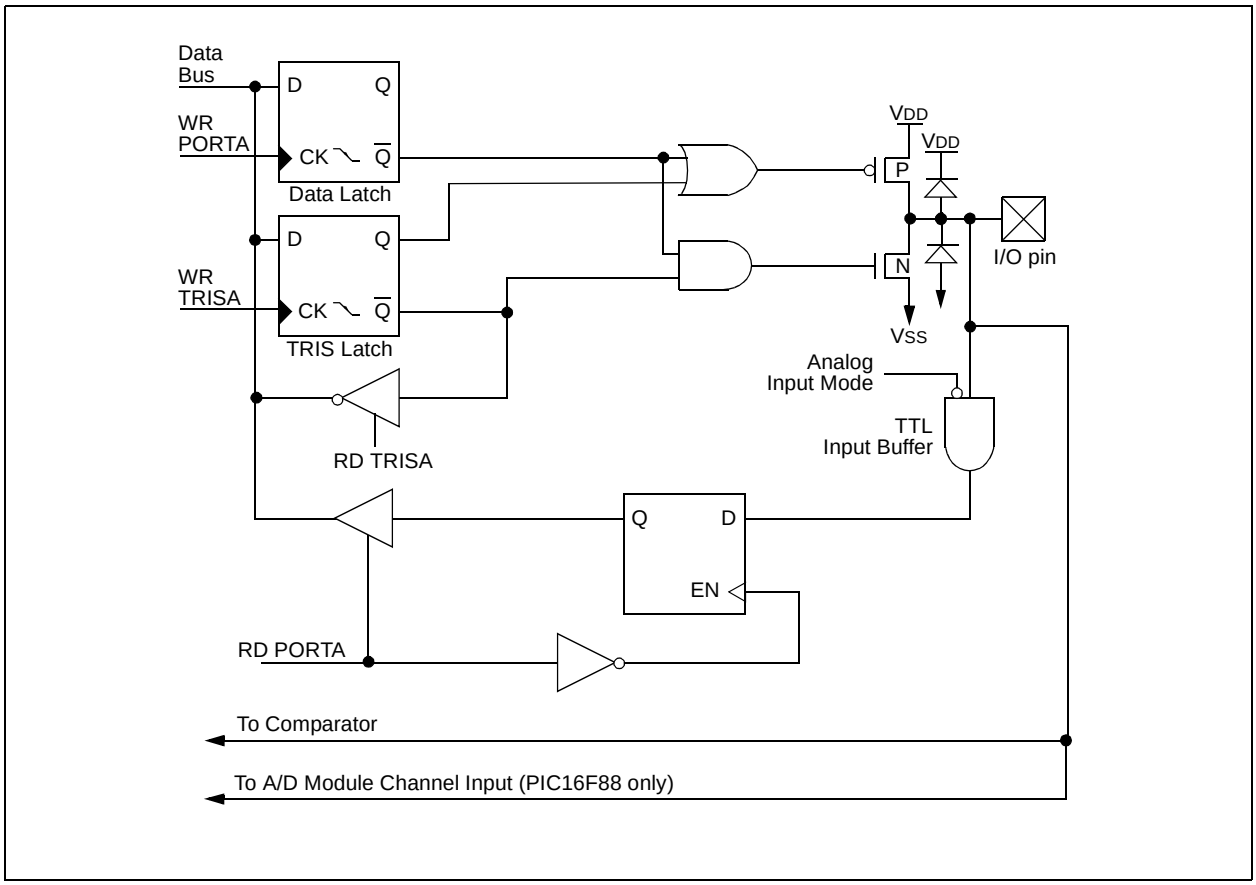


FIGURE 5-2: BLOCK DIAGRAM OF RA3/AN3/VREF+/C1OUT PIN

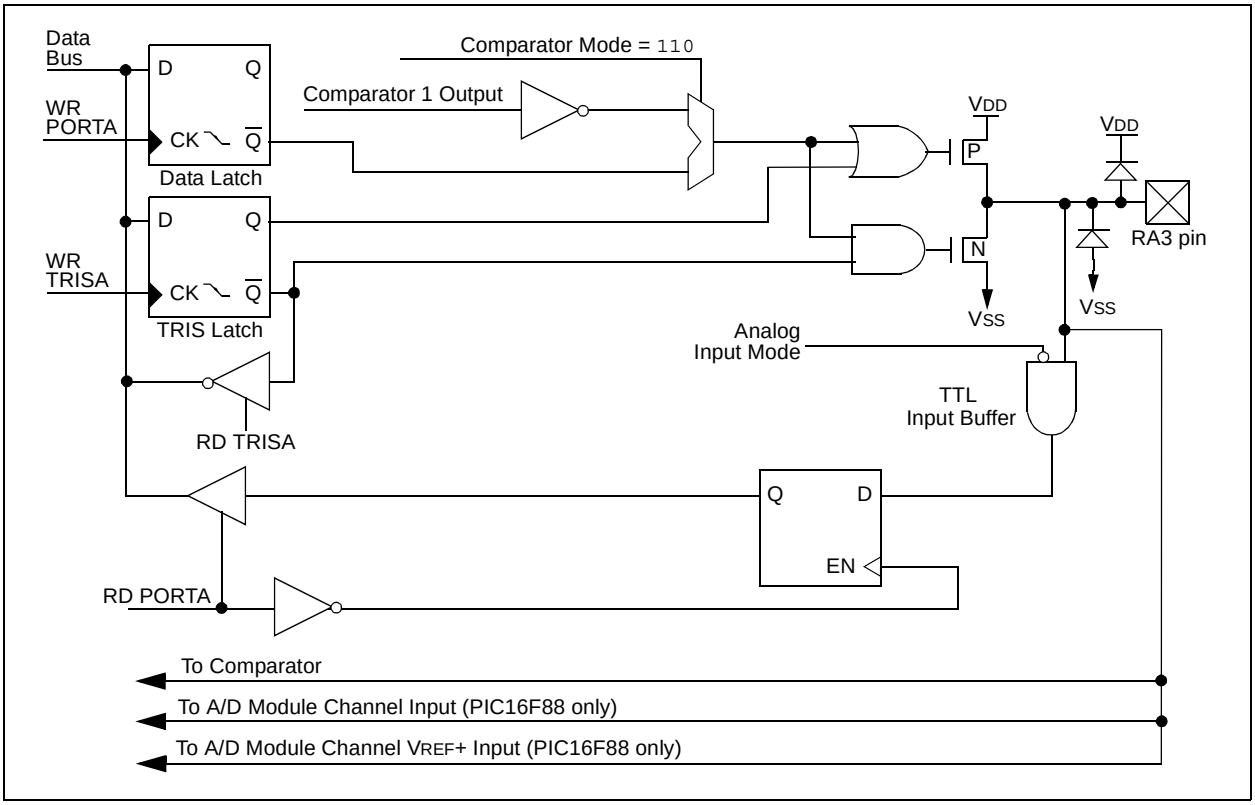


FIGURE 5-3: BLOCK DIAGRAM OF RA2/AN2/CVREF/VREF- PIN

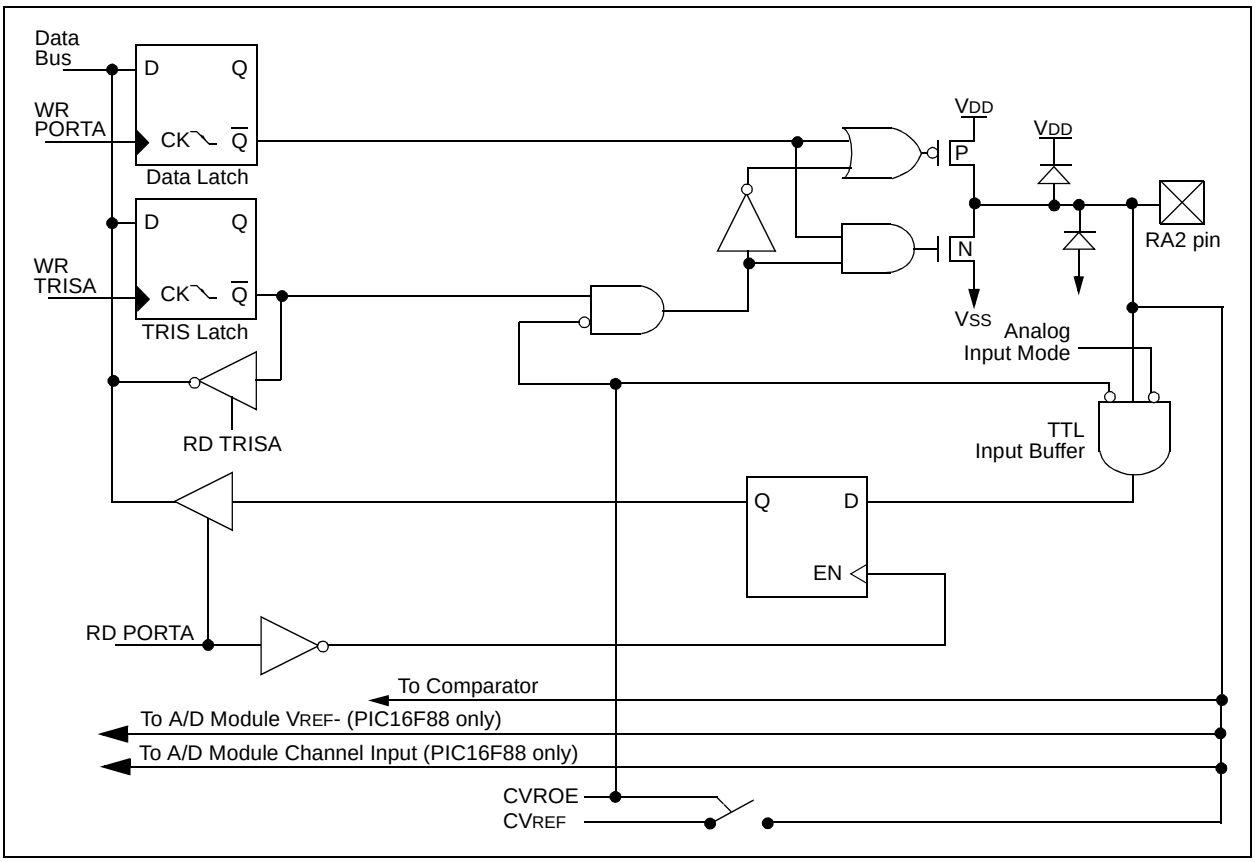


FIGURE 5-4: BLOCK DIAGRAM OF RA4/AN4/T0CKI/C2OUT PIN

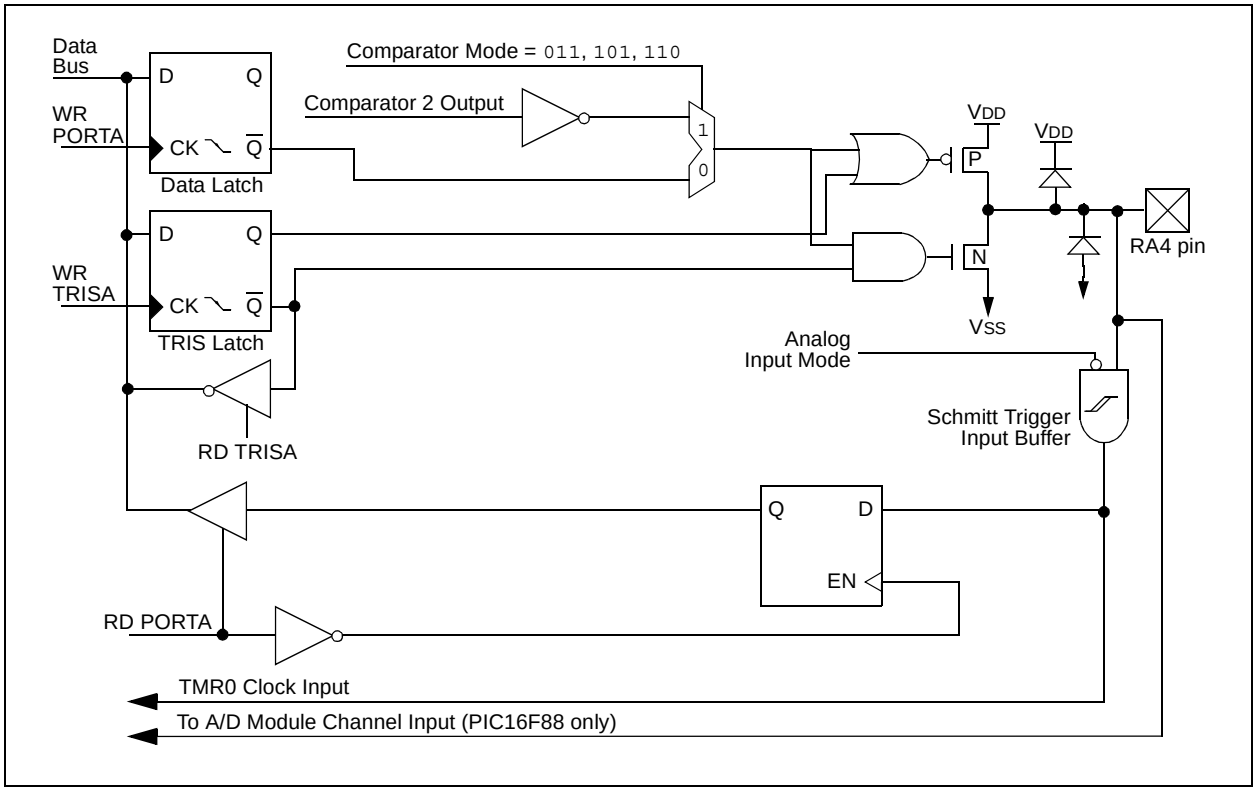


FIGURE 5-5: BLOCK DIAGRAM OF RA5/$\overline{\text{MCLR}}$/VPP PIN

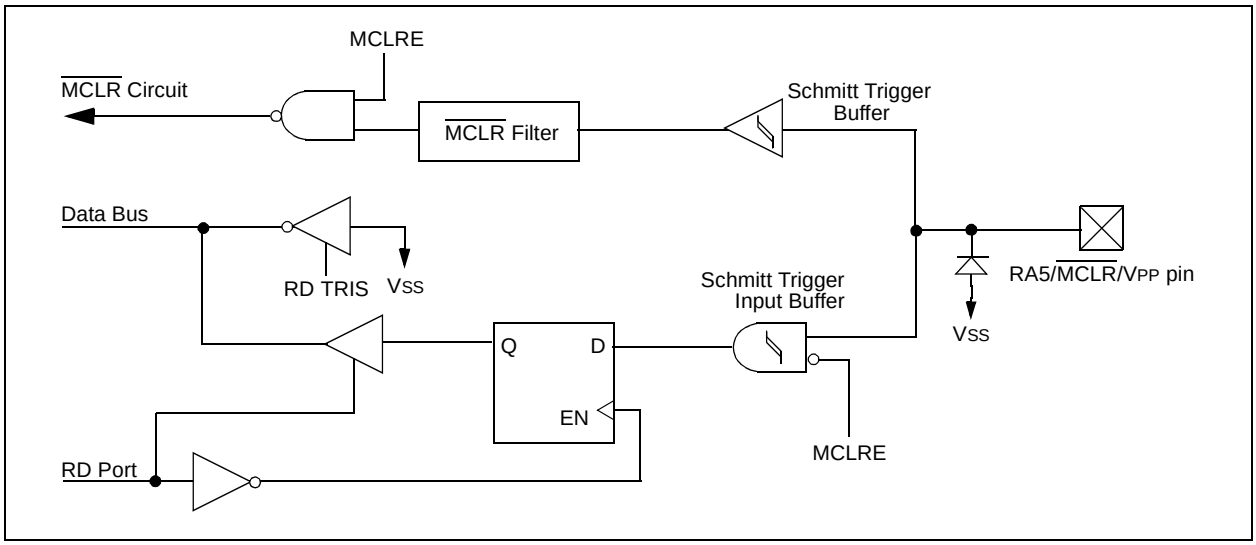
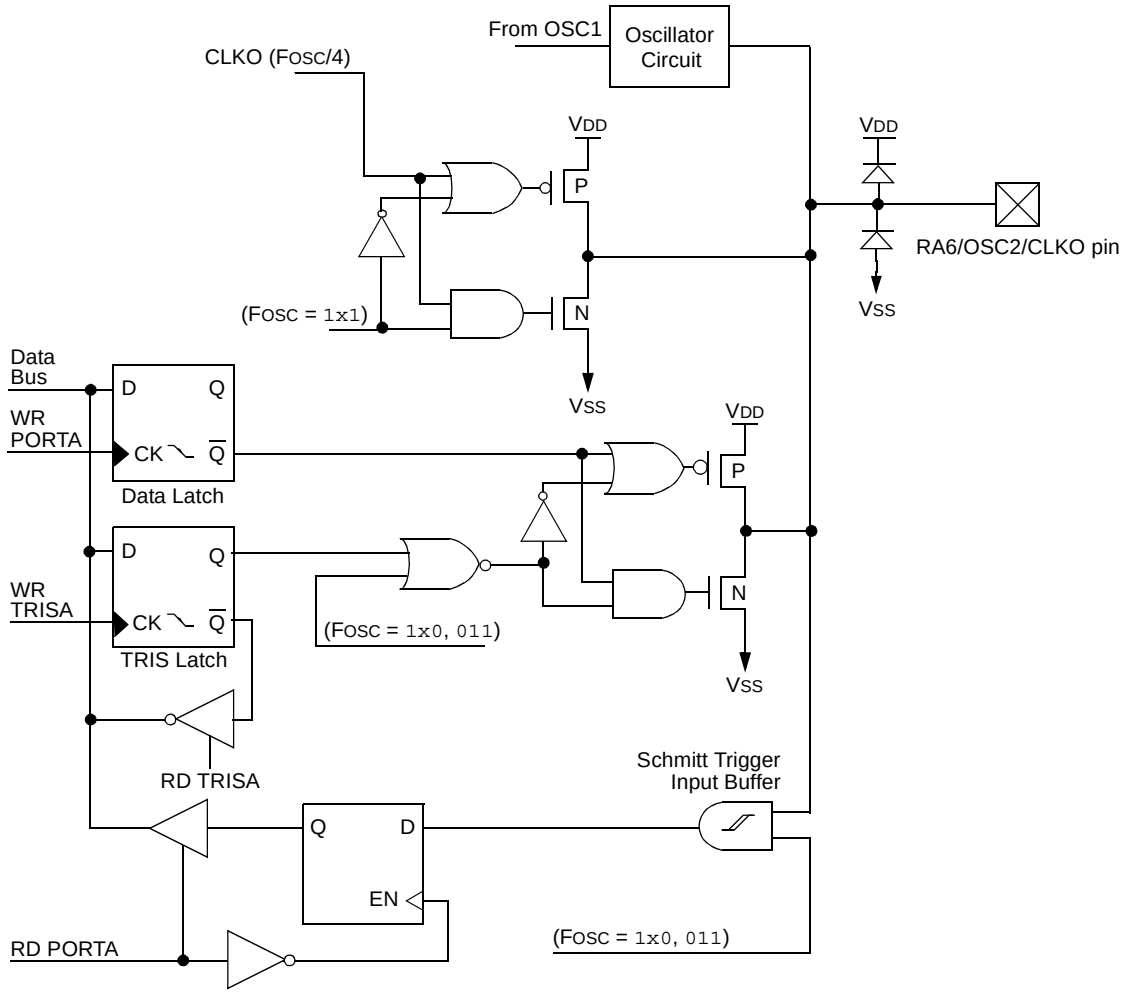
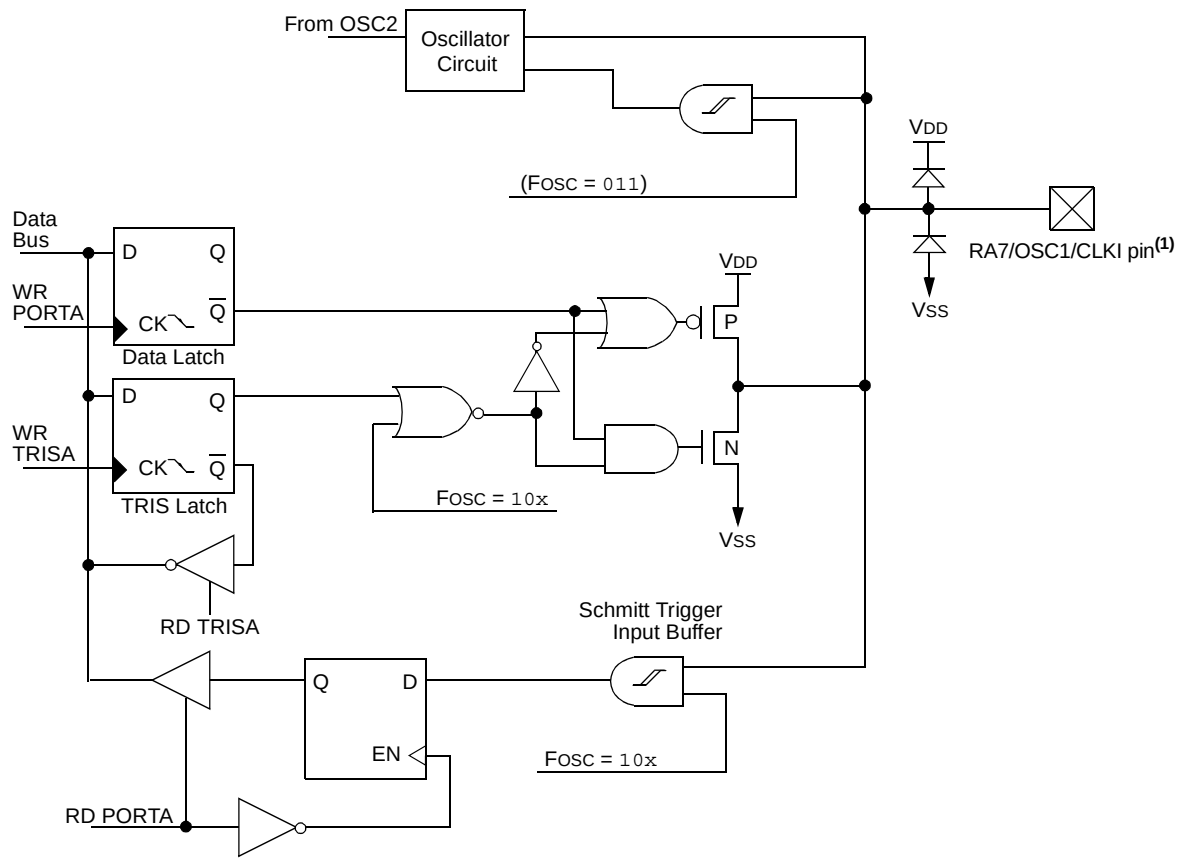


FIGURE 5-6: BLOCK DIAGRAM OF RA6/OSC2/CLKO PIN



Note 1: I/O pins have protection diodes to VDD and VSS.
2: CLKO signal is 1/4 of the FOSC frequency.

FIGURE 5-7: BLOCK DIAGRAM OF RA7/OSC1/CLKI PIN



Note 1: I/O pins have protection diodes to VDD and VSS.

5.2 PORTB and the TRISB Register

PORTB is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISB. Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., put the contents of the output latch on the selected pin).

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit $\overline{\text{RBPU}}$ (OPTION_REG<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

Four of PORTB's pins, RB7:RB4, have an interrupt-on-change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB7:RB4 pin configured as an output is excluded from the interrupt-on-change comparison). The input pins (of RB7:RB4) are compared with the old value latched on the last read of PORTB. The "mismatch" outputs of RB7:RB4 are OR'ed together to generate the RB Port Change Interrupt with Flag bit RBIF (INTCON<0>).

This interrupt can wake the device from Sleep. The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of PORTB. This will end the mismatch condition.
b) Clear flag bit RBIF.

A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition and allow flag bit RBIF to be cleared.

The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

RB0/INT is an external interrupt input pin and is configured using the INTEDG bit (OPTION_REG<6>).

PORTB is multiplexed with several peripheral functions (see Table 5-3). PORTB pins have Schmitt Trigger input buffers.

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTB pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. Since the TRIS bit override is in effect while the peripheral is enabled, read-modify-write instructions (`BSF, BCF, XORWF`) with TRISB as the destination should be avoided. The user should refer to the corresponding peripheral section for the correct TRIS bit settings.

TABLE 5-3: PORTB FUNCTIONS

Name	Bit#	Buffer	Function
RB0/INT/CCP1(7)	bit 0	TTL/ST(1)	Input/output pin or external interrupt input. Capture input/Compare output/PWM output pin. Internal software programmable weak pull-up.
RB1/SDI/SDA	bit 1	TTL/ST(5)	Input/output pin, SPI™ data input pin or I^2C™ data I/O pin. Internal software programmable weak pull-up.
RB2/SDO/RX/DT	bit 2	TTL/ST(4)	Input/output pin, SPI data output pin. AUSART asynchronous receive or synchronous data. Internal software programmable weak pull-up.
RB3/PGM/CCP1(3,7)	bit 3	TTL/ST(2)	Input/output pin, programming in LVP mode or Capture input/Compare output/PWM output pin. Internal software programmable weak pull-up.
RB4/SCK/SCL	bit 4	TTL/ST(5)	Input/output pin or SPI and I^2C clock pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB5/$\overline{SS}$/TX/CK	bit 5	TTL	Input/output pin or SPI slave select pin (with interrupt-on-change). AUSART asynchronous transmit or synchronous clock. Internal software programmable weak pull-up.
RB6/AN5(6)/PGC/ T1OSO/T1CKI	bit 6	TTL/ST(2)	Input/output pin, analog input(6), serial programming clock (with interrupt-on-change), Timer1 oscillator output pin or Timer1 clock input pin. Internal software programmable weak pull-up.
RB7/AN6(6)/PGD/ T1OSI	bit 7	TTL/ST(2)	Input/output pin, analog input(6), serial programming data (with interrupt-on-change) or Timer1 oscillator input pin. Internal software programmable weak pull-up.

Legend: TTL = TTL input, ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.

2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.

3: Low-Voltage ICSP™ Programming (LVP) is enabled by default, which disables the RB3 I/O function. LVP must be disabled to enable RB3 as an I/O pin and allow maximum compatibility to the other 18-pin mid-range devices.

4: This buffer is a Schmitt Trigger input when configured for CCP or SSP mode.

5: This buffer is a Schmitt Trigger input when configured for SPI or I^2C mode.

6: PIC16F88 only.

7: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

TABLE 5-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

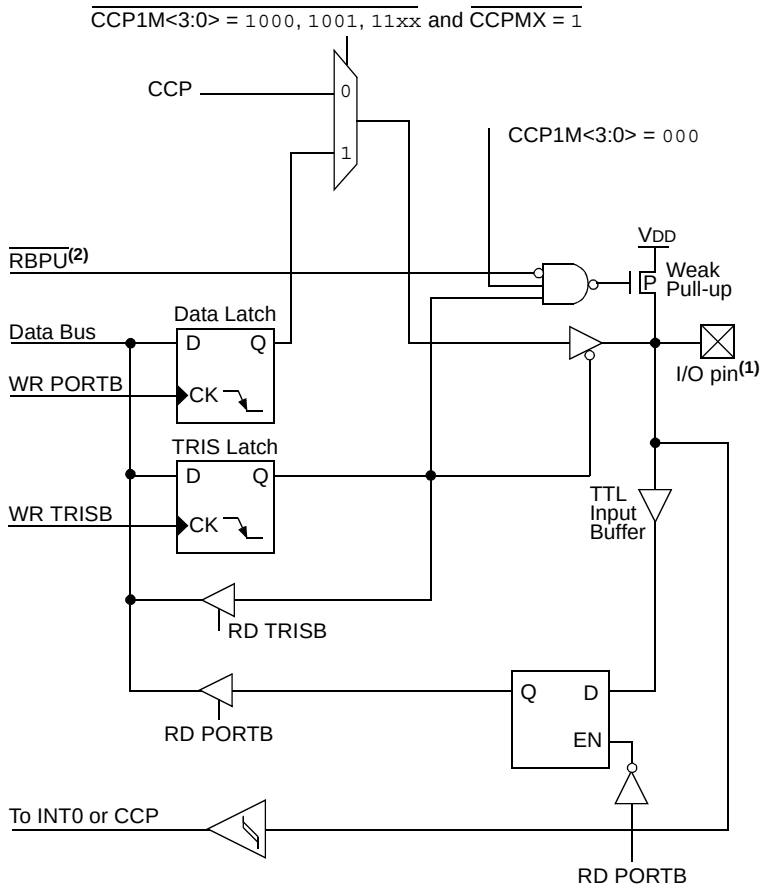
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
06h, 106h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx(1) 00xx xxxx(2)	uuuu uuuu(1) 00uu uuuu(2)
86h, 186h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
81h, 181h	OPTION_REG	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
9Bh	ANSEL(2)	—	ANS6	ANS5	ANS4	ANS3	ANS2	ANS1	ANS0	-111 1111	-111 1111

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by PORTB.

Note 1: This value applies only to the PIC16F87.

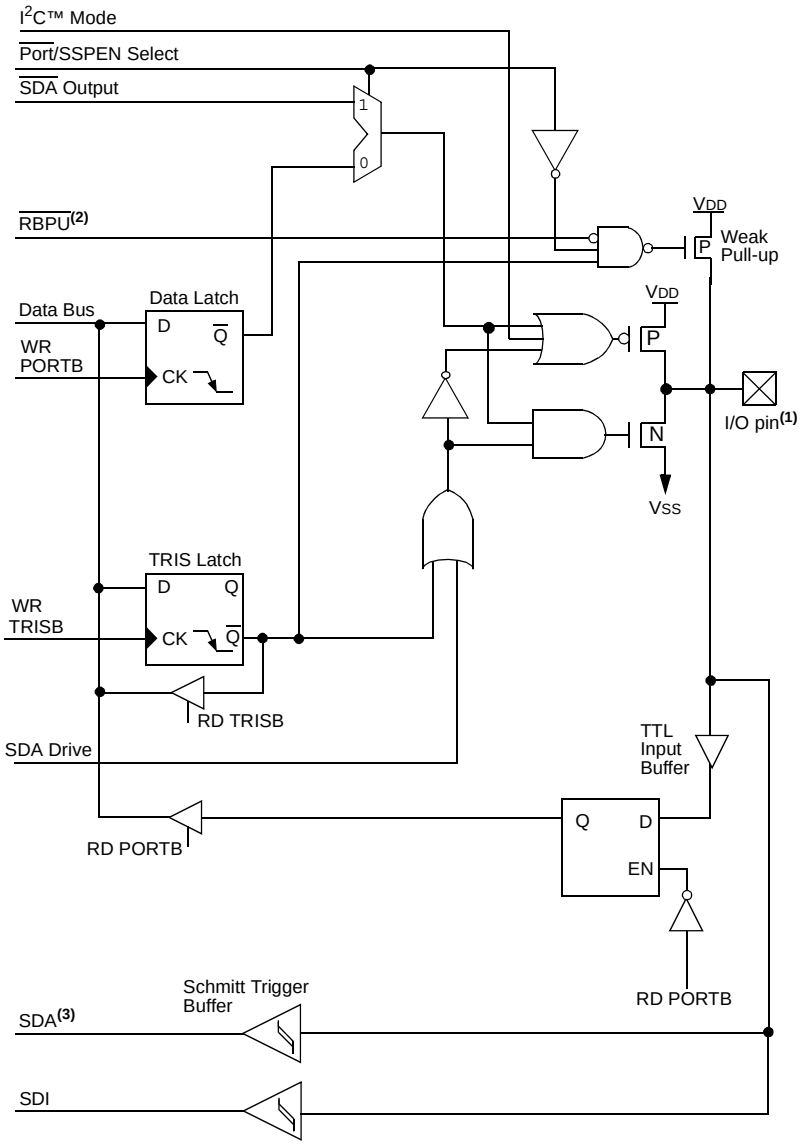
2: This value applies only to the PIC16F88.

FIGURE 5-8: BLOCK DIAGRAM OF RB0/INT/CCP1(3) PIN



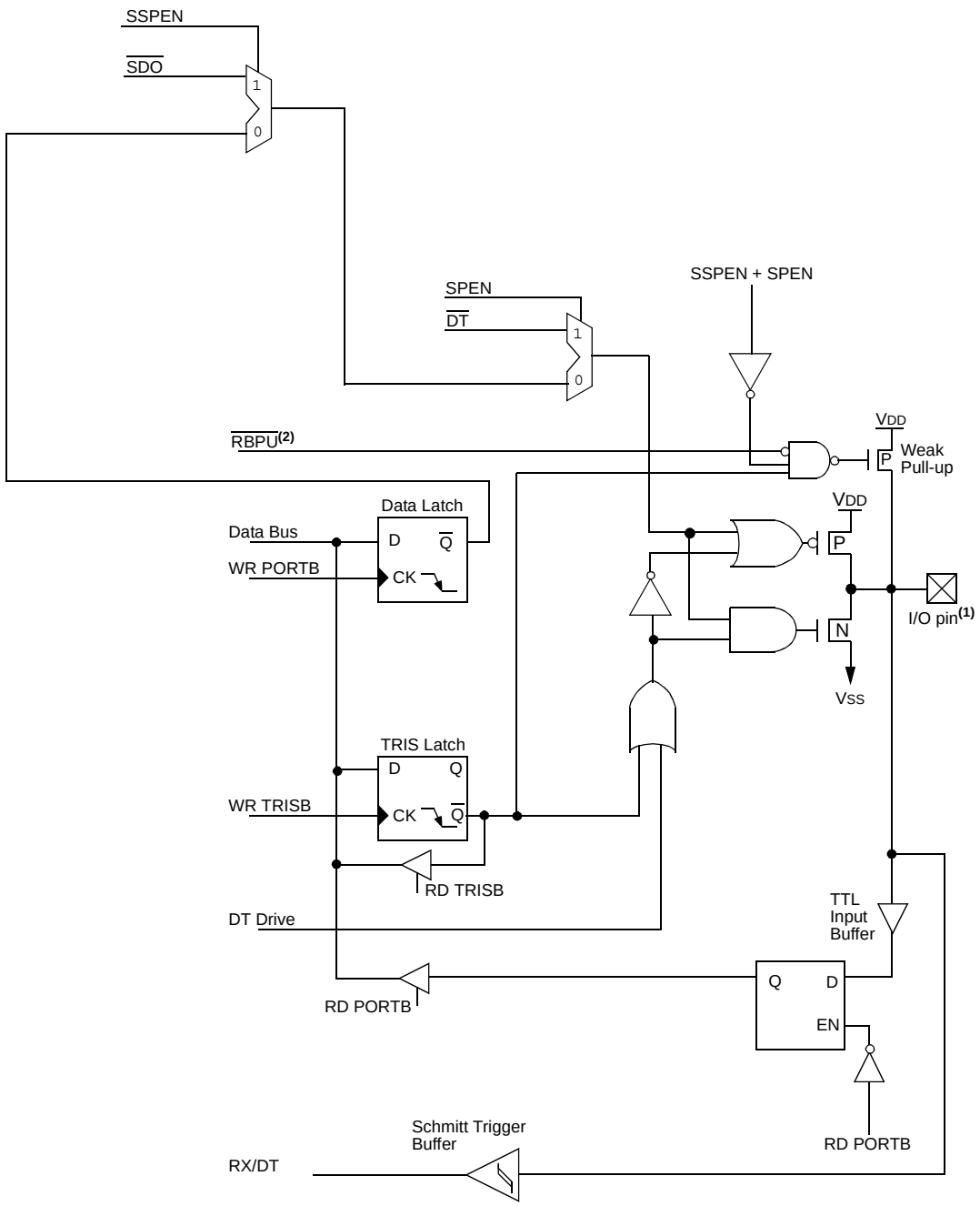
Note 1: I/O pins have diode protection to VDD and VSS.
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit.
3: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

FIGURE 5-9: BLOCK DIAGRAM OF RB1/SDI/SDA PIN



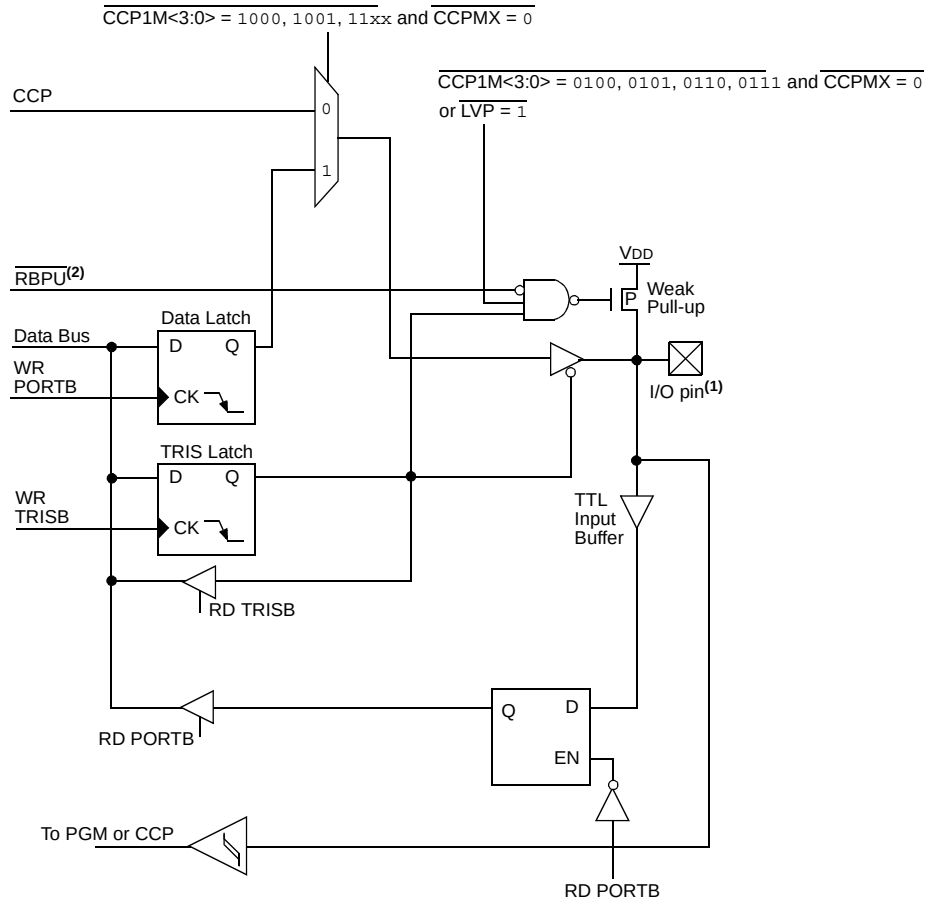
Note 1: I/O pins have diode protection to VDD and VSS.
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit.
3: The SDA Schmitt conforms to the I2C specification.

FIGURE 5-10: BLOCK DIAGRAM OF RB2/SDO/RX/DT PIN



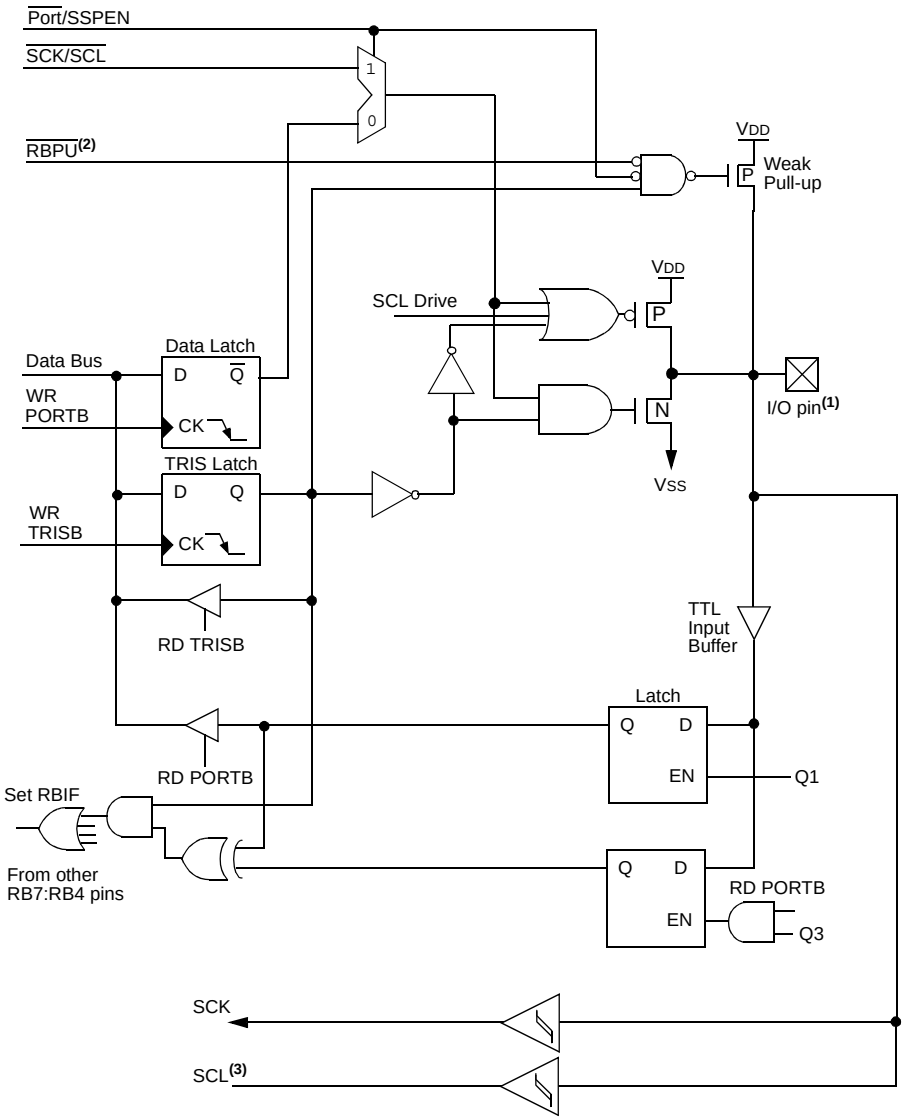
Note 1: I/O pins have diode protection to VDD and VSS.
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit.

FIGURE 5-11: BLOCK DIAGRAM OF RB3/PGM/CCP1(3) PIN



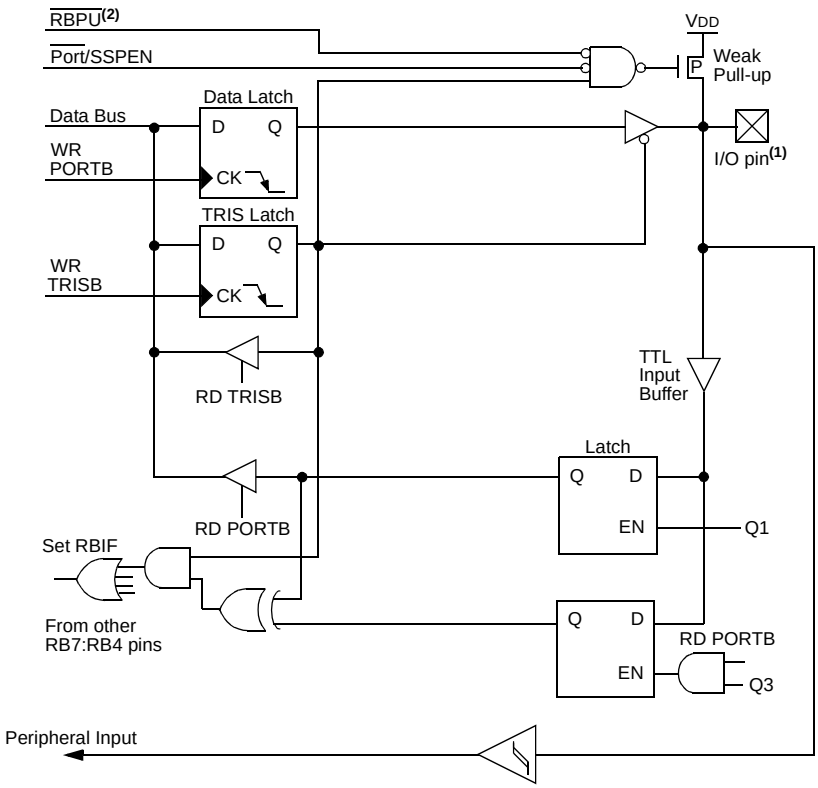
Note 1: I/O pins have diode protection to VDD and VSS.
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit.
3: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

FIGURE 5-12: BLOCK DIAGRAM OF RB4/SCK/SCL PIN



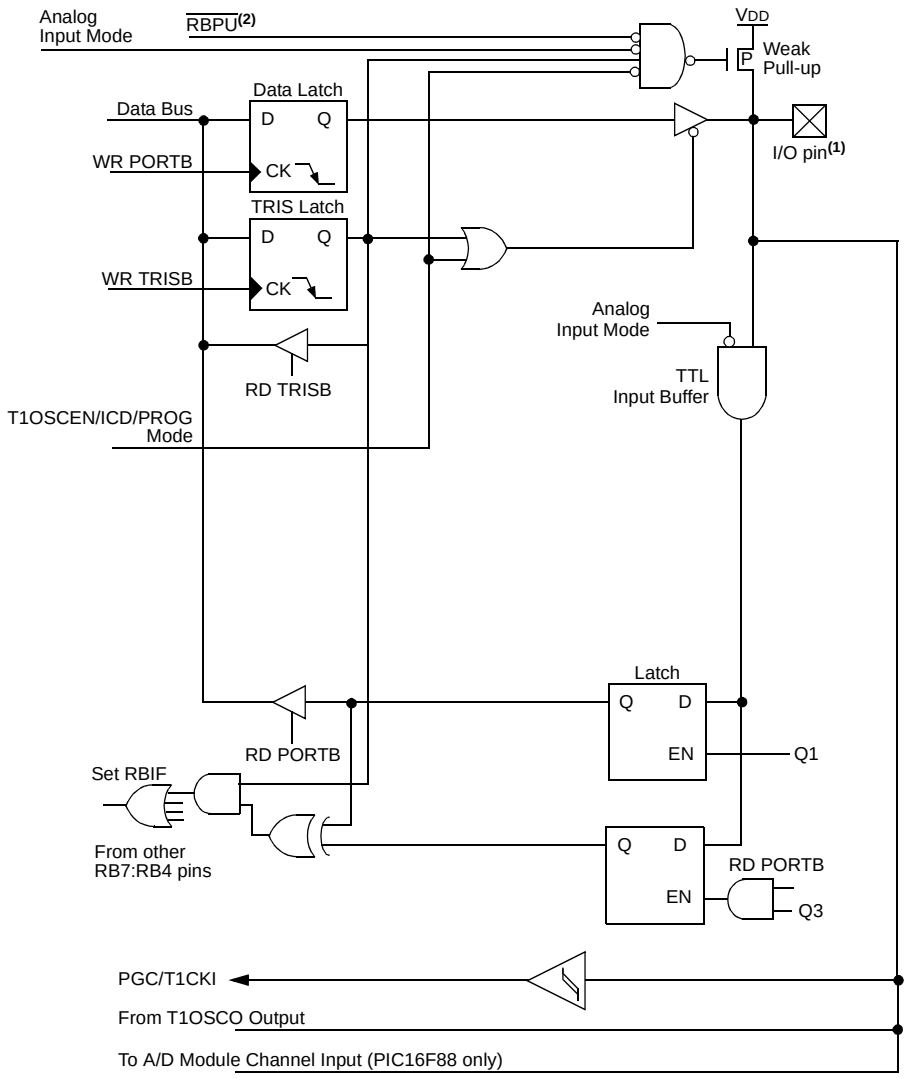
Note 1: I/O pins have diode protection to VDD and VSS.
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit.
3: The SCL Schmitt conforms to the I^2C™ specification.

FIGURE 5-13: BLOCK DIAGRAM OF RB5/$\overline{SS}$/TX/CK PIN



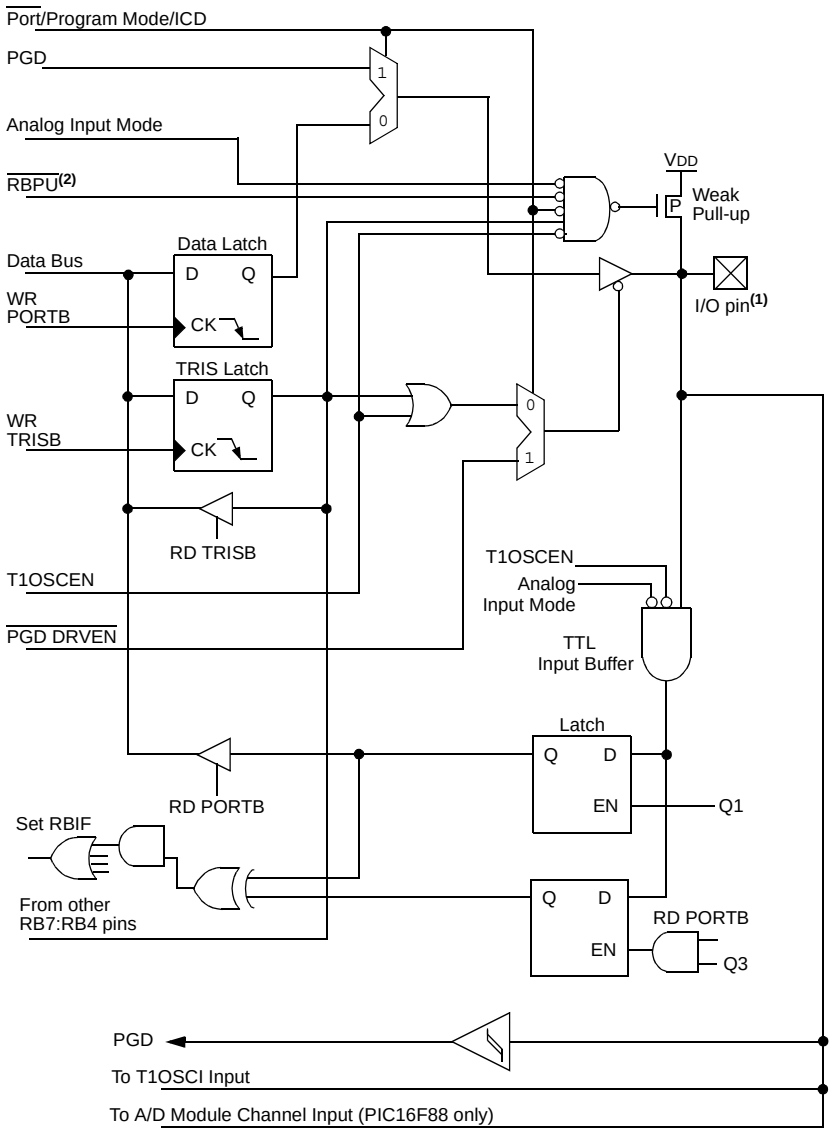
Note 1: I/O pins have diode protection to VDD and VSS.
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{RBPU}$ bit.

FIGURE 5-14: BLOCK DIAGRAM OF RB6/AN5(3)/PGC/T1OSO/T1CKI PIN



Note 1: I/O pins have diode protection to VDD and VSS.
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit.
3: PIC16F88 devices only.

FIGURE 5-15: BLOCK DIAGRAM OF RB7/AN6(3)/PGD/T1OSI PIN



Note 1: I/O pins have diode protection to VDD and VSS.
2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit.
3: PIC16F88 devices only.

6.0 TIMER0 MODULE

The Timer0 module timer/counter has the following features:

- 8-bit timer/counter
- Readable and writable
- 8-bit software programmable prescaler
- Internal or external clock select
- Interrupt-on-overflow from FFh to 00h
- Edge select for external clock

Additional information on the Timer0 module is available in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023).

Figure 6-1 is a block diagram of the Timer0 module and the prescaler shared with the WDT.

6.1 Timer0 Operation

Timer0 operation is controlled through the OPTION_REG register (see Register 2-2). Timer mode is selected by clearing bit T0CS (OPTION_REG<5>). In Timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

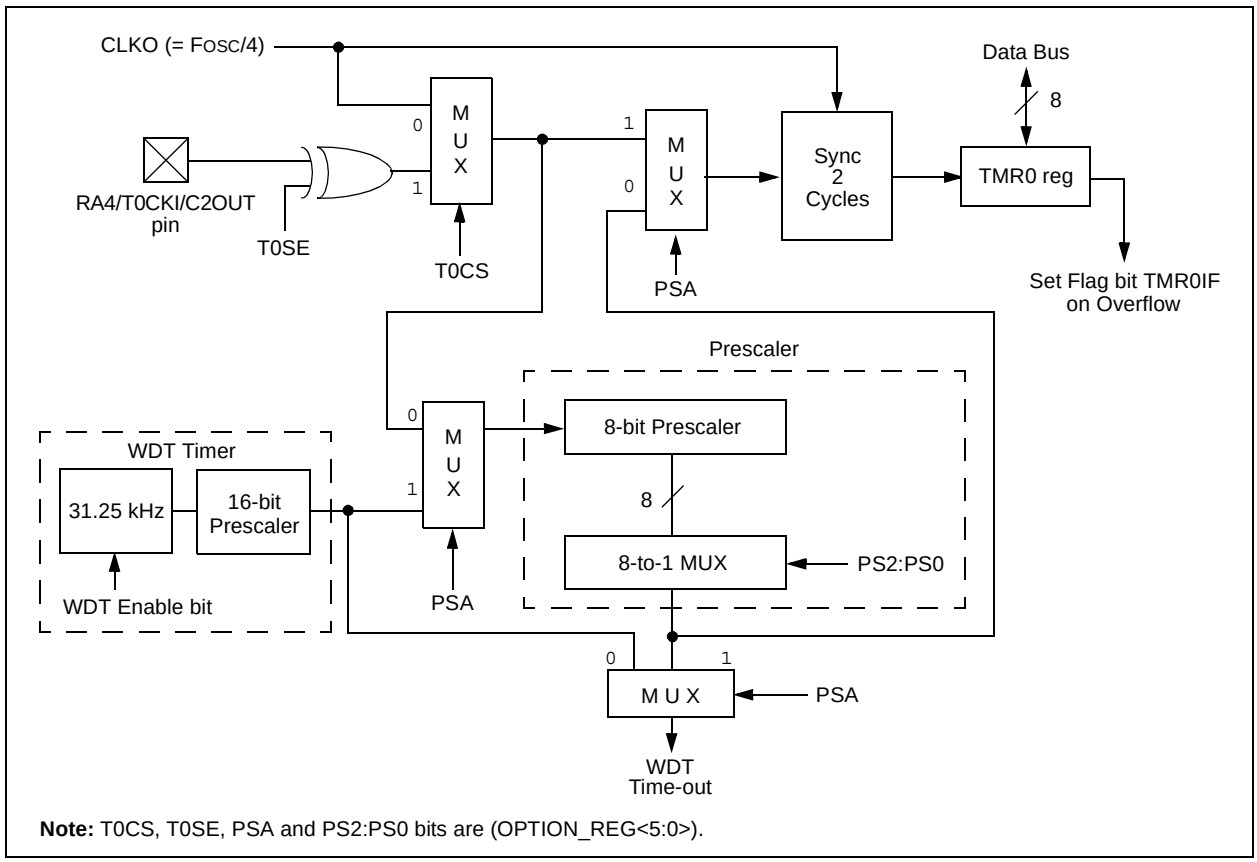
Counter mode is selected by setting bit T0CS (OPTION_REG<5>). In Counter mode, Timer0 will increment, either on every rising or falling edge of pin RA4/T0CKI/C2OUT. The incrementing edge is determined by the Timer0 Source Edge Select bit, T0SE (OPTION_REG<4>). Clearing bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in **Section 6.3 "Using Timer0 with an External Clock"**.

The prescaler is mutually, exclusively shared between the Timer0 module and the Watchdog Timer. The prescaler is not readable or writable. **Section 6.4 "Prescaler"** details the operation of the prescaler.

6.2 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h. This overflow sets bit TMR0IF (INTCON<2>). The interrupt can be masked by clearing bit TMR0IE (INTCON<5>). Bit TMR0IF must be cleared in software by the Timer0 module Interrupt Service Routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from Sleep, since the timer is shut off during Sleep.

FIGURE 6-1: BLOCK DIAGRAM OF THE TIMER0/WDT PRESCALER



Note: T0CS, T0SE, PSA and PS2:PS0 bits are (OPTION_REG<5:0>).

6.3 Using Timer0 with an External Clock

When no prescaler is used, the external clock input is the same as the prescaler output. The synchronization of T0CKI, with the internal phase clocks, is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks. Therefore, it is necessary for T0CKI to be high for at least 2 TOSC (and a small RC delay of 20 ns) and low for at least 2 TOSC (and a small RC delay of 20 ns). Refer to the electrical specification of the desired device.

6.4 Prescaler

There is only one prescaler available, which is mutually exclusively shared between the Timer0 module and the Watchdog Timer. A prescaler assignment for the Timer0 module means that the prescaler cannot be used by the Watchdog Timer and vice versa. This prescaler is not readable or writable (see Figure 6-1).

> **Note:** Although the prescaler can be assigned to either the WDT or Timer0, but not both, a new divide counter is implemented in the WDT circuit to give multiple WDT time-out selections. This allows TMR0 and WDT to each have their own scaler. Refer to **Section 15.12 "Watchdog Timer (WDT)"** for further details.

The PSA and PS2:PS0 bits (OPTION_REG<3:0>) determine the prescaler assignment and prescale ratio.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF 1`, `MOVWF 1`, `BSF 1,x`....etc.) will clear the prescaler. When assigned to WDT, a `CLRWDT` instruction will clear the prescaler along with the Watchdog Timer. The prescaler is not readable or writable.

> **Note:** Writing to TMR0, when the prescaler is assigned to Timer0, will clear the prescaler count but will not change the prescaler assignment.

REGISTER 6-1: OPTION_REG: OPTION CONTROL REGISTER (ADDRESS 81h, 181h)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$:** PORTB Pull-up Enable bit

bit 6 **INTEDG:** Interrupt Edge Select bit

bit 5 **T0CS**: TMR0 Clock Source Select bit
`1` = Transition on T0CKI pin
`0` = Internal instruction cycle clock (CLKO)

bit 4 **T0SE**: TMR0 Source Edge Select bit
`1` = Increment on high-to-low transition on T0CKI pin
`0` = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Prescaler Assignment bit
`1` = Prescaler is assigned to the WDT
`0` = Prescaler is assigned to the Timer0 module

bit 2-0 **PS<2:0>:** Prescaler Rate Select bits

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

> **Note:** To avoid an unintended device Reset, the instruction sequence shown in the *"PICmicro® Mid-Range MCU Family Reference Manual"* (DS33023) must be executed when changing the prescaler assignment from Timer0 to the WDT. This sequence must be followed even if the WDT is disabled.

EXAMPLE 6-1: CHANGING THE PRESCALER ASSIGNMENT FROM WDT TO TIMER0

```
CLRWDT                  ; Clear WDT and prescaler
BANKSEL   OPTION_REG    ; Select Bank of OPTION_REG
MOVLW     b'xxxx0xxx'   ; Select TMR0, new prescale
MOVWF     OPTION_REG    ; value and clock source
```

TABLE 6-1: REGISTERS ASSOCIATED WITH TIMER0

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
01h,101h	TMR0	Timer0 Module Register								xxxx xxxx	uuuu uuuu
0Bh,8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
81h,181h	OPTION_REG	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged. Shaded cells are not used by Timer0.

NOTES:

7.0 TIMER1 MODULE

The Timer1 module is a 16-bit timer/counter consisting of two 8-bit registers (TMR1H and TMR1L) which are readable and writable. The TMR1 register pair (TMR1H:TMR1L) increments from 0000h to FFFFh and rolls over to 0000h. The TMR1 interrupt, if enabled, is generated on overflow which is latched in interrupt flag bit, TMR1IF (PIR1<0>). This interrupt can be enabled/disabled by setting/clearing TMR1 interrupt enable bit, TMR1IE (PIE1<0>).

The Timer1 oscillator can be used as a secondary clock source in low-power modes. When the T1RUN bit is set along with SCS<1:0> = `01`, the Timer1 oscillator is providing the system clock. If the Fail-Safe Clock Monitor is enabled and the Timer1 oscillator fails while providing the system clock, polling the T1RUN bit will indicate whether the clock is being provided by the Timer1 oscillator or another source.

Timer1 can also be used to provide Real-Time Clock (RTC) functionality to applications with only a minimal addition of external components and code overhead.

7.1 Timer1 Operation

Timer1 can operate in one of three modes:

- as a Timer
- as a Synchronous Counter
- as an Asynchronous Counter

The operating mode is determined by the clock select bit, TMR1CS (T1CON<1>).

In Timer mode, Timer1 increments every instruction cycle. In Counter mode, it increments on every rising edge of the external clock input.

Timer1 can be enabled/disabled by setting/clearing control bit, TMR1ON (T1CON<0>).

Timer1 also has an internal “Reset input”. This Reset can be generated by the CCP1 module as the special event trigger (see **Section 9.1 “Capture Mode”**). Register 7-1 shows the Timer1 Control register.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RB6/PGC/T1OSO/T1CKI and RB7/PGD/T1OSI pins become inputs. That is, the TRISB<7:6> value is ignored and these pins read as ‘`0`’.

Additional information on timer modules is available in the “*PICmicro® Mid-Range MCU Family Reference Manual*” (DS33023).

REGISTER 7-1: T1CON: TIMER1 CONTROL REGISTER (ADDRESS 10h)

U-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6 **T1RUN**: Timer1 System Clock Status bit
1 = System clock is derived from Timer1 oscillator
0 = System clock is derived from another source

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable Control bit
1 = Oscillator is enabled
0 = Oscillator is shut off (the oscillator inverter is turned off to eliminate power drain)

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Control bit
TMR1CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
TMR1CS = 0:
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS:** Timer1 Clock Source Select bit
1 = External clock from pin RB6/AN5[(1)]/PGC/T1OSO/T1CKI (on the rising edge)
0 = Internal clock (FOSC/4)

Note 1: Available on PIC16F88 devices only.

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

7.2 Timer1 Operation in Timer Mode

Timer mode is selected by clearing the TMR1CS (T1CON<1>) bit. In this mode, the input clock to the timer is FOSC/4. The synchronize control bit, $\overline{\text{T1SYNC}}$ (T1CON<2>), has no effect since the internal clock is always in sync.

7.3 Timer1 Counter Operation

Timer1 may operate in Asynchronous or Synchronous mode, depending on the setting of the TMR1CS bit.

When Timer1 is being incremented via an external source, increments occur on a rising edge. After Timer1 is enabled in Counter mode, the module must first have a falling edge before the counter begins to increment.

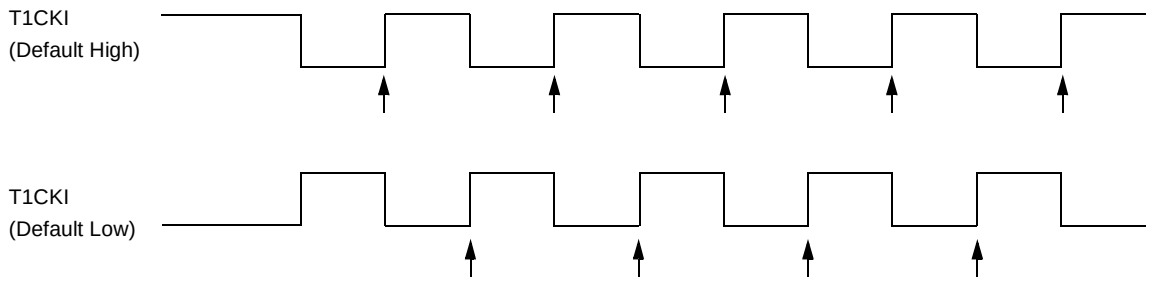
7.4 Timer1 Operation in Synchronized Counter Mode

Counter mode is selected by setting bit TMR1CS. In this mode, the timer increments on every rising edge of clock input on pin RB7/PGD/T1OSI when bit T1OSCEN is set, or on pin RB6/PGC/T1OSO/T1CKI when bit T1OSCEN is cleared.

If $\overline{\text{T1SYNC}}$ is cleared, then the external clock input is synchronized with internal phase clocks. The synchronization is done after the prescaler stage. The prescaler stage is an asynchronous ripple counter.

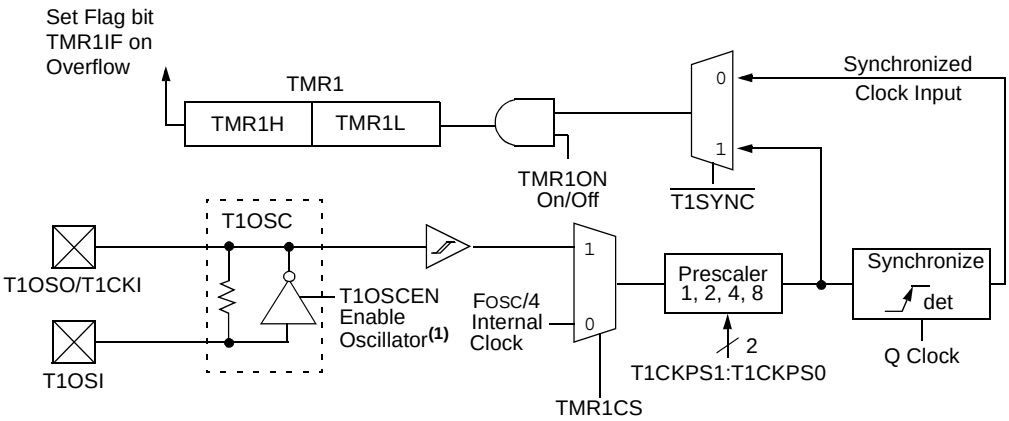
In this configuration, during Sleep mode, Timer1 will not increment even if the external clock is present since the synchronization circuit is shut off. The prescaler, however, will continue to increment.

FIGURE 7-1: TIMER1 INCREMENTING EDGE



Note: Arrows indicate counter increments.

FIGURE 7-2: TIMER1 BLOCK DIAGRAM



Note 1: When the T1OSCEN bit is cleared, the inverter is turned off. This eliminates power drain.

7.5 Timer1 Operation in Asynchronous Counter Mode

If control bit $\overline{\text{T1SYNC}}$ (T1CON<2>) is set, the external clock input is not synchronized. The timer continues to increment asynchronous to the internal phase clocks. The timer will continue to run during Sleep and can generate an interrupt-on-overflow that will wake-up the processor. However, special precautions in software are needed to read/write the timer (see **Section 7.5.1 "Reading and Writing Timer1 in Asynchronous Counter Mode"**).

In Asynchronous Counter mode, Timer1 cannot be used as a time base for capture or compare operations.

7.5.1 READING AND WRITING TIMER1 IN ASYNCHRONOUS COUNTER MODE

Reading TMR1H or TMR1L while the timer is running from an external asynchronous clock will ensure a valid read (taken care of in hardware). However, the user should keep in mind that reading the 16-bit timer in two 8-bit values itself, poses certain problems, since the timer may overflow between the reads.

For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the timer registers while the register is incrementing. This may produce an unpredictable value in the timer register.

Reading the 16-bit value requires some care. The example codes provided in Example 7-1 and Example 7-2 demonstrate how to write to and read Timer1 while it is running in Asynchronous mode.

EXAMPLE 7-1: WRITING A 16-BIT FREE RUNNING TIMER

```
; All interrupts are disabled
CLRF     TMR1L       ; Clear Low byte, Ensures no rollover into TMR1H
MOVLW    HI_BYTE     ; Value to load into TMR1H
MOVWF    TMR1H, F    ; Write High byte
MOVLW    LO_BYTE     ; Value to load into TMR1L
MOVWF    TMR1H, F    ; Write Low byte
; Re-enable the Interrupt (if required)
CONTINUE             ; Continue with your code
```

EXAMPLE 7-2: READING A 16-BIT FREE RUNNING TIMER

```
; All interrupts are disabled
MOVF      TMR1H, W     ; Read high byte
MOVWF     TMPH
MOVF      TMR1L, W     ; Read low byte
MOVWF     TMPL
MOVF      TMR1H, W     ; Read high byte
SUBWF     TMPH, W      ; Sub 1st read with 2nd read
BTFSC     STATUS, Z    ; Is result = 0
GOTO      CONTINUE     ; Good 16-bit read
; TMR1L may have rolled over between the read of the high and low bytes.
; Reading the high and low bytes now will read a good value.
MOVF      TMR1H, W     ; Read high byte
MOVWF     TMPH
MOVF      TMR1L, W     ; Read low byte
MOVWF     TMPL         ; Re-enable the Interrupt (if required)
CONTINUE               ; Continue with your code
```

7.6 Timer1 Oscillator

A crystal oscillator circuit is built between pins T1OSI (input) and T1OSO (amplifier output). It is enabled by setting control bit T1OSCEN (T1CON<3>). The oscillator is a low-power oscillator, rated up to 32.768 kHz. It will continue to run during all power-managed modes. It is primarily intended for a 32 kHz crystal. The circuit for a typical LP oscillator is shown in Figure 7-3. Table 7-1 shows the capacitor selection for the Timer1 oscillator.

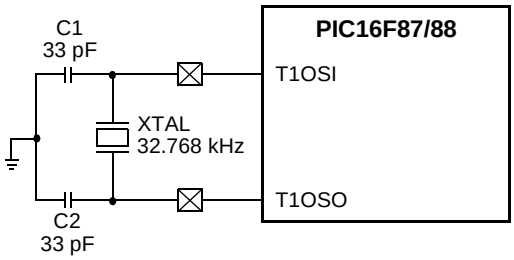
The user must provide a software time delay to ensure proper oscillator start-up.

Note: The Timer1 oscillator shares the T1OSI and T1OSO pins with the PGD and PGC pins used for programming and debugging.

When using the Timer1 oscillator, In-Circuit Serial Programming™ (ICSP™) may not function correctly (high voltage or low voltage), or the In-Circuit Debugger (ICD) may not communicate with the controller. As a result of using either ICSP or ICD, the Timer1 crystal may be damaged.

If ICSP or ICD operations are required, the crystal should be disconnected from the circuit (disconnect either lead) or installed after programming. The oscillator loading capacitors may remain in-circuit during ICSP or ICD operation.

FIGURE 7-3: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR



Note: See the Notes with Table 7-1 for additional information about capacitor selection.

TABLE 7-1: CAPACITOR SELECTION FOR THE TIMER1 OSCILLATOR

Osc Type	Freq	C1	C2
LP	32 kHz	33 pF	33 pF

Note 1: Microchip suggests this value as a starting point in validating the oscillator circuit.

2: Higher capacitance increases the stability of the oscillator but also increases the start-up time.

3: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

4: Capacitor values are for design guidance only.

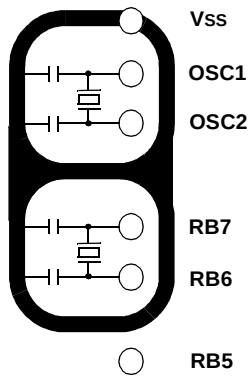
7.7 Timer1 Oscillator Layout Considerations

The Timer1 oscillator circuit draws very little power during operation. Due to the low-power nature of the oscillator, it may also be sensitive to rapidly changing signals in close proximity.

The oscillator circuit, shown in Figure 7-3, should be located as close as possible to the microcontroller. There should be no circuits passing within the oscillator circuit boundaries other than VSS or VDD.

If a high-speed circuit must be located near the oscillator, a grounded guard ring around the oscillator circuit, as shown in Figure 7-4, may be helpful when used on a single-sided PCB or in addition to a ground plane.

FIGURE 7-4: OSCILLATOR CIRCUIT WITH GROUNDED GUARD RING



7.8 Resetting Timer1 Using a CCP Trigger Output

If the CCP1 module is configured in Compare mode to generate a “special event trigger” signal (CCP1M3:CCP1M0 = `1011`), the signal will reset Timer1 and start an A/D conversion (if the A/D module is enabled).

Note:	The special event triggers from the CCP1 module will not set interrupt flag bit, TMR1IF (PIR1<0>).

Timer1 must be configured for either Timer or Synchronized Counter mode to take advantage of this feature. If Timer1 is running in Asynchronous Counter mode, this Reset operation may not work.

In the event that a write to Timer1 coincides with a special event trigger from CCP1, the write will take precedence.

In this mode of operation, the CCPR1H:CCPR1L register pair effectively becomes the period register for Timer1.

7.9 Resetting Timer1 Register Pair (TMR1H, TMR1L)

TMR1H and TMR1L registers are not reset to 00h on a POR, or any other Reset, except by the CCP1 special event triggers.

T1CON register is reset to 00h on a Power-on Reset or a Brown-out Reset, which shuts off the timer and leaves a 1:1 prescale. In all other Resets, the register is unaffected.

7.10 Timer1 Prescaler

The prescaler counter is cleared on writes to the TMR1H or TMR1L registers.

7.11 Using Timer1 as a Real-Time Clock

Adding an external LP oscillator to Timer1 (such as the one described in **Section 7.6 “Timer1 Oscillator”**) gives users the option to include RTC functionality to their applications. This is accomplished with an inexpensive watch crystal to provide an accurate time base and several lines of application code to calculate the time. When operating in Sleep mode and using a battery or supercapacitor as a power source, it can completely eliminate the need for a separate RTC device and battery backup.

The application code routine, `RTCisr`, shown in Example 7-3, demonstrates a simple method to increment a counter at one-second intervals using an Interrupt Service Routine. Incrementing the TMR1 register pair to overflow triggers the interrupt and calls the routine, which increments the seconds counter by one; additional counters for minutes and hours are incremented as the previous counter overflows.

Since the register pair is 16 bits wide, counting up to overflow the register directly from a 32.768 kHz clock would take 2 seconds. To force the overflow at the required one-second intervals, it is necessary to preload it; the simplest method is to set the MSb of TMR1H with a `BSF` instruction. Note that the TMR1L register is never preloaded or altered; doing so may introduce cumulative error over many cycles.

For this method to be accurate, Timer1 must operate in Asynchronous mode and the Timer1 overflow interrupt must be enabled (PIE1<0> = `1`), as shown in the routine, `RTCinit`. The Timer1 oscillator must also be enabled and running at all times.

EXAMPLE 7-3: IMPLEMENTING A REAL-TIME CLOCK USING A TIMER1 INTERRUPT SERVICE

```
RTCinit     BANKSEL     TMR1H
            MOVLW       0x80                ; Preload TMR1 register pair
            MOVWF       TMR1H               ; for 1 second overflow
            CLRF        TMR1L
            MOVLW       b'00001111'         ; Configure for external clock,
            MOVWF       T1CON               ; Asynchronous operation, external oscillator
            CLRF        secs                ; Initialize timekeeping registers
            CLRF        mins
            MOVLW       .12
            MOVWF       hours
            BANKSEL     PIE1
            BSF         PIE1, TMR1IE        ; Enable Timer1 interrupt
            RETURN
RTCisr      BANKSEL     TMR1H
            BSF         TMR1H, 7            ; Preload for 1 sec overflow
            BCF         PIR1, TMR1IF        ; Clear interrupt flag
            INCF        secs, F             ; Increment seconds
            MOVF        secs, w
            SUBLW       .60
            BTFSS       STATUS, Z           ; 60 seconds elapsed?
            RETURN                          ; No, done
            CLRF        seconds             ; Clear seconds
            INCF        mins, f             ; Increment minutes
            MOVF        mins, w
            SUBLW       .60
            BTFSS       STATUS, Z           ; 60 seconds elapsed?
            RETURN                          ; No, done
            CLRF        mins                ; Clear minutes
            INCF        hours, f            ; Increment hours
            MOVF        hours, w
            SUBLW       .24
            BTFSS       STATUS, Z           ; 24 hours elapsed?
            RETURN                          ; No, done
            CLRF        hours               ; Clear hours
            RETURN                          ; Done
```

TABLE 7-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
10h	T1CON	—	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	-000 0000	-uuu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the Timer1 module.

Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

NOTES:

8.0 TIMER2 MODULE

Timer2 is an 8-bit timer with a prescaler and a postscaler. It can be used as the PWM time base for the PWM mode of the CCP1 module. The TMR2 register is readable and writable and is cleared on any device Reset.

The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits T2CKPS1:T2CKPS0 (T2CON<1:0>).

The Timer2 module has an 8-bit period register, PR2. Timer2 increments from 00h until it matches PR2 and then resets to 00h on the next increment cycle. PR2 is a readable and writable register. The PR2 register is initialized to FFh upon Reset.

The match output of TMR2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR2 interrupt (latched in flag bit TMR2IF (PIR1<1>)).

Timer2 can be shut off by clearing control bit TMR2ON (T2CON<2>) to minimize power consumption.

Register 8-1 shows the Timer2 Control register.

Additional information on timer modules is available in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023).

8.1 Timer2 Prescaler and Postscaler

The prescaler and postscaler counters are cleared when any of the following occurs:

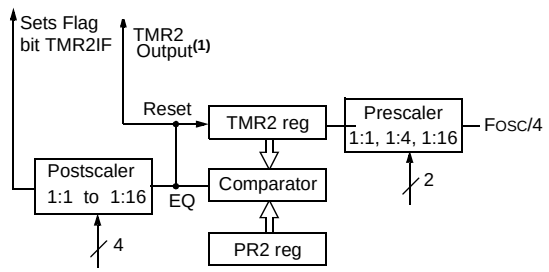
- A write to the TMR2 register
- A write to the T2CON register
- Any device Reset (Power-on Reset, $\overline{\text{MCLR}}$, WDT Reset or Brown-out Reset)

TMR2 is not cleared when T2CON is written.

8.2 Output of TMR2

The output of TMR2 (before the postscaler) is fed to the Synchronous Serial Port module (SSP) which optionally uses it to generate a shift clock.

FIGURE 8-1: TIMER2 BLOCK DIAGRAM



Note 1: TMR2 register output can be software selected by the SSP module as a baud clock.

REGISTER 8-1: T2CON: TIMER2 CONTROL REGISTER (ADDRESS 12h)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6-3 **TOUTPS<3:0>:** Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
0010 = 1:3 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON:** Timer2 On bit
1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

TABLE 8-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
11h	TMR2	Timer2 Module Register								0000 0000	0000 0000
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
92h	PR2	Timer2 Period Register								1111 1111	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the Timer2 module.
Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

9.0 CAPTURE/COMPARE/PWM (CCP) MODULE

The Capture/Compare/PWM (CCP) module contains a 16-bit register that can operate as a:

- 16-bit Capture register
- 16-bit Compare register
- PWM Master/Slave Duty Cycle register.

Table 9-1 shows the timer resources of the CCP module modes.

Capture/Compare/PWM Register 1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. The special event trigger is generated by a compare match which will reset Timer1 and start an A/D conversion (if the A/D module is enabled).

The CCP module's input/output pin (CCP1) can be configured as RB0 or RB3. This selection is set in bit 12 (CCPMX) of the Configuration Word.

Additional information on the CCP module is available in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023) and in Application Note *AN594,* "*Using the CCP Module(s)*" (DS00594).

TABLE 9-1: CCP MODE – TIMER RESOURCE

CCP Mode	Timer Resource
Capture	Timer1
Compare	Timer1
PWM	Timer2

REGISTER 9-1: CCP1CON: CAPTURE/COMPARE/PWM CONTROL REGISTER 1 (ADDRESS 17h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **CCP1X:CCP1Y:** PWM Least Significant bits

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPR1L.

bit 3-0 **CCP1M<3:0>:** CCP1 Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCP1 module)
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, set output on match (CCP1IF bit is set)
1001 = Compare mode, clear output on match (CCP1IF bit is set)
1010 = Compare mode, generate software interrupt on match (CCP1IF bit is set, CCP1 pin is unaffected)
1011 = Compare mode, trigger special event (CCP1IF bit is set, CCP1 pin is unaffected); CCP1 resets TMR1 and starts an A/D conversion (if A/D module is enabled)
11xx = PWM mode

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

9.1 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 register when an event occurs on the CCP1 pin. An event is defined as:

- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

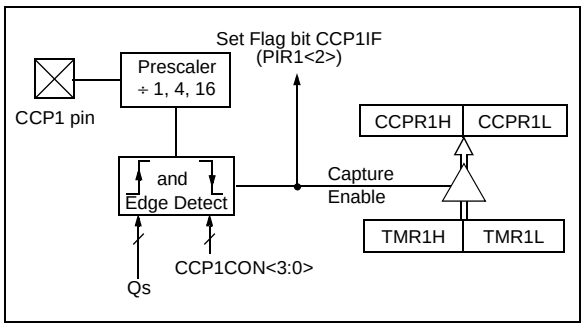
An event is selected by control bits CCP1M3:CCP1M0 (CCP1CON<3:0>). When a capture is made, the interrupt request flag bit, CCP1IF (PIR1<2>), is set. It must be cleared in software. If another capture occurs before the value in register CCPR1 is read, the old captured value is overwritten by the new captured value.

9.1.1 CCP PIN CONFIGURATION

In Capture mode, the CCP1 pin should be configured as an input by setting the TRISB<x> bit.

Note 1:	If the CCP1 pin is configured as an output, a write to the port can cause a capture condition.
2:	The TRISB bit (0 or 3) is dependent upon the setting of configuration bit 12 (CCPMX).

FIGURE 9-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



9.1.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode for the CCP module to use the capture feature. In Asynchronous Counter mode, the capture operation may not work.

9.1.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep bit CCP1IE (PIE1<2>) clear to avoid false interrupts and should clear the flag bit, CCP1IF, following any such change in operating mode.

9.1.4 CCP PRESCALER

There are four prescaler settings, specified by bits CCP1M3:CCP1M0. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared, therefore, the first capture may be from a non-zero prescaler. Example 9-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

EXAMPLE 9-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF    CCP1CON      ;Turn CCP module off
MOVLW   NEW_CAPT_PS  ;Load the W reg with
                     ;the new prescaler
                     ;move value and CCP ON
MOVWF   CCP1CON      ;Load CCP1CON with this
                     ;value
```

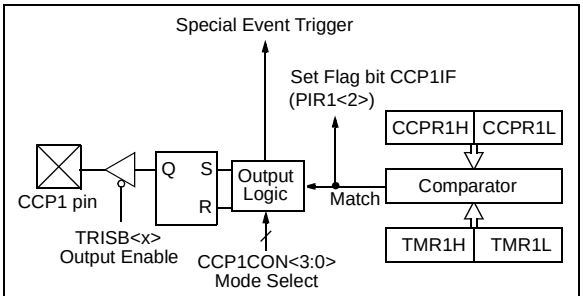
9.2 Compare Mode

In Compare mode, the 16-bit CCPR1 register value is constantly compared against the TMR1 register pair value. When a match occurs, the CCP1 pin is:

- Driven high
- Driven low
- Remains unchanged

The action on the pin is based on the value of control bits CCP1M3:CCP1M0 (CCP1CON<3:0>). At the same time, interrupt flag bit, CCP1IF, is set.

FIGURE 9-2: COMPARE MODE OPERATION BLOCK DIAGRAM



Special Event Trigger will:

- Reset Timer1 but not set interrupt flag bit, TMR1IF (PIR1<0>)
- Set bit GO/$\overline{\text{DONE}}$ (ADCON0<2>) which starts an A/D conversion

9.2.1 CCP PIN CONFIGURATION

The user must configure the CCP1 pin as an output by clearing the TRISB<x> bit.

> **Note 1:** Clearing the CCP1CON register will force the CCP1 compare output latch to the default low level. This is not the data latch.
>
> **2:** The TRISB bit (0 or 3) is dependent upon the setting of configuration bit 12 (CCPMX).

9.2.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

9.2.3 SOFTWARE INTERRUPT MODE

When generate software interrupt is chosen, the CCP1 pin is not affected. Only a CCP interrupt is generated (if enabled).

9.2.4 SPECIAL EVENT TRIGGER

In this mode, an internal hardware trigger is generated that may be used to initiate an action.

The special event trigger output of CCP1 resets the TMR1 register pair and starts an A/D conversion (if the A/D module is enabled). This allows the CCPR1 register to effectively be a 16-bit programmable period register for Timer1.

> **Note:** The special event trigger from the CCP1 module will not set interrupt flag bit TMR1IF (PIR1<0>).

TABLE 9-2: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE AND TIMER1

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
0Bh,8Bh 10BH,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
86h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	uuuu uuuu
10h	T1CON	—	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	-000 0000	-uuu uuuu
15h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	uuuu uuuu
16h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	uuuu uuuu
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by Capture and Timer1.

Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

9.3 PWM Mode

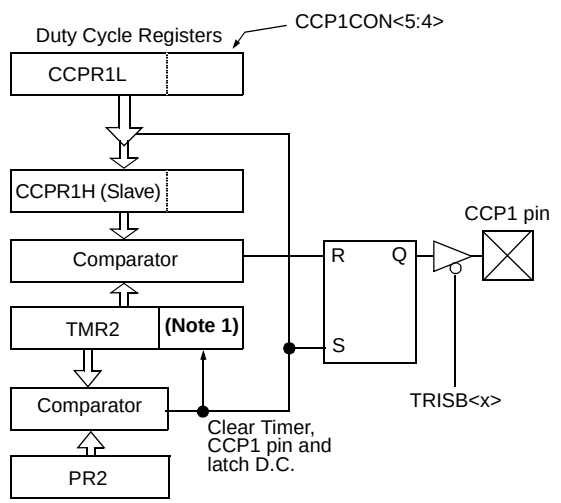
In Pulse-Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTB data latch, the TRISB<x> bit must be cleared to make the CCP1 pin an output.

Note:	Clearing the CCP1CON register will force the CCP1 PWM output latch to the default low level. This is not the PORTB I/O data latch.

Figure 9-3 shows a simplified block diagram of the CCP module in PWM mode.

For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 9.3.3 "Setup for PWM Operation"**.

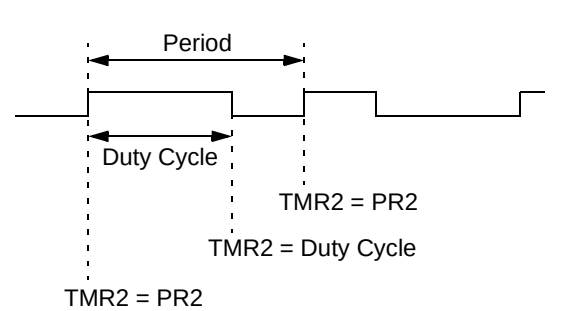
FIGURE 9-3: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: 8-bit timer is concatenated with 2-bit internal Q clock, or 2 bits of the prescaler, to create 10-bit time base.

A PWM output (Figure 9-4) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 9-4: PWM OUTPUT



9.3.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula.

EQUATION 9-1:

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period].

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (exception: if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is latched from CCPR1L into CCPR1H

Note:	The Timer2 postscaler (see **Section 8.0 "Timer2 Module"**) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

9.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSbs and the CCP1CON<5:4> bits contain the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time.

EQUATION 9-2:

$$\text{PWM Duty Cycle} = (CCPR1L{:}CCP1CON\langle 5{:}4\rangle) \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR1H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read-only register.

The CCPR1H register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation.

When the CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the following formula.

EQUATION 9-3:

$$\text{Resolution} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ bits}$$

Note:	If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.

9.3.3 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the TRISB<x> bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCP1 module for PWM operation.

Note:	The TRISB bit (0 or 3) is dependant upon the setting of configuration bit 12 (CCPMX).

TABLE 9-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 20 MHz

PWM Frequency	1.22 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	6.6

TABLE 9-4: REGISTERS ASSOCIATED WITH PWM AND TIMER2

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
0Bh,8Bh 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
86h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
11h	TMR2	Timer2 Module Register								0000 0000	0000 0000
92h	PR2	Timer2 Period Register								1111 1111	1111 1111
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
15h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	uuuu uuuu
16h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	uuuu uuuu
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by PWM and Timer2.
Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

NOTES:

10.0 SYNCHRONOUS SERIAL PORT (SSP) MODULE

10.1 SSP Module Overview

The Synchronous Serial Port (SSP) module is a serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The SSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI™)
- Inter-Integrated Circuit (I^2C™)

An overview of I^2C operations and additional information on the SSP module can be found in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023).

Refer to Application Note *AN578, "Use of the SSP Module in the I^2C™ Multi-Master Environment"* (DS00578).

10.2 SPI Mode

This section contains register definitions and operational characteristics of the SPI module.

SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. To accomplish communication, typically three pins are used:

- Serial Data Out (SDO) RB2/SDO/RX/DT
- Serial Data In (SDI) RB1/SDI/SDA
- Serial Clock (SCK) RB4/SCK/SCL

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SS}$) RB5/$\overline{SS}$/TX/CK

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits in the SSPCON register (SSPCON<5:0>) and the SSPSTAT register (SSPSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCK is the clock output)
- Slave mode (SCK is the clock input)
- Clock Polarity (Idle state of SCK)
- Clock Edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

Note:	Before enabling the module in SPI Slave mode, the state of the clock line (SCK) must match the polarity selected for the Idle state. The clock line can be observed by reading the SCK pin. The polarity of the Idle state is determined by the CKP bit (SSPCON<4>).

REGISTER 10-1: SSPSTAT: SYNCHRONOUS SERIAL PORT STATUS REGISTER (ADDRESS 94h)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P(1)	S(1)	R/$\overline{W}$	UA	BF
bit 7							bit 0

bit 7 **SMP:** SPI Data Input Sample Phase bit

SPI Master mode:
1 = Input data sampled at end of data output time
0 = Input data sampled at middle of data output time (Microwire)

SPI Slave mode:
This bit must be cleared when SPI is used in Slave mode.

I^2C mode:
This bit must be maintained clear.

bit 6 **CKE:** SPI Clock Edge Select bit

1 = Transmit occurs on transition from active to Idle clock state
0 = Transmit occurs on transition from Idle to active clock state

Note: Polarity of clock state is set by the CKP bit (SSPCON<4>).

bit 5 **D/$\overline{A}$:** Data/$\overline{Address}$ bit (I^2C mode only)

In I^2C Slave mode:
1 = Indicates that the last byte received was data
0 = Indicates that the last byte received was address

bit 4 **P:** Stop bit(1) (I^2C mode only)

1 = Indicates that a Start bit has been detected last
0 = Stop bit was not detected last

bit 3 **S:** Start bit(1) (I^2C mode only)

1 = Indicates that a Start bit has been detected last (this bit is '0' on Reset)
0 = Start bit was not detected last

bit 2 **R/$\overline{W}$:** Read/$\overline{Write}$ Information bit (I^2C mode only)

Holds the R/$\overline{W}$ bit information following the last address match and is only valid from address match to the next Start bit, Stop bit or $\overline{ACK}$ bit.

1 = Read
0 = Write

bit 1 **UA:** Update Address bit (10-bit I^2C mode only)

1 = Indicates that the user needs to update the address in the SSPADD register
0 = Address does not need to be updated

bit 0 **BF:** Buffer Full Status bit

Receive (SPI and I^2C modes):
1 = Receive complete, SSPBUF is full
0 = Receive not complete, SSPBUF is empty

Transmit (in I^2C mode only):
1 = Transmit in progress, SSPBUF is full (8 bits)
0 = Transmit complete, SSPBUF is empty

Note 1: This bit is cleared when the SSP module is disabled (i.e., the SSPEN bit is cleared).

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 10-2: SSPCON: SYNCHRONOUS SERIAL PORT CONTROL REGISTER (ADDRESS 14h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN(1)	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL:** Write Collision Detect bit

1 = An attempt to write the SSPBUF register failed because the SSP module is busy (must be cleared in software)

0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator bit

In SPI mode:

1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. The user must read the SSPBUF, even if only transmitting data, to avoid setting overflow. In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPBUF register.

0 = No overflow

In I^2C mode:

1 = A byte is received while the SSPBUF register is still holding the previous byte. SSPOV is a “don’t care” in Transmit mode. SSPOV must be cleared in software in either mode.

0 = No overflow

bit 5 **SSPEN:** Synchronous Serial Port Enable bit(1)

In SPI mode:

1 = Enables serial port and configures SCK, SDO and SDI as serial port pins

0 = Disables serial port and configures these pins as I/O port pins

In I^2C mode:

1 = Enables the serial port and configures the SDA and SCL pins as serial port pins

0 = Disables serial port and configures these pins as I/O port pins

Note 1: In both modes, when enabled, these pins must be properly configured as input or output.

bit 4 **CKP:** Clock Polarity Select bit

In SPI mode:

1 = Transmit happens on falling edge, receive on rising edge. Idle state for clock is a high level.

0 = Transmit happens on rising edge, receive on falling edge. Idle state for clock is a low level.

In I^2C Slave mode:

SCK release control

1 = Enable clock

0 = Holds clock low (clock stretch). (Used to ensure data setup time.)

bit 3-0 **SSPM<3:0>:** Synchronous Serial Port Mode Select bits

0000 = SPI Master mode, clock = OSC/4

0001 = SPI Master mode, clock = OSC/16

0010 = SPI Master mode, clock = OSC/64

0011 = SPI Master mode, clock = TMR2 output/2

0100 = SPI Slave mode, clock = SCK pin. $\overline{SS}$ pin control enabled.

0101 = SPI Slave mode, clock = SCK pin. $\overline{SS}$ pin control disabled. $\overline{SS}$ can be used as I/O pin.

0110 = I^2C Slave mode, 7-bit address

0111 = I^2C Slave mode, 10-bit address

1011 = I^2C Firmware Controlled Master mode (Slave Idle)

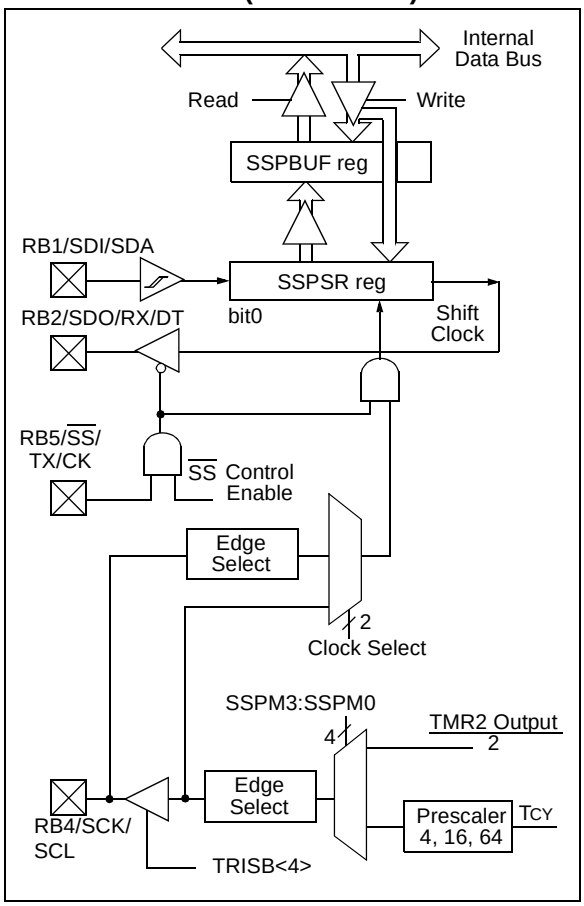
1110 = I^2C Slave mode, 7-bit address with Start and Stop bit interrupts enabled

1111 = I^2C Slave mode, 10-bit address with Start and Stop bit interrupts enabled

1000, 1001, 1010, 1100, 1101 = Reserved

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

FIGURE 10-1: SSP BLOCK DIAGRAM (SPI™ MODE)



To enable the serial port, SSP Enable bit, SSPEN (SSPCON<5>), must be set. To reset or reconfigure SPI mode, clear bit SSPEN, reinitialize the SSPCON register and then set bit SSPEN. This configures the SDI, SDO, SCK and $\overline{SS}$ pins as serial port pins. For the pins to behave as the serial port function, they must have their data direction bits (in the TRISB register) appropriately programmed. That is:

- SDI must have TRISB<1> set
- SDO must have TRISB<2> cleared
- SCK (Master mode) must have TRISB<4> cleared
- SCK (Slave mode) must have TRISB<4> set
- $\overline{SS}$ must have TRISB<5> set

> **Note 1:** When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPCON<3:0> = 0100), the SPI module will reset if the $\overline{SS}$ pin is set to VDD.
>
> **2:** If the SPI is used in Slave mode with CKE = 1, then the $\overline{SS}$ pin control must be enabled.

TABLE 10-1: REGISTERS ASSOCIATED WITH SPI™ OPERATION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
0Bh,8Bh 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
86h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	uuuu uuuu
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
94h	SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the SSP in SPI™ mode.

Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

FIGURE 10-2: SPI™ MODE TIMING (MASTER MODE)

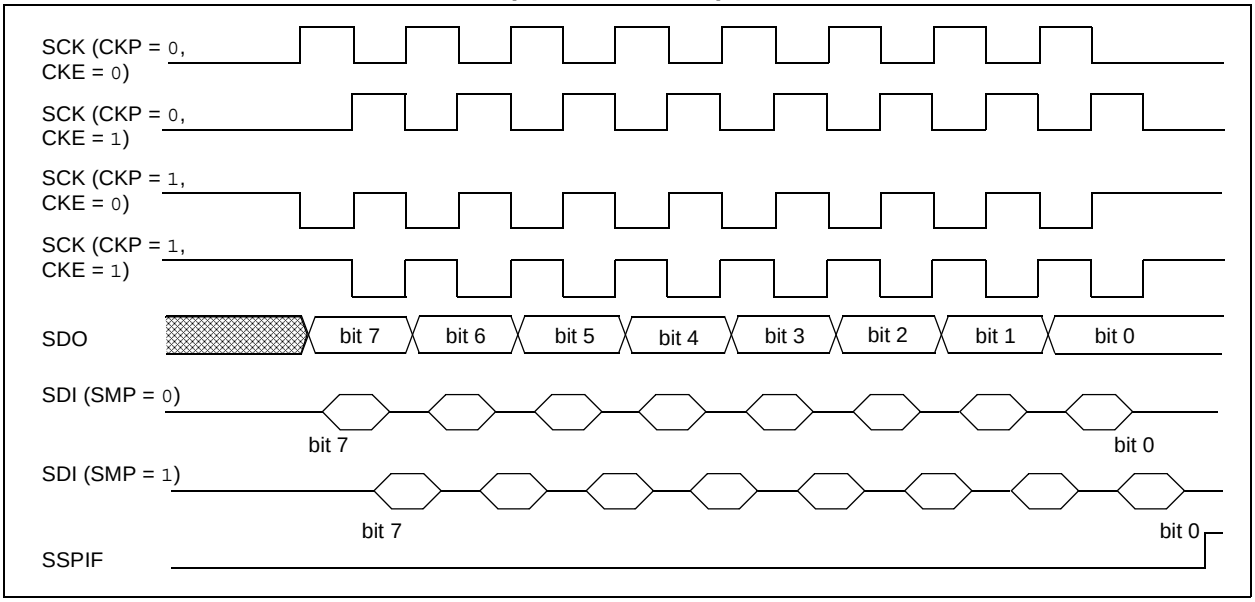


FIGURE 10-3: SPI™ MODE TIMING (SLAVE MODE WITH CKE = 0)

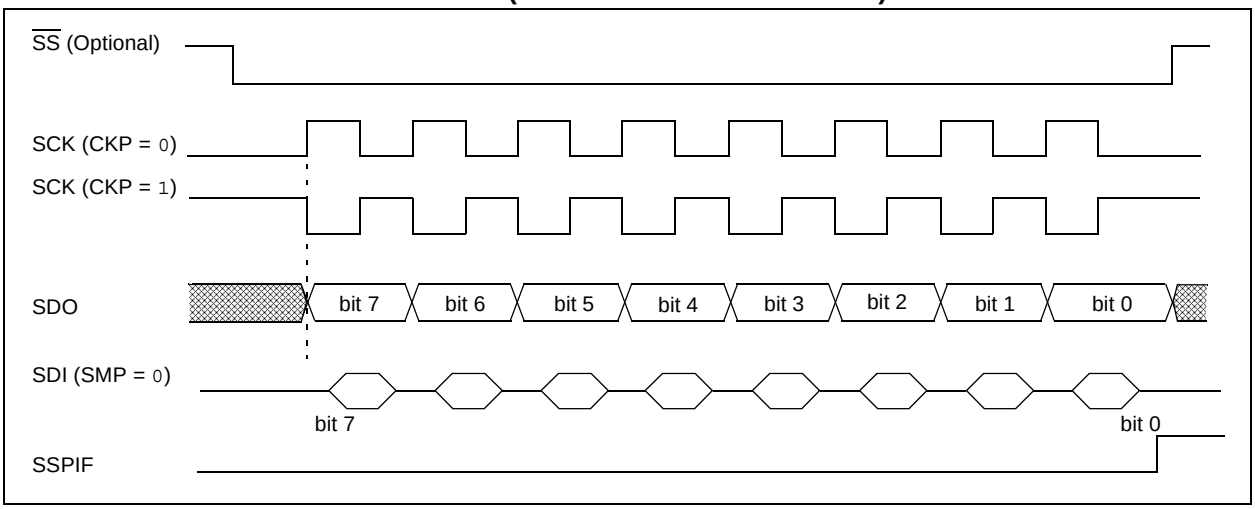
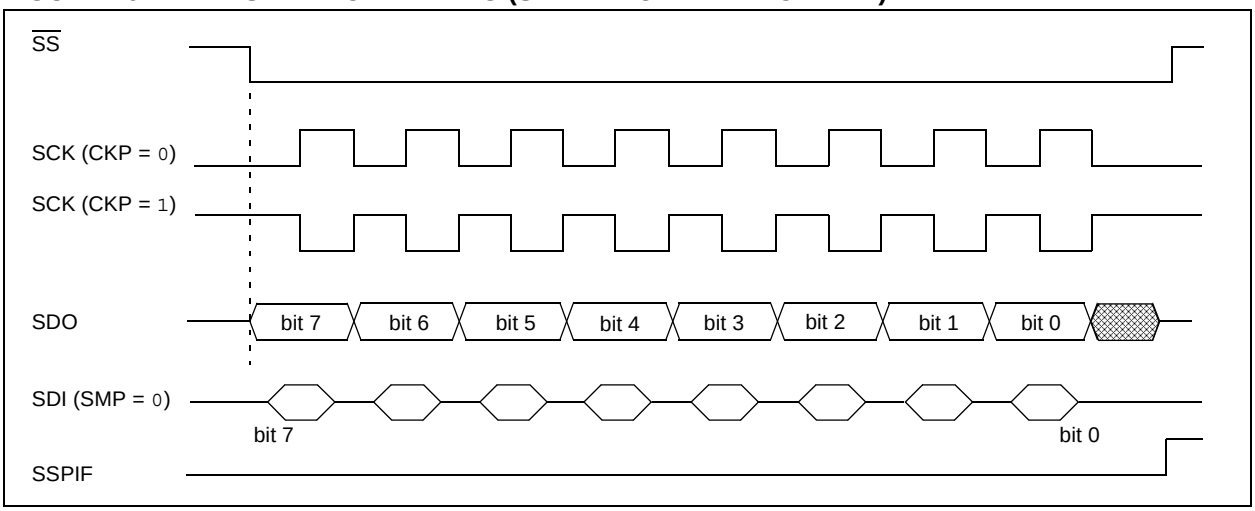


FIGURE 10-4: SPI™ MODE TIMING (SLAVE MODE WITH CKE = 1)



10.3 SSP I²C Mode Operation

The SSP module in I²C mode fully implements all slave functions, except general call support and provides interrupts on Start and Stop bits in hardware to facilitate firmware implementations of the master functions. The SSP module implements the standard mode specifications, as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer. These are the RB4/SCK/SCL pin, which is the clock (SCL) and the RB1/SDI/SDA pin, which is the data (SDA). The user must configure these pins as inputs or outputs through the TRISB<4,1> bits.

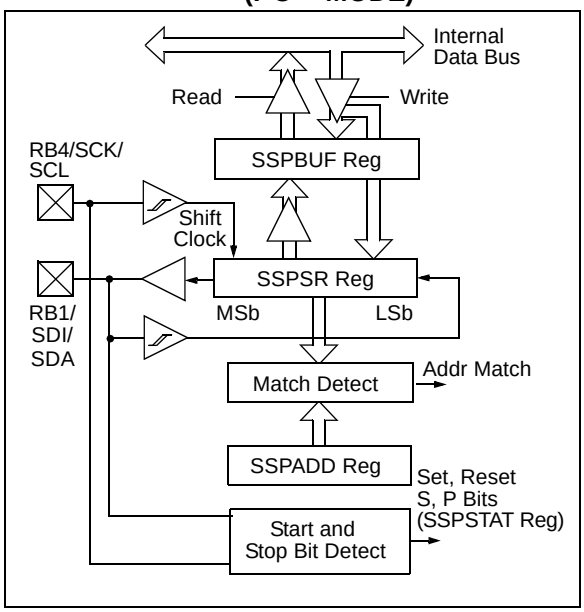
To ensure proper communication of the I²C Slave mode, the TRIS bits (TRISx [SDA, SCL]) corresponding to the I²C pins must be set to '1'. If any TRIS bits (TRISx<7:0>) of the port containing the I²C pins (PORTx [SDA, SCL]) are changed in software during I²C communication using a Read-Modify-Write instruction (BSF, BCF), then the I²C mode may stop functioning properly and I²C communication may suspend. Do not change any of the TRISx bits (TRIS bits of the port containing the I²C pins) using the instruction BSF or BCF during I²C communication. If it is absolutely necessary to change the TRISx bits during communication, the following method can be used:

EXAMPLE 10-1:

```
MOVF    TRISC, W        ; Example for an 18-pin part such as the PIC16F818/819
IORLW   0x18            ; Ensures <4:3> bits are '11'
ANDLW   B'11111001'     ; Sets <2:1> as output, but will not alter other bits
                        ; User can use their own logic here, such as IORLW, XORLW and ANDLW
MOVWF   TRISC
```

The SSP module functions are enabled by setting SSP Enable bit, SSPEN (SSPCON<5>).

FIGURE 10-5: SSP BLOCK DIAGRAM (I²C™ MODE)



The SSP module has five registers for I²C operation:

- SSP Control register (SSPCON)
- SSP Status register (SSPSTAT)
- Serial Receive/Transmit Buffer register (SSPBUF)
- SSP Shift register (SSPSR) – Not directly accessible
- SSP Address register (SSPADD)

The SSPCON register allows control of the I²C operation. Four mode selection bits (SSPCON<3:0>) allow one of the following I²C modes to be selected:

- I²C Slave mode (7-bit address)
- I²C Slave mode (10-bit address)
- I²C Slave mode (7-bit address) with Start and Stop bit interrupts enabled to support Firmware Controlled Master mode
- I²C Slave mode (10-bit address) with Start and Stop bit interrupts enabled to support Firmware Controlled Master mode
- I²C Firmware Controlled Master mode operation with Start and Stop bit interrupts enabled; slave is Idle

Selection of any I²C mode, with the SSPEN bit set, forces the SCL and SDA pins to be open-drain, provided these pins are programmed to inputs by setting the appropriate TRISB bits. Pull-up resistors must be provided externally to the SCL and SDA pins for proper operation of the I²C module.

Additional information on SSP I²C operation may be found in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023).

10.3.1 SLAVE MODE

In Slave mode, the SCL and SDA pins must be configured as inputs (TRISB<4,1> set). The SSP module will override the input state with the output data when required (slave-transmitter).

When an address is matched, or the data transfer after an address match is received, the hardware automatically will generate the Acknowledge ($\overline{ACK}$) pulse and then load the SSPBUF register with the received value currently in the SSPSR register.

Either or both of the following conditions will cause the SSP module not to give this $\overline{ACK}$ pulse:

a) The Buffer Full bit, BF (SSPSTAT<0>), was set before the transfer was received.
b) The Overflow bit, SSPOV (SSPCON<6>), was set before the transfer was received.

In this case, the SSPSR register value is not loaded into the SSPBUF, but bit SSPIF (PIR1<3>) is set. Table 10-2 shows what happens when a data transfer byte is received, given the status of bits BF and SSPOV. The shaded cells show the condition where user software did not properly clear the overflow condition. Flag bit, BF, is cleared by reading the SSPBUF register while bit, SSPOV, is cleared through software.

The SCL clock input must have a minimum high and low for proper operation. The high and low times of the I^2C specification, as well as the requirement of the SSP module, are shown in timing parameter #100 and parameter #101.

10.3.1.1 Addressing

Once the SSP module has been enabled, it waits for a Start condition to occur. Following the Start condition, the eight bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register SSPSR<7:1> is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match and the BF and SSPOV bits are clear, the following events occur:

a) The SSPSR register value is loaded into the SSPBUF register.
b) The Buffer Full bit, BF, is set.
c) An $\overline{ACK}$ pulse is generated.
d) SSP Interrupt Flag bit, SSPIF (PIR1<3>), is set (interrupt is generated if enabled) – on the falling edge of the ninth SCL pulse.

In 10-bit Address mode, two address bytes need to be received by the slave device. The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. Bit R/$\overline{W}$ (SSPSTAT<2>) must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '`1111 0 A9 A8 0`', where `A9` and `A8` are the two MSbs of the address.

The sequence of events for 10-bit Address mode is as follows, with steps 7-9 for slave transmitter:

1. Receive first (high) byte of address (bits SSPIF, BF and UA (SSPSTAT<1>) are set).
2. Update the SSPADD register with second (low) byte of address (clears bit UA and releases the SCL line).
3. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.
4. Receive second (low) byte of address (bits SSPIF, BF and UA are set).
5. Update the SSPADD register with the first (high) byte of address; if match releases SCL line, this will clear bit UA.
6. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.
7. Receive Repeated Start condition.
8. Receive first (high) byte of address (bits SSPIF and BF are set).
9. Read the SSPBUF register (clears bit BF) and clear flag bit SSPIF.

10.3.1.2 Reception

When the R/$\overline{W}$ bit of the address byte is clear and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is cleared. The received address is loaded into the SSPBUF register.

When the address byte overflow condition exists, then a no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is indicated if either bit, BF (SSPSTAT<0>), is set or bit, SSPOV (SSPCON<6>), is set.

An SSP interrupt is generated for each data transfer byte. Flag bit, SSPIF (PIR1<3>), must be cleared in software. The SSPSTAT register is used to determine the status of the byte.

10.3.1.3 Transmission

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit and pin RB4/SCK/SCL is held low. The transmit data must be loaded into the SSPBUF register which also loads the SSPSR register. Then, pin RB4/SCK/SCL should be enabled by setting bit CKP (SSPCON<4>). The master device must monitor the SCL pin prior to asserting another clock pulse. The slave devices may be holding off the master device by stretching the clock. The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 10-7).

An SSP interrupt is generated for each data transfer byte. Flag bit, SSPIF, must be cleared in software and the SSPSTAT register is used to determine the status of the byte. Flag bit, SSPIF, is set on the falling edge of the ninth clock pulse.

As a slave transmitter, the $\overline{ACK}$ pulse from the master receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line was high (not $\overline{ACK}$), then the data transfer is complete. When the $\overline{ACK}$ is latched by the slave device, the slave logic is reset (resets SSPSTAT register) and the slave device then monitors for another occurrence of the Start bit. If the SDA line was low ($\overline{ACK}$), the transmit data must be loaded into the SSPBUF register which also loads the SSPSR register. Then, pin RB4/SCK/SCL should be enabled by setting bit CKP.

TABLE 10-2: DATA TRANSFER RECEIVED BYTE ACTIONS

Status Bits as Data Transfer is Received		SSPSR → SSPBUF	Generate $\overline{ACK}$ Pulse	Set SSPIF Bit (SSP Interrupt Occurs if Enabled)
BF	SSPOV			
0	0	Yes	Yes	Yes
1	0	No	No	Yes
1	1	No	No	Yes
0	1	No	No	Yes

Note 1: Shaded cells show the conditions where the user software did not properly clear the overflow condition.

FIGURE 10-6: I²C™ WAVEFORMS FOR RECEPTION (7-BIT ADDRESS)

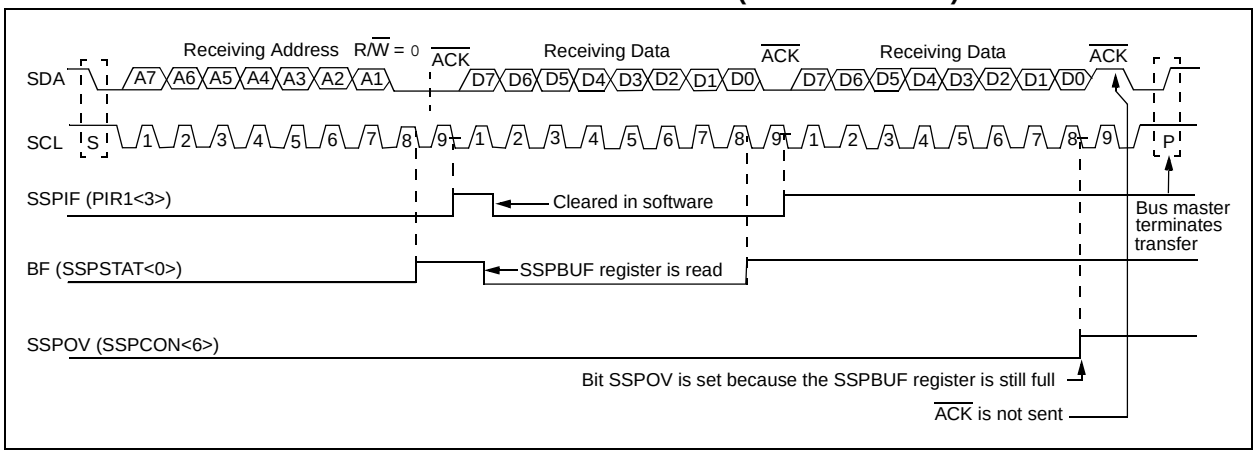
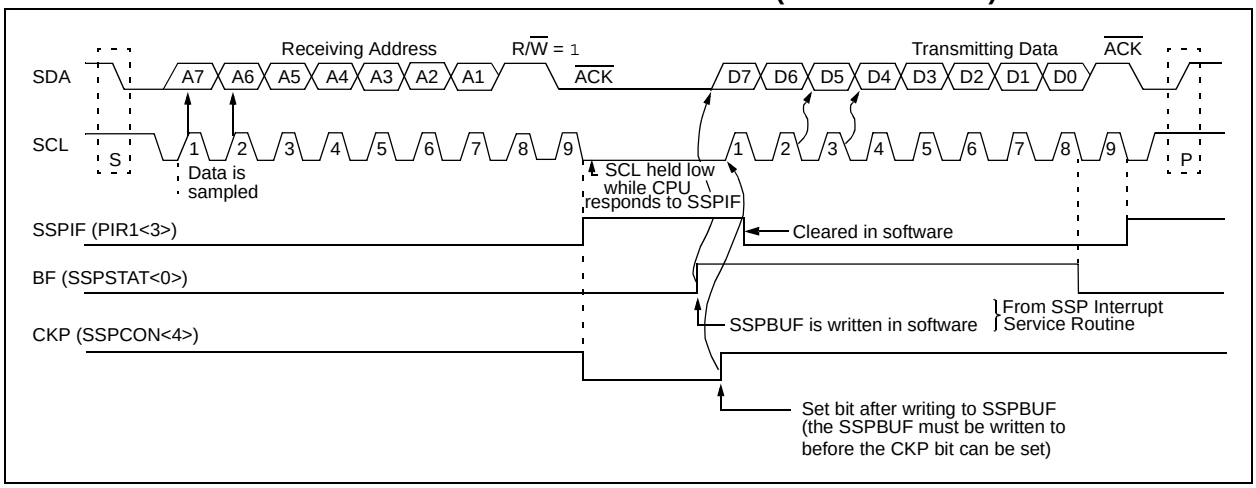


FIGURE 10-7: I²C™ WAVEFORMS FOR TRANSMISSION (7-BIT ADDRESS)



10.3.2 MASTER MODE OPERATION

Master mode operation is supported in firmware using interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset, or when the SSP module is disabled. The Stop (P) and Start (S) bits will toggle based on the Start and Stop conditions. Control of the I^2C bus may be taken when the P bit is set, or the bus is Idle and both the S and P bits are clear.

In Master mode operation, the SCL and SDA lines are manipulated in firmware by clearing the corresponding TRISB<4,1> bit(s). The output level is always low, irrespective of the value(s) in PORTB<4,1>. So, when transmitting data, a '1' data bit must have the TRISB<1> bit set (input) and a '0' data bit must have the TRISB<1> bit cleared (output). The same scenario is true for the SCL line with the TRISB<4> bit. Pull-up resistors must be provided externally to the SCL and SDA pins for proper operation of the I^2C module.

The following events will cause the SSP Interrupt Flag bit, SSPIF, to be set (SSP Interrupt if enabled):

- Start condition
- Stop condition
- Data transfer byte transmitted/received

Master mode operation can be done with either the Slave mode Idle (SSPM3:SSPM0 = 1011), or with the Slave mode active. When both Master mode operation and Slave modes are used, the software needs to differentiate the source(s) of the interrupt.

For more information on Master mode operation, see Application Note *AN554, "Software Implementation of I^2C™ Bus Master"*.

10.3.3 MULTI-MASTER MODE OPERATION

In Multi-Master mode operation, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset, or when the SSP module is disabled. The Stop (P) and Start (S) bits will toggle based on the Start and Stop conditions. Control of the I^2C bus may be taken when bit P (SSPSTAT<4>) is set, or the bus is Idle and both the S and P bits clear. When the bus is busy, enabling the SSP interrupt will generate the interrupt when the Stop condition occurs.

In Multi-Master mode operation, the SDA line must be monitored to see if the signal level is the expected output level. This check only needs to be done when a high level is output. If a high level is expected and a low level is present, the device needs to release the SDA and SCL lines (set TRISB<4,1>). There are two stages where this arbitration can be lost:

- Address Transfer
- Data Transfer

When the slave logic is enabled, the slave device continues to receive. If arbitration was lost during the address transfer stage, communication to the device may be in progress. If addressed, an $\overline{ACK}$ pulse will be generated. If arbitration was lost during the data transfer stage, the device will need to retransfer the data at a later time.

For more information on Multi-Master mode operation, see Application Note *AN578, "Use of the SSP Module in the of I^2C™ Multi-Master Environment"*.

TABLE 10-3: REGISTERS ASSOCIATED WITH I^2C™ OPERATION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	uuuu uuuu
93h	SSPADD	Synchronous Serial Port (I^2C mode) Address Register								0000 0000	0000 0000
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
94h	SSPSTAT	SMP(2)	CKE(2)	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000
86h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'.
Shaded cells are not used by SSP module in SPI™ mode.

Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

2: Maintain these bits clear in I^2C™ mode.

NOTES:

11.0 ADDRESSABLE UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (AUSART)

The Addressable Universal Synchronous Asynchronous Receiver Transmitter (AUSART) module is one of the two serial I/O modules. (AUSART is also known as a Serial Communications Interface or SCI.) The AUSART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers, or it can be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The AUSART can be configured in the following modes:

- Asynchronous (full-duplex)
- Synchronous – Master (half-duplex)
- Synchronous – Slave (half-duplex)

Bit SPEN (RCSTA<7>) and bits TRISB<5,2> have to be set in order to configure pins, RB5/$\overline{SS}$/TX/CK and RB2/SDO/RX/DT, as the Addressable Universal Synchronous Asynchronous Receiver Transmitter.

The AUSART module also has a multi-processor communication capability, using 9-bit address detection.

REGISTER 11-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER (ADDRESS 98h)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC:** Clock Source Select bit
Asynchronous mode:
Don't care.
Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9**: 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN**: Transmit Enable bit
1 = Transmit enabled
0 = Transmit disabled

Note: SREN/CREN overrides TXEN in Sync mode.

bit 4 **SYNC**: AUSART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

bit 3 **Unimplemented:** Read as '0'

bit 2 **BRGH**: High Baud Rate Select bit
Asynchronous mode:
1 = High speed
0 = Low speed
Synchronous mode:
Unused in this mode.

bit 1 **TRMT**: Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

bit 0 **TX9D:** 9th bit of Transmit Data, can be Parity bit

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 11-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER (ADDRESS 18h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN:** Serial Port Enable bit
1 = Serial port enabled (configures RB2/SDO/RX/DT and RB5/$\overline{SS}$/TX/CK pins as serial port pins)
0 = Serial port disabled

bit 6 **RX9**: 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN**: Single Receive Enable bit
Asynchronous mode:
Don't care.
Synchronous mode – Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode – Slave:
Don't care.

bit 4 **CREN**: Continuous Receive Enable bit
Asynchronous mode:
1 = Enables continuous receive
0 = Disables continuous receive
Synchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection, enables interrupt and load of the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit

bit 2 **FERR**: Framing Error bit
1 = Framing error (can be updated by reading RCREG register and receive next valid byte)
0 = No framing error

bit 1 **OERR**: Overrun Error bit
1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error

bit 0 **RX9D:** 9th bit of Received Data (can be Parity bit, but must be calculated by user firmware)

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

11.1 AUSART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the AUSART. It is a dedicated 8-bit Baud Rate Generator. The SPBRG register controls the period of a free running 8-bit timer. In Asynchronous mode, bit BRGH (TXSTA<2>) also controls the baud rate. In Synchronous mode, bit BRGH is ignored. Table 11-1 shows the formula for computation of the baud rate for different AUSART modes which only apply in Master mode (internal clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRG register can be calculated using the formula in Table 11-1. From this, the error in baud rate can be determined.

It may be advantageous to use the high baud rate (BRGH = 1) even for slower baud clocks. This is because the FOSC/(16(X + 1)) equation can reduce the baud rate error in some cases.

Writing a new value to the SPBRG register causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

11.1.1 AUSART AND INTRC OPERATION

The PIC16F87/88 has an 8 MHz INTRC that can be used as the system clock, thereby eliminating the need for external components to provide the clock source. When the INTRC provides the system clock, the AUSART module will also use the INTRC as its system clock. Table 11-1 shows some of the INTRC frequencies that can be used to generate the AUSART module's baud rate.

11.1.2 LOW-POWER MODE OPERATION

The system clock is used to generate the desired baud rate; however, when a low-power mode is entered, the low-power clock source may be operating at a different frequency than in full power execution. In Sleep mode, no clocks are present. This may require the value in SPBRG to be adjusted.

11.1.3 SAMPLING

The data on the RB2/SDO/RX/DT pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

TABLE 11-1: BAUD RATE FORMULA

SYNC	BRGH = 0 (Low Speed)	BRGH = 1 (High Speed)
0	(Asynchronous) Baud Rate = FOSC/(64(X + 1))	Baud Rate = FOSC/(16(X + 1))
1	(Synchronous) Baud Rate = FOSC/(4(X + 1))	N/A

Legend: X = value in SPBRG (0 to 255)

TABLE 11-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used by the BRG.

TABLE 11-3: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 0)

BAUD RATE (K)	Fosc = 20 MHz			Fosc = 16 MHz			Fosc = 10 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	—	—	—	—	—	—	—	—	—
1.2	1.221	+1.75	255	1.202	+0.17	207	1.202	+0.17	129
2.4	2.404	+0.17	129	2.404	+0.17	103	2.404	+0.17	64
9.6	9.766	+1.73	31	9.615	+0.16	25	9.766	+1.73	15
19.2	19.531	+ 1.72	15	19.231	+0.16	12	19.531	+1.72	7
28.8	31.250	+8.51	9	27.778	-3.55	8	31.250	+8.51	4
33.6	34.722	+3.34	8	35.714	+6.29	6	31.250	-6.99	4
57.6	62.500	+8.51	4	62.500	+8.51	3	52.083	-9.58	2
HIGH	1.221	—	255	0.977	—	255	0.610	—	255
LOW	312.500	—	0	250.000	—	0	156.250	—	0

BAUD RATE (K)	Fosc = 4 MHz			Fosc = 3.6864 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	0.300	0	207	0.3	0	191
1.2	1.202	+0.17	51	1.2	0	47
2.4	2.404	+0.17	25	2.4	0	23
9.6	8.929	+6.99	6	9.6	0	5
19.2	20.833	+8.51	2	19.2	0	2
28.8	31.250	+8.51	1	28.8	0	1
33.6	—	—	—	—	—	—
57.6	62.500	+8.51	0	57.6	0	0
HIGH	0.244	—	255	0.225	—	255
LOW	62.500	—	0	57.6	—	0

TABLE 11-4: BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 1)

BAUD RATE (K)	Fosc = 20 MHz			Fosc = 16 MHz			Fosc = 10 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—
2.4	—	—	—	—	—	—	2.441	+1.71	255
9.6	9.615	+0.16	129	9.615	+0.16	103	9.615	+0.16	64
19.2	19.231	+0.16	64	19.231	+0.16	51	19.531	+1.72	31
28.8	29.070	+0.94	42	29.412	+2.13	33	28.409	-1.36	21
33.6	33.784	+0.55	36	33.333	-0.79	29	32.895	-2.10	18
57.6	59.524	+3.34	20	58.824	+2.13	16	56.818	-1.36	10
HIGH	4.883	—	255	3.906	—	255	2.441	—	255
LOW	1250.000	—	0	1000.000	—	0	625.000	—	0

BAUD RATE (K)	Fosc = 4 MHz			Fosc = 3.6864 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	—	—	—	—	—	—
1.2	1.202	+0.17	207	1.2	0	191
2.4	2.404	+0.17	103	2.4	0	95
9.6	9.615	+0.16	25	9.6	0	23
19.2	19.231	+0.16	12	19.2	0	11
28.8	27.798	-3.55	8	28.8	0	7
33.6	35.714	+6.29	6	32.9	-2.04	6
57.6	62.500	+8.51	3	57.6	0	3
HIGH	0.977	—	255	0.9	—	255
LOW	250.000	—	0	230.4	—	0

TABLE 11-5: INTRC BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 0)

BAUD RATE (K)	Fosc = 8 MHz			Fosc = 4 MHz			Fosc = 2 MHz			Fosc = 1 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	—	—	0.300	0	207	0.300	0	103	0.300	0	51
1.2	1.202	+0.16	103	1.202	+0.16	51	1.202	+0.16	25	1.202	+0.16	12
2.4	2.404	+0.16	51	2.404	+0.16	25	2.404	+0.16	12	2.232	-6.99	6
9.6	9.615	+0.16	12	8.929	-6.99	6	10.417	+8.51	2	NA	—	—
19.2	17.857	-6.99	6	20.833	+8.51	2	NA	—	—	NA	—	—
28.8	31.250	+8.51	3	31.250	+8.51	1	31.250	+8.51	0	NA	—	—
38.4	41.667	+8.51	2	NA	—	—	NA	—	—	NA	—	—
57.6	62.500	+8.51	1	62.500	8.51	0	NA	—	—	NA	—	—

TABLE 11-6: INTRC BAUD RATES FOR ASYNCHRONOUS MODE (BRGH = 1)

BAUD RATE (K)	Fosc = 8 MHz			Fosc = 4 MHz			Fosc = 2 MHz			Fosc = 1 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	NA	—	—	NA	—	—	NA	—	—	0.300	0	207
1.2	NA	—	—	1.202	+0.16	207	1.202	+0.16	103	1.202	+0.16	51
2.4	2.404	+0.16	207	2.404	+0.16	103	2.404	+0.16	51	2.404	+0.16	25
9.6	9.615	+0.16	51	9.615	+0.16	25	9.615	+0.16	12	8.929	-6.99	6
19.2	19.231	+0.16	25	19.231	+0.16	12	17.857	-6.99	6	20.833	+8.51	2
28.8	29.412	+2.12	16	27.778	-3.55	8	31.250	+8.51	3	31.250	+8.51	1
38.4	38.462	+0.16	12	35.714	-6.99	6	41.667	+8.51	2	NA	—	—
57.6	55.556	-3.55	8	62.500	+8.51	3	62.500	+8.51	1	62.500	+8.51	0

11.2 AUSART Asynchronous Mode

In this mode, the AUSART uses standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip, dedicated, 8-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator. The AUSART transmits and receives the LSb first. The transmitter and receiver are functionally independent, but use the same data format and baud rate. The Baud Rate Generator produces a clock, either x16 or x64 of the bit shift rate, depending on bit BRGH (TXSTA<2>). Parity is not supported by the hardware, but can be implemented in software (and stored as the ninth data bit). Asynchronous mode is stopped during Sleep.

Asynchronous mode is selected by clearing bit SYNC (TXSTA<4>).

The AUSART Asynchronous module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver

11.2.1 AUSART ASYNCHRONOUS TRANSMITTER

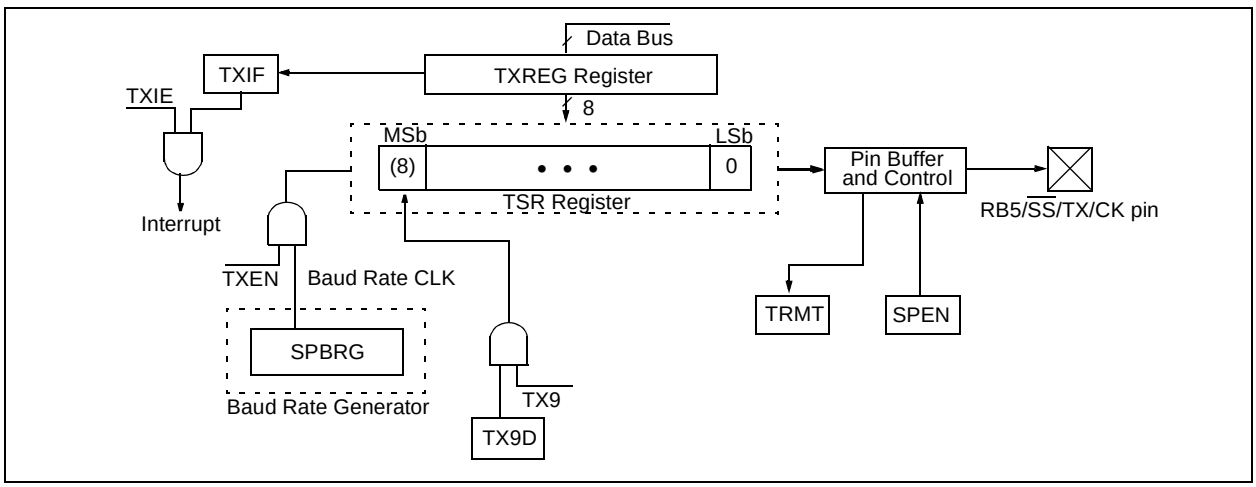
The AUSART transmitter block diagram is shown in Figure 11-1. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the Stop bit has been transmitted from the previous load. As soon as the Stop bit is transmitted, the TSR is loaded with new data from the TXREG register (if available). Once the TXREG register transfers the data to the TSR register (occurs in one TCY), the TXREG register is empty and flag bit, TXIF (PIR1<4>), is set. This interrupt can be enabled/disabled by setting/clearing enable bit, TXIE (PIE1<4>). Flag bit TXIF will be set, regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. Status bit TRMT is a read-only bit which is set when the TSR register is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty.

Note 1:	The TSR register is not mapped in data memory, so it is not available to the user.
2:	Flag bit TXIF is set when enable bit TXEN is set. TXIF is cleared by loading TXREG.

Transmission is enabled by setting enable bit TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data and the Baud Rate Generator (BRG) has produced a shift clock (Figure 11-2). The transmission can also be started by first loading the TXREG register and then setting enable bit TXEN. Normally, when transmission is first started, the TSR register is empty. At that point, transfer to the TXREG register will result in an immediate transfer to TSR, resulting in an empty TXREG. A back-to-back transfer is thus possible (Figure 11-3). Clearing enable bit TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. As a result, the RB5/$\overline{SS}$/TX/CK pin will revert to high-impedance.

In order to select 9-bit transmission, transmit bit, TX9 (TXSTA<6>), should be set and the ninth bit should be written to TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG register can result in an immediate transfer of the data to the TSR register (if the TSR is empty). In such a case, an incorrect ninth data bit may be loaded in the TSR register.

FIGURE 11-1: AUSART TRANSMIT BLOCK DIAGRAM



When setting up an asynchronous transmission, follow these steps:

1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH (**Section 11.1 "AUSART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, then set enable bit TXIE.
4. If 9-bit transmission is desired, then set transmit bit TX9.
5. Enable the transmission by setting bit TXEN which will also set bit TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREG register (starts transmission).
8. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

FIGURE 11-2: ASYNCHRONOUS MASTER TRANSMISSION

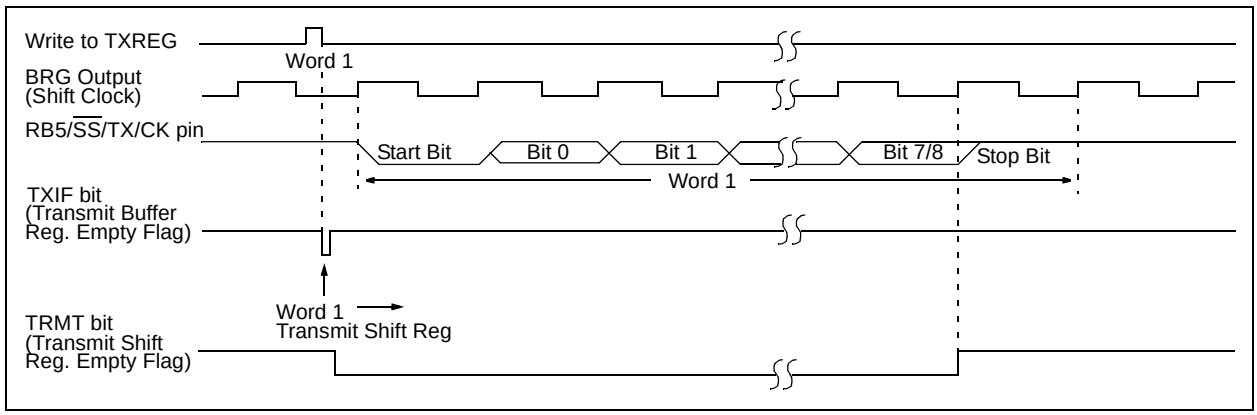
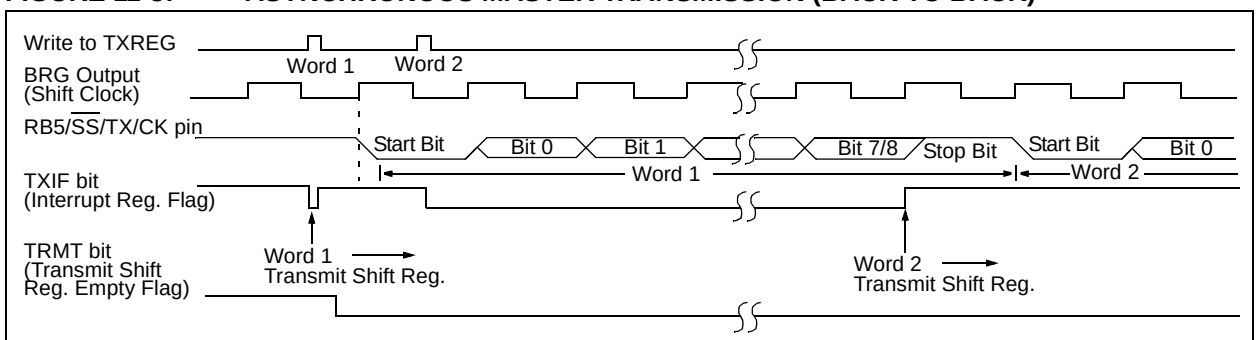


FIGURE 11-3: ASYNCHRONOUS MASTER TRANSMISSION (BACK TO BACK)



Note: This timing diagram shows two consecutive transmissions.

TABLE 11-7: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
19h	TXREG	AUSART Transmit Data Register								0000 0000	0000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.

Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

11.2.2 AUSART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 11-4. The data is received on the RB2/SDO/RX/DT pin and drives the data recovery block. The data recovery block is actually a high-speed shifter, operating at x16 times the baud rate; whereas, the main receive serial shifter operates at the bit rate or at FOSC.

Once Asynchronous mode is selected, reception is enabled by setting bit CREN (RCSTA<4>).

The heart of the receiver is the Receive (Serial) Shift Register (RSR). After sampling the Stop bit, the received data in the RSR is transferred to the RCREG register (if it is empty). If the transfer is complete, flag bit, RCIF (PIR1<5>), is set. The actual interrupt can be enabled/disabled by setting/clearing enable bit RCIE (PIE1<5>). Flag bit RCIF is a read-only bit which is cleared by the hardware. It is cleared when the RCREG register has been read and is empty. The RCREG is a double-buffered register (i.e., it is a two-deep FIFO). It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte to begin shifting to the RSR register. On the detection of the Stop bit of the third byte, if the RCREG register is still full, the Overrun Error bit, OERR (RCSTA<1>), will be set. The word in the RSR will be lost. The RCREG register can be read twice to retrieve the two bytes in the FIFO. Overrun bit OERR has to be cleared in software. This is done by resetting the receive logic (CREN is cleared and then set). If bit OERR is set, transfers from the RSR register to the RCREG register are inhibited and no further data will be received. It is, therefore, essential to clear error bit OERR if it is set. Framing Error bit, FERR (RCSTA<2>), is set if a Stop bit is detected as clear. Bit FERR and the 9th receive bit are buffered the same way as the receive data. Reading the RCREG will load bits RX9D and FERR with new values; therefore, it is essential for the user to read the RCSTA register, before reading the RCREG register, in order not to lose the old FERR and RX9D information.

FIGURE 11-4: AUSART RECEIVE BLOCK DIAGRAM

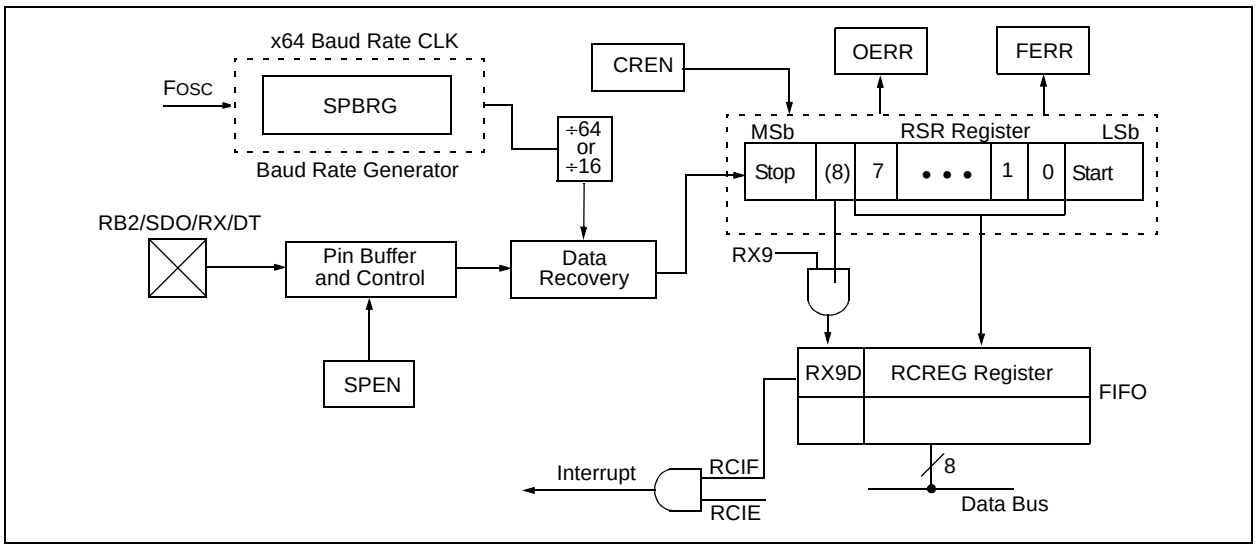
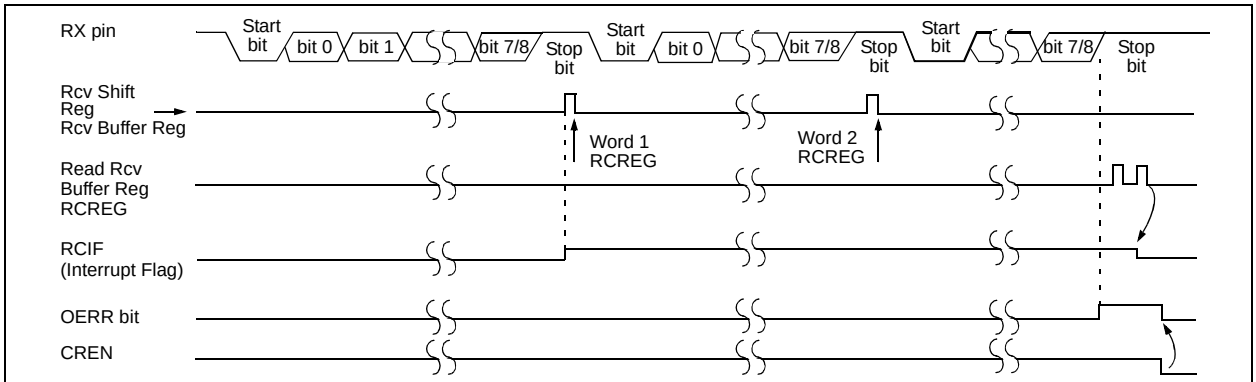


FIGURE 11-5: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RX input. The RCREG (Receive Buffer) is read after the third word, causing the OERR (Overrun) bit to be set.

When setting up an asynchronous reception, follow these steps:

1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH (**Section 11.1 "AUSART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, then set enable bit RCIE.
4. If 9-bit reception is desired, then set bit RX9.
5. Enable the reception by setting bit CREN.
6. Flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE is set.
7. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG register.
9. If any error occurred, clear the error by clearing enable bit CREN.
10. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

TABLE 11-8: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
1Ah	RCREG	AUSART Receive Data Register								0000 0000	0000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

11.2.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

When setting up an asynchronous reception with address detect enabled:

- Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit BRGH.
- Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
- If interrupts are desired, then set enable bit RCIE.
- Set bit RX9 to enable 9-bit reception.
- Set ADDEN to enable address detect.
- Enable the reception by setting enable bit CREN.
- Flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE was set.
- Read the RCSTA register to get the ninth bit and determine if any error occurred during reception.
- Read the 8-bit received data by reading the RCREG register to determine if the device is being addressed.
- If any error occurred, clear the error by clearing enable bit CREN.
- If the device has been addressed, clear the ADDEN bit to allow data bytes and address bytes to be read into the receive buffer and interrupt the CPU.

FIGURE 11-6: AUSART RECEIVE BLOCK DIAGRAM

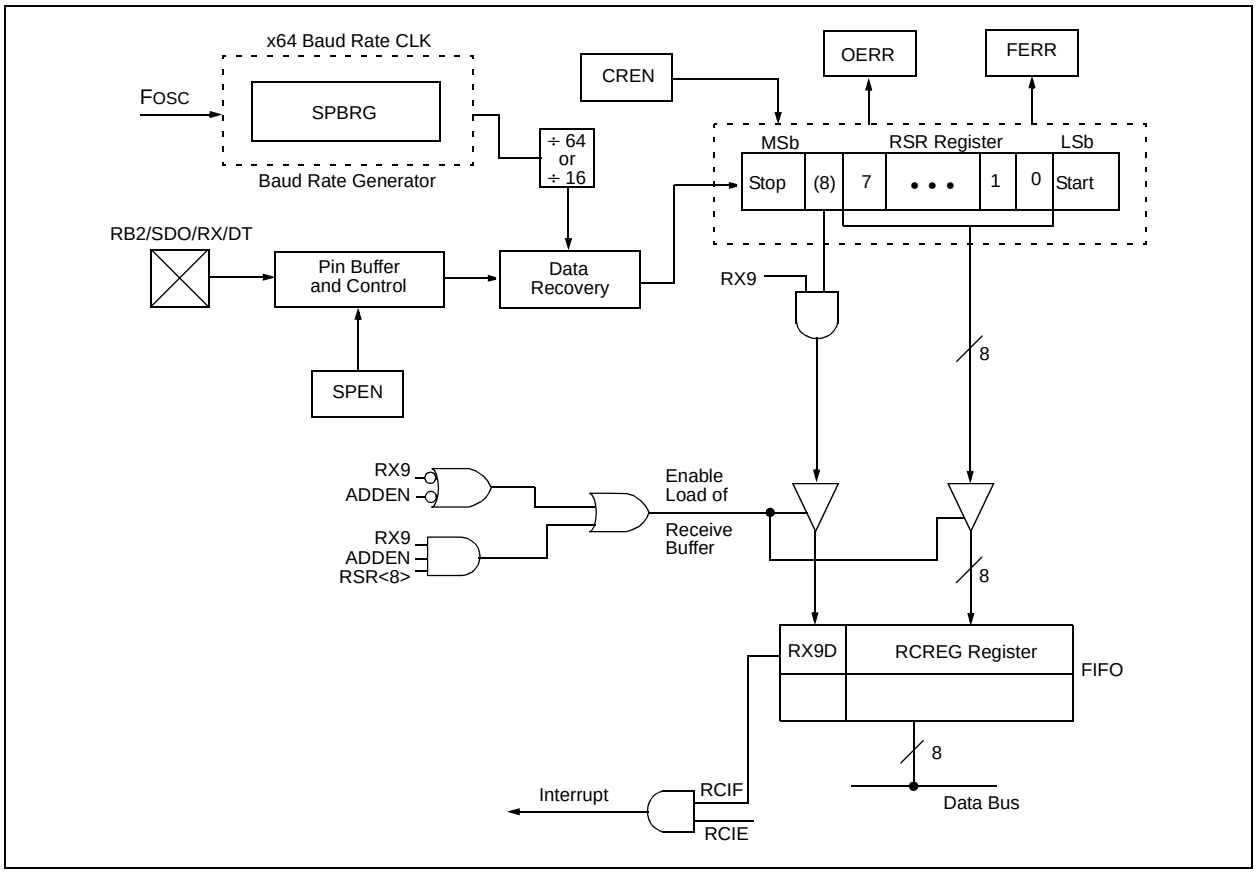
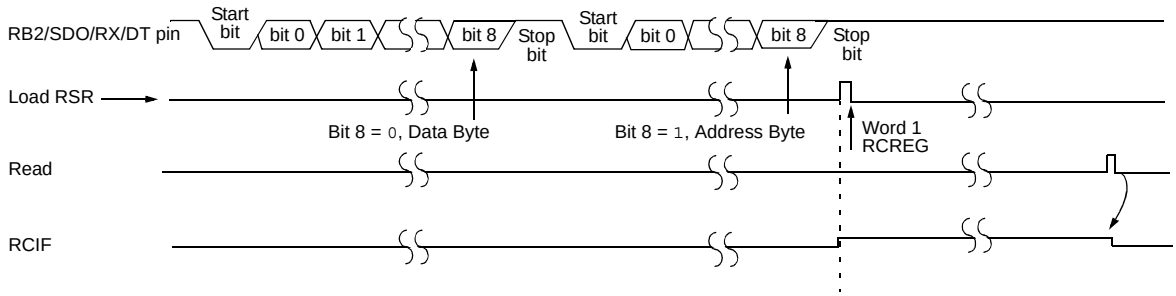
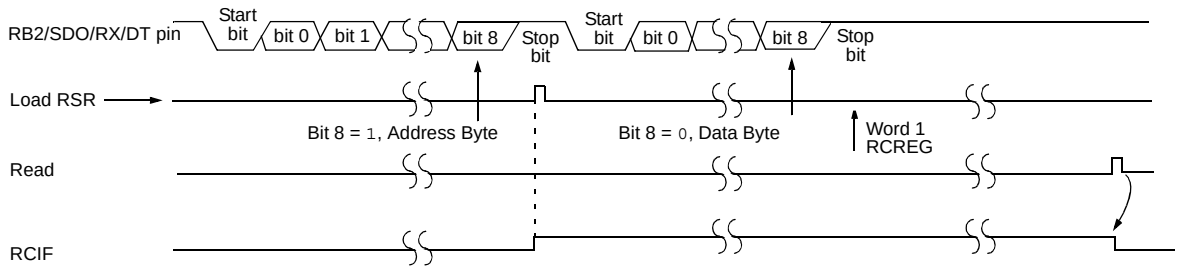


FIGURE 11-7: ASYNCHRONOUS RECEPTION WITH ADDRESS DETECT



Note: This timing diagram shows a data byte followed by an address byte. The data byte is not read into the RCREG (Receive Buffer) because ADDEN = 1.

FIGURE 11-8: ASYNCHRONOUS RECEPTION WITH ADDRESS BYTE FIRST



Note: This timing diagram shows a data byte followed by an address byte. The data byte is not read into the RCREG (Receive Buffer) because ADDEN was not updated and still = 0.

TABLE 11-9: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
1Ah	RCREG	AUSART Receive Data Register								0000 0000	0000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.

Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

11.3 AUSART Synchronous Master Mode

In Synchronous Master mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit SYNC (TXSTA<4>). In addition, enable bit SPEN (RCSTA<7>) is set in order to configure the RB5/$\overline{SS}$/TX/CK and RB2/SDO/RX/DT I/O pins to CK (clock) and DT (data) lines, respectively. The Master mode indicates that the processor transmits the master clock on the CK line. The Master mode is entered by setting bit CSRC (TXSTA<7>).

11.3.1 AUSART SYNCHRONOUS MASTER TRANSMISSION

The AUSART transmitter block diagram is shown in Figure 11-6. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREG (if available). Once the TXREG register transfers the data to the TSR register (occurs in one TCYCLE), the TXREG is empty and interrupt bit TXIF (PIR1<4>) is set. The interrupt can be enabled/disabled by setting/clearing enable bit TXIE (PIE1<4>). Flag bit TXIF will be set, regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory, so it is not available to the user.

Transmission is enabled by setting enable bit TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data. The first data bit will be shifted out on the next available rising edge of the clock on the CK line. Data out is stable around the falling edge of the synchronous clock (Figure 11-9). The transmission can also be started by first loading the TXREG register and then setting bit TXEN (Figure 11-10). This is advantageous when slow baud rates are selected, since the BRG is kept in Reset when bits TXEN, CREN and SREN are clear. Setting enable bit TXEN will start the BRG, creating a shift clock immediately. Normally, when transmission is first started, the TSR register is empty, so a transfer to the TXREG register will result in an immediate transfer to TSR, resulting in an empty TXREG. Back-to-back transfers are possible.

Clearing enable bit TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. The DT and CK pins will revert to high-impedance. If either bit CREN or bit SREN is set during a transmission, the transmission is aborted and the DT pin reverts to a high-impedance state (for a reception). The CK pin will remain an output if bit CSRC is set (internal clock). The transmitter logic, however, is not reset, although it is disconnected from the pins. In order to reset the transmitter, the user has to clear bit TXEN. If bit SREN is set (to interrupt an on-going transmission and receive a single word), then after the single word is received, bit SREN will be cleared and the serial port will revert back to transmitting, since bit TXEN is still set. The DT line will immediately switch from High-Impedance Receive mode to transmit and start driving. To avoid this, bit TXEN should be cleared.

In order to select 9-bit transmission, the TX9 (TXSTA<6>) bit should be set and the ninth bit should be written to bit TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG can result in an immediate transfer of the data to the TSR register (if the TSR is empty). If the TSR was empty and the TXREG was written before writing the “new” TX9D, the “present” value of bit TX9D is loaded.

Steps to follow when setting up a synchronous master transmission:

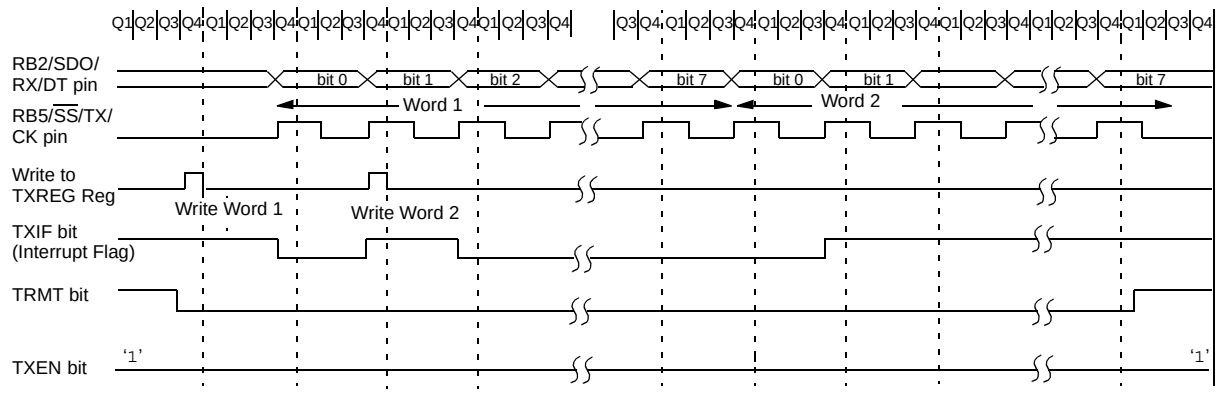
1. Initialize the SPBRG register for the appropriate baud rate (**Section 11.1 “AUSART Baud Rate Generator (BRG)”**).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

TABLE 11-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
19h	TXREG	AUSART Transmit Data Register								0000 0000	0000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

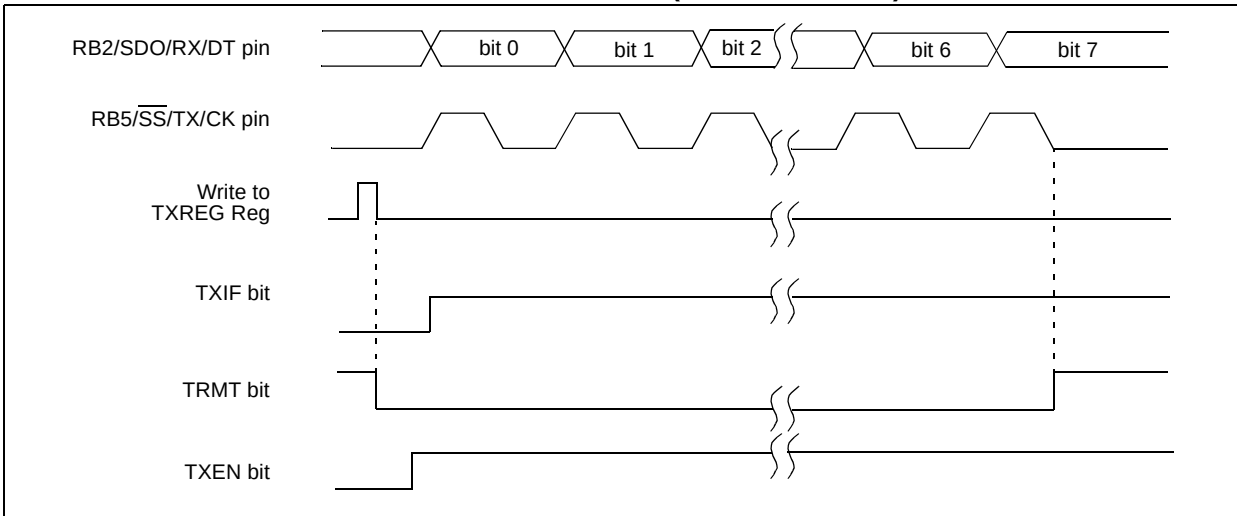
Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.
Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

FIGURE 11-9: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode; SPBRG = 0. Continuous transmission of two 8-bit words.

FIGURE 11-10: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)



11.3.2 AUSART SYNCHRONOUS MASTER RECEPTION

Once Synchronous mode is selected, reception is enabled by setting either enable bit SREN (RCSTA<5>), or enable bit CREN (RCSTA<4>). Data is sampled on the RB2/SDO/RX/DT pin on the falling edge of the clock. If enable bit SREN is set, then only a single word is received. If enable bit CREN is set, the reception is continuous until CREN is cleared. If both bits are set, CREN takes precedence.

After clocking the last bit, the received data in the Receive Shift Register (RSR) is transferred to the RCREG register (if it is empty). When the transfer is complete, interrupt flag bit, RCIF (PIR1<5>), is set. The actual interrupt can be enabled/disabled by setting/clearing enable bit RCIE (PIE1<5>).

Flag bit RCIF is a read-only bit which is reset by the hardware. In this case, it is reset when the RCREG register has been read and is empty. The RCREG is a double-buffered register (i.e., it is a two-deep FIFO). It is possible for two bytes of data to be received and transferred to the RCREG FIFO and a third byte to begin shifting into the RSR register. On the clocking of the last bit of the third byte, if the RCREG register is still full, then Overrun Error bit, OERR (RCSTA<1>), is set. The word in the RSR will be lost. The RCREG register can be read twice to retrieve the two bytes in the FIFO. Bit OERR has to be cleared in software (by clearing bit CREN). If bit OERR is set, transfers from the RSR to the RCREG are inhibited, so it is essential to clear bit OERR if it is set. The ninth receive bit is buffered the same way as the receive data. Reading the RCREG register will load bit RX9D with a new value, therefore, it is essential for the user to read the RCSTA register, before reading RCREG, in order not to lose the old RX9D information.

When setting up a synchronous master reception:

1. Initialize the SPBRG register for the appropriate baud rate (**Section 11.1 “AUSART Baud Rate Generator (BRG)”**).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. Ensure bits CREN and SREN are clear.
4. If interrupts are desired, then set enable bit RCIE.
5. If 9-bit reception is desired, then set bit RX9.
6. If a single reception is required, set bit SREN. For continuous reception, set bit CREN.
7. Interrupt flag bit, RCIF, will be set when reception is complete and an interrupt will be generated if enable bit, RCIE, was set.
8. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREG register.
10. If any error occurred, clear the error by clearing bit CREN.
11. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

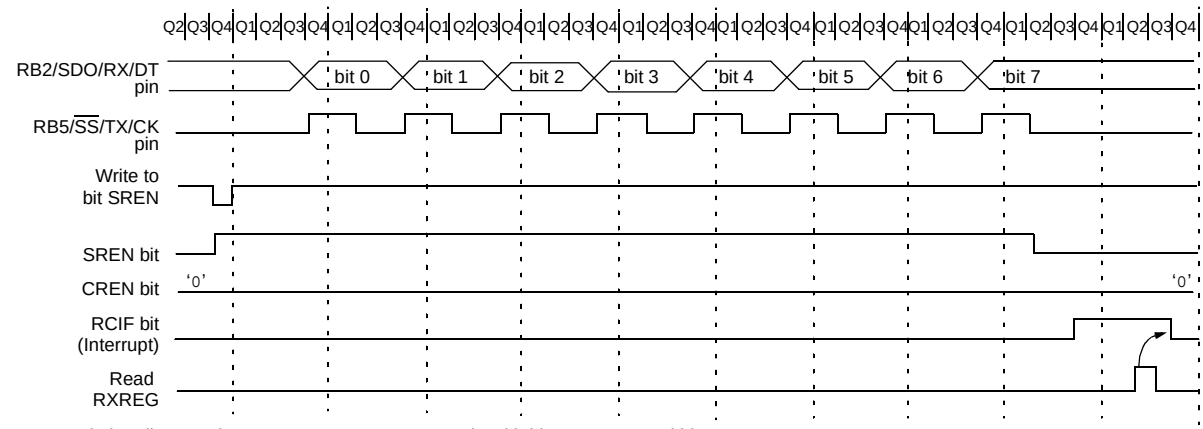
TABLE 11-11: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
1Ah	RCREG	AUSART Receive Data Register								0000 0000	0000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as ‘0’. Shaded cells are not used for synchronous master reception.

Note 1: This bit is only implemented on the PIC16F88. The bit will read ‘0’ on the PIC16F87.

FIGURE 11-11: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)



Note: Timing diagram demonstrates Sync Master mode with bit SREN = 1 and bit BRG = 0.

11.4 AUSART Synchronous Slave Mode

Synchronous Slave mode differs from the Master mode in the fact that the shift clock is supplied externally at the RB5/$\overline{SS}$/TX/CK pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in Sleep mode. Slave mode is entered by clearing bit CSRC (TXSTA<7>).

11.4.1 AUSART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes is identical, except in the case of the Sleep mode.

If two words are written to the TXREG and then the SLEEP instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in the TXREG register.
c) Flag bit TXIF will not be set.
d) When the first word has been shifted out of TSR, the TXREG register will transfer the second word to the TSR and flag bit TXIF will now be set.
e) If enable bit TXIE is set, the interrupt will wake the chip from Sleep and if the global interrupt is enabled, the program will branch to the interrupt vector (0004h).

When setting up a synchronous slave transmission, follow these steps:

1. Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. Clear bits CREN and SREN.
3. If interrupts are desired, then set enable bit TXIE.
4. If 9-bit transmission is desired, then set bit TX9.
5. Enable the transmission by setting enable bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

TABLE 11-12: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
19h	TXREG	AUSART Transmit Data Register								0000 0000	0000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.
Note 1: This bit is only implemented on the PIC16F88. The bit will read '0' on the PIC16F87.

11.4.2 AUSART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of the Sleep mode. Bit SREN is a “don’t care” in Slave mode.

If receive is enabled by setting bit CREN prior to the `SLEEP` instruction, then a word may be received during Sleep. On completely receiving the word, the RSR register will transfer the data to the RCREG register and if enable bit RCIE bit is set, the interrupt generated will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector (0004h).

When setting up a synchronous slave reception, follow these steps:

1. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. If interrupts are desired, set enable bit RCIE.
3. If 9-bit reception is desired, set bit RX9.
4. To enable reception, set enable bit CREN.
5. Flag bit RCIF will be set when reception is complete and an interrupt will be generated if enable bit RCIE was set.
6. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG register.
8. If any error occurred, clear the error by clearing bit CREN.
9. If using interrupts, ensure that GIE and PEIE (bits 7 and 6) of the INTCON register are set.

TABLE 11-13: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
0Bh, 8Bh, 10Bh,18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	`0000 000x`	`0000 000u`
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	`-000 0000`	`-000 0000`
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	`0000 000x`	`0000 000x`
1Ah	RCREG	AUSART Receive Data Register								`0000 0000`	`0000 0000`
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	`-000 0000`	`-000 0000`
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	`0000 -010`	`0000 -010`
99h	SPBRG	Baud Rate Generator Register								`0000 0000`	`0000 0000`

Legend: `x` = unknown, - = unimplemented, read as ‘`0`’. Shaded cells are not used for synchronous slave reception.

Note 1: This bit is only implemented on the PIC16F88. The bit will read ‘`0`’ on the PIC16F87.

12.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) converter module has seven inputs for 18/20 pin devices (PIC16F88 devices only).

The conversion of an analog input signal results in a corresponding 10-bit digital number. The A/D module has a high and low-voltage reference input that is software selectable to some combination of VDD, VSS, VREF- (RA2) or VREF+ (RA3).

The A/D converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D conversion clock must be derived from the A/D's internal RC oscillator.

The A/D module has five registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- Analog Select Register (ANSEL)

The ADCON0 register, shown in Register 12-2, controls the operation of the A/D module. The ANSEL register, shown in Register 12-1 and the ADCON1 register, shown in Register 12-3, configure the functions of the port pins. The port pins can be configured as analog inputs (RA3/RA2 can also be voltage references) or as digital I/O.

Additional information on using the A/D module can be found in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023).

REGISTER 12-1: ANSEL: ANALOG SELECT REGISTER (ADDRESS 9Bh) PIC16F88 DEVICES ONLY

U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	ANS6	ANS5	ANS4	ANS3	ANS2	ANS1	ANS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6-0 **ANS<6:0>:** Analog Input Select bits

Bits select input function on corresponding AN<6:0> pins.

1 = Analog I/O(1,2)

0 = Digital I/O

Note 1: Setting a pin to an analog input disables the digital input buffer. The corresponding TRIS bit should be set to input mode when using pins as analog inputs. Only AN2 is an analog I/O, all other ANx pins are analog inputs.

2: See the block diagrams for the analog I/O pins to see how ANSEL interacts with the CHS bits of the ADCON0 register.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 12-2: ADCON0: A/D CONTROL REGISTER (ADDRESS 1Fh) PIC16F88 DEVICES ONLY

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7-6 **ADCS<1:0>:** A/D Conversion Clock Select bits

If ADCS2 = 0:
00 = Fosc/2
01 = Fosc/8
10 = Fosc/32
11 = FRC (clock derived from the internal A/D module RC oscillator)

If ADCS2 = 1:
00 = Fosc/4
01 = Fosc/16
10 = Fosc/64
11 = FRC (clock derived from the internal A/D module RC oscillator)

bit 5-3 **CHS<2:0>:** Analog Channel Select bits

000 = Channel 0 (RA0/AN0)
001 = Channel 1 (RA1/AN1)
010 = Channel 2 (RA2/AN2)
011 = Channel 3 (RA3/AN3)
100 = Channel 4 (RA4/AN4)
101 = Channel 5 (RB6/AN5)
110 = Channel 6 (RB7/AN6)

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit

If ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion)
0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

bit 1 **Unimplemented:** Read as '0'

bit 0 **ADON:** A/D On bit

1 = A/D converter module is operating
0 = A/D converter module is shut off and consumes no operating current

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 12-3: ADCON1: A/D CONTROL REGISTER 1 (ADDRESS 9Fh) PIC16F88 DEVICES ONLY

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
ADFM	ADCS2	VCFG1	VCFG0	—	—	—	—
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit

1 = Right justified. Six Most Significant bits of ADRESH are read as '0'.
0 = Left justified. Six Least Significant bits of ADRESL are read as '0'.

bit 6 **ADCS2:** A/D Clock Divide by 2 Select bit

1 = A/D clock source is divided by 2 when system clock is used
0 = Disabled

bit 5-4 **VCFG<1:0>:** A/D Voltage Reference Configuration bits

Logic State	V_{REF+}	V_{REF-}
00	AV_{DD}	AV_{SS}
01	AV_{DD}	V_{REF-}
10	V_{REF+}	AV_{SS}
11	V_{REF+}	V_{REF-}

Note: The ANSEL bits for AN3 and AN2 inputs must be configured as analog inputs for the V_{REF+} and V_{REF-} external pins to be used.

bit 3-0 **Unimplemented:** Read as '0'

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

The ADRESH:ADRESL registers contain the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the A/D Result register pair, the GO/$\overline{\text{DONE}}$ bit (ADCON0<2>) is cleared and A/D Interrupt Flag bit, ADIF, is set. The block diagram of the A/D module is shown in Figure 12-1.

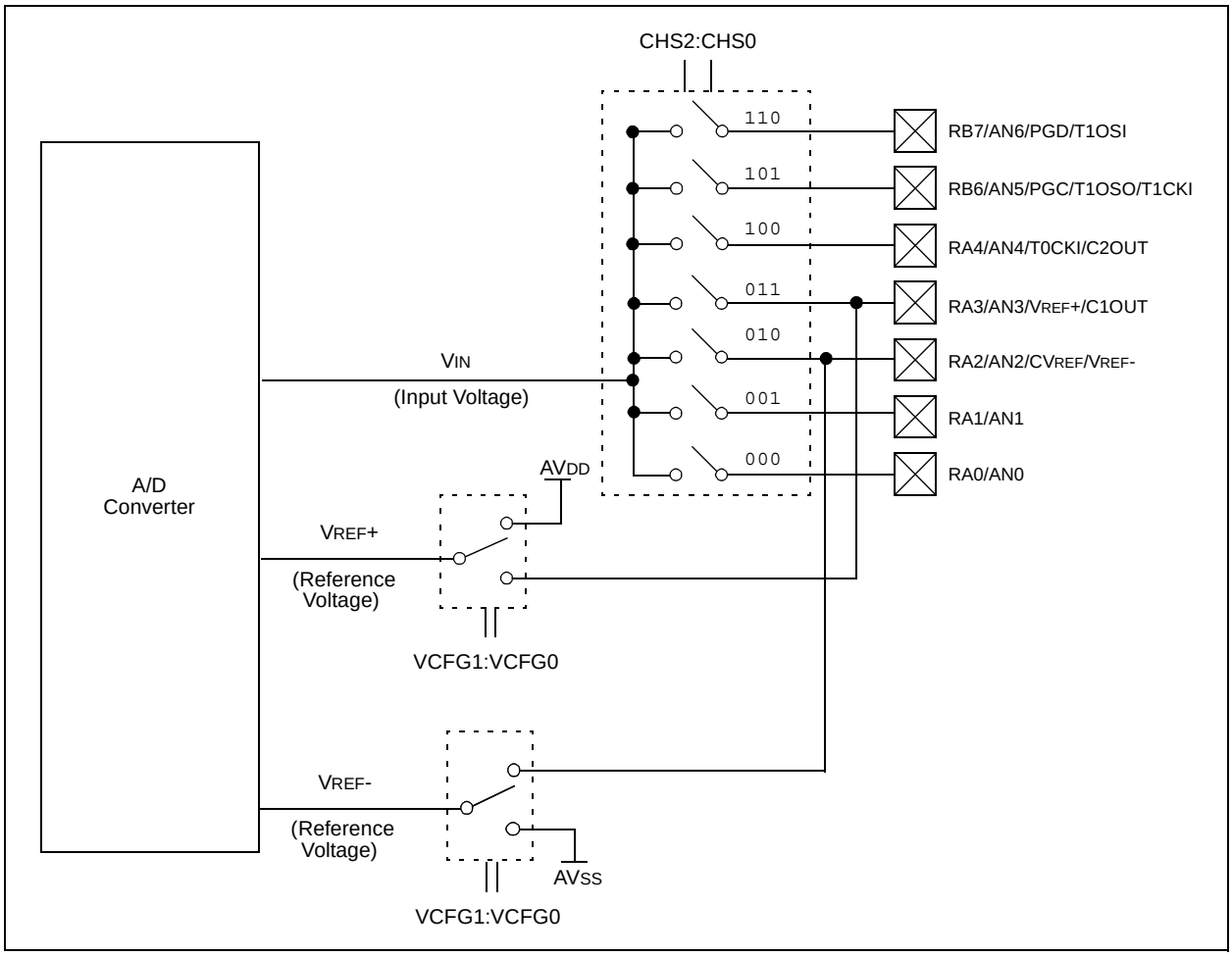
After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as inputs.

To determine sample time, see **Section 12.1 "A/D Acquisition Requirements"**. After this sample time has elapsed, the A/D conversion can be started.

These steps should be followed for doing an A/D conversion:

1. Configure the A/D module:
   - Configure analog/digital I/O (ANSEL)
   - Configure voltage reference (ADCON1)
   - Select A/D input channel (ADCON0)
   - Select A/D conversion clock (ADCON0)
   - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
   - Clear ADIF bit
   - Set ADIE bit
   - SET PEIE bit
   - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion:
   - Set GO/$\overline{\text{DONE}}$ bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
   - Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared (with interrupts disabled); OR
   - Waiting for the A/D interrupt
6. Read A/D Result register pair (ADRESH:ADRESL), clear bit ADIF if required.
7. For next conversion, go to step 1 or step 2 as required. The A/D conversion time per bit is defined as T_{AD}. A minimum wait of 2 T_{AD} is required before the next acquisition starts.

FIGURE 12-1: A/D BLOCK DIAGRAM



12.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 12-2. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD), see Figure 12-2. **The maximum recommended impedance for analog sources is 10 kΩ.** As the impedance is decreased, the acquisition time may be decreased. After the analog input channel is selected (changed), this acquisition must be done before the conversion can be started.

To calculate the minimum acquisition time, Equation 12-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

To calculate the minimum acquisition time, TACQ, see the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023).

EQUATION 12-1: ACQUISITION TIME

TACQ	=	Amplifier Settling Time + Hold Capacitor Charging Time + Temperature Coefficient
	=	TAMP + TC + TCOFF
	=	2 µs + TC + [(Temperature -25°C)(0.05 µs/°C)]
TC	=	CHOLD (RIC + RSS + RS) In(1/2047)
	=	-120 pF (1 kΩ + 7 kΩ + 10 kΩ) In(0.0004885)
	=	16.47 µs
TACQ	=	2 µs + 16.47 µs + [(50°C – 25°C)(0.05 µs/°C)
	=	19.72 µs

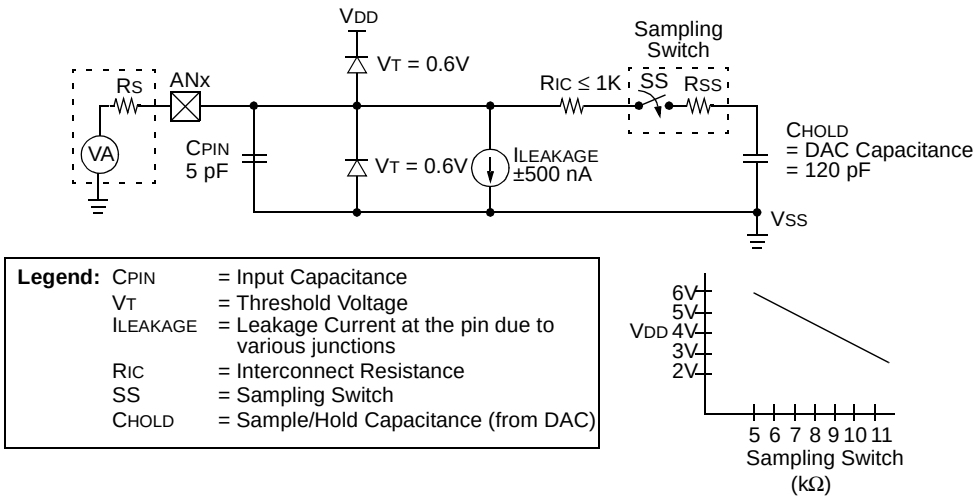
Note 1: The reference voltage (VREF) has no effect on the equation, since it cancels itself out.

2: The charge holding capacitor (CHOLD) is not discharged after each conversion.

3: The maximum recommended impedance for analog sources is 10 kΩ. This is required to meet the pin leakage specification.

4: After a conversion has completed, a 2.0 TAD delay must complete before acquisition can begin again. During this time, the holding capacitor is not connected to the selected A/D input channel.

FIGURE 12-2: ANALOG INPUT MODEL



12.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 9.0 TAD per 10-bit conversion. The source of the A/D conversion clock is software selectable. The seven possible options for TAD are:

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- Internal A/D module RC oscillator (2-6 µs)

For correct A/D conversions, the A/D conversion clock (TAD) must be selected to ensure a minimum TAD time as small as possible, but no less than 1.6 µs and not greater than 6.4 µs.

Table 12-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

12.3 Operation in Power-Managed Modes

The selection of the automatic acquisition time and A/D conversion clock is determined in part by the clock source and frequency while in a power-managed mode.

If the A/D is expected to operate while the device is in a power-managed mode, the ADCS2:ADCS0 bits in ADCON0 and ADCON1 should be updated in accordance with the power-managed mode clock that will be used. After the power-managed mode is entered (either of the power-managed Run modes), an A/D acquisition or conversion may be started. Once an acquisition or conversion is started, the device should continue to be clocked by the same power-managed mode clock source until the conversion has been completed.

If the power-managed mode clock frequency is less than 1 MHz, the A/D RC clock source should be selected.

TABLE 12-1: TAD VS. MAXIMUM DEVICE OPERATING FREQUENCIES – STANDARD DEVICES (C)

AD Clock Source (TAD)			Maximum Device Frequency
Operation	**ADCS<2>**	**ADCS<1:0>**	**Max.**
2 TOSC	0	00	1.25 MHz
4 TOSC	1	00	2.5 MHz
8 TOSC	0	01	5 MHz
16 TOSC	1	01	10 MHz
32 TOSC	0	10	20 MHz
64 TOSC	1	10	20 MHz
RC(1,2,3)	x	11	**(Note 1)**

Note 1: The RC source has a typical TAD time of 4 µs, but can vary between 2-6 µs.

2: When the device frequencies are greater than 1 MHz, the RC A/D conversion clock source is only recommended for Sleep operation.

3: For extended voltage devices (LF), please refer to **Section 18.0 "Electrical Characteristics"**.

12.4 Configuring Analog Port Pins

The ADCON1, ANSEL, TRISA and TRISB registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (V_{OH} or V_{OL}) will be converted.

The A/D operation is independent of the state of the CHS<2:0> bits and the TRIS bits.

Note 1:	When reading the Port register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert an analog input. Analog levels on a digitally configured input will not affect the conversion accuracy.
2:	Analog levels on any pin that is defined as a digital input (including the RA4:RA0 and RB7:RB6 pins), may cause the input buffer to consume current out of the device specification.

12.5 A/D Conversions

Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The A/D Result register pair will NOT be updated with the partially completed A/D conversion sample. That is, the ADRESH:ADRESL registers will continue to contain the value of the last completed conversion (or the last value written to the ADRESH:ADRESL registers). After the A/D conversion is aborted, a 2 T_{AD} wait is required before the next acquisition is started. After this 2 T_{AD} wait, acquisition on the selected channel is automatically started. The GO/$\overline{\text{DONE}}$ bit can then be set to start the conversion.

In Figure 12-3, after the GO/$\overline{\text{DONE}}$ bit is set, the first time segment has a minimum of T_{CY} and a maximum of T_{AD}.

Note:	The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D.

12.5.1 A/D RESULT REGISTERS

The ADRESH:ADRESL register pair is the location where the 10-bit A/D result is loaded at the completion of the A/D conversion. This register pair is 16 bits wide. The A/D module gives the flexibility to left or right justify the 10-bit result in the 16-bit result register. The A/D Format Select bit (ADFM) controls this justification. Figure 12-4 shows the operation of the A/D result justification. The extra bits are loaded with '0's. When an A/D result will not overwrite these locations (A/D disable), these registers may be used as two general purpose 8-bit registers.

FIGURE 12-3: A/D CONVERSION T_{AD} CYCLES

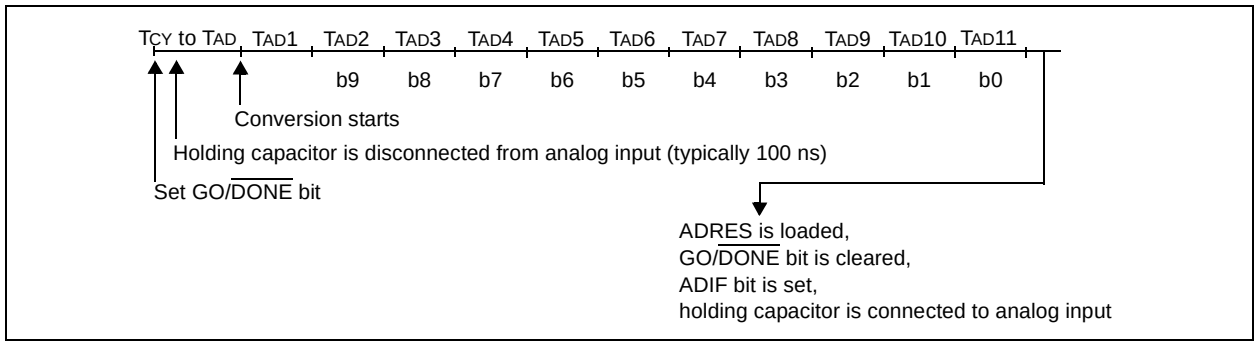
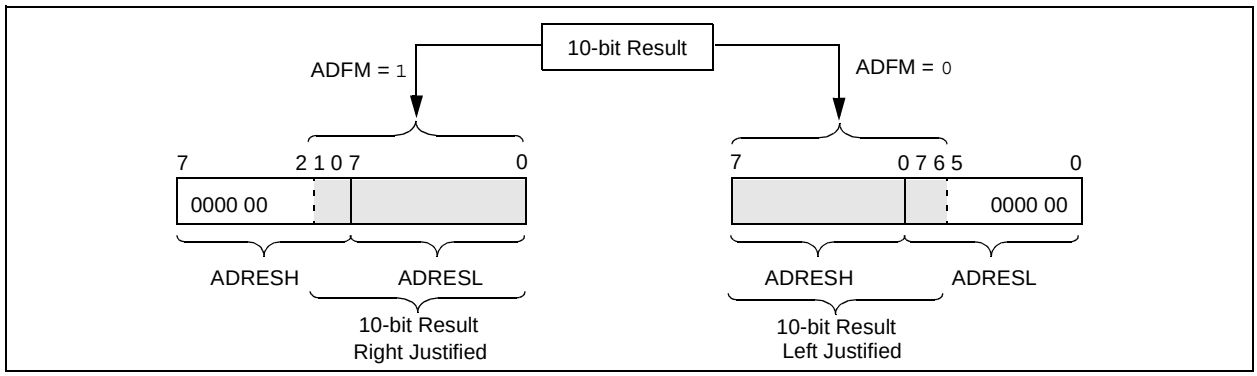


FIGURE 12-4: A/D RESULT JUSTIFICATION



12.6 A/D Operation During Sleep

The A/D module can operate during Sleep mode. This requires that the A/D clock source be set to RC (ADCS1:ADCS0 = 11). When the RC clock source is selected, the A/D module waits one instruction cycle before starting the conversion. This allows the SLEEP instruction to be executed which eliminates all digital switching noise from the conversion. When the conversion is completed, the GO/$\overline{\text{DONE}}$ bit will be cleared and the result loaded into the ADRES registers. If the A/D interrupt is enabled, the device will wake-up from Sleep. If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

When the A/D clock source is another clock option (not RC), a SLEEP instruction will cause the present conversion to be aborted and the A/D module to be turned off, though the ADON bit will remain set.

Turning off the A/D places the A/D module in its lowest current consumption state.

Note:	For the A/D module to operate in Sleep, the A/D clock source must be set to RC (ADCS1:ADCS0 = 11). To perform an A/D conversion in Sleep, ensure the SLEEP instruction immediately follows the instruction that sets the GO/$\overline{\text{DONE}}$ bit.

12.7 Effects of a Reset

A device Reset forces all registers to their Reset state. The A/D module is disabled and any conversion in progress is aborted. All A/D input pins are configured as analog inputs.

The value that is in the ADRESH:ADRESL registers is not modified for a Power-on Reset. The ADRESH:ADRESL registers will contain unknown data after a Power-on Reset.

12.8 Use of the CCP Trigger

An A/D conversion can be started by the “special event trigger” of the CCP module. This requires that the CCP1M3:CCP1M0 bits (CCP1CON<3:0>) be programmed as ‘1011’ and that the A/D module is enabled (ADON bit is set). When the trigger occurs, the GO/$\overline{\text{DONE}}$ bit will be set, starting the A/D conversion and the Timer1 counter will be reset to zero. Timer1 is reset to automatically repeat the A/D acquisition period with minimal software overhead (moving the ADRESH:ADRESL to the desired location). The appropriate analog input channel must be selected and the minimum acquisition done before the “special event trigger” sets the GO/$\overline{\text{DONE}}$ bit (starts a conversion).

If the A/D module is not enabled (ADON is cleared), then the “special event trigger” will be ignored by the A/D module, but will still reset the Timer1 counter.

TABLE 12-2: REGISTERS/BITS ASSOCIATED WITH A/D

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
0Bh, 8Bh 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF(1)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
8Ch	PIE1	—	ADIE(1)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
1Eh	ADRESH(2)	A/D Result Register High Byte								xxxx xxxx	uuuu uuuu
9Eh	ADRESL(2)	A/D Result Register Low Byte								xxxx xxxx	uuuu uuuu
1Fh	ADCON0(2)	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON	0000 00-0	0000 00-0
9Fh	ADCON1(2)	ADFM	ADCS2	VCFG1	VCFG0	—	—	—	—	0000 ----	0000 ----
9Bh	ANSEL(2	—	ANS6	ANS5	ANS4	ANS3	ANS2	ANS1	ANS0	-111 1111	-111 1111
05h	PORTA (PIC16F87) (PIC16F88)	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxx 0000 xxx0 0000	uuuu 0000 uuu0 0000
05h, 106h	PORTB (PIC16F87) (PIC16F88)	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx 00xx xxxx	uuuu uuuu 00uu uuuu
85h	TRISA	TRISA7	TRISA6	TRISA5(3)	PORTA Data Direction Register (TRISA<4:0>)					1111 1111	1111 1111
86h, 186h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented, read as ‘0’. Shaded cells are not used for A/D conversion.

Note 1: This bit is only implemented on the PIC16F88. The bit will read ‘0’ on the PIC16F87.

2: PIC16F88 only.

3: Pin 5 is an input only; the state of the TRISA5 bit has no effect and will always read ‘1’.

13.0 COMPARATOR MODULE

The comparator module contains two analog comparators. The inputs to the comparators are multiplexed with I/O port pins RA0 through RA3, while the outputs are multiplexed to pins RA3 and RA4. The on-chip Voltage Reference (**Section 14.0 "Comparator Voltage Reference Module"**) can also be an input to the comparators.

The CMCON register (Register 13-1) controls the comparator input and output multiplexors. A block diagram of the various comparator configurations is shown in Figure 13-1.

REGISTER 13-1: CMCON: COMPARATOR MODULE CONTROL REGISTER (ADDRESS 9Ch)

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

bit 7 **C2OUT:** Comparator 2 Output bit

When C2INV = 0:
1 = C2 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-

When C2INV = 1:
1 = C2 VIN+ < C2 VIN-
0 = C2 VIN+ > C2 VIN-

bit 6 **C1OUT:** Comparator 1 Output bit

When C1INV = 0:
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-

When C1INV = 1:
1 = C1 VIN+ < C1 VIN-
0 = C1 VIN+ > C1 VIN-

bit 5 **C2INV:** Comparator 2 Output Inversion bit

1 = C2 output inverted
0 = C2 output not inverted

bit 4 **C1INV:** Comparator 1 Output Inversion bit

1 = C1 output inverted
0 = C1 output not inverted

bit 3 **CIS:** Comparator Input Switch bit

When CM2:CM0 = 001:
1 = C1 VIN- connects to RA3
0 = C1 VIN- connects to RA0

When CM2:CM0 = 010:
1 = C1 VIN- connects to RA3
C2 VIN- connects to RA2
0 = C1 VIN- connects to RA0
C2 VIN- connects to RA1

bit 2-0 **CM<2:0>:** Comparator Mode bits

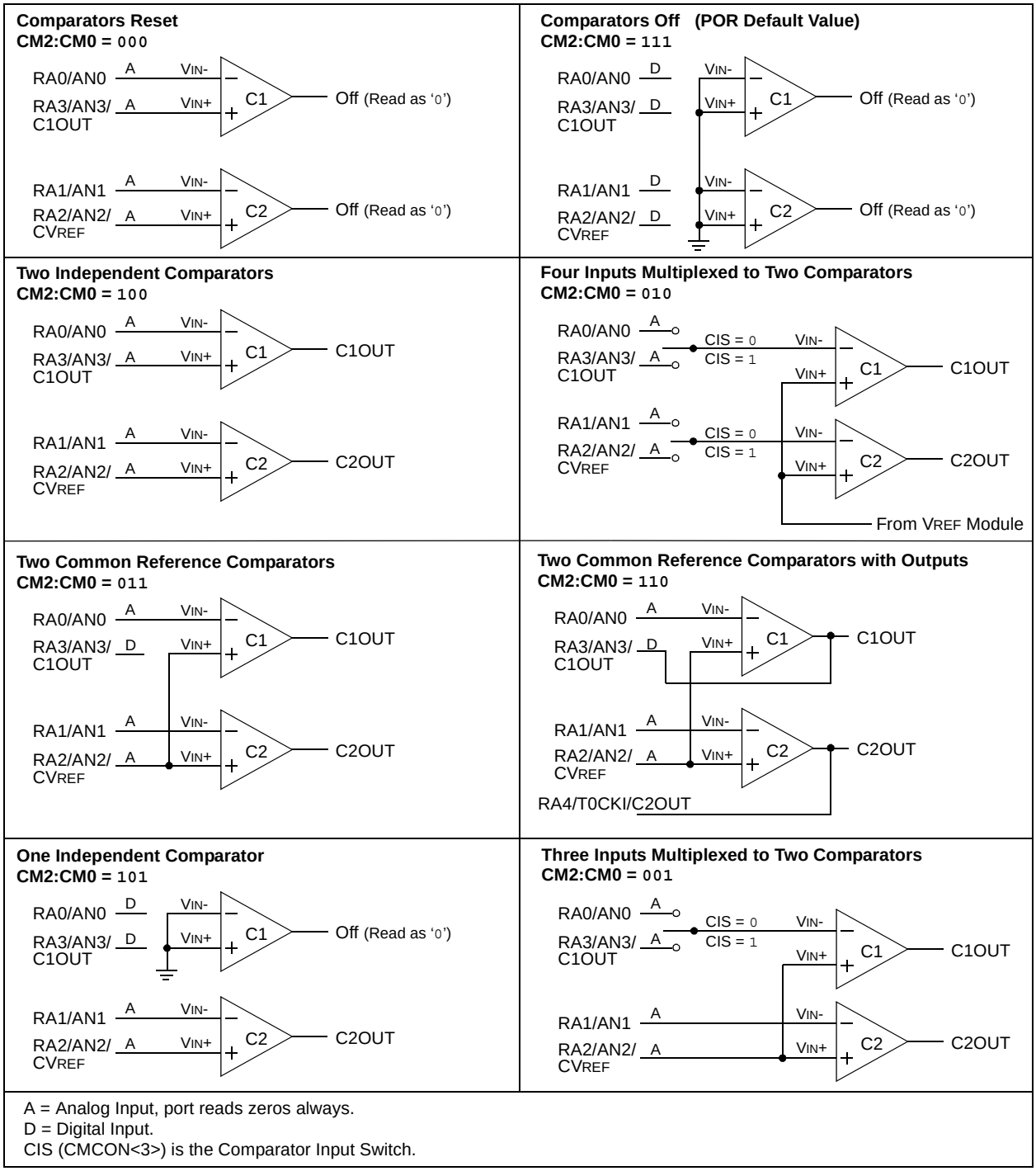
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

13.1 Comparator Configuration

There are eight modes of operation for the comparators. The CMCON register is used to select these modes. Figure 13-1 shows the eight possible modes. The TRISA register controls the data direction of the comparator pins for each mode. If the Comparator mode is changed, the comparator output level may not be valid for the specified mode change delay shown in **Section 18.0 "Electrical Characteristics"**.

> **Note:** Comparator interrupts should be disabled during a Comparator mode change; otherwise, a false interrupt may occur.

FIGURE 13-1: COMPARATOR I/O OPERATING MODES



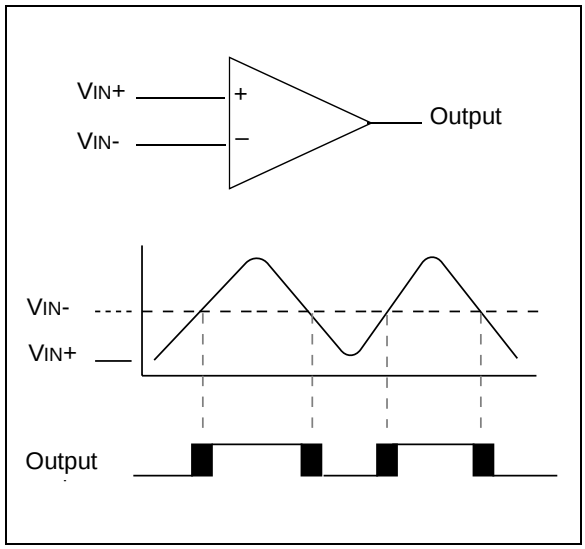
13.2 Comparator Operation

A single comparator is shown in Figure 13-2, along with the relationship between the analog input levels and the digital output. When the analog input at VIN+ is less than the analog input VIN-, the output of the comparator is a digital low level. When the analog input at VIN+ is greater than the analog input VIN-, the output of the comparator is a digital high level. The shaded areas of the output of the comparator in Figure 13-2 represent the uncertainty due to input offsets and response time.

13.3 Comparator Reference

An external or internal reference signal may be used depending on the comparator operating mode. The analog signal present at VIN- is compared to the signal at VIN+ and the digital output of the comparator is adjusted accordingly (Figure 13-2).

FIGURE 13-2: SINGLE COMPARATOR



13.3.1 EXTERNAL REFERENCE SIGNAL

When external voltage references are used, the comparator module can be configured to have the comparators operate from the same, or different reference sources. However, threshold detector applications may require the same reference. The reference signal must be between VSS and VDD and can be applied to either pin of the comparator(s).

13.3.2 INTERNAL REFERENCE SIGNAL

The comparator module also allows the selection of an internally generated voltage reference for the comparators. **Section 14.0 "Comparator Voltage Reference Module"** contains a detailed description of the Comparator Voltage Reference module that provides this signal. The internal reference signal is used when comparators are in mode CM<2:0> = `010` (Figure 13-1). In this mode, the internal voltage reference is applied to the VIN+ pin of both comparators.

13.4 Comparator Response Time

Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output has a valid level. If the internal reference is changed, the maximum delay of the internal voltage reference must be considered when using the comparator outputs. Otherwise, the maximum delay of the comparators should be used (**Section 18.0 "Electrical Characteristics"**).

13.5 Comparator Outputs

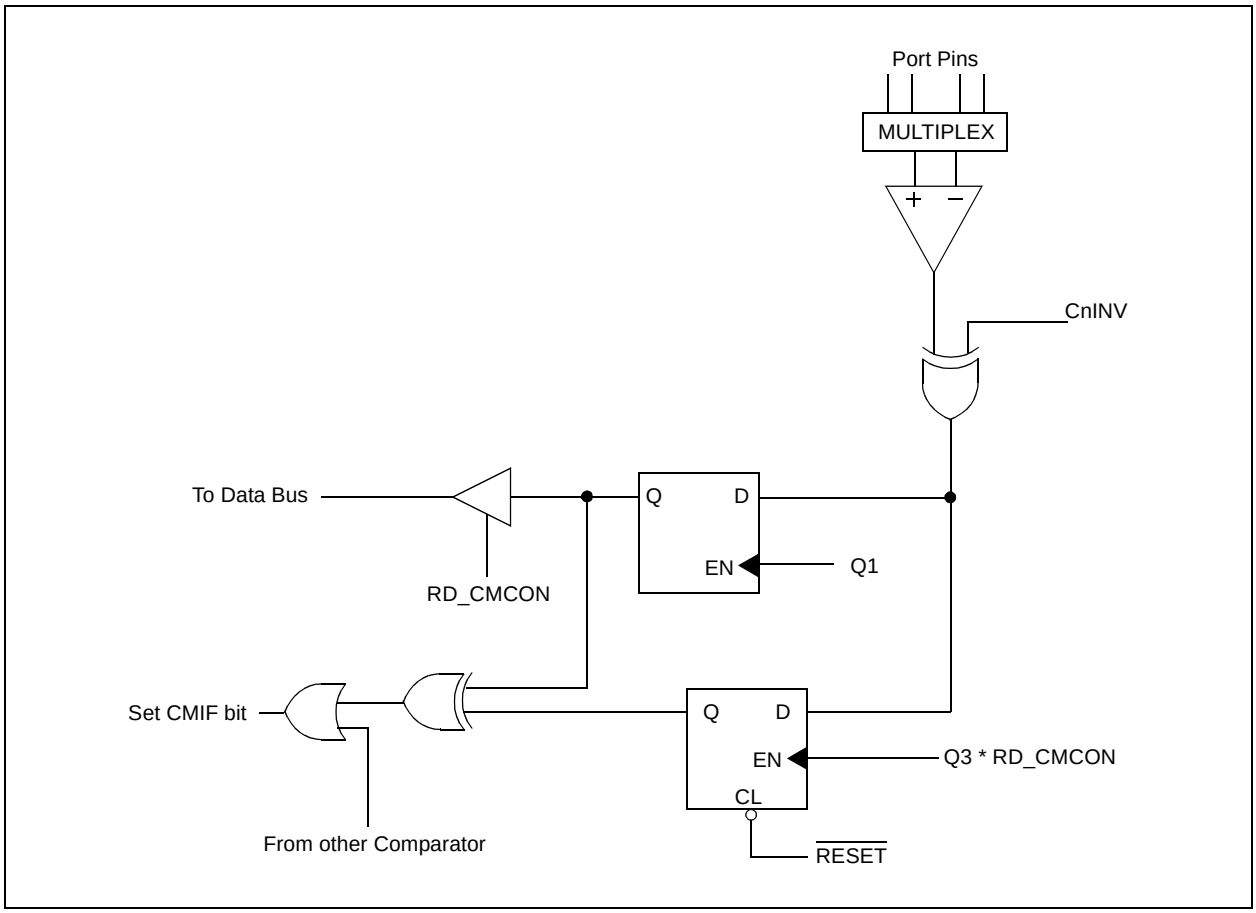
The comparator outputs are read through the CMCON register. These bits are read-only. The comparator outputs may also be directly output to the RA3 and RA4 I/O pins. When enabled, multiplexors in the output path of the RA3 and RA4 pins will switch and the output of each pin will be the unsynchronized output of the comparator. The uncertainty of each of the comparators is related to the input offset voltage and the response time given in the specifications. Figure 13-3 shows the comparator output block diagram.

The TRISA bits will still function as an output enable/disable for the RA3 and RA4 pins while in this mode.

The polarity of the comparator outputs can be changed using the C2INV and C1INV bits (CMCON<5:4>).

Note 1:	When reading the Port register, all pins configured as analog inputs will read as '`0`'. Pins configured as digital inputs will convert an analog input, according to the Schmitt Trigger input specification.
2:	Analog levels, on any pin defined as a digital input, may cause the input buffer to consume more current than is specified.

FIGURE 13-3: COMPARATOR OUTPUT BLOCK DIAGRAM



13.6 Comparator Interrupts

The comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMCON<7:6>, to determine the actual change that occurred. The CMIF bit (PIR2 register) is the Comparator Interrupt Flag. The CMIF bit must be reset by clearing it ('0'). Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated.

The CMIE bit (PIE2 register) and the PEIE bit (INTCON register) must be set to enable the interrupt. In addition, the GIE bit must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMIF bit will still be set if an interrupt condition occurs.

Note:	If a change in the CMCON register (C1OUT or C2OUT) should occur when a read operation is being executed (start of the Q2 cycle), then the CMIF (PIR2 register) interrupt flag may not get set.

The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of CMCON will end the mismatch condition.

b) Clear flag bit CMIF.

A mismatch condition will continue to set flag bit CMIF. Reading CMCON will end the mismatch condition and allow flag bit CMIF to be cleared.

13.7 Comparator Operation During Sleep

When a comparator is active and the device is placed in Sleep mode, the comparator remains active and the interrupt is functional, if enabled. This interrupt will wake-up the device from Sleep mode when enabled. While the comparator is powered up, higher Sleep currents than shown in the power-down current specification will occur. Each operational comparator will consume additional current, as shown in the comparator specifications. To minimize power consumption while in Sleep mode, turn off the comparators, CM<2:0> = `111`, before entering Sleep. If the device wakes up from Sleep, the contents of the CMCON register are not affected.

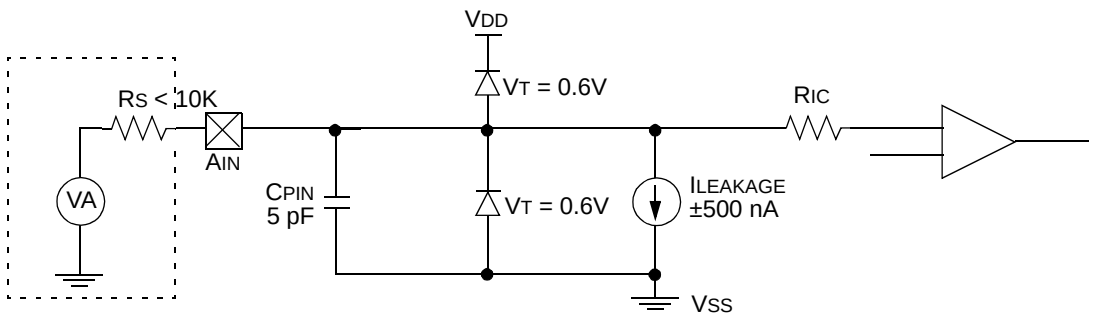
13.8 Effects of a Reset

A device Reset forces the CMCON register to its Reset state, causing the comparator module to be in the Comparator Off mode, CM<2:0> = `111`.

13.9 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 13-4. Since the analog pins are connected to a digital output, they have reverse biased diodes to V_{DD} and V_{SS}. The analog input, therefore, must be between V_{SS} and V_{DD}. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up condition may occur. A maximum source impedance of 10 kΩ is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

FIGURE 13-4: ANALOG INPUT MODEL



Legend:

C_{PIN}	=	Input Capacitance
V_T	=	Threshold Voltage
$I_{LEAKAGE}$	=	Leakage Current at the pin due to various junctions
R_{IC}	=	Interconnect Resistance
R_S	=	Source Impedance
VA	=	Analog Voltage

TABLE 13-1: REGISTERS ASSOCIATED WITH THE COMPARATOR MODULE

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
9Ch	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	0000 0111
9Dh	CVRCON	CVREN	CVROE	CVRR	—	CVR3	CVR2	CVR1	CVR0	000- 0000	000- 0000
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
0Dh	PIR2	OSFIF	CMIF	—	EEIF	—	—	—	—	00-0 ----	00-0 ----
8Dh	PIE2	OSFIE	CMIE	—	EEIE	—	—	—	—	00-0 ----	00-0 ----
05h	PORTA (PIC16F87) (PIC16F88)	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	 xxxx 0000 xxx0 0000	 uuuu 0000 uuu0 0000
85h	TRISA	TRISA7	TRISA6	TRISA5(1)	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by the comparator module.

Note 1: Pin 5 is an input only; the state of the TRISA5 bit has no effect and will always read '1'.

14.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference generator is a 16-tap resistor ladder network that provides a fixed voltage reference when the comparators are in mode '010'. A programmable register controls the function of the reference generator. Register 14-1 lists the bit functions of the CVRCON register.

As shown in Figure 14-1, the resistor ladder is segmented to provide two ranges of CVREF values and has a power-down function to conserve power when the reference is not being used. The comparator reference supply voltage (also referred to as CVRSRC) comes directly from VDD. It should be noted, however, that the voltage at the top of the ladder is CVRSRC – VSAT, where VSAT is the saturation voltage of the power switch transistor. This reference will only be as accurate as the values of CVRSRC and VSAT.

The output of the reference generator may be connected to the RA2/AN2/CVREF/VREF- pin (VREF- is available on the PIC16F88 device only). This can be used as a simple D/A function by the user if a very high-impedance load is used. The primary purpose of this function is to provide a test path for testing the reference generator function.

REGISTER 14-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER (ADDRESS 9Dh)

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE	CVRR	—	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

bit 7 **CVREN:** Comparator Voltage Reference Enable bit
1 = CVREF circuit powered on
0 = CVREF circuit powered down

bit 6 **CVROE:** Comparator VREF Output Enable bit
1 = CVREF voltage level is output on the RA2/AN2/CVREF/VREF- pin(1)
0 = CVREF voltage level is disconnected from the RA2/AN2/CVREF/VREF- pin(1)

bit 5 **CVRR:** Comparator VREF Range Selection bit(1)
1 = 0.00 CVRSRC to 0.625 CVRSRC with CVRSRC/24 step size
0 = 0.25 CVRSRC to 0.72 CVRSRC with CVRSRC/32 step size

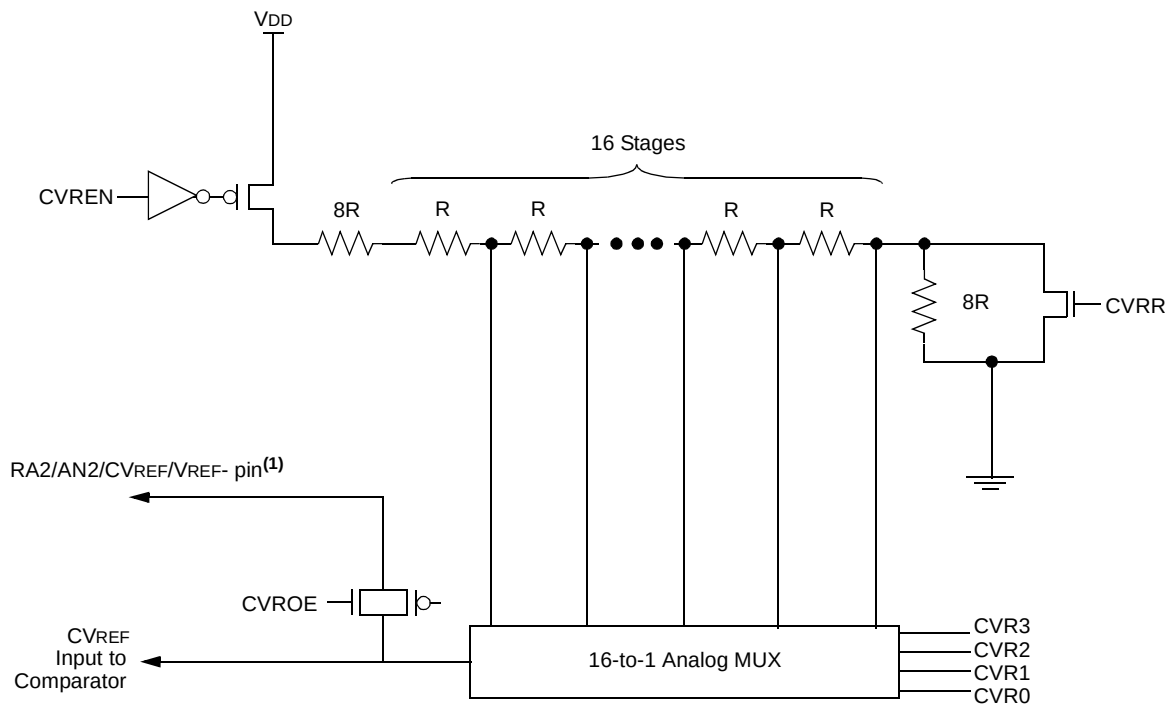
bit 4 **Unimplemented:** Read as '0'

bit 3-0 **CVR<3:0>:** Comparator VREF Value Selection 0 ≤ VR3:VR0 ≤ 15 bits(1)
When CVRR = 1:
CVREF = (VR<3:0>/24) • (CVRSRC)
When CVRR = 0:
CVREF = 1/4 • (CVRSRC) + (VR3:VR0/32) • (CVRSRC)

Note 1: VREF is available on the PIC16F88 device only.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

FIGURE 14-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM



Note 1: VREF is available on the PIC16F88 device only.

TABLE 14-1: REGISTERS ASSOCIATED WITH COMPARATOR VOLTAGE REFERENCE

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
9Dh	CVRCON	CVREN	CVROE	CVRR	—	CVR3	CVR2	CVR1	CVR0	000- 0000	000- 0000
9Ch	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	0000 0111

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used with the comparator voltage reference.

15.0 SPECIAL FEATURES OF THE CPU

These devices have a host of features intended to maximize system reliability, minimize cost through elimination of external components, provide power-saving operating modes and offer code protection:

- Reset
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT)
- Two-Speed Start-up
- Fail-Safe Clock Monitor
- Sleep
- Code Protection
- ID Locations
- In-Circuit Serial Programming™ (ICSP™)

There are two timers that offer necessary delays on power-up. One is the Oscillator Start-up Timer (OST), intended to keep the chip in Reset until the crystal oscillator is stable. The other is the Power-up Timer (PWRT), which provides a fixed delay of 72 ms (nominal) on power-up only. It is designed to keep the part in Reset while the power supply stabilizes and is enabled or disabled using a configuration bit. With these two timers on-chip, most applications need no external Reset circuitry.

Sleep mode is designed to offer a very low-current Power-down mode. The user can wake-up from Sleep through external Reset, Watchdog Timer wake-up or through an interrupt.

Additional information on special features is available in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023).

15.1 Configuration Bits

The configuration bits can be programmed (read as '0'), or left unprogrammed (read as '1'), to select various device configurations. These bits are mapped in program memory locations 2007h and 2008h.

The user will note that address 2007h is beyond the user program memory space which can be accessed only during programming.

REGISTER 15-1: CONFIG1: CONFIGURATION WORD 1 REGISTER (ADDRESS 2007h)

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
CP	CCPMX	DEBUG	WRT1	WRT0	CPD	LVP	BOREN	MCLRE	FOSC2	$\overline{\text{PWRTEN}}$	WDTEN	FOSC1	FOSC0
bit 13													bit 0

bit 13 **CP:** Flash Program Memory Code Protection bits
1 = Code protection off
0 = 0000h to 0FFFh code-protected (all protected)

bit 12 **CCPMX:** CCP1 Pin Selection bit
1 = CCP1 function on RB0
0 = CCP1 function on RB3

bit 11 **DEBUG:** In-Circuit Debugger Mode bit
1 = In-Circuit Debugger disabled, RB6 and RB7 are general purpose I/O pins
0 = In-Circuit Debugger enabled, RB6 and RB7 are dedicated to the debugger

bit 10-9 **WRT<1:0>:** Flash Program Memory Write Enable bits
11 = Write protection off
10 = 0000h to 00FFh write-protected, 0100h to 0FFFh may be modified by EECON control
01 = 0000h to 07FFh write-protected, 0800h to 0FFFh may be modified by EECON control
00 = 0000h to 0FFFh write-protected

bit 8 **CPD:** Data EE Memory Code Protection bit
1 = Code protection off
0 = Data EE memory code-protected

bit 7 **LVP:** Low-Voltage Programming Enable bit
1 = RB3/PGM pin has PGM function, Low-Voltage Programming enabled
0 = RB3 is digital I/O, HV on $\overline{\text{MCLR}}$ must be used for programming

bit 6 **BOREN:** Brown-out Reset Enable bit
1 = BOR enabled
0 = BOR disabled

bit 5 **MCLRE:** RA5/$\overline{\text{MCLR}}$/V_{PP} Pin Function Select bit
1 = RA5/$\overline{\text{MCLR}}$/V_{PP} pin function is $\overline{\text{MCLR}}$
0 = RA5/$\overline{\text{MCLR}}$/V_{PP} pin function is digital I/O, $\overline{\text{MCLR}}$ internally tied to V_{DD}

bit 3 **$\overline{\text{PWRTEN}}$:** Power-up Timer Enable bit
1 = PWRT disabled
0 = PWRT enabled

bit 2 **WDTEN:** Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled

bit 4, 1-0 **FOSC<2:0>:** Oscillator Selection bits
111 = EXTRC oscillator; CLKO function on RA6/OSC2/CLKO
110 = EXTRC oscillator; port I/O function on RA6/OSC2/CLKO
101 = INTRC oscillator; CLKO function on RA6/OSC2/CLKO pin and port I/O function on RA7/OSC1/CLKI pin
100 = INTRC oscillator; port I/O function on both RA6/OSC2/CLKO pin and RA7/OSC1/CLKI pin
011 = ECIO; port I/O function on RA6/OSC2/CLKO
010 = HS oscillator
001 = XT oscillator
000 = LP oscillator

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

REGISTER 15-2: CONFIG2: CONFIGURATION WORD 2 REGISTER (ADDRESS 2008h)

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1	R/P-1	R/P-1
—	—	—	—	—	—	—	—	—	—	—	—	IESO	FCMEN
bit 13													bit 0

bit 13-2 **Unimplemented:** Read as '1'

bit 1 **IESO:** Internal External Switchover bit
1 = Internal External Switchover mode enabled
0 = Internal External Switchover mode disabled

bit 0 **FCMEN:** Fail-Safe Clock Monitor Enable bit
1 = Fail-Safe Clock Monitor enabled
0 = Fail-Safe Clock Monitor disabled

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

15.2 Reset

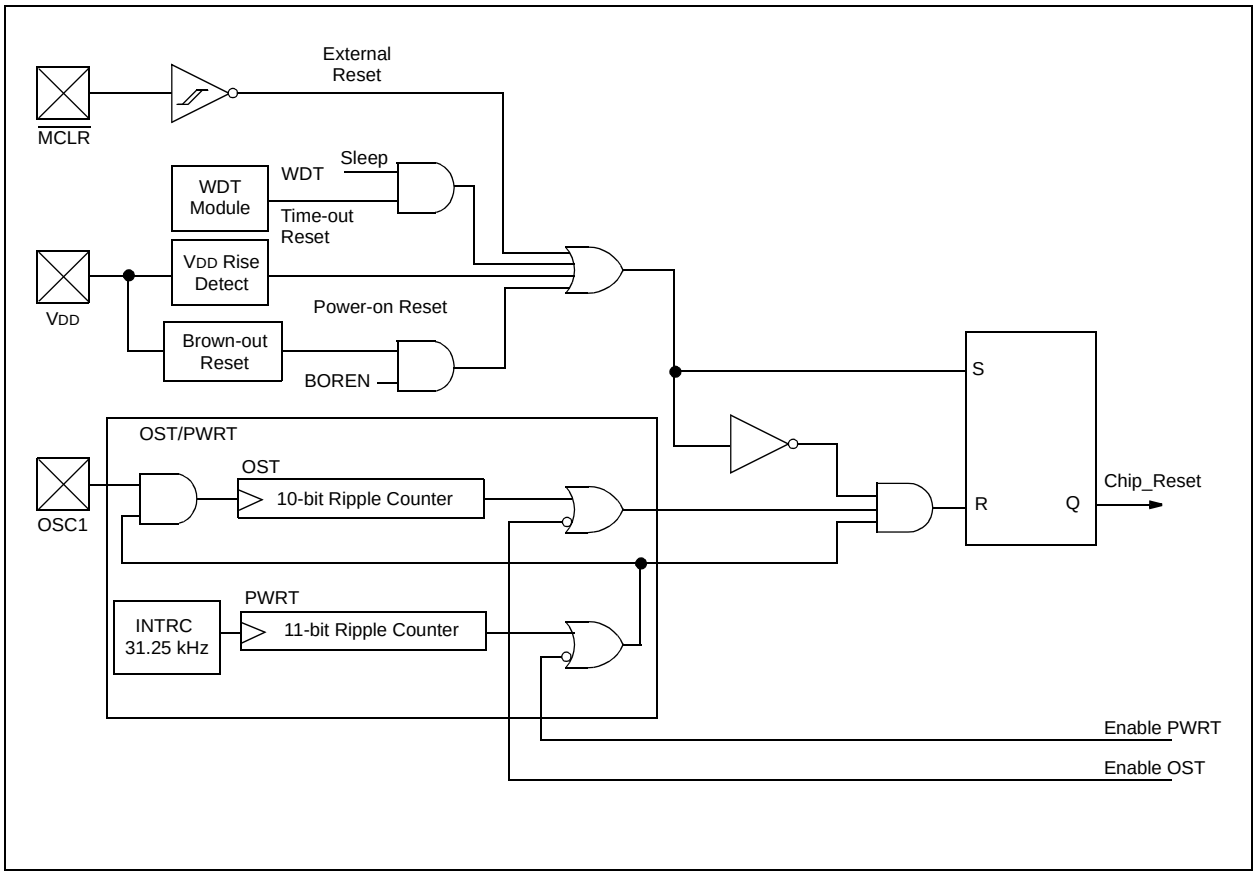
The PIC16F87/88 differentiates between various kinds of Reset:

- Power-on Reset (POR)
- $\overline{\text{MCLR}}$ Reset during normal operation
- $\overline{\text{MCLR}}$ Reset during Sleep
- WDT Reset during normal operation
- WDT wake-up during Sleep
- Brown-out Reset (BOR)

Some registers are not affected in any Reset condition. Their status is unknown on POR and unchanged in any other Reset. Most other registers are reset to a "Reset state" on Power-on Reset (POR), on the $\overline{\text{MCLR}}$ and WDT Reset, on $\overline{\text{MCLR}}$ Reset during Sleep and Brown-out Reset (BOR). They are not affected by a WDT wake-up which is viewed as the resumption of normal operation. The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits are set or cleared differently in different Reset situations, as indicated in Table 15-3. These bits are used in software to determine the nature of the Reset. Upon a POR, BOR or wake-up from Sleep, the CPU requires approximately 5-10 µs to become ready for code execution. This delay runs in parallel with any other timers. See Table 15-4 for a full description of Reset states of all registers.

A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 15-1.

FIGURE 15-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



15.3 $\overline{\text{MCLR}}$

PIC16F87/88 devices have a noise filter in the $\overline{\text{MCLR}}$ Reset path. The filter will detect and ignore small pulses.

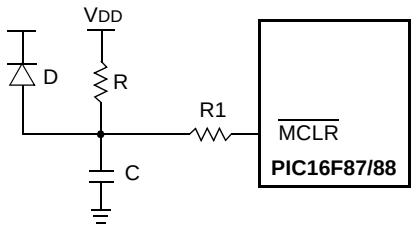
It should be noted that a WDT Reset does not drive $\overline{\text{MCLR}}$ pin low.

The behavior of the ESD protection on the $\overline{\text{MCLR}}$ pin has been altered from previous devices of this family. Voltages applied to the pin that exceed its specification can result in both $\overline{\text{MCLR}}$ and excessive current beyond the device specification during the ESD event. The circuit, as shown in Figure 15-2, is suggested.

Note:	For this reason, Microchip recommends that the $\overline{\text{MCLR}}$ pin no longer be tied directly to VDD.

The RA5/$\overline{\text{MCLR}}$/VPP pin can be configured for $\overline{\text{MCLR}}$ (default), or as an I/O pin (RA5). This is configured through the MCLRE bit in Configuration Word 1.

FIGURE 15-2: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)



Note 1: External Power-on Reset circuit is required only if the VDD power-up slope is too slow. The diode D helps discharge the capacitor quickly when VDD powers down.

2: R < 40 kΩ is recommended to make sure that the voltage drop across R does not violate the device's electrical specification.

3: R1 = 1 kΩ to 10 kΩ will limit any current flowing into $\overline{\text{MCLR}}$ from external capacitor C (0.1 µF), in the event of RA5/$\overline{\text{MCLR}}$/VPP pin breakdown due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS).

15.4 Power-on Reset (POR)

A Power-on Reset pulse is generated on-chip when VDD rise is detected (in the range of 1.2V-1.7V). To take advantage of the POR, tie the $\overline{\text{MCLR}}$ pin to VDD, as described in **Section 15.3 "MCLR"**. A maximum rise time for VDD is specified. See **Section 18.0 "Electrical Characteristics"** for details.

When the device starts normal operation (exits the Reset condition), device operating parameters (voltage, frequency, temperature,...) must be met to ensure operation. If these conditions are not met, the device must be held in Reset until the operating conditions are met. For more information, see Application Note, *AN607 "Power-up Trouble Shooting"* (DS00607).

15.5 Power-up Timer (PWRT)

The Power-up Timer (PWRT) of the PIC16F87/88 is a counter that uses the INTRC oscillator as the clock input. This yields a count of 72 ms. While the PWRT is counting, the device is held in Reset.

The power-up time delay depends on the INTRC and will vary from chip-to-chip due to temperature and process variation. See DC parameter #33 for details.

The PWRT is enabled by clearing configuration bit $\overline{\text{PWRTEN}}$.

15.6 Oscillator Start-up Timer (OST)

The Oscillator Start-up Timer (OST) provides a 1024 oscillator cycle (from OSC1 input) delay after the PWRT delay is over (if enabled). This helps to ensure that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for XT, LP and HS modes and only on Power-on Reset or wake-up from Sleep.

15.7 Brown-out Reset (BOR)

The configuration bit, BOREN, can enable or disable the Brown-out Reset circuit. If VDD falls below VBOR (parameter D005, about 4V) for longer than TBOR (parameter #35, about 100 µs), the brown-out situation will reset the device. If VDD falls below VBOR for less than TBOR, a Reset may not occur.

Once the brown-out occurs, the device will remain in Brown-out Reset until VDD rises above VBOR. The Power-up Timer (if enabled) will keep the device in Reset for TPWRT (parameter #33, about 72 ms). If VDD should fall below VBOR during TPWRT, the Brown-out Reset process will restart when VDD rises above VBOR with the Power-up Timer Reset. Unlike previous PIC16 devices, the PWRT is no longer automatically enabled when the Brown-out Reset circuit is enabled. The $\overline{\text{PWRTEN}}$ and BOREN configuration bits are independent of each other.

15.8 Time-out Sequence

On power-up, the time-out sequence is as follows: the PWRT delay starts (if enabled) when a POR occurs. Then, OST starts counting 1024 oscillator cycles when PWRT ends (LP, XT, HS). When the OST ends, the device comes out of Reset.

If $\overline{\text{MCLR}}$ is kept low long enough, all delays will expire. Bringing $\overline{\text{MCLR}}$ high will begin execution immediately. This is useful for testing purposes, or to synchronize more than one PIC16F87/88 device operating in parallel.

Table 15-3 shows the Reset conditions for the STATUS, PCON and PC registers, while Table 15-4 shows the Reset conditions for all the registers.

15.9 Power Control/Status Register (PCON)

The Power Control/Status Register, PCON, has two bits to indicate the type of Reset that last occurred.

Bit 0 is Brown-out Reset Status bit, $\overline{\text{BOR}}$. Bit $\overline{\text{BOR}}$ is unknown on a Power-on Reset. It must then be set by the user and checked on subsequent Resets to see if bit $\overline{\text{BOR}}$ cleared, indicating a Brown-out Reset occurred. When the Brown-out Reset is disabled, the state of the $\overline{\text{BOR}}$ bit is unpredictable.

Bit 1 is $\overline{\text{POR}}$ (Power-on Reset Status bit). It is cleared on a Power-on Reset and unaffected otherwise. The user must set this bit following a Power-on Reset.

TABLE 15-1: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up		Brown-out Reset		Wake-up from Sleep
	$\overline{\text{PWRTE}}$ = 0	$\overline{\text{PWRTE}}$ = 1	$\overline{\text{PWRTE}}$ = 0	$\overline{\text{PWRTE}}$ = 1	
XT, HS, LP	TPWRT + 1024 • TOSC	1024 • TOSC	TPWRT + 1024 • TOSC	1024 • TOSC	1024 • TOSC
EXTRC, INTRC	TPWRT	5-10 µs(1)	TPWRT	5-10 µs(1)	5-10 µs(1)
T1OSC	—	—	—	—	5-10 µs(1)

Note 1: CPU start-up is always invoked on POR, BOR and wake-up from Sleep. The 5-10 µs delay is based on a 1 MHz system clock.

TABLE 15-2: STATUS BITS AND THEIR SIGNIFICANCE

$\overline{\text{POR}}$	$\overline{\text{BOR}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	
0	x	1	1	Power-on Reset
0	x	0	x	Illegal, $\overline{\text{TO}}$ is set on $\overline{\text{POR}}$
0	x	x	0	Illegal, $\overline{\text{PD}}$ is set on $\overline{\text{POR}}$
1	0	1	1	Brown-out Reset
1	1	0	1	WDT Reset
1	1	0	0	WDT Wake-up
1	1	u	u	$\overline{\text{MCLR}}$ Reset during Normal Operation
1	1	1	0	$\overline{\text{MCLR}}$ Reset during Sleep or Interrupt Wake-up from Sleep

Legend: u = unchanged, x = unknown

TABLE 15-3: RESET CONDITION FOR SPECIAL REGISTERS

Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	000h	0001 1xxx	---- --0x
$\overline{\text{MCLR}}$ Reset during normal operation	000h	000u uuuu	---- --uu
$\overline{\text{MCLR}}$ Reset during Sleep	000h	0001 0uuu	---- --uu
WDT Reset	000h	0000 1uuu	---- --uu
WDT Wake-up	PC + 1	uuu0 0uuu	---- --uu
Brown-out Reset	000h	0001 1uuu	---- --u0
Interrupt Wake-up from Sleep	PC + 1(1)	uuu1 0uuu	---- --uu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0'

Note 1: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

TABLE 15-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Reset, WDT Reset	Wake-up via WDT or Interrupt
W	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	N/A	N/A	N/A
TMR0	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	0000h	0000h	PC + 1(2)
STATUS	0001 1xxx	000q quuu(3)	uuuq quuu(3)
FSR	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA (PIC16F87) PORTA (PIC16F88)	xxxx 0000 xxx0 0000	uuuu 0000 uuu0 0000	uuuu uuuu uuuu uuuu
PORTB (PIC16F87) PORTB (PIC16F87)	xxxx xxxx 00xx xxxx	uuuu uuuu 00uu uuuu	uuuu uuuu uuuu uuuu
PCLATH	---0 0000	---0 0000	---u uuuu
INTCON	0000 000x	0000 000u	uuuu uuuu(1)
PIR1	-000 0000	-000 0000	-uuu uuuu(1)
PIR2	00-0 ----	00-0 ----	uu-u ----(1)
TMR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	-000 0000	-uuu uuuu	-uuu uuuu
TMR2	0000 0000	0000 0000	uuuu uuuu
T2CON	-000 0000	-000 0000	-uuu uuuu
SSPBUF	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPCON	0000 0000	0000 0000	uuuu uuuu
CCPR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	--00 0000	--00 0000	--uu uuuu
RCSTA	0000 000x	0000 000x	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition

Note 1: One or more bits in INTCON, PIR1 and PR2 will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

3: See Table 15-3 for Reset value for specific condition.

TABLE 15-4: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Reset, WDT Reset	Wake-up via WDT or Interrupt
TXREG	0000 0000	0000 0000	uuuu uuuu
RCREG	0000 0000	0000 0000	uuuu uuuu
ADRESH	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	0000 00-0	0000 00-0	uuuu uu-u
OPTION_REG	1111 1111	1111 1111	uuuu uuuu
TRISA	1111 1111	1111 1111	uuuu uuuu
TRISB	1111 1111	1111 1111	uuuu uuuu
PIE1	-000 0000	-000 0000	-uuu uuuu
PIE2	00-0 ----	00-0 ----	uu-u ----
PCON	---- --0q	---- --uu	---- --uu
OSCCON	-000 0000	-000 0000	-uuu uuuu
OSCTUNE	--00 0000	--00 0000	--uu uuuu
PR2	1111 1111	1111 1111	1111 1111
SSPADD	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	0000 0000	0000 0000	uuuu uuuu
TXSTA	0000 -010	0000 -010	uuuu -u1u
SPBRG	0000 0000	0000 0000	uuuu uuuu
ANSEL	-111 1111	-111 1111	-111 1111
CMCON	0000 0111	0000 0111	uuuu u111
CVRCON	000- 0000	000- 0000	uuu- uuuu
WDTCON	---0 1000	---0 1000	---u uuuu
ADRESL	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON1	0000 ----	0000 ----	uuuu ----
EEDATA	xxxx xxxx	uuuu uuuu	uuuu uuuu
EEADR	xxxx xxxx	uuuu uuuu	uuuu uuuu
EEDATH	--xx xxxx	--uu uuuu	--uu uuuu
EEADRH	---- -xxx	---- -uuu	---- -uuu
EECON1	x--x x000	u--x u000	u--u uuuu
EECON2	---- ----	---- ----	---- ----

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition

Note 1: One or more bits in INTCON, PIR1 and PR2 will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

3: See Table 15-3 for Reset value for specific condition.

FIGURE 15-3: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO VDD THROUGH PULL-UP RESISTOR)

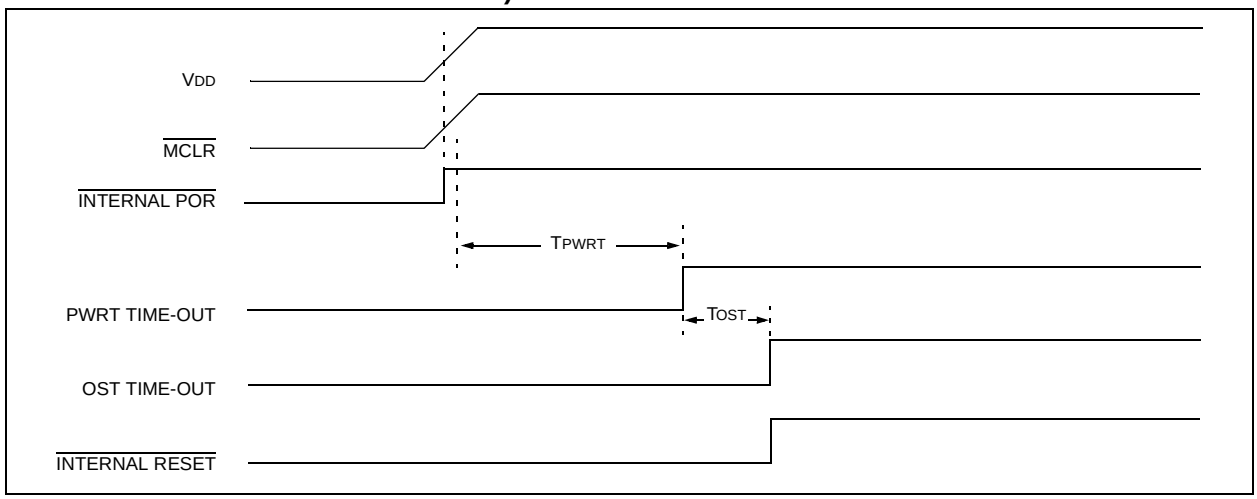


FIGURE 15-4: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO VDD THROUGH RC NETWORK): CASE 1

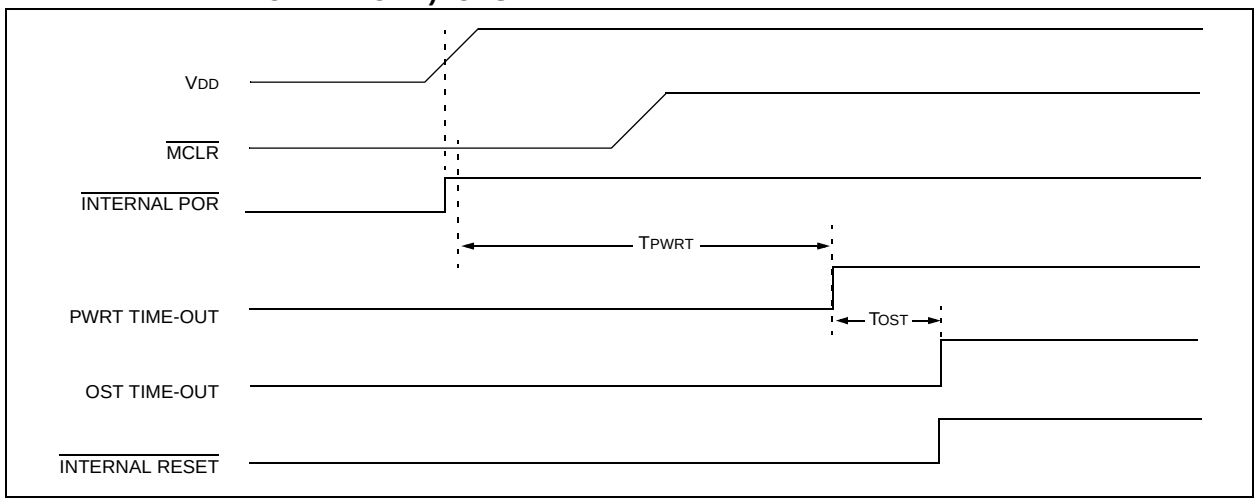


FIGURE 15-5: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO VDD THROUGH RC NETWORK): CASE 2

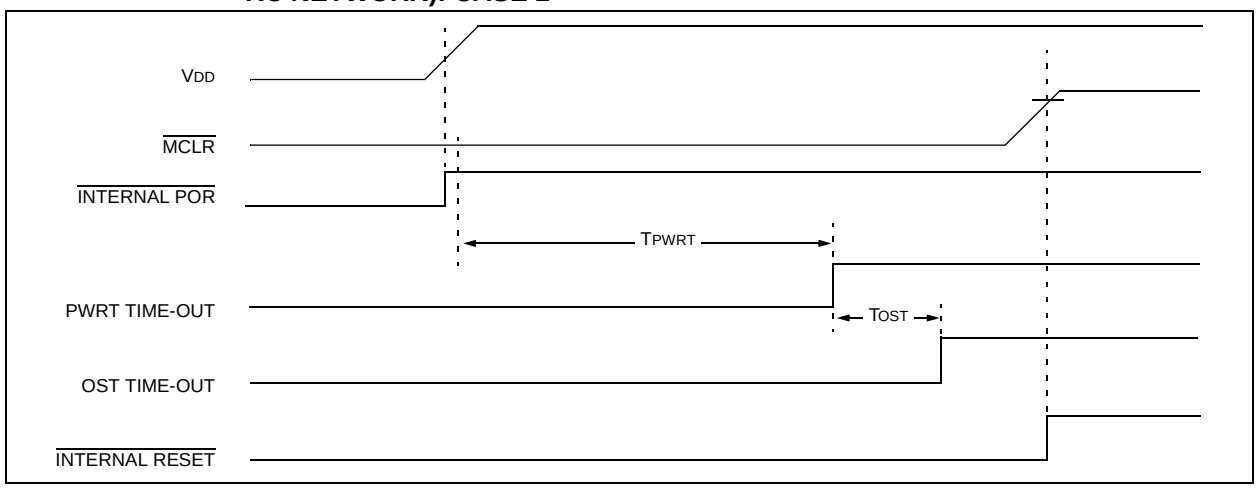
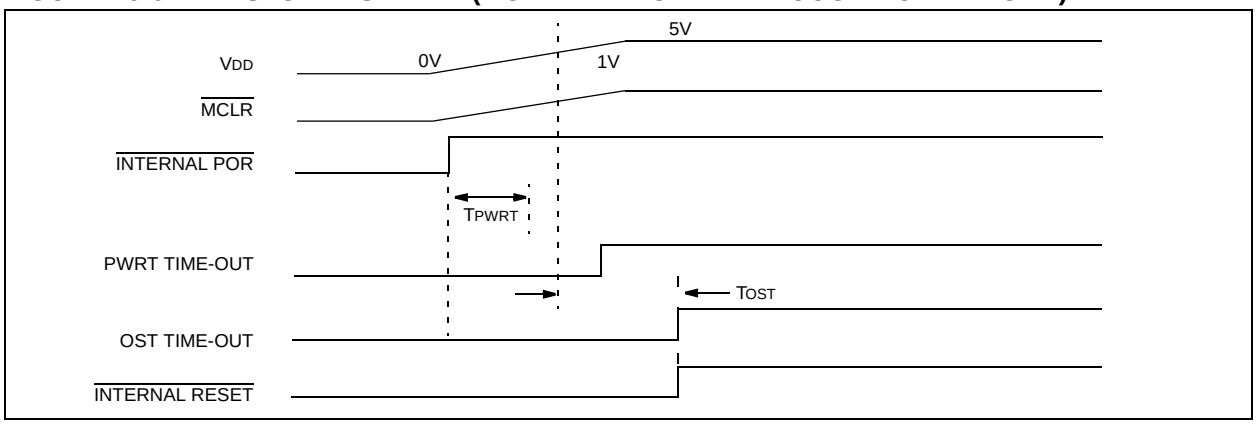


FIGURE 15-6: SLOW RISE TIME ($\overline{\text{MCLR}}$ TIED TO V_{DD} THROUGH RC NETWORK)



15.10 Interrupts

The PIC16F87/88 has up to 12 sources of interrupt. The Interrupt Control register (INTCON) records individual interrupt requests in flag bits. It also has individual and global interrupt enable bits.

Note:	Individual interrupt flag bits are set regardless of the status of their corresponding mask bit or the GIE bit.

A global interrupt enable bit, GIE (INTCON<7>), enables (if set) all unmasked interrupts, or disables (if cleared) all interrupts. When bit GIE is enabled and an interrupt's flag bit and mask bit are set, the interrupt will vector immediately. Individual interrupts can be disabled through their corresponding enable bits in various registers. Individual interrupt bits are set regardless of the status of the GIE bit. The GIE bit is cleared on Reset.

The "return from interrupt" instruction, `RETFIE`, exits the interrupt routine, as well as sets the GIE bit which re-enables interrupts.

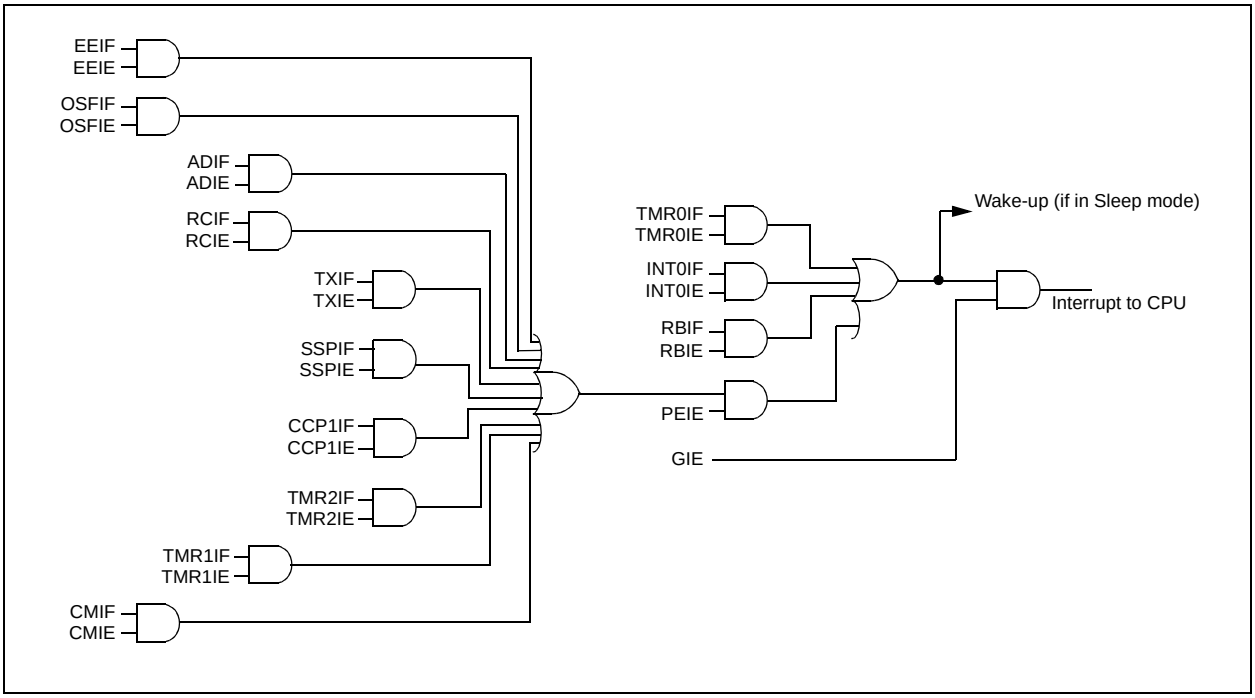
The RB0/INT pin interrupt, the RB port change interrupt and the TMR0 overflow interrupt flags are contained in the INTCON register.

The peripheral interrupt flags are contained in the Special Function Register, PIR1. The corresponding interrupt enable bits are contained in Special Function Register, PIE1 and the peripheral interrupt enable bit is contained in Special Function Register, INTCON.

When an interrupt is serviced, the GIE bit is cleared to disable any further interrupt, the return address is pushed onto the stack and the PC is loaded with 0004h. Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

For external interrupt events, such as the INT pin or PORTB change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends on when the interrupt event occurs, relative to the current Q cycle. The latency is the same for one or two cycle instructions. Individual interrupt flag bits are set regardless of the status of their corresponding mask bit, PEIE bit or the GIE bit.

FIGURE 15-7: INTERRUPT LOGIC



15.10.1 INT INTERRUPT

External interrupt on the RB0/INT pin is edge-triggered, either rising if bit INTEDG (OPTION_REG<6>) is set, or falling if the INTEDG bit is clear. When a valid edge appears on the RB0/INT pin, flag bit, INT0IF (INTCON<1>), is set. This interrupt can be disabled by clearing enable bit INT0IE (INTCON<4>). Flag bit INT0IF must be cleared in software in the Interrupt Service Routine before re-enabling this interrupt. The INT interrupt can wake-up the processor from Sleep, if bit INT0IE was set prior to going into Sleep. The status of global interrupt enable bit GIE decides whether or not the processor branches to the interrupt vector, following wake-up. See **Section 15.13 "Power-Down Mode (Sleep)"** for details on Sleep mode.

15.10.2 TMR0 INTERRUPT

An overflow (FFh → 00h) in the TMR0 register will set flag bit TMR0IF (INTCON<2>). The interrupt can be enabled/disabled by setting/clearing enable bit TMR0IE (INTCON<5>), see **Section 6.0 "Timer0 Module"**.

15.10.3 PORTB INTCON CHANGE

An input change on PORTB<7:4> sets flag bit RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit RBIE (INTCON<4>), see **Section 3.2 "EECON1 and EECON2 Registers"**.

15.11 Context Saving During Interrupts

During an interrupt, only the return PC value is saved on the stack. Typically, users may wish to save key registers during an interrupt (i.e., W, STATUS registers).

Since the upper 16 bytes of each bank are common in the PIC16F87/88 devices, temporary holding registers W_TEMP, STATUS_TEMP and PCLATH_TEMP should be placed in here. These 16 locations don't require banking and therefore, make it easier for context save and restore. The same code shown in Example 15-1 can be used.

EXAMPLE 15-1: SAVING STATUS, W AND PCLATH REGISTERS IN RAM

```
MOVWF    W_TEMP           ;Copy W to TEMP register
SWAPF    STATUS, W        ;Swap status to be saved into W
CLRF     STATUS           ;bank 0, regardless of current bank, Clears IRP,RP1,RP0
MOVWF    STATUS_TEMP      ;Save status to bank zero STATUS_TEMP register
MOVF     PCLATH, W        ;Only required if using page 1
MOVWF    PCLATH_TEMP      ;Save PCLATH into W
CLRF     PCLATH           ;Page zero, regardless of current page
:
:(ISR)                    ;(Insert user code here)
:
MOVF     PCLATH_TEMP, W   ;Restore PCLATH
MOVWF    PCLATH           ;Move W into PCLATH
SWAPF    STATUS_TEMP, W   ;Swap STATUS_TEMP register into W
                          ;(sets bank to original state)
MOVWF    STATUS           ;Move W into STATUS register
SWAPF    W_TEMP, F        ;Swap W_TEMP
SWAPF    W_TEMP, W        ;Swap W_TEMP into W
```

15.12 Watchdog Timer (WDT)

For PIC16F87/88 devices, the WDT has been modified from previous PIC16 devices. The new WDT is code and functionally backward compatible with previous PIC16 WDT modules and allows the user to have a scaler value for the WDT and TMR0 at the same time. In addition, the WDT time-out value can be extended to 268 seconds, using the prescaler with the postscaler when PSA is set to '`1`'.

15.12.1 WDT OSCILLATOR

The WDT derives its time base from the 31.25 kHz INTRC. The value of WDTCON is '`---0 1000`' on all Resets. This gives a nominal time base of 16.38 ms, which is compatible with the time base generated with previous PIC16 microcontroller versions.

Note:	When the OST is invoked, the WDT is held in Reset because the WDT ripple counter is used by the OST to perform the oscillator delay count. When the OST count has expired, the WDT will begin counting (if enabled).

A new prescaler has been added to the path between the internal RC and the multiplexors used to select the path for the WDT. This prescaler is 16 bits and can be programmed to divide the internal RC by 32 to 65536, giving the time base used for the WDT a nominal range of 1 ms to 2.097s.

15.12.2 WDT CONTROL

The WDTEN bit is located in Configuration Word 1 and when this bit is set, the WDT runs continuously.

The SWDTEN bit is in the WDTCON register. When the WDTEN bit in the Configuration Word 1 register is set, the SWDTEN bit has no effect. If WDTEN is clear, then the SWDTEN bit can be used to enable and disable the WDT. Setting the bit will enable it and clearing the bit will disable it.

The PSA and PS<2:0> bits (OPTION_REG register) have the same function as in previous versions of the PIC16 family of microcontrollers.

FIGURE 15-8: WATCHDOG TIMER BLOCK DIAGRAM

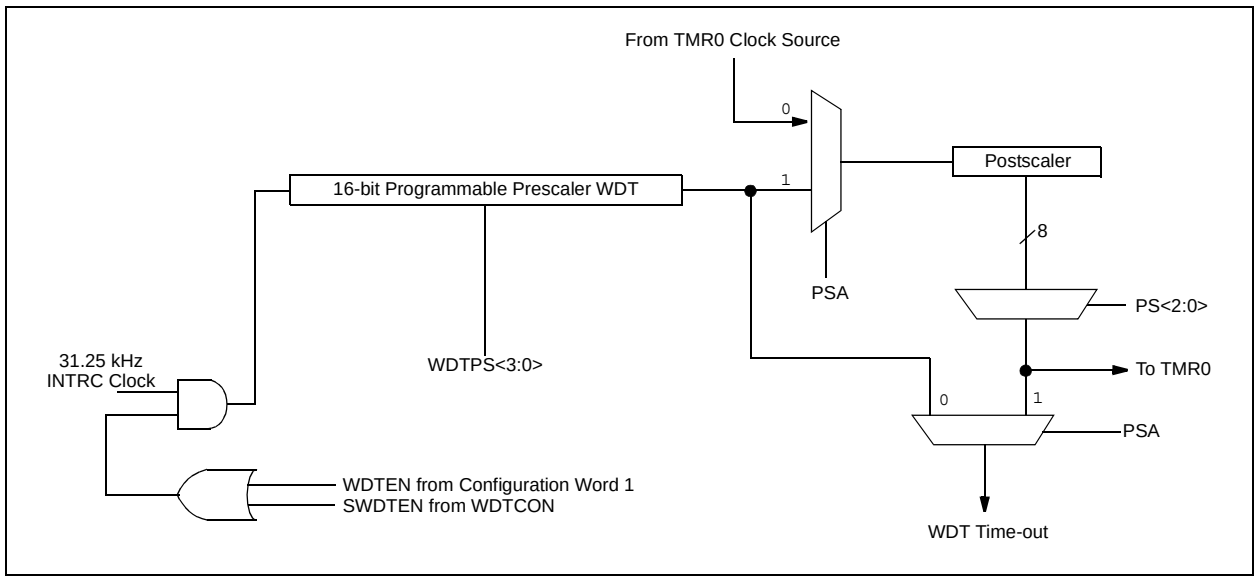


TABLE 15-5: PRESCALER/POSTSCALER BIT STATUS

Conditions	Prescaler	Postscaler (PSA = 1)
WDTEN = 0	Cleared	Cleared
CLRWDT command		
Oscillator fail detected		
Exit Sleep + System Clock = T1OSC, EXTRC, INTRC, ECIO		
Exit Sleep + System Clock = XT, HS, LP	Cleared at end of OST	Cleared at end of OST

REGISTER 15-3: WDTCON: WATCHDOG CONTROL REGISTER (ADDRESS 105h)

U-0	U-0	U-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0
—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	SWDTEN(1)
bit 7							bit 0

bit 7-5 **Unimplemented:** Read as '0'

bit 4-1 **WDTPS<3:0>:** Watchdog Timer Period Select bits

Bit Value		Prescale Rate
0000	=	1:32
0001	=	1:64
0010	=	1:128
0011	=	1:256
0100	=	1:512
0101	=	1:1024
0110	=	1:2048
0111	=	1:4096
1000	=	1:8192
1001	=	1:16394
1010	=	1:32768
1011	=	1:65536

bit 0 **SWDTEN:** Software Enable/Disable for Watchdog Timer bit(1)

1 = WDT is turned on
0 = WDT is turned off

Note 1: If WDTEN configuration bit = 1, then WDT is always enabled, irrespective of this control bit. If WDTEN configuration bit = 0, then it is possible to turn WDT on/off with this control bit.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

TABLE 15-6: SUMMARY OF WATCHDOG TIMER REGISTERS

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
81h,181h	OPTION_REG	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
2007h	Configuration bits	LVP	BOREN	MCLRE	FOSC2	$\overline{\text{PWRTEN}}$	WDTEN	FOSC1	FOSC0
105h	WDTCON	—	—	—	WDTPS3	$\overline{\text{WDTPS2}}$	WSTPS1	WDTPS0	SWDTEN

Legend: Shaded cells are not used by the Watchdog Timer.
Note 1: See Register 15-1 for operation of these bits.

15.12.3 TWO-SPEED CLOCK START-UP MODE

Two-Speed Start-up mode minimizes the latency between oscillator start-up and code execution that may be selected with the IESO (Internal/External Switchover) bit in Configuration Word 2. This mode is achieved by initially using the INTRC for code execution until the primary oscillator is stable.

If this mode is enabled and any of the following conditions exist, the system will begin execution with the INTRC oscillator. This results in almost immediate code execution with a minimum of delay.

- POR and after the Power-up Timer has expired (if $\overline{\text{PWRTEN}}$ = 0);
- or following a wake-up from Sleep;
- or a Reset when running from T1OSC or INTRC (after a Reset, SCS<1:0> are always set to '00').

Note:	Following any Reset, the IRCF bits are zeroed and the frequency selection is forced to 31.25 kHz. The user can modify the IRCF bits to select a higher internal oscillator frequency.

If the primary oscillator is configured to be anything other than XT, LP or HS, then Two-Speed Start-up mode is disabled because the primary oscillator will not require any time to become stable after POR, or an exit from Sleep.

If the IRCF bits of the OSCCON register are configured to a non-zero value prior to entering Sleep mode, the system clock frequency will come from the output of the INTOSC. The IOFS bit in the OSCCON register will be clear until the INTOSC is stable. This will allow the user to determine when the internal oscillator can be used for time critical applications.

Checking the state of the OSTS bit will confirm whether the primary clock configuration is engaged. If not, the OSTS bit will remain clear.

When the device is auto-configured in INTRC mode following a POR or wake-up from Sleep, the rules for entering other oscillator modes still apply, meaning the SCS<1:0> bits in OSCCON can be modified before the OST time-out has occurred. This would allow the application to wake-up from Sleep, perform a few instructions using the INTRC as the clock source and go back to Sleep without waiting for the primary oscillator to become stable.

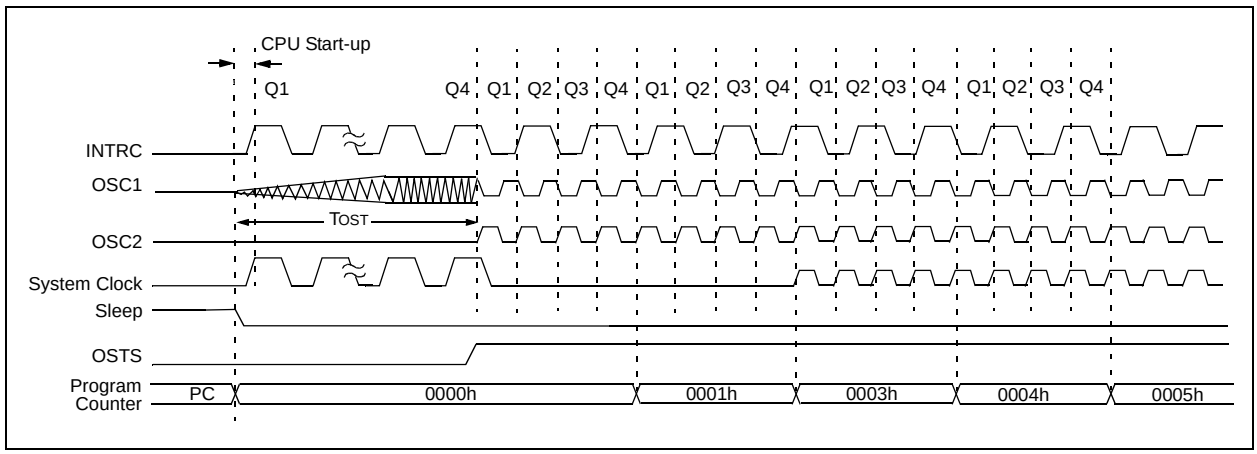
Note:	Executing a SLEEP instruction will abort the oscillator start-up time and will cause the OSTS bit to remain clear.

15.12.3.1 Two-Speed Start-up Mode Sequence

1. Wake-up from Sleep, Reset or POR.
2. OSCCON bits configured to run from INTRC (31.25 kHz).
3. Instructions begin execution by INTRC (31.25 kHz).
4. OST enabled to count 1024 clock cycles.
5. OST timed out, wait for falling edge of INTRC.
6. OSTS is set.
7. System clock held low for eight falling edges of new clock (LP, XT or HS).
8. System clock is switched to primary source (LP, XT or HS).

The software may read the OSTS bit to determine when the switchover takes place so that any software timing edges can be adjusted.

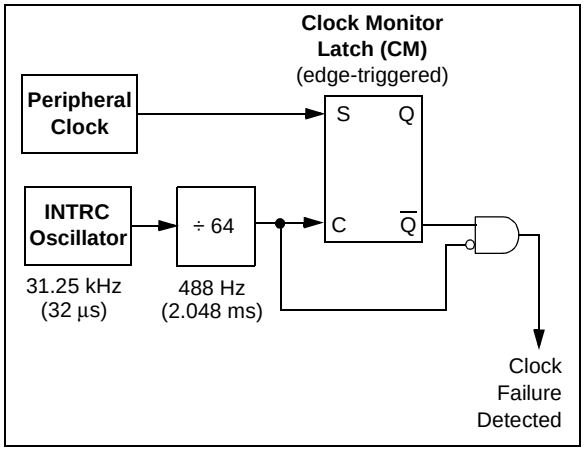
FIGURE 15-9: TWO-SPEED START-UP MODE



15.12.4 FAIL-SAFE OPTION

The Fail-Safe Clock Monitor (FSCM) is designed to allow the device to continue to operate even in the event of an oscillator failure.

FIGURE 15-10: FSCM BLOCK DIAGRAM



The FSCM function is enabled by setting the FCMEN bit in Configuration Word 2.

In the event of an oscillator failure, the FSCM will generate an oscillator fail interrupt and will switch the system clock over to the internal oscillator. The system will continue to come from the internal oscillator until the fail-safe condition is exited. The fail-safe condition is exited with either a Reset, the execution of a `SLEEP` instruction or a write to the OSCCON register.

The frequency of the internal oscillator will depend upon the value contained in the IRCF bits. Another clock source can be selected via the IRCF and the SCS bits of the OSCCON register.

The FSCM sample clock is generated by dividing the INTRC clock by 64. This will allow enough time between FSCM sample clocks for a system clock edge to occur.

On the rising edge of the postscaled clock, the monitoring latch (CM = `0`) will be cleared. On a falling edge of the primary or secondary system clock, the monitoring latch will be set (CM = `1`). In the event that a falling edge of the postscaled clock occurs and the monitoring latch is not set, a clock failure has been detected.

While in Fail-Safe mode, a Reset will exit the fail-safe condition. If the primary clock source is configured for a crystal, the OST timer will wait for the 1024 clock cycles for the OST time-out and the device will continue running from the internal oscillator until the OST is complete. A `SLEEP` instruction, or a write to the SCS bits (where SCS bits do not = `00`), can be performed to put the device into a low-power mode.

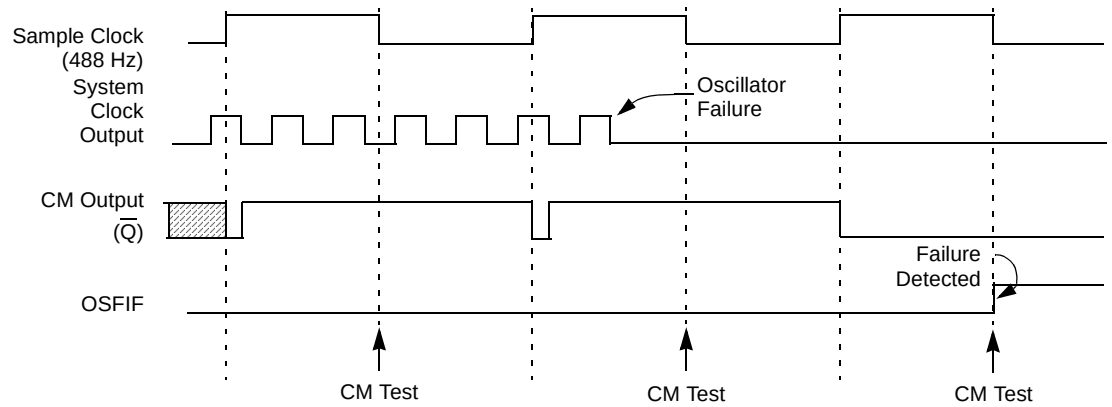
Note:	Two-Speed Start-up mode is automatically enabled when the fail-safe option is enabled.

If Reset occurs while in Fail-Safe mode and the primary clock source is EC or RC, then the device will immediately switch back to EC or RC mode.

15.12.4.1 Fail-Safe in Low-Power Mode

A write to the OSCCON register, or `SLEEP` instruction, will end the fail-safe condition. The system clock will default to the source selected by the SCS bits, which is either T1OSC, INTRC or none (Sleep mode). However, the FSCM will continue to monitor the system clock. If the secondary clock fails, the device will immediately switch to the internal oscillator clock. If OSFIE is set, an interrupt will be generated.

FIGURE 15-11: FSCM TIMING DIAGRAM



Note: The system clock is normally at a much higher frequency than the sample clock. The relative frequencies in this example have been chosen for clarity.

15.12.4.2 FSCM and the Watchdog Timer

When a clock failure is detected, SCS<1:0> will be forced to '10' which will reset the WDT (if enabled).

15.12.4.3 POR or Wake From Sleep

The FSCM is designed to detect oscillator failure at any point after the device has exited Power-on Reset (POR) or low-power Sleep mode. When the primary system clock is EC, RC or INTRC modes, monitoring can begin immediately following these events.

For Oscillator modes involving a crystal or resonator (HS, LP or XT), the situation is somewhat different. Since the oscillator may require a start-up time considerably longer than the FSCM sample clock time, a false clock failure may be detected. To prevent this, the internal oscillator block is automatically configured as the system clock and functions until the primary clock is stable (the OST timer has timed out). This is identical to Two-Speed Start-up mode. Once the primary clock is stable, the INTRC returns to its role as the FSCM source.

Note:	The same logic that prevents false oscillator failure interrupts on port or wake from Sleep, will also prevent the detection of the oscillator's failure to start at all following these events. This can be avoided by monitoring the OSTS bit and using a timing routine to determine if the oscillator is taking too long to start. Even so, no oscillator failure interrupt will be flagged.

15.12.4.4 Example Fail-Safe Conditions

1. CONDITIONS:

   The device is clocked from a crystal, crystal operation fails and then Sleep mode is entered.

   OSTS = 0

   SCS = 00

   OSFIF = 1

   USER ACTION:

   Sleep mode will exit the fail-safe condition. Therefore, if the user code did not handle the detected fail-safe prior to the SLEEP command, then upon wake-up, the device will try to start the crystal that failed and a fail-safe condition will not be detected. Monitoring the OSTS bit will determine if the crystal is operating. The user should not enter Sleep mode without handling the fail-safe condition first.

2. CONDITIONS:

   After a POR (Power-on Reset), the device is running in Two-Speed Start-up mode. The crystal fails before the OST has expired. If a crystal fails during the OST period, a fail-safe condition will not be detected (OSFIF will not get set).

   OSTS = 0

   SCS = 00

   OSFIF = 0

   USER ACTION:

   Check the OSTS bit. If it's clear and the OST should have expired at this point, then the user can assume the crystal has failed. The user should change the SCS bit to cause a clock switch which will also release the 10-bit ripple counter for WDT operation (if enabled).

3. CONDITIONS:

   The device is clocked from a crystal during normal operation and it fails.

   OSTS = 0

   SCS = 00

   OSFIF = 1

   USER ACTION:

   Clear the OSFIF bit. Configure the SCS bits for a clock switch and the fail-safe condition will be cleared. Later, if the user decides to, the crystal can be retried for operation. If this is done, the OSTS bit should be monitored to determine if the crystal operates.

15.13 Power-Down Mode (Sleep)

Power-Down mode is entered by executing a SLEEP instruction.

If enabled, the Watchdog Timer will be cleared but keeps running, the $\overline{PD}$ bit (STATUS<3>) is cleared, the $\overline{TO}$ (STATUS<4>) bit is set and the oscillator driver is turned off. The I/O ports maintain the status they had before the SLEEP instruction was executed (driving high, low or high-impedance).

For lowest current consumption in this mode, place all I/O pins at either V_{DD} or V_{SS}, ensure no external circuitry is drawing current from the I/O pin, power-down the A/D and disable external clocks. Pull all I/O pins that are high-impedance inputs, high or low externally, to avoid switching currents caused by floating inputs. The T0CKI input should also be at V_{DD} or V_{SS} for lowest current consumption. The contribution from on-chip pull-ups on PORTB should also be considered.

The $\overline{MCLR}$ pin must be at a logic high level (V_{IHMC}).

15.13.1 WAKE-UP FROM SLEEP

The device can wake-up from Sleep through one of the following events:

1. External Reset input on $\overline{\text{MCLR}}$ pin.
2. Watchdog Timer wake-up (if WDT was enabled).
3. Interrupt from INT pin, RB port change or a peripheral interrupt.

External $\overline{\text{MCLR}}$ Reset will cause a device Reset. All other events are considered a continuation of program execution and cause a “wake-up”. The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits in the STATUS register can be used to determine the cause of the device Reset. The $\overline{\text{PD}}$ bit, which is set on power-up, is cleared when Sleep is invoked. The $\overline{\text{TO}}$ bit is cleared if a WDT time-out occurred and caused wake-up.

The following peripheral interrupts can wake the device from Sleep:

1. TMR1 interrupt. Timer1 must be operating as an asynchronous counter.
2. CCP Capture mode interrupt.
3. Special event trigger (Timer1 in Asynchronous mode using an external clock).
4. SSP (Start/Stop) bit detect interrupt.
5. SSP transmit or receive in Slave mode (SPI/I^2C).
6. A/D conversion (when A/D clock source is RC).
7. EEPROM write operation completion.
8. Comparator output changes state.
9. AUSART RX or TX (Synchronous Slave mode).

Other peripherals cannot generate interrupts, since during Sleep, no on-chip clocks are present.

When the `SLEEP` instruction is being executed, the next instruction (PC + 1) is prefetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up occurs regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the `SLEEP` instruction. If the GIE bit is set (enabled), the device executes the instruction after the `SLEEP` instruction and then branches to the interrupt address (0004h). In cases where the execution of the instruction following `SLEEP` is not desirable, the user should have a `NOP` after the `SLEEP` instruction.

15.13.2 WAKE-UP USING INTERRUPTS

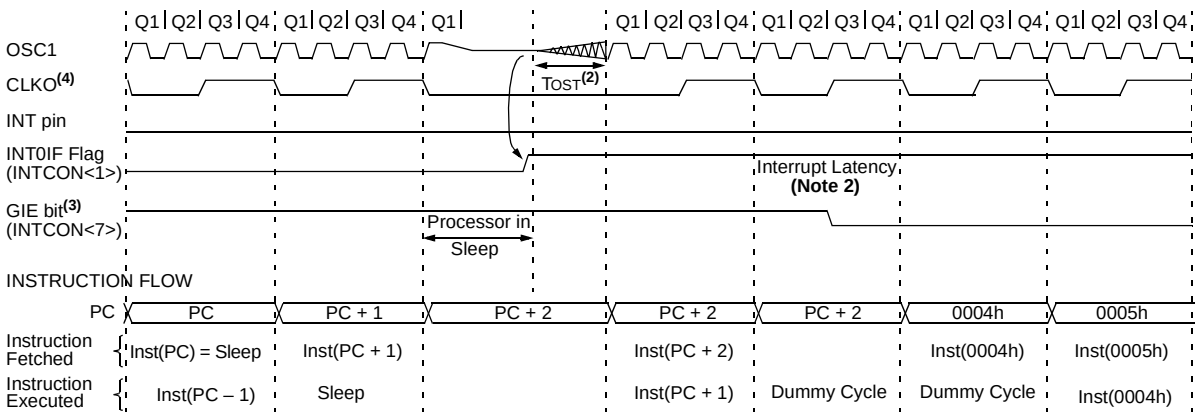
When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a `SLEEP` instruction, the `SLEEP` instruction will complete as a `NOP`. Therefore, the WDT and WDT prescaler and postscaler (if enabled) will not be cleared, the $\overline{\text{TO}}$ bit will not be set and the $\overline{\text{PD}}$ bit will not be cleared.
- If the interrupt occurs **during or after** the execution of a `SLEEP` instruction, the device will immediately wake-up from Sleep. The `SLEEP` instruction will be completely executed before the wake-up. Therefore, the WDT and WDT prescaler and postscaler (if enabled) will be cleared, the $\overline{\text{TO}}$ bit will be set and the $\overline{\text{PD}}$ bit will be cleared.

Even if the flag bits were checked before executing a `SLEEP` instruction, it may be possible for flag bits to become set before the `SLEEP` instruction completes. To determine whether a `SLEEP` instruction executed, test the $\overline{\text{PD}}$ bit. If the $\overline{\text{PD}}$ bit is set, the `SLEEP` instruction was executed as a `NOP`.

To ensure that the WDT is cleared, a `CLRWDT` instruction should be executed before a `SLEEP` instruction.

FIGURE 15-12: WAKE-UP FROM SLEEP THROUGH INTERRUPT(1)



Note 1: XT, HS or LP Oscillator mode assumed.
2: TOST = 1024 TOSC (drawing not to scale). This delay will not be there for RC Oscillator mode.
3: GIE = `1` assumed. In this case, after wake-up, the processor jumps to the interrupt routine. If GIE = `0`, execution will continue in-line.
4: CLKO is not available in these oscillator modes, but shown here for timing reference.

15.14 In-Circuit Debugger

When the DEBUG bit in the Configuration Word is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® ICD. When the microcontroller has this feature enabled, some of the resources are not available for general use. Table 15-7 shows which features are consumed by the background debugger.

TABLE 15-7: DEBUGGER RESOURCES

I/O pins	RB6, RB7
Stack	1 level
Program Memory	Address 0000h must be NOP
	Last 100h words
Data Memory	0x070 (0x0F0, 0x170, 0x1F0) 0x1EB-0x1EF

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to RA5/$\overline{\text{MCLR}}$/VPP, VDD, GND, RB7 and RB6. This will interface to the In-Circuit Debugger module available from Microchip, or one of the third party development tool companies.

15.15 Program Verification/Code Protection

If the code protection bit(s) have not been programmed, the on-chip program memory can be read out for verification purposes.

15.16 ID Locations

Four memory locations (2000h-2003h) are designated as ID locations, where the user can store checksum or other code identification numbers. These locations are not accessible during normal execution but are readable and writable during program/verify. It is recommended that only the four Least Significant bits of the ID location are used.

15.17 In-Circuit Serial Programming

PIC16F87/88 microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground and the programming voltage (see Figure 15-13 for an example). This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

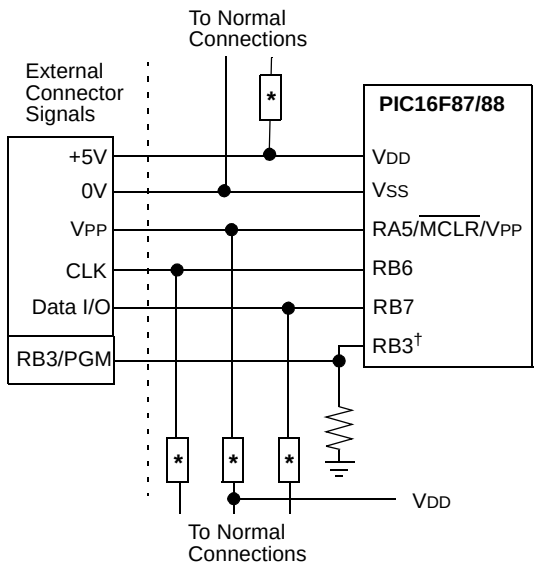
For more information on serial programming, please refer to the *"PIC16F87/88 Flash Memory Programming Specification"* (DS39607).

Note: The Timer1 oscillator shares the T1OSI and T1OSO pins with the PGD and PGC pins used for programming and debugging.

When using the Timer1 oscillator, In-Circuit Serial Programming™ (ICSP™) may not function correctly (high voltage or low voltage), or the In-Circuit Debugger (ICD) may not communicate with the controller. As a result of using either ICSP or ICD, the Timer1 crystal may be damaged.

If ICSP or ICD operations are required, the crystal should be disconnected from the circuit (disconnect either lead), or installed after programming. The oscillator loading capacitors may remain in-circuit during ICSP or ICD operation.

FIGURE 15-13: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION



* Isolation devices (as required).
† RB3 only used in LVP mode.

15.18 Low-Voltage ICSP Programming

The LVP bit of the Configuration Word enables Low-Voltage ICSP Programming. This mode allows the microcontroller to be programmed via ICSP using a V_{DD} source in the operating voltage range. This only means that V_{PP} does not have to be brought to V_{IHH}, but can instead be left at the normal operating voltage. In this mode, the RB3/PGM pin is dedicated to the programming function and ceases to be a general purpose I/O pin.

If Low-Voltage Programming mode is not used, the LVP bit can be programmed to a '0' and RB3/PGM becomes a digital I/O pin. However, the LVP bit may only be programmed when Programming mode is entered with V_{IHH} on $\overline{MCLR}$. The LVP bit can only be changed when using high voltage on $\overline{MCLR}$.

It should be noted that once the LVP bit is programmed to '0', only the High-Voltage Programming mode is available and only this mode can be used to program the device.

When using Low-Voltage ICSP, the part must be supplied at 4.5V to 5.5V if a bulk erase will be executed. This includes reprogramming of the code-protect bits from an ON state to an OFF state. For all other cases of Low-Voltage ICSP, the part may be programmed at the normal operating voltage. This means calibration values, unique user IDs or user code can be reprogrammed or added.

The following LVP steps assume the LVP bit is set in the Configuration register.

1. Apply V_{DD} to the V_{DD} pin.
2. Drive $\overline{MCLR}$ low.
3. Apply V_{DD} to the RB3/PGM pin.
4. Apply V_{DD} to the $\overline{MCLR}$ pin.
5. Follow with the associated programming steps.

Note 1: The High-Voltage Programming mode is always available, regardless of the state of the LVP bit, by applying V_{IHH} to the $\overline{MCLR}$ pin.

2: While in Low-Voltage ICSP mode (LVP = 1), the RB3 pin can no longer be used as a general purpose I/O pin.

3: When using Low-Voltage ICSP Programming (LVP) and the pull-ups on PORTB are enabled, bit 3 in the TRISB register must be cleared to disable the pull-up on RB3 and ensure the proper operation of the device.

4: RB3 should not be allowed to float if LVP is enabled. An external pull-down device should be used to default the device to normal operating mode. If RB3 floats high, the PIC16F87/88 devices will enter Programming mode.

5: LVP mode is enabled by default on all devices shipped from Microchip. It can be disabled by clearing the LVP bit in the CONFIG1 register.

6: Disabling LVP will provide maximum compatibility to other PIC16CXXX devices.

16.0 INSTRUCTION SET SUMMARY

The PIC16 instruction set is highly orthogonal and is comprised of three basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal and control** operations

Each PIC16 instruction is a 14-bit word divided into an **opcode,** which specifies the instruction type and one or more **operands,** which further specify the operation of the instruction. The formats for each of the categories are presented in Figure 16-1, while the various opcode fields are summarized in Table 16-1.

Table 16-2 lists the instructions recognized by the MPASM™ assembler. A complete description of each instruction is also available in the *"PICmicro® Mid-Range MCU Family Reference Manual"* (DS33023).

For **byte-oriented** instructions, 'f' represents a file register designator and 'd' represents a destination designator. The file register designator specifies which file register is to be used by the instruction.

The destination designator specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the W register. If 'd' is one, the result is placed in the file register specified in the instruction.

For **bit-oriented** instructions, 'b' represents a bit field designator, which selects the bit affected by the operation, while 'f' represents the address of the file in which the bit is located.

For **literal and control** operations, 'k' represents an eight or eleven-bit constant or literal value

One instruction cycle consists of four oscillator periods. For an oscillator frequency of 4 MHz, this gives a normal instruction execution time of 1 µs. All instructions are executed within a single instruction cycle, unless a conditional test is true, or the program counter is changed as a result of an instruction. When this occurs, the execution takes two instruction cycles, with the second cycle executed as a `NOP`.

Note:	To maintain upward compatibility with future PIC16F87/88 products, do not use the `OPTION` and `TRIS` instructions.

All instruction examples use the format '`0xhh`' to represent a hexadecimal number, where '`h`' signifies a hexadecimal digit.

16.1 Read-Modify-Write Operations

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (RMW) operation. The register is read, the data is modified and the result is stored according to either the instruction, or the destination designator 'd'. A read operation is performed on a register even if the instruction writes to that register.

For example, a "`CLRF PORTB`" instruction will read PORTB, clear all the data bits, then write the result back to PORTB. This example would have the unintended result that the condition that sets the RBIF flag would be cleared.

TABLE 16-1: OPCODE FIELD DESCRIPTIONS

Field	Description
`f`	Register file address (0x00 to 0x7F)
`W`	Working register (accumulator)
`b`	Bit address within an 8-bit file register
`k`	Literal field, constant data or label
`x`	Don't care location (= `0` or `1`). The assembler will generate code with x = `0`. It is the recommended form of use for compatibility with all Microchip software tools.
`d`	Destination select; d = `0`: store result in W, d = `1`: store result in file register f. Default is d = `1`.
`PC`	Program Counter
`TO`	Time-out bit
`PD`	Power-Down bit

FIGURE 16-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations

13 … 8	7	6 … 0
OPCODE	d	f (FILE #)

d = `0` for destination W
d = `1` for destination f
f = 7-bit file register address

Bit-oriented file register operations

13 … 10	9 … 7	6 … 0
OPCODE	b (BIT #)	f (FILE #)

b = 3-bit bit address
f = 7-bit file register address

Literal and control operations

General

13 … 8	7 … 0
OPCODE	k (literal)

k = 8-bit immediate value

`CALL` and `GOTO` instructions only

13 … 11	10 … 0
OPCODE	k (literal)

k = 11-bit immediate value

TABLE 16-2: PIC16F87/88 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb			LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	**f, d**	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	**f, d**	AND W with f	1	00	0101	dfff	ffff	Z	1,2
CLRF	**f**	Clear f	1	00	0001	1fff	ffff	Z	2
CLRW	**-**	Clear W	1	00	0001	0xxx	xxxx	Z	
COMF	**f, d**	Complement f	1	00	1001	dfff	ffff	Z	1,2
DECF	**f, d**	Decrement f	1	00	0011	dfff	ffff	Z	1,2
DECFSZ	**f, d**	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2,3
INCF	**f, d**	Increment f	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	**f, d**	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2,3
IORWF	**f, d**	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2
MOVF	**f, d**	Move f	1	00	1000	dfff	ffff	Z	1,2
MOVWF	**f**	Move W to f	1	00	0000	1fff	ffff		
NOP	**-**	No Operation	1	00	0000	0xx0	0000		
RLF	**f, d**	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	1,2
RRF	**f, d**	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	1,2
SUBWF	**f, d**	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z	1,2
SWAPF	**f, d**	Swap nibbles in f	1	00	1110	dfff	ffff		1,2
XORWF	**f, d**	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	**f, b**	Bit Clear f	1	01	00bb	bfff	ffff		1,2
BSF	**f, b**	Bit Set f	1	01	01bb	bfff	ffff		1,2
BTFSC	**f, b**	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		3
BTFSS	**f, b**	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		3
LITERAL AND CONTROL OPERATIONS									
ADDLW	**k**	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	**k**	AND literal with W	1	11	1001	kkkk	kkkk	Z	
CALL	**k**	Call subroutine	2	10	0kkk	kkkk	kkkk		
CLRWDT	**-**	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}$,$\overline{PD}$	
GOTO	**k**	Go to address	2	10	1kkk	kkkk	kkkk		
IORLW	**k**	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLW	**k**	Move literal to W	1	11	00xx	kkkk	kkkk		
RETFIE	**-**	Return from interrupt	2	00	0000	0000	1001		
RETLW	**k**	Return with literal in W	2	11	01xx	kkkk	kkkk		
RETURN	**-**	Return from Subroutine	2	00	0000	0000	1000		
SLEEP	**-**	Go into Standby mode	1	00	0000	0110	0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	**k**	Subtract W from literal	1	11	110x	kkkk	kkkk	C,DC,Z	
XORLW	**k**	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	Z	

Note 1: When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '`1`' for a pin configured as input and is driven low by an external device, the data will be written back with a '`0`'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = `1`), the prescaler will be cleared if assigned to the Timer0 module.

3: If the Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

Note: Additional information on the mid-range instruction set is available in the "*PICmicro® Mid-Range MCU Family Reference Manual*" (DS33023).

16.2 Instruction Descriptions

ADDLW	Add Literal and W
Syntax:	[*label*] ADDLW k
Operands:	$0 \le k \le 255$
Operation:	(W) + k → (W)
Status Affected:	C, DC, Z
Description:	The contents of the W register are added to the eight-bit literal ‘k’ and the result is placed in the W register.

ADDWF	Add W and f
Syntax:	[*label*] ADDWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) + (f) → (destination)
Status Affected:	C, DC, Z
Description:	Add the contents of the W register with register ‘f’. If ‘d’ = `0`, the result is stored in the W register. If ‘d’ = `1`, the result is stored back in register ‘f’.

ANDLW	AND Literal with W
Syntax:	[*label*] ANDLW k
Operands:	$0 \le k \le 255$
Operation:	(W) .AND. (k) → (W)
Status Affected:	Z
Description:	The contents of W register are AND’ed with the eight-bit literal ‘k’. The result is placed in the W register.

ANDWF	AND W with f
Syntax:	[*label*] ANDWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .AND. (f) → (destination)
Status Affected:	Z
Description:	AND the W register with register ‘f’. If ‘d’ = `0`, the result is stored in the W register. If ‘d’ = `1`, the result is stored back in register ‘f’.

BCF	Bit Clear f
Syntax:	[*label*] BCF f,b
Operands:	$0 \le f \le 127$ $0 \le b \le 7$
Operation:	0 → (f<b>)
Status Affected:	None
Description:	Bit ‘b’ in register ‘f’ is cleared.

BSF	Bit Set f
Syntax:	[*label*] BSF f,b
Operands:	$0 \le f \le 127$ $0 \le b \le 7$
Operation:	1 → (f<b>)
Status Affected:	None
Description:	Bit ‘b’ in register ‘f’ is set.

BTFSS	Bit Test f, Skip if Set
Syntax:	[*label*] BTFSS f,b
Operands:	$0 \le f \le 127$ $0 \le b < 7$
Operation:	skip if (f<b>) = 1
Status Affected:	None
Description:	If bit 'b' in register 'f' = 0, the next instruction is executed. If bit 'b' = 1, then the next instruction is discarded and a NOP is executed instead, making this a 2 TCY instruction.

BTFSC	Bit Test, Skip if Clear
Syntax:	[*label*] BTFSC f,b
Operands:	$0 \le f \le 127$ $0 \le b \le 7$
Operation:	skip if (f<b>) = 0
Status Affected:	None
Description:	If bit 'b' in register 'f' = 1, the next instruction is executed. If bit 'b', in register 'f', = 0, the next instruction is discarded and a NOP is executed instead, making this a 2 TCY instruction.

CALL	Call Subroutine
Syntax:	[*label*] CALL k
Operands:	$0 \le k \le 2047$
Operation:	(PC) + 1 → TOS, k → PC<10:0>, (PCLATH<4:3>) → PC<12:11>
Status Affected:	None
Description:	Call subroutine. First, return address (PC + 1) is pushed onto the stack. The eleven-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a two-cycle instruction.

CLRF	Clear f
Syntax:	[*label*] CLRF f
Operands:	$0 \le f \le 127$
Operation:	00h → (f), 1 → Z
Status Affected:	Z
Description:	The contents of register 'f' are cleared and the Z bit is set.

CLRW	Clear W
Syntax:	[*label*] CLRW
Operands:	None
Operation:	00h → (W), 1 → Z
Status Affected:	Z
Description:	W register is cleared. Zero bit (Z) is set.

CLRWDT	Clear Watchdog Timer
Syntax:	[*label*] CLRWDT
Operands:	None
Operation:	00h → WDT, 0 → WDT prescaler, 1 → $\overline{TO}$, 1 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Description:	CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.

COMF	Complement f
Syntax:	[*label*] COMF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) → (destination)
Status Affected:	Z
Description:	The contents of register 'f' are complemented. If 'd' = 0, the result is stored in W. If 'd' = 1, the result is stored back in register 'f'.

GOTO	Unconditional Branch
Syntax:	[*label*] GOTO k
Operands:	$0 \le k \le 2047$
Operation:	k → PC<10:0>, PCLATH<4:3> → PC<12:11>
Status Affected:	None
Description:	GOTO is an unconditional branch. The eleven-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a two-cycle instruction.

DECF	Decrement f
Syntax:	[*label*] DECF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) – 1 → (destination)
Status Affected:	Z
Description:	Decrement register 'f'. If 'd' = 0, the result is stored in the W register. If 'd' = 1, the result is stored back in register 'f'.

INCF	Increment f
Syntax:	[*label*] INCF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) + 1 → (destination)
Status Affected:	Z
Description:	The contents of register 'f' are incremented. If 'd' = 0, the result is placed in the W register. If 'd' = 1, the result is placed back in register 'f'.

DECFSZ	Decrement f, Skip if 0
Syntax:	[*label*] DECFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) – 1 → (destination); skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are decremented. If 'd' = 0, the result is placed in the W register. If 'd' = 1, the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', then a NOP is executed instead, making it a 2 TCY instruction.

INCFSZ	Increment f, Skip if 0
Syntax:	[*label*] INCFSZ f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) + 1 → (destination), skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are incremented. If 'd' = 0, the result is placed in the W register. If 'd' = 1, the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', a NOP is executed instead, making it a 2 TCY instruction.

IORLW	Inclusive OR Literal with W
Syntax:	[*label*] IORLW k
Operands:	$0 \le k \le 255$
Operation:	(W) .OR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are OR'ed with the eight-bit literal 'k'. The result is placed in the W register.

MOVLW	Move Literal to W
Syntax:	[*label*] MOVLW k
Operands:	$0 \le k \le 255$
Operation:	k → (W)
Status Affected:	None
Description:	The eight-bit literal 'k' is loaded into W register. The don't cares will assemble as '0's.

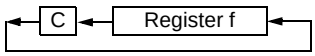
IORWF	Inclusive OR W with f
Syntax:	[*label*] IORWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .OR. (f) → (destination)
Status Affected:	Z
Description:	Inclusive OR the W register with register 'f'. If 'd' = 0, the result is placed in the W register. If 'd' = 1, the result is placed back in register 'f'.

MOVWF	Move W to f
Syntax:	[*label*] MOVWF f
Operands:	$0 \le f \le 127$
Operation:	(W) → (f)
Status Affected:	None
Description:	Move data from W register to register 'f'.

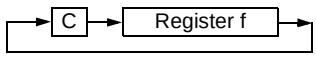
MOVF	Move f
Syntax:	[*label*] MOVF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) → (destination)
Status Affected:	Z
Description:	The contents of register 'f' are moved to a destination dependant upon the status of 'd'. If 'd' = 0, the destination is W register. If 'd' = 1, the destination is file register 'f' itself. 'd' = 1 is useful to test a file register, since status flag Z is affected.

NOP	No Operation
Syntax:	[*label*] NOP
Operands:	None
Operation:	No operation
Status Affected:	None
Description:	No operation.

RETFIE	Return from Interrupt
Syntax:	[*label*] RETFIE
Operands:	None
Operation:	TOS → PC, 1 → GIE
Status Affected:	None

RLF	Rotate Left f through Carry
Syntax:	[*label*] RLF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	See description below
Status Affected:	C
Description:	The contents of register 'f' are rotated one bit to the left through the Carry flag. If 'd' = `0`, the result is placed in the W register. If 'd' = `1`, the result is stored back in register 'f'. 

RETLW	Return with Literal in W
Syntax:	[*label*] RETLW k
Operands:	$0 \le k \le 255$
Operation:	k → (W); TOS → PC
Status Affected:	None
Description:	The W register is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.

RRF	Rotate Right f through Carry
Syntax:	[*label*] RRF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	See description below
Status Affected:	C
Description:	The contents of register 'f' are rotated one bit to the right through the Carry flag. If 'd' = `0`, the result is placed in the W register. If 'd' = `1`, the result is placed back in register 'f'. 

RETURN	Return from Subroutine
Syntax:	[*label*] RETURN
Operands:	None
Operation:	TOS → PC
Status Affected:	None
Description:	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.

SLEEP	Sleep
Syntax:	[*label*] SLEEP
Operands:	None
Operation:	00h → WDT, 0 → WDT prescaler, 1 → $\overline{TO}$, 0 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Description:	The Power-Down status bit, $\overline{PD}$, is cleared. Time-out status bit, $\overline{TO}$, is set. Watchdog Timer and its prescaler are cleared. The processor is put into Sleep mode with the oscillator stopped.

SUBLW	Subtract W from Literal
Syntax:	[*label*] SUBLW k
Operands:	$0 \le k \le 255$
Operation:	k – (W) → (W)
Status Affected:	C, DC, Z
Description:	The W register is subtracted (two's complement method) from the eight-bit literal 'k'. The result is placed in the W register.

SUBWF	Subtract W from f
Syntax:	[*label*] SUBWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) – (W) → (destination)
Status Affected:	C, DC, Z
Description:	Subtract (two's complement method) W register from register 'f'. If 'd' = `0`, the result is stored in the W register. If 'd' = `1`, the result is stored back in register 'f'.

SWAPF	Swap Nibbles in f
Syntax:	[*label*] SWAPF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f<3:0>) → (destination<7:4>), (f<7:4>) → (destination<3:0>)
Status Affected:	None
Description:	The upper and lower nibbles of register 'f' are exchanged. If 'd' = `0`, the result is placed in W register. If 'd' = `1`, the result is placed in register 'f'.

XORLW	Exclusive OR Literal with W
Syntax:	[*label*] XORLW k
Operands:	$0 \le k \le 255$
Operation:	(W) .XOR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are XOR'ed with the eight-bit literal 'k'. The result is placed in the W register.

XORWF	Exclusive OR W with f
Syntax:	[*label*] XORWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .XOR. (f) → (destination)
Status Affected:	Z
Description:	Exclusive OR the contents of the W register with register 'f'. If 'd' = `0`, the result is stored in the W register. If 'd' = `1`, the result is stored back in register 'f'.

17.0 DEVELOPMENT SUPPORT

The PICmicro® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
  - MPASM™ Assembler
  - MPLAB C17 and MPLAB C18 C Compilers
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB C30 C Compiler
  - MPLAB ASM30 Assembler/Linker/Library
- Simulators
  - MPLAB SIM Software Simulator
  - MPLAB dsPIC30 Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - MPLAB ICE 4000 In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD 2
- Device Programmers
  - PRO MATE® II Universal Device Programmer
  - PICSTART® Plus Development Programmer
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration Boards
  - PICDEM™ 1 Demonstration Board
  - PICDEM.net™ Demonstration Board
  - PICDEM 2 Plus Demonstration Board
  - PICDEM 3 Demonstration Board
  - PICDEM 4 Demonstration Board
  - PICDEM 17 Demonstration Board
  - PICDEM 18R Demonstration Board
  - PICDEM LIN Demonstration Board
  - PICDEM USB Demonstration Board
- Evaluation Kits
  - KEELOQ® Evaluation and Programming Tools
  - PICDEM MSC
  - microID® Developer Kits
  - CAN
  - PowerSmart® Developer Kits
  - Analog

17.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16-bit microcontroller market. The MPLAB IDE is a Windows® based application that contains:

- An interface to debugging tools
  - simulator
  - programmer (sold separately)
  - emulator (sold separately)
  - in-circuit debugger (sold separately)
- A full-featured editor with color coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Mouse over variable inspection
- Extensive on-line help

The MPLAB IDE allows you to:

- Edit your source files (either assembly or C)
- One touch assemble (or compile) and download to PICmicro emulator and simulator tools (automatically updates all project information)
- Debug using:
  - source files (assembly or C)
  - mixed assembly and C
  - machine code

MPLAB IDE supports multiple debugging tools in a single development paradigm, from the cost effective simulators, through low-cost in-circuit debuggers, to full-featured emulators. This eliminates the learning curve when upgrading to tools with increasing flexibility and power.

17.2 MPASM Assembler

The MPASM assembler is a full-featured, universal macro assembler for all PICmicro MCUs.

The MPASM assembler generates relocatable object files for the MPLINK object linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code and COFF files for debugging.

The MPASM assembler features include:

- Integration into MPLAB IDE projects
- User defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

17.3 MPLAB C17 and MPLAB C18 C Compilers

The MPLAB C17 and MPLAB C18 Code Development Systems are complete ANSI C compilers for Microchip's PIC17CXXX and PIC18CXXX family of microcontrollers. These compilers provide powerful integration capabilities, superior code optimization and ease of use not found with other compilers.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

17.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK object linker combines relocatable objects created by the MPASM assembler and the MPLAB C17 and MPLAB C18 C compilers. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB object librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

17.5 MPLAB C30 C Compiler

The MPLAB C30 C compiler is a full-featured, ANSI compliant, optimizing compiler that translates standard ANSI C programs into dsPIC30F assembly language source. The compiler also supports many command line options and language extensions to take full advantage of the dsPIC30F device hardware capabilities and afford fine control of the compiler code generator.

MPLAB C30 is distributed with a complete ANSI C standard library. All library functions have been validated and conform to the ANSI C library standard. The library includes functions for string manipulation, dynamic memory allocation, data conversion, time-keeping and math functions (trigonometric, exponential and hyperbolic). The compiler provides symbolic information for high-level source debugging with the MPLAB IDE.

17.6 MPLAB ASM30 Assembler, Linker and Librarian

MPLAB ASM30 assembler produces relocatable machine code from symbolic assembly language for dsPIC30F devices. MPLAB C30 compiler uses the assembler to produce it's object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire dsPIC30F instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

17.7 MPLAB SIM Software Simulator

The MPLAB SIM software simulator allows code development in a PC hosted environment by simulating the PICmicro series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user defined key press, to any pin. The execution can be performed in Single-Step, Execute Until Break or Trace mode.

The MPLAB SIM simulator fully supports symbolic debugging using the MPLAB C17 and MPLAB C18 C Compilers, as well as the MPASM assembler. The software simulator offers the flexibility to develop and debug code outside of the laboratory environment, making it an excellent, economical software development tool.

17.8 MPLAB SIM30 Software Simulator

The MPLAB SIM30 software simulator allows code development in a PC hosted environment by simulating the dsPIC30F series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file, or user defined key press, to any of the pins.

The MPLAB SIM30 simulator fully supports symbolic debugging using the MPLAB C30 C Compiler and MPLAB ASM30 assembler. The simulator runs in either a Command Line mode for automated tasks, or from MPLAB IDE. This high-speed simulator is designed to debug, analyze and optimize time intensive DSP routines.

17.9 MPLAB ICE 2000 High-Performance Universal In-Circuit Emulator

The MPLAB ICE 2000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro microcontrollers. Software control of the MPLAB ICE 2000 in-circuit emulator is advanced by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 2000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB ICE in-circuit emulator allows expansion to support new PICmicro microcontrollers.

The MPLAB ICE 2000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft® Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

17.10 MPLAB ICE 4000 High-Performance Universal In-Circuit Emulator

The MPLAB ICE 4000 universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for high-end PICmicro microcontrollers. Software control of the MPLAB ICE in-circuit emulator is provided by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICD 4000 is a premium emulator system, providing the features of MPLAB ICE 2000, but with increased emulation memory and high-speed performance for dsPIC30F and PIC18XXXX devices. Its advanced emulator features include complex triggering and timing, up to 2 Mb of emulation memory and the ability to view variables in real-time.

The MPLAB ICE 4000 in-circuit emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

17.11 MPLAB ICD 2 In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD 2, is a powerful, low-cost, run-time development tool, connecting to the host PC via an RS-232 or high-speed USB interface. This tool is based on the Flash PICmicro MCUs and can be used to develop for these and other PICmicro microcontrollers. The MPLAB ICD 2 utilizes the in-circuit debugging capability built into the Flash devices. This feature, along with Microchip's In-Circuit Serial Programming™ (ICSP™) protocol, offers cost effective in-circuit Flash debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by setting breakpoints, single-stepping and watching variables, CPU status and peripheral registers. Running at full speed enables testing hardware and applications in real-time. MPLAB ICD 2 also serves as a development programmer for selected PICmicro devices.

17.12 PRO MATE II Universal Device Programmer

The PRO MATE II is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features an LCD display for instructions and error messages and a modular detachable socket assembly to support various package types. In Stand-Alone mode, the PRO MATE II device programmer can read, verify and program PICmicro devices without a PC connection. It can also set code protection in this mode.

17.13 MPLAB PM3 Device Programmer

The MPLAB PM3 is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular detachable socket assembly to support various package types. The ICSP™ cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 device programmer can read, verify and program PICmicro devices without a PC connection. It can also set code protection in this mode. MPLAB PM3 connects to the host PC via an RS-232 or USB cable. MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices and incorporates an SD/MMC card for file storage and secure data applications.

17.14 PICSTART Plus Development Programmer

The PICSTART Plus development programmer is an easy-to-use, low-cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient. The PICSTART Plus development programmer supports most PICmicro devices up to 40 pins. Larger pin count devices, such as the PIC16C92X and PIC17C76X, may be supported with an adapter socket. The PICSTART Plus development programmer is CE compliant.

17.15 PICDEM 1 PICmicro Demonstration Board

The PICDEM 1 demonstration board demonstrates the capabilities of the PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The sample microcontrollers provided with the PICDEM 1 demonstration board can be programmed with a PRO MATE II device programmer or a PICSTART Plus development programmer. The PICDEM 1 demonstration board can be connected to the MPLAB ICE in-circuit emulator for testing. A prototype area extends the circuitry for additional application components. Features include an RS-232 interface, a potentiometer for simulated analog input, push button switches and eight LEDs.

17.16 PICDEM.net Internet/Ethernet Demonstration Board

The PICDEM.net demonstration board is an Internet/Ethernet demonstration board using the PIC18F452 microcontroller and TCP/IP firmware. The board supports any 40-pin DIP device that conforms to the standard pinout used by the PIC16F877 or PIC18C452. This kit features a user friendly TCP/IP stack, web server with HTML, a 24L256 Serial EEPROM for Xmodem download to web pages into Serial EEPROM, ICSP/MPLAB ICD 2 interface connector, an Ethernet interface, RS-232 interface and a 16 x 2 LCD display. Also included is the book and CD-ROM *"TCP/IP Lean, Web Servers for Embedded Systems,"* by Jeremy Bentham

17.17 PICDEM 2 Plus Demonstration Board

The PICDEM 2 Plus demonstration board supports many 18, 28 and 40-pin microcontrollers, including PIC16F87X and PIC18FXX2 devices. All the necessary hardware and software is included to run the demonstration programs. The sample microcontrollers provided with the PICDEM 2 demonstration board can be programmed with a PRO MATE II device programmer, PICSTART Plus development programmer, or MPLAB ICD 2 with a Universal Programmer Adapter. The MPLAB ICD 2 and MPLAB ICE in-circuit emulators may also be used with the PICDEM 2 demonstration board to test firmware. A prototype area extends the circuitry for additional application components. Some of the features include an RS-232 interface, a 2 x 16 LCD display, a piezo speaker, an on-board temperature sensor, four LEDs and sample PIC18F452 and PIC16F877 Flash microcontrollers.

17.18 PICDEM 3 PIC16C92X Demonstration Board

The PICDEM 3 demonstration board supports the PIC16C923 and PIC16C924 in the PLCC package. All the necessary hardware and software is included to run the demonstration programs.

17.19 PICDEM 4 8/14/18-Pin Demonstration Board

The PICDEM 4 can be used to demonstrate the capabilities of the 8, 14 and 18-pin PIC16XXXX and PIC18XXXX MCUs, including the PIC16F818/819, PIC16F87/88, PIC16F62XA and the PIC18F1320 family of microcontrollers. PICDEM 4 is intended to showcase the many features of these low pin count parts, including LIN and Motor Control using ECCP. Special provisions are made for low-power operation with the supercapacitor circuit and jumpers allow on-board hardware to be disabled to eliminate current draw in this mode. Included on the demo board are provisions for Crystal, RC or Canned Oscillator modes, a five volt regulator for use with a nine volt wall adapter or battery, DB-9 RS-232 interface, ICD connector for programming via ICSP and development with MPLAB ICD 2, 2 x 16 liquid crystal display, PCB footprints for H-Bridge motor driver, LIN transceiver and EEPROM. Also included are: header for expansion, eight LEDs, four potentiometers, three push buttons and a prototyping area. Included with the kit is a PIC16F627A and a PIC18F1320. Tutorial firmware is included along with the User's Guide.

17.20 PICDEM 17 Demonstration Board

The PICDEM 17 demonstration board is an evaluation board that demonstrates the capabilities of several Microchip microcontrollers, including PIC17C752, PIC17C756A, PIC17C762 and PIC17C766. A programmed sample is included. The PRO MATE II device programmer, or the PICSTART Plus development programmer, can be used to reprogram the device for user tailored application development. The PICDEM 17 demonstration board supports program download and execution from external on-board Flash memory. A generous prototype area is available for user hardware expansion.

17.21 PICDEM 18R PIC18C601/801 Demonstration Board

The PICDEM 18R demonstration board serves to assist development of the PIC18C601/801 family of Microchip microcontrollers. It provides hardware implementation of both 8-bit Multiplexed/Demultiplexed and 16-bit Memory modes. The board includes 2 Mb external Flash memory and 128 Kb SRAM memory, as well as serial EEPROM, allowing access to the wide range of memory types supported by the PIC18C601/801.

17.22 PICDEM LIN PIC16C43X Demonstration Board

The powerful LIN hardware and software kit includes a series of boards and three PICmicro microcontrollers. The small footprint PIC16C432 and PIC16C433 are used as slaves in the LIN communication and feature on-board LIN transceivers. A PIC16F874 Flash microcontroller serves as the master. All three microcontrollers are programmed with firmware to provide LIN bus communication.

17.23 PICkit™ 1 Flash Starter Kit

A complete “development system in a box”, the PICkit™ Flash Starter Kit includes a convenient multi-section board for programming, evaluation and development of 8/14-pin Flash PIC® microcontrollers. Powered via USB, the board operates under a simple Windows GUI. The PICkit 1 Starter Kit includes the User’s Guide (on CD ROM), PICkit 1 tutorial software and code for various applications. Also included are MPLAB® IDE (Integrated Development Environment) software, software and hardware “Tips 'n Tricks for 8-pin Flash PIC® Microcontrollers” Handbook and a USB interface cable. Supports all current 8/14-pin Flash PIC microcontrollers, as well as many future planned devices.

17.24 PICDEM USB PIC16C7X5 Demonstration Board

The PICDEM USB Demonstration Board shows off the capabilities of the PIC16C745 and PIC16C765 USB microcontrollers. This board provides the basis for future USB products.

17.25 Evaluation and Programming Tools

In addition to the PICDEM series of circuits, Microchip has a line of evaluation kits and demonstration software for these products.

- KEELOQ evaluation and programming tools for Microchip’s HCS Secure Data Products
- CAN developers kit for automotive network applications
- Analog design boards and filter design software
- PowerSmart battery charging evaluation/ calibration kits
- IrDA® development kit
- microID development and rfLab™ development software
- SEEVAL® designer kit for memory evaluation and endurance calculations
- PICDEM MSC demo boards for Switching mode power supply, high-power IR driver, delta sigma ADC and flow rate sensor

Check the Microchip web page and the latest Product Selector Guide for the complete list of demonstration and evaluation kits.

NOTES:

18.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings †

Ambient temperature under bias ..... -40°C to +125°C

Storage temperature ..... -65°C to +150°C

Voltage on any pin with respect to V_{SS} (except V_{DD} and $\overline{MCLR}$) ..... -0.3V to (V_{DD} + 0.3V)

Voltage on V_{DD} with respect to V_{SS} ..... -0.3 to +7.5V

Voltage on $\overline{MCLR}$ with respect to V_{SS} **(Note 2)** ..... -0.3 to +14V

Total power dissipation **(Note 1)** 1W

Maximum current out of V_{SS} pin ..... 200 mA

Maximum current into V_{DD} pin ..... 200 mA

Input clamp current, I_{IK} (V_I < 0 or V_I > V_{DD}) ..... ±20 mA

Output clamp current, I_{OK} (V_O < 0 or V_O > V_{DD}) ..... ±20 mA

Maximum output current sunk by any I/O pin ..... 25 mA

Maximum output current sourced by any I/O pin ..... 25 mA

Maximum current sunk by PORTA ..... 100 mA

Maximum current sourced by PORTA ..... 100 mA

Maximum current sunk by PORTB ..... 100 mA

Maximum current sourced by PORTB ..... 100 mA

Note 1: Power dissipation is calculated as follows: $P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum(V_{OL} \times I_{OL})$

2: Voltage spikes at the $\overline{MCLR}$ pin may cause latch-up. A series resistor of greater than 1 kΩ should be used to pull $\overline{MCLR}$ to V_{DD}, rather than tying the pin directly to V_{DD}.

† NOTICE: Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

FIGURE 18-1: PIC16F87/88 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL, EXTENDED)

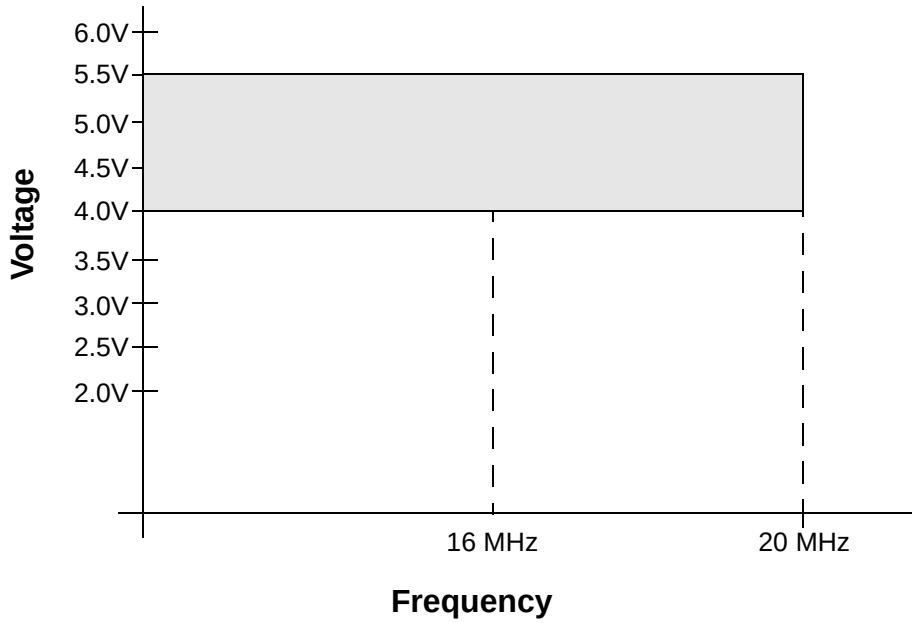
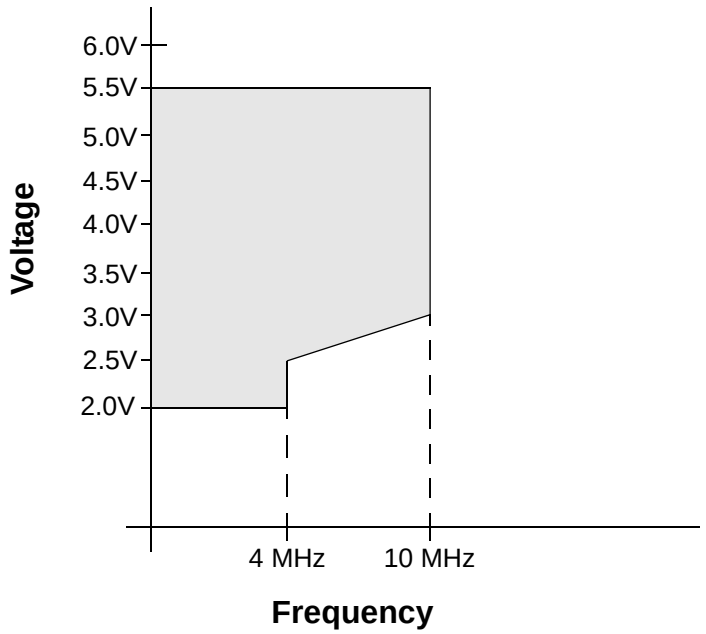


FIGURE 18-2: PIC16LF87/88 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)



F_{MAX} = (12 MHz/V) ($V_{DDAPPMIN}$ – 2.5V) + 4 MHz

Note 1: $V_{DDAPPMIN}$ is the minimum voltage of the PICmicro® device in the application.

2: F_{MAX} has a maximum frequency of 10 MHz.

18.1 DC Characteristics: Supply Voltage PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial)

PIC16LF87/88 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC16F87/88 (Industrial, Extended)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
	VDD	**Supply Voltage**					
D001		PIC16LF87/88	2.0	—	5.5	V	HS, XT, RC and LP Oscillator mode
D001		PIC16F87/88	4.0	—	5.5	V	
D002	VDR	**RAM Data Retention Voltage(1)**	1.5	—	—	V	
D003	VPOR	**VDD Start Voltage** to ensure internal Power-on Reset signal	—	—	0.7	V	See **Section 15.4 "Power-on Reset (POR)"** for details
D004	SVDD	**VDD Rise Rate** to ensure internal Power-on Reset signal	0.05	—	—	V/ms	See **Section 15.4 "Power-on Reset (POR)"** for details
	VBOR	**Brown-out Reset Voltage**					
D005		PIC16LF87/88	3.65	—	4.35	V	
D005		PIC16F87/88	3.65	—	4.35	V	FMAX = 14 MHz(2)

Legend: Shading of rows is to assist in readability of the table.

Note 1: This is the limit to which VDD can be lowered in Sleep mode, or during a device Reset, without losing RAM data.

2: When BOR is enabled, the device will operate correctly until the VBOR voltage trip point is reached.

18.2 DC Characteristics: Power-Down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial)

PIC16LF87/88 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC16F87/88 (Industrial, Extended)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Device	Typ	Max	Units	Conditions		
	Power-Down Current (IPD)(1)						
	PIC16LF87/88	0.1	0.4	μA	-40°C	VDD = 2.0V	
		0.1	0.4	μA	+25°C		
		0.4	1.5	μA	+85°C		
	PIC16LF87/88	0.3	0.5	μA	-40°C	VDD = 3.0V	
		0.3	0.5	μA	+25°C		
		0.7	1.7	μA	+85°C		
	All devices	0.6	1.0	μA	-40°C	VDD = 5.0V	
		0.6	1.0	μA	+25°C		
		1.2	5.0	μA	+85°C		
	Extended devices	6	28	μA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

18.2 DC Characteristics: Power-Down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

PIC16LF87/88 (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC16F87/88 (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2,3)						
	PIC16LF87/88	9	20	μA	-40°C	VDD = 2.0V	FOSC = 32 kHz (LP Oscillator)
		7	15	μA	+25°C		
		7	15	μA	+85°C		
	PIC16LF87/88	16	30	μA	-40°C	VDD = 3.0V	
		14	25	μA	+25°C		
		14	25	μA	+85°C		
	All devices	32	40	μA	-40°C	VDD = 5.0V	
		26	35	μA	+25°C		
		26	35	μA	+85°C		
	Extended Devices	35	53	μA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

18.2 DC Characteristics: Power-Down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

PIC16LF87/88 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC16F87/88 (Industrial, Extended)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Device	Typ	Max	Units	Conditions		
	Supply Current (IDD)(2,3)						
	PIC16LF87/88	72	95	μA	-40°C	VDD = 2.0V	FOSC = 1 MHz (RC Oscillator)(3)
		76	90	μA	+25°C		
		76	90	μA	+85°C		
	PIC16LF87/88	138	175	μA	-40°C	VDD = 3.0V	
		136	170	μA	+25°C		
		136	170	μA	+85°C		
	All devices	310	380	μA	-40°C	VDD = 5.0V	
		290	360	μA	+25°C		
		280	360	μA	+85°C		
	Extended devices	330	500	μA	125°C		
	PIC16LF87/88	270	335	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz (RC Oscillator)(3)
		280	330	μA	+25°C		
		285	330	μA	+85°C		
	PIC16LF87/88	460	610	μA	-40°C	VDD = 3.0V	
		450	600	μA	+25°C		
		450	600	μA	+85°C		
	All devices	900	1060	μA	-40°C	VDD = 5.0V	
		890	1050	μA	+25°C		
		890	1050	μA	+85°C		
	Extended devices	.920	1.5	mA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

18.2 DC Characteristics: Power-Down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

PIC16LF87/88 (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC16F87/88 (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2,3)						
	All devices	1.8	2.3	mA	-40°C	VDD = 4.0V	FOSC = 20 MHz (HS Oscillator)
		1.6	2.2	mA	+25°C		
		1.3	2.2	mA	+85°C		
	All devices	3.0	4.2	mA	-40°C	VDD = 5.0V	
		2.5	4.0	mA	+25°C		
		2.5	4.0	mA	+85°C		
	Extended devices	3.0	5.0	mA	+85°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

18.2 DC Characteristics: Power-Down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

PIC16LF87/88 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC16F87/88 (Industrial, Extended)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Device	Typ	Max	Units	Conditions		
	Supply Current (IDD)(2,3)						
	PIC16LF87/88	8	20	µA	-40°C	VDD = 2.0V	FOSC = 31.25 kHz (**RC_RUN** mode, Internal RC Oscillator)
		7	15	µA	+25°C		
		7	15	µA	+85°C		
	PIC16LF87/88	16	30	µA	-40°C	VDD = 3.0V	
		14	25	µA	+25°C		
		14	25	µA	+85°C		
	All devices	32	40	µA	-40°C	VDD = 5.0V	
		29	35	µA	+25°C		
		29	35	µA	+85°C		
	Extended devices	35	45	µA	+125°C		
	PIC16LF87/88	132	160	µA	-40°C	VDD = 2.0V	FOSC = 1 MHz (**RC_RUN** mode, Internal RC Oscillator)
		126	155	µA	+25°C		
		126	155	µA	+85°C		
	PIC16LF87/88	260	310	µA	-40°C	VDD = 3.0V	
		230	300	µA	+25°C		
		230	300	µA	+85°C		
	All devices	560	690	µA	-40°C	VDD = 5.0V	
		500	650	µA	+25°C		
		500	650	µA	+85°C		
	Extended devices	570	710	µA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{\text{MCLR}}$ = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

18.2 DC Characteristics: Power-Down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

PIC16LF87/88 (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC16F87/88 (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2,3)						
	PIC16LF87/88	310	420	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz (**RC_RUN** mode, Internal RC Oscillator)
		300	410	μA	+25°C		
		300	410	μA	+85°C		
	PIC16LF87/88	550	650	μA	-40°C	VDD = 3.0V	
		530	620	μA	+25°C		
		530	620	μA	+85°C		
	All devices	1.2	1.5	mA	-40°C	VDD = 5.0V	
		1.1	1.4	mA	+25°C		
		1.1	1.4	mA	+85°C		
	Extended devices	1.3	1.6	mA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

18.2 DC Characteristics: Power-Down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

PIC16LF87/88 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC16F87/88 (Industrial, Extended)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Device	Typ	Max	Units	Conditions		
	Supply Current (IDD)(2,3)						
	PIC16LF87/88	.950	1.3	mA	-40°C	VDD = 3.0V	FOSC = 8 MHz (**RC_RUN** mode, Internal RC Oscillator)
		.930	1.2	mA	+25°C		
		.930	1.2	mA	+85°C		
	All devices	1.8	3.0	mA	-40°C	VDD = 5.0V	
		1.7	2.8	mA	+25°C		
		1.7	2.8	mA	+85°C		
	Extended devices	2.0	4.0	mA	+125°C		
	PIC16LF87/88	9	13	µA	-10°C	VDD = 2.0V	FOSC = 32 kHz (**SEC_RUN** mode, Timer1 as clock)
		9	14	µA	+25°C		
		11	16	µA	+70°C		
	PIC16LF87/88	12	34	µA	-10°C	VDD = 3.0V	
		12	31	µA	+25°C		
		14	28	µA	+70°C		
	All devices	20	72	µA	-10°C	VDD = 5.0V	
		20	65	µA	+25°C		
		25	59	µA	+70°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

18.2 DC Characteristics: Power-Down and Supply Current PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

PIC16LF87/88 (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC16F87/88 (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
D022 (ΔIWDT)	**Module Differential Currents (ΔIWDT, ΔIBOR, ΔILVD, ΔIOSCB, ΔIAD)**						
	Watchdog Timer	1.5	3.8	μA	-40°C	VDD = 2.0V	
		2.2	3.8	μA	+25°C		
		2.7	4.0	μA	+85°C		
		2.3	4.6	μA	-40°C	VDD = 3.0V	
		2.7	4.6	μA	+25°C		
		3.1	4.8	μA	+85°C		
		3.0	10.0	μA	-40°C	VDD = 5.0V	
		3.3	10.0	μA	+25°C		
		3.9	13.0	μA	+85°C		
	Extended devices	5.0	21.0	μA	+125°C		
D022A (ΔIBOR)	**Brown-out Reset**	40	60	μA	-40°C to +85°C	VDD = 5.0V	
D025 (ΔIOSCB)	**Timer1 Oscillator**	1.7	2.3	μA	-40°C	VDD = 2.0V	32 kHz on Timer1
		1.8	2.3	μA	+25°C		
		2.0	2.3	μA	+85°C		
		2.2	3.8	μA	-40°C	VDD = 3.0V	
		2.6	3.8	μA	+25°C		
		2.9	3.8	μA	+85°C		
		3.0	6.0	μA	-40°C	VDD = 5.0V	
		3.2	6.0	μA	+25°C		
		3.4	7.0	μA	+85°C		
D026 (ΔIAD)	**A/D Converter**	0.001	2.0	μA	-40°C to +85°C	VDD = 2.0V	A/D on, Sleep, not converting
		0.001	2.0	μA	-40°C to +85°C	VDD = 3.0V	
		0.003	2.0	μA	-40°C to +85°C	VDD = 5.0V	
	Extended devices	4.0	8.0	μA	-40°C to +125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

18.3 DC Characteristics: Internal RC Accuracy PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial)

PIC16LF87/88 (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC16F87/88 (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Min**	**Typ**	**Max**	**Units**	**Conditions**	
	INTOSC Accuracy @ Freq = 8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz(1)						
	PIC16LF87/88	-2	±1	2	%	+25°C	VDD = 2.7-3.3V
		-5	—	5	%	-10°C to +85°C	
		-10	—	10	%	-40°C to +85°C	
	PIC16F87/88	-2	±1	2	%	25°C	VDD = 4.5-5.5V
		-5	—	5	%	-10°C to +85°C	
		-10	—	10	%	-40°C to +85°C	
	Extended devices	-15	—	15	%	-40°C to +125°C	VDD = 4.5-5.5V
	INTRC Accuracy @ Freq = 31 kHz(2)						
	PIC16LF87/88	26.562	—	35.938	kHz	-40°C to +85°C	VDD = 2.7-3.3V
	PIC16F87/88	26.562	—	35.938	kHz	-40°C to +85°C	VDD = 4.5-5.5V

Legend: Shading of rows is to assist in readability of the table.

Note 1: Frequency calibrated at 25°C. OSCTUNE register can be used to compensate for temperature drift.

2: INTRC frequency after calibration.

18.4 DC Characteristics: PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended Operating voltage VDD range as described in DC Specification, **Section 18.1 "DC Characteristics: Supply Voltage"**.				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
	VIL	**Input Low Voltage**					
		I/O ports:					
D030		with TTL buffer	VSS	—	0.15 VDD	V	For entire VDD range
D030A			VSS	—	0.8V	V	4.5V ≤ VDD ≤ 5.5V
D031		with Schmitt Trigger buffer	VSS	—	0.2 VDD	V	
D032		$\overline{\text{MCLR}}$, OSC1 (in RC mode)	VSS	—	0.2 VDD	V	**(Note 1)**
D033		OSC1 (in XT and LP mode)	VSS	—	0.3V	V	
		OSC1 (in HS mode)	VSS	—	0.3 VDD	V	
		Ports RB1 and RB4:					
D034		with Schmitt Trigger buffer	VSS	—	0.3 VDD	V	For entire VDD range
	VIH	**Input High Voltage**					
		I/O ports:					
D040		with TTL buffer	2.0	—	VDD	V	4.5V ≤ VDD ≤ 5.5V
D040A			0.25 VDD + 0.8V	—	VDD	V	For entire VDD range
D041		with Schmitt Trigger buffer	0.8 VDD	—	VDD	V	For entire VDD range
D042		$\overline{\text{MCLR}}$	0.8 VDD	—	VDD	V	
D042A		OSC1 (in XT and LP mode)	1.6V	—	VDD	V	
		OSC1 (in HS mode)	0.7 VDD	—	VDD	V	
D043		OSC1 (in RC mode)	0.9 VDD	—	VDD	V	**(Note 1)**
		Ports RB1 and RB4:					
D044		with Schmitt Trigger buffer	0.7 VDD	—	VDD	V	For entire VDD range
D070	IPURB	**PORTB Weak Pull-up Current**	50	250	400	μA	VDD = 5V, VPIN = VSS
	IIL	**Input Leakage Current (Notes 2, 3)**					
D060		I/O ports	—	—	±1	μA	Vss ≤ VPIN ≤ VDD, pin at high-impedance
D061		MCLR	—	—	±5	μA	Vss ≤ VPIN ≤ VDD
D063		OSC1	—	—	±5	μA	Vss ≤ VPIN ≤ VDD, XT, HS and LP oscillator configuration

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC16F87/88 be driven with external clock in RC mode.

2: The leakage current on the $\overline{\text{MCLR}}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as current sourced by the pin.

18.4 DC Characteristics: PIC16F87/88 (Industrial, Extended) PIC16LF87/88 (Industrial) (Continued)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended Operating voltage VDD range as described in DC Specification, **Section 18.1 "DC Characteristics: Supply Voltage"**.				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
	VOL	**Output Low Voltage**					
D080		I/O ports	—	—	0.6	V	IOL = 8.5 mA, VDD = 4.5V, -40°C to +125°C
D083		OSC2/CLKO (RC oscillator configuration)	—	—	0.6	V	IOL = 1.6 mA, VDD = 4.5V, -40°C to +125°C
	VOH	**Output High Voltage**					
D090		I/O ports **(Note 3)**	VDD – 0.7	—	—	V	IOH = -3.0 mA, VDD = 4.5V, -40°C to +125°C
D092		OSC2/CLKO (RC oscillator configuration)	VDD – 0.7	—	—	V	IOH = -1.3 mA, VDD = 4.5V, -40°C to +125°C
		Capacitive Loading Specs on Output Pins					
D100	COSC2	OSC2 pin	—	—	15	pF	In XT, HS and LP modes when external clock is used to drive OSC1
D101	CIO	All I/O pins and OSC2 (in RC mode)	—	—	50	pF	
D102	CB	SCL, SDA in I²C™ mode	—	—	400	pF	
		Data EEPROM Memory					
D120	ED	Endurance	100K 10K	1M 100K	— —	E/W E/W	-40°C to 85°C +85°C to +125°C
D121	VDRW	VDD for Read/Write	VMIN	—	5.5	V	Using EECON to read/write, VMIN = min. operating voltage
D122	TDEW	Erase/Write Cycle Time	—	4	8	ms	
		Program Flash Memory					
D130	EP	Endurance	10K 1K	100K 10K	— —	E/W E/W	-40°C to 85°C +85°C to +125°C
D131	VPR	VDD for Read	VMIN	—	5.5	V	
D132A		VDD for Erase/Write	VMIN	—	5.5	V	Using EECON to read/write, VMIN = min. operating voltage
D133	TPE	Erase Cycle Time	—	2	4	ms	
D134	TPW	Write Cycle Time	—	2	4	ms	

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC16F87/88 be driven with external clock in RC mode.

2: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as current sourced by the pin.

TABLE 18-1: COMPARATOR SPECIFICATIONS

Operating Conditions: 3.0V < V_{DD} < 5.5V, -40°C < T_A < +85°C, unless otherwise stated							
Param No.	**Sym**	**Characteristics**	**Min**	**Typ**	**Max**	**Units**	**Comments**
D300	V_{IOFF}	Input Offset Voltage	—	±5.0	±10	mV	
D301	V_{ICM}	Input Common Mode Voltage*	0	—	V_{DD} – 1.5	V	
D302	CMRR	Common Mode Rejection Ratio*	55	—	—	dB	
300 300A	T_{RESP}	Response Time(1)*	—	150	400 600	ns ns	PIC16F87/88 PIC16LF87/88
301	T_{MC2OV}	Comparator Mode Change to Output Valid*	—	—	10	µs	

* These parameters are characterized but not tested.

Note 1: Response time measured with one comparator input at (V_{DD} – 1.5)/2 while the other input transitions from V_{SS} to V_{DD}.

TABLE 18-2: VOLTAGE REFERENCE SPECIFICATIONS

Operating Conditions: 3.0V < V_{DD} < 5.5V, -40°C < T_A < +85°C, unless otherwise stated							
Spec No.	**Sym**	**Characteristics**	**Min**	**Typ**	**Max**	**Units**	**Comments**
D310	V_{RES}	Resolution	V_{DD}/24	—	V_{DD}/32	LSb	
D311	V_{RAA}	Absolute Accuracy	— —	— —	1/2 1/2	LSb LSb	Low Range (CVRR = 1) High Range (CVRR = 0)
D312	V_{RUR}	Unit Resistor Value (R)*	—	2k	—	Ω	
310	T_{SET}	Settling Time(1)*	—	—	10	µs	

* These parameters are characterized but not tested.

Note 1: Settling time measured while CVRR = 1 and CVR<3:0> transitions from '0000' to '1111'.

18.5 Timing Parameter Symbology

The timing parameter symbols have been created using one of the following formats:

1. TppS2ppS
2. TppS
3. TCC:ST (I²C specifications only)
4. Ts (I²C specifications only)

T			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

pp			
cc	CCP1	osc	OSC1
ck	CLKO	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{SS}$
dt	Data in	t0	T0CKI
io	I/O port	t1	T1CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

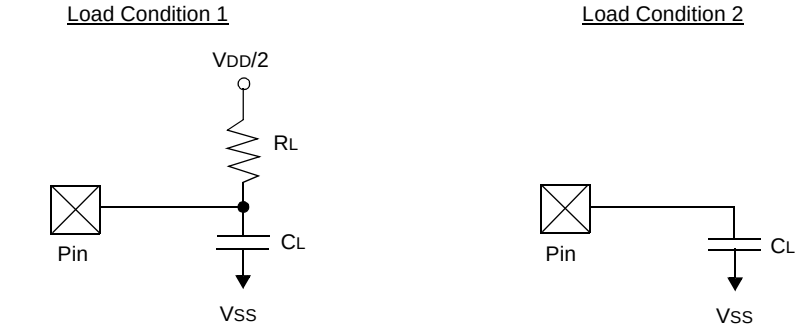
Uppercase letters and their meanings:

S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (High-impedance)	V	Valid
L	Low	Z	High-impedance
I²C only			
AA	output access	High	High
BUF	Bus free	Low	Low

TCC:ST (I²C specifications only)

CC			
HD	Hold	SU	Setup
ST			
DAT	DATA input hold	STO	Stop condition
STA	Start condition		

FIGURE 18-3: LOAD CONDITIONS



RL = 464Ω
CL = 50 pF for all pins except OSC2, but including PORTD and PORTE outputs as ports
15 pF for OSC2 output

FIGURE 18-4: EXTERNAL CLOCK TIMING

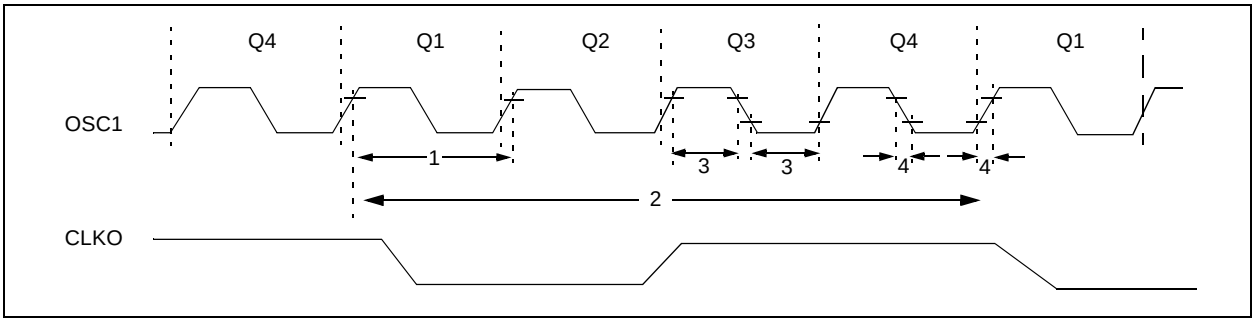


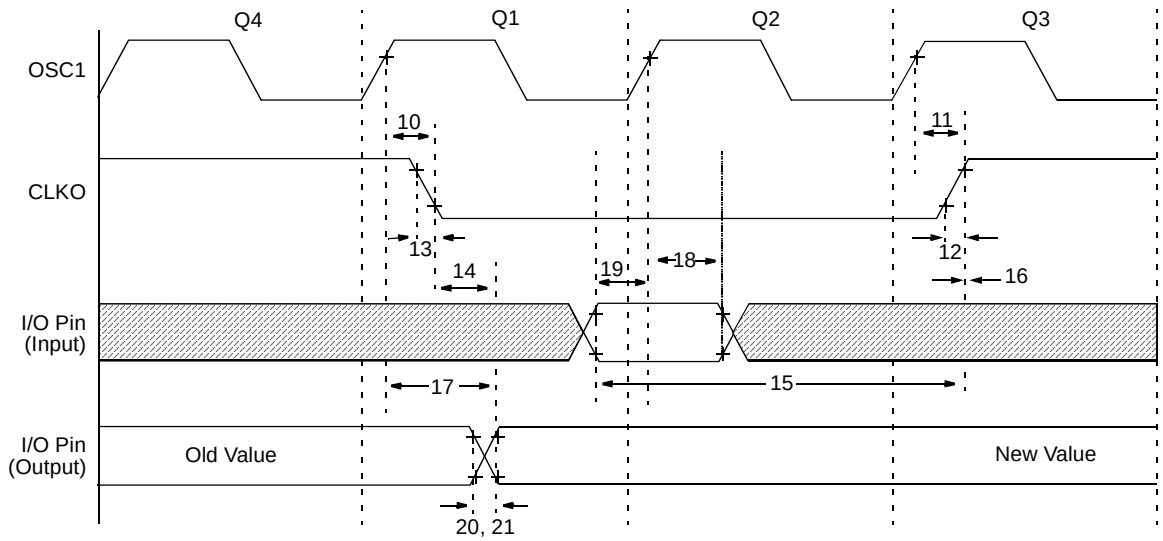
TABLE 18-3: EXTERNAL CLOCK TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	FOSC	External CLKI Frequency **(Note 1)**	DC	—	1	MHz	XT and RC Oscillator mode
			DC	—	20	MHz	HS Oscillator mode
			DC	—	32	kHz	LP Oscillator mode
		Oscillator Frequency **(Note 1)**	DC	—	4	MHz	RC Oscillator mode
			0.1	—	4	MHz	XT Oscillator mode
			4	—	20	MHz	HS Oscillator mode
			5	—	200	kHz	LP Oscillator mode
1	TOSC	External CLKI Period **(Note 1)**	1000	—	—	ns	XT and RC Oscillator modes
			50	—	—	ns	HS Oscillator mode
			5	—	—	ms	LP Oscillator mode
		Oscillator Period **(Note 1)**	250	—	—	ns	RC Oscillator mode
			250	—	10,000	ns	XT Oscillator mode
			50	—	250	ns	HS Oscillator mode
			5	—	—	ms	LP Oscillator mode
2	TCY	Instruction Cycle Time **(Note 1)**	200	TCY	DC	ns	TCY = 4/FOSC
3	TosL, TosH	External Clock in (OSC1) High or Low Time	500	—	—	ns	XT oscillator
			2.5	—	—	ms	LP oscillator
			15	—	—	ns	HS oscillator
4	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	—	—	25	ns	XT oscillator
			—	—	50	ns	LP oscillator
			—	—	15	ns	HS oscillator

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min.” values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the “max.” cycle time limit is “DC” (no clock) for all devices.

FIGURE 18-5: CLKO AND I/O TIMING



Note: Refer to Figure 18-3 for load conditions.

TABLE 18-4: CLKO AND I/O TIMING REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Typ†	Max	Units	Conditions
10*	TosH2ckL	OSC1 ↑ to CLKO ↓		—	75	200	ns	**(Note 1)**
11*	TosH2ckH	OSC1 ↑ to CLKO ↑		—	75	200	ns	**(Note 1)**
12*	TckR	CLKO Rise Time		—	35	100	ns	**(Note 1)**
13*	TckF	CLKO Fall Time		—	35	100	ns	**(Note 1)**
14*	TckL2ioV	CLKO ↓ to Port Out Valid		—	—	0.5 TCY + 20	ns	**(Note 1)**
15*	TioV2ckH	Port In Valid before CLKO ↑		TOSC + 200	—	—	ns	**(Note 1)**
16*	TckH2ioI	Port In Hold after CLKO ↑		0	—	—	ns	**(Note 1)**
17*	TosH2ioV	OSC1 ↑ (Q1 cycle) to Port Out Valid		—	100	255	ns	
18*	TosH2ioI	OSC1 ↑ (Q2 cycle) to Port Input Invalid (I/O in hold time)	PIC16**F**87/88	100	—	—	ns	
			PIC16**LF**87/88	200	—	—	ns	
19*	TioV2osH	Port Input Valid to OSC1 ↑ (I/O in setup time)		0	—	—	ns	
20*	TIOR	Port Output Rise Time	PIC16**F**87/88	—	10	40	ns	
			PIC16**LF**87/88	—	—	145	ns	
21*	TIOF	Port Output Fall Time	PIC16**F**87/88	—	10	40	ns	
			PIC16**LF**87/88	—	—	145	ns	
22††*	TINP	INT Pin High or Low Time		TCY	—	—	ns	
23††*	TRBP	RB7:RB4 Change INT High or Low Time		TCY	—	—	ns	

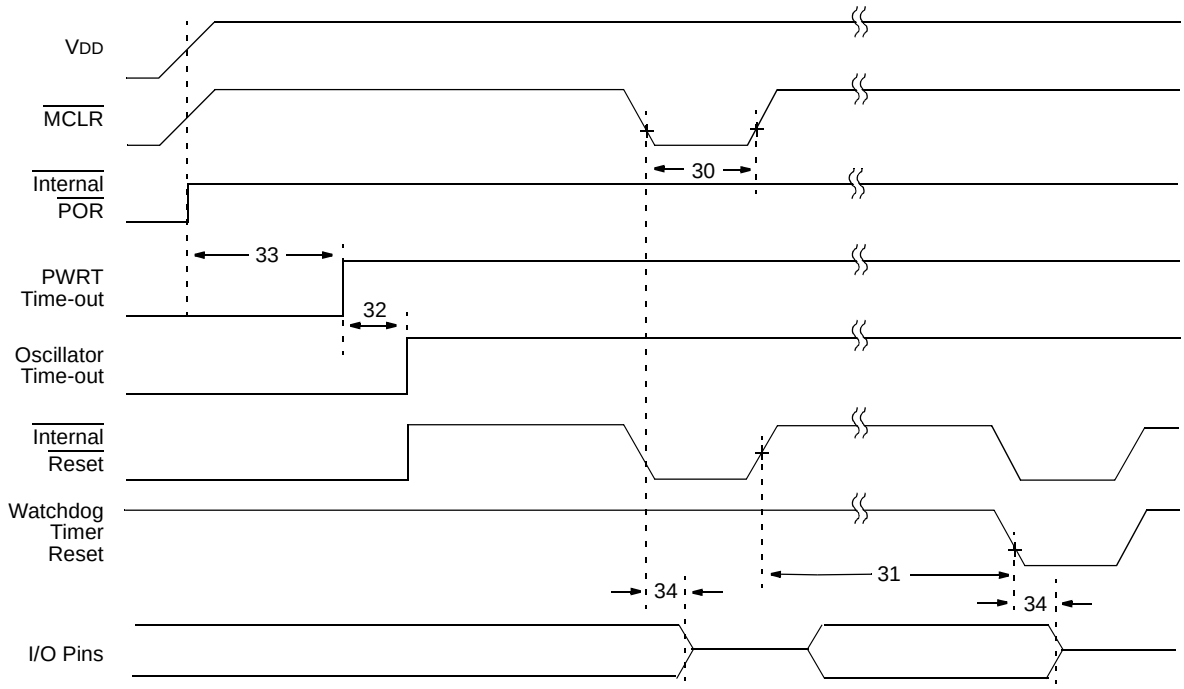
* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

†† These parameters are asynchronous events not related to any internal clock edges.

Note 1: Measurements are taken in RC mode where CLKO output is 4 x TOSC.

FIGURE 18-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING



Note: Refer to Figure 18-3 for load conditions.

FIGURE 18-7: BROWN-OUT RESET TIMING

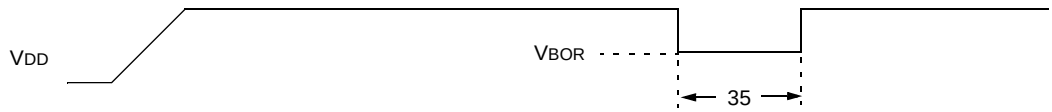


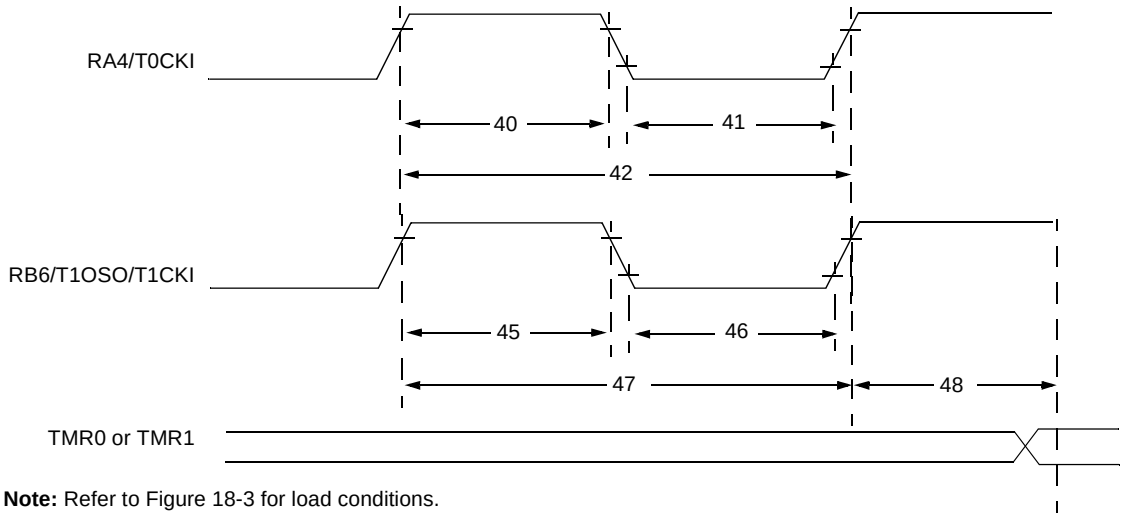
TABLE 18-5: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
30	TmcL	$\overline{\text{MCLR}}$ Pulse Width (Low)	2	—	—	µs	VDD = 5V, -40°C to +85°C
31*	TWDT	Watchdog Timer Time-out Period (16-bit prescaler = 0100 and no postscaler)	13.6	16	18.4	ms	VDD = 5V, -40°C to +85°C
32	TOST	Oscillation Start-up Timer Period	—	1024 TOSC	—	—	TOSC = OSC1 period
33*	TPWRT	Power-up Timer Period	61.2	72	82.8	ms	VDD = 5V, -40°C to +85°C
34	TIOZ	I/O High-impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset	—	—	2.1	µs	
35	TBOR	Brown-out Reset Pulse Width	100	—	—	µs	VDD ≤ VBOR (D005)

* These parameters are characterized but not tested.

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 18-8: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS



Note: Refer to Figure 18-3 for load conditions.

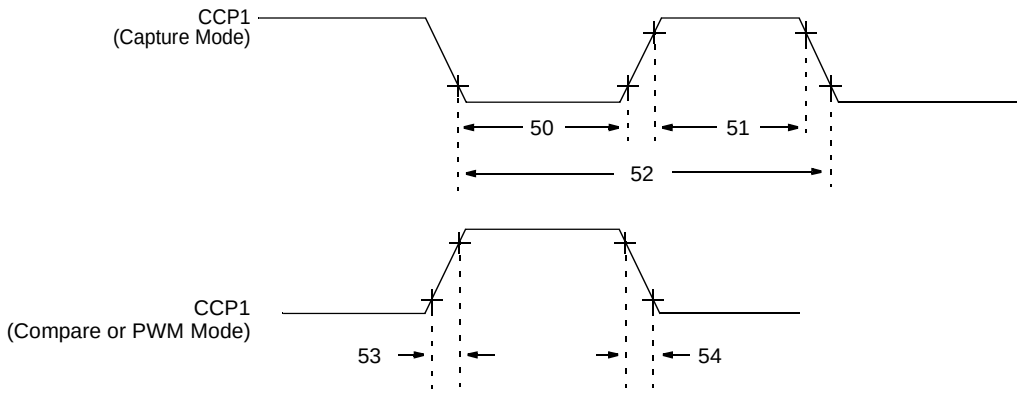
TABLE 18-6: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param No.	Symbol	Characteristic			Min	Typ†	Max	Units	Conditions
40*	Tt0H	T0CKI High Pulse Width		No Prescaler	0.5 TCY + 20	—	—	ns	Must also meet parameter 42
				With Prescaler	10	—	—	ns	
41*	Tt0L	T0CKI Low Pulse Width		No Prescaler	0.5 TCY + 20	—	—	ns	Must also meet parameter 42
				With Prescaler	10	—	—	ns	
42*	Tt0P	T0CKI Period		No Prescaler	TCY + 40	—	—	ns	
				With Prescaler	Greater of: 20 or TCY + 40 / N	—	—	ns	N = prescale value (2, 4, ..., 256)
45*	Tt1H	T1CKI High Time	Synchronous, Prescaler = 1		0.5 TCY + 20	—	—	ns	Must also meet parameter 47
			Synchronous, Prescaler = 2, 4, 8	PIC16**F**87/88	15	—	—	ns	
				PIC16**LF**87/88	25	—	—	ns	
			Asynchronous	PIC16**F**87/88	30	—	—	ns	
				PIC16**LF**87/88	50	—	—	ns	
46*	Tt1L	T1CKI Low Time	Synchronous, Prescaler = 1		0.5 TCY + 20	—	—	ns	Must also meet parameter 47
			Synchronous, Prescaler = 2, 4, 8	PIC16**F**87/88	15	—	—	ns	
				PIC16**LF**87/88	25	—	—	ns	
			Asynchronous	PIC16**F**87/88	30	—	—	ns	
				PIC16**LF**87/88	50	—	—	ns	
47*	Tt1P	T1CKI Input Period	Synchronous	PIC16**F**87/88	Greater of: 30 or TCY + 40 / N	—	—	ns	N = prescale value (1, 2, 4, 8)
				PIC16**LF**87/88	Greater of: 50 or TCY + 40 / N				N = prescale value (1, 2, 4, 8)
			Asynchronous	PIC16**F**87/88	60	—	—	ns	
				PIC16**LF**87/88	100	—	—	ns	
	Ft1	Timer1 Oscillator Input Frequency Range (Oscillator enabled by setting bit T1OSCEN)			DC	—	32.768	kHz	
48	TCKEZtmr1	Delay from External Clock Edge to Timer Increment			2 TOSC	—	7 TOSC	—	

* These parameters are characterized but not tested.

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 18-9: CAPTURE/COMPARE/PWM TIMINGS (CCP1)



Note: Refer to Figure 18-3 for load conditions.

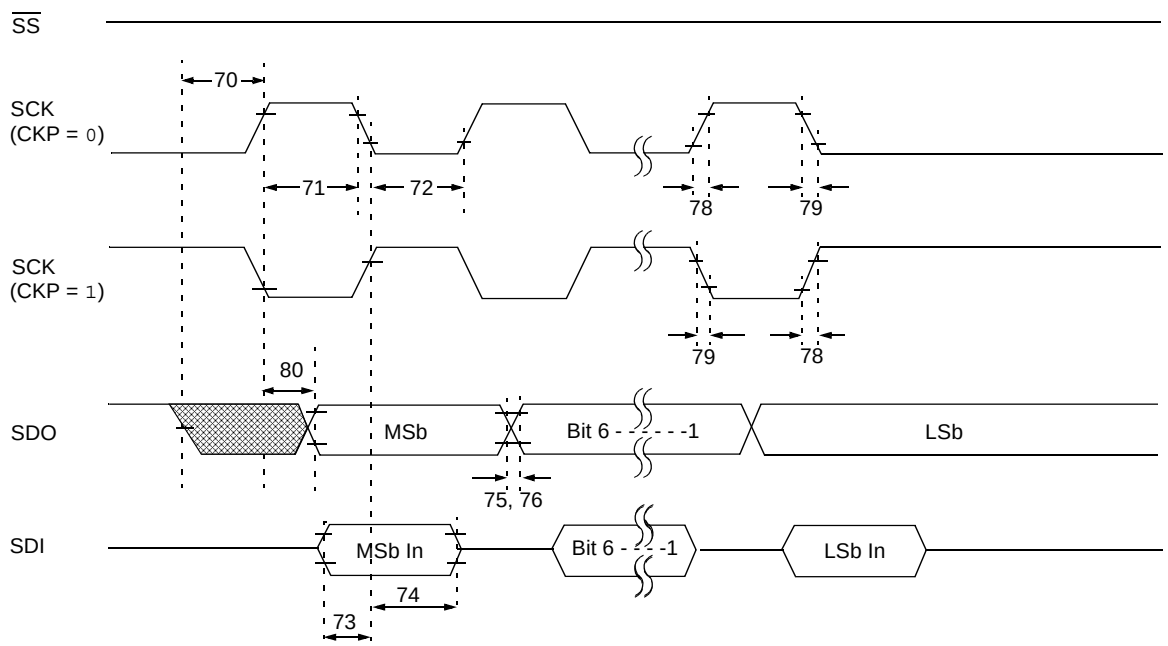
TABLE 18-7: CAPTURE/COMPARE/PWM REQUIREMENTS (CCP1)

Param No.	Symbol	Characteristic			Min	Typ†	Max	Units	Conditions
50*	TccL	CCP1 Input Low Time	No Prescaler		0.5 TCY + 20	—	—	ns	
			With Prescaler	PIC16**F**87/88	10	—	—	ns	
				PIC16**LF**87/88	20	—	—	ns	
51*	TccH	CCP1 Input High Time	No Prescaler		0.5 TCY + 20	—	—	ns	
			With Prescaler	PIC16**F**87/88	10	—	—	ns	
				PIC16**LF**87/88	20	—	—	ns	
52*	TccP	CCP1 Input Period			$\frac{3\ T_{CY} + 40}{N}$	—	—	ns	N = prescale value (1, 4 or 16)
53*	TccR	CCP1 Output Rise Time		PIC16**F**87/88	—	10	25	ns	
				PIC16**LF**87/88	—	25	50	ns	
54*	TccF	CCP1 Output Fall Time		PIC16**F**87/88	—	10	25	ns	
				PIC16**LF**87/88	—	25	45	ns	

* These parameters are characterized but not tested.

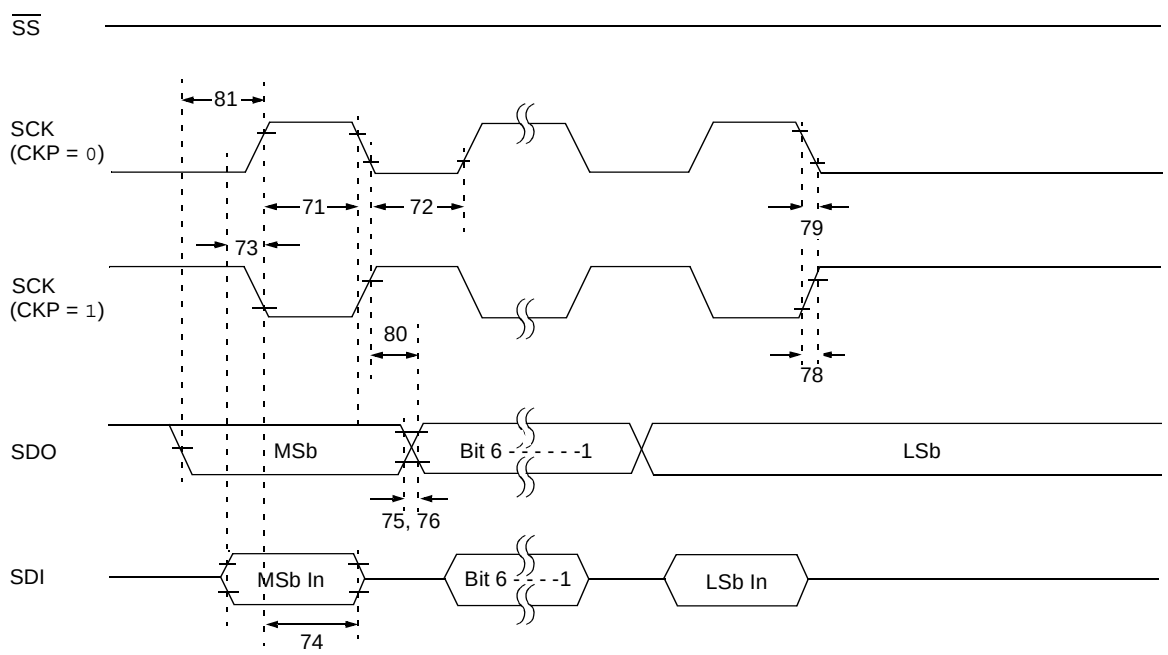
† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 18-10: SPI™ MASTER MODE TIMING (CKE = 0, SMP = 0)



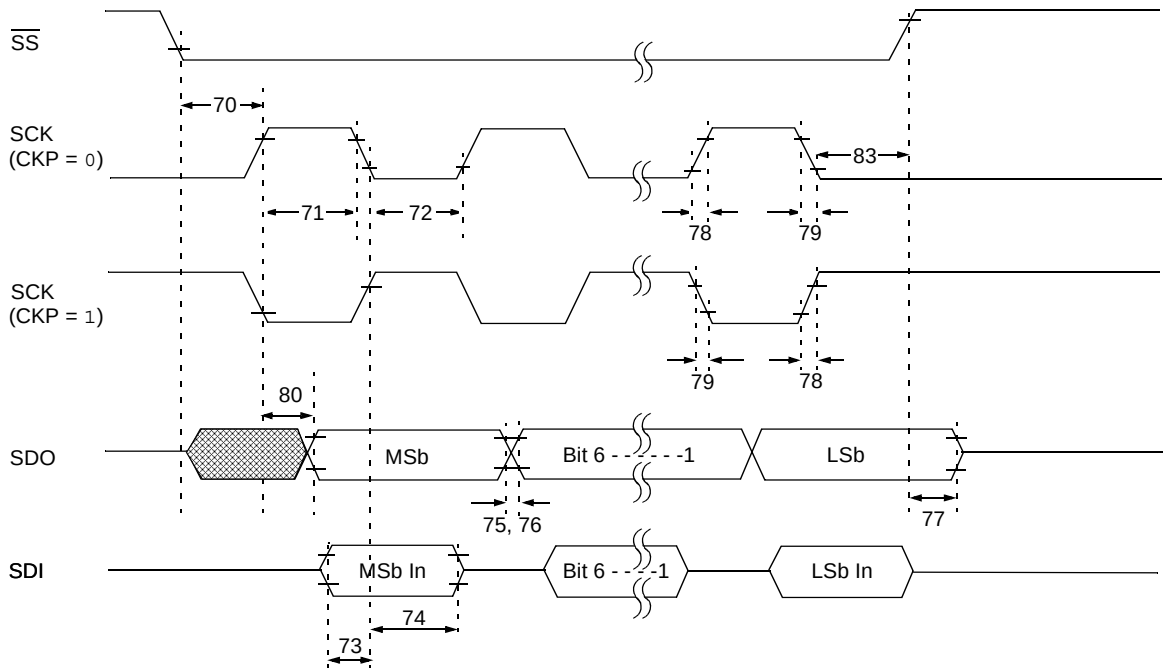
Note: Refer to Figure 18-3 for load conditions.

FIGURE 18-11: SPI™ MASTER MODE TIMING (CKE = 1, SMP = 1)



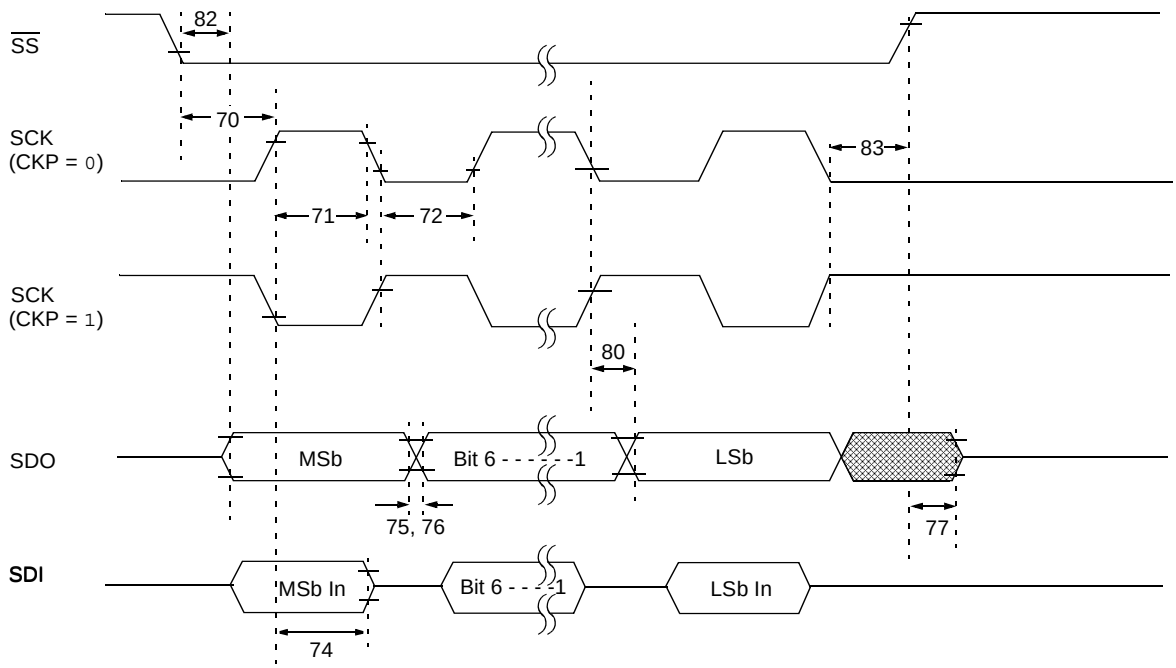
Note: Refer to Figure 18-3 for load conditions.

FIGURE 18-12: SPI™ SLAVE MODE TIMING (CKE = 0)



Note: Refer to Figure 18-3 for load conditions.

FIGURE 18-13: SPI™ SLAVE MODE TIMING (CKE = 1)



Note: Refer to Figure 18-3 for load conditions.

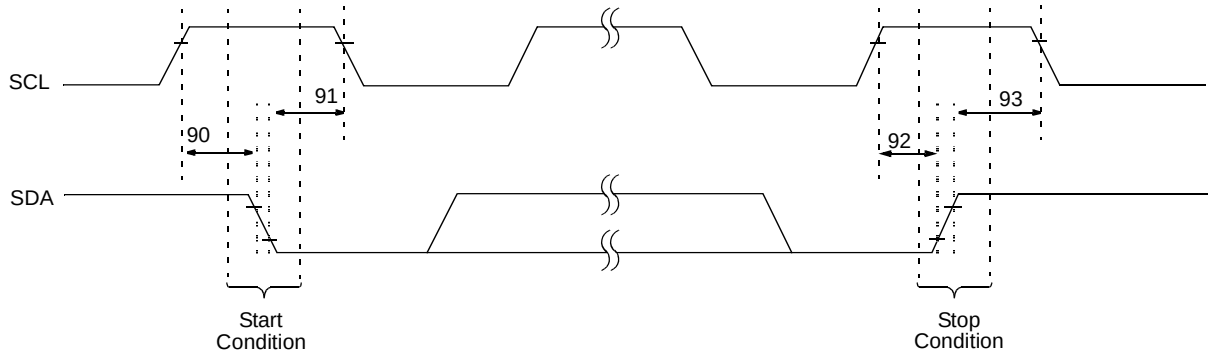
TABLE 18-8: SPI™ MODE REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Typ†	Max	Units	Conditions
70*	TssL2scH, TssL2scL	$\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input		TCY	—	—	ns	
71*	TscH	SCK Input High Time (Slave mode)		TCY + 20	—	—	ns	
72*	TscL	SCK Input Low Time (Slave mode)		TCY + 20	—	—	ns	
73*	TdiV2scH, TdiV2scL	Setup Time of SDI Data Input to SCK Edge		100	—	—	ns	
74*	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		100	—	—	ns	
75*	TdoR	SDO Data Output Rise Time	PIC16**F**87/88 PIC16**LF**87/88	— —	10 25	25 50	ns ns	
76*	TdoF	SDO Data Output Fall Time		—	10	25	ns	
77*	TssH2doZ	$\overline{SS}$ ↑ to SDO Output High-Impedance		10	—	50	ns	
78*	TscR	SCK Output Rise Time (Master mode)	PIC16**F**87/88 PIC16**LF**87/88	— —	10 25	25 50	ns ns	
79*	TscF	SCK Output Fall Time (Master mode)		—	10	25	ns	
80*	TscH2doV, TscL2doV	SDO Data Output Valid after SCK Edge	PIC16**F**87/88 PIC16**LF**87/88	— —	— —	50 145	ns ns	
81*	TdoV2scH, TdoV2scL	SDO Data Output Setup to SCK Edge		TCY	—	—	ns	
82*	TssL2doV	SDO Data Output Valid after $\overline{SS}$ ↓ Edge		—	—	50	ns	
83*	TscH2ssH, TscL2ssH	$\overline{SS}$ ↑ after SCK Edge		1.5 TCY + 40	—	—	ns	

* These parameters are characterized but not tested.

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 18-14: I²C™ BUS START/STOP BITS TIMING



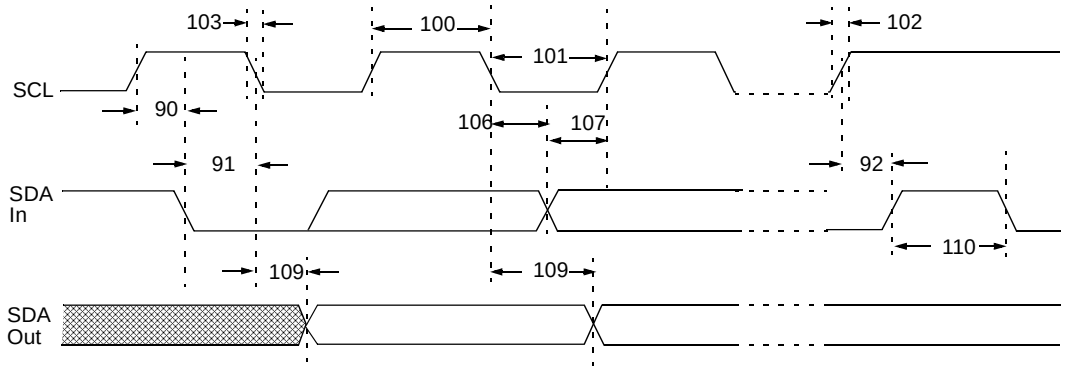
Note: Refer to Figure 18-3 for load conditions.

TABLE 18-9: I²C™ BUS START/STOP BITS REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
90*	TSU:STA	Start Condition Setup Time	100 kHz mode	4700	—	—	ns	Only relevant for Repeated Start condition
			400 kHz mode	600	—	—		
91*	THD:STA	Start Condition Hold Time	100 kHz mode	4000	—	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	600	—	—		
92*	TSU:STO	Stop Condition Setup Time	100 kHz mode	4700	—	—	ns	
			400 kHz mode	600	—	—		
93	THD:STO	Stop Condition Hold Time	100 kHz mode	4000	—	—	ns	
			400 kHz mode	600	—	—		

* These parameters are characterized but not tested.

FIGURE 18-15: I²C™ BUS DATA TIMING



Note: Refer to Figure 18-3 for load conditions.

TABLE 18-10: I²C™ BUS DATA REQUIREMENTS

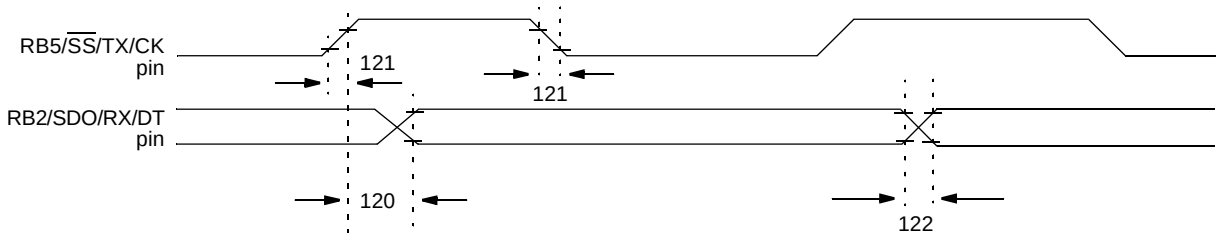
Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100*	THIGH	Clock High Time	100 kHz mode	4.0	—	μs	
			400 kHz mode	0.6	—	μs	
			SSP Module	1.5 TCY	—		
101*	TLOW	Clock Low Time	100 kHz mode	4.7	—	μs	
			400 kHz mode	1.3	—	μs	
			SSP Module	1.5 TCY	—		
102*	TR	SDA and SCL Rise Time	100 kHz mode	—	1000	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10-400 pF
103*	TF	SDA and SCL Fall Time	100 kHz mode	—	300	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10-400 pF
90*	TSU:STA	Start Condition Setup Time	100 kHz mode	4.7	—	μs	Only relevant for Repeated Start condition
			400 kHz mode	0.6	—	μs	
91*	THD:STA	Start Condition Hold Time	100 kHz mode	4.0	—	μs	After this period, the first clock pulse is generated
			400 kHz mode	0.6	—	μs	
106*	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	μs	
107*	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
92*	TSU:STO	Stop Condition Setup Time	100 kHz mode	4.7	—	μs	
			400 kHz mode	0.6	—	μs	
109*	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	**(Note 1)**
			400 kHz mode	—	—	ns	
110*	TBUF	Bus Free Time	100 kHz mode	4.7	—	μs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	μs	
	CB	Bus Capacitive Loading		—	400	pF	

* These parameters are characterized but not tested.

Note 1: As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of Start or Stop conditions.

2: A Fast mode (400 kHz) I²C™ bus device can be used in a Standard mode (100 kHz) I²C bus system, but the requirement, TSU:DAT ≥ 250 ns, must then be met. This will automatically be the case if the device does not stretch the low period of the SCL signal. If such a device does stretch the low period of the SCL signal, it must output the next data bit to the SDA line, TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I²C bus specification), before the SCL line is released.

FIGURE 18-16: AUSART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING



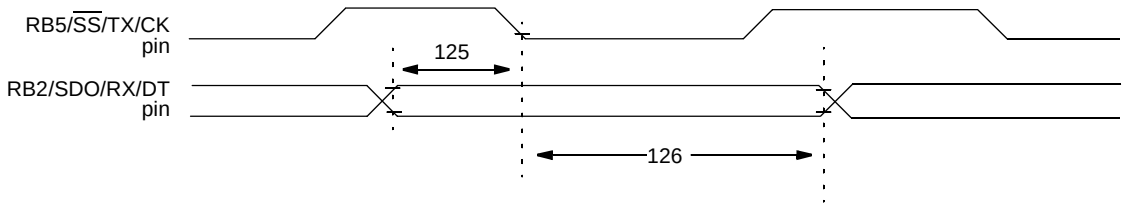
Note: Refer to Figure 18-3 for load conditions.

TABLE 18-11: AUSART SYNCHRONOUS TRANSMISSION REQUIREMENTS

Param No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
120	TckH2dtV	SYNC XMIT (MASTER & SLAVE) Clock High to Data Out Valid	PIC16**F**87/88	—	—	80	ns	
			PIC16**LF**87/88	—	—	100	ns	
121	Tckrf	Clock Out Rise Time and Fall Time (Master mode)	PIC16**F**87/88	—	—	45	ns	
			PIC16**LF**87/88	—	—	50	ns	
122	Tdtrf	Data Out Rise Time and Fall Time	PIC16**F**87/88	—	—	45	ns	
			PIC16**LF**87/88	—	—	50	ns	

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 18-17: AUSART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING



Note: Refer to Figure 18-3 for load conditions.

TABLE 18-12: AUSART SYNCHRONOUS RECEIVE REQUIREMENTS

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
125	TdtV2ckL	SYNC RCV (MASTER & SLAVE) Data Setup before CK ↓ (DT setup time)	15	—	—	ns	
126	TckL2dtl	Data Hold after CK ↓ (DT hold time)	15	—	—	ns	

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

TABLE 18-13: A/D CONVERTER CHARACTERISTICS: PIC16F87/88 (INDUSTRIAL, EXTENDED) PIC16LF87/88 (INDUSTRIAL)

Param No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
A01	NR	Resolution		—	—	10-bit	bit	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A03	EIL	Integral Linearity Error		—	—	<±1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A04	EDL	Differential Linearity Error		—	—	<±1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A06	EOFF	Offset Error		—	—	<±2	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A07	EGN	Gain Error		—	—	<±1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A10	—	Monotonicity		—	guaranteed(3)	—	—	VSS ≤ VAIN ≤ VREF
A20	VREF	Reference Voltage (VREF+ – VREF-)		2.0	—	VDD + 0.3	V	
A21	VREF+	Reference Voltage High		AVDD – 2.5V		AVDD + 0.3V	V	
A22	VREF-	Reference Voltage Low		AVSS – 0.3V		VREF+ – 2.0V	V	
A25	VAIN	Analog Input Voltage		VSS – 0.3V	—	VREF + 0.3V	V	
A30	ZAIN	Recommended Impedance of Analog Voltage Source		—	—	2.5	kΩ	**(Note 4)**
A40	IAD	A/D Conversion Current (VDD)	PIC16F87/88	—	220	—	μA	Average current consumption when A/D is on **(Note 1)**
			PIC16LF87/88	—	90	—	μA	
A50	IREF	VREF Input Current **(Note 2)**		—	—	5	μA	During VAIN acquisition. Based on differential of VHOLD to VAIN to charge CHOLD, see **Section 12.1 "A/D Acquisition Requirements"**.
				—	—	150	μA	During A/D conversion cycle

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

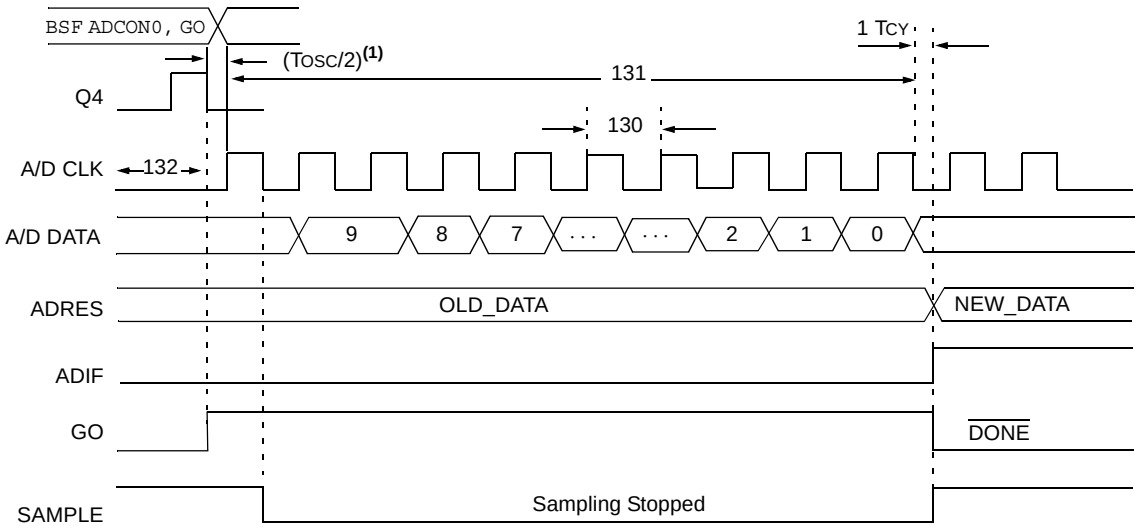
Note 1: When A/D is off, it will not consume any current other than minor leakage current. The power-down current specification includes any such leakage from the A/D module.

2: VREF current is from RA3 pin or VDD pin, whichever is selected as reference input.

3: The A/D conversion result never decreases with an increase in the input voltage and has no missing codes.

4: Maximum allowed impedance for analog voltage source is 10 kΩ. This requires higher acquisition time.

FIGURE 18-18: A/D CONVERSION TIMING



Note: If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

TABLE 18-14: A/D CONVERSION REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Typ†	Max	Units	Conditions
130	TAD	A/D Clock Period	PIC16F87/88	1.6	—	—	µs	TOSC based, VREF ≥ 3.0V
			PIC16LF87/88	3.0	—	—	µs	TOSC based, VREF ≥ 2.0V
			PIC16F87/88	2.0	4.0	6.0	µs	A/D RC mode
			PIC16LF87/88	3.0	6.0	9.0	µs	A/D RC mode
131	TCNV	Conversion Time (not including S/H time) **(Note 1)**			—	12	TAD	
132	TACQ	Acquisition Time		**(Note 2)**	40	—	µs	
				10*	—	—	µs	The minimum time is the amplifier settling time. This may be used if the "new" input voltage has not changed by more than 1 LSb (i.e., 5.0 mV @ 5.12V) from the last sampled voltage (as stated on CHOLD).
134	TGO	Q4 to A/D Clock Start		—	TOSC/2 §	—	—	If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

§ This specification ensured by design.

Note 1: ADRES registers may be read on the following TCY cycle.

2: See **Section 12.1 "A/D Acquisition Requirements"** for minimum conditions.

NOTES:

19.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

Note:	The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore, outside the warranted range.

"Typical" represents the mean of the distribution at 25°C. "Maximum" or "minimum" represents (mean + 3σ) or (mean – 3σ) respectively, where σ is a standard deviation, over the whole temperature range.

FIGURE 19-1: TYPICAL IDD vs. FOSC OVER VDD (HS MODE)

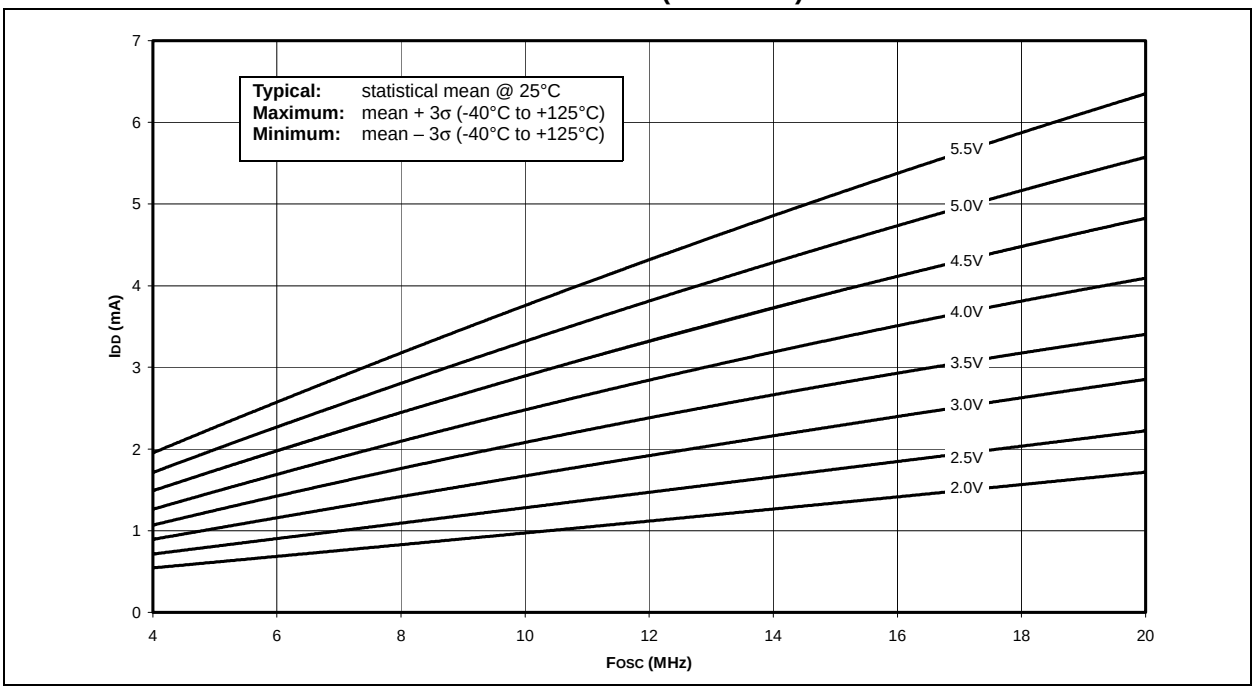


FIGURE 19-2: MAXIMUM IDD vs. FOSC OVER VDD (HS MODE)

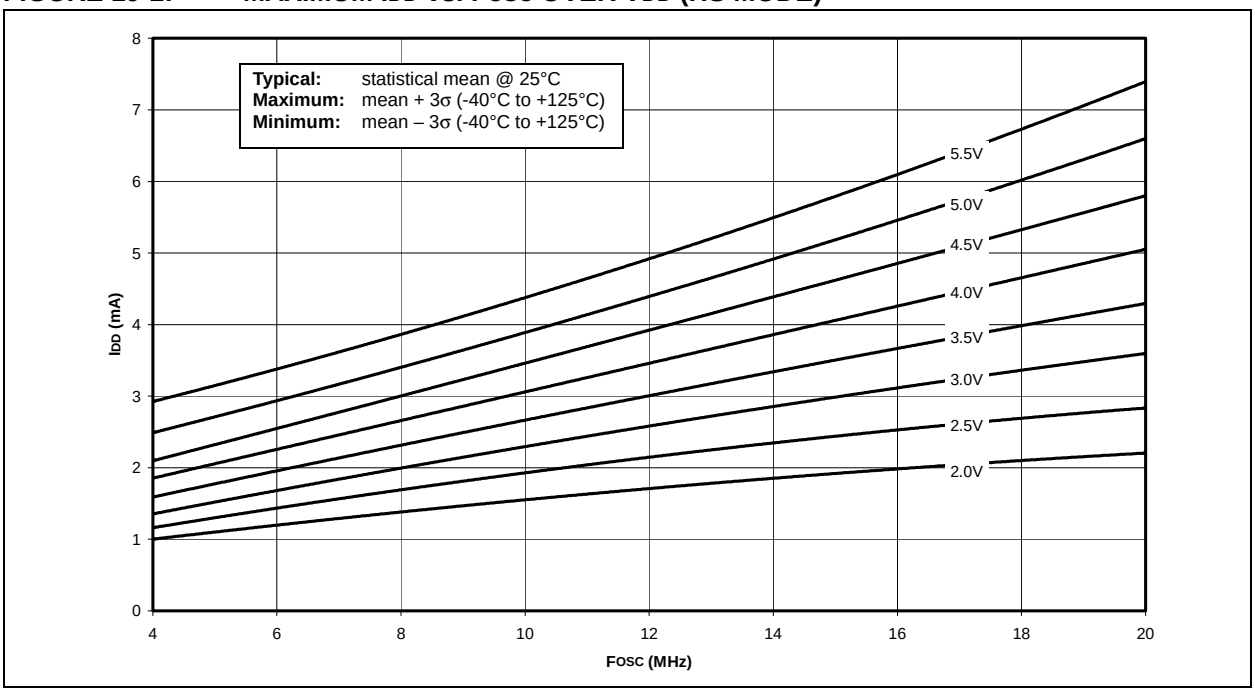


FIGURE 19-3: TYPICAL IDD vs. FOSC OVER VDD (XT MODE)

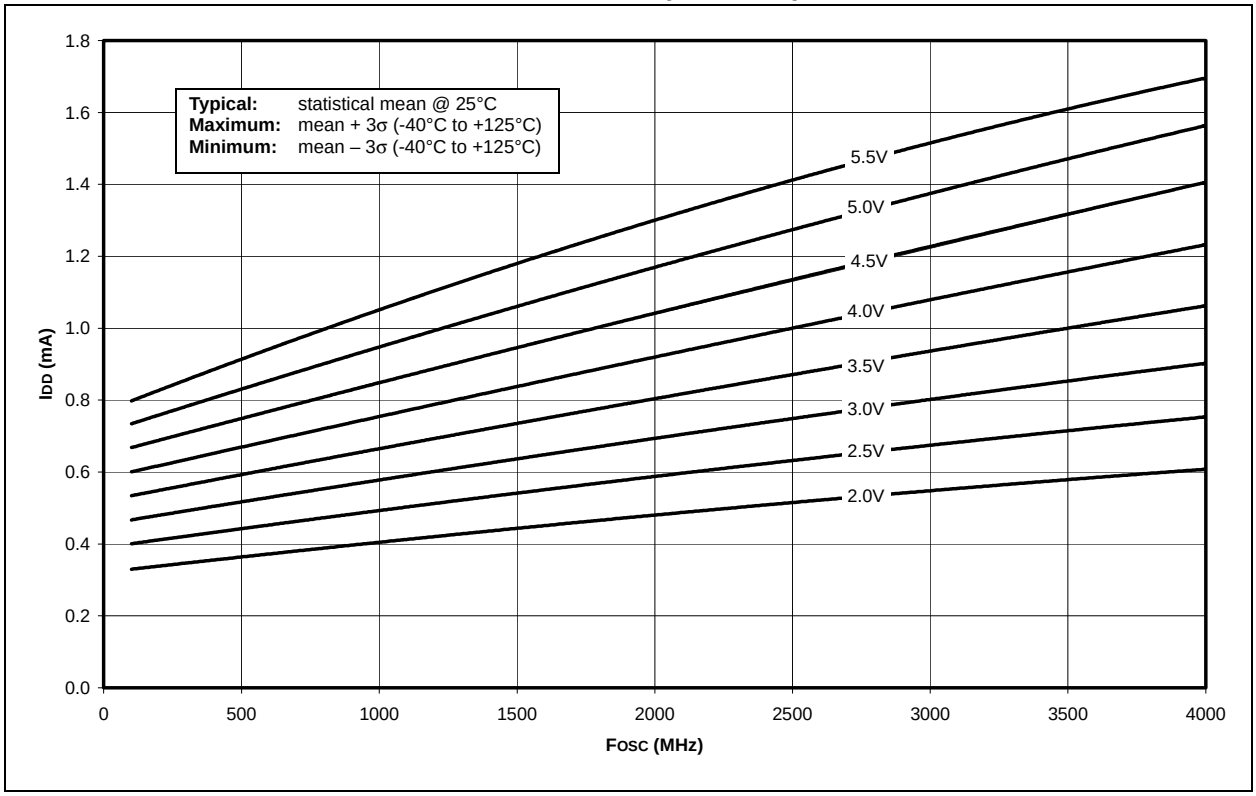


FIGURE 19-4: MAXIMUM IDD vs. FOSC OVER VDD (XT MODE)

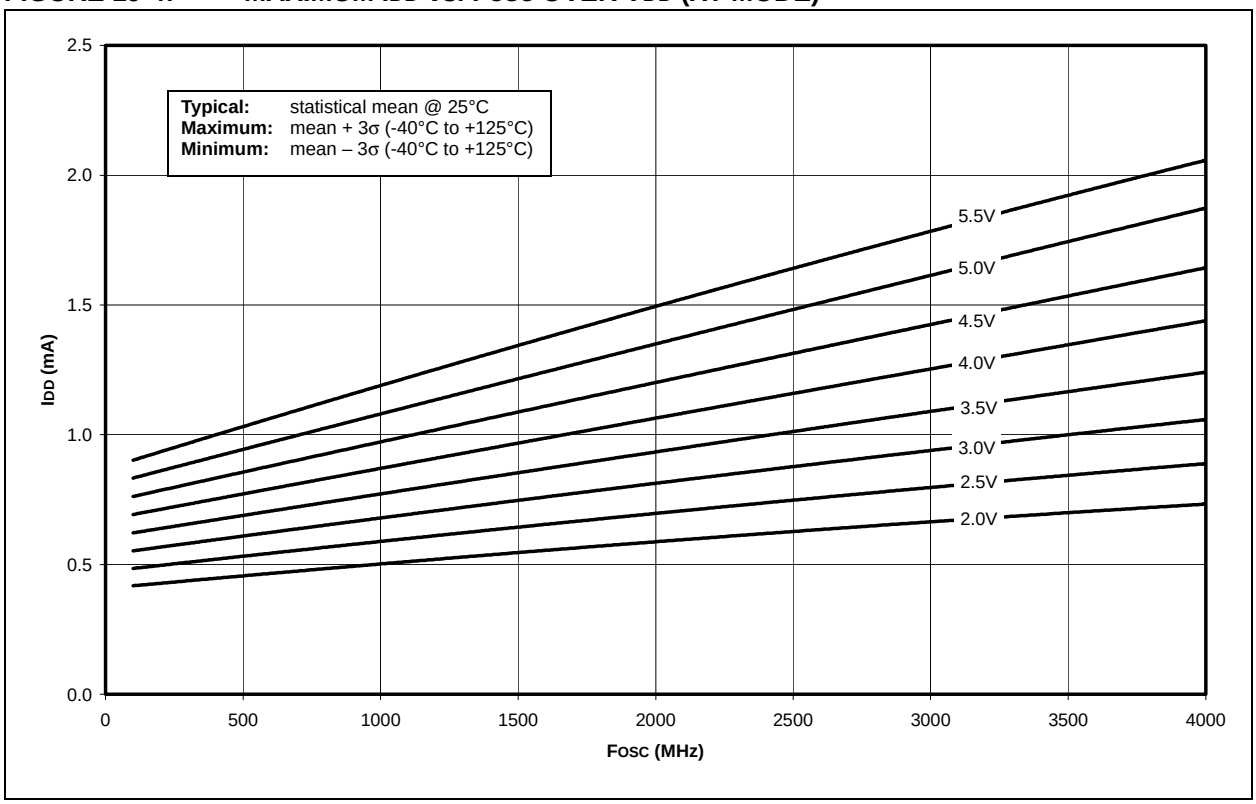


FIGURE 19-5: TYPICAL IDD vs. FOSC OVER VDD (LP MODE)

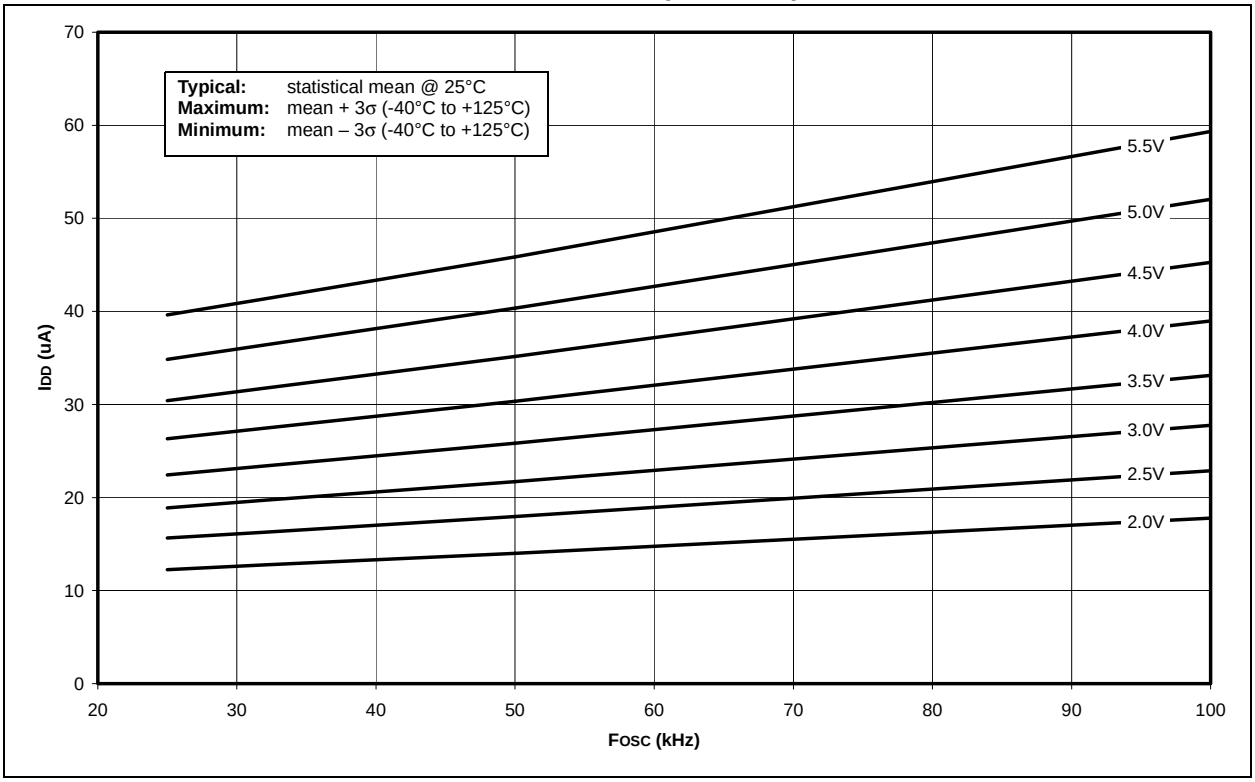


FIGURE 19-6: MAXIMUM IDD vs. FOSC OVER VDD (LP MODE)

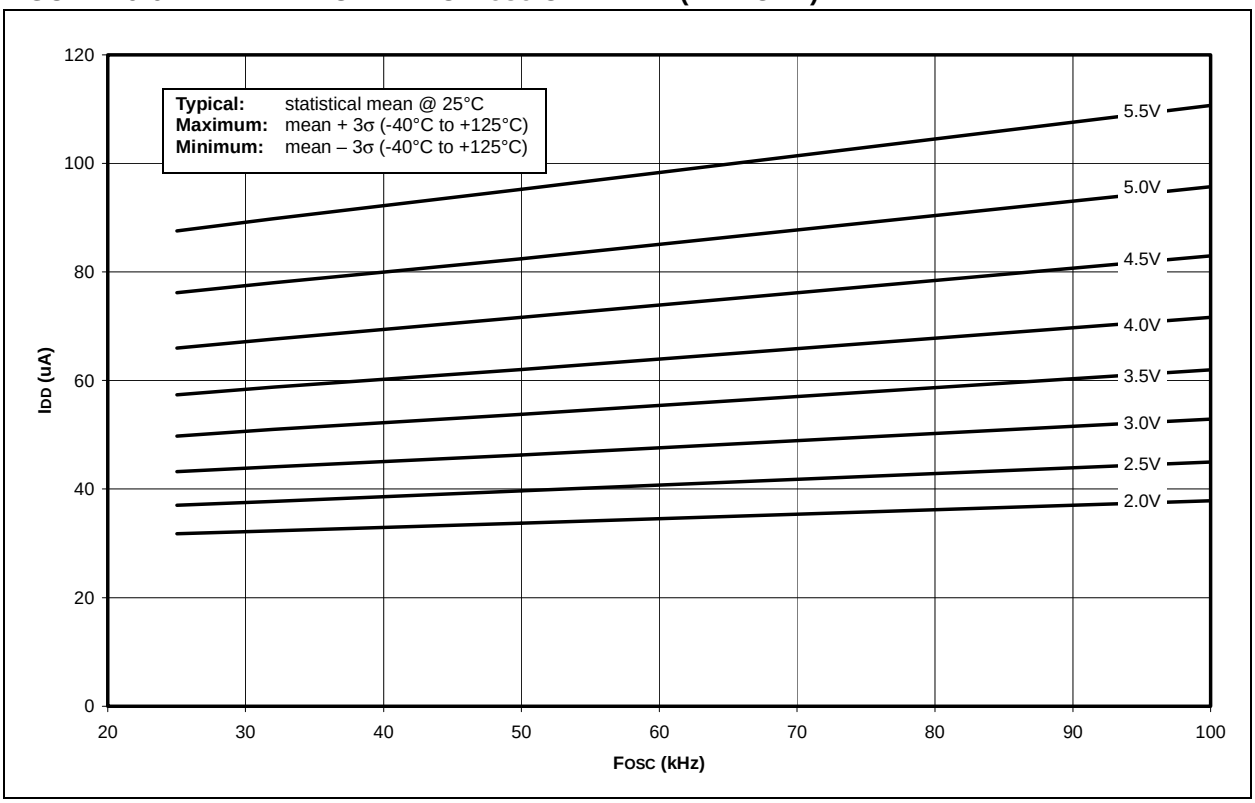


FIGURE 19-7: TYPICAL IDD vs. VDD, -40°C TO +125°C, 1 MHz TO 8 MHz (RC_RUN MODE, ALL PERIPHERALS DISABLED)

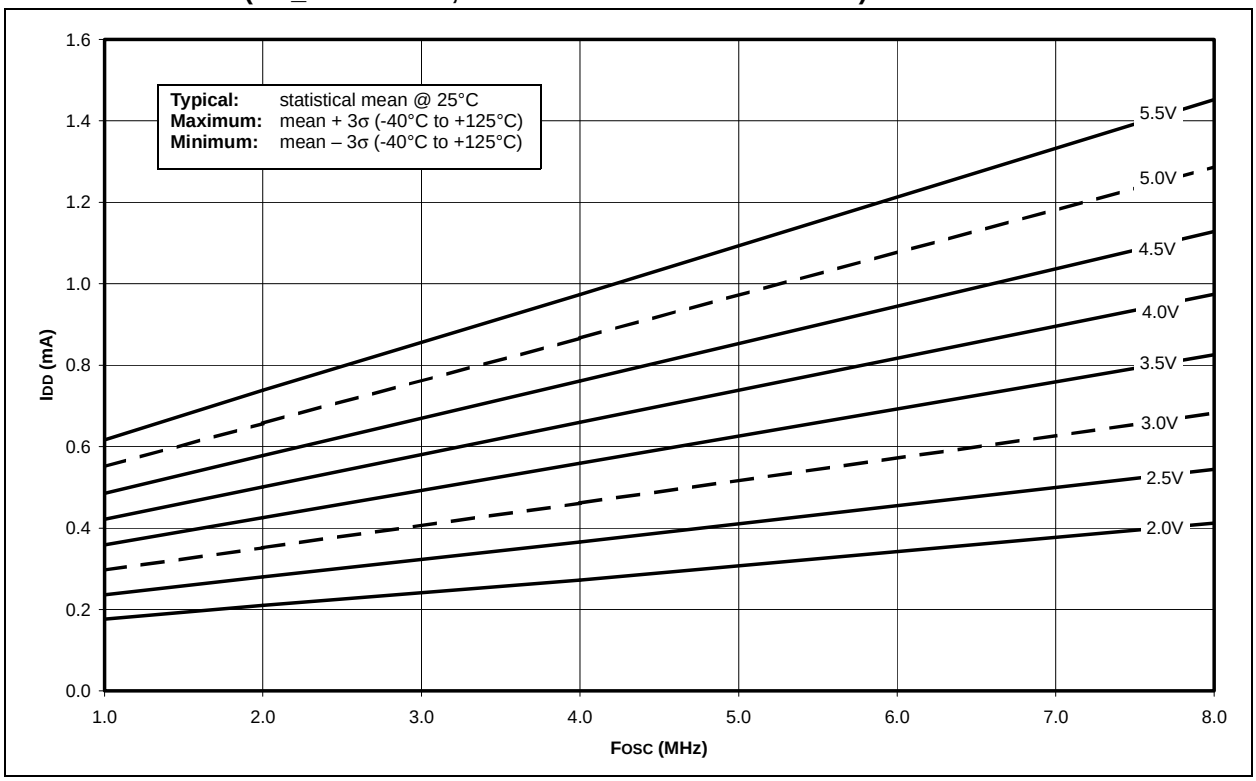


FIGURE 19-8: MAXIMUM IDD vs. VDD, -40°C TO +125°C, 1 MHz TO 8 MHz (RC_RUN MODE, ALL PERIPHERALS DISABLED)

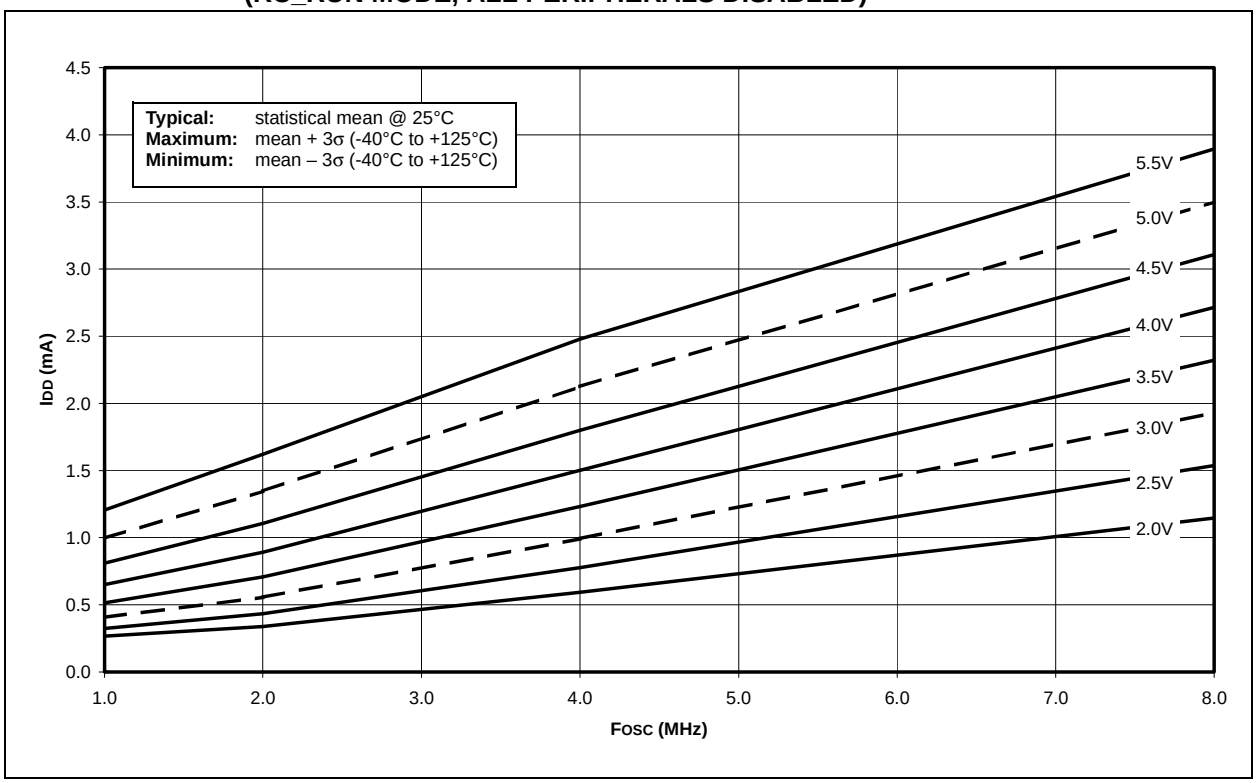


FIGURE 19-9: IDD vs. VDD, SEC_RUN MODE, -10°C TO +125°C, 32.768 kHz (XTAL 2 x 22 pF, ALL PERIPHERALS DISABLED)

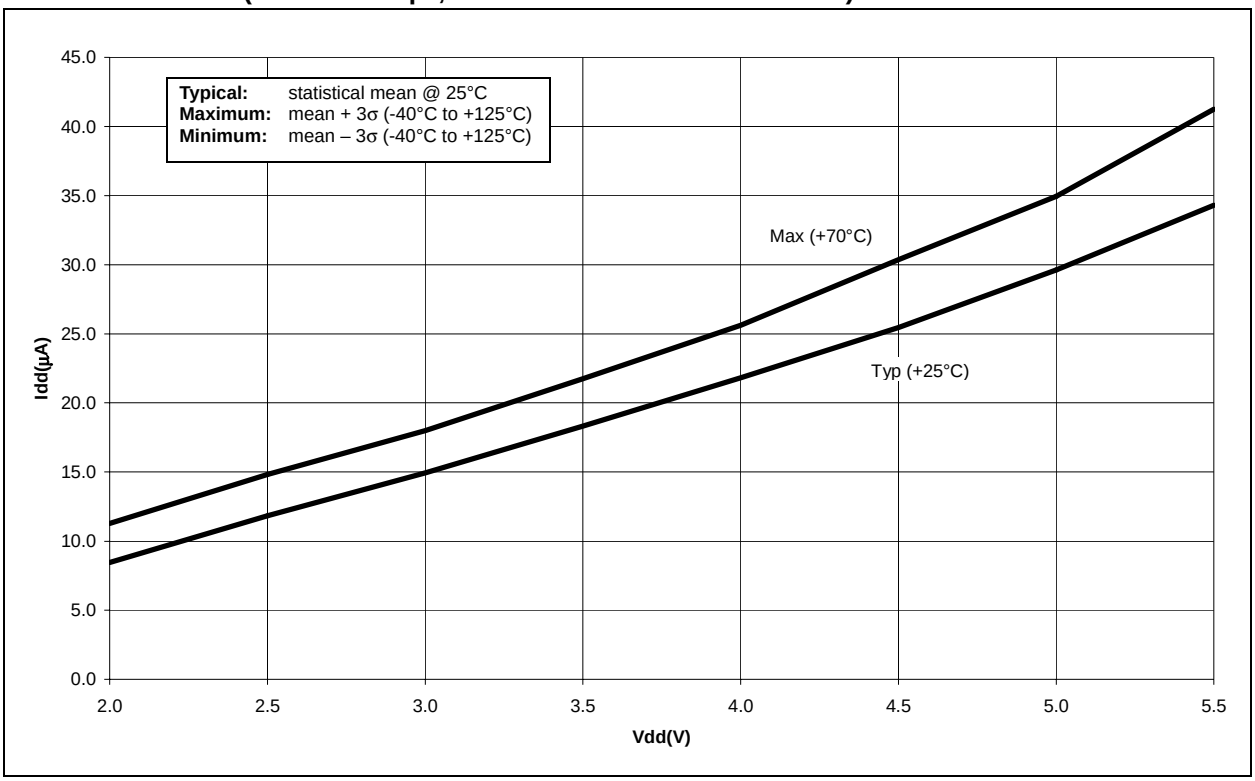


FIGURE 19-10: IPD vs. VDD, -40°C TO +125°C (SLEEP MODE, ALL PERIPHERALS DISABLED)

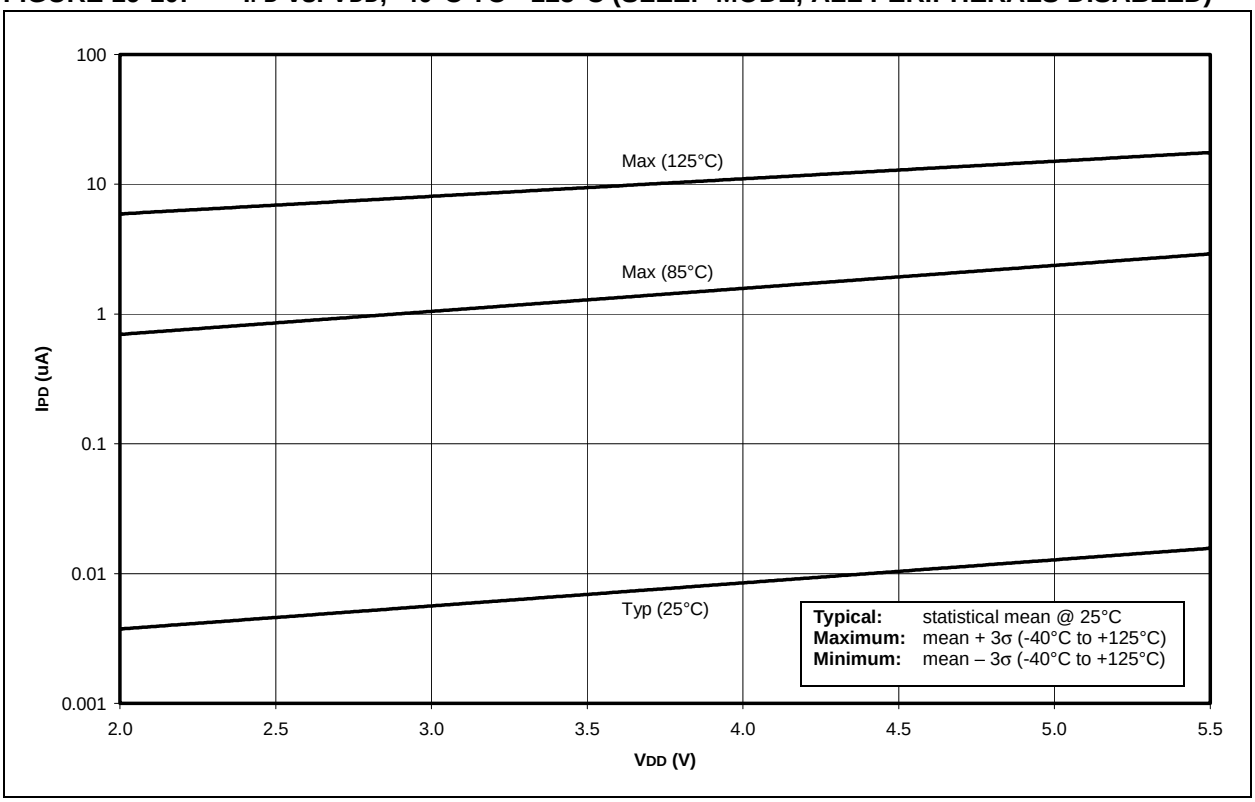


FIGURE 19-11: AVERAGE FOSC vs. VDD FOR VARIOUS VALUES OF R (RC MODE, C = 20 pF, +25°C)

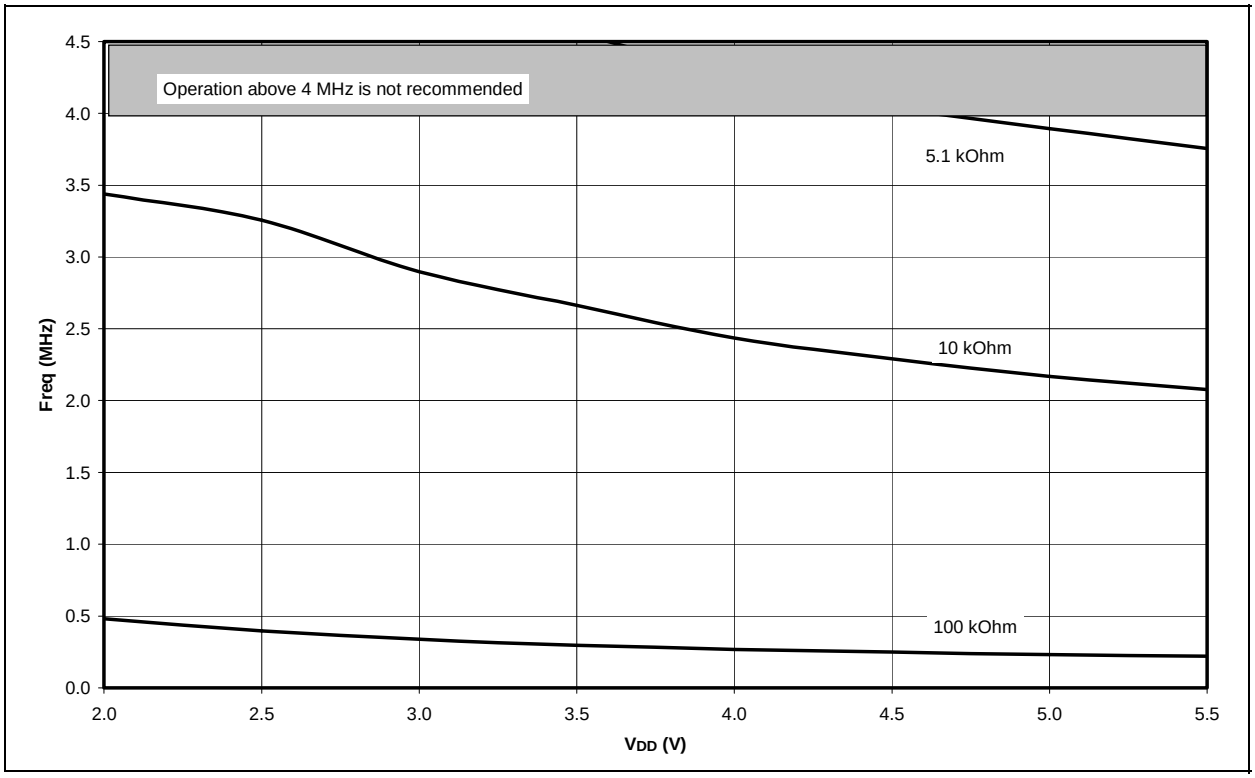


FIGURE 19-12: AVERAGE FOSC vs. VDD FOR VARIOUS VALUES OF R (RC MODE, C = 100 pF, +25°C)

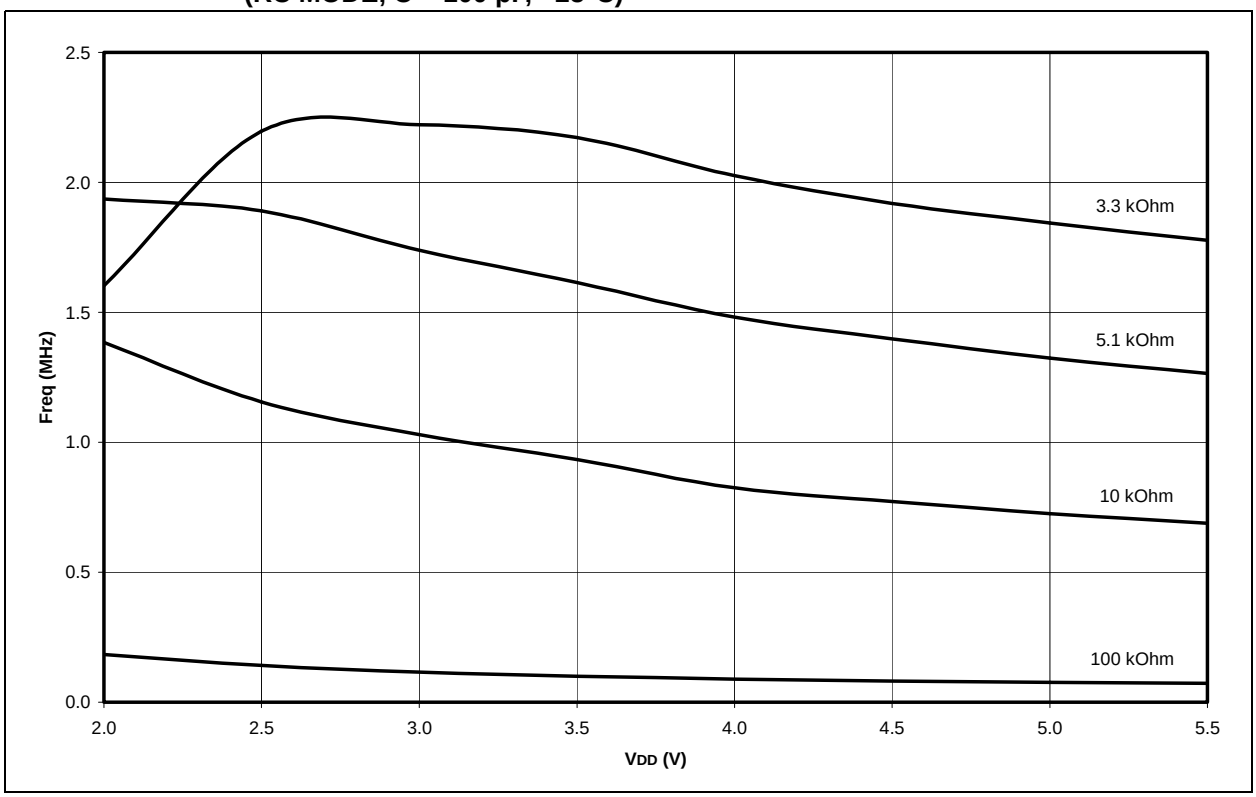


FIGURE 19-13: AVERAGE FOSC vs. VDD FOR VARIOUS VALUES OF R (RC MODE, C = 300 pF, +25°C)

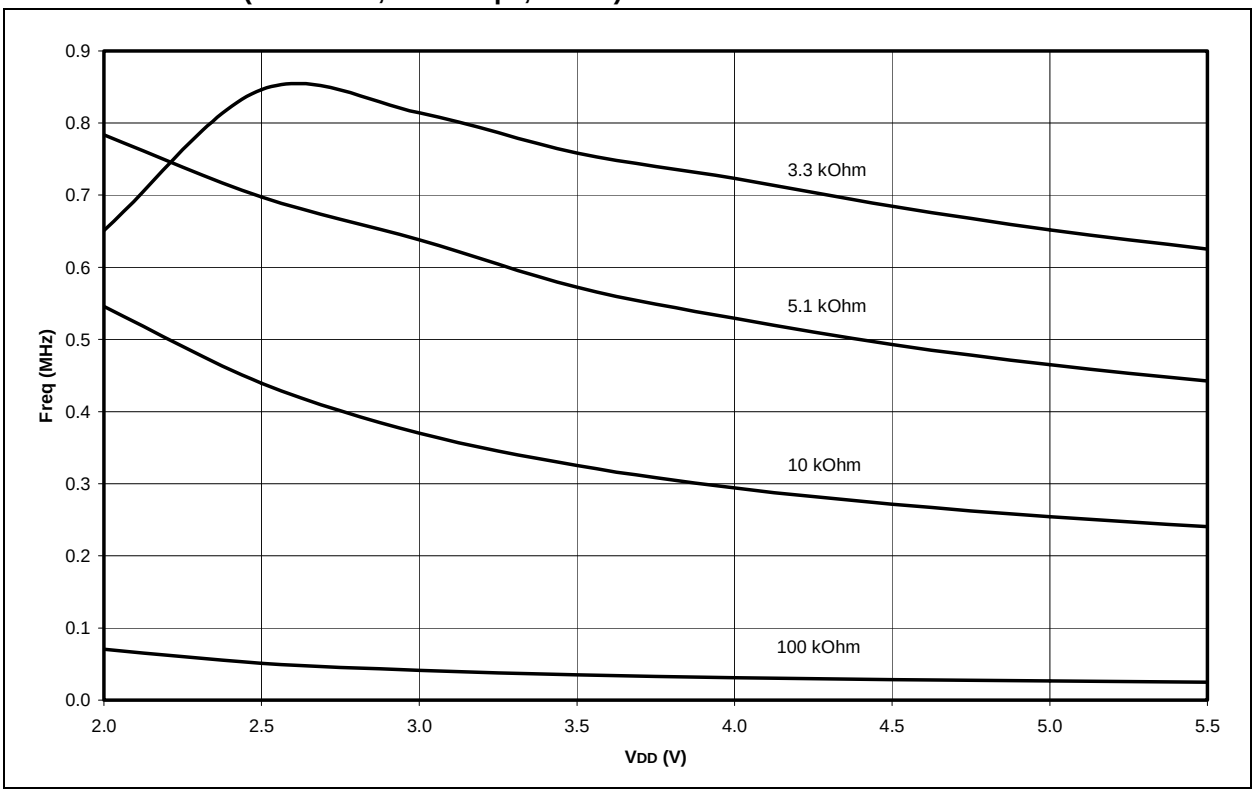


FIGURE 19-14: ΔIPD TIMER1 OSCILLATOR, -10°C TO +70°C (SLEEP MODE, TMR1 COUNTER DISABLED)

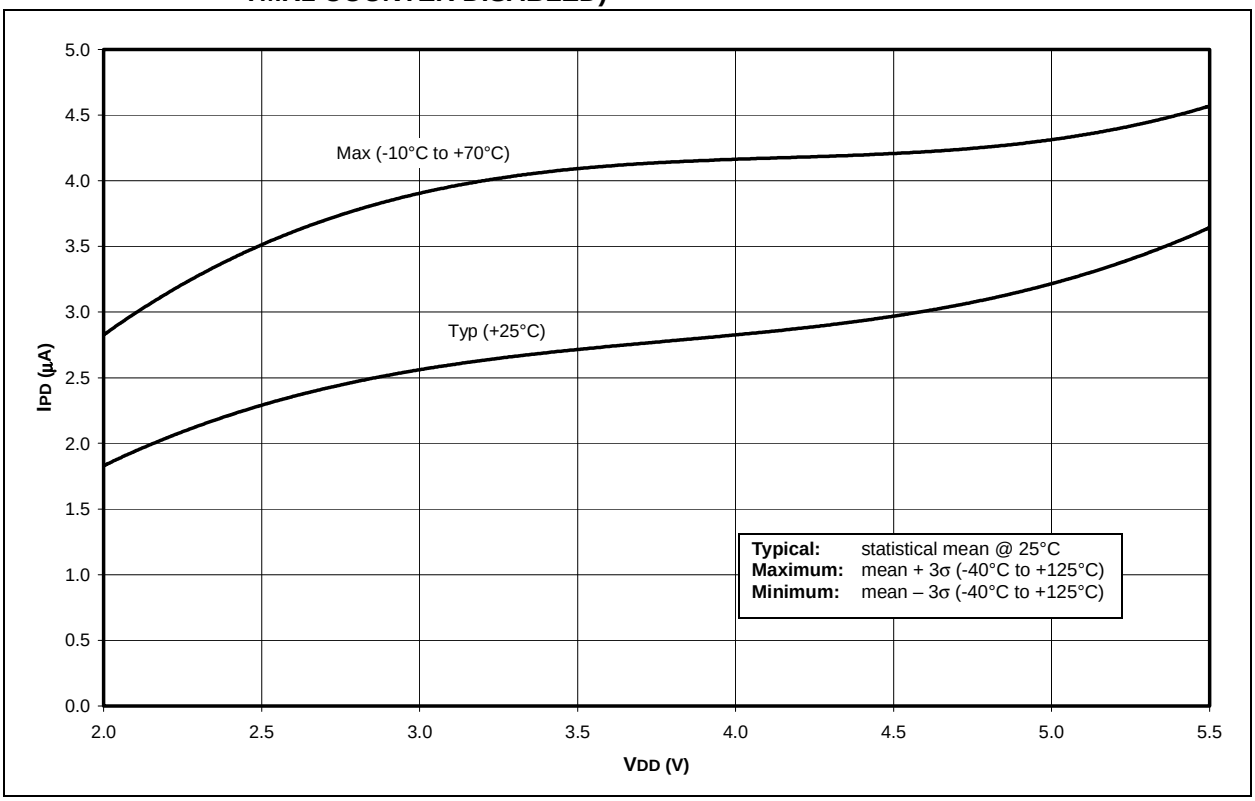


FIGURE 19-15: ΔIPD WDT, -40°C TO +125°C (SLEEP MODE, ALL PERIPHERALS DISABLED)

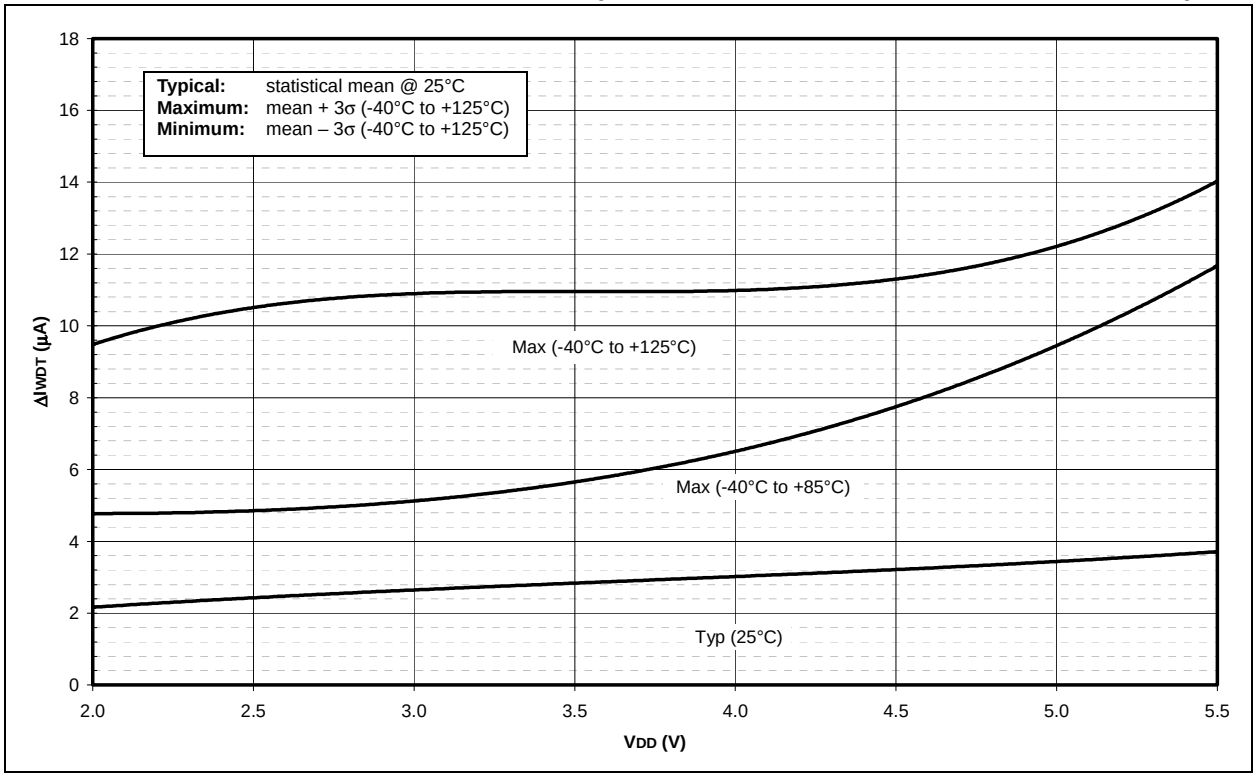


FIGURE 19-16: ΔIPD BOR vs. VDD, -40°C TO +125°C (SLEEP MODE, BOR ENABLED AT 2.00V-2.16V)

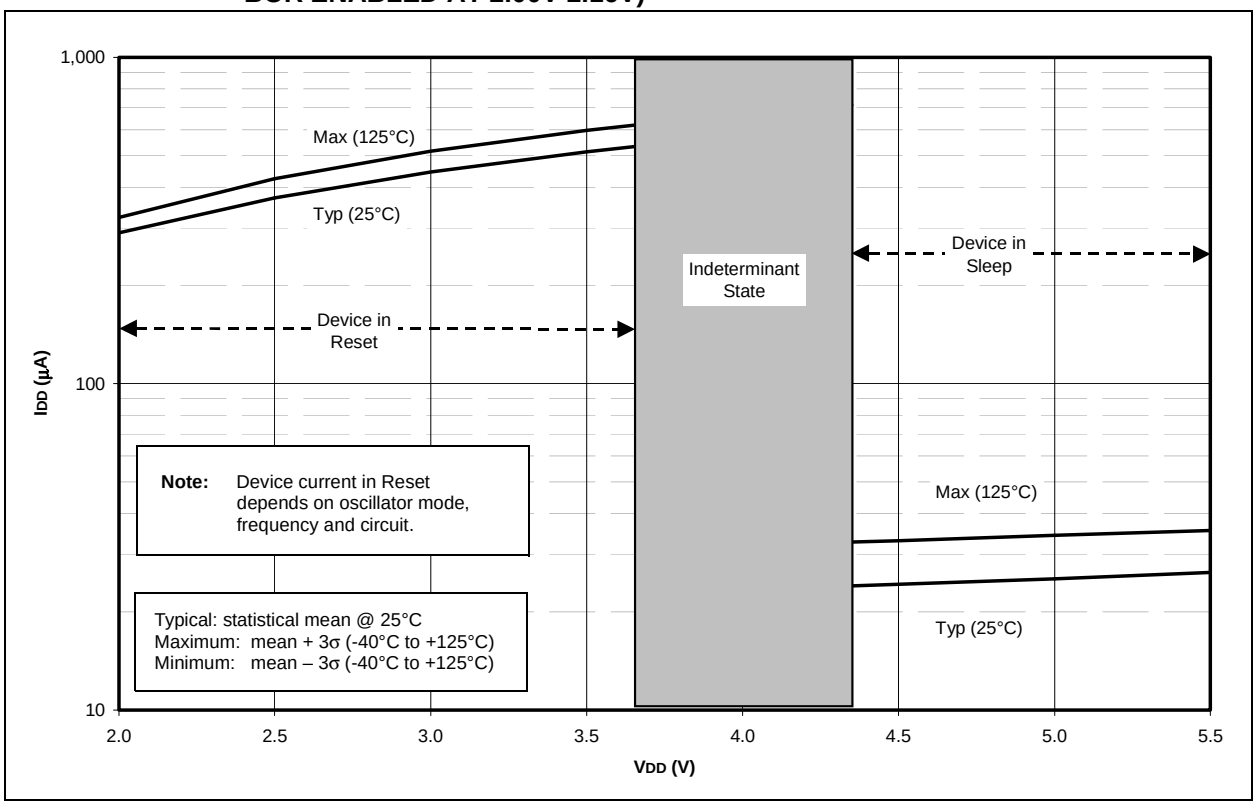


FIGURE 19-17: IPD A/D, -40°C TO +125°C, SLEEP MODE, A/D ENABLED (NOT CONVERTING)

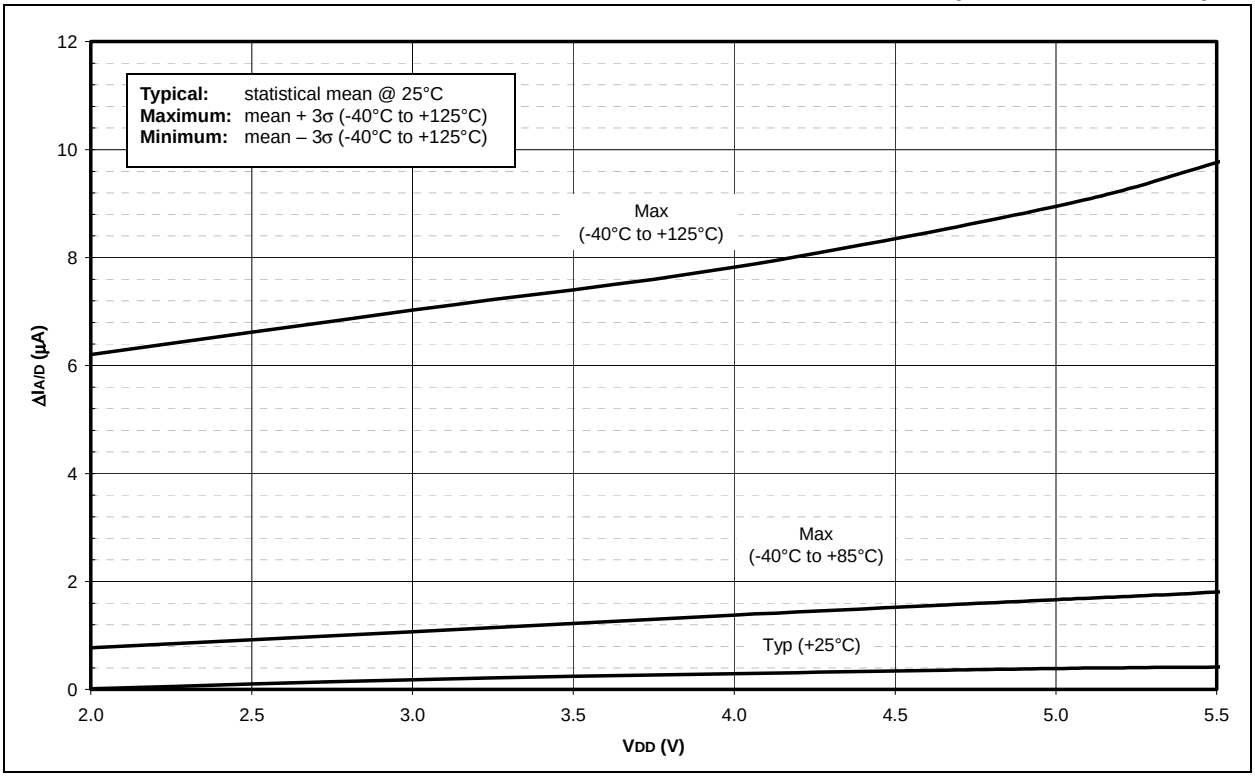


FIGURE 19-18: TYPICAL, MINIMUM AND MAXIMUM VOH vs. IOH (VDD = 5V, -40°C TO +125°C)

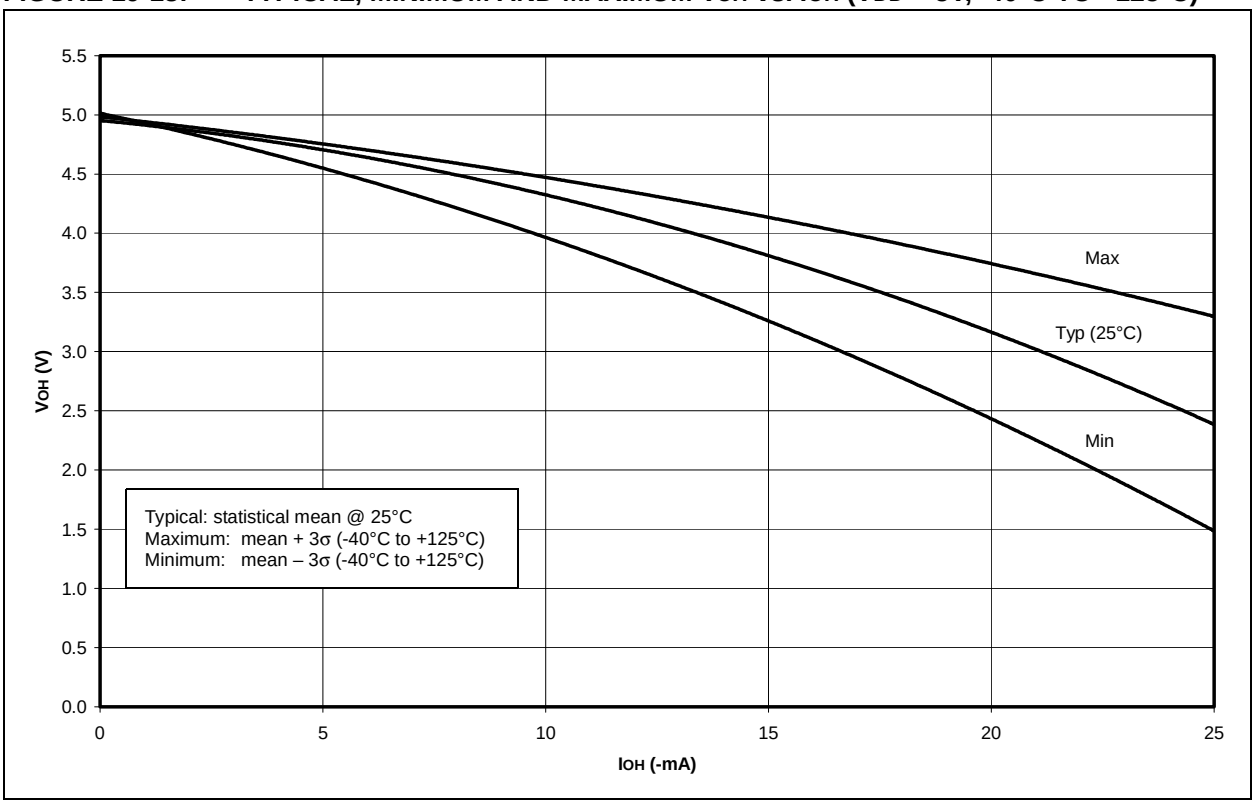


FIGURE 19-19: TYPICAL, MINIMUM AND MAXIMUM V_{OH} vs. I_{OH} (V_{DD} = 3V, -40°C TO +125°C)

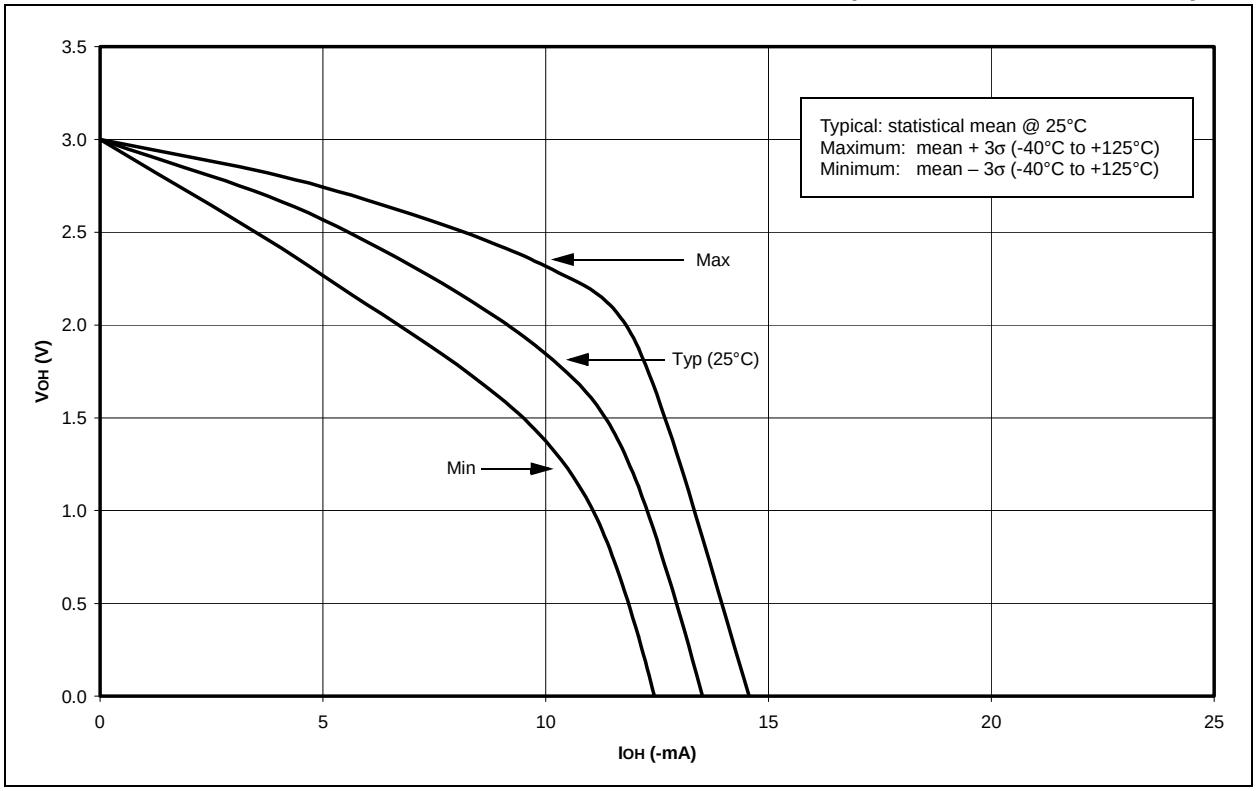


FIGURE 19-20: TYPICAL, MINIMUM AND MAXIMUM V_{OL} vs. I_{OL} (V_{DD} = 5V, -40°C TO +125°C)

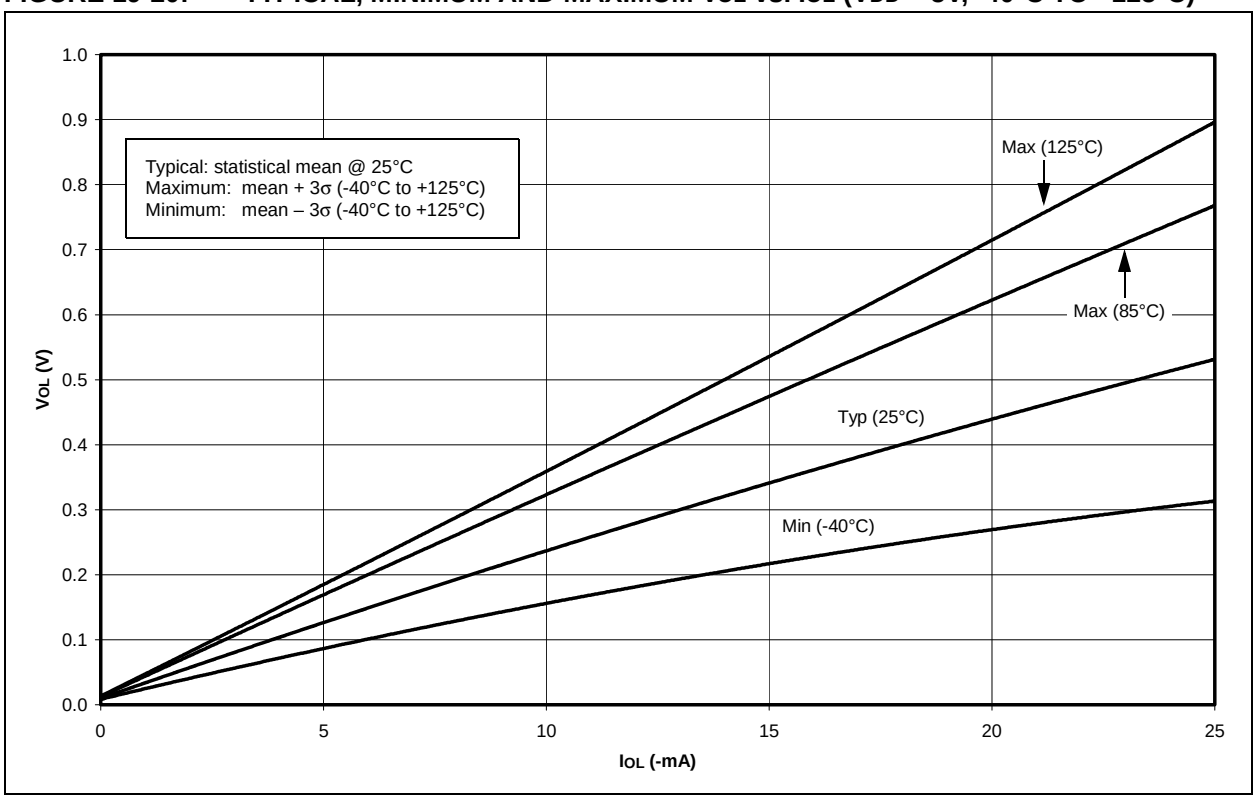


FIGURE 19-21: TYPICAL, MINIMUM AND MAXIMUM V_{OL} vs. I_{OL} (V_{DD} = 3V, -40°C TO +125°C)

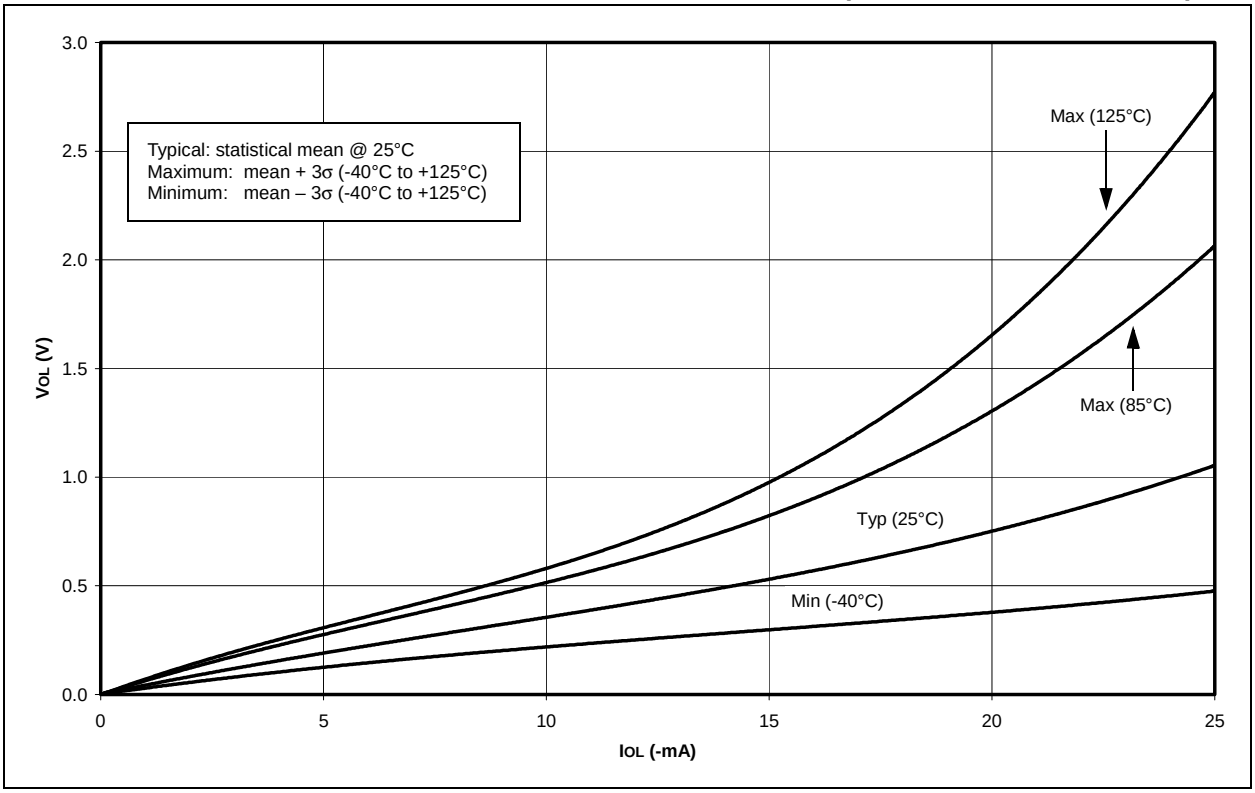


FIGURE 19-22: MINIMUM AND MAXIMUM V_{IN} vs. V_{DD} (TTL INPUT, -40°C TO +125°C)

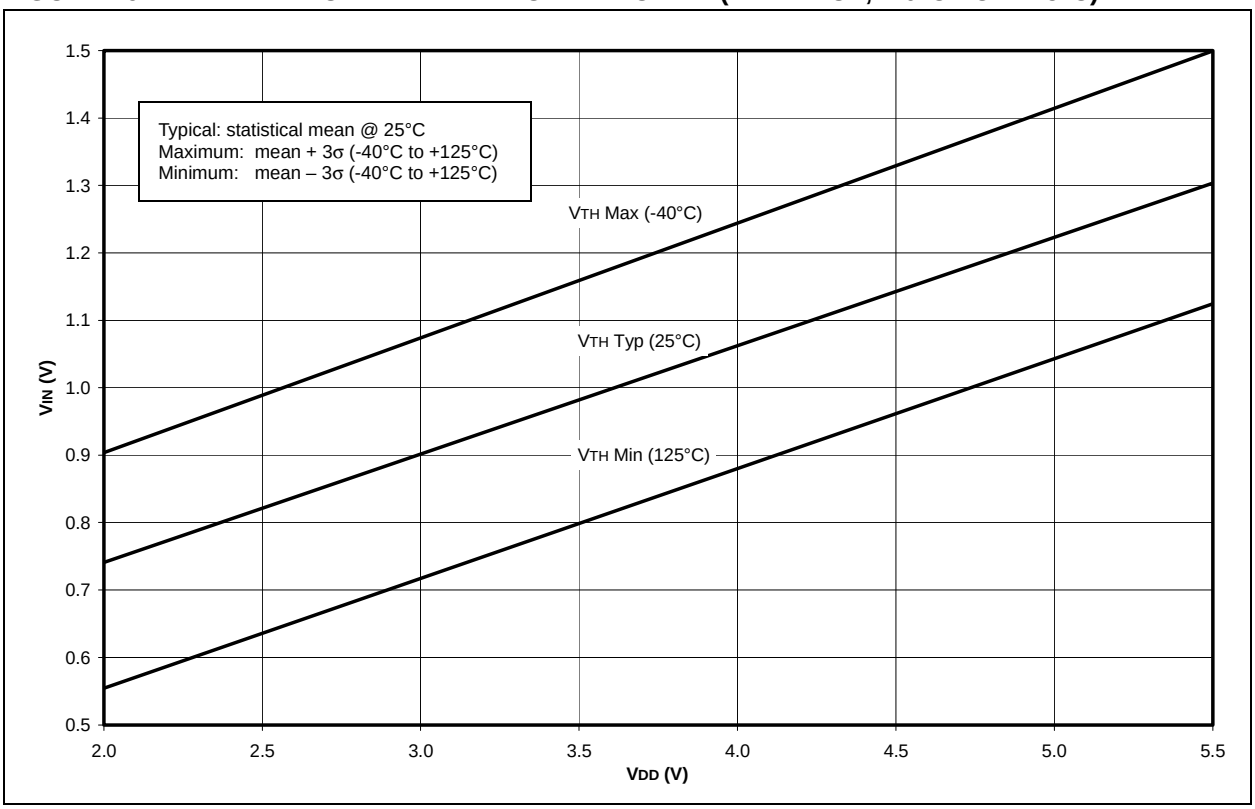


FIGURE 19-23: MINIMUM AND MAXIMUM VIN vs. VDD (ST INPUT, -40°C TO +125°C)

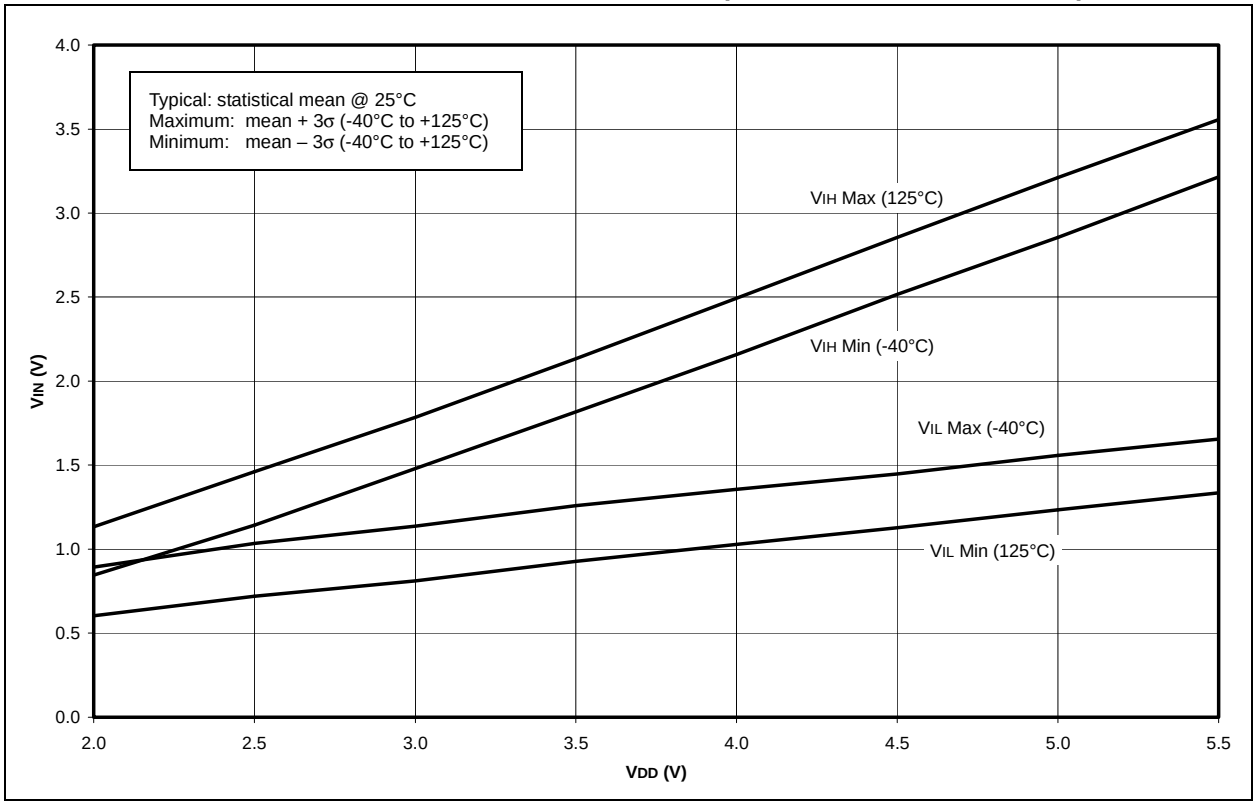


FIGURE 19-24: MINIMUM AND MAXIMUM VIN vs. VDD (I^2C™ INPUT, -40°C TO +125°C)

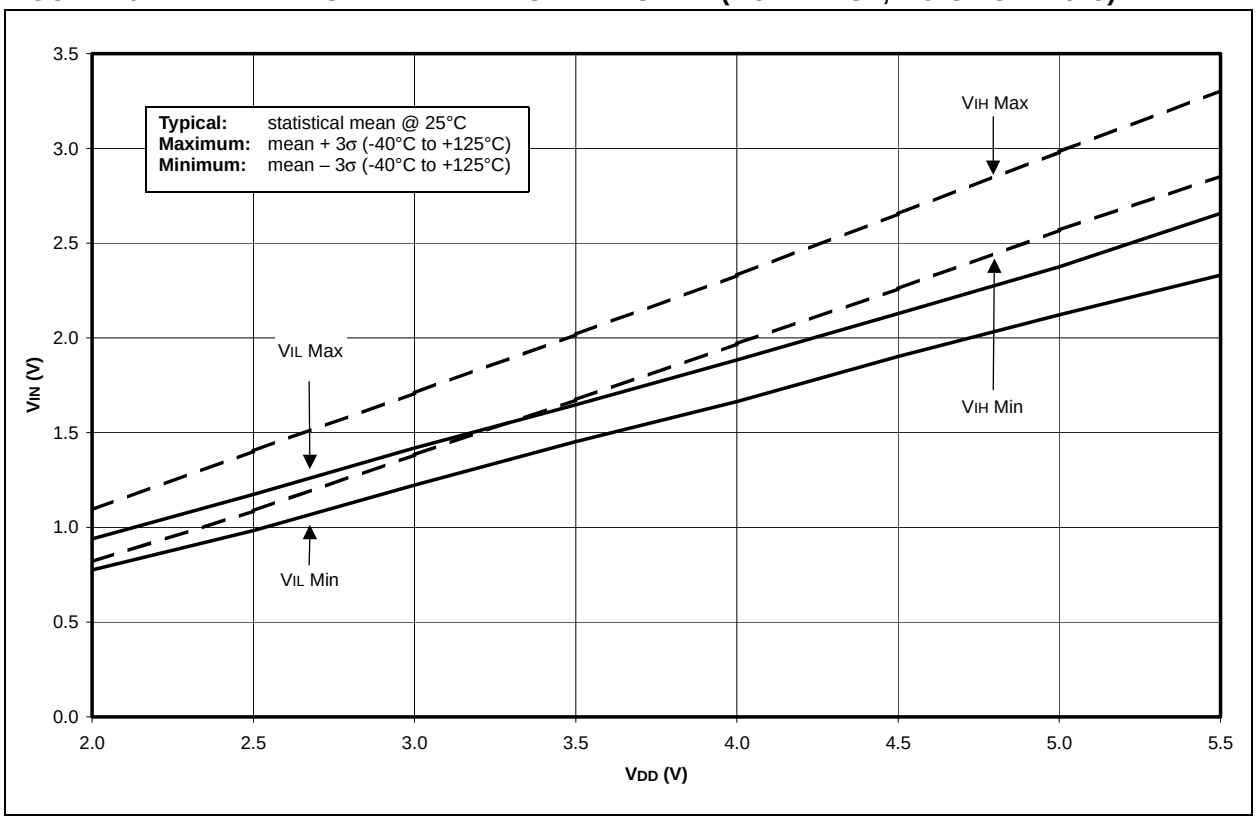


FIGURE 19-25: A/D NONLINEARITY vs. VREFH (VDD = VREFH, -40°C TO +125°C)

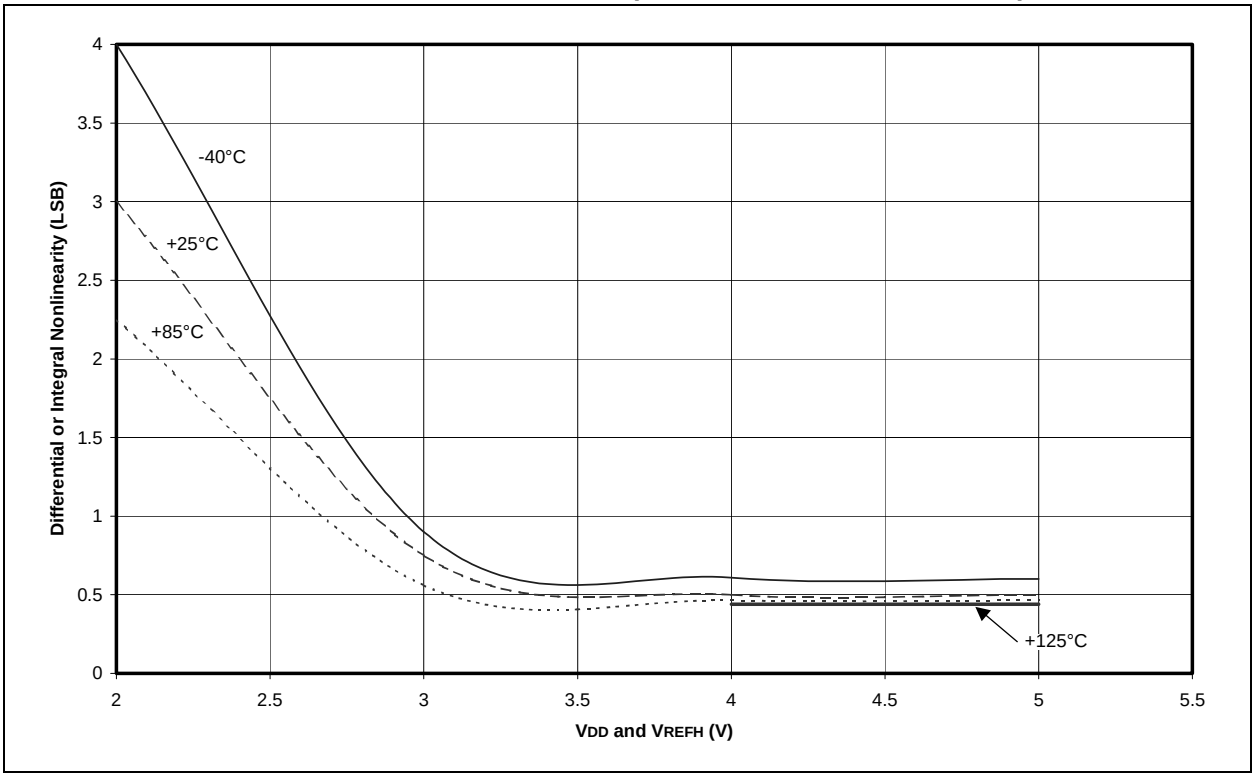
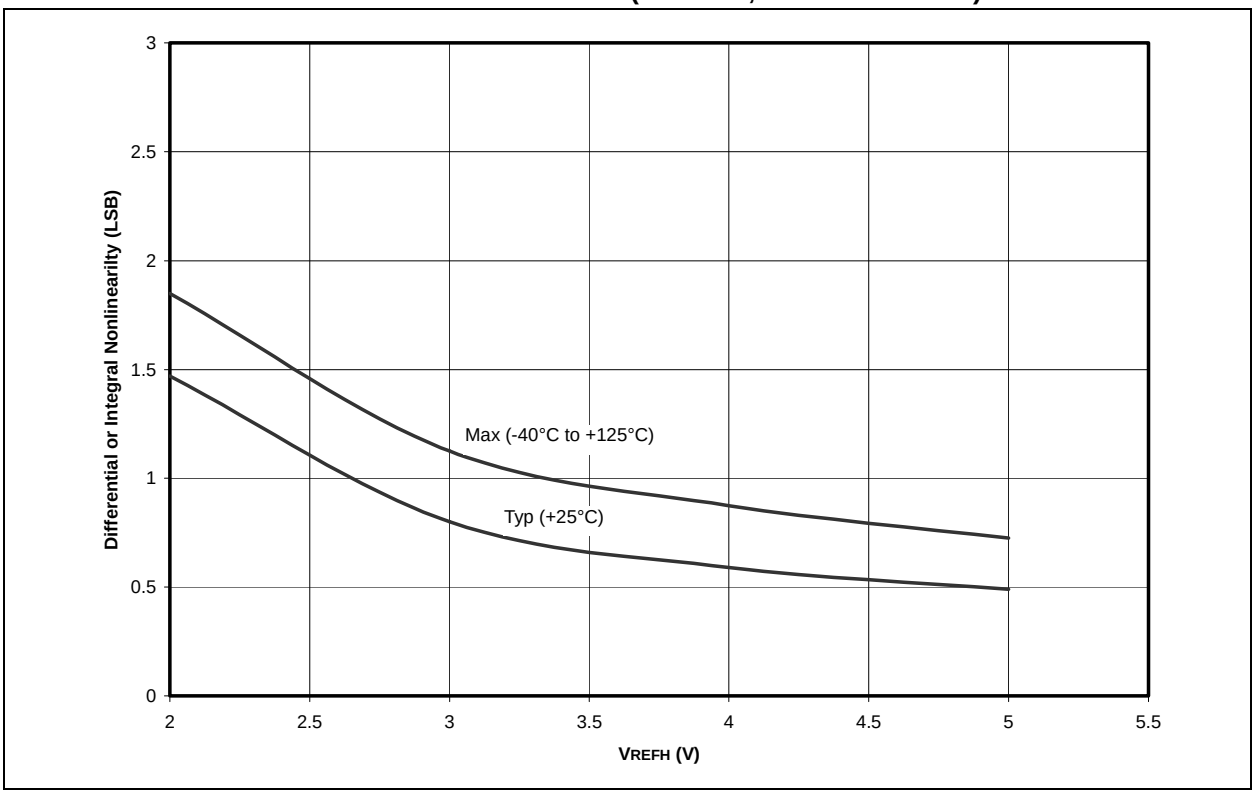


FIGURE 19-26: A/D NONLINEARITY vs. VREFH (VDD = 5V, -40°C TO +125°C)

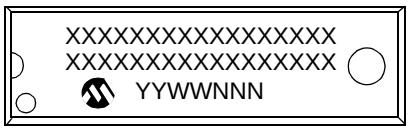


NOTES:

20.0 PACKAGING INFORMATION

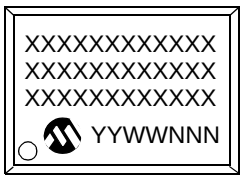
20.1 Package Marking Information

18-Lead PDIP

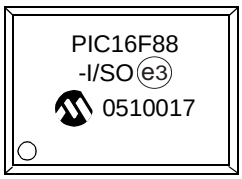


Example

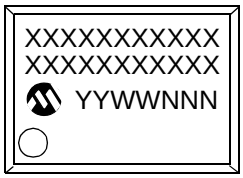
18-Lead SOIC



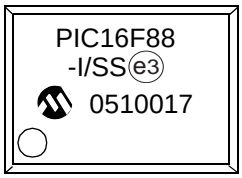
Example



20-Lead SSOP



Example



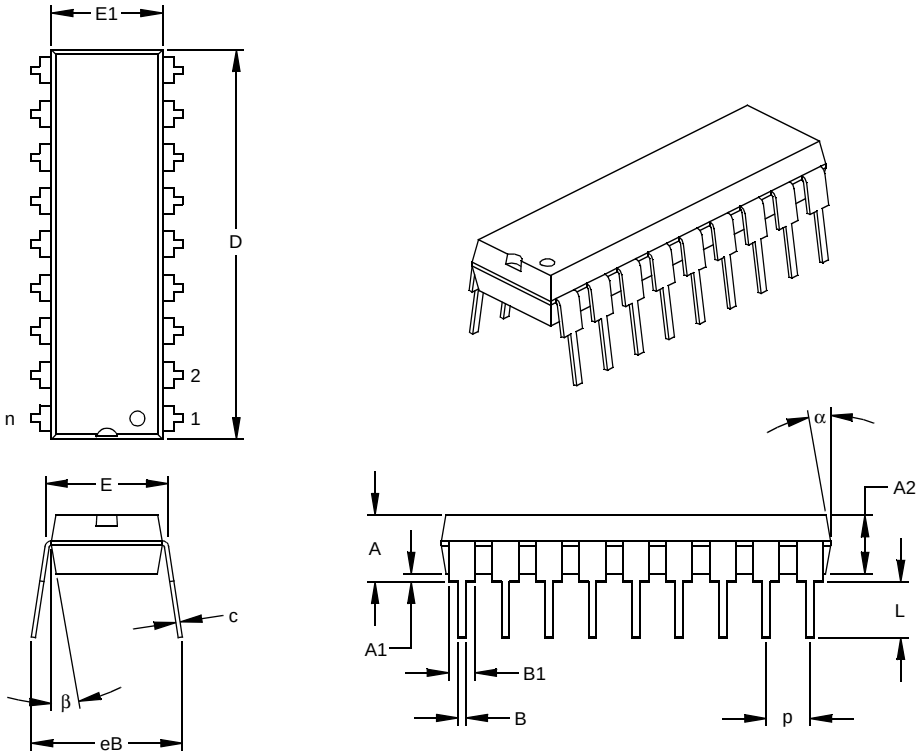
28-Lead QFN

Example

Legend:	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code
	e3	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package.

Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

18-Lead Plastic Dual In-line (P) – 300 mil Body (PDIP)

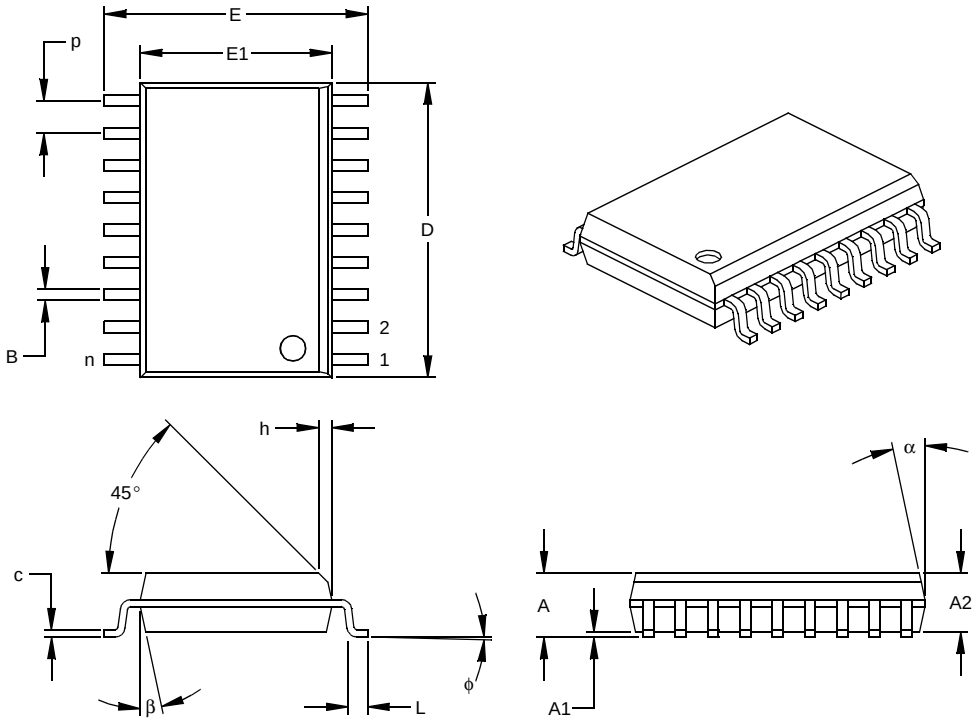


Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		18			18	
Pitch	p		.100			2.54	
Top to Seating Plane	A	.140	.155	.170	3.56	3.94	4.32
Molded Package Thickness	A2	.115	.130	.145	2.92	3.30	3.68
Base to Seating Plane	A1	.015			0.38		
Shoulder to Shoulder Width	E	.300	.313	.325	7.62	7.94	8.26
Molded Package Width	E1	.240	.250	.260	6.10	6.35	6.60
Overall Length	D	.890	.898	.905	22.61	22.80	22.99
Tip to Seating Plane	L	.125	.130	.135	3.18	3.30	3.43
Lead Thickness	c	.008	.012	.015	0.20	0.29	0.38
Upper Lead Width	B1	.045	.058	.070	1.14	1.46	1.78
Lower Lead Width	B	.014	.018	.022	0.36	0.46	0.56
Overall Row Spacing §	eB	.310	.370	.430	7.87	9.40	10.92
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MS-001
Drawing No. C04-007

18-Lead Plastic Small Outline (SO) – Wide, 300 mil Body (SOIC)

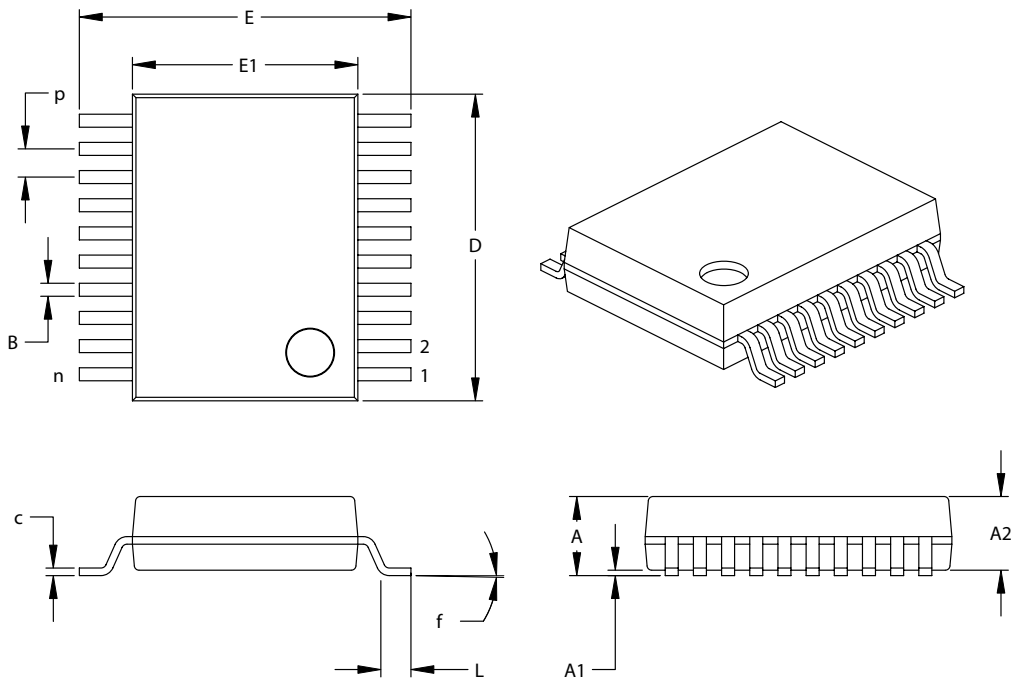


Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		18			18	
Pitch	p		.050			1.27	
Overall Height	A	.093	.099	.104	2.36	2.50	2.64
Molded Package Thickness	A2	.088	.091	.094	2.24	2.31	2.39
Standoff §	A1	.004	.008	.012	0.10	0.20	0.30
Overall Width	E	.394	.407	.420	10.01	10.34	10.67
Molded Package Width	E1	.291	.295	.299	7.39	7.49	7.59
Overall Length	D	.446	.454	.462	11.33	11.53	11.73
Chamfer Distance	h	.010	.020	.029	0.25	0.50	0.74
Foot Length	L	.016	.033	.050	0.41	0.84	1.27
Foot Angle	φ	0	4	8	0	4	8
Lead Thickness	c	.009	.011	.012	0.23	0.27	0.30
Lead Width	B	.014	.017	.020	0.36	0.42	0.51
Mold Draft Angle Top	α	0	12	15	0	12	15
Mold Draft Angle Bottom	β	0	12	15	0	12	15

* Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.
JEDEC Equivalent: MS-013
Drawing No. C04-051

20-Lead Plastic Shrink Small Outline (SS) – 209 mil Body, 5.30 mm (SSOP)



Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		20			20	
Pitch	p		.026			0.65	
Overall Height	A	-	-	.079	-	-	2.00
Molded Package Thickness	A2	.065	.069	.073	1.65	1.75	1.85
Standoff	A1	.002	-	-	0.05	-	-
Overall Width	E	.291	.307	.323	7.40	7.80	8.20
Molded Package Width	E1	.197	.209	.220	5.00	5.30	5.60
Overall Length	D	.272	.283	.289	.295	7.20	7.50
Foot Length	L	.022	.030	.037	0.55	0.75	0.95
Lead Thickness	c	.004	-	.010	0.09	-	0.25
Foot Angle	f	0°	4°	8°	0°	4°	8°
Lead Width	B	.009	-	.015	0.22	-	0.38

*Controlling Parameter

Notes:

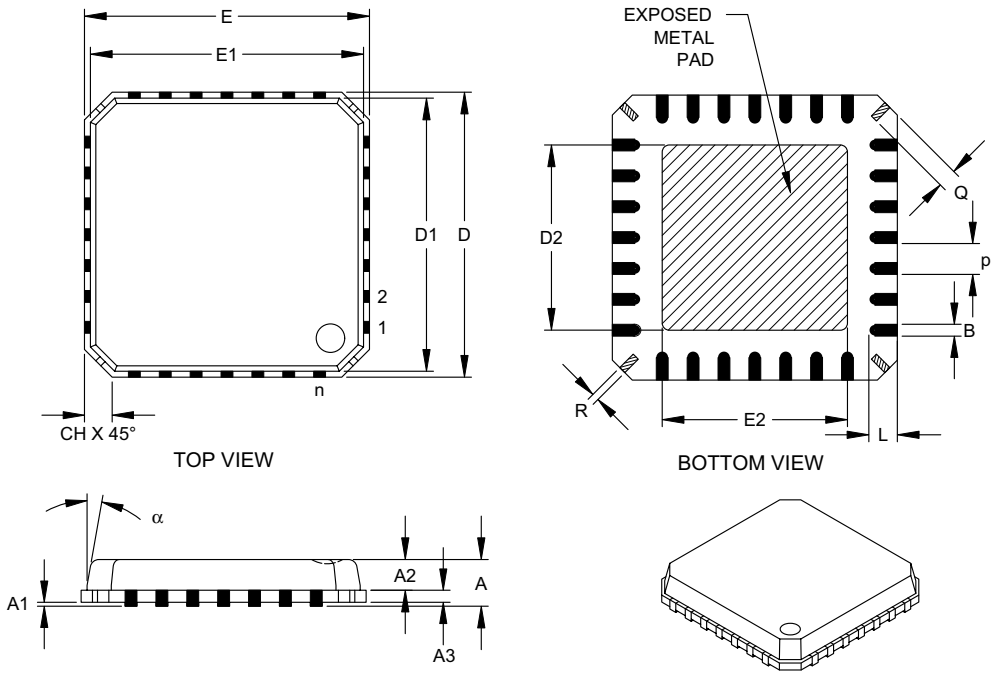
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MO-150

Drawing No. C04-072

Revised 11/03/03

28-Lead Plastic Quad Flat No Lead Package (ML) 6x6 mm Body, Punch Singulated (QFN)



Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		28			28	
Pitch	p		.026 BSC			0.65 BSC	
Overall Height	A		.033	.039		0.85	1.00
Molded Package Thickness	A2		.026	.031		0.65	0.80
Standoff	A1	.000	.0004	.002	0.00	0.01	0.05
Base Thickness	A3		.008 REF			0.20 REF	
Overall Width	E		.236 BSC			6.00 BSC	
Molded Package Width	E1		.226 BSC			5.75 BSC	
Exposed Pad Width	E2	.140	.146	.152	3.55	3.70	3.85
Overall Length	D		.236 BSC			6.00 BSC	
Molded Package Length	D1		.226 BSC			5.75 BSC	
Exposed Pad Length	D2	.140	.146	.152	3.55	3.70	3.85
Lead Width	B	.009	.011	.014	0.23	0.28	0.35
Lead Length	L	.020	.024	.030	0.50	0.60	0.75
Tie Bar Width	R	.005	.007	.010	0.13	0.17	0.23
Tie Bar Length	Q	.012	.016	.026	0.30	0.40	0.65
Chamfer	CH	.009	.017	.024	0.24	0.42	0.60
Mold Draft Angle Top	α			12°			12°

*Controlling Parameter

Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC equivalent: MO-220

Drawing No. C04-114

NOTES:

APPENDIX A: REVISION HISTORY

Revision A (November 2003)

Original data sheet for PIC16F87/88 devices.

Revision B (August 2003)

The specifications in **Section 18.0 "Electrical Characteristics"** have been updated to include the addition of maximum specifications to the DC Characteristics tables, text clarification has been made to **Section 4.6.2 "Clock Switching"** and there have been minor updates to the data sheet text.

Revision C (January 2005)

This revision includes the DC and AC Characteristics Graphs and Tables. The Electrical Specifications in **Section 18.0 "Electrical Characteristics"** have been updated and there have been minor corrections to the data sheet text.

APPENDIX B: DEVICE DIFFERENCES

The differences between the devices in this data sheet are listed in Table B-1.

TABLE B-1: DIFFERENCES BETWEEN THE PIC16F87 AND PIC16F88

Features	PIC16F87	PIC16F88
Analog-to-Digital Converter	N/A	10-bit, 7-channel

NOTES:

INDEX

A

B

C

D

L

M

O

P

R

S

T

V

W

NOTES:

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com, click on Customer Change Notification and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support
- Development Systems Information Line

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://support.microchip.com

In addition, there is a Development Systems Information Line which lists the latest versions of Microchip's development systems software products. This line also provides information on how customers can receive currently available upgrade kits.

The Development Systems Information Line numbers are:

1-800-755-2345 – United States and most of Canada

1-480-792-7302 – Other International Locations

READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

To: Technical Publications Manager Total Pages Sent _____

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply?____Y ____N

Device: PIC16F87/88 Literature Number: DS30487C

Questions:

1. What are the best features of this document?

_____

_____

2. How does this document meet your hardware and software development needs?

_____

_____

3. Do you find the organization of this document easy to follow? If not, why?

_____

_____

4. What additions to the document do you think would enhance the structure and subject?

_____

_____

5. What deletions from the document could be made without affecting the overall usefulness?

_____

_____

6. Is there any incorrect or misleading information (what and where)?

_____

_____

7. How would you improve this document?

_____

_____

PIC16F87/88 PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	X	/XX	XXX
Device	Temperature Range	Package	Pattern

Device	PIC16F87: Standard VDD range PIC16F87T: (Tape and Reel) PIC16LF87: Extended VDD range
Temperature Range	- = 0°C to +70°C I = -40°C to +85°C (Industrial) E = -40°C to +125°C (Extended)
Package	P = PDIP SO = SOIC SS = SSOP ML = QFN
Pattern	QTP, SQTP, ROM Code (factory specified) or Special Requirements. Blank for OTP and Windowed devices.

Examples:

a) PIC16F87-I/P = Industrial temp., PDIP package, Extended VDD limits.

b) PIC16F87-I/SO = Industrial temp., SOIC package, normal VDD limits.

Note 1: F = CMOS Flash
LF = Low-power CMOS Flash
2: T = in tape and reel – SOIC, SSOP packages only.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://support.microchip.com
Web Address:
www.microchip.com

Atlanta
Alpharetta, GA
Tel: 770-640-0034
Fax: 770-640-0307

Boston
Westford, MA
Tel: 978-692-3848
Fax: 978-692-3821

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

San Jose
Mountain View, CA
Tel: 650-215-1444
Fax: 650-961-0286

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8676-6200
Fax: 86-28-8676-6599

China - Fuzhou
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

China - Qingdao
Tel: 86-532-502-7355
Fax: 86-532-502-7205

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-2229-0061
Fax: 91-80-2229-0062

India - New Delhi
Tel: 91-11-5160-8631
Fax: 91-11-5160-8632

Japan - Kanagawa
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Taiwan - Hsinchu
Tel: 886-3-572-9526
Fax: 886-3-572-6459

EUROPE

Austria - Weis
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark - Ballerup
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Massy
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Ismaning
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

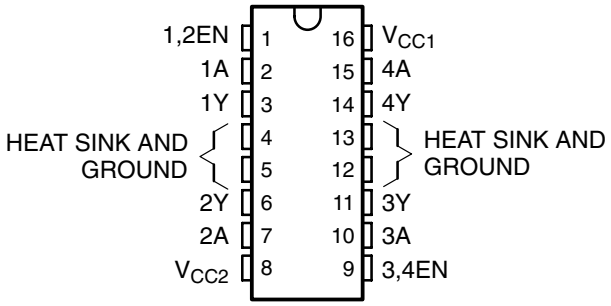
Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

England - Berkshire
Tel: 44-118-921-5869
Fax: 44-118-921-5820

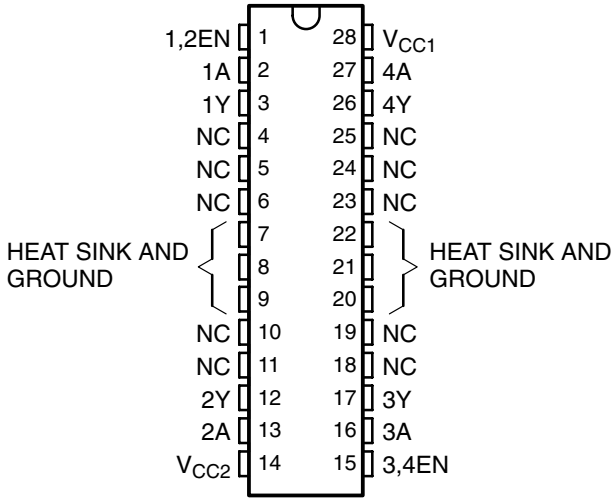
10/20/04

- **Featuring Unitrode L293 and L293D Products Now From Texas Instruments**
- **Wide Supply-Voltage Range: 4.5 V to 36 V**
- **Separate Input-Logic Supply**
- **Internal ESD Protection**
- **Thermal Shutdown**
- **High-Noise-Immunity Inputs**
- **Functionally Similar to SGS L293 and SGS L293D**
- **Output Current 1 A Per Channel (600 mA for L293D)**
- **Peak Output Current 2 A Per Channel (1.2 A for L293D)**
- **Output Clamp Diodes for Inductive Transient Suppression (L293D)**

L293 . . . N OR NE PACKAGE
L293D . . . NE PACKAGE
(TOP VIEW)



L293 . . . DWP PACKAGE
(TOP VIEW)



description/ordering information

The L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

All inputs are TTL compatible. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are enabled, and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled, and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a full-H (or bridge) reversible drive suitable for solenoid or motor applications.

ORDERING INFORMATION

T_A	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	HSOP (DWP)	Tube of 20	L293DWP	L293DWP
	PDIP (N)	Tube of 25	L293N	L293N
	PDIP (NE)	Tube of 25	L293NE	L293NE
		Tube of 25	L293DNE	L293DNE

† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

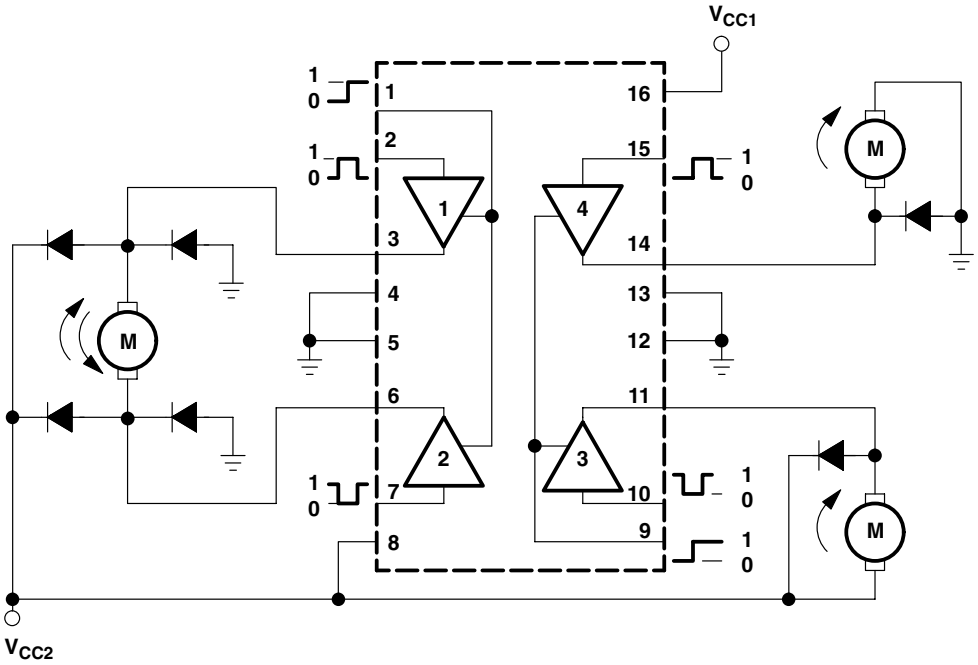
description/ordering information (continued)

On the L293, external high-speed output clamp diodes should be used for inductive transient suppression.

A V_{CC1} terminal, separate from V_{CC2}, is provided for the logic inputs to minimize device power dissipation.

The L293and L293D are characterized for operation from 0°C to 70°C.

block diagram



NOTE: Output diodes are internal in L293D.

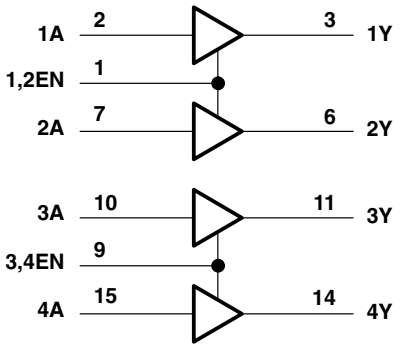
FUNCTION TABLE
(each driver)

INPUTS†		OUTPUT
A	EN	Y
H	H	H
L	H	L
X	L	Z

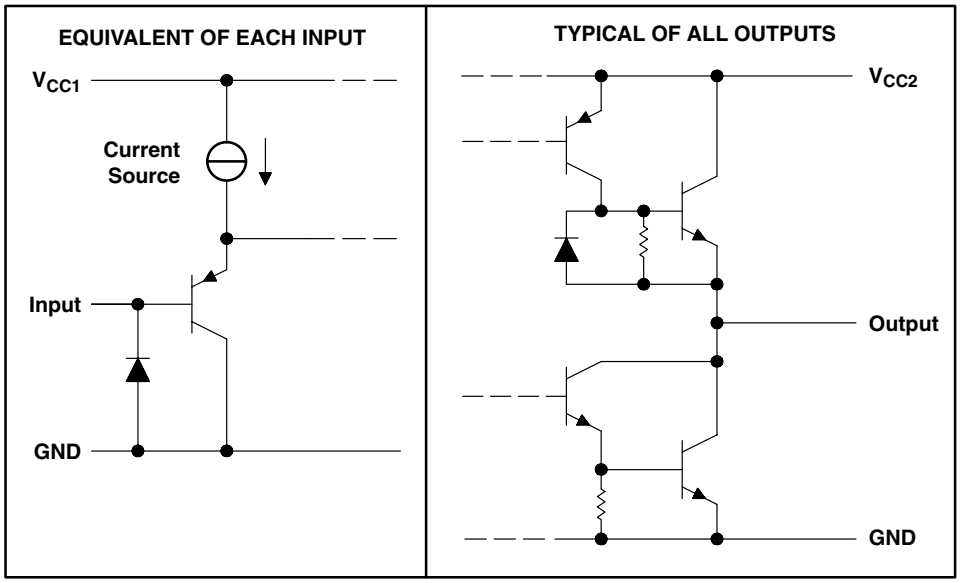
H = high level, L = low level, X = irrelevant, Z = high impedance (off)

† In the thermal shutdown mode, the output is in the high-impedance state, regardless of the input levels.

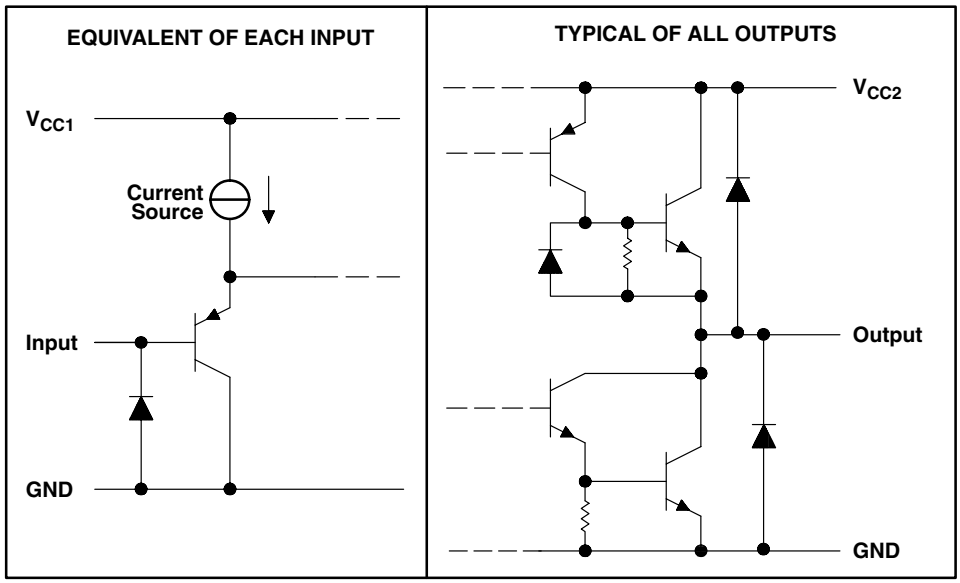
logic diagram



schematics of inputs and outputs (L293)



schematics of inputs and outputs (L293D)



absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{CC1} (see Note 1)		36 V
Output supply voltage, V_{CC2}		36 V
Input voltage, V_I		7 V
Output voltage range, V_O		–3 V to V_{CC2} + 3 V
Peak output current, I_O (nonrepetitive, t ≤ 5 ms): L293		±2 A
Peak output current, I_O (nonrepetitive, t ≤ 100 µs): L293D		±1.2 A
Continuous output current, I_O: L293		±1 A
Continuous output current, I_O: L293D		±600 mA
Package thermal impedance, θ_{JA} (see Notes 2 and 3):	DWP package	TBD°C/W
	N package	67°C/W
	NE package	TBD°C/W
Maximum junction temperature, T_J		150°C
Storage temperature range, T_{stg}		–65°C to 150°C

† Stresses beyond those listed under “absolute maximum ratings” may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under “recommended operating conditions” is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTES:
1. All voltage values are with respect to the network ground terminal.
2. Maximum power dissipation is a function of $T_J(max)$, θ_{JA}, and T_A. The maximum allowable power dissipation at any allowable ambient temperature is $P_D = (T_J(max) - T_A)/\theta_{JA}$. Operating at the absolute maximum T_J of 150°C can affect reliability.
3. The package thermal impedance is calculated in accordance with JESD 51-7.

recommended operating conditions

			MIN	MAX	UNIT
	Supply voltage	V_{CC1}	4.5	7	V
		V_{CC2}	V_{CC1}	36	
V_{IH}	High-level input voltage	$V_{CC1} \leq 7$ V	2.3	V_{CC1}	V
		$V_{CC1} \geq 7$ V	2.3	7	V
V_{IL}	Low-level output voltage		–0.3†	1.5	V
T_A	Operating free-air temperature		0	70	°C

† The algebraic convention, in which the least positive (most negative) designated minimum, is used in this data sheet for logic voltage levels.

electrical characteristics, V_{CC1} = 5 V, V_{CC2} = 24 V, T_A = 25°C

PARAMETER			TEST CONDITIONS		MIN	TYP	MAX	UNIT
V_{OH}	High-level output voltage		L293: $I_{OH} = -1$ A L293D: $I_{OH} = -0.6$ A		$V_{CC2} - 1.8$	$V_{CC2} - 1.4$		V
V_{OL}	Low-level output voltage		L293: $I_{OL} = 1$ A L293D: $I_{OL} = 0.6$ A			1.2	1.8	V
V_{OKH}	High-level output clamp voltage		L293D: $I_{OK} = -0.6$ A			$V_{CC2} + 1.3$		V
V_{OKL}	Low-level output clamp voltage		L293D: $I_{OK} = 0.6$ A			1.3		V
I_{IH}	High-level input current	A	$V_I = 7$ V			0.2	100	µA
		EN				0.2	10	
I_{IL}	Low-level input current	A	$V_I = 0$			–3	–10	µA
		EN				–2	–100	
I_{CC1}	Logic supply current		$I_O = 0$	All outputs at high level		13	22	mA
				All outputs at low level		35	60	
				All outputs at high impedance		8	24	
I_{CC2}	Output supply current		$I_O = 0$	All outputs at high level		14	24	mA
				All outputs at low level		2	6	
				All outputs at high impedance		2	4	

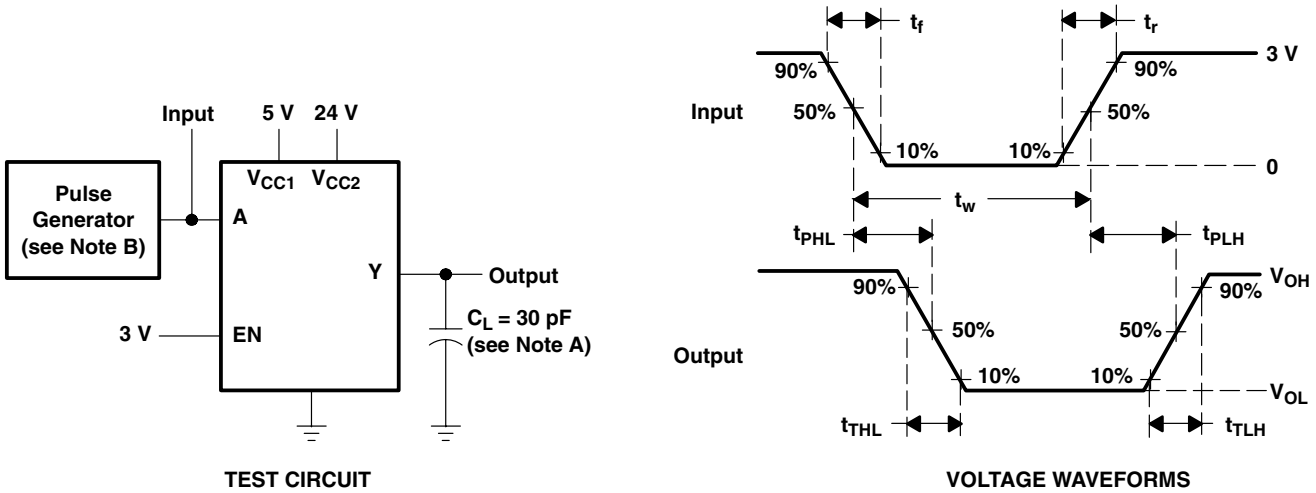
switching characteristics, V_{CC1} = 5 V, V_{CC2} = 24 V, T_A = 25°C

PARAMETER		TEST CONDITIONS	L293NE, L293DNE			UNIT
			MIN	TYP	MAX	
t_{PLH}	Propagation delay time, low-to-high-level output from A input	$C_L = 30$ pF, See Figure 1		800		ns
t_{PHL}	Propagation delay time, high-to-low-level output from A input			400		ns
t_{TLH}	Transition time, low-to-high-level output			300		ns
t_{THL}	Transition time, high-to-low-level output			300		ns

switching characteristics, V_{CC1} = 5 V, V_{CC2} = 24 V, T_A = 25°C

PARAMETER		TEST CONDITIONS	L293DWP, L293N L293DN			UNIT
			MIN	TYP	MAX	
t_{PLH}	Propagation delay time, low-to-high-level output from A input	$C_L = 30$ pF, See Figure 1		750		ns
t_{PHL}	Propagation delay time, high-to-low-level output from A input			200		ns
t_{TLH}	Transition time, low-to-high-level output			100		ns
t_{THL}	Transition time, high-to-low-level output			350		ns

PARAMETER MEASUREMENT INFORMATION



NOTES: A. C_L includes probe and jig capacitance.
B. The pulse generator has the following characteristics: $t_r \leq 10$ ns, $t_f \leq 10$ ns, $t_w = 10$ µs, PRR = 5 kHz, $Z_O = 50\ \Omega$.

Figure 1. Test Circuit and Voltage Waveforms

APPLICATION INFORMATION

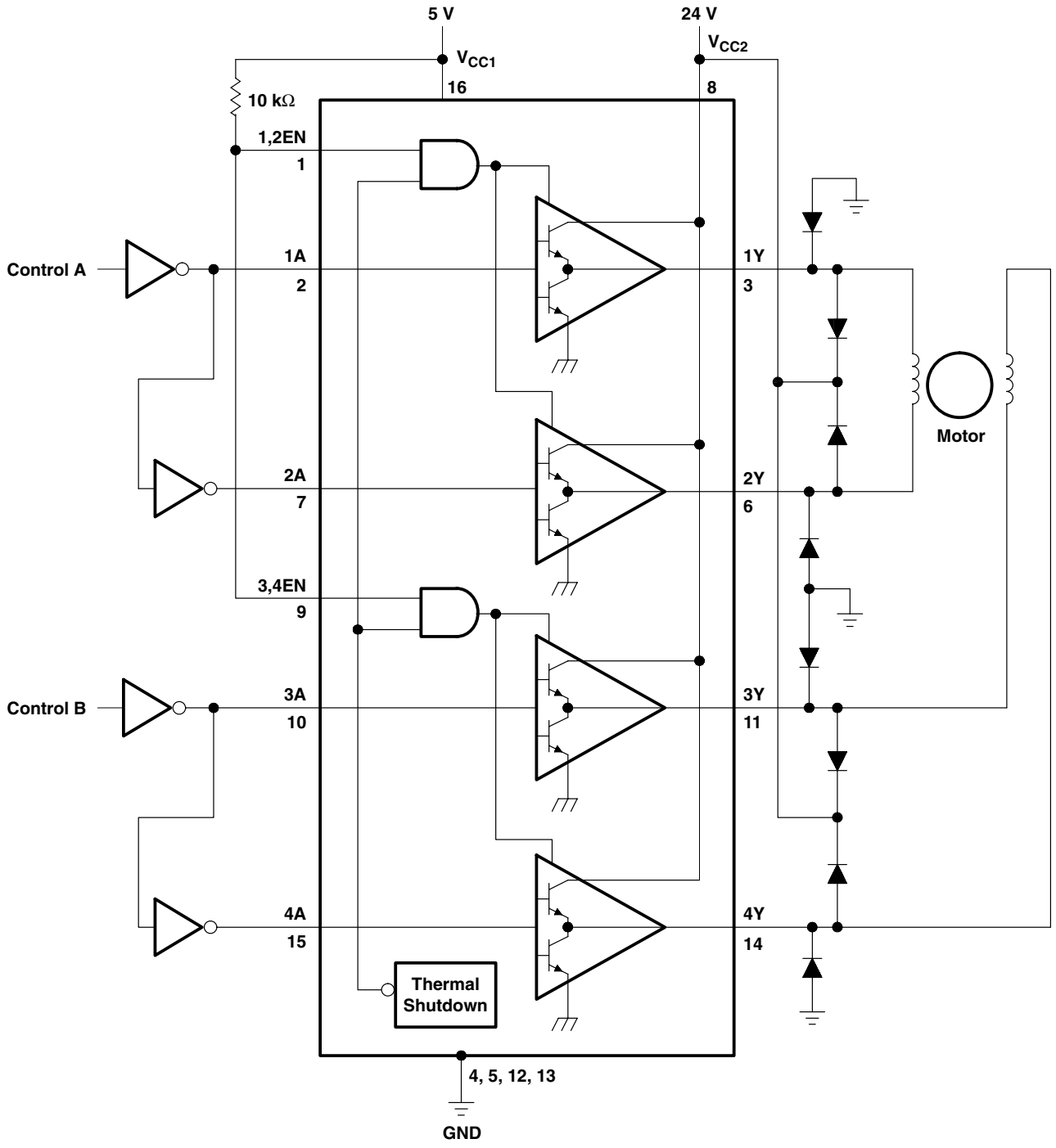


Figure 2. Two-Phase Motor Driver (L293)

APPLICATION INFORMATION

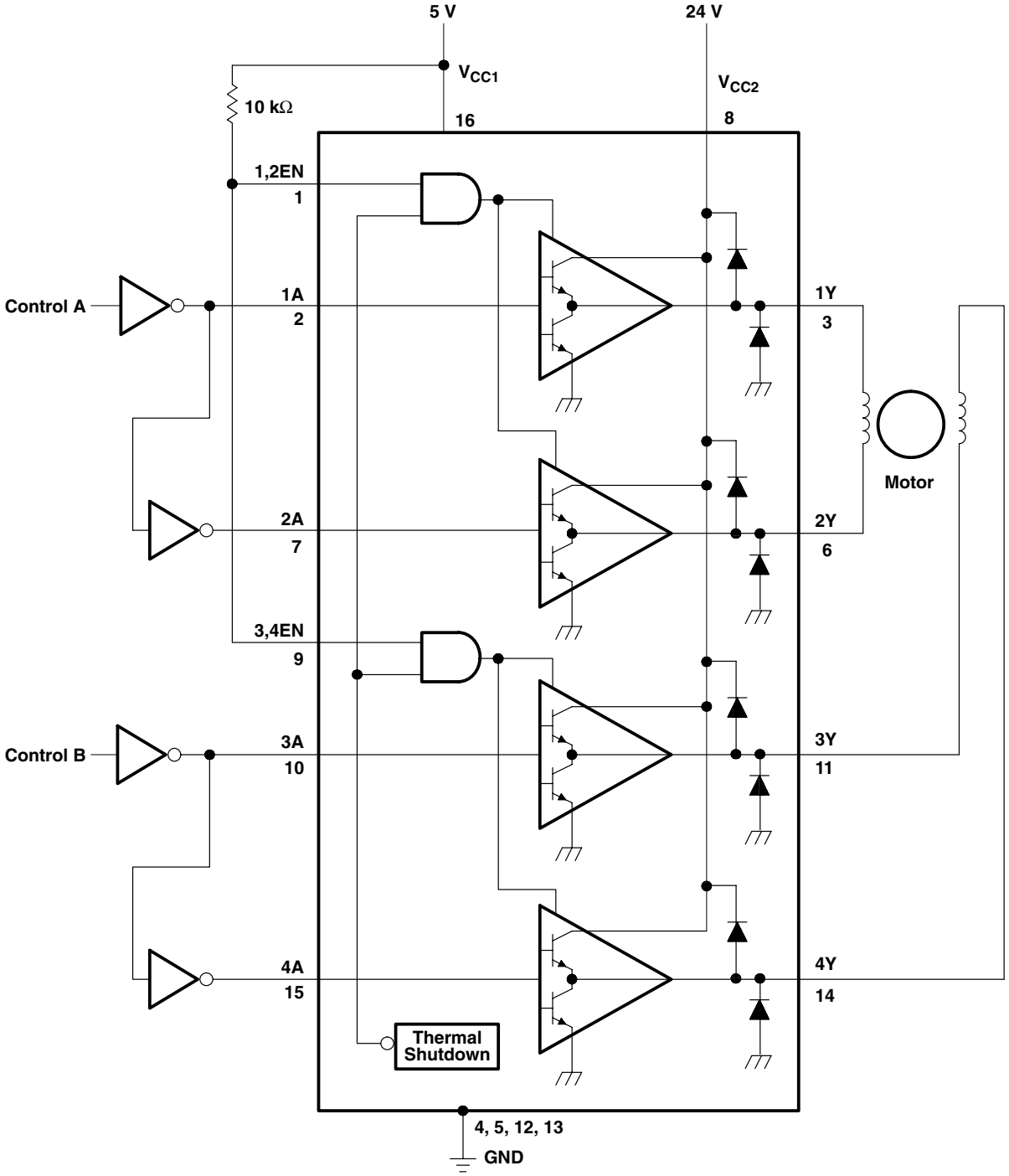


Figure 3. Two-Phase Motor Driver (L293D)

APPLICATION INFORMATION

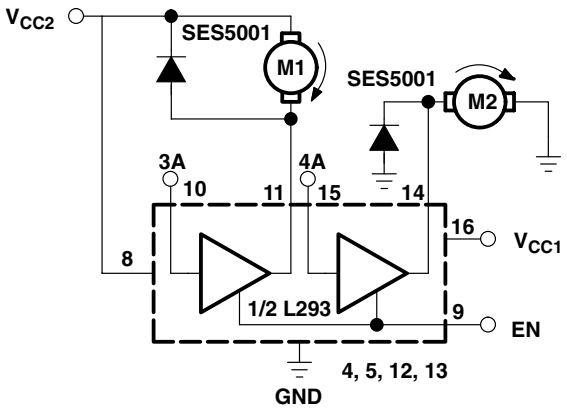


Figure 4. DC Motor Controls (connections to ground and to supply voltage)

EN	3A	M1	4A	M2
H	H	Fast motor stop	H	Run
H	L	Run	L	Fast motor stop
L	X	Free-running motor stop	X	Free-running motor stop

L = low, H = high, X = don't care

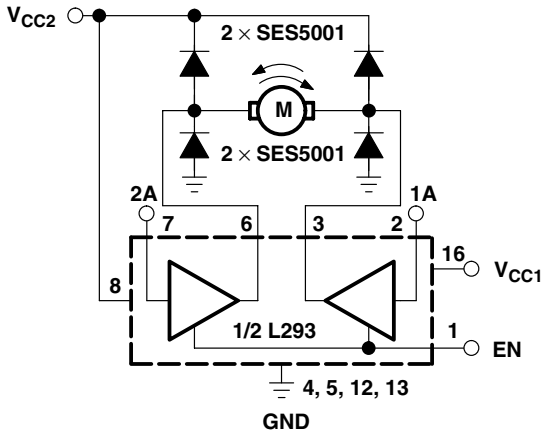
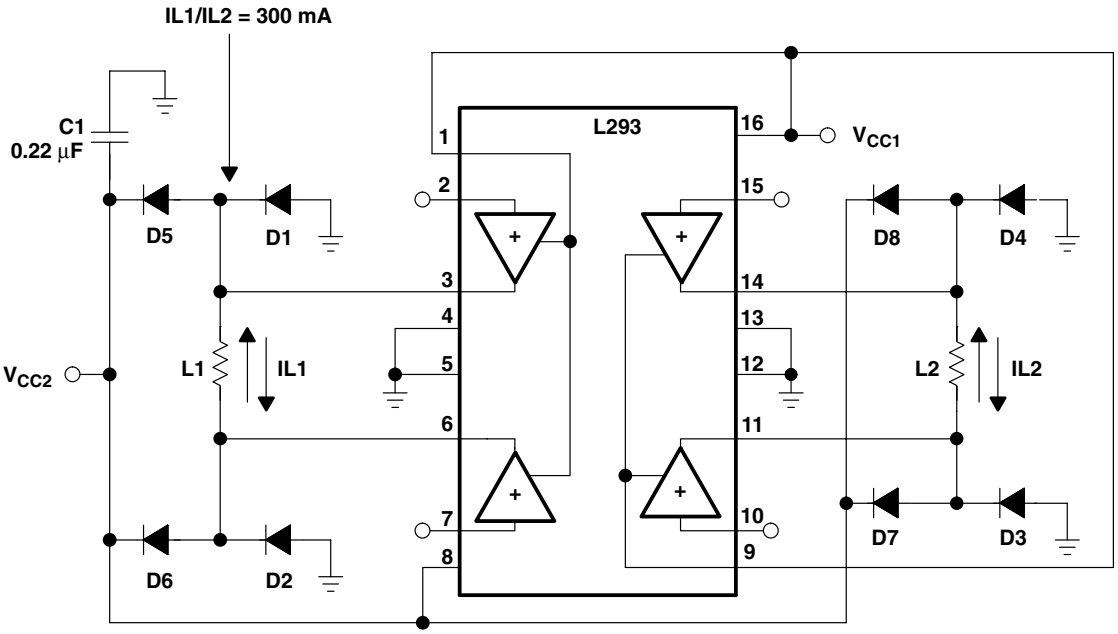


Figure 5. Bidirectional DC Motor Control

EN	1A	2A	FUNCTION
H	L	H	Turn right
H	H	L	Turn left
H	L	L	Fast motor stop
H	H	H	Fast motor stop
L	X	X	Fast motor stop

L = low, H = high, X = don't care

APPLICATION INFORMATION



D1–D8 = SES5001

Figure 6. Bipolar Stepping-Motor Control

mounting instructions

The Rthj-amp of the L293 can be reduced by soldering the GND pins to a suitable copper area of the printed circuit board or to an external heat sink.

Figure 9 shows the maximum package power P_{TOT} and the θ_{JA} as a function of the side ℓ of two equal square copper areas having a thickness of 35 µm (see Figure 7). In addition, an external heat sink can be used (see Figure 8).

During soldering, the pin temperature must not exceed 260°C, and the soldering time must not exceed 12 seconds.

The external heatsink or printed circuit copper area must be connected to electrical ground.

APPLICATION INFORMATION

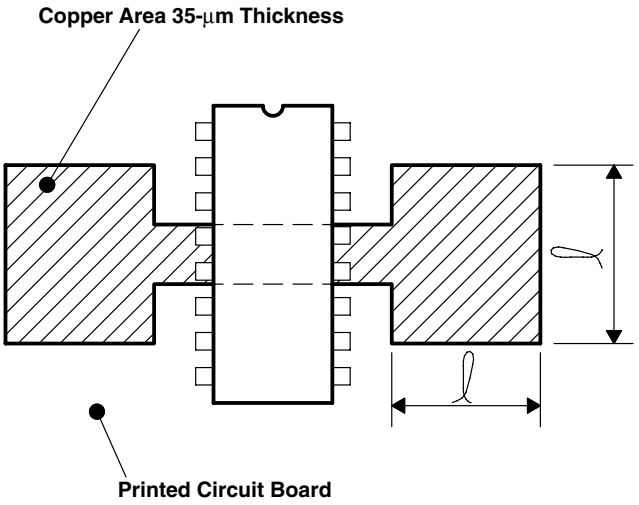


Figure 7. Example of Printed Circuit Board Copper Area (used as heat sink)

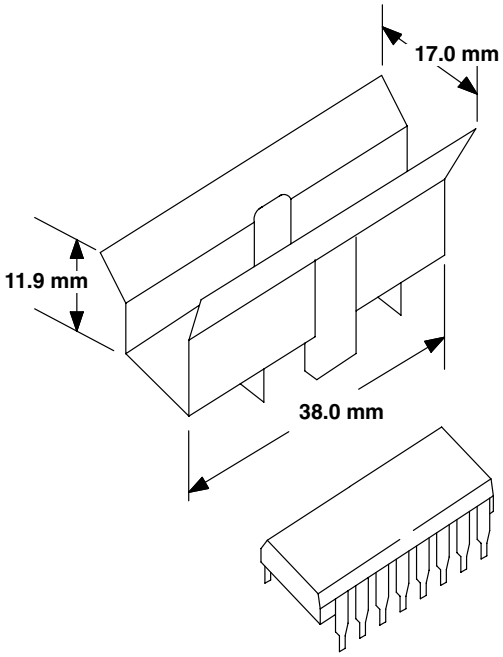


Figure 8. External Heat Sink Mounting Example (θ_{JA} = 25°C/W)

APPLICATION INFORMATION

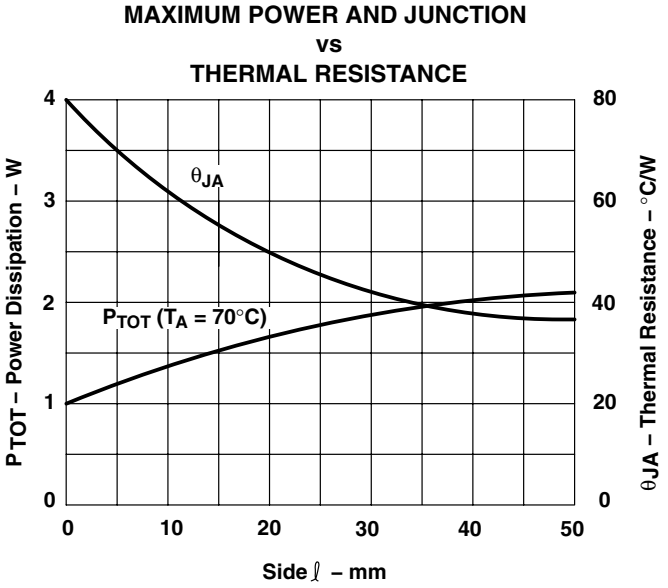


Figure 9

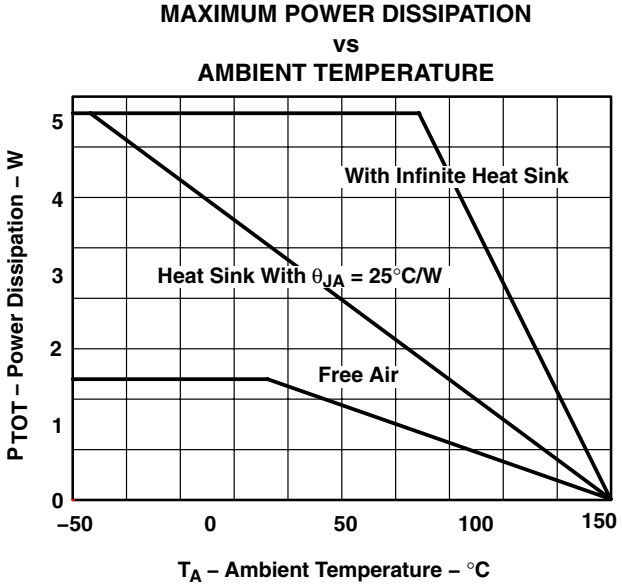


Figure 10

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/ Ball Finish	MSL Peak Temp (3)	Samples (Requires Login)
L293DNE	ACTIVE	PDIP	NE	16	25	Pb-Free (RoHS)	CU NIPDAU	N / A for Pkg Type	
L293DNEE4	ACTIVE	PDIP	NE	16	25	Pb-Free (RoHS)	CU NIPDAU	N / A for Pkg Type	
L293DWP	OBSOLETE	SOIC	DW	28		TBD	Call TI	Call TI	
L293DWPG4	OBSOLETE	SOIC	DW	28		TBD	Call TI	Call TI	
L293DWPTR	OBSOLETE	SO PowerPAD	DWP	28		TBD	Call TI	Call TI	
L293N	OBSOLETE	PDIP	N	16		TBD	Call TI	Call TI	
L293NE	ACTIVE	PDIP	NE	16	25	Pb-Free (RoHS)	CU NIPDAU	N / A for Pkg Type	
L293NEE4	ACTIVE	PDIP	NE	16	25	Pb-Free (RoHS)	CU NIPDAU	N / A for Pkg Type	
L293NG4	OBSOLETE	PDIP	N	16		TBD	Call TI	Call TI	

(1) The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

(2) Eco Plan - The planned eco-friendly classification: Pb-Free (RoHS), Pb-Free (RoHS Exempt), or Green (RoHS & no Sb/Br) - please check http://www.ti.com/productcontent for the latest availability information and additional product content details.
TBD: The Pb-Free/Green conversion plan has not been defined.
Pb-Free (RoHS): TI's terms "Lead-Free" or "Pb-Free" mean semiconductor products that are compatible with the current RoHS requirements for all 6 substances, including the requirement that lead not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, TI Pb-Free products are suitable for use in specified lead-free processes.
Pb-Free (RoHS Exempt): This component has a RoHS exemption for either 1) lead-based flip-chip solder bumps used between the die and package, or 2) lead-based die adhesive used between the die and leadframe. The component is otherwise considered Pb-Free (RoHS compatible) as defined above.
Green (RoHS & no Sb/Br): TI defines "Green" to mean Pb-Free (RoHS compatible), and free of Bromine (Br) and Antimony (Sb) based flame retardants (Br or Sb do not exceed 0.1% by weight in homogeneous material)

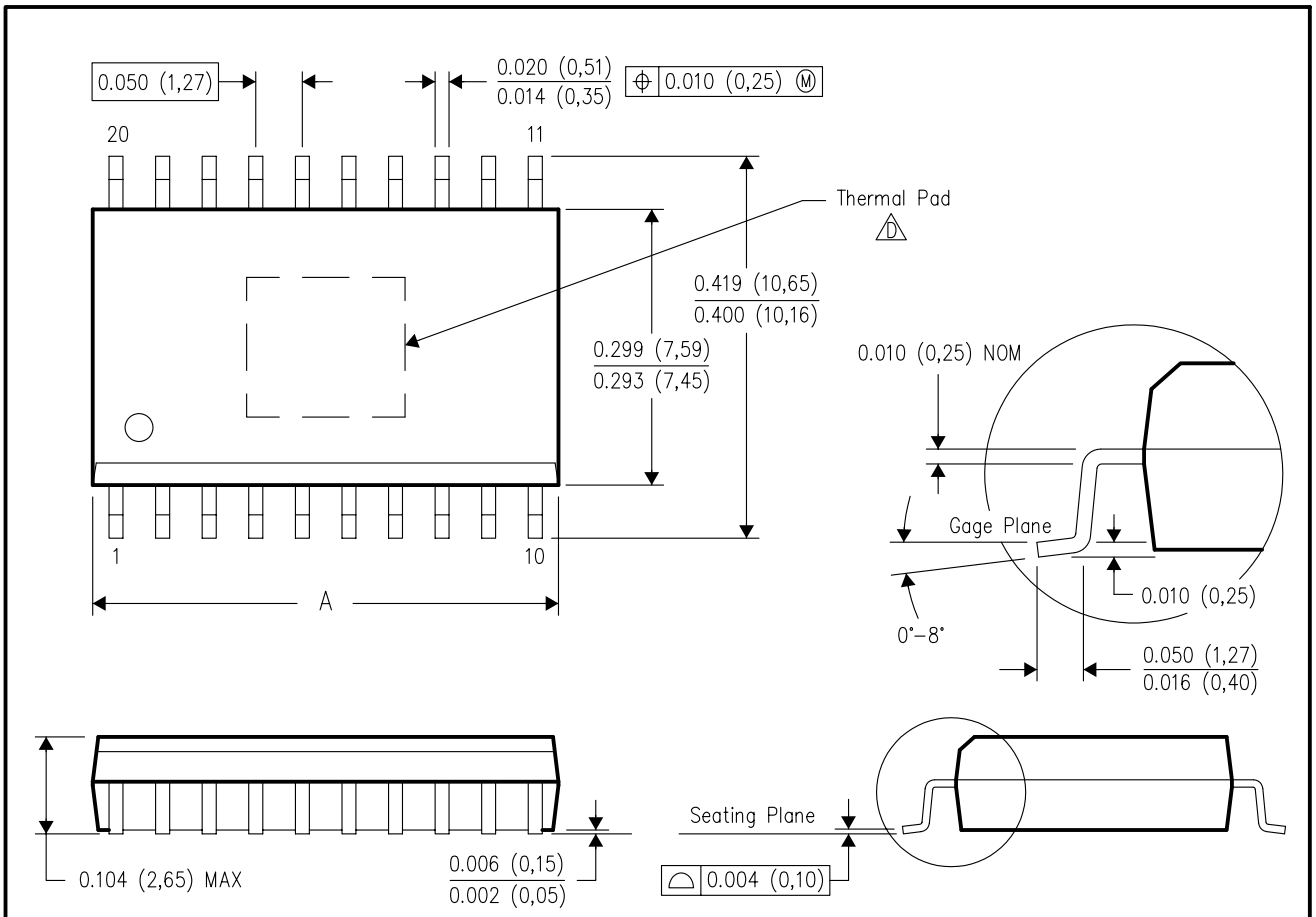
(3) MSL, Peak Temp. -- The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

DWP (R–PDSO–G**) PowerPAD™ PLASTIC SMALL–OUTLINE PACKAGE

20 PINS SHOWN



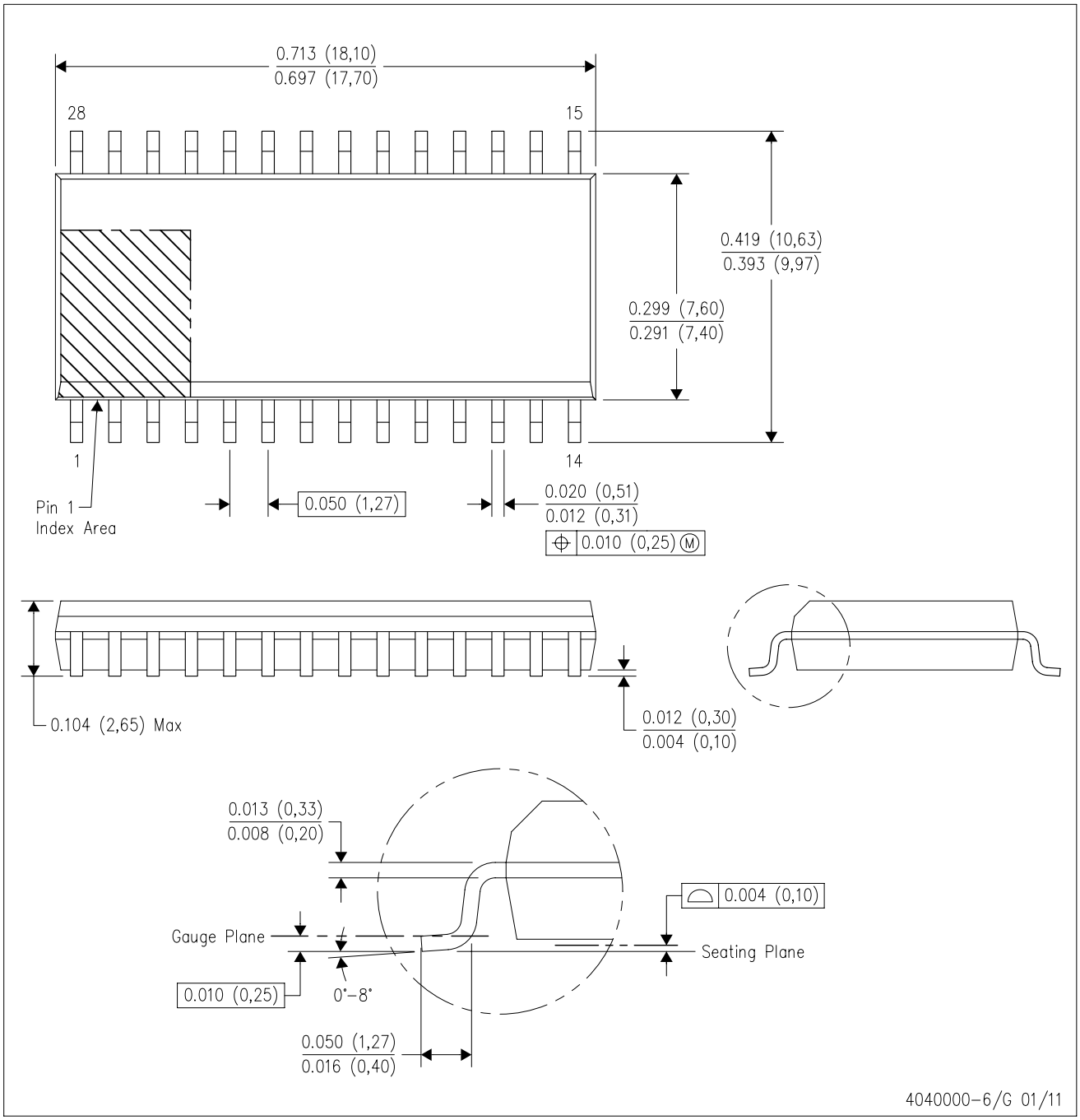
DIM \ PINS **	16	20	24	28
A MAX	0.410 (10,41)	0.510 (12,95)	0.610 (15,49)	0.710 (18,03)
A MIN	0.400 (10,16)	0.500 (12,70)	0.600 (15,24)	0.700 (17,78)

4147575/C 02/05

NOTES:
- A. All linear dimensions are in inches (millimeters).
- B. This drawing is subject to change without notice.
- C. Body dimensions do not include mold flash or protrusion not to exceed 0.006 (0,15).
- D. This package is designed to be soldered to a thermal pad on the board. Refer to Technical Brief, PowerPad Thermally Enhanced Package, Texas Instruments Literature No. SLMA002 for information regarding recommended board layout. This document is available at www.ti.com <http://www.ti.com>. See the product data sheet for details regarding the exposed thermal pad dimensions.

PowerPAD is a trademark of Texas Instruments.

DW (R–PDSO–G28) PLASTIC SMALL OUTLINE



NOTES:
A. All linear dimensions are in inches (millimeters). Dimensioning and tolerancing per ASME Y14.5M–1994.
B. This drawing is subject to change without notice.
C. Body dimensions do not include mold flash or protrusion not to exceed 0.006 (0,15).
D. Falls within JEDEC MS–013 variation AE.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

National Semiconductor

January 2003

LM2940/LM2940C
1A Low Dropout Regulator

General Description

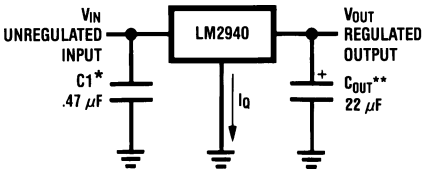
The LM2940/LM2940C positive voltage regulator features the ability to source 1A of output current with a dropout voltage of typically 0.5V and a maximum of 1V over the entire temperature range. Furthermore, a quiescent current reduction circuit has been included which reduces the ground current when the differential between the input voltage and the output voltage exceeds approximately 3V. The quiescent current with 1A of output current and an input-output differential of 5V is therefore only 30 mA. Higher quiescent currents only exist when the regulator is in the dropout mode ($V_{IN} - V_{OUT} \leq 3V$).

Designed also for vehicular applications, the LM2940/LM2940C and all regulated circuitry are protected from reverse battery installations or 2-battery jumps. During line transients, such as load dump when the input voltage can momentarily exceed the specified maximum operating voltage, the regulator will automatically shut down to protect both the internal circuits and the load. The LM2940/LM2940C cannot be harmed by temporary mirror-image insertion. Familiar regulator features such as short circuit and thermal overload protection are also provided.

Features

- Dropout voltage typically 0.5V @I_O = 1A
- Output current in excess of 1A
- Output voltage trimmed before assembly
- Reverse battery protection
- Internal short circuit current limit
- Mirror image insertion protection
- P^+ Product Enhancement tested

Typical Application



00882203

*Required if regulator is located far from power supply filter.

**C_{OUT} must be at least 22 μF to maintain stability. May be increased without bound to maintain regulation during transients. Locate as close as possible to the regulator. This capacitor must be rated over the same operating temperature range as the regulator and the ESR is critical; see curve.

Ordering Information

Temperature Range	Output Voltage						Package
	5.0	8.0	9.0	10	12	15	
0˚C ≤ T_J ≤ 125˚C	LM2940CT-5.0		LM2940CT-9.0		LM2940CT-12	LM2940CT-15	TO-220
	LM2940CS-5.0		LM2940CS-9.0		LM2940CS-12	LM2940CS-15	TO-263
–40˚C ≤ T_J ≤ 125˚C	LM2940LD-5.0	LM2940LD-8.0	LM2940LD-9.0	LM2940LD-10	LM2940LD-12	LM2940LD-15	LLP 1k Units Tape and Reel
	LM2940LDX-5.0	LM2940LDX-8.0	LM2940LDX-9.0	LM2940LDX-10	LM2940LDX-12	LM2940LDX-15	LLP 4.5k Units Tape and Reel
–40˚C ≤ T_J ≤ 125˚C	LM2940T-5.0	LM2940T-8.0	LM2940T-9.0	LM2940T-10	LM2940T-12		TO-220
	LM2940S-5.0	LM2940S-8.0	LM2940S-9.0	LM2940S-10	LM2940S-12		TO-263
–40˚C ≤ T_J ≤ 85˚C	LM2940IMP-5.0	LM2940IMP-8.0	LM2940IMP-9.0	LM2940IMP-10	LM2940IMP-12	LM2940IMP-15	SOT-223
	LM2940IMPX-5.0	LM2940IMPX-8.0	LM2940IMPX-9.0	LM2940IMPX-10	LM2940IMPX-12	LM2940IMPX-15	SOT-223 in Tape and Reel
SOT-223 Package Marking	L53B	L54B	L0EB	L55B	L56B	L70B	

The physical size of the SOT-223 is too small to contain the full device part number. The package markings indicated are what will appear on the actual device.

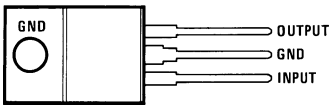
Ordering Information (Continued)

Temperature Range	Output Voltage				Package
	5.0	8.0	12	15	
−55˚C ≤ T_J ≤ 125˚C	LM2940J-5.0/883 5962-8958701EA	LM2940J-8.0/883 5962-9088301QEA	LM2940J-12/883 5962-9088401QEA	LM2940J-15/883 5962-9088501QEA	J16A
	LM2940WG5.0/883 5962-8958701XA				WG16A

For information on military temperature range products, please go to the Mil/Aero Web Site at http://www.national.com/appinfo/milaero/index.html.

Connection Diagrams

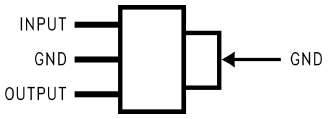
(TO-220) Plastic Package



00882202

Front View
Order Number LM2940CT-5.0, LM2940CT-9.0, LM2940CT-12, LM2940CT-15, LM2940T-5.0, LM2940T-8.0, LM2940T-9.0, LM2940T-10 or LM2940T-12
See NS Package Number TO3B

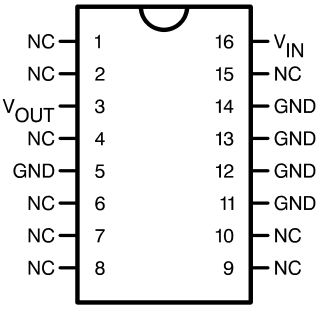
3-Lead SOT-223



00882242

Front View
Order Part Number LM2940IMP-5.0, LM2940IMP-8.0, LM2940IMP-9.0, LM2940IMP-10, LM2940IMP-12 or LM2940IMP-15
See NS Package Number MP04A

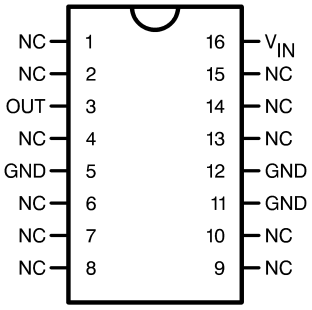
16-Lead Dual-in-Line Package (J)



00882243

Top View
Order Number LM2940J-5.0/883 (5962-8958701EA), LM2940J-8.0/883 (5962-9088301QEA), LM2940J-12/883 (5962-9088401QEA), LM2940J-15/883 (5962-9088501QEA)
See NS Package Number J16A

16-Lead Ceramic Surface-Mount Package (WG)

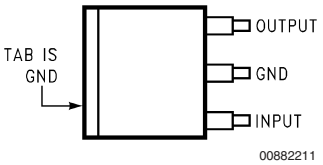


00882244

Top View
Order Number LM2940WG5.0/883 (5962-8958701XA)
See NS Package Number WG16A

Connection Diagrams (Continued)

(TO-263) Surface-Mount Package



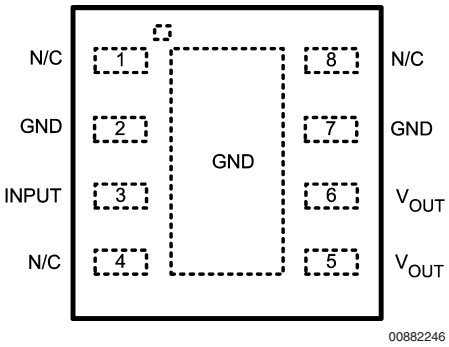
Top View

Side View

Order Number LM2940CS-5.0, LM2940CS-9.0, LM2940CS-12, LM2940CS-15, LM2940S-5.0, LM2940S-8.0, LM2940S-9.0, LM2940S-10 or LM2940S-12

See NS Package Number TS3B

8-Lead LLP



Top View

Order Number LM2940LD-5.0, LM2940LD-8.0, LM2940LD-9.0, LM2940LD-10, LM2940LD-12, LM2940LD-15

See NS Package Number LDC08A

Pin 2 and pin 7 are fused to center DAP

Pin 5 ans 6 need to be tied together on PCB board

Absolute Maximum Ratings (Note 1)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

LM2940S, T, MP ≤ 100 ms	60V
LM2940CS, T ≤ 1 ms	45V
Internal Power Dissipation (Note 2)	Internally Limited
Maximum Junction Temperature	150˚C
Storage Temperature Range	−65˚C ≤ T_J ≤ +150˚C
Lead Temperature, Time for Wave Soldering	
TO-220 (T) Package	260˚C, 10s
TO-263 (S) Package	260˚C, 4s
SOT-223 (MP) Package	260˚C, 4s
ESD Susceptibility (Note 3)	2 kV

Operating Conditions (Note 1)

Input Voltage	26V
Temperature Range	
LM2940T, LM2940S	−40˚C ≤ T_J ≤ 125˚C
LM2940CT, LM2940CS	0˚C ≤ T_J ≤ 125˚C
LM2940IMP	−40˚C ≤ T_J ≤ 85˚C
LM2940J, LM2940WG	−55˚C ≤ T_J ≤ 125˚C
LM2940LD	−40˚C ≤ T_J ≤ 125˚C

Electrical Characteristics

V_{IN} = V_O + 5V, I_O = 1A, C_O = 22 µF, unless otherwise specified. **Boldface limits apply over the entire operating temperature range of the indicated device.** All other specifications apply for T_A = T_J = 25˚C.

Output Voltage (V_O)		5V			8V			Units
Parameter	**Conditions**	**Typ**	**LM2940 Limit** (Note 4)	**LM2940/883 Limit** (Note 5)	**Typ**	**LM2940 Limit** (Note 4)	**LM2940/883 Limit** (Note 5)	
		6.25V ≤ V_{IN} ≤ 26V			**9.4V ≤ V_{IN} ≤ 26V**			
Output Voltage	5 mA ≤ I_O ≤ 1A	5.00	4.85/**4.75** 5.15/**5.25**	4.85/**4.75** 5.15/**5.25**	8.00	7.76/**7.60** 8.24/**8.40**	7.76/**7.60** 8.24/**8.40**	V_{MIN} V_{MAX}
Line Regulation	V_O + 2V ≤ V_{IN} ≤ 26V, I_O = 5 mA	20	50	40/**50**	20	80	50/**80**	mV_{MAX}
Load Regulation	50 mA ≤ I_O ≤ 1A LM2940, LM2940/883 LM2940C	 35 35	 50/**80** 50	 50/**100**	 55 55	 80/**130** 80	 80/**130**	mV_{MAX}
Output Impedance	100 mADC and 20 mArms, f_O = 120 Hz	35		1000/**1000**	55		1000/**1000**	mΩ
Quiescent Current	V_O +2V ≤ V_{IN} ≤ 26V, I_O = 5 mA LM2940, LM2940/883 LM2940C	 10 10	 15/**20** 15	 15/**20**	 10	 15/**20**	 15/**20**	mA_{MAX}
	V_{IN} = V_O + 5V, I_O = 1A	30	45/**60**	50/**60**	30	45/**60**	50/**60**	mA_{MAX}
Output Noise Voltage	10 Hz – 100 kHz, I_O = 5 mA	150		700/**700**	240		1000/**1000**	$µV_{rms}$
Ripple Rejection	f_O = 120 Hz, 1 V_{rms}, I_O = 100 mA LM2940 LM2940C	 72 72	 60/**54** 60		 66 66	 54/**48** 54		dB_{MIN}
	f_O = 1 kHz, 1 V_{rms}, I_O = 5 mA			60/**50**			54/**48**	dB_{MIN}
Long Term Stability		20			32			mV/ 1000 Hr
Dropout Voltage	I_O = 1A	0.5	0.8/**1.0**	0.7/**1.0**	0.5	0.8/**1.0**	0.7/**1.0**	V_{MAX}
	I_O = 100 mA	110	150/**200**	150/**200**	110	150/**200**	150/**200**	mV_{MAX}
Short Circuit Current	(Note 6)	1.9	1.6	1.5/**1.3**	1.9	1.6	1.6/**1.3**	A_{MIN}

Electrical Characteristics (Continued)

$V_{IN} = V_O + 5V$, $I_O = 1A$, $C_O = 22\ \mu F$, unless otherwise specified. **Boldface limits apply over the entire operating temperature range of the indicated device.** All other specifications apply for $T_A = T_J = 25^\circ C$.

Output Voltage (V_O)		5V			8V			Units
Parameter	**Conditions**	**Typ**	**LM2940 Limit** (Note 4)	**LM2940/883 Limit** (Note 5)	**Typ**	**LM2940 Limit** (Note 4)	**LM2940/883 Limit** (Note 5)	
Maximum Line Transient	$R_O = 100\Omega$							
	LM2940, T ≤ 100 ms	75	60/**60**		75	60/**60**		V_{MIN}
	LM2940/883, T ≤ 20 ms			40/**40**			40/**40**	
	LM2940C, T ≤ 1 ms	55	45		55	45		
Reverse Polarity DC Input Voltage	$R_O = 100\Omega$							
	LM2940, LM2940/883	–30	–15/**–15**	–15/**–15**	–30	–15/**–15**	–15/**–15**	V_{MIN}
	LM2940C	–30	–15		–30	–15		
Reverse Polarity Transient Input Voltage	$R_O = 100\Omega$							
	LM2940, T ≤ 100 ms	–75	–50/**–50**		–75	–50/**–50**		V_{MIN}
	LM2940/883, T ≤ 20 ms			–45/**–45**			–45/**–45**	
	LM2940C, T ≤ 1 ms	–55	–45/**–45**					

Electrical Characteristics

$V_{IN} = V_O + 5V$, $I_O = 1A$, $C_O = 22\ \mu F$, unless otherwise specified. **Boldface limits apply over the entire operating temperature range of the indicated device.** All other specifications apply for $T_A = T_J = 25^\circ C$.

Output Voltage (V_O)		9V		10V		Units
Parameter	**Conditions**	**Typ**	**LM2940 Limit** (Note 4)	**Typ**	**LM2940 Limit** (Note 4)	
		10.5V ≤ V_{IN} ≤ 26V		**11.5V ≤ V_{IN} ≤ 26V**		
Output Voltage	5 mA ≤ I_O ≤1A	9.00	8.73/**8.55**	10.00	9.70/**9.50**	V_{MIN}
			9.27/**9.45**		10.30/**10.50**	V_{MAX}
Line Regulation	$V_O + 2V \le V_{IN} \le 26V$, $I_O = 5$ mA	20	90	20	100	mV_{MAX}
Load Regulation	50 mA ≤ I_O ≤ 1A					
	LM2940	60	90/**150**	65	100/**165**	mV_{MAX}
	LM2940C	60	90			
Output Impedance	100 mADC and 20 mArms, $f_O = 120$ Hz	60		65		mΩ
Quiescent Current	$V_O + 2V \le V_{IN} < 26V$, $I_O = 5$ mA					
	LM2940	10	15/**20**	10	15/**20**	mA_{MAX}
	LM2940C	10	15			
	$V_{IN} = V_O + 5V$, $I_O = 1A$	30	45/**60**	30	45/**60**	mA_{MAX}
Output Noise Voltage	10 Hz – 100 kHz, $I_O = 5$ mA	270		300		μV_{rms}
Ripple Rejection	$f_O = 120$ Hz, 1 V_{rms}, $I_O = 100$ mA					
	LM2940	64	52/**46**	63	51/**45**	dB_{MIN}
	LM2940C	64	52			
Long Term Stability		34		36		mV/1000 Hr
Dropout Voltage	$I_O = 1A$	0.5	0.8/**1.0**	0.5	0.8/**1.0**	V_{MAX}
	$I_O = 100$ mA	110	150/**200**	110	150/**200**	mV_{MAX}

Electrical Characteristics (Continued)

$V_{IN} = V_O + 5V$, $I_O = 1A$, $C_O = 22\ \mu F$, unless otherwise specified. **Boldface limits apply over the entire operating temperature range of the indicated device.** All other specifications apply for $T_A = T_J = 25^\circ C$.

Output Voltage (V_O)		9V		10V		Units
Parameter	**Conditions**	**Typ**	**LM2940 Limit** (Note 4)	**Typ**	**LM2940 Limit** (Note 4)	
Short Circuit Current	(Note 6)	1.9	1.6	1.9	1.6	A_{MIN}
Maximum Line Transient	$R_O = 100\Omega$ $T \leq 100$ ms LM2940 LM2940C	 75 55	 60/**60** 45	75	60/**60**	V_{MIN}
Reverse Polarity DC Input Voltage	$R_O = 100\Omega$ LM2940 LM2940C	 –30 –30	 –15/**–15** –15	–30	–15/**–15**	V_{MIN}
Reverse Polarity Transient Input Voltage	$R_O = 100\Omega$ $T \leq 100$ ms LM2940 LM2940C	 –75 –55	 –50/**–50** –45/**–45**	–75	–50/**–50**	V_{MIN}

Electrical Characteristics

$V_{IN} = V_O + 5V$, $I_O = 1A$, $C_O = 22\ \mu F$, unless otherwise specified. **Boldface limits apply over the entire operating temperature range of the indicated device.** All other specifications apply for $T_A = T_J = 25^\circ C$.

Output Voltage (V_O)		12V			15V			Units
Parameter	**Conditions**	**Typ**	**LM2940 Limit** (Note 4)	**LM2940/833 Limit** (Note 5)	**Typ**	**LM2940 Limit** (Note 4)	**LM2940/833 Limit** (Note 5)	
		$13.6V \leq V_{IN} \leq 26V$			**$16.75V \leq V_{IN} \leq 26V$**			
Output Voltage	$5\ mA \leq I_O \leq 1A$	12.00	11.64/**11.40** 12.36/**12.60**	11.64/**11.40** 12.36/**12.60**	15.00	14.55/**14.25** 15.45/**15.75**	14.55/**14.25** 15.45/**15.75**	V_{MIN} V_{MAX}
Line Regulation	$V_O + 2V \leq V_{IN} \leq 26V$, $I_O = 5\ mA$	20	120	75/**120**	20	150	95/**150**	mV_{MAX}
Load Regulation	$50\ mA \leq I_O \leq 1A$ LM2940, LM2940/883 LM2940C	 55 55	 120/**200** 120	 120/**190**	 70	 150	 150/**240**	mV_{MAX}
Output Impedance	100 mADC and 20 mArms, $f_O = 120$ Hz	80		1000/**1000**	100		1000/**1000**	$m\Omega$
Quiescent Current	$V_O + 2V \leq V_{IN} \leq 26V$, $I_O = 5\ mA$ LM2940, LM2940/883 LM2940C	 10 10	 15/**20** 15	 15/**20**	 10	 15	 15/**20**	mA_{MAX}
	$V_{IN} = V_O + 5V$, $I_O = 1A$	30	45/**60**	50/**60**	30	45/**60**	50/**60**	mA_{MAX}
Output Noise Voltage	10 Hz – 100 kHz, $I_O = 5\ mA$	360		1000/**1000**	450		1000/**1000**	μV_{rms}
Ripple Rejection	$f_O = 120$ Hz, 1 V_{rms}, $I_O = 100\ mA$ LM2940 LM2940C	 66 66	 54/**48** 54		 64	 52		dB_{MIN}
	$f_O = 1$ kHz, 1 V_{rms}, $I_O = 5\ mA$			52/**46**			48/**42**	dB_{MIN}

Electrical Characteristics (Continued)

$V_{IN} = V_O + 5V$, $I_O = 1A$, $C_O = 22\ \mu F$, unless otherwise specified. **Boldface limits apply over the entire operating temperature range of the indicated device.** All other specifications apply for $T_A = T_J = 25^\circ C$.

Output Voltage (V_O)		12V			15V			Units
Parameter	**Conditions**	**Typ**	**LM2940 Limit** (Note 4)	**LM2940/833 Limit** (Note 5)	**Typ**	**LM2940 Limit** (Note 4)	**LM2940/833 Limit** (Note 5)	
Long Term Stability		48			60			mV/ 1000 Hr
Dropout Voltage	I_O = 1A	0.5	0.8/**1.0**	0.7/**1.0**	0.5	0.8/**1.0**	0.7/**1.0**	V_{MAX}
	I_O = 100 mA	110	150/**200**	150/**200**	110	150/**200**	150/**200**	mV_{MAX}
Short Circuit Current	(Note 6)	1.9	1.6	1.6/**1.3**	1.9	1.6	1.6/**1.3**	A_{MIN}
Maximum Line Transient	R_O = 100Ω							
	LM2940, T ≤ 100 ms	75	60/**60**					
	LM2940/883, T ≤ 20 ms			40/**40**			40/**40**	V_{MIN}
	LM2940C, T ≤ 1 ms	55	45		55	45		
Reverse Polarity DC Input Voltage	R_O = 100Ω							
	LM2940, LM2940/883	–30	–15/**–15**	–15/**–15**			–15/**–15**	V_{MIN}
	LM2940C	–30	–15		–30	–15		
Reverse Polarity Transient Input Voltage	R_O = 100Ω							
	LM2940, T ≤ 100 ms	–75	–50/**–50**					
	LM2940/883, T ≤ 20 ms			–45/**–45**			–45/**–45**	V_{MIN}
	LM2940C, T ≤ 1 ms	–55	–45/**–45**		–55	–45/**–45**		

Thermal Performance

Thermal Resistance Junction-to-Case	3-Lead TO-220	4		˚C/W
	3-Lead TO-263	4		˚C/W
Thermal Resistance Junction-to-Ambient	3-Lead TO-220	60		˚C/W
	3-Lead TO-263	80		˚C/W
	8-Lead LLP (Note 2)	35		˚C/W

Note 1: Absolute Maximum Ratings are limits beyond which damage to the device may occur. Operating Conditions are conditions under which the device functions but the specifications might not be guaranteed. For guaranteed specifications and test conditions see the Electrical Characteristics.

Note 2: The maximum allowable power dissipation is a function of the maximum junction temperature, T_J, the junction-to-ambient thermal resistance, θ_{JA}, and the ambient temperature, T_A. Exceeding the maximum allowable power dissipation will cause excessive die temperature, and the regulator will go into thermal shutdown. The value of θ_{JA} (for devices in still air with no heatsink) is 60˚C/W for the TO-220 package, 80˚C/W for the TO-263 package, and 174˚C/W for the SOT-223 package. The effective value of θ_{JA} can be reduced by using a heatsink (see Application Hints for specific information on heatsinking). The value of θ_{JA} for the LLP package is specifically dependent on PCB trace area, trace material, and the number of layers and thermal vias. For improved thermal resistance and power dissipation for the LLP package, refer to Application Note AN-1187. It is recommended that 6 vias be placed under the center pad to improve thermal performance.

Note 3: ESD rating is based on the human body model, 100 pF discharged through 1.5 kΩ.

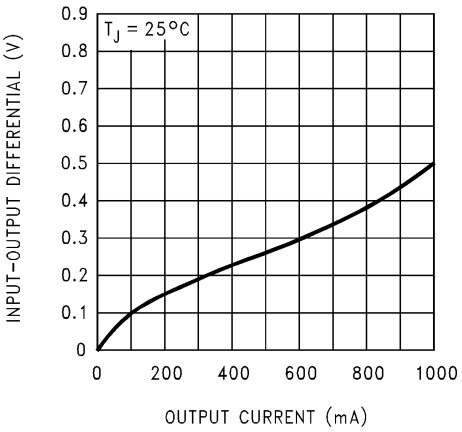
Note 4: All limits are guaranteed at $T_A = T_J = 25^\circ C$ only (standard typeface) or over the entire operating temperature range of the indicated device (boldface type). All limits at $T_A = T_J = 25^\circ C$ are 100% production tested. All limits at temperature extremes are guaranteed via correlation using standard Statistical Quality Control methods.

Note 5: All limits are guaranteed at $T_A = T_J = 25^\circ C$ only (standard typeface) or over the entire operating temperature range of the indicated device (boldface type). All limits are 100% production tested and are used to calculate Outgoing Quality Levels.

Note 6: Output current will decrease with increasing temperature but will not drop below 1A at the maximum specified temperature.

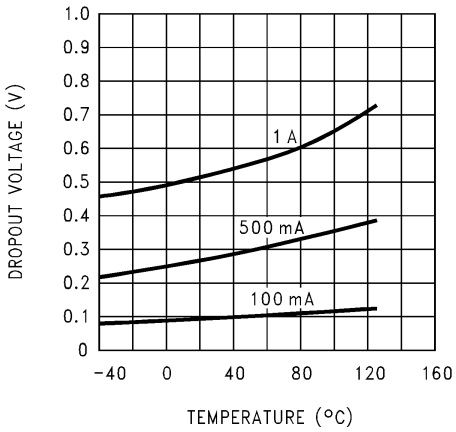
Typical Performance Characteristics

Dropout Voltage



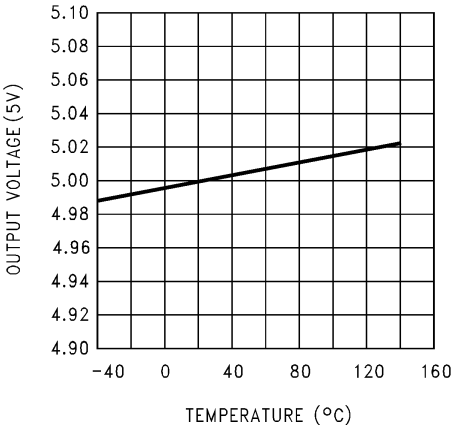
00882213

Dropout Voltage vs. Temperature



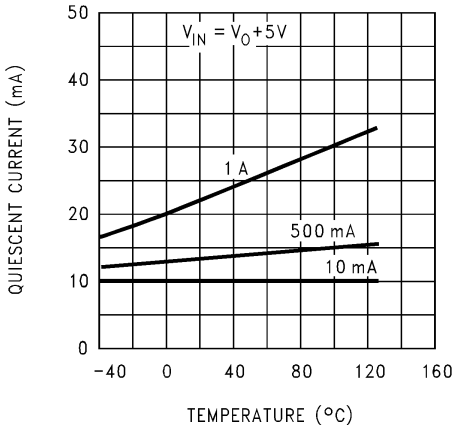
00882214

Output Voltage vs. Temperature



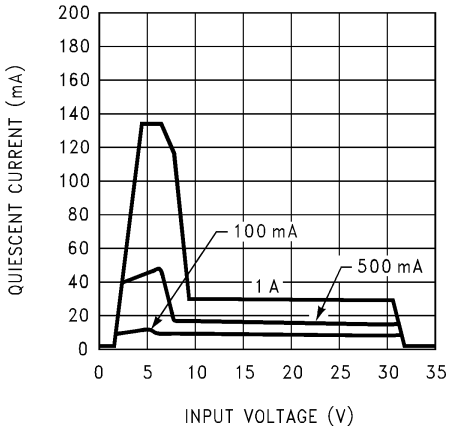
00882215

Quiescent Current vs. Temperature



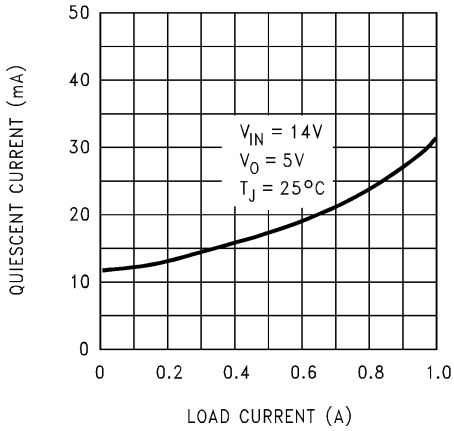
00882216

Quiescent Current



00882217

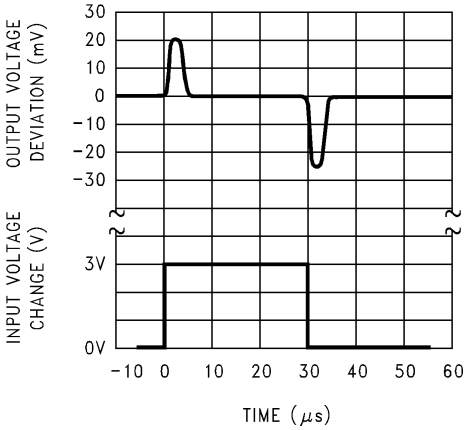
Quiescent Current



00882218

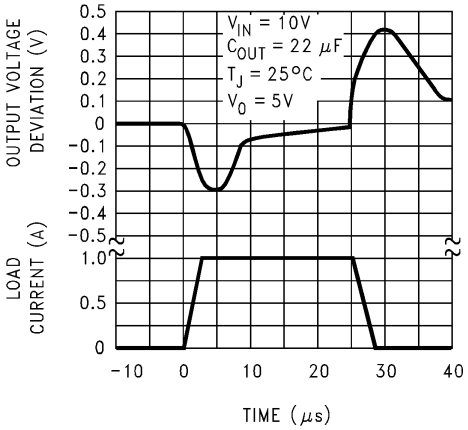
Typical Performance Characteristics (Continued)

Line Transient Response



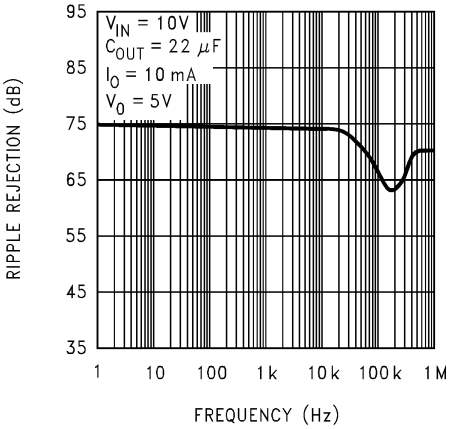
00882219

Load Transient Response



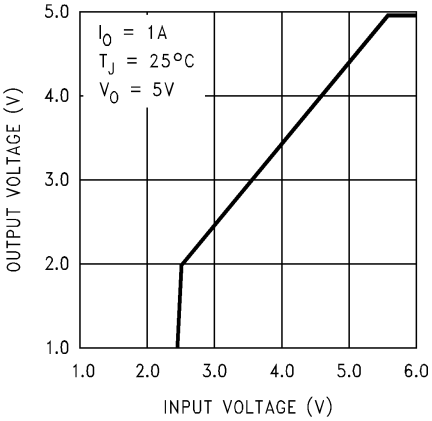
00882220

Ripple Rejection



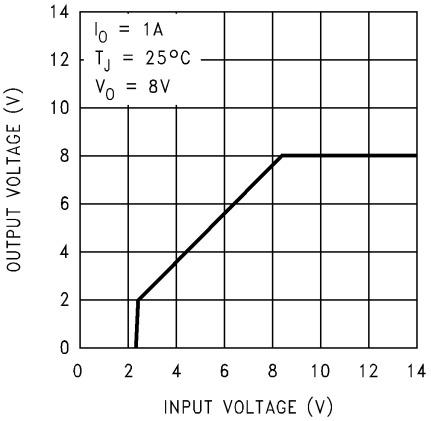
00882221

Low Voltage Behavior



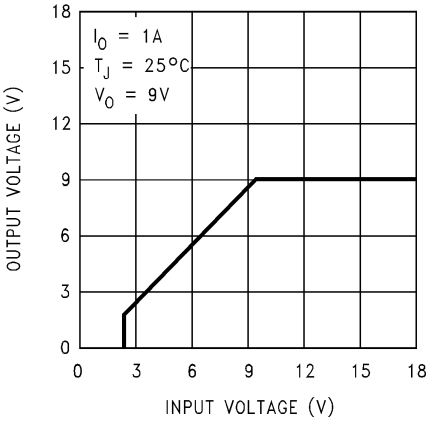
00882225

Low Voltage Behavior



00882226

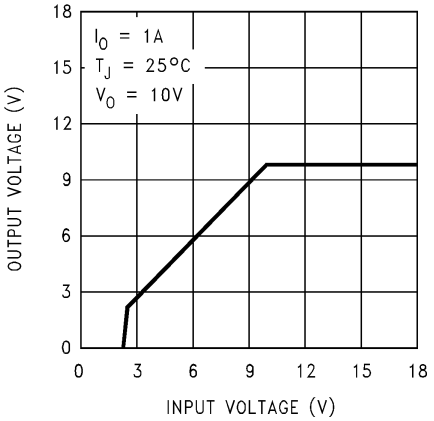
Low Voltage Behavior



00882227

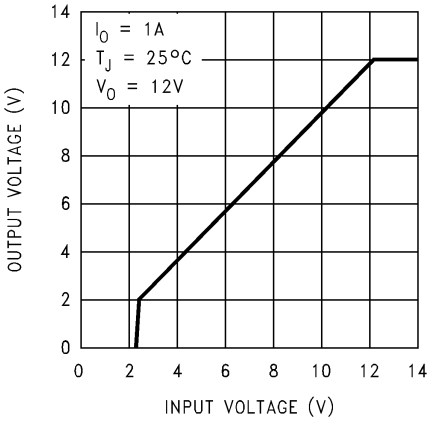
Typical Performance Characteristics (Continued)

Low Voltage Behavior



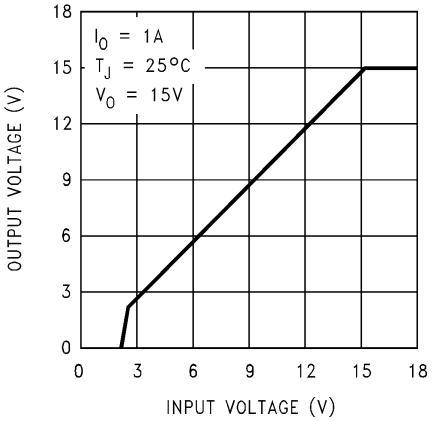
00882228

Low Voltage Behavior



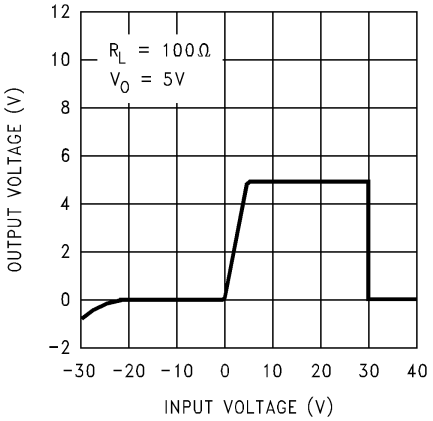
00882229

Low Voltage Behavior



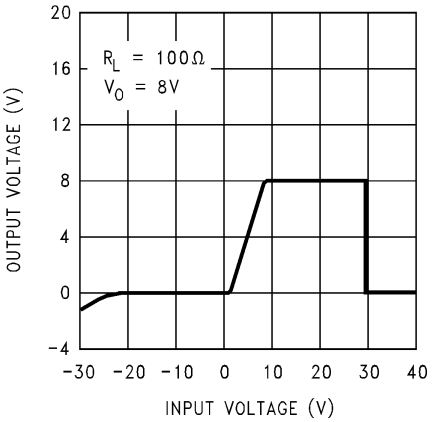
00882230

Output at Voltage Extremes



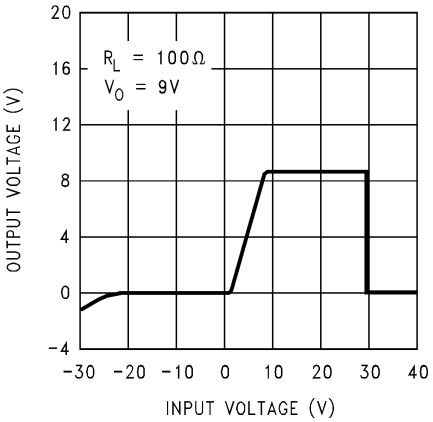
00882231

Output at Voltage Extremes



00882232

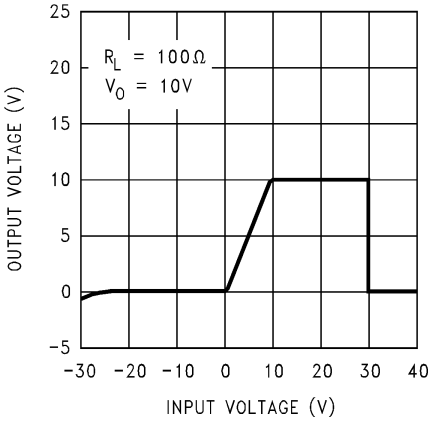
Output at Voltage Extremes



00882233

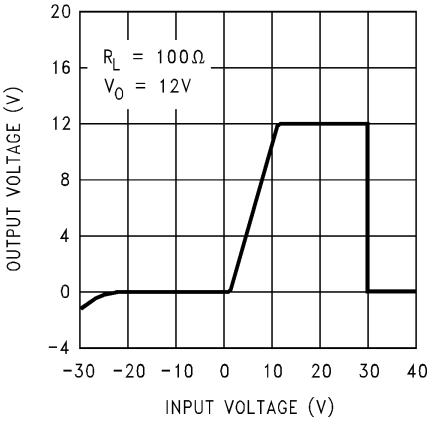
Typical Performance Characteristics (Continued)

Output at Voltage Extremes



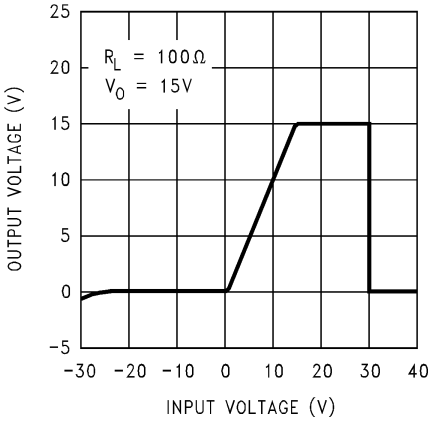
00882234

Output at Voltage Extremes



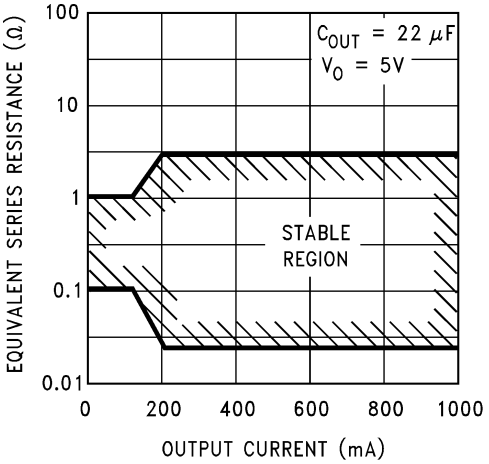
00882235

Output at Voltage Extremes



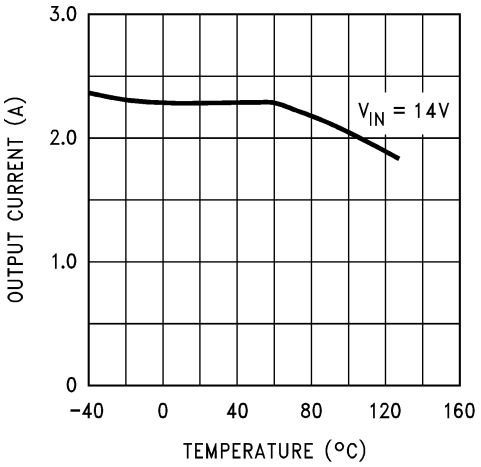
00882236

Output Capacitor ESR



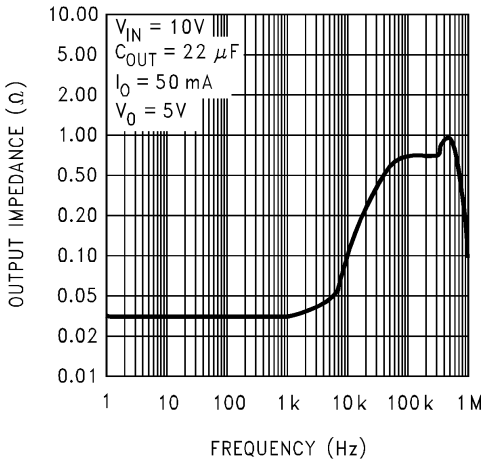
00882206

Peak Output Current



00882208

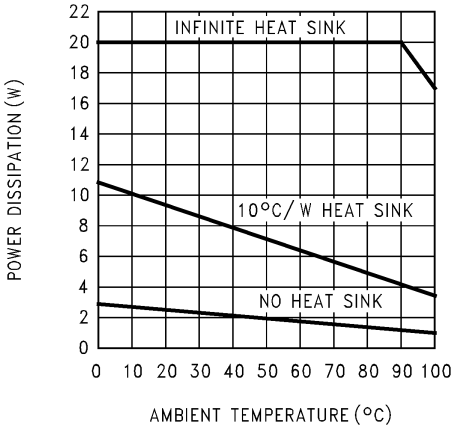
Output Impedance



00882222

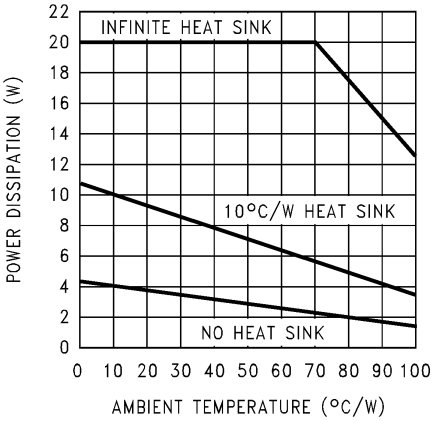
Typical Performance Characteristics (Continued)

Maximum Power Dissipation (TO-220)



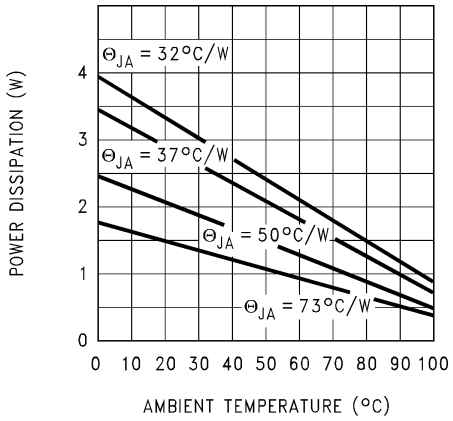
00882223

Maximum Power Dissipation (TO-3)



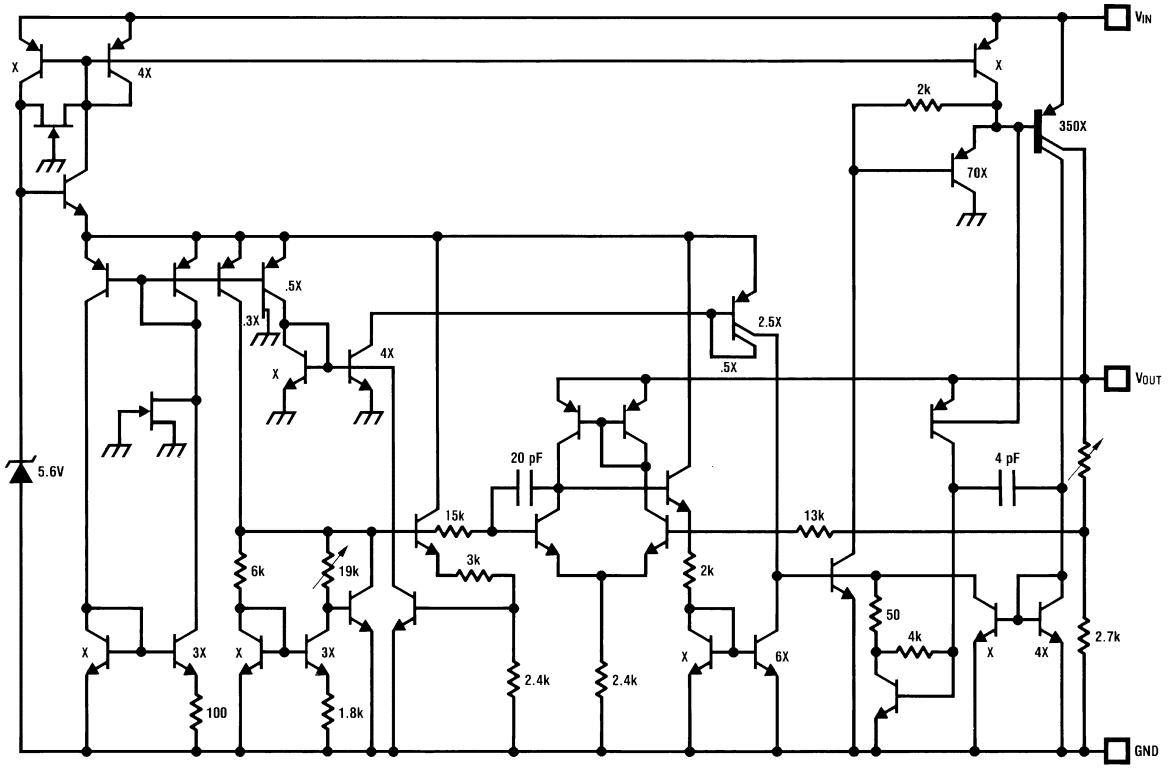
00882224

Maximum Power Dissipation (TO-263)
See (Note 2)



00882210

Equivalent Schematic Diagram



00882201

Application Hints

EXTERNAL CAPACITORS

The output capacitor is critical to maintaining regulator stability, and must meet the required conditions for both ESR (Equivalent Series Resistance) and minimum amount of capacitance.

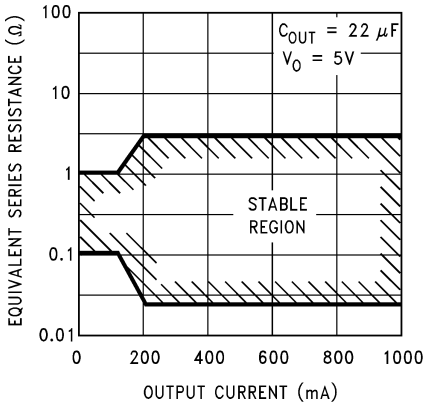
MINIMUM CAPACITANCE:

The minimum output capacitance required to maintain stability is 22 µF (this value may be increased without limit). Larger values of output capacitance will give improved transient response.

ESR LIMITS:

The ESR of the output capacitor will cause loop instability if it is too high or too low. The acceptable range of ESR plotted versus load current is shown in the graph below. ***It is essential that the output capacitor meet these requirements, or oscillations can result.***

Output Capacitor ESR



00882206

FIGURE 1. ESR Limits

It is important to note that for most capacitors, ESR is specified only at room temperature. However, the designer must ensure that the ESR will stay inside the limits shown over the entire operating temperature range for the design.

For aluminum electrolytic capacitors, ESR will increase by about 30X as the temperature is reduced from 25˚C to –40˚C. This type of capacitor is not well-suited for low temperature operation.

Solid tantalum capacitors have a more stable ESR over temperature, but are more expensive than aluminum electrolytics. A cost-effective approach sometimes used is to

Application Hints (Continued)

parallel an aluminum electrolytic with a solid Tantalum, with the total capacitance split about 75/25% with the Aluminum being the larger value.

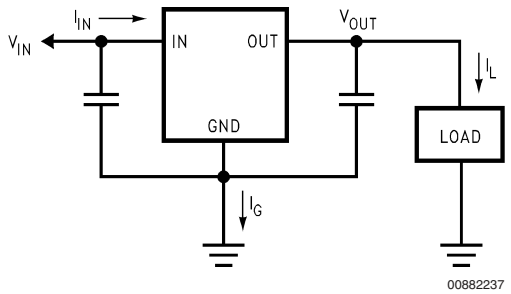
If two capacitors are paralleled, the effective ESR is the parallel of the two individual values. The "flatter" ESR of the Tantalum will keep the effective ESR from rising as quickly at low temperatures.

HEATSINKING

A heatsink may be required depending on the maximum power dissipation and maximum ambient temperature of the application. Under all possible operating conditions, the junction temperature must be within the range specified under Absolute Maximum Ratings.

To determine if a heatsink is required, the power dissipated by the regulator, P_D, must be calculated.

The figure below shows the voltages and currents which are present in the circuit, as well as the formula for calculating the power dissipated in the regulator:



$I_{IN} = I_L \div I_G$

$P_D = (V_{IN} - V_{OUT})\, I_L + (V_{IN})\, I_G$

FIGURE 2. Power Dissipation Diagram

The next parameter which must be calculated is the maximum allowable temperature rise, T_R (max). This is calculated by using the formula:

$$T_R\,(max) = T_J(max) - T_A\,(max)$$

where: T_J (max) is the maximum allowable junction temperature, which is 125˚C for commercial grade parts.

T_A (max) is the maximum ambient temperature which will be encountered in the application.

Using the calculated values for T_R(max) and P_D, the maximum allowable value for the junction-to-ambient thermal resistance, $\theta_{(JA)}$, can now be found:

$$\theta_{(JA)} = T_R\,(max)/P_D$$

IMPORTANT: If the maximum allowable value for $\theta_{(JA)}$ is found to be ≥ 53˚C/W for the TO-220 package, ≥ 80˚C/W for the TO-263 package, or ≥174˚C/W for the SOT-223 package, no heatsink is needed since the package alone will dissipate enough heat to satisfy these requirements.

If the calculated value for $\theta_{(JA)}$falls below these limits, a heatsink is required.

HEATSINKING TO-220 PACKAGE PARTS

The TO-220 can be attached to a typical heatsink, or secured to a copper plane on a PC board. If a copper plane is to be used, the values of $\theta_{(JA)}$ will be the same as shown in the next section for the TO-263.

If a manufactured heatsink is to be selected, the value of heatsink-to-ambient thermal resistance, $\theta_{(H-A)}$, must first be calculated:

$$\theta_{(H-A)} = \theta_{(JA)} - \theta_{(C-H)} - \theta_{(J-C)}$$

Where: $\theta_{(J-C)}$ is defined as the thermal resistance from the junction to the surface of the case. A value of 3˚C/W can be assumed for $\theta_{(J-C)}$ for this calculation.

$\theta_{(C-H)}$ is defined as the thermal resistance between the case and the surface of the heatsink. The value of $\theta_{(C-H)}$ will vary from about 1.5˚C/W to about 2.5˚C/W (depending on method of attachment, insulator, etc.). If the exact value is unknown, 2˚C/W should be assumed for $\theta_{(C-H)}$.

When a value for $\theta_{(H-A)}$ is found using the equation shown, *a heatsink must be selected that has a value that is less than or equal to this number.*

$\theta_{(H-A)}$ is specified numerically by the heatsink manufacturer in the catalog, or shown in a curve that plots temperature rise vs power dissipation for the heatsink.

HEATSINKING TO-263 AND SOT-223 PACKAGE PARTS

Both the TO-263 ("S") and SOT-223 ("MP") packages use a copper plane on the PCB and the PCB itself as a heatsink. To optimize the heat sinking ability of the plane and PCB, solder the tab of the package to the plane.

Figure 3 shows for the TO-263 the measured values of $\theta_{(JA)}$ for different copper area sizes using a typical PCB with 1 ounce copper *and no solder mask over the copper area used for heatsinking.*

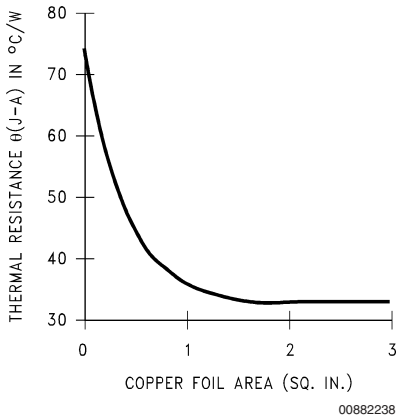


FIGURE 3. $\theta_{(JA)}$ vs. Copper (1 ounce) Area for the TO-263 Package

As shown in the figure, increasing the copper area beyond 1 square inch produces very little improvement. It should also be observed that the minimum value of $\theta_{(JA)}$ for the TO-263 package mounted to a PCB is 32˚C/W.

As a design aid, *Figure 4* shows the maximum allowable power dissipation compared to ambient temperature for the TO-263 device (assuming $\theta_{(JA)}$ is 35˚C/W and the maximum junction temperature is 125˚C).

Application Hints (Continued)

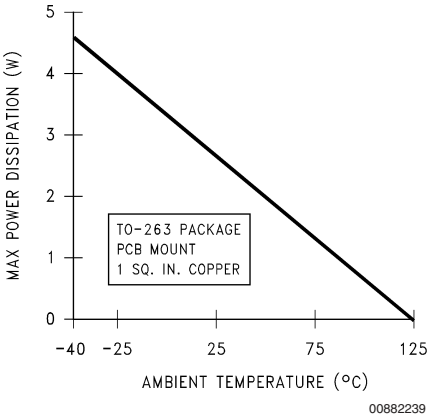


FIGURE 4. Maximum Power Dissipation vs. T_{AMB} for the TO-263 Package

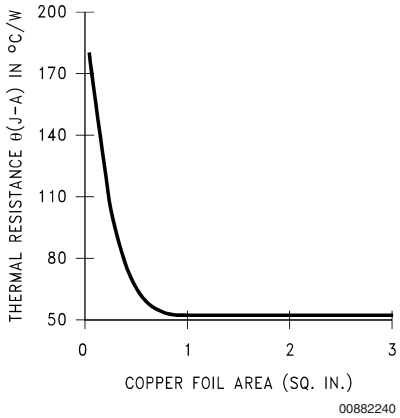


FIGURE 5. $\theta_{(JA)}$ vs. Copper (2 ounce) Area for the SOT-223 Package

Figure 5 and *Figure 6* show the information for the SOT-223 package. *Figure 6* assumes a $\theta_{(JA)}$ of 74˚C/W for 1 ounce copper and 51˚C/W for 2 ounce copper and a maximum junction temperature of 125˚C.

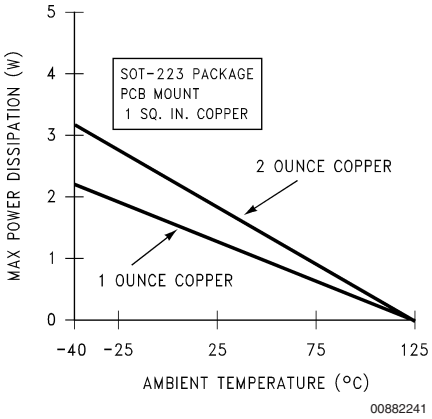
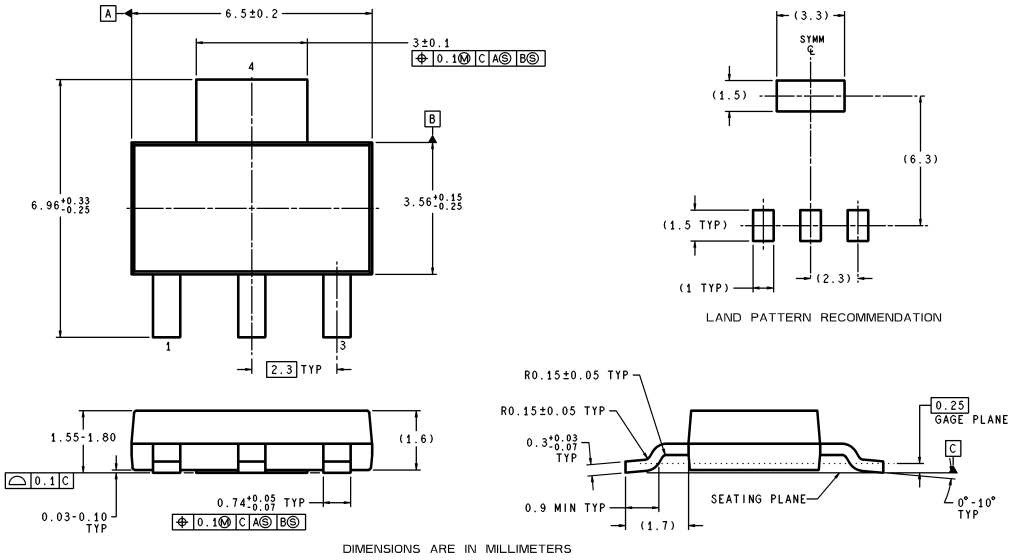
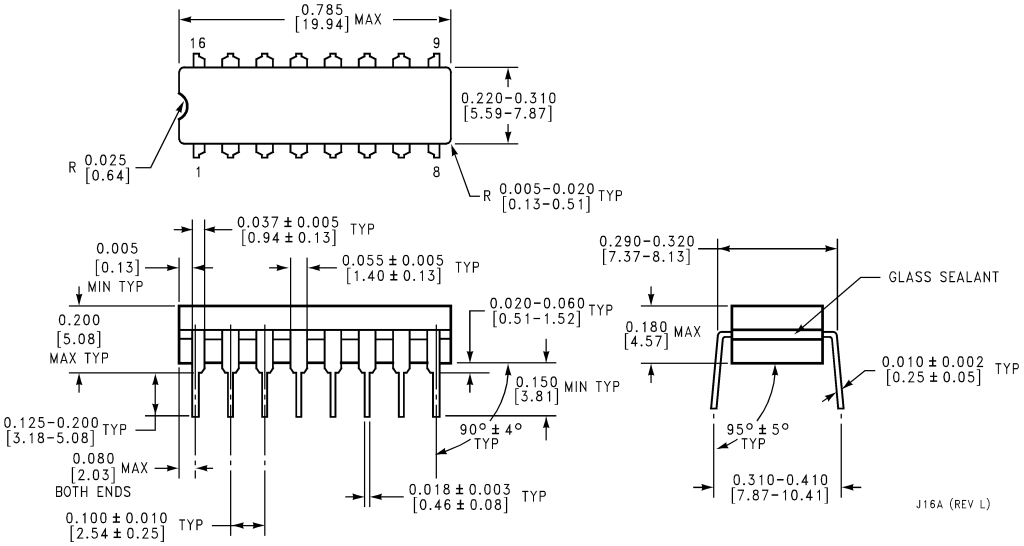


FIGURE 6. Maximum Power Dissipation vs. T_{AMB} for the SOT-223 Package

Physical Dimensions inches (millimeters) unless otherwise noted

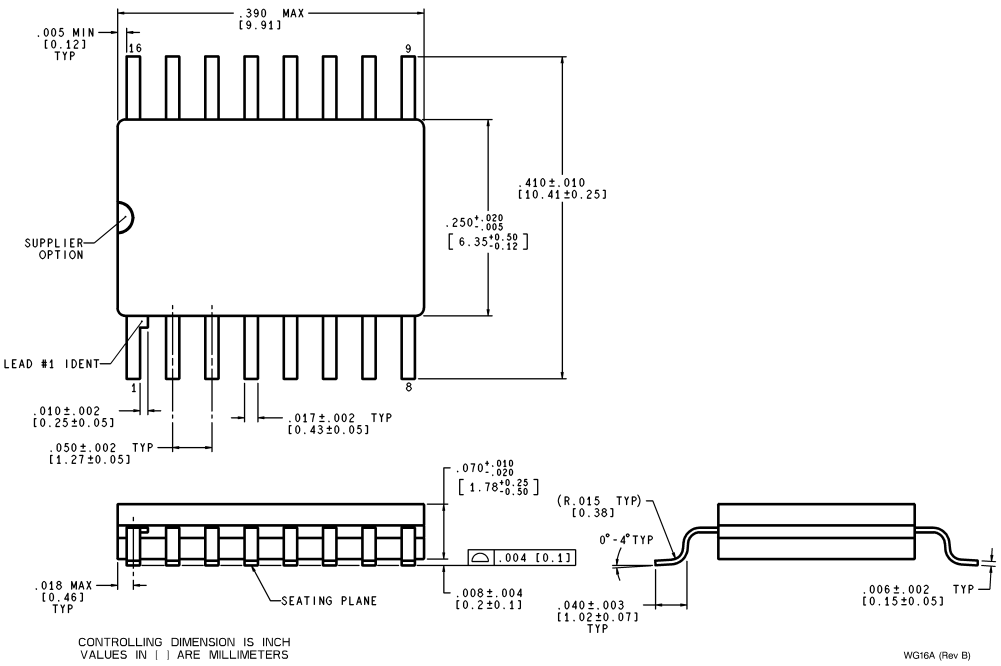


3-Lead SOT-223 Package
Order Part Number LM2940IMP-5.0
LM2940IMP-8.0 LM2940IMP-9.0
LM2940IMP-10 LM2940IMP-12 LM2940IMP-15
NS Package Number MP04A

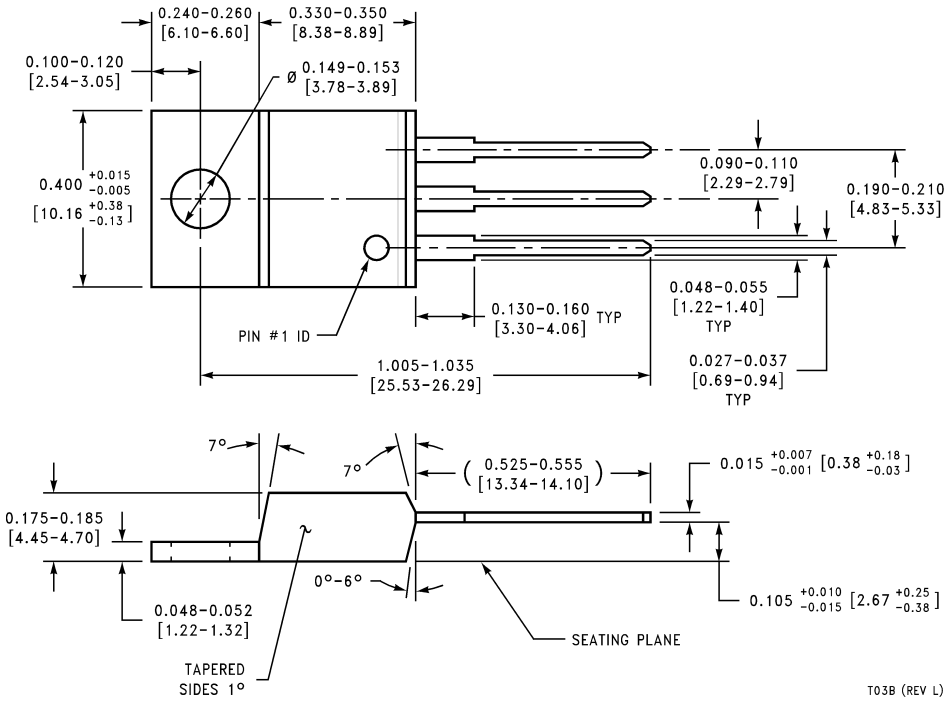


16 Lead Dual-in-Line Package (J)
Order Number LM2940J-5.0/883 (5962-8958701EA),
LM2940J-8.0/883 (5962-9088301QEA),
LM2940J-12/883 (5962-9088401QEA),
LM2940J-15/883 (5962-9088501QEA)
See NS Package Number J16A

Physical Dimensions inches (millimeters) unless otherwise noted (Continued)

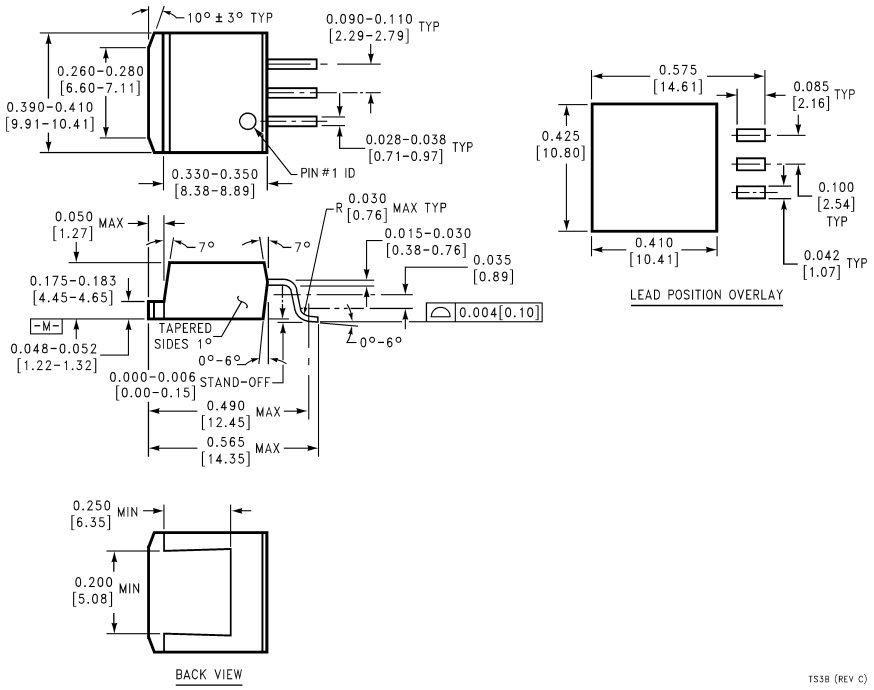


16 Lead Surface Mount Package (WG)
Order Number LM2940WG5.0/883 (5962-8958701XA)
See NS Package Number WG16A

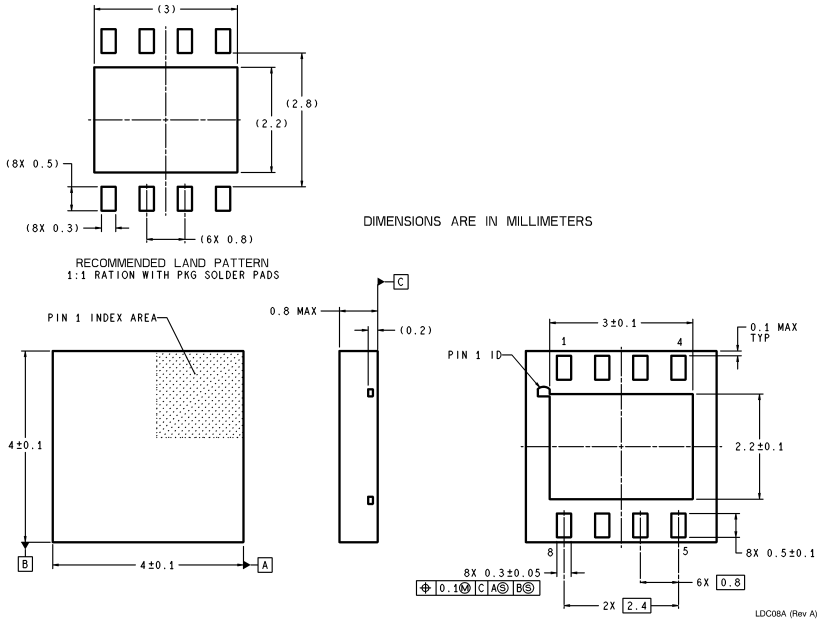


3-Lead TO-220 Plastic Package (T)
Order Number LM2940T-5.0, LM2940T-8.0,
LM2940T-9.0, LM2940T-10, LM2940T-12, LM2940CT-5.0,
LM2940CT-12 or LM2940CT-15
NS Package Number TO3B

Physical Dimensions inches (millimeters) unless otherwise noted (Continued)



3-Lead TO-263 Surface Mount Package (MP)
Order Number LM2940S-5.0, LM2940S-8.0,
LM2940S-9.0, LM2940S-10, LM2940S-12,
LM2940CS-5.0, LM2940CS-12 or LM2940CS-15
NS Package Number TS3B



8-Lead LLP
Order Number LM2940LD-5.0, LM2940LD-8.0,
LM2940LD-9.0, LM2940LD-10,
LM2940LD-12 or LM2940LD-15
NS Package Number LDC08A

Notes

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor
Americas Customer
Support Center
Email: new.feedback@nsc.com
Tel: 1-800-272-9959

www.national.com

National Semiconductor
Europe Customer Support Center
Fax: +49 (0) 180-530 85 86
Email: europe.support@nsc.com
Deutsch Tel: +49 (0) 69 9508 6208
English Tel: +44 (0) 870 24 0 2171
Français Tel: +33 (0) 1 41 91 8790

National Semiconductor
Asia Pacific Customer
Support Center
Fax: 65-6250 4466
Email: ap.support@nsc.com
Tel: 65-6254 4466

National Semiconductor
Japan Customer Support Center
Fax: 81-3-5639-7507
Email: nsj.crc@jksmtp.nsc.com
Tel: 81-3-5639-7560

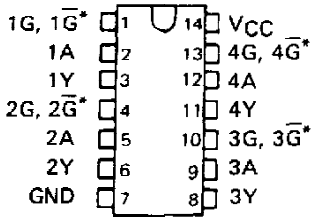
SDLS044

SN54125, SN54126, SN54LS125A, SN54LS126A, SN74125, SN74126, SN74LS125A, SN74LS126A
QUADRUPLE BUS BUFFERS WITH 3-STATE OUTPUTS

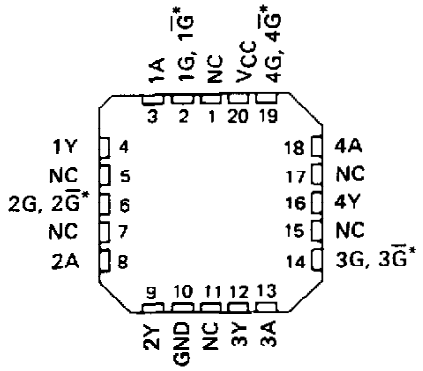
DECEMBER 1983 — REVISED MARCH 1988

- **Quad Bus Buffers**
- **3-State Outputs**
- **Separate Control for Each Channel**

SN54125, SN54126, SN54LS125A, SN54LS126A . . . J OR W PACKAGE
SN74125, SN74126 . . . N PACKAGE
SN74LS125A, SN74LS126A . . . D OR N PACKAGE
(TOP VIEW)



SN54LS125A, SN54LS126A . . . FK PACKAGE
(TOP VIEW)



*G̅ on '125 and 'LS125A; G on 126 and 'LS126A

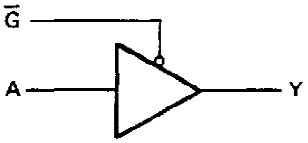
NC — No internal connection

description

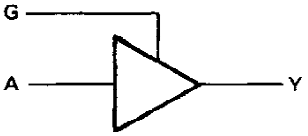
These bus buffers feature three-state outputs that, when enabled, have the low impedence characteristics of a TTL output with additional drive capability at high logic levels to permit driving heavily loaded bus lines without external pull-up resistors, when disabled, both output transistors are turned off presenting a high-impedance state to the bus so the output will act neither as a significant load nor as a driver. The '125 and 'LS125A outputs are disabled when G̅ is high. The '126 and 'LS126A outputs are disabled when G is low.

logic diagram (each gate)

'125, 'LS125A



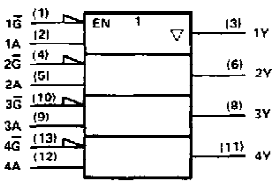
'126, LS126A



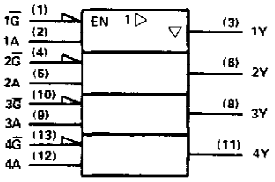
positive logic Y = A

logic symbols†

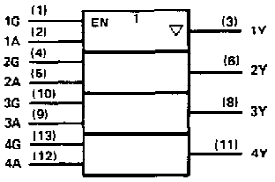
'125



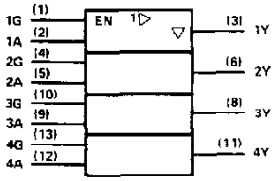
'LS125A



'126



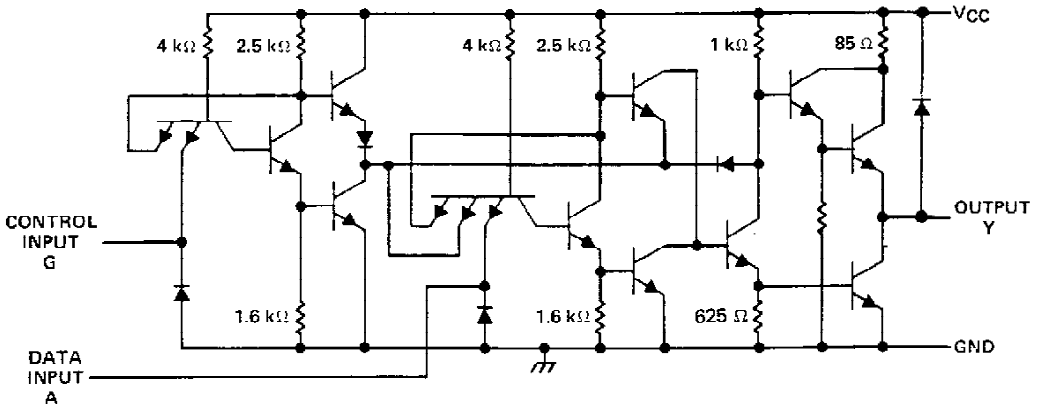
'LS126A



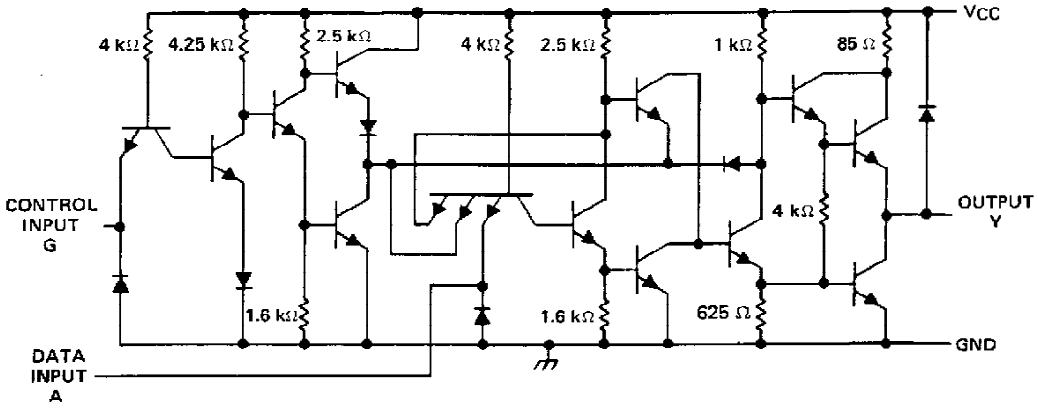
†These symbols are in accordance with ANSI/IEEE Std. 91-1984 and IEC Publication 617-12.
Pin numbers shown are for D, J, N, and W packages.

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

schematics (each gate)



'125 CIRCUITS



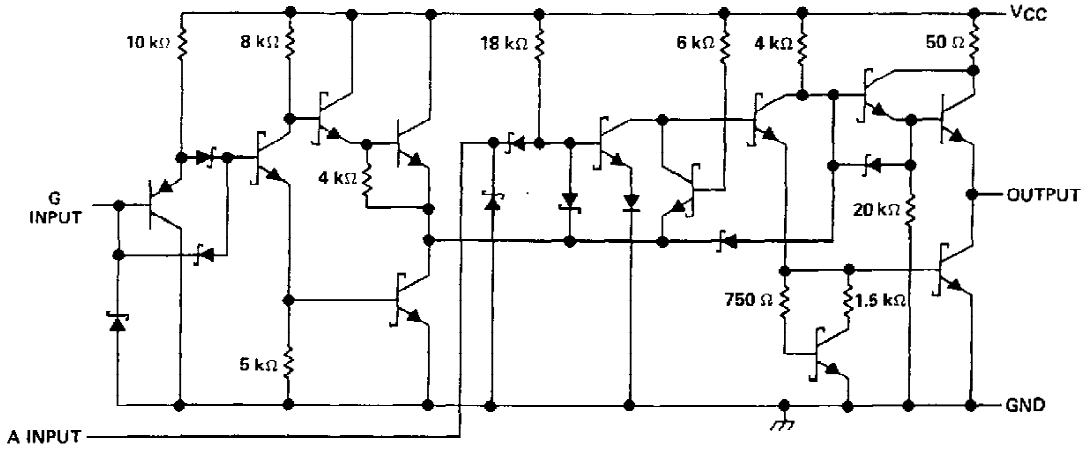
'126 CIRCUITS

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

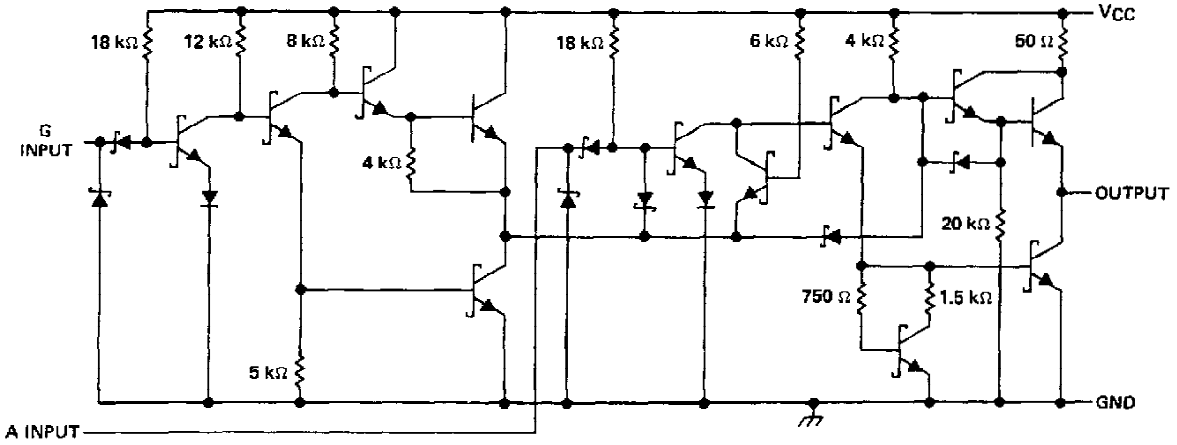
Supply voltage, V_{CC} (See Note 1)		7 V
Input voltage		5.5 V
Operating free-air temperature range:	SN54'	−55°C to 125°C
	SN74'	0°C to 70°C
Storage temperature range		−65°C to 150°C

NOTE 1: Voltage values are with respect to network ground terminal.

schematics (each gate)



'LS125A CIRCUITS



'LS126A CIRCUITS

Resistor values shown are nominal.

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1) . . . . . 7 V
Input voltage . . . . . 7 V
Operating free-air temperature range: SN54' . . . . . −55°C to 125°C
SN74' . . . . . 0°C to 70°C
Storage temperature range . . . . . −65°C to 150°C

NOTE 1: Voltage values are with respect to network ground terminals.

recommended operating conditions

		SN54125, SN54126			SN74125, SN74126			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.8			0.8	V
I_{OH}	High-level output current			−2			−5.2	mA
I_{OL}	Low-level output current			16			16	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS†			SN54125, SN54126			SN74125, SN74126			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = MIN,	I_I = −12 mA				1.5			1.5	V
V_{OH}	V_{CC} = MIN, V_{IL} = 0.8 V	V_{IH} = 2 V,	I_{OH} = −2 mA	2.4	3.3					V
			I_{OH} = −5.2 mA				2.4	3.1		
V_{OL}	V_{CC} = MIN, I_{OL} = 16 mA	V_{IH} = 2 V,	V_{IL} = 0.8 V,			0.4			0.4	V
I_{OZ}	V_{CC} = MAX, V_{IL} = 0.8 V	V_{IH} = 2 V,	V_O = 2.4 V			40			40	μA
			V_O = 0.4 V			−40			−40	
I_I	V_{CC} = MAX,	V_I = 6.5 V				1			1	mA
I_{IH}	V_{CC} = MAX,	V_I = 2.4 V				40			40	μA
I_{IL}	V_{CC} = MAX,	V_I = 0.4 V				−1.6			−1.6	mA
I_{OS}§	V_{CC} = MAX			−30		−70	−28		−70	mA
I_{CC}	V_{CC} = MAX, (see Note 2)		'125		32	54		32	54	mA
			'126		36	62		36	62	

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output should be shorted at a time.
NOTE 2: Data inputs = 0.V; output control = 4.5 V for '125 and 0 V for '126.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 3)

PARAMETER	TEST CONDITIONS		SN54/74125			SN54/74126			UNIT
			MIN	TYP	MAX	MIN	TYP	MAX	
t_{PLH}	R_L = 400 Ω,	C_L = 50 pF		8	13		8	13	ns
t_{PHL}				12	18		12	18	ns
t_{PZH}				11	17		11	18	ns
t_{PZL}				16	25		16	25	ns
t_{PHZ}	R_L = 400 Ω,	C_L = 5 pF		5	8		10	16	ns
t_{PLZ}				7	12		12	18	ns

NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

recommended operating conditions

		SN54LS125A SN54LS126A			SN74LS125A SN74LS126A			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.7			0.8	V
I_{OH}	High-level output current			−1			−2.6	mA
I_{OL}	Low-level output current			12			24	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS†			SN54LS125A SN54LS126A			SN74LS125A SN74LS126A			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = MIN,	I_I = −18 mA				−1.5			−1.5	V
V_{OH}	V_{CC} = MIN, V_{IH} = 2 V	V_{IL} = 0.7 V,	I_{OH} = −1 mA	2.4						V
		V_{IL} = 0.8 V,	I_{OH} = −2.6 mA				2.4			
V_{OL}	V_{CC} = MIN, V_{IH} = 2 V	V_{IL} = 0.7 V,	I_{OL} = 12 mA		0.25	0.4				V
		V_{IL} = 0.8 V,	I_{OL} = 12 mA					0.25	0.4	
		V_{IL} = 0.8 V,	I_{OL} = 24 mA					0.35	0.5	
I_{OZ}	V_{CC} = MAX, V_{IH} = 2 V	V_{IL} = 0.7 V	V_O = 2.4 V			20				μA
			V_O = 0.4 V			−20				
		V_{IL} = 0.8 V	V_O = 2.4 V						20	
			V_O = 0.4 V						−20	
I_I	V_{CC} = MAX,	V_I = 7 V				0.1			0.1	mA
I_{IH}	V_{CC} = MAX,	V_I = 2.7 V				20			20	μA
I_{IL}	V_{CC} = MAX, V_I = 0.4 V	'LS125A-$\overline{G}$ inputs				−0.2			−0.2	mA
		'LS125A-A inputs; 'LS126A All inputs				−0.4			−0.4	mA
I_{OS}§	V_{CC} = MAX			−40		−225	−40		−225	mA
I_{CC}	V_{CC} = MAX, (see Note 2)		'LS125A		11	20		11	20	mA
			'LS126A		12	22		12	22	

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output should be shorted at a time, and duration of the short circuit should not exceed one second.
NOTE 2: Data inputs = 0 V; Output controls = 4.5 V for 'LS125A and 0 V for 'LS126A.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 3)

PARAMETER	TEST CONDITIONS		SN54/74LS125A			SN54/74LS126A			UNIT
			MIN	TYP	MAX	MIN	TYP	MAX	
t_{PLH}	R_L = 667 Ω,	C_L = 45 pF		9	15		9	15	ns
t_{PHL}				7	18		8	18	ns
t_{PZH}				12	20		16	25	ns
t_{PZL}				15	25		21	35	ns
t_{PHZ}	R_L = 667 Ω,	C_L = 5 pF			20			25	ns
t_{PLZ}					20			25	ns

NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

GM12-N20VA-EN

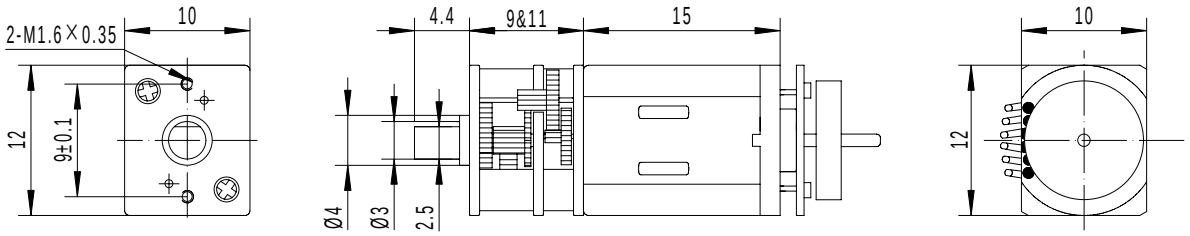
PMDC Dc Gear Motor

12X10mm gearbox plus N20 & N30 permanent magnet DC motor

Characters:

1. Small size dc gear motor with low speed and big torque
2. 12mm gear motor provide 0.1Nm torque and more reliable
3. Suitable to small diameter, low noise and big toque application
4. Dc Gear motors can match encoder,3-16pps
5. Reduction Ratio: 3、5、10、20、30、50、63、100、150、210、250、298、380、1000

Gear Motor Dimensions (mm)



15 g

Motor Technical Data

Type	Rated voltage	No-load speed	No-load current	Rated speed	Rated torque	Rated current	Output power	Stall torque	Stall current
	VDC	r/min	mA	r/min	g.cm	mA	W	g.cm	A
N20VA-13110	2.4	14800	75	12000	2.5	200	0.31	13	1.20
N20VA-09220	5.0	14800	30	12000	2.5	120	0.35	13	0.52
N20VA-05430	12.0	16000	12	12500	3.0	60	0.35	13	0.21

Gear motor Technical Data

Gearbox with motor：GM12-N20VA-13110-XXX-EN

Reduction ratio	10	30	50	100	150	210	250	298	380	1000
Length mm	9.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	12
No-load speed rpm	1400	470	285	145	96	69	58	50	38	14.5
Rated speed rpm	1200	400	240	120	80	57	48	40	31	12
Rated torque kg.cm	0.015	0.05	0.07	0.15	0.22	0.31	0.37	0.45	0.57	1.0
Max.momentary tolerance torque kg.cm	0.08	0.23	0.4	0.8	1.1	1.6	1.9	2.1	2.5	3.0

Gearbox with motor：GM12-N20VA-09220-XXX-EN

Reduction ratio	10	30	50	100	150	210	250	298	380	1000
Length mm	9.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	12
No-load speed rpm	1400	470	285	145	96	69	58	50	38	14.5
Rated speed rpm	1200	400	240	120	80	57	48	40	31	12
Rated torque kg.cm	0.015	0.05	0.07	0.15	0.22	0.31	0.37	0.45	0.57	1.0
Max.momentary tolerance torque kg.cm	0.08	0.23	0.4	0.8	1.1	1.6	1.9	2.1	2.5	3.0

Gearbox with motor：GM12-N20VA-05430-XXX-EN

Reduction ratio	10	30	50	100	150	210	250	298	380	1000
Length mm	9.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	12
No-load speed rpm	1500	530	310	160	110	78	60	55	43	14
Rated speed rpm	1100	400	250	125	83	60	50	42	33	13
Rated torque kg.cm	0.02	0.05	0.08	0.18	0.27	0.37	0.45	0.53	0.68	1.0
Max.momentary tolerance torque kg.cm	0.1	0.2	0.4	0.8	1.1	1.6	1.9	2.1	2.5	3.0

Radial Lead Type

Series: NHG Type : A

Japan
Malaysia
China
Taiwan

■ Features Endurance: +105°C 1000 to 2000 h
Smaller than Series NHE

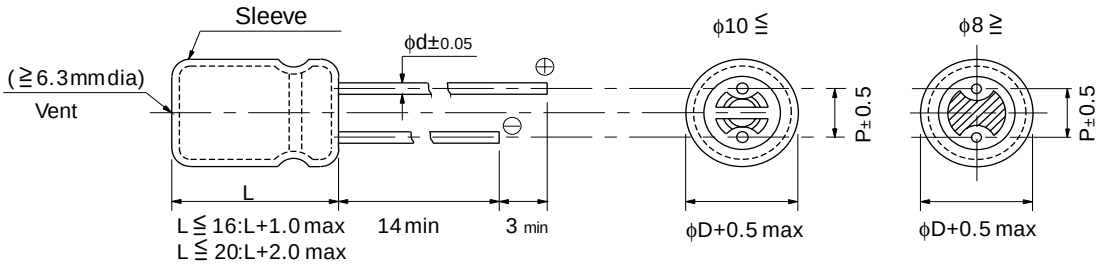
■ Specifications

Category temp. range	-55 to +105°C	-25 to +105°C
Rated W.V. Range	6.3 to 100 V.DC	160 to 450 V .DC
Nominal Cap. Range	0.1 to 22000 µF	1.0 to 330 µ F
Capacitance Tolerance	±20 % (120Hz/+20°C)	
DC Leakage Current	I ≦ 0.01 CV or 3(µ A) after 2 minutes application of rated working voltage at +20°C	I ≦ 0.06 CV + 10(µ A) after 2 minutes application of rated working voltage at +20°C
tan δ	Please see the attached standard products list	
Endurance	After the life test with DC voltage and +105±2°C ripple current value applied (The sum of DC and ripple peak voltage shall not exceed the rated working voltage), the capacitors shall meet the limits specified bellow. Duration : 6.3 to 100V.W. ; (ϕ5 to 8) =1000 hours , (ϕ10 to 18)= 2000 hours 160 to 450V.W. ; 2000 hours Post test requirements at +20 °C	
	Capacitance change: ±20% of initial measured value	
	tan δ: ≦ 200 % of initial specified value	
	DC leakage current: ≦ initial specified value	
Shelf Life	After storage for 1000 hours at +105±2 °C with no voltage applied and then being stabilized at +20 °C, capacitors shall meet the limits specified in Endurance. (With voltage treatment)	

■ Frequency correction factor for ripple current

W.V. (V.DC)	Cap. (µF)	Frequency(Hz)				
		60	120	1k	10k	100k
6.3 to 100	0.1 to 33	0.75	1	1.55	1.80	2.00
	47 to 470	0.80	1	1.35	1.50	1.50
	1000 to 22000	0.85	1	1.10	1.15	1.15
160 to 450	1 to 330	0.80	1	1.35	1.50	1.50

■ Dimensions in mm (not to scale)



Body Dia. ϕD	5	6.3	8	10	12.5	16	18
Lead Dia. ϕd	0.5	0.5	0.6	0.6	0.6	0.8	0.8
Lead space P	2.0	2.5	3.5	5.0	5.0	7.5	7.5

Rev. 02/04

■ Standard Products

W.V. (V)	Cap. (±20%) (µF)	Case size Dia. (mm)	Case size Length (mm)	Specification Ripple current (120Hz) (+105°C) (mA)	Specification tan δ (120Hz) (+20°C)	Specification Endurance (hours)	Lead Length Lead Dia. (mm)	Lead Space Straight (mm)	Lead Space Taping *B (mm)	Lead Space Taping *i (mm)	Part No.	Min. Packaging Q'ty Straight Leads (pcs)	Min. Packaging Q'ty Taping (pcs)
6.3	220	5	11	140	0.28	1000	0.5	2.0	5.0	2.5	ECA0JHG221( )	200	2000
	470	6.3	11.2	230	0.28	1000	0.5	2.5	5.0	2.5	ECA0JHG471( )	200	2000
	1000	8	11.5	380	0.28	1000	0.6	3.5	5.0		ECA0JHG102( )	200	1000
	2200	10	16	710	0.30	2000	0.6	5.0	5.0		ECA0JHG222( )	200	500
	3300	10	20	840	0.32	2000	0.6	5.0	5.0		ECA0JHG332( )	200	500
	4700	12.5	20	1090	0.34	2000	0.6	5.0	5.0		ECA0JHG472( )	200	500
	6800	12.5	25	1350	0.38	2000	0.6	5.0	5.0		ECA0JHG682( )	200	500
	10000	16	25	1650	0.46	2000	0.8	7.5	7.5		ECA0JHG103( )	100	250
	15000	16	31.5	2010	0.56	2000	0.8	7.5			ECA0JHG153	100	
	22000	18	35.5	2350	0.70	2000	0.8	7.5			ECA0JHG223	50	
10	330	6.3	11.2	200	0.24	1000	0.5	2.5	5.0	2.5	ECA1AHG331( )	200	2000
	470	8	11.5	250	0.24	1000	0.6	3.5	5.0		ECA1AHG471( )	200	1000
	1000	10	12.5	460	0.24	2000	0.6	5.0	5.0		ECA1AHG102( )	200	500
	2200	10	20	760	0.26	2000	0.6	5.0	5.0		ECA1AHG222( )	200	500
	3300	12.5	20	1000	0.28	2000	0.6	5.0	5.0		ECA1AHG332( )	200	500
	4700	12.5	25	1260	0.30	2000	0.6	5.0	5.0		ECA1AHG472( )	200	500
	6800	16	25	1570	0.34	2000	0.8	7.5	7.5		ECA1AHG682( )	100	250
	10000	16	31.5	1890	0.42	2000	0.8	7.5			ECA1AHG103	100	
	15000	18	35.5	2180	0.52	2000	0.8	7.5			ECA1AHG153	50	
16	100	5	11	110	0.20	1000	0.5	2.0	5.0	2.5	ECA1CHG101( )	200	2000
	220	6.3	11.2	180	0.20	1000	0.5	2.5	5.0	2.5	ECA1CHG221( )	200	2000
	330	8	11.5	260	0.20	1000	0.6	3.5	5.0		ECA1CHG331( )	200	1000
	470	8	11.5	310	0.20	1000	0.6	3.5	5.0		ECA1CHG471( )	200	1000
	1000	10	16	560	0.20	2000	0.6	5.0	5.0		ECA1CHG102( )	200	500
	2200	12.5	20	920	0.22	2000	0.6	5.0	5.0		ECA1CHG222( )	200	500
	3300	12.5	25	1170	0.24	2000	0.6	5.0	5.0		ECA1CHG332( )	200	500
	4700	16	25	1480	0.26	2000	0.8	7.5	7.5		ECA1CHG472( )	100	250
	6800	16	31.5	1780	0.30	2000	0.8	7.5			ECA1CHG682	100	
	10000	18	35.5	2060	0.38	2000	0.8	7.5			ECA1CHG103	50	
25	47	5	11	91	0.16	1000	0.5	2.0	5.0	2.5	ECA1EHG470( )	200	2000
	100	6.3	11.2	130	0.16	1000	0.5	2.5	5.0	2.5	ECA1EHG101( )	200	2000
	220	8	11.5	230	0.16	1000	0.6	3.5	5.0		ECA1EHG221( )	200	1000
	330	8	11.5	310	0.16	1000	0.6	3.5	5.0		ECA1EHG331( )	200	1000
	470	10	12.5	380	0.16	2000	0.6	5.0	5.0		ECA1EHG471( )	200	500
	1000	10	20	680	0.16	2000	0.6	5.0	5.0		ECA1EHG102( )	200	500
	2200	12.5	25	1090	0.18	2000	0.6	5.0	5.0		ECA1EHG222( )	200	500
	3300	16	25	1400	0.20	2000	0.8	7.5	7.5		ECA1EHG332( )	100	250
	4700	16	31.5	1750	0.22	2000	0.8	7.5			ECA1EHG472	100	
	6800	18	35.5	2040	0.26	2000	0.8	7.5			ECA1EHG682	50	
35	47	5	11	90	0.14	1000	0.5	2.0	5.0	2.5	ECA1VHG470( )	200	2000
	100	6.3	11.2	150	0.14	1000	0.5	2.5	5.0	2.5	ECA1VHG101( )	200	2000
	220	8	11.5	270	0.14	1000	0.6	3.5	5.0		ECA1VHG221( )	200	1000

When requesting taped product, please put the letter "B" or "i" between the "( )". Lead wire pitch B=5mm, 7.5mm. i=2.5mm.
The taping dimensions are explained on p.43 of our Catalog. Please use it as a reference guide.

■ Standard Products

W.V. (V)	Cap. (±20%) (μF)	Case size Dia. (mm)	Case size Length (mm)	Specification Ripple current (120Hz) (+105°C) (mA)	Specification tan δ (120Hz) (+20°C)	Specification Endurance (hours)	Lead Length Lead Dia. (mm)	Lead Length Lead Space Straight (mm)	Lead Length Lead Space Taping *B (mm)	Lead Length Lead Space Taping *i (mm)	Part No.	Min. Packaging Q'ty Straight Leads (pcs)	Min. Packaging Q'ty Taping (pcs)
35	330	10	12.5	350	0.14	2000	0.6	5.0	5.0		ECA1VHG331( )	200	500
	470	10	16	460	0.14	2000	0.6	5.0	5.0		ECA1VHG471( )	200	500
	1000	12.5	20	810	0.14	2000	0.6	5.0	5.0		ECA1VHG102( )	200	500
	2200	16	25	1260	0.16	2000	0.8	7.5	7.5		ECA1VHG222( )	100	250
	3300	16	31.5	1610	0.18	2000	0.8	7.5			ECA1VHG332	100	
	4700	18	35.5	1910	0.20	2000	0.8	7.5			ECA1VHG472	50	
50	0.1	5	11	1.1	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHG0R1( )	200	2000
	0.22	5	11	2.3	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHGR22( )	200	2000
	0.33	5	11	3.5	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHGR33( )	200	2000
	0.47	5	11	5	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHGR47( )	200	2000
	1	5	11	10	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHG010( )	200	2000
	2.2	5	11	18	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHG2R2( )	200	2000
	3.3	5	11	22	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHG3R3( )	200	2000
	4.7	5	11	26	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHG4R7( )	200	2000
	10	5	11	39	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHG100( )	200	2000
	22	5	11	65	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHG220( )	200	2000
	33	5	11	90	0.12	1000	0.5	2.0	5.0	2.5	ECA1HHG330( )	200	2000
	47	6.3	11.2	110	0.12	1000	0.5	2.5	5.0	2.5	ECA1HHG470( )	200	2000
	100	8	11.5	180	0.12	1000	0.6	3.5	5.0		ECA1HHG101( )	200	1000
	220	10	12.5	300	0.12	2000	0.6	5.0	5.0		ECA1HHG221( )	200	500
	330	10	16	410	0.12	2000	0.6	5.0	5.0		ECA1HHG331( )	200	500
	470	10	20	530	0.12	2000	0.6	5.0	5.0		ECA1HHG471( )	200	500
	1000	12.5	25	950	0.12	2000	0.6	5.0	5.0		ECA1HHG102( )	200	500
	2200	16	31.5	1470	0.14	2000	0.8	7.5			ECA1HHG222	100	
	3300	18	35.5	1770	0.16	2000	0.8	7.5			ECA1HHG332	50	
63	10	5	11	46	0.10	1000	0.5	2.0	5.0	2.5	ECA1JHG100( )	200	2000
	22	5	11	71	0.10	1000	0.5	2.0	5.0	2.5	ECA1JHG220( )	200	2000
	33	6.3	11.2	100	0.10	1000	0.5	2.5	5.0	2.5	ECA1JHG330( )	200	2000
	47	6.3	11.2	120	0.10	1000	0.5	2.5	5.0	2.5	ECA1JHG470( )	200	2000
	100	10	12.5	215	0.10	2000	0.6	5.0	5.0		ECA1JHG101( )	200	500
	220	10	16	335	0.10	2000	0.6	5.0	5.0		ECA1JHG221( )	200	500
	330	10	20	510	0.10	2000	0.6	5.0	5.0		ECA1JHG331( )	200	500
	470	12.5	20	640	0.10	2000	0.6	5.0	5.0		ECA1JHG471( )	200	500
	1000	16	25	930	0.10	2000	0.8	7.5	7.5		ECA1JHG102( )	100	250
	2200	18	35.5	1610	0.12	2000	0.8	7.5			ECA1JHG222	50	
100	0.47	5	11	9	0.08	1000	0.5	2.0	5.0	2.5	ECA2AHGR47( )	200	2000
	1	5	11	14	0.08	1000	0.5	2.0	5.0	2.5	ECA2AHG010( )	200	2000
	2.2	5	11	21	0.08	1000	0.5	2.0	5.0	2.5	ECA2AHG2R2( )	200	2000
	3.3	5	11	31	0.08	1000	0.5	2.0	5.0	2.5	ECA2AHG3R3( )	200	2000
	4.7	5	11	38	0.08	1000	0.5	2.0	5.0	2.5	ECA2AHG4R7( )	200	2000
	10	6.3	11.2	54	0.08	1000	0.5	2.5	5.0	2.5	ECA2AHG100( )	200	2000
	22	6.3	11.2	93	0.08	1000	0.5	2.5	5.0	2.5	ECA2AHG220( )	200	2000

When requesting taped product, please put the letter "B" or "i" between the "( )". Lead wire pitch B=5mm, 7.5mm. i=2.5mm.
The taping dimensions are explained on p.43 of our Catalog. Please use it as a reference guide.

■ Standard Products

W.V. (V)	Cap. (±20%) (μF)	Case size Dia. (mm)	Case size Length (mm)	Specification Ripple current (120Hz) (+105°C) (mA)	Specification tan δ (120Hz) (+20°C)	Specification Endurance (hours)	Lead Length Lead Dia. (mm)	Lead Space Straight (mm)	Lead Space Taping *B (mm)	Lead Space Taping *i (mm)	Part No.	Min. Packaging Q'ty Straight Leads (pcs)	Min. Packaging Q'ty Taping (pcs)
100	33	8	11.5	130	0.08	1000	0.6	3.5	5.0		ECA2AHG330( )	200	1000
	47	10	12.5	165	0.08	2000	0.6	5.0	5.0		ECA2AHG470( )	200	500
	100	10	20	265	0.08	2000	0.6	5.0	5.0		ECA2AHG101( )	200	500
	220	12.5	25	440	0.08	2000	0.6	5.0	5.0		ECA2AHG221( )	200	500
	330	16	25	540	0.08	2000	0.8	7.5	7.5		ECA2AHG331( )	100	250
	470	16	25	715	0.08	2000	0.8	7.5	7.5		ECA2AHG471( )	100	250
	1000	18	35.5	985	0.08	2000	0.8	7.5			ECA2AHG102	50	
160	1	6.3	11.2	17	0.15	2000	0.5	2.5	5.0	2.5	ECA2CHG010( )	200	2000
	2.2	6.3	11.2	25	0.15	2000	0.5	2.5	5.0	2.5	ECA2CHG2R2( )	200	2000
	3.3	6.3	11.2	36	0.15	2000	0.5	2.5	5.0	2.5	ECA2CHG3R3( )	200	2000
	4.7	6.3	11.2	43	0.15	2000	0.5	2.5	5.0	2.5	ECA2CHG4R7( )	200	2000
	10	10	12.5	70	0.15	2000	0.6	5.0	5.0		ECA2CHG100( )	200	500
	22	10	20	130	0.15	2000	0.6	5.0	5.0		ECA2CHG220( )	200	500
	33	10	20	180	0.15	2000	0.6	5.0	5.0		ECA2CHG330( )	200	500
	47	12.5	20	220	0.15	2000	0.6	5.0	5.0		ECA2CHG470( )	200	500
	100	16	25	335	0.15	2000	0.8	7.5	7.5		ECA2CHG101( )	100	250
	220	16	31.5	540	0.15	2000	0.8	7.5			ECA2CHG221	100	
	330	18	31.5	705	0.15	2000	0.8	7.5			ECA2CHG331	50	
200	1	6.3	11.2	17	0.15	2000	0.5	2.5	5.0	2.5	ECA2DHG010( )	200	2000
	2.2	6.3	11.2	25	0.15	2000	0.5	2.5	5.0	2.5	ECA2DHG2R2( )	200	2000
	3.3	6.3	11.2	36	0.15	2000	0.5	2.5	5.0	2.5	ECA2DHG3R3( )	200	2000
	4.7	8	11.5	50	0.15	2000	0.6	3.5	5.0		ECA2DHG4R7( )	200	1000
	10	10	16	80	0.15	2000	0.6	5.0	5.0		ECA2DHG100( )	200	500
	22	10	20	140	0.15	2000	0.6	5.0	5.0		ECA2DHG220( )	200	500
	33	12.5	20	190	0.15	2000	0.6	5.0	5.0		ECA2DHG330( )	200	500
	47	12.5	20	220	0.15	2000	0.6	5.0	5.0		ECA2DHG470( )	200	500
	100	16	25	335	0.15	2000	0.8	7.5	7.5		ECA2DHG101( )	100	250
	220	18	31.5	575	0.15	2000	0.8	7.5			ECA2DHG221	50	
250	1	6.3	11.2	17	0.15	2000	0.5	2.5	5.0	2.5	ECA2EHG010( )	200	2000
	2.2	6.3	11.2	29	0.15	2000	0.5	2.5	5.0	2.5	ECA2EHG2R2( )	200	2000
	3.3	8	11.5	42	0.15	2000	0.6	3.5	5.0		ECA2EHG3R3( )	200	1000
	4.7	8	11.5	50	0.15	2000	0.6	3.5	5.0		ECA2EHG4R7( )	200	1000
	10	10	16	88	0.15	2000	0.6	5.0	5.0		ECA2EHG100( )	200	500
	22	12.5	20	155	0.15	2000	0.6	5.0	5.0		ECA2EHG220( )	200	500
	33	12.5	20	190	0.15	2000	0.6	5.0	5.0		ECA2EHG330( )	200	500
	47	12.5	25	230	0.15	2000	0.6	5.0	5.0		ECA2EHG470( )	200	500
	100	16	31.5	365	0.15	2000	0.8	7.5			ECA2EHG101	100	
350	1	6.3	11.2	18	0.20	2000	0.5	2.5	5.0	2.5	ECA2VHG010( )	200	2000
	2.2	8	11.5	31	0.20	2000	0.6	3.5	5.0		ECA2VHG2R2( )	200	1000
	3.3	10	12.5	38	0.20	2000	0.6	5.0	5.0		ECA2VHG3R3( )	200	500
	4.7	10	16	50	0.20	2000	0.6	5.0	5.0		ECA2VHG4R7( )	200	500
	10	10	20	82	0.20	2000	0.6	5.0	5.0		ECA2VHG100( )	200	500

When requesting taped product, please put the letter "B" or "i" between the "( )". Lead wire pitch B=5mm, 7.5mm. i=2.5mm.
The taping dimensions are explained on p.43 of our Catalog. Please use it as a reference guide.

■ Standard Products

W.V. (V)	Cap. (±20%) (µF)	Case size Dia. (mm)	Case size Length (mm)	Specification Ripple current (120Hz) (+105°C) (mA)	Specification tan δ (120Hz) (+20°C)	Specification Endurance (hours)	Lead Length Lead Dia. (mm)	Lead Length Lead Space Straight (mm)	Lead Length Lead Space Taping *B (mm)	Lead Length Lead Space Taping *i (mm)	Part No.	Min. Packaging Q'ty Straight Leads (pcs)	Min. Packaging Q'ty Taping (pcs)
350	22	12.5	20	130	0.20	2000	0.6	5.0	5.0		ECA2VHG220( )	200	500
	33	16	25	195	0.20	2000	0.8	7.5	7.5		ECA2VHG330( )	100	250
	47	16	25	230	0.20	2000	0.8	7.5	7.5		ECA2VHG470( )	100	250
	100	18	31.5	375	0.20	2000	0.8	7.5			ECA2VHG101	50	
400	1	6.3	11.2	18	0.24	2000	0.5	2.5	5.0	2.5	ECA2GHG010( )	200	2000
	2.2	8	11.5	30	0.24	2000	0.6	3.5	5.0		ECA2GHG2R2( )	200	1000
	3.3	10	12.5	40	0.24	2000	0.6	5.0	5.0		ECA2GHG3R3( )	200	500
	4.7	10	16	50	0.24	2000	0.6	5.0	5.0		ECA2GHG4R7( )	200	500
	10	10	20	80	0.24	2000	0.6	5.0	5.0		ECA2GHG100( )	200	500
	22	12.5	25	145	0.24	2000	0.6	5.0	5.0		ECA2GHG220( )	200	500
	33	16	25	195	0.24	2000	0.8	7.5	7.5		ECA2GHG330( )	100	250
	47	16	31.5	250	0.24	2000	0.8	7.5			ECA2GHG470	100	
450	1	8	11.5	18	0.24	2000	0.6	3.5	5.0		ECA2WHG010( )	200	1000
	2.2	10	12.5	29	0.24	2000	0.6	5.0	5.0		ECA2WHG2R2( )	200	500
	3.3	10	16	41	0.24	2000	0.6	5.0	5.0		ECA2WHG3R3( )	200	500
	4.7	10	20	49	0.24	2000	0.6	5.0	5.0		ECA2WHG4R7( )	200	500
	10	12.5	20	75	0.24	2000	0.6	5.0	5.0		ECA2WHG100( )	200	500
	22	16	25	115	0.24	2000	0.8	7.5	7.5		ECA2WHG220( )	100	250
	33	16	31.5	155	0.24	2000	0.8	7.5			ECA2WHG330	100	

When requesting taped product, please put the letter "B" or "i" between the "( )". Lead wire pitch B=5mm, 7.5mm. i=2.5mm.
The taping dimensions are explained on p.43 of our Catalog. Please use it as a reference guide.

Order this document
by 1N4001/D

Axial Lead Standard Recovery Rectifiers

This data sheet provides information on subminiature size, axial lead mounted rectifiers for general–purpose low–power applications.

1N4001 thru 1N4007

1N4004 and 1N4007 are Motorola Preferred Devices

LEAD MOUNTED RECTIFIERS 50–1000 VOLTS DIFFUSED JUNCTION

CASE 59–03
DO–41

Mechanical Characteristics

- Case: Epoxy, Molded
- Weight: 0.4 gram (approximately)
- Finish: All External Surfaces Corrosion Resistant and Terminal Leads are Readily Solderable
- Lead and Mounting Surface Temperature for Soldering Purposes: 220°C Max. for 10 Seconds, 1/16″ from case
- Shipped in plastic bags, 1000 per bag.
- Available Tape and Reeled, 5000 per reel, by adding a “RL” suffix to the part number
- Polarity: Cathode Indicated by Polarity Band
- Marking: 1N4001, 1N4002, 1N4003, 1N4004, 1N4005, 1N4006, 1N4007

MAXIMUM RATINGS

Rating	Symbol	1N4001	1N4002	1N4003	1N4004	1N4005	1N4006	1N4007	Unit
*Peak Repetitive Reverse Voltage Working Peak Reverse Voltage DC Blocking Voltage	V_{RRM} V_{RWM} V_R	50	100	200	400	600	800	1000	Volts
*Non–Repetitive Peak Reverse Voltage (halfwave, single phase, 60 Hz)	V_{RSM}	60	120	240	480	720	1000	1200	Volts
*RMS Reverse Voltage	$V_{R(RMS)}$	35	70	140	280	420	560	700	Volts
*Average Rectified Forward Current (single phase, resistive load, 60 Hz, see Figure 8, T_A = 75°C)	I_O	1.0							Amp
*Non–Repetitive Peak Surge Current (surge applied at rated load conditions, see Figure 2)	I_{FSM}	30 (for 1 cycle)							Amp
Operating and Storage Junction Temperature Range	T_J T_{stg}	– 65 to +175							°C

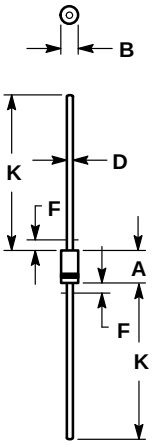
ELECTRICAL CHARACTERISTICS*

Rating	Symbol	Typ	Max	Unit
Maximum Instantaneous Forward Voltage Drop (i_F = 1.0 Amp, T_J = 25°C) Figure 1	v_F	0.93	1.1	Volts
Maximum Full–Cycle Average Forward Voltage Drop (I_O = 1.0 Amp, T_L = 75°C, 1 inch leads)	$V_{F(AV)}$	—	0.8	Volts
Maximum Reverse Current (rated dc voltage) (T_J = 25°C) (T_J = 100°C)	I_R	 0.05 1.0	 10 50	µA
Maximum Full–Cycle Average Reverse Current (I_O = 1.0 Amp, T_L = 75°C, 1 inch leads)	$I_{R(AV)}$	—	30	µA

*Indicates JEDEC Registered Data

Preferred devices are Motorola recommended choices for future use and best overall value.

PACKAGE DIMENSIONS



NOTES:
1. ALL RULES AND NOTES ASSOCIATED WITH JEDEC DO–41 OUTLINE SHALL APPLY.
2. POLARITY DENOTED BY CATHODE BAND.
3. LEAD DIAMETER NOT CONTROLLED WITHIN F DIMENSION.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	4.07	5.20	0.160	0.205
B	2.04	2.71	0.080	0.107
D	0.71	0.86	0.028	0.034
F	—	1.27	—	0.050
K	27.94	—	1.100	—

CASE 59–03
(DO–41)
ISSUE M

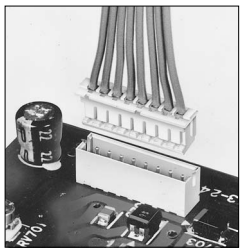
How to reach us:
USA / EUROPE / Locations Not Listed: Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217. 303–675–2140 or 1–800–441–2447

JAPAN: Nippon Motorola Ltd.: SPD, Strategic Planning Office, 4–32–1,
Nishi–Gotanda, Shinagawa–ku, Tokyo 141, Japan. 81–3–5487–8488

Mfax™: RMFAX0@email.sps.mot.com – TOUCHTONE 602–244–6609
– US & Canada ONLY 1–800–774–1848

ASIA / PACIFIC: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,
51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852–26629298

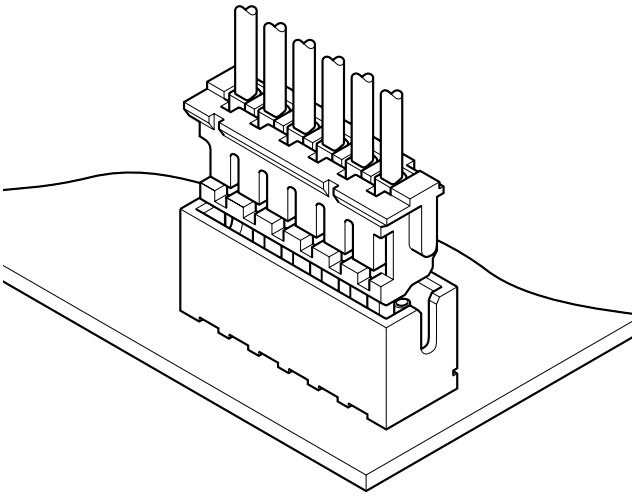
INTERNET: http://motorola.com/sps



ZH CONNECTOR

Emboss Tape

1.5mm pitch/Disconnectable Crimp style connectors



The ZH connector is very small with a mounting height of 5.6mm and a thickness of only 3.5mm. Contact retention lances on the housing make insertion easier, while the dimple at the center of the contact ensures reliable contact with the post.

- ***Compact, low profile design***
- ***Housing lances***
- ***Reliable contact construction***
- ***Distortion resistant construction***
- ***Compatible with the ZR insulation displacement connectors***

Specifications

- Current rating: 1.0A AC, DC (AWG #26)
- Voltage rating: 50V AC, DC
- Temperature range: -25˚C to +85˚C (including temperature rise in applying electrical current)
- Contact resistance: Initial value/20m Ω max. After environmental testing/30m Ω max.
- Insulation resistance: 500M Ω min.
- Withstanding voltage: 500V AC/minute
- Applicable wire: AWG #32 to #26
- Applicable PC board thickness: 0.6 to 1.2mm,1.6mm

* Compliant with RoHS.
* Refer to "General Instruction and Notice when using Terminals and Connectors" at the end of this catalog.
* Contact JST for details.

Standards

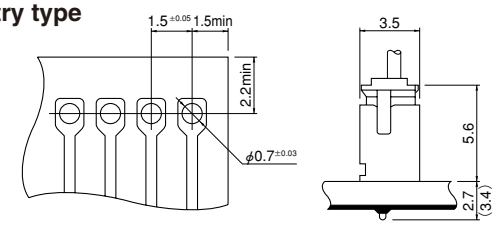
Recognized E60389

Certified LR20812

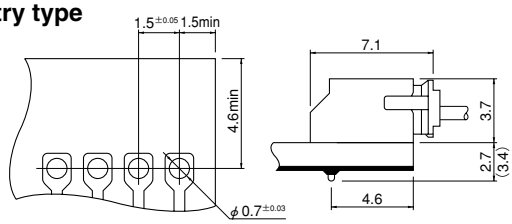
PC board layout and Assembly layout

〈Through-hole type (viewed from soldering side)〉

Top entry type



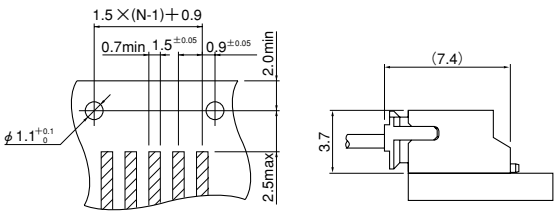
Side entry type



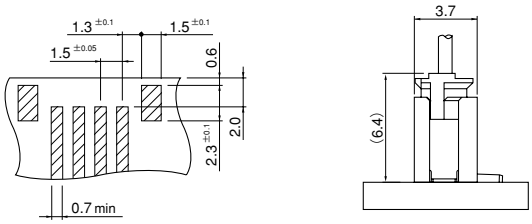
Note: 1. Tolerances are non-cumulative: ±0.05mm for all centers.
2. Hole dimension differ according to the kind of PC board and piercing method. If PC boards made of hard material are used, the hole dimensions should be larger. The dimensions above should serve as a guideline. Contact JST for details.

〈SMT type (viewed from component side)〉

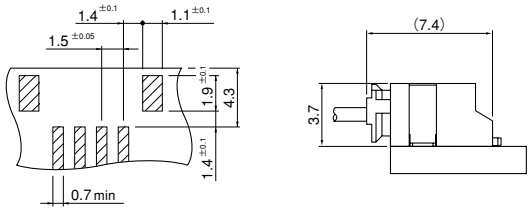
SM2 type
Side entry type



SM4 type
Top entry type

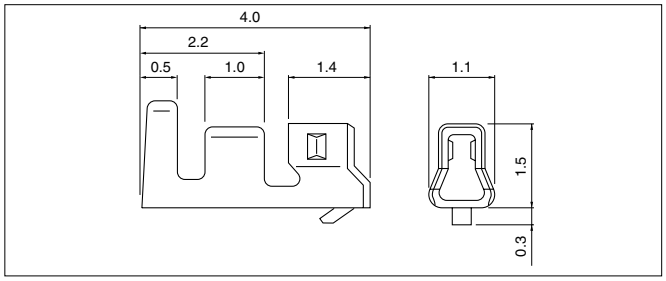


Side entry type



Note: 1. N: Number of circuits
2. Tolerances are non-cumulative: ±0.05mm for all centers.
3. The dimensions above should serve as a guideline. Contact JST for details.

Contact



Model No.	Applicable wire			Q'ty / reel
	mm²	AWG#	Insulation O.D. (mm)	
SZH-002T-P0.5	0.08 ～0.13	28～26	0.8～1.1	13,000
SZH-003T-P0.5	0.032～0.08	32～28	0.5～0.9	16,000

Material and Finish
Phosphor bronze, tin-plated (reflow treatment)

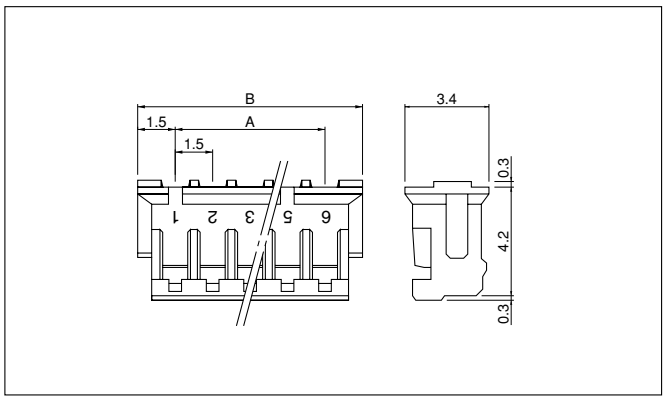
RoHS compliance

Contact	Crimping machine	Applicator		
		Crimp applicator	Dies	Crimp applicator with dies
SZH-002T-P0.5	AP-K2N	MKS-L	MK/SZH-002-05	APLMK SZH002-05
		—	—	—

*Note: *Strip-crimp applicator.*

Contact	Crimping machine	Applicator		
		Crimp applicator	Dies	Crimp applicator with dies
SZH-003T-P0.5	AP-K2N	MKS-L	MK/SZH-003-05	APLMK SZH003-05
		*MKS-SC	SC/SZH-003-05	APLSC SZH003-05

Housing



<For reference> As the color identification,
the following alphabet shall be put in the underlined part.
For availability, delivery and minimum order quantity, contact JST.

ex. **ZHR-2-oo**
(blank)…natural (white)
K…black R…red E…blue M…green

Circuits	Model No.	Dimensions (mm)		Q'ty / bag
		A	B	
2	**ZHR-2**	1.5	4.5	1,000
3	**ZHR-3**	3.0	6.0	1,000
4	**ZHR-4**	4.5	7.5	1,000
5	**ZHR-5**	6.0	9.0	1,000
6	**ZHR-6**	7.5	10.5	1,000
6	**ZHR-6(3.0)**	15.0	18.0	1,000
7	**ZHR-7**	9.0	12.0	1,000
8	**ZHR-8**	10.5	13.5	1,000
9	**ZHR-9**	12.0	15.0	1,000
10	**ZHR-10**	13.5	16.5	1,000
11	**ZHR-11**	15.0	18.0	1,000
12	**ZHR-12**	16.5	19.5	1,000
13	**ZHR-13**	18.0	21.0	1,000

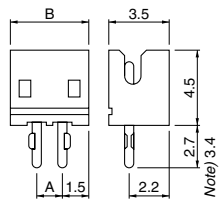
Material
PA 66, UL94V-0, natural (white)

RoHS compliance
Note: 1. ZHR-6(3.0) is 6 circuits 3.0mm pitch plugged up.
As for this housing, only circuit No.1 is marked.
2. ZHR-6(3.0) is not UL/CSA approved.

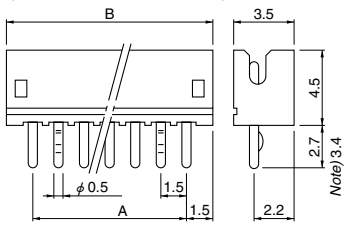
Through-hole type shrouded header

Top entry type

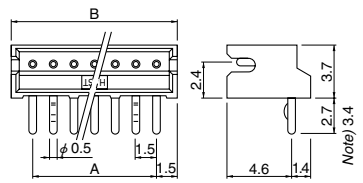
(2 circuits)



(3 circuits or more)



Side entry type



Circuits	Model No.				Dimensions (mm)		Q'ty / box		
	Top entry type	Side entry type	Note) Top entry type	Note) Side entry type	A	B	Top entry type	Side entry type	Side entry type-3.4
2	**B2B-ZR**	**S2B-ZR**	**B2B-ZR-3.4**	**S2B-ZR-3.4**	1.5	4.5	2,000	2,000	2,000
3	**B3B-ZR**	**S3B-ZR**	**B3B-ZR-3.4**	**S3B-ZR-3.4**	3.0	6.0	2,000	2,000	1,000
4	**B4B-ZR**	**S4B-ZR**	**B4B-ZR-3.4**	**S4B-ZR-3.4**	4.5	7.5	2,000	2,000	2,000
5	**B5B-ZR**	**S5B-ZR**	**B5B-ZR-3.4**	**S5B-ZR-3.4**	6.0	9.0	2,000	1,000	1,000
6	**B6B-ZR**	**S6B-ZR**	**B6B-ZR-3.4**	**S6B-ZR-3.4**	7.5	10.5	2,000	1,000	1,000
7	**B7B-ZR**	**S7B-ZR**	**B7B-ZR-3.4**	**S7B-ZR-3.4**	9.0	12.0	1,000	1,000	1,000
8	**B8B-ZR**	**S8B-ZR**	**B8B-ZR-3.4**	**S8B-ZR-3.4**	10.5	13.5	1,000	1,000	1,000
9	**B9B-ZR**	**S9B-ZR**	**B9B-ZR-3.4**	**S9B-ZR-3.4**	12.0	15.0	1,000	1,000	1,000
10	**B10B-ZR**	**S10B-ZR**	**B10B-ZR-3.4**	**S10B-ZR-3.4**	13.5	16.5	1,000	1,000	1,000
11	**B11B-ZR**	**S11B-ZR**	**B11B-ZR-3.4**	**S11B-ZR-3.4**	15.0	18.0	1,000	500	500
12	**B12B-ZR**	**S12B-ZR**	**B12B-ZR-3.4**	**S12B-ZR-3.4**	16.5	19.5	1,000	500	500
13	**B13B-ZR**	**S13B-ZR**	**B13B-ZR-3.4**	**S13B-ZR-3.4**	18.0	21.0	500	500	500

Material and Finish
Pin: Brass, copper-undercoated, tin-plated (reflow treatment) Wafer: Glass-filled PA 66, UL94V-0, natural (ivory)

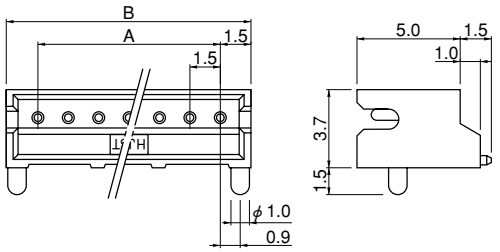
RoHS compliance *This product displays (LF)(SN) on a label.*
Note: Headers with 3.4mm long solder tails, suited for the 1.6mm thick PC board.

<For reference> As the color identification,
the following alphabet shall be put in the underlined part.
For availability, delivery and minimum order quantity, contact JST.

ex. **B2B-ZR-3.4-oo**
(blank)…natural (ivory)
K…black R…red E…blue M…green

SMT type shrouded header

SM2 type
Side entry type



SM2 type

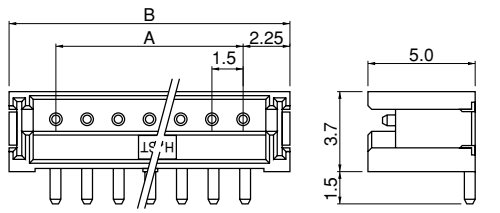
Circuits	Model No.	Dimensions (mm)		Q'ty / reel
	Side entry type	A	B	
2	**S2B-ZR-SM2-TF**	1.5	4.5	1,000
3	**S3B-ZR-SM2-TF**	3.0	6.0	1,000
4	**S4B-ZR-SM2-TF**	4.5	7.5	1,000
5	**S5B-ZR-SM2-TF**	6.0	9.0	1,000
6	**S6B-ZR-SM2-TF**	7.5	10.5	1,000
7	**S7B-ZR-SM2-TF**	9.0	12.0	1,000
8	**S8B-ZR-SM2-TF**	10.5	13.5	1,000
9	**S9B-ZR-SM2-TF**	12.0	15.0	1,000
10	**S10B-ZR-SM2-TF**	13.5	16.5	1,000
11	**S11B-ZR-SM2-TF**	15.0	18.0	1,000
12	**S12B-ZR-SM2-TF**	16.5	19.5	1,000
13	**S13B-ZR-SM2-TF**	18.0	21.0	1,000

Material and Finish

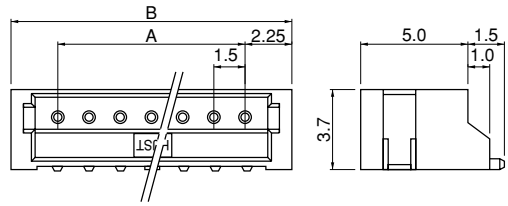
Pin: Brass, copper-undercoated, tin-plated (reflow treatment)
Wafer: Glass-filled PA 46, UL94V-0, natural (ivory)

RoHS compliance *This product displays (LF)(SN) on a label.*

SM4 type (with solder tabs)
Top entry type



Side entry type



SM4 type

Circuits	Model No.		Dimensions (mm)		Q'ty / reel	
	Top entry type	Side entry type	A	B	Top entry type	Side entry type
2	**B2B-ZR-SM4-TF**	**S2B-ZR-SM4A-TF**	1.5	6.0	1,000	1,000
3	**B3B-ZR-SM4-TF**	**S3B-ZR-SM4A-TF**	3.0	7.5	1,000	1,000
4	**B4B-ZR-SM4-TF**	**S4B-ZR-SM4A-TF**	4.5	9.0	1,000	1,000
5	**B5B-ZR-SM4-TF**	**S5B-ZR-SM4A-TF**	6.0	10.5	1,000	1,000
6	**B6B-ZR-SM4-TF**	**S6B-ZR-SM4A-TF**	7.5	12.0	1,000	1,000
7	**B7B-ZR-SM4-TF**	**S7B-ZR-SM4A-TF**	9.0	13.5	1,000	1,000
8	**B8B-ZR-SM4-TF**	**S8B-ZR-SM4A-TF**	10.5	15.0	1,000	1,000
9	**B9B-ZR-SM4-TF**	**S9B-ZR-SM4A-TF**	12.0	16.5	1,000	1,000
10	**B10B-ZR-SM4-TF**	**S10B-ZR-SM4A-TF**	13.5	18.0	1,000	1,000
11	**B11B-ZR-SM4-TF**	**S11B-ZR-SM4A-TF**	15.0	19.5	1,000	1,000
12	**B12B-ZR-SM4-TF**	**S12B-ZR-SM4A-TF**	16.5	21.0	1,000	1,000
13	**B13B-ZR-SM4-TF**	**S13B-ZR-SM4A-TF**	18.0	22.5	1,000	1,000

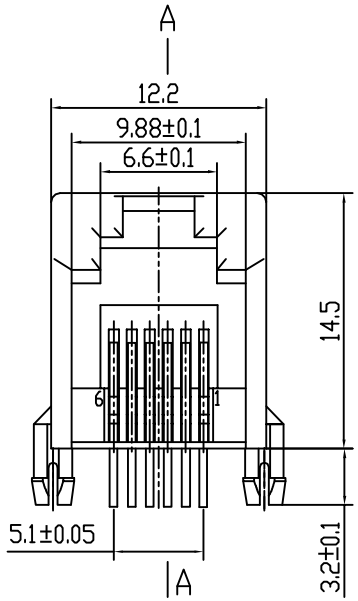
Material and Finish

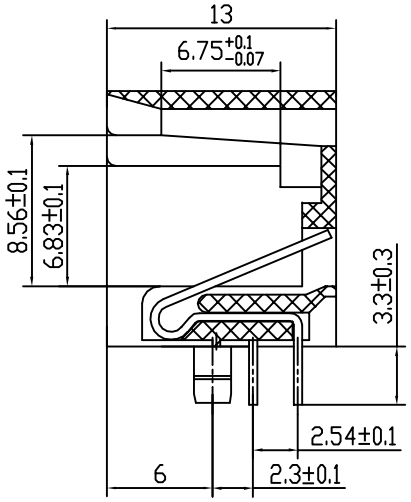
Pin: Brass, copper-undercoated, tin-plated (reflow treatment)
Wafer: PA 6T, UL94V-0, natural (ivory)
Solder tab: Brass, copper-undercoated, tin-plated (reflow treatment)

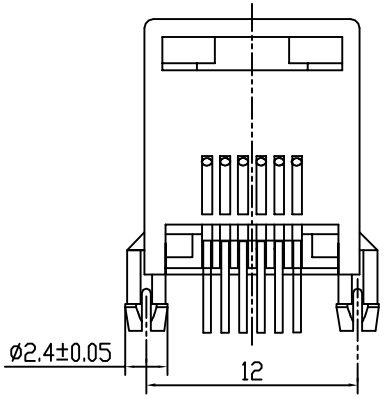
RoHS compliance *This product displays (LF)(SN) on a label.*
Note: 1. The products listed above are supplied on embossed-tape.
2. Contact JST for the top entry type headers with suction cap.

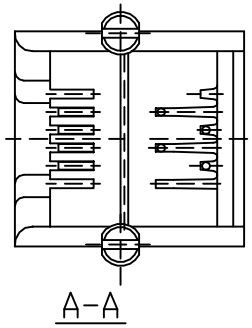
<For reference> As the color identification,
the following alphabet shall be put in the underlined part.
For availability, delivery and minimum order quantity, contact JST.

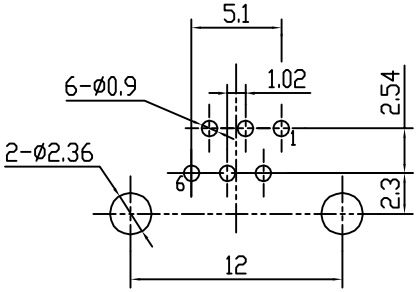
ex. **S2B-ZR-SM4A-oo-TF**
(blank)…natural (ivory)
K…black R…red E…blue M…green











PCB LAYOUT - COMPONENT VIEW

TECHNICAL CHARACTERISTICS

MATERIAL
INSULATOR MATERIAL: PBT
FLAMMABILITY RATING UL94-V0
COLOUR: BLACK
CONTACT MATERIAL: COPPER ALLOY
CONTACT TYPE: STAMPED
CONTACT PLATING: GOLD
SHIELDING: BRONZE NI PLATED
QUALITY CLASS: 500 MATING CYCLES

ENVIRONMENTAL
OPERATING TEMPERATURE: –25 °C UP TO 85 °C
COMPLIANCE: LEAD FREE AND ROHS

ELECTRICAL
CURRENT RATING: 1.5 A
WORKING VOLTAGE: 120 VAC
INSULATION RESISTANCE: >500 MOHM
DIELECTRIC WITHSTANDING VOLTAGE: 1 000 VAC/MN
CONTACT RESISTANCE: 20 mOHM MAX

SOLDERING
WAVE PROCESS ONLY

PACKAGING
TRAY

RoHS Compliant

REV	DATE	FILE	BY
G			
F			
E			
D			
C			
B			
A	02-NOV-08	PDF	CH

PROJECTION:

GENERAL TOLERANCE
.X = +/_ 0.2
.XX = +/_ 0.15

WÜRTH ELEKTRONIK

APPROVAL: RJ

UNIT: MM
SCALE:
SHEET: 1/1
DRAW:

DESCRIPTION: HORIZONTAL PLASTIC 6P6C MODULAR JACK
TAP UP

WERI PART NO: 615 006 138 421

SIZE
A4

25000N Series

Miniature Slide Switches

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

E

SPECIFICATIONS

Contact ratings:
Gold; 0.4 Volt-Amps (VA) max. at 20 V max. (AC or DC)
Silver; 4 A at 125 VAC,2 A at 250 VAC or 3 A at 30 VDC
Initial contact resistance: Gold; 50 milliohms max.
Silver; 10 milliohms max.
Insulation resistance: 1000 megohms min. at 500 VDC
Dielectric strength: 1000 V rms min. between terms.
Electrical life: Gold; 20,000 cycles, Silver; 10,000 cycles
Operating temperature range: -40°C to +85°C
Solder heat resistance: 300°C max. for 5 seconds
Washing not recommended.

FEATURES

- **Molded-in terminals.**
- **.100" (2,54mm) and .200" (5,08mm) terminal spacing.**
- **Many additional models & options available (see page E51).**
- **Optional UL recognized & marked models available .**

MATERIALS

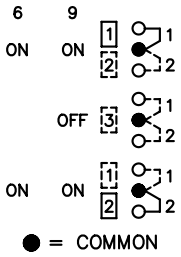
Contacts & terminals: Gold or silver available (see contact ratings)
Actuator and Cover: Glass filled polyamide
Case: PBT high temp. thermoplastic **Terminal seal:** Epoxy

MODEL NO.	PLATING	ACTION
25136NLDB	Gold	on - on
25139NLDB	Gold	on off on
25136NAB	Silver	on - on
25139NAB	Silver	on off on

VERTICAL ACTUATOR

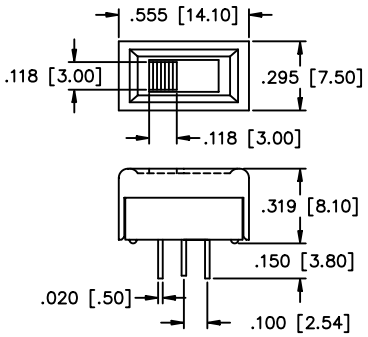
SINGLE POLE
FLUSH ACTUATOR

Function
(see pg. E51)

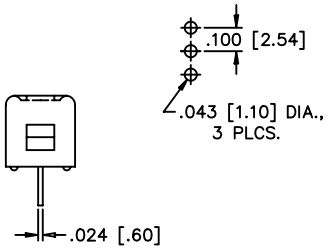


SCHEMATIC

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)



MECHANICAL OUTLINE



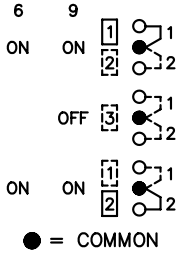
P.C. BOARD LAYOUT

MODEL NO.	PLATING	ACTION
25336NLD	Gold	on - on
25339NLD	Gold	on off on
25336NA	Silver	on - on
25339NA	Silver	on off on

RIGHT ANGLE ACTUATOR

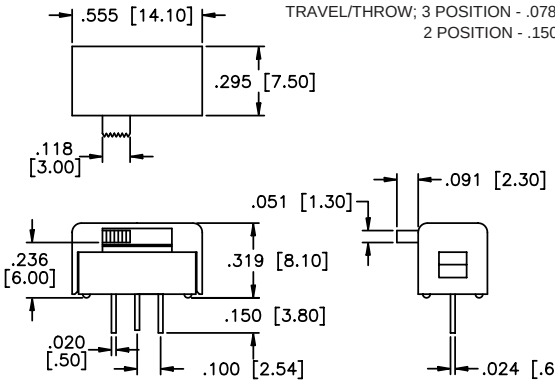
SINGLE POLE

Function
(see pg. E51)

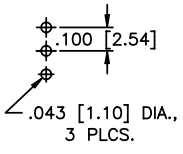


SCHEMATIC

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)



MECHANICAL OUTLINE



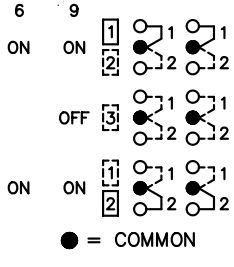
P.C. BOARD LAYOUT

MODEL NO.	PLATING	ACTION
25146NLDB	Gold	on - on
25149NLDB	Gold	on off on
25146NAB	Silver	on - on
25149NAB	Silver	on off on

VERTICAL ACTUATOR

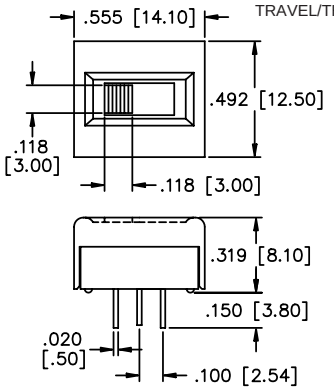
DOUBLE POLE
FLUSH ACTUATOR

Function
(see pg. E51)

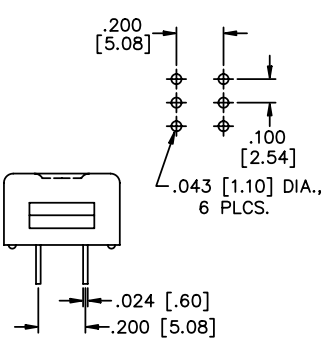


SCHEMATIC

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)



MECHANICAL OUTLINE



P.C. BOARD LAYOUT

25000N Series

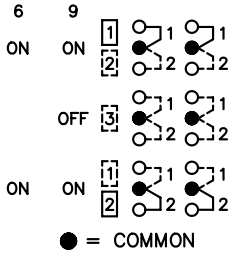
Miniature Slide Switches

MODEL NO.	PLATING	ACTION
25346NLD	Gold	on - on
25349NLD	Gold	on off on
25346NA	Silver	on - on
25349NA	Silver	on off on

RIGHT ANGLE ACTUATOR

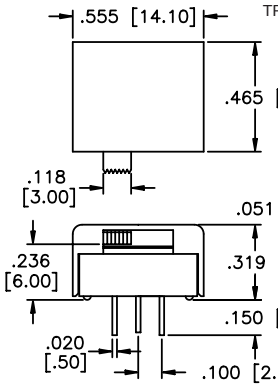
DOUBLE POLE

Function (see pg. E51)

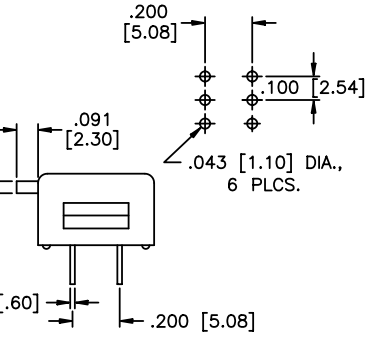


SCHEMATIC

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)



MECHANICAL OUTLINE



P.C. BOARD LAYOUT

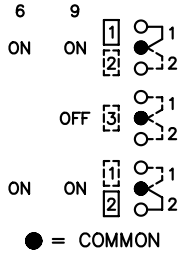
MODEL NO.	PLATING	ACTION
25436NLDB	Gold	on - on
25439NLDB	Gold	on off on
25436NAB	Silver	on - on
25439NAB	Silver	on off on

VERTICAL ACTUATOR

SINGLE POLE

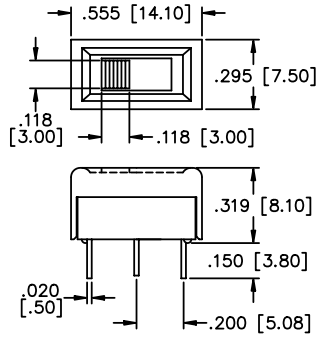
FLUSH ACTUATOR

Function (see pg. E51)

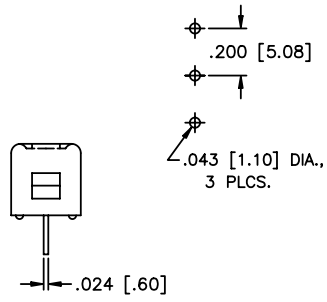


SCHEMATIC

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)



MECHANICAL OUTLINE



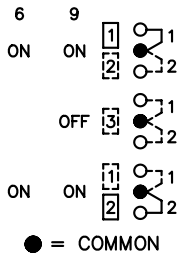
P.C. BOARD LAYOUT

MODEL NO.	PLATING	ACTION
25536NLD	Gold	on - on
25539NLD	Gold	on off on
25536NA	Silver	on - on
25539NA	Silver	on off on

RIGHT ANGLE ACTUATOR

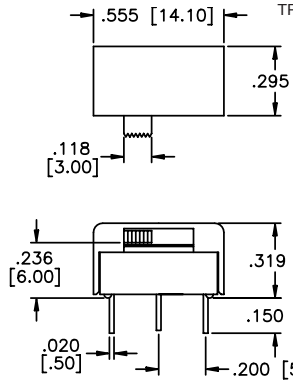
SINGLE POLE

Function (see pg. E51)

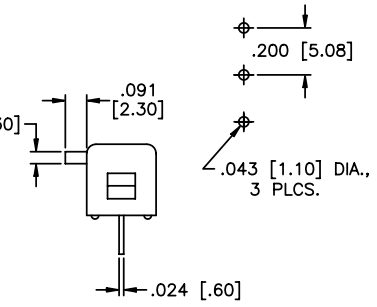


SCHEMATIC

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)



MECHANICAL OUTLINE



P.C. BOARD LAYOUT

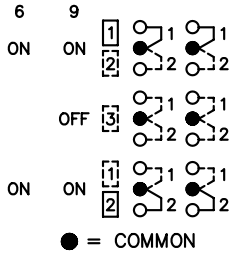
MODEL NO.	PLATING	ACTION
25446NLDB	Gold	on - on
25449NLDB	Gold	on off on
25446NAB	Silver	on - on
25449NAB	Silver	on off on

VERTICAL ACTUATOR

DOUBLE POLE

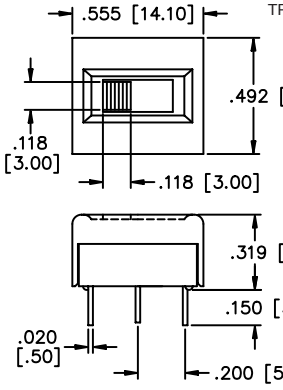
FLUSH ACTUATOR

Function (see pg. E51)

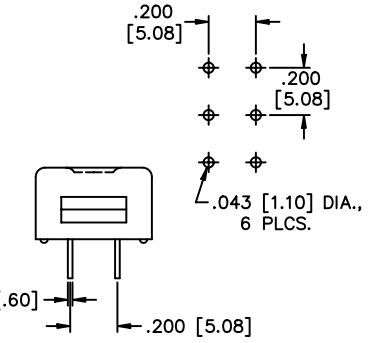


SCHEMATIC

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)



MECHANICAL OUTLINE



P.C. BOARD LAYOUT

E

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

, PO BOX 544, WAKEFIELD, MA USA 01880 TEL: (781) 246-1007 FAX: (781) 245-4531 TOLL FREE SALES: (877) 246-7890

25000N Series

Miniature Slide Switches

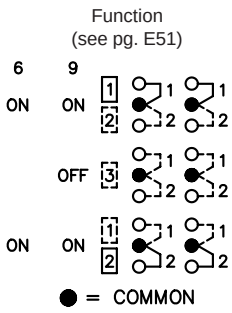
SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

E

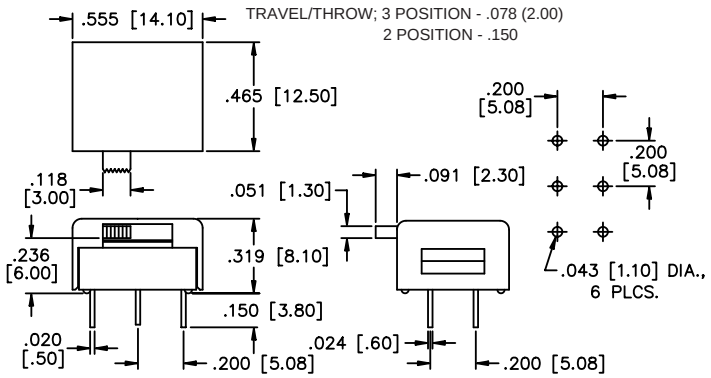
MODEL NO.	PLATING	ACTION
25546NLD	Gold	on - on
25549NLD	Gold	on off on
25546NA	Silver	on - on
25549NA	Silver	on off on

RIGHT ANGLE ACTUATOR

DOUBLE POLE



SCHEMATIC



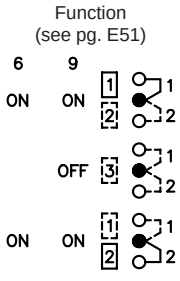
MECHANICAL OUTLINE

P.C. BOARD LAYOUT

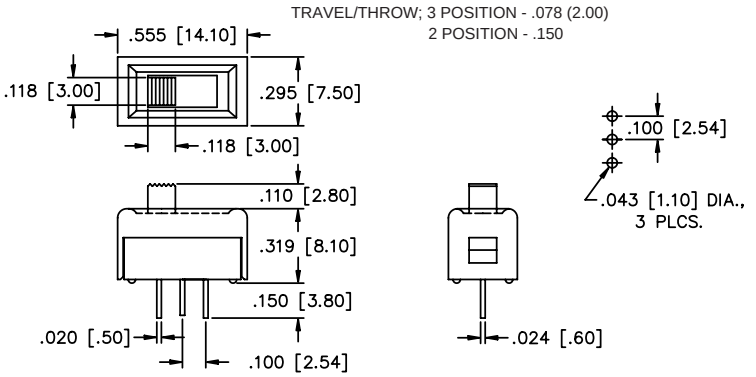
MODEL NO.	PLATING	ACTION
25136NLDH	Gold	on - on
25139NLDH	Gold	on off on
25136NAH	Silver	on - on
25139NAH	Silver	on off on

VERTICAL ACTUATOR

SINGLE POLE



SCHEMATIC



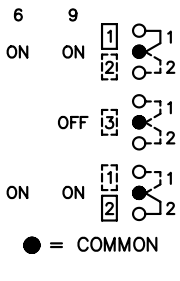
MECHANICAL OUTLINE

P.C. BOARD LAYOUT

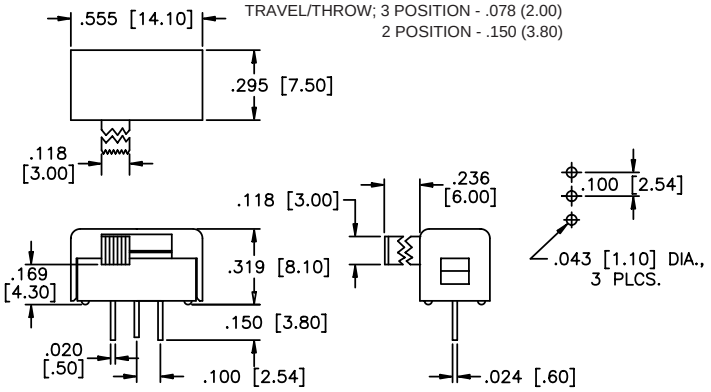
MODEL NO.	PLATING	ACTION
25336NLD6	Gold	on - on
25339NLD6	Gold	on off on
25336NA6	Silver	on - on
25339NA6	Silver	on off on

RIGHT ANGLE ACTUATOR

SINGLE POLE



SCHEMATIC



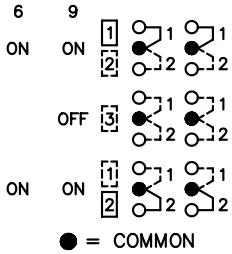
MECHANICAL OUTLINE

P.C. BOARD LAYOUT

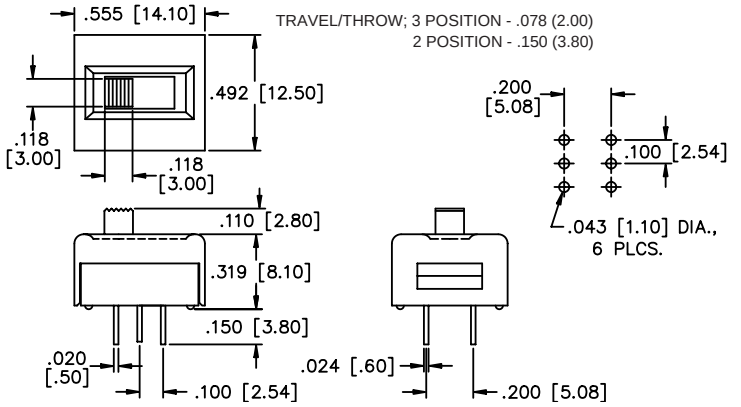
MODEL NO.	PLATING	ACTION
25146NLDH	Gold	on - on
25149NLDH	Gold	on off on
25146NAH	Silver	on - on
25149NAH	Silver	on off on

VERTICAL ACTUATOR

DOUBLE POLE



SCHEMATIC



MECHANICAL OUTLINE

P.C. BOARD LAYOUT

25000N Series

Miniature Slide Switches

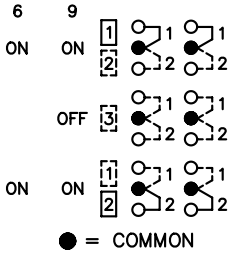
MODEL NO.	PLATING	ACTION
25346NLD6	Gold	on - on
25349NLD6	Gold	on off on
25346NA6	Silver	on - on
25349NA6	Silver	on off on

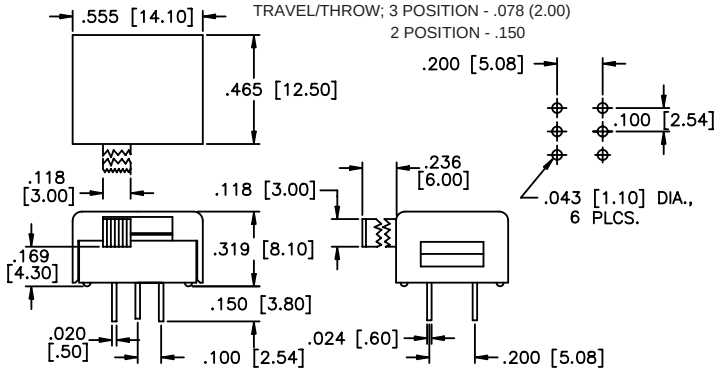
RIGHT ANGLE ACTUATOR

DOUBLE POLE

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150

Function
(see pg. E51)





SCHEMATIC MECHANICAL OUTLINE P.C. BOARD LAYOUT

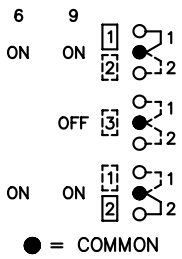
MODEL NO.	PLATING	ACTION
25436NLDH	Gold	on - on
25439NLDH	Gold	on off on
25436NAH	Silver	on - on
25439NAH	Silver	on off on

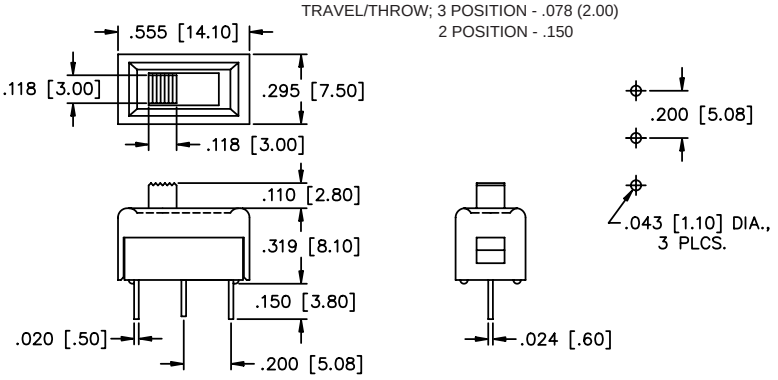
VERTICAL ACTUATOR

SINGLE POLE

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150

Function
(see pg. E51)





SCHEMATIC MECHANICAL OUTLINE P.C. BOARD LAYOUT

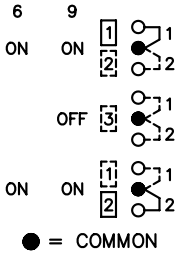
MODEL NO.	PLATING	ACTION
25536NLD6	Gold	on - on
25539NLD6	Gold	on off on
25536NA6	Silver	on - on
25539NA6	Silver	on off on

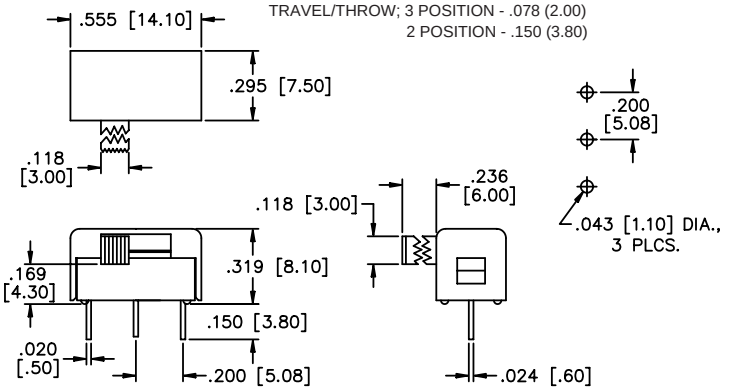
RIGHT ANGLE ACTUATOR

SINGLE POLE

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)

Function
(see pg. E51)





SCHEMATIC MECHANICAL OUTLINE P.C. BOARD LAYOUT

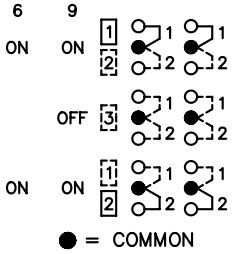
MODEL NO.	PLATING	ACTION
25446NLDH	Gold	on - on
25449NLDH	Gold	on off on
25446NAH	Silver	on - on
25449NAH	Silver	on off on

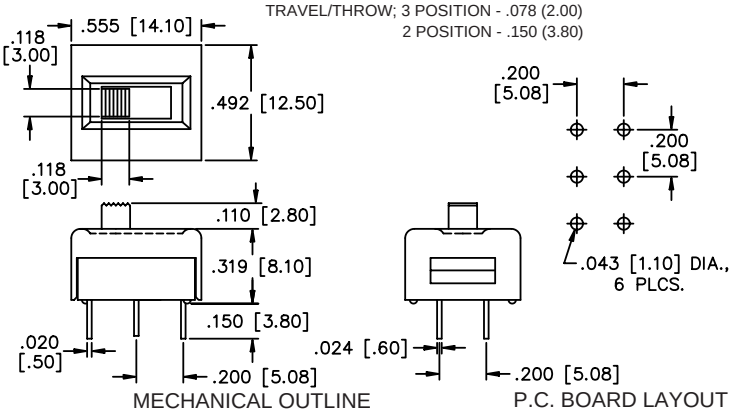
VERTICAL ACTUATOR

DOUBLE POLE

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)

Function
(see pg. E51)





SCHEMATIC MECHANICAL OUTLINE P.C. BOARD LAYOUT

E

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

NEM, PO BOX 544, WAKEFIELD, MA USA 01880 TEL: (781) 246-1007 FAX: (781) 245-4531 TOLL FREE SALES: (877) 246-7890

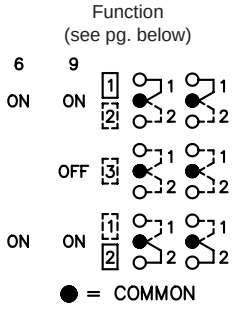
25000N Series

Miniature Slide Switches

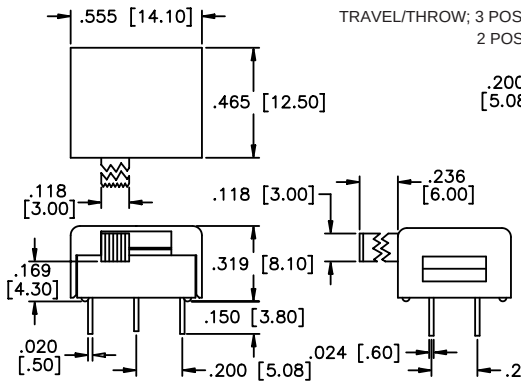
MODEL NO.	PLATING	ACTION
25546NLD6	Gold	on - on
25549NLD6	Gold	on off on
25546NA6	Silver	on - on
25549NA6	Silver	on off on

RIGHT ANGLE ACTUATOR

DOUBLE POLE

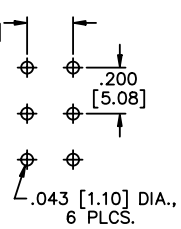


SCHEMATIC



MECHANICAL OUTLINE

TRAVEL/THROW; 3 POSITION - .078 (2.00)
2 POSITION - .150 (3.80)



P.C. BOARD LAYOUT

Additional ordering information: Commonly ordered models are shown above and on preceeding pages with complete model numbers and specifications. Below is an order format that enables you to 'build your own switch' and specify the correct switch model number by simply selecting from all available options shown and filling in the boxes.

'BUILD YOUR OWN SWITCH' ORDER FORMAT

Fill in boxes shown with options selected from below.
* Allowable options for each box are shown under each respective box.

Series	Actuator/ Term. Spacing	Poles	Functions	Version	Contact & Term. Mat'l.	Actuators	Approvals
2 5	1 4 3 5	3 4	6 9	N	A L D	B H H 6 6	(NONE) X 6 9 3 U L

* If a single suffix selection is chosen from the **Contact & Term. Mat'l.** or **Actuator** categories above, eliminate the extra box.

Actuator/Terminal spacing: Dimensions in inches and millimeters.
1 = Vertical actuator and .100" (2.54mm) terminal spacing
4 = Vertical actuator and .200" (5.08mm) terminal spacing
3 = Side actuator and .100" (2.54mm) terminal spacing
5 = Side actuator and .200" (5.08mm) terminal spacing

Poles: **3** = Single Pole **4** = Double Pole

Functions: **6** = ON - ON **9** = ON OFF ON

Version: **N** = Version specification (Specify for all models)

Contact & terminal materials: See contact ratings under "specifications". **A** = Silver **LD** = Gold plated brass

Actuators: Dimensions in inches and millimeters.
B = Flush vertical actuator
H = Vertical actuator .110" (2.80mm) high
H6 = Vertical actuator .236" (6.00mm) high
6 = Side actuator .236" (6.00mm) long
Note: If **Actuator** boxes are not used, a side actuator .090" (2.30mm) long will be supplied.

Approvals: **X693UL** = Switches recognized and marked UL - silver (A) contacts only.

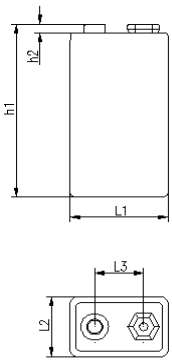
SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

E

			Conditions
cell type:		NiMH	
cell size:		E-block	
nominal voltage:	8.4	V	
max. charge voltage:	10.5	V	at standard charge (0.1C / 20℃)
capacity			
nominal:	300	mAh	discharge at 0.2C
minimum:	270	mAh	discharge at 0.2C
	240	mAh	discharge at 1C 7.0V end discharge voltage ta: 20℃
max. continuous discharge current:	400	mA	ta: 0....45℃
charge	current		time
standard charge:	30	mA	14....16hrs
quick charge:	85	mA	4hrs
fast charge:	300	mA	1.1hrs
recommended charge termination control parameters:	15....35	mV	- delta V
	120	%	of nominal input by timer cut off
trickle charge current:	2...15	mA	(recommended)
continuous overcharge: (less than 1 year)	≤ 30	mA	no conspicuous deformation no leakage
internal resistance: (impedance)	≤ 0.8	Ω	at 1KHz battery fully charged
life expectance:	≥ 500	cycles	acc. IEC standard
self discharge			
charge retention:	≥ 85	%	after 6 months storage
(at ≤ 20℃ ambient)	≥ 80	%	after 12 months storage
initial capacity:	≥200	mAh	within 30 days after delivery discharge at 0.2C
ambient temperature range:	0...45	℃	standard charge
	10...40	℃	fast charge
	0...45	℃	discharge (≥1C)
	- 20...65	℃	discharge (<1C)
	- 20...50	℃	storage (≤3months)
	- 20...40	℃	storage (≤6months)
	- 20...30	℃	storage (≤24months)
QCT1:	20/260/750		
QCT2:	30/240/800		

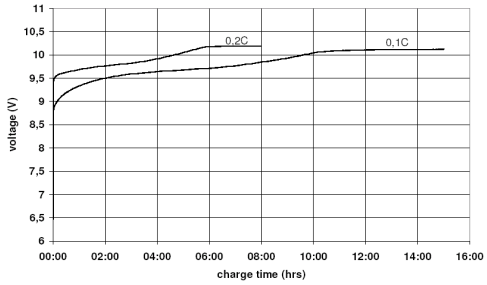
mechanical specifications

cell dimensions (incl. label)			
Length L1:	26.5	- 2.0	mm
Length L2:	17.5	- 2.0	mm
Length L3:	12.7	± 0.25	mm
height h1:	48.5	- 1.0	mm
weight:	45	± 4	g
blister card dimensions:	123	x 85	mm
blister card weight (incl. battery):		53	g

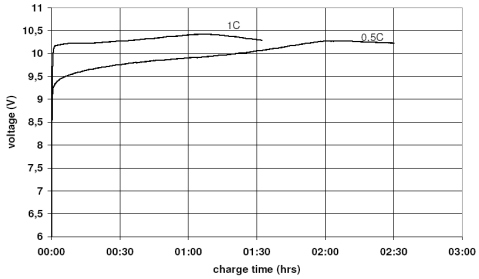


Diagrams

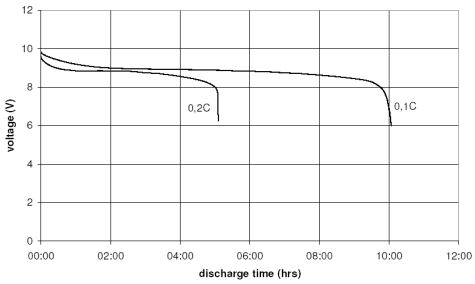
standard charge



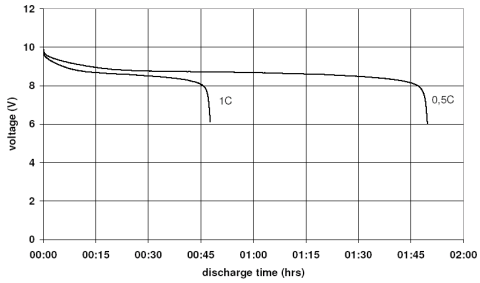
fast charge



low rate discharge

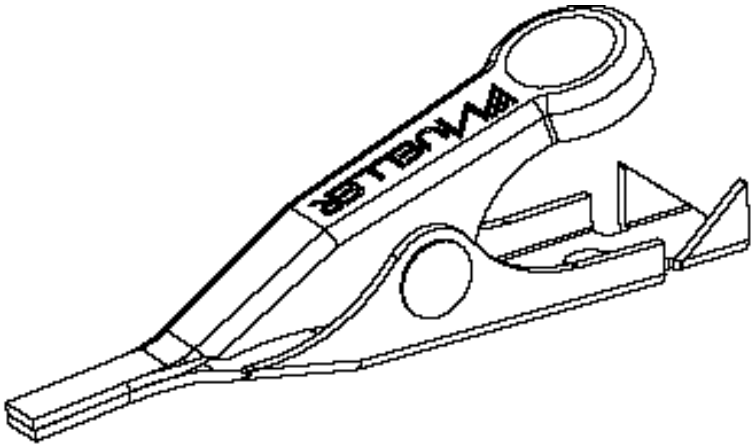


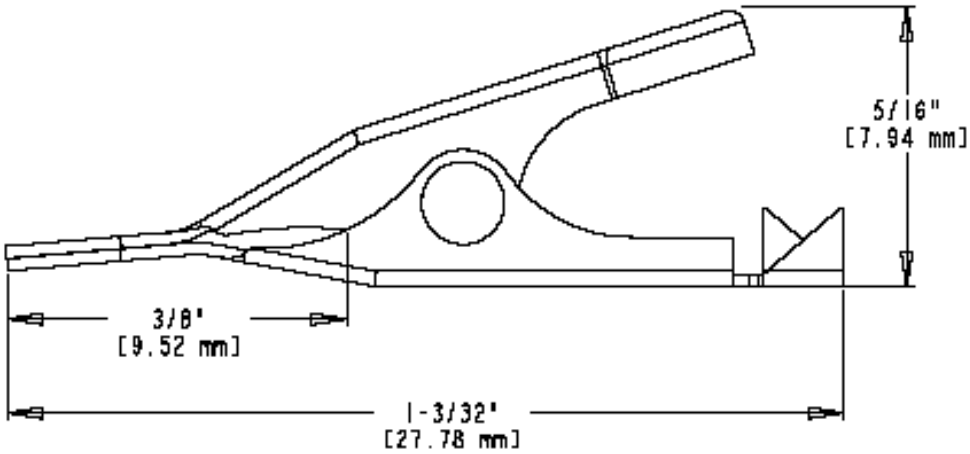
high rate discharge



	ANSMANN Specifications for model:	9V-E-block 300mAh low self discharge 1pc blister package
ANSMANN®	data sheet no. / part no.	5035453
	supplier no.	701364
	author / date	Gramlich / 05.03.2012

Part Number: **BU-34/BU-34C**
Description: **Micro-Alligator**
3107





- Micro-Alligator
- Solder or crimp connection
- Jaw opens 0.22” (5.6 mm) max
- Material: nickel-plated steel (BU-34) or solid copper (BU-34C)
- Length: 1.09” (27.78 mm)
- Rating: 5 Amps
- RoHS Compliant

Mueller Electric Company
1625 E. 31st Street
Cleveland, Ohio 44114

ISO 9001:2000
Over 95 Years Of Innovation
www.muellerelectric.com

TEL. 800-955-2629
TEL. 216-771-5225
FAX. 216-771-3068

MJTP SERIES 6MM THRU-HOLE MTG. TACT SWITCHES

FEATURES

- Numerous single pole-single throw momentary configurations.
- Excellent tactile feed-back (Snap dome).
- Ultra-compact size.
- Long operating life for high reliability.
- Molded-in terminals minimize wicking of flux or solder.

MATERIALS

Gold plating available on contacts & terminals - consult factory.

Case:	Thermoplastic, color black. UL94HB Standard.
Actuator:	Thermoplastic
Cover/Bracket:	PET film, tin plated brass or stainless steel.
Terminals:	Silver plated brass.
Shorting Contact:	Phosphor bronze or stainless steel with silver plating.
L.E.D. rating:	Forward voltage- 2.1 V (3.0 V Max.). Cont. fwd. current 20mA max.@ 25°C.

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

GENERAL SPECIFICATIONS

ELECTRICALS

Contact rating:	50 mA @ 12 VDC
Insulation resistance:	100 megohms min. (100 VDC)
Dielectric withstanding voltage:	250 VAC for 1 minute
Contact resistance:	100 milliohms max. @ 5 VDC, 10 mA
Electrical life:	100,000 cycles min. (except models MJTP1243 & 1250- 50,000 cycles) 1,000,000 cycles available - consult factory
Contact bounce:	less than 10 msec.

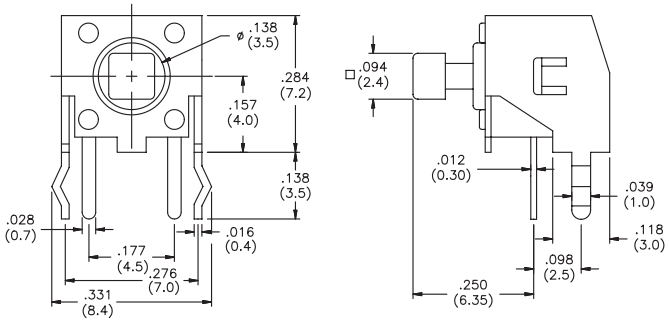
MECHANICALS, THERMALS, ENVIRONMENTALS

Plunger travel:	.010"+ .008", - .004" (0,25 +0,2, -0,1 mm)
Actuation force:	160 grams ±30 grams
Operating temperature range:	-20°C to +70°C
Storage temperature range:	-30°C to +80°C for 96 hours
Shock resistance:	30G per method 213, MIL-STD-202
Vibration resistance:	passes method 201, MIL-STD-202

SOLDERING (note - non-washable except for models indicated 'process compatible' below).

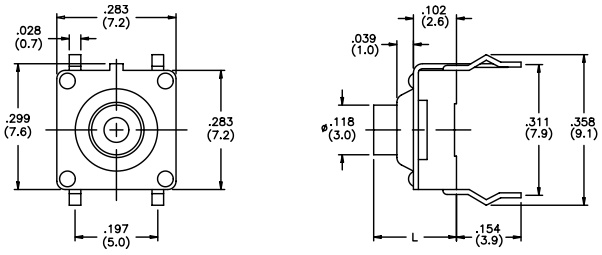
IR reflow soldering:	240°C max. for 20 seconds max.
Wave soldering:	255°C max. for 5 seconds max.
Hand soldering:	320°C max. for 3.5 seconds max. (40 watt iron max.)

MJTP1105T
Right angle, grounding



Caps available - see end of section. Bulk packaged.

US # MJTP1109
EU # PHAP3350
Process compatible, grounding



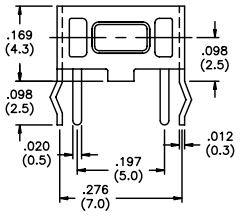
US Model No.	Dim. L (In./mm)	EU Model No.
MJTP1109	.197 (5.0)	PHAP3350
MJTP1109A	.354 (9.0)	PHAP3350A
MJTP1109B	.512 (13.0)	PHAP3350B

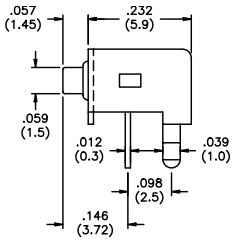
Bulk packaged.

M

US # MJTP1117
EU # PHAP3363
Right angle, grounding

New!

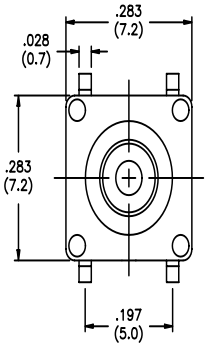


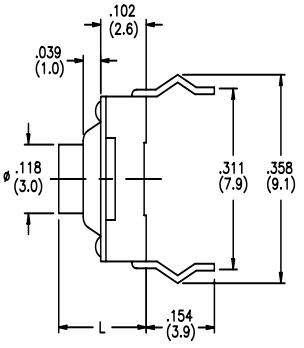


Bulk packaged.

US # MJTP1119
EU # PHAP3351
Process compatible

New!

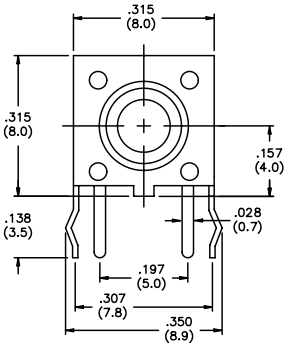


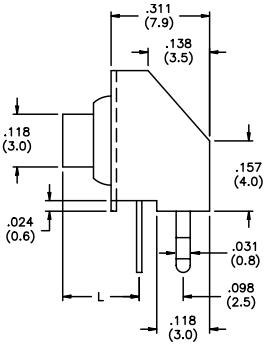


US Model No.	Dim. L (in./mm)	EU Model No.
MJTP1119	.197 (5.0)	PHAP3351
MJTP1119A	.354 (9.0)	PHAP3351A
MJTP1119B	.512 (13.0)	PHAP3351B

Bulk packaged.

US # MJTP1129
EU # PHAP1152
Right angle, grounding, process compatible

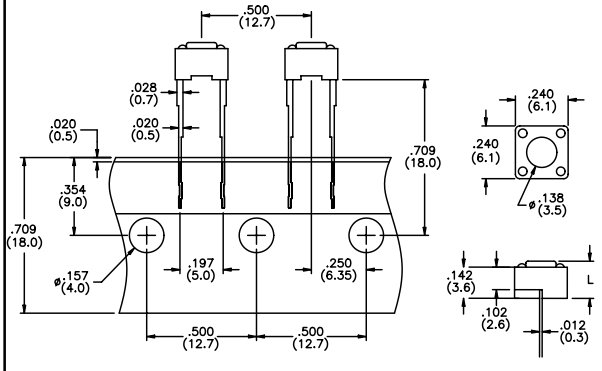




US Model No.	Dim. L (in./mm)	EU Model No.
MJTP1129	.171 (4.35)	PHAP3352
MJTP1129A	.329 (8.35)	PHAP3352A
MJTP1129B	.486 (12.35)	PHAP3352B

Bulk packaged.

US # MJTP1141
EU # PHAP3307
Long terminals



US Model No.	Dim. L (in./mm)	EU Model No.
MJTP1141	.169 (4.3)	PHAP3307
MJTP1141A	.197 (5.0)	PHAP3307A

Packaged on tape. Note: other models available with this tape packaging - consult factory.

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

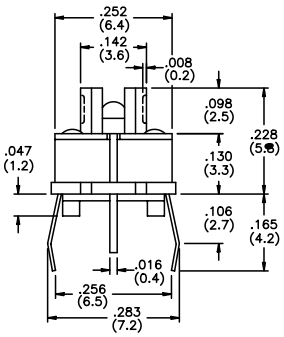
M

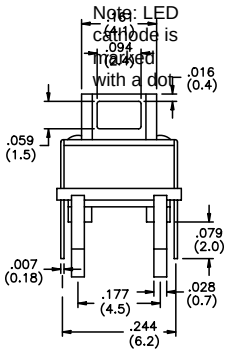
APEM, 63 NECK ROAD, HAVERHILL, MA USA 01835-0788 TOLL FREE: (877) 246-7890 FAX: (978) 372-3534 E-mail: info@apem.com

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

US # MJTP1193
EU # PHAP3308
With L.E.D.

New!

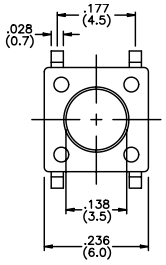


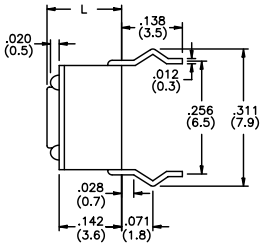


US Model No.	L.E.D. Color	EU Model No.
MJTP1193	No L.E.D.	PHAP3308
MJTP11936	Red	PHAP3308R
MJTP11935	Yellow	PHAP3308Y
MJTP11933	Green	PHAP3308G

Caps available with lens - see end of section.
Bulk packaged.

US # MJTP1230
EU # PHAP3301



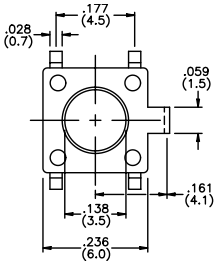


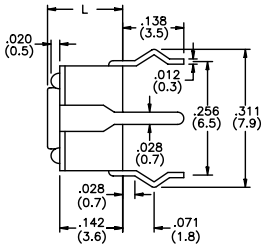
US Model No.	Dim. L (in./mm)	EU Model No.
MJTP1230	.170 (4,3)	PHAP3301
MJTP1230A	.197 (5,0)	PHAP3301A
MJTP1230B	.374 (9,5)	PHAP3301B
MJTP1230C	.315 (8,0)	PHAP3301C
MJTP1230D	.512 (13,0)	PHAP3301D
MJTP1230E	.287 (7,3)	PHAP3301E
MJTP1230F	.335 (8,5)	PHAP3301F
MJTP1230G	.275 (7,0)	N/A

Bulk packaged.

M

US # MJTP1232
EU # PHAP3302
Grounding

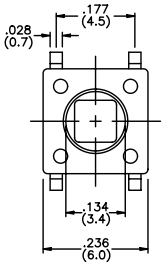


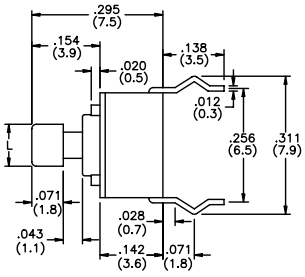


US Model No.	Dim. L (in./mm)	EU Model No.
MJTP1232	.170 (4,3)	PHAP3302
MJTP1232A	.197 (5,0)	PHAP3302A
MJTP1232B	.374 (9,5)	PHAP3302B
MJTP1232C	.315 (8,0)	PHAP3302C
MJTP1232D	.512 (13,0)	PHAP3302D
MJTP1232E	.287 (7,3)	PHAP3302E
MJTP1232F	.335 (8,5)	PHAP3302F
MJTP1232G	.275 (7,0)	PHAP3302G

Bulk packaged.

US # MJTP1234
EU # PHAP3303

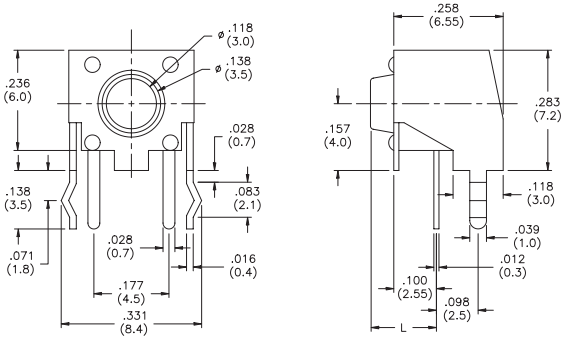




US Model No.	Dim. L (in./mm)	EU Model No.
MJTP1234	.094 (2,4)	PHAP3303
MJTP1234A	.110 (2,8)	PHAP3303A

Caps available - see end of section.
Bulk packaged.

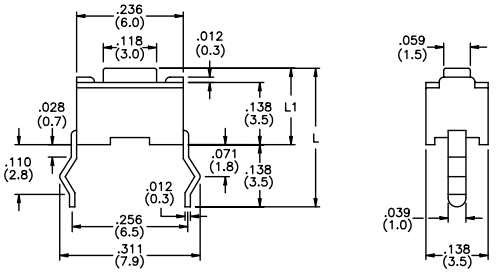
US # MJTP1236
EU # PHAP3305
Right angle, grounding



US Model No.	Dim. L (in./mm)	EU Model No.
MJTP1236	.124 (3,1)	PHAP3305
MJTP1236A	.152 (3,9)	PHAP3305A
MJTP1236B	.329 (8,4)	PHAP3305B
MJTP1236C	.270 (6,9)	PHAP3305C
MJTP1236D	.466 (11,9)	PHAP3305D
MJTP1236E	.244 (6,2)	PHAP3305E
MJTP1236F	.289 (7,3)	N/A
MJTP1236G	.230 (5,8)	N/A

Bulk packaged.

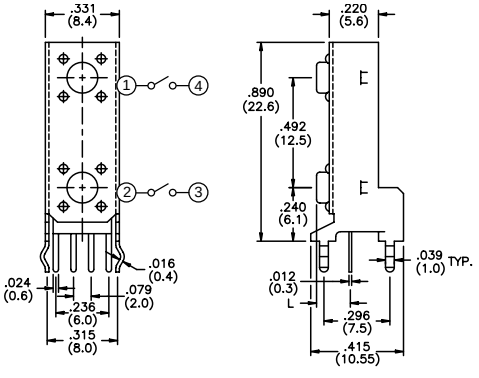
US # MJTP1243 &
MJTP1250



US # Model No.	Dim. L1(in./mm)	Dim. L (in./mm)
MJTP1243	.169 (4,3)	.307 (7,8)
MJTP1250	.197 (5,0)	.335 (8,5)

Bulk packaged.

US # MJTP2205
EU # PHAP3306
2 switch vertical gang assembly, black actuators



US Model No.	Dim. L (in./mm)	EU Model No.
MJTP2205	.124 (3,15)	PHAP3306
MJTP2205A	.152 (3,85)	PHAP3306A
MJTP2205B	.329 (8,35)	PHAP3306B
MJTP2205C	.270 (6,85)	PHAP3306C
MJTP2205D	.466 (11,85)	PHAP3306D
MJTP2205E	.242 (6,15)	PHAP3306E

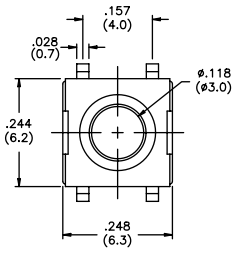
Bulk packaged.

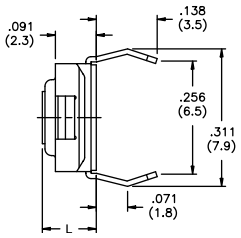
SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

M

APEM, 63 NECK ROAD, HAVERHILL, MA USA 01835-0788 TOLL FREE: (877) 246-7890 FAX: (978) 372-3534 E-mail: info@apem.com

US # MJTPSHW
Process compatible



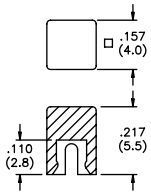


US # Model No.	Dim. L (in./mm)
MJTPSHW	.122 (3,1)
MJTPSHWA	.091 (2,3)
MJTPSHWB	.146 (3,8)

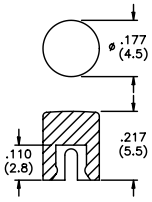
Bulk packaged.

CAPS (for MJTP1105T and MJTP1234 only)

MJ61 Square cap

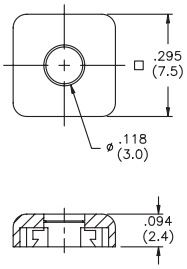


MJ62 Round cap



CAPS (for MJTP1193 only) New!

MJ3 Square cap (w/clear lens in center for LED illumination)



CAP COLORS

Note - add digit from table to cap part number for color. Example - **MJ612** indicates square black cap.

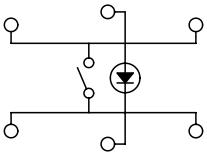
No.	Color
1	blue
2	black
3	green
5	yellow
6	red
7	ivory

SCHEMATICS

MJTP1109
MJTP1119
MJTP1193
MJTP1230
MJTP1232

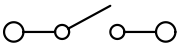
MJTP1234
MJTP2205
MJTP4102
MJTP5302
MJTPSHW

Note: LED is for MJTP1193 only.



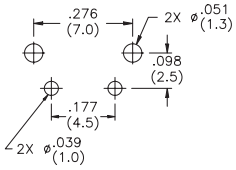
MJTP1105
MJTP1117
MJTP1129
MJTP1141
MJTP1144
MJTP1145

MJTP1236
MJTP1243
MJTP1250
MJTP5302

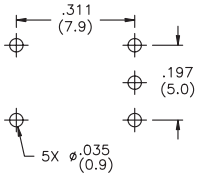


PRINTED CIRCUIT BOARD LAYOUTS

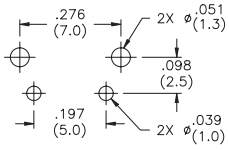
Layout tolerances; Linear: ± .004" (± 0,1MM) Holes: ± .002" (± 0,05MM)



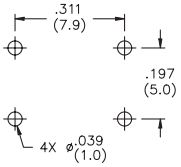
MJTP1105T



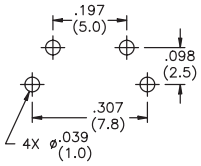
MJTP1109



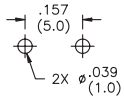
MJTP1117



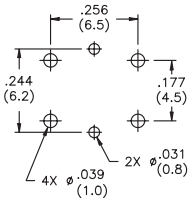
MJTP1119



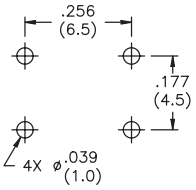
MJTP1129



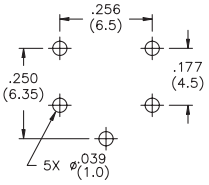
MJTP1141, 44 & 45



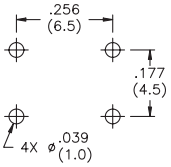
MJTP1193



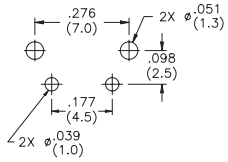
MJTP1230



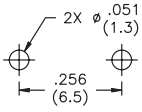
MJTP1232



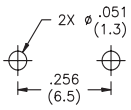
MJTP1234



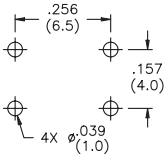
MJTP1236



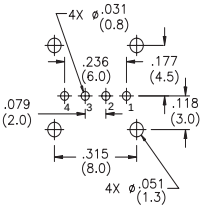
MJTP1243



MJTP1250

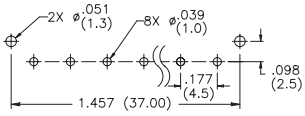


MJTPSHW

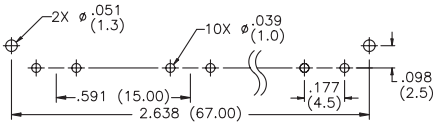


MJTP2205

MJTP4102



MJTP5302



SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

M

SHIELDED 9 VOLT BATTERY STRAPS

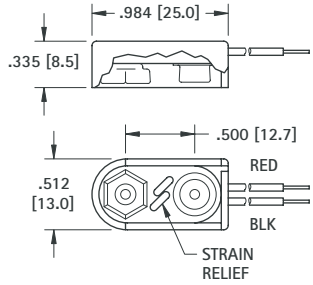
- Insulates and shields 9 Volt battery snap-on connector contacts
- Assures safe secure use, reduces the potential of short circuits and prevents tampering with battery contacts
- Protects circuits, prevents unwanted electrical discharge due to unintended grounding
- Protects battery contacts from dust, dirt, and physical damage
- Ideal for rugged applications, made of high impact molded plastic with integral wire lead strain relief

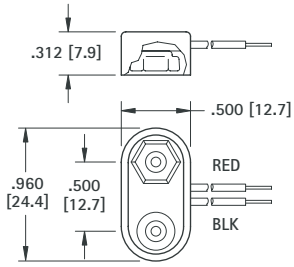
Modifications available with different wire lengths.

84-4
84-6
84-8

SPECIFICATIONS
Base: ABS
Male/Female Contacts: Brass, Nickel Plate
Wires (Red & Black): #26 AWG, Tinned, .187 (4.8) end strip, UL/CSA1007

*Tolerance on lead length +/-.250 (6.4)





81-4
81-6
81-8

I STYLE MOLDED		T STYLE MOLDED
CAT. NO.	*LEAD LENGTH	CAT. NO.
84-4	4.00 (101.6)	81-4
84-6	6.00 (152.4)	81-6
84-8	8.00 (203.2)	81-8

PREMIUM 9 VOLT BATTERY STRAPS

- Fully Assembled • Rigid Assembly • Ideal for rugged applications, made of high impact molded plastic with integral wire lead strain relief

Rigidly constructed. Nickel plate, male and female contacts. Female contacts are spring temper. Accepts numerous insertions to meet U.L. approved standards. Provides excellent mechanical strength and low contact resistance. Wires have tinned copper conductors, PVC insulation and the ends are stripped for easy connection.

Wires are rated to 176°F (80°C) and 300V and meet UL/CSA #1061/#1007 requirements.

Modifications available with different wire lengths.

2238
2239
2240

Base Size:
1.062 (27.0) × .512 (13.0)

SPECIFICATIONS
Contacts: Phosphor Bronze, Nickel Plate
Wire: #26 AWG, 7/34
End Strip: .250 (6.4) Long, Tinned
Base: Polyethylene

*Tolerance on lead length +/-.250 (6.4)

I STYLE MOLDED		T STYLE MOLDED
CAT. NO.	*LEAD LENGTH	CAT. NO.
2238	4.00 (101.6)	2241
2239	6.00 (152.4)	2242
2240	8.00 (203.2)	2243

2241
2242
2243

Base Size:
1.00 (25.4) × .512 (13.0)

ECONOMY 9 VOLT BATTERY STRAPS

- Fully Assembled • Rigid Assembly

When product cost is of the utmost importance these economy battery straps will suit your needs. Brass contacts, nickel plated with vinyl or ABS covering. Wires have tinned copper conductors, PVC insulation and the ends are stripped for easy connection. Wires meet UL/CSA #1007 requirements.

Modifications available with different wire lengths

232-M
233-M
234-M

Molded ABS Base Size:
.984 (25.0) × .469 (11.9)
(Strain Relief)

SPECIFICATIONS
Contacts: Brass, Nickel Plate
Wire: #26 AWG, 7/34
End Strip: .250 (6.4) Long, Tinned

*Tolerance on lead length +/-.250 (6.4)

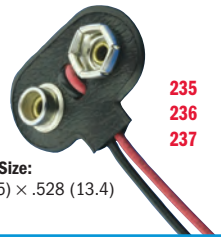
235-M
236-M
237-M

Molded ABS Base Size:
.965 (25.0) × .469 (11.9)
(Strain Relief)

232
233
234

Vinyl Base Size:
1.125 (31.8) × .530 (13.5)

I STYLE			T STYLE	
MOLDED	VINYL		MOLDED	VINYL
CAT. NO.	CAT. NO.	*LEAD LENGTH	CAT. NO.	CAT. NO.
232-M	232	4.00 (101.6)	235-M	235
233-M	233	6.00 (152.4)	236-M	236
234-M	234	8.00 (203.2)	237-M	237



235
236
237

Vinyl Base Size:
1.000 (24.5) × .528 (13.4)

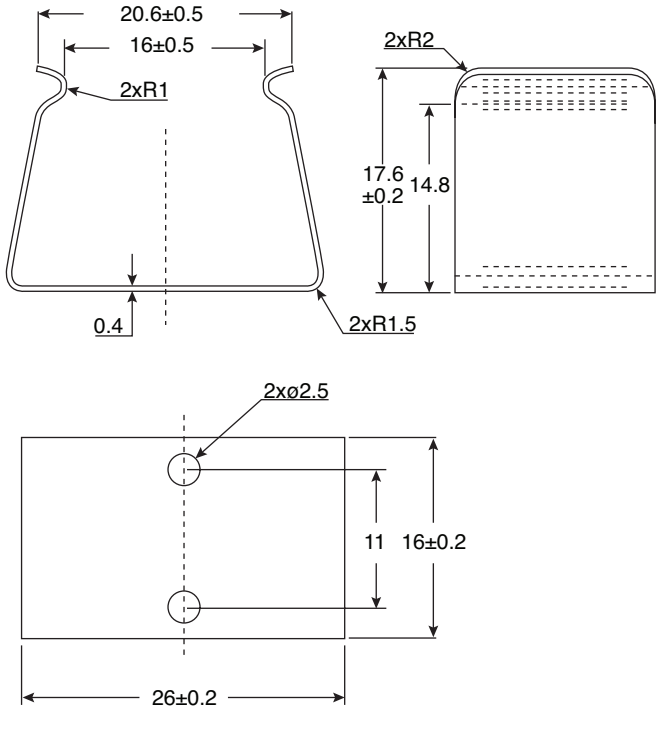
Tel (718) 956-8900 • Fax (718) 956-9040
(800) 221-5510 • kec@keyelco.com

www.keyelco.com
31-07 20th Road – Astoria, NY 11105-2017

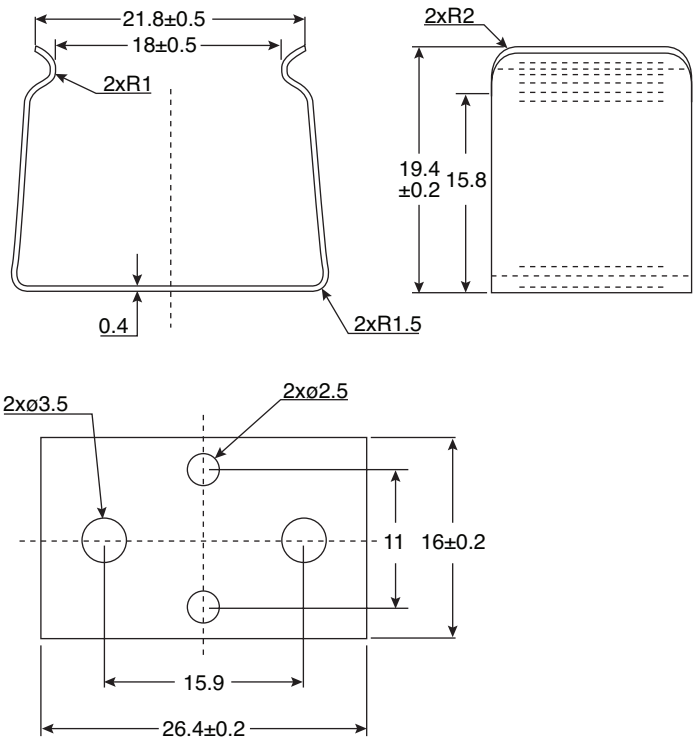
9V Horizontal Battery Clip

12BH071-GR, 12BH079-GR

Date Last Revised: 6-16-06



12BH071-GR



12BH079-GR

Dimensions: mm (In.)

SPECIFICATIONS	
Material:	Steel
RoHS Compliant	

Available from Mouser Electronics
1-800-346-6873
www.mouser.com

EPD-200423

CTS Electrocomponents

Series 206RA/ 208RA Data Sheet
Right Angle Panel DIP Switch

Right Angle Panel DIP Switches

Features

Optimized Mechanical Features

- Designed for panel applications
  Saves cost of right angle socket
- Multiple circuit configurations available
- Standard 2.54mm (.100") x 2.54mm (.100") centers, uses minimum PCB real estate
- Sealed version optional: assures contaminant-free switch operation after wave soldering and cleaning
- Available with low profile, standand, or extended actuators for SPST circuitry

Optimized Contact System

- Contact wiping action on make and break
- Integral terminal and contact locked into high temperature plastic base
- Electrostatic discharge shield rated to 22 KV

Materials

- Series 206RA, gold plated contacts for long-term contact corrosion resistance
- Series 208RA, tin plated contacts
- UL 94V-0 plastics
- RoHS compliant

Series 206RA – Premium Gold Plated Contacts
Series 208RA – Economical Tin Plated Contacts

Electrical and Mechanical Specifications

Switch Function

SPST - 2 through 10 and 12 positions
SPDT&DPST- 1 through 6 positions
DPDT&4PST - 1 through 3 positions
3PST - 1 through 4 positions

Switch Contact Resistance

Switch Series	Initial, max.	End of life , max.
206RA	50 milliohms	100 milliohms
208RA	100 milliohms	500 milliohms

Insulation Resistance

1,000 megohms minimum between insulated terminals

Dielectric Strength

500 VAC for 1 minute between adjacent switches

Nonswitching Rating

100 mA or 50 VDC maximum

Switch Capacitance

5.0 pF maximum between adjacent closed switches

Operating Temperature

–55°C to +85°C

Actuation Life

Series 206RA: 10,000 cycles switching 50 mA @ 24 VDC
Series 208RA: 2,000 cycles switching 50 mA @ 24 VDC

Allowable Solder Time

Wave soldering up to 5 seconds at 260°C

Vibration

Per MIL-STD-202F, method 204D, condition B (.06" or 15 G's between 10 HZ to 2K HZ) with no contact inconsistencies greater than 1 microsecond

Shock

Per MIL-STD-202F, method 213B, condition A (50 G's) with no contact inconsistencies greater than 1 microsecond

Sealing

Bottom epoxy seal optional
Top tape seal optional

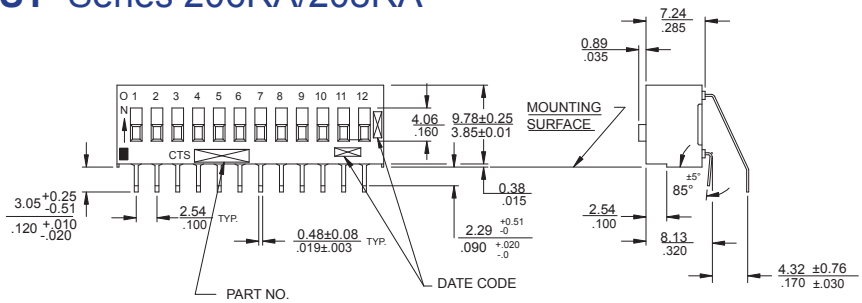
Marking

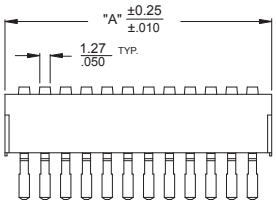
Special side or top marking available-consult CTS

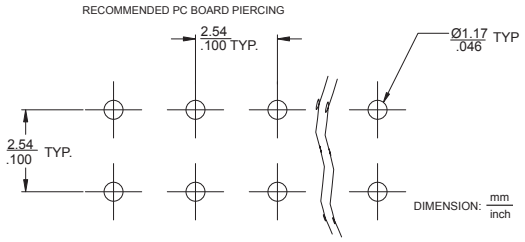
Packaging

Standard : Anti-Static tube packaging

SPST Series 206RA/208RA







Please find panel drawing of 206/208 RA DIP switches with SPDT,DPST,DPDT,3PST, or 4PST circuitry from 206/208 series data sheet

Ordering Information

206/208 – 1 2 2 RA LP S T N

NUMBER OF POLES
- **BLANK – (FOR SPST)**
- **1** – SINGLE POLE
- **2** – DOUBLE POLES
- **3** – 3 POLES
- **4** – 4 POLES

NUMBER OF THROWS
- **BLANK – (FOR SPST)**
- **1** – SINGLE THROW
- **2** – DOUBLE THROWS

NUMBER OF SWITCH POSITIONS
- **1** – 1 POSITION
- **2** – 2 POSITIONS
- **3** – 3 POSITIONS
- **4** – 4 POSITIONS
- **5** – 5 POSITIONS
- **6** – 6 POSITIONS
- **7** – 7 POSITIONS
- **8** – 8 POSITIONS
- **9** – 9 POSITIONS
- **10** – 10 POSITIONS
- **12** – 12 POSITIONS

ACTUATOR HEIGHT
- **BLANK** – STANDARD(0.89mm/.035")
- * **E** – EXTENDED(3.43mm/.135")
- ** **LP** – LOW PROFILE(Flush)

* Extended: SPST only
** Low profile: SPST & SPDT only

BOTTOM EPOXY SEAL
- **BLANK** - NO EPOXY SEAL
- **S** - EPOXY SEAL

TOP TAPE SEAL
- **BLANK** – NO TOP TAPE SEAL
- ***T** – TOP TAPE SEAL

* Not available on extended actuator

ACTUATION PRESET
- **BLANK** – SHIP AT "OFF" POSITION
- **N** – SHIP AT "ON" POSITION

Standard Part #

Dimensions	SPST Switches		SPDT&DPST Switches			3PST Switches		DPDT/ 4PST Switches		
"A" Overall (mm/inch)	No of SW positions	SPST P/N	No of SW positions	SPDT P/N	DPST P/N	No of SW positions	3PST P/N	No of SW positions	DPDT P/N	4PST P/N
7.26 / .286	2	206/208- 2 RA	1	206/208-121RA	206/208-211RA	—	—	—	—	—
9.80 / .386	3	206/208- 3 RA	—	—	—	1	206/208-311RA	—		—
12.34 / .486	4	206/208- 4 RA	2	206/208-122RA	206/208-212RA	—	—	1	206/208-221RA	206/208-411RA
14.88 / .586	5	206/208- 5 RA	—	—	—	—	—	—	—	—
17.42 / .686	6	206/208- 6 RA	3	206/208-123RA	206/208-213RA	2	206/208-312RA	—	—	—
19.96 / .786	7	206/208- 7 RA	—	—	—	—	—	—	—	—
22.50 / .886	8	206/208- 8 RA	4	206/208-124RA	206/208-214RA	—	—	2	206/208-222RA	206/208-412RA
25.04 / .986	9	206/208- 9 RA	—	—	—	3	206/208-313RA	—	—	—
27.58 / 1.086	10	206/208-10 RA	5	206/208-125RA	206/208-215RA	—	—	—	—	—
32.66 / 1.286	12	206/208-12 RA	6	206/208-126RA	206/208-216RA	4	206/208-314RA	3	206/208-223RA	206/208-413RA

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

CTS:

206-121RAST 206-122RAST 206-123RAST 206-124RAST 206-125RAST 206-126RAST 206-211RAST 206-212RAST 206-213RAST 206-215RAST 206-216RAST 206-221RAST 206-222RAST 206-223RAST 206-411RAST 206-412RAST 206-413RAST 206-2RAS 206-2RA 206-3RAS 206-3RA 206-4RAS 206-4RA 206-5RAS 206-5RA 206-6RAS 206-6RA 206-7RAS 206-7RA 206-7ES 206-8RAS 206-8RA 206-9RAS 206-9RA 206-10RAS 206-10RA 206-12RAS 206-12RA 206-121RAS 206-121RA 206-122RAS 206-122RA 206-123RAS 206-123RA 206-124RAS 206-124RA 206-125RAS 206-125RA 206-126RAS 206-126RA 206-211RAS 206-211RA 206-212RAS 206-212RA 206-213RAS 206-213RA 206-214RAS 206-214RA 206-215RAS 206-215RA 206-216RAS 206-216RA 206-221RAS 206-221RA 206-222RAS 206-222RA 206-223RAS 206-223RA 206-312RAST 206-312RAS 206-312RA 206-411RAS 206-411RA 206-412RAS 206-412RA 206-413RAS 206-413RA 206-10RAST 206-12RAST 206-214RAST 206-2RAST 206-3RAST 206-4RAST 206-5RAST 206-6RAST 206-7RAST 206-8RAST 206-9RAST

PICDEM™ 2 Plus
User's Guide

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Trademarks

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip received ISO/TS-16949:2002 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company’s quality system processes and procedures are for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Table of Contents

NOTES:

Chapter 1. Introduction

1.1 WELCOME

Thank you for purchasing the PICDEM 2 Plus demonstration board from Microchip Technology Incorporated. The PICDEM 2 Plus is a simple board which demonstrates the capabilities of the 18, 28 and 40-pin PIC16 and PIC18 devices.

The PICDEM 2 Plus can be used stand-alone with a programmed part, with an in-circuit emulator (e.g., MPLAB® ICE) or with an in-circuit debugger (e.g., MPLAB ICD 2). Sample programs are provided to demonstrate the unique features of the supported devices.

The PICDEM 2 Plus Kit comes with the following:

1. PICDEM 2 Plus Demonstration Board (Figure 1-1)
2. Sample devices
3. CD-ROM, which contains:
   a) Sample programs
   b) PICDEM 2 Plus Demonstration Board User's Guide
   c) Application Notes

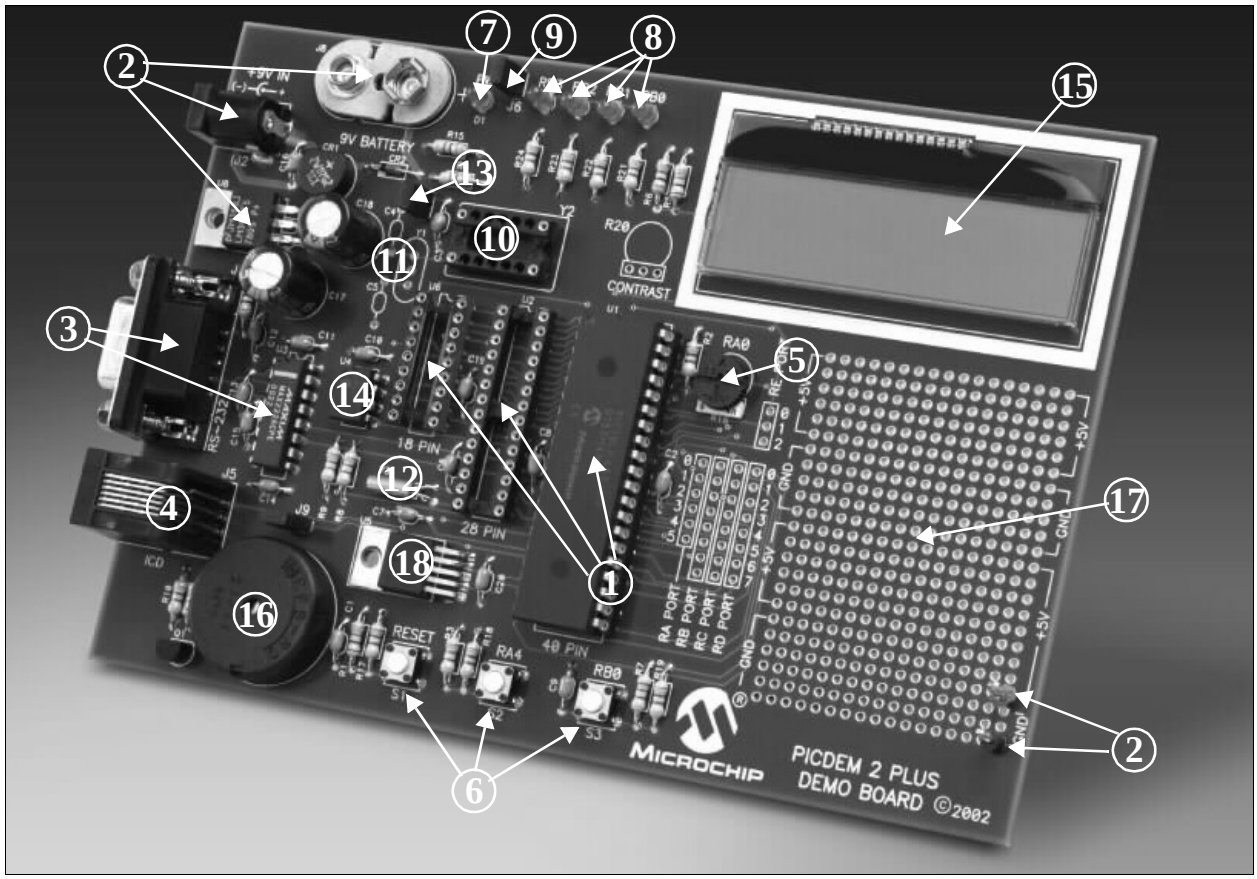
If you are missing any part of the kit, please contact your nearest Microchip sales office listed in the back of this publication for help.

1.2 PICDEM 2 PLUS DEMONSTRATION BOARD

The PICDEM 2 Plus demonstration board has the following hardware features:

1. 18, 28 and 40-pin DIP sockets. (Although three sockets are provided, only one device may be used at a time.)
2. On-board +5V regulator for direct input from 9V, 100 mA AC/DC wall adapter or 9V battery, or hooks for a +5V, 100 mA regulated DC supply.
3. RS-232 socket and associated hardware for direct connection to an RS-232 interface.
4. In-Circuit Debugger (ICD) connector.
5. 5 KΩ pot for devices with analog inputs.
6. Three push button switches for external stimulus and Reset.
7. Green power-on indicator LED.
8. Four red LEDs connected to PORTB.
9. Jumper J6 to disconnect LEDs from PORTB.
10. 4 MHz canned crystal oscillator.
11. Unpopulated holes provided for crystal connection.
12. 32.768 kHz crystal for Timer1 clock operation.
13. Jumper J7 to disconnect on-board RC oscillator (approximately 2 MHz).
14. 32K x 8 Serial EEPROM.
15. LCD display.
16. Piezo buzzer.
17. Prototype area for user hardware.
18. Microchip TC74 thermal sensor.

FIGURE 1-1: PICDEM 2 PLUS HARDWARE



1.3 SAMPLE DEVICES

Two FLASH devices are included. The device types may change, but will generally include PIC16 and PIC18 40-pin DIP devices.

1.4 SAMPLE PROGRAMS

The PICDEM 2 Plus Kit includes a CD-ROM with sample demonstration programs. These programs may be used with the included sample devices, with an In-Circuit Emulator (ICE) or with an In-Circuit Debugger (ICD). For each type of device (PIC16 or PIC18), demo source code (several ASM files) and compiled code (one Hex file) are provided.

1.5 PICDEM 2 PLUS USER'S GUIDE

This document describes the PICDEM 2 Plus demonstration board, tutorial and demonstration software. Detailed information on individual microcontrollers may be found in the device's respective data sheet. Detailed information on In-Circuit Emulator (ICE) or In-Circuit Debugger (ICD) systems may be found in the respective tool's user guide.

Chapter 1: Introduction – This chapter introduces the PICDEM 2 Plus and provides a brief description of the hardware.

Chapter 2: Getting Started – This chapter goes through a basic step-by-step process for getting your PICDEM 2 Plus up and running as a stand-alone board or with an ICE or ICD.

Chapter 3: Tutorial – This chapter provides a detailed description of the tutorial program.

Appendix A: Hardware Description: This appendix describes in detail the hardware of the PICDEM 2 Plus board.

1.6 REFERENCE DOCUMENTS

Reference Documents may be obtained by contacting your nearest Microchip sales office (listed in the back of this document) or by downloading via the Microchip web site (www.microchip.com).

- Individual Data Sheets and Reference Manuals:
  - *PIC16F87X Data Sheet* (DS30292)
  - *PIC18FXX2 Data Sheet* (DS39564)
  - *PICmicro® Mid-Range MCU Family Reference Manual* (DS33023)
  - *PICmicro® 18C MCU Family Reference Manual* (DS39500)
  - *TC74 Data Sheet* (DS21462)
- *MPLAB® IDE Simulator, Editor User's Guide* (DS51025)
- *MPASM™ User's Guide with MPLINK™ Linker and MPLIB™ Librarian* (DS33014)
- *PRO MATE® II User's Guide* (DS30082)
- *PICSTART® Plus User's Guide* (DS51028)
- *MPLAB® ICE User's Guide* (DS51159)
- *MPLAB® ICD 2 Quick Start Guide* (DS51268)
- *Microchip Third Party Guide* (DS00104)

Chapter 2. Getting Started

The PICDEM 2 Plus may be used as a stand-alone board with a preprogrammed device, with an In-Circuit Emulator (ICE) or with an In-Circuit Debugger (ICD). For a list of PICmicro microcontroller compatible ICEs or ICDs, please refer to the *Development Systems Ordering Guide* or the *Microchip Third Party Guide.*

2.1 PICDEM 2 PLUS AS A STAND-ALONE BOARD – PREPROGRAMMED DEVICE

The PICDEM 2 Plus may be demonstrated immediately by following the steps listed below:

- Place the preprogrammed sample device in the appropriate socket on the PICDEM 2 Plus board.
- Place a jumper on J6 (to enable the LEDs).
- Verify that the board is set up for a 4 MHz canned oscillator (i.e., no jumper on J7; a 4 MHz oscillator in Y2; Y1, C4 and C5 are empty).
- Apply power to the PICDEM 2 Plus. For information on acceptable power sources, see Appendix A.

To reprogram the sample device, the following will be necessary:

1. Program source code.

   User source code may be used to program the device or, if this has previously been done, the sample program may be restored from the file on the included CD-ROM.

2. An assembler, such as MPASM™ assembler (available with MPLAB IDE), or a compiler, such as MPLAB C18 (PIC18 devices only).

   Source code must be assembled or compiled into a Hex file before it can be programmed into the device. Microchip Technology's MPASM assembler or MPLAB C18 C compiler may be used. Both are compatible with MPLAB IDE. However, other assemblers/compilers may be used. For a list of these PICmicro® MCU compatible language tools, please refer to the *Microchip Third Party Guide.*

3. A device programmer, such as PRO MATE® II, MPLAB® PM3[(1)], PICSTART® Plus or MPLAB® ICD 2 (programmer functionality available with MPLAB IDE v6.00 or greater).

   Once the sample program is in Hex file format, a programmer may be used to program a Flash device. Microchip Technology's PRO MATE II device programmer, PICSTART Plus development programmer or MPLAB ICD 2 may be used. All are compatible with MPLAB IDE. However, other programmers may be used. For a list of these PICmicro MCU compatible programmers, please refer to the *Microchip Third Party Guide.*

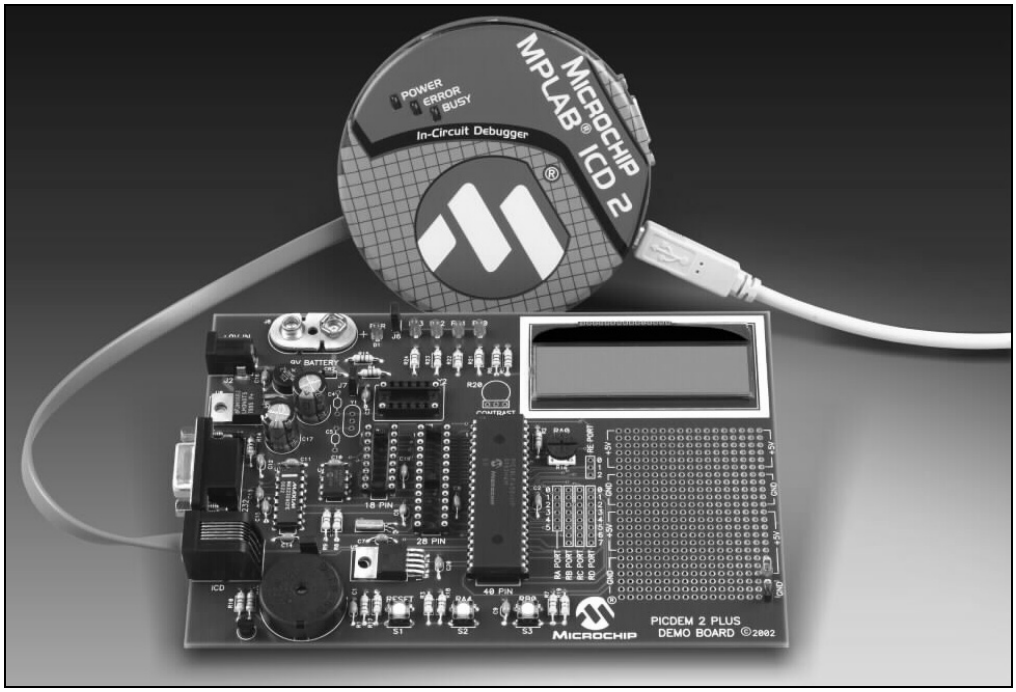
   If the code protection bit(s) have not been programmed, the on-chip program memory can be read out for verification purposes.

Note 1: The MPLAB PM3 device programmer will be available in Q2 2004. Check the Microchip web site (www.microchip.com) for further information.

2.2 PICDEM 2 PLUS USED WITH AN IN-CIRCUIT EMULATOR OR IN-CIRCUIT DEBUGGER

To use PICDEM 2 Plus with an In-Circuit Emulator (ICE) or In-Circuit Debugger (ICD), refer to the tool's user guide for instructions on how to power-up and configure the ICE/ICD, as well as how to connect to target boards (e.g., Figure 2-1).

FIGURE 2-1: PICDEM 2 PLUS CONNECTED TO MPLAB ICD 2 USING USB



Configure the PICDEM 2 Plus for the desired oscillator as described in Table 2-1. Refer to the ICE/ICD user's guide for any oscillator configuration requirements.

TABLE 2-1: OSCILLATOR SELECTION

Oscillator Selection on PICDEM 2 Plus	Modification on PICDEM 2 Plus
RC	J7 installed, Y2 empty, Y1 empty
Crystal	J7 removed, Y2 empty, crystal in Y1, caps in C4 and C5
Canned Oscillator	J7 removed, oscillator in Y2 (Y1, C4, C5 empty)
Resonator – no internal caps	J7 removed, Y2 empty, resonator in Y1, caps in C4 and C5
Resonator – with internal caps	J7 removed, Y2 empty, resonator in Y1, C4 and C5 empty

Chapter 3. Tutorial

The tutorial program is preprogrammed into the sample device, (i.e., `p16demo.hex` for a PIC16 device and `p18demo.hex` for a PIC18 device). Also, this program is on the included CD-ROM program disk for user reference, (i.e., if the sample device has been reprogrammed with another program, the tutorial may be reprogrammed into the device).

For detailed information on the PICDEM 2 Plus hardware, please refer to Appendix A.

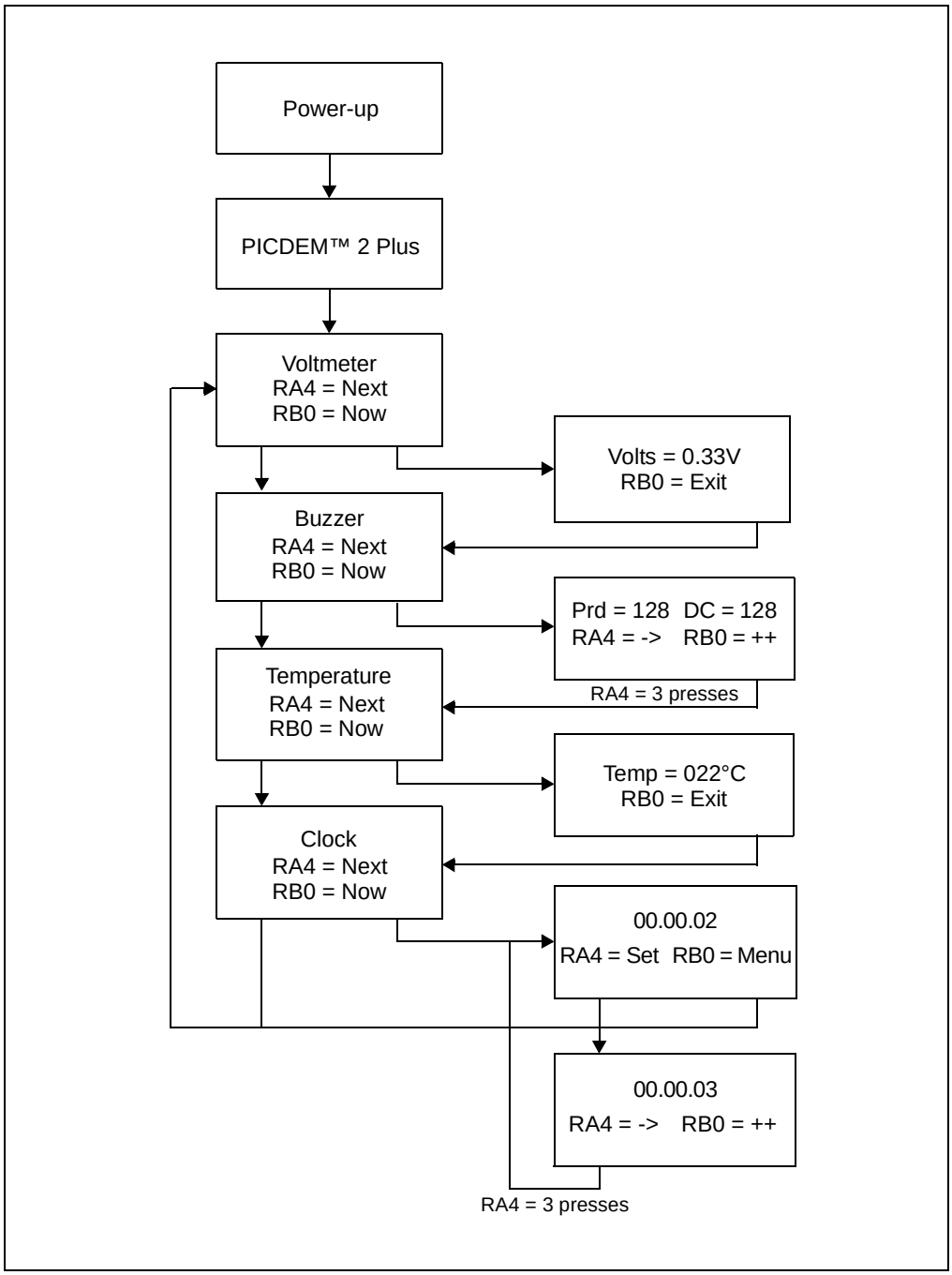
3.1 TUTORIAL PROGRAM OPERATION

The tutorial program is made up of four components, which are individually displayed on the LCD.

1. **Voltmeter**
   This mode uses the A/D module to measure the voltage of the R16 pot and display a voltage between 0.00V and 5.00V on the LCD. Voltage is continually updated until the mode is exited by pressing RB0.
2. **Buzzer**
   This mode turns on the Piezo buzzer, using the CCP1 module I/O pin, RC2. The period and duty cycle of the CCP1 frequency can be changed while the buzzer is on. The changes in period and duty cycle are recognized immediately in the buzzer tone. To change the period and/or the duty cycle, press RB0 under the "Buzzer" menu. The buzzer will then sound off with the default setting of 80h for the period and duty cycle. The cursor will flash over the period's first digit, indicating that the PR2 register is ready to be incremented. To change the duty cycle, press RA4 once and the cursor will now flash over the duty cycle's first digit, indicating it is now ready to increment the CCPR1L register. The next press of RA4 will exit the buzzer function.
3. **Temperature**
   This mode uses a TC74 thermal sensor to measure ambient temperature in Celsius and then display that temperature on the LCD. Communication between the PICmicro MCU and sensor is accomplished using the MSSP module. This mode is exited by pressing RB0. This mode contains code that will write to the external on-board EEPROM. Every two seconds, the code will write to a defined EEPROM address and store the current temperature in that address.
4. **Clock**
   Once this mode is entered from the main menu, a real-time clock will start counting from 00:00:00. The Timer1 module and a 32 kHz clock crystal are used to establish a real-time clock. By pressing RA4, the clock time can be set to the user's preference. When RA4 is pressed to set the time, the cursor will flash over the hours ten digit. Press RA4 again and the cursor will now flash over the minutes ten digit. RB0 is used to increment hours and minutes whenever the cursor is flashing over either. After the minutes have been set, press RA4 and the time will be set and the LCD is returned to an active clock display.

The data that is sent to the LCD is also sent to the RS-232 serial port using the USART on the PICmicro MCU. A HyperTerminal™ program on the PC will be able to display the same information that is displayed on the LCD

FIGURE 3-1: TUTORIAL PROGRAM FLOW CHART



3.2 SOURCE CODE AND APPLICATION NOTES

In addition to the assembled tutorial program (Hex files), source code used to create these Hex files is included on the PICDEM 2 Plus CD-ROM. Both source code and related Hex files are found in device-specific directories.

Application Notes are also included on the CD-ROM for additional examples of use.

For information on how to reprogram the device with new or modified code, or how to restore the tutorial program, please see **Section 2.1 "PICDEM 2 Plus as a Stand-Alone Board – Preprogrammed Device"**.

NOTES:

Appendix A. Hardware Detail

The PICDEM 2 Plus hardware is extremely simple and is intended to illustrate the ease of use of various PICmicro MCUs. The PICDEM 2 Plus features the following hardware elements:

A.1 PROCESSOR SOCKETS

Although three sockets are provided, only one device may be used at a time.

- 18-pin socket
- 28-pin socket
- 40-pin socket

A.2 DISPLAY

Four red LEDs are connected to PORTB of each processor type. The PORTB pins are set high to light the LEDs. These LEDs may be disconnected from PORTB by removing jumper J6.

One green LED is provided to determine whether there is power to the PICDEM 2 Plus board (LED on) or not (LED off).

A.3 POWER SUPPLY

There are three ways to supply power to the PICDEM 2 Plus:

- A 9V battery can be plugged into J8.
- A 9V, 100 mA unregulated AC or DC supply can be plugged into J2. A power supply can be purchased through Microchip, Part #AC162039.
- A +5V, 100 mA regulated DC supply can be connected to the hooks provided.

Note: The PICDEM 2 Plus kit does not include a power supply.

MPLAB ICE 2000 users have a regulated +5V power supply available in the logic probe connector and can easily connect to the hooks on PICDEM 2 Plus (Red probe to +5V and Black probe to GND).

MPLAB ICD 2 users may use the ICD to power the target board to 5V, up to 200 mA, if the MPLAB ICD 2 is connected to the PC with a serial cable.

A.4 RS-232 SERIAL PORT

An RS-232 level shifting IC has been provided with all necessary hardware to support connection of an RS-232 host through the DB9 connector. The port is configured as DCE and can be connected to a PC using a straight-through cable.

The PIC16/PIC18 RX and TX pins are tied to the RX and TX lines of the MAX232A.

A.5 SWITCHES

Three switches provide the following functions:

- S1 – $\overline{\text{MCLR}}$ to hard reset the processor
- S2 – Active-low switch connected to RA4
- S3 – Active-low switch connected to RB0

Switches S1 and S3 have debounce capacitors, whereas S2 does not, allowing the user to investigate debounce techniques.

When pressed, the switches are grounded. When Idle, they are pulled high (+5V).

A.6 OSCILLATOR OPTIONS

- RC oscillator (2 MHz approximately) supplied. This oscillator may be disabled by removing jumper J7.
- Pads provided for user furnished crystal and two capacitors.
- Removable 4 MHz canned oscillator.
- 32.768 kHz (watch type) crystal for Timer1.

A.7 ANALOG INPUT

A 5 KΩ potentiometer is connected through a series 470 ohm resistor to AN0.

The pot can be adjusted from V_{DD} to GND to provide an analog input to the parts with an A/D module.

A.8 ICD CONNECTOR

By way of the modular connector (J5), the MPLAB ICD 2 can be connected for low-cost debugging. The ICD connector utilizes RB6 and RB7 of the microcontroller for in-circuit debugging.

A.9 TEMPERATURE SENSOR

This is a serial digital thermal sensor (TC74) connected to the 28 and 40-pin microcontrollers via RC3 and RC4. Communication is accomplished with the TC74 via it's 2-wire I^2C™ compatible serial port. This device has an address of `1001101b`.

A.10 SERIAL EEPROM

A 24L256 256K (32K x 8) serial EEPROM is included on the board to illustrate I^2C bus concepts.

A.11 LCD

An LCD display with two lines, 16 characters each, is connected to the 28 and 40-pin sockets. There are three control lines (RA3:RA1) and four data lines (RD3:RD0).

A 5 KΩ pot may be installed into R20 to adjust contrast on the LCD. If this is done, R5 and R6 need to be removed.

A.12 SAMPLE DEVICES

A sample part programmed with a simple program is included in the PICDEM 2 Plus kits.

Table A-1 lists the I/O features and port connections for each processor type.

TABLE A-1: PORT CONNECTIONS

Device	LEDs	RS-232	S1	S2	S3	Pot R16	LCD	EEPROM	Buzzer	ICD	Temp Sensor	Y1/Y2
18-pin	RB3:RB0	N/A	$\overline{\text{MCLR}}$	RA4	RB0	RA0	N/A	N/A	N/A	RB6/RB7	N/A	Yes
28-pin	RB3:RB0	RC6/RC7	$\overline{\text{MCLR}}$	RA4	RB0	RA0	RA3:RA1	RC3/RC4	RC2	RB6/RB7	RC3/RC4	Yes
40-pin	RB3:RB0	RC6/RC7	$\overline{\text{MCLR}}$	RA4	RB0	RA0	RA3:RA1 RD3:RD0	RC3/RC4	RC2	RB6/RB7	RC3/RC4	Yes

A.13 BOARD LAYOUT AND SCHEMATICS

The following figures show the parts layout (silkscreen) and schematics for the PICDEM 2 Plus board.

FIGURE A-1: PICDEM 2 PLUS PARTS LAYOUT

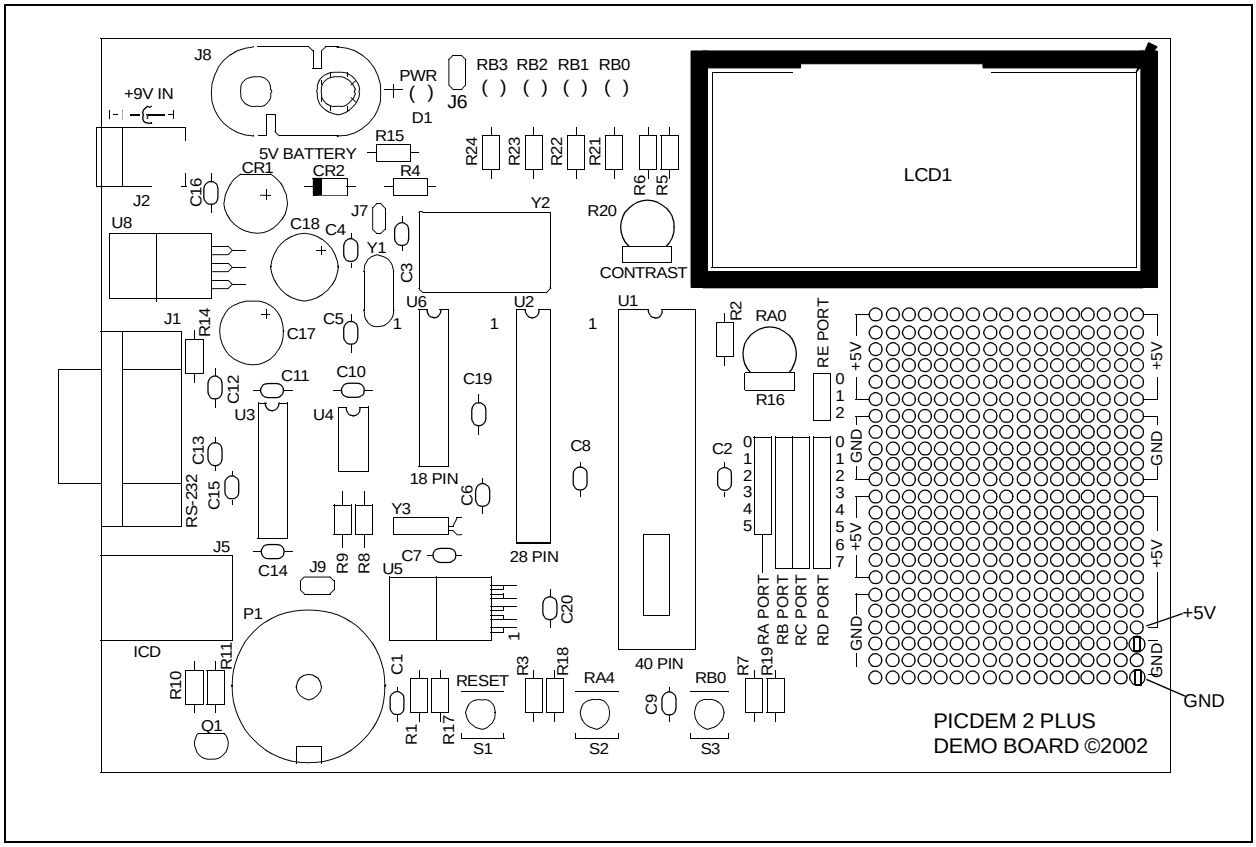


FIGURE A-2: PICDEM 2 PLUS SCHEMATIC

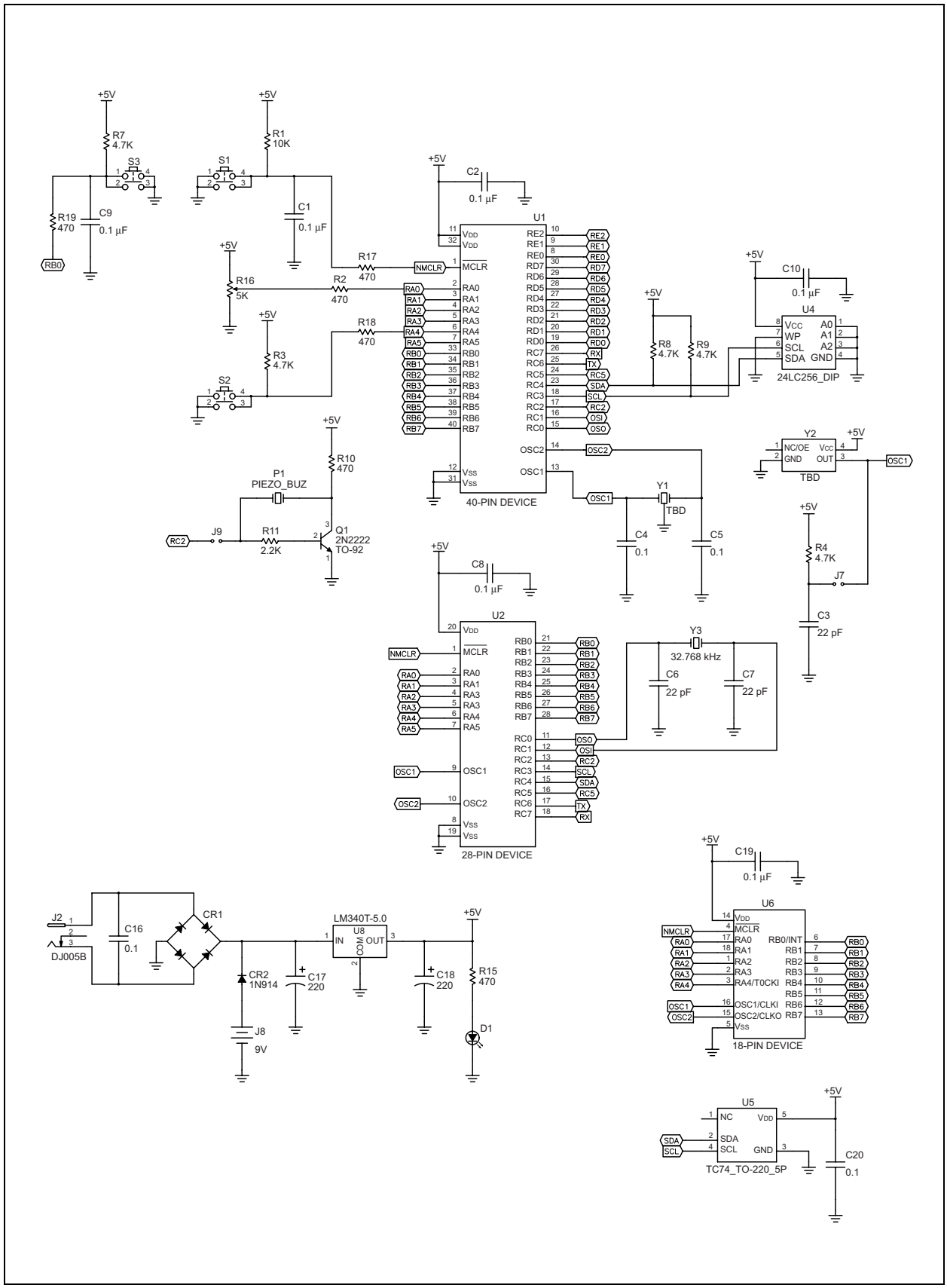
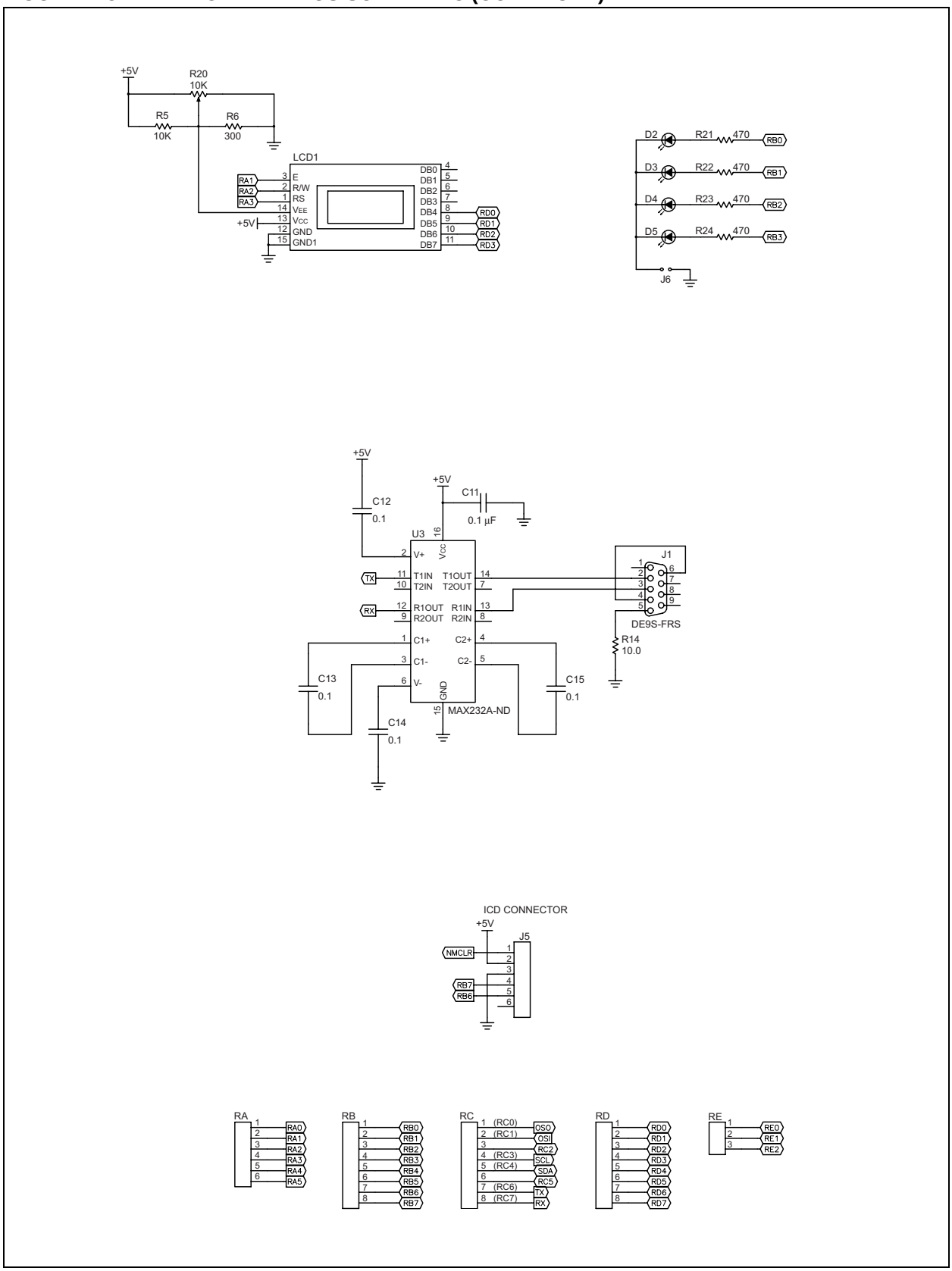


FIGURE A-3: PICDEM 2 PLUS SCHEMATIC (CONTINUED)



Index

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

Atlanta
3780 Mansell Road, Suite 130
Alpharetta, GA 30022
Tel: 770-640-0034
Fax: 770-640-0307

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848
Fax: 978-692-3821

Chicago
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
2767 S. Albright Road
Kokomo, IN 46902
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888
Fax: 949-263-1338

San Jose
1300 Terra Bella Avenue
Mountain View, CA 94043
Tel: 650-215-1444
Fax: 650-961-0286

Toronto
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Unit 706B
Wan Tai Bei Hai Bldg.
No. 6 Chaoyangmen Bei Str.
Beijing, 100027, China
Tel: 86-10-85282100
Fax: 86-10-85282104

China - Chengdu
Rm. 2401-2402, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200
Fax: 86-28-86766599

China - Fuzhou
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506
Fax: 86-591-7503521

China - Hong Kong SAR
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Shanghai
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700
Fax: 86-21-6275-5060

China - Shenzhen
Rm. 1812, 18/F, Building A, United Plaza
No. 5022 Binhe Road, Futian District
Shenzhen 518033, China
Tel: 86-755-82901380
Fax: 86-755-8295-1393

China - Shunde
Room 401, Hongjian Building, No. 2
Fengxiangnan Road, Ronggui Town, Shunde District, Foshan City, Guangdong 528303, China
Tel: 86-757-28395507 Fax: 86-757-28395571

China - Qingdao
Rm. B505A, Fullhope Plaza,
No. 12 Hong Kong Central Rd.
Qingdao 266071, China
Tel: 86-532-5027355 Fax: 86-532-5027205

India
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5932 or 82-2-558-5934

Singapore
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-6334-8870 Fax: 65-6334-8850

Taiwan
Kaohsiung Branch
30F - 1 No. 8
Min Chuan 2nd Road
Kaohsiung 806, Taiwan
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan
Taiwan Branch
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Austria
Durisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45-4420-9895 Fax: 45-4420-9910

France
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany
Steinheilstrasse 10
D-85737 Ismaning, Germany
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy
Via Quasimodo, 12
20025 Legnano (MI)
Milan, Italy
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands
P. A. De Biesbosch 14
NL-5152 SC Drunen, Netherlands
Tel: 31-416-690399
Fax: 31-416-690340

United Kingdom
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44-118-921-5869
Fax: 44-118-921-5820

01/26/04