upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa

MEMORIA

PROYECTO FIN DE CARRERA

"AUTOMATIZACIÓN DE UN SISTEMA DE RIEGO POR GOTEO MEDIANTE PLATAFORMA ARDUINO"

Departamento de Ingeniería Eléctrica y Electrónica

Alumno: Gorka Echarte Vidaurre
Tutor: Vicente Senosiáin Miquélez
Pamplona, 12 de enero 2012

ÍNDICE

1. INTRODUCCIÓN

En este proyecto se ha diseñado un dispositivo automático para gestionar los tiempos de riego óptimos aplicado a una pequeña extensión agrícola de aproximadamente 100 de superficie. Actualmente en el mercado hay un vacio entre los sistemas de riego profesional (agricultura de mercado) y los equipos para los aficionados a la agricultura (agricultura de subsistencia). El equipo realizado va orientado a un sector meramente aficionado.

Para obtener una buena gestión se han analizado los factores que causan la perdida de agua de las plantas. Como se estudia posteriormente, esto es debido al proceso de transpiración, ya que el 90% del agua recogida por la raíces se pierde en forma de vapor. Esta pérdida varía en función de la temperatura, humedad, radiación solar, viento y el tipo de cultivo.

Una vez analizado el problema, se ha procedido a diseñar un sistema electrónico que registre los datos de temperatura, humedad del aire, radiación solar, velocidad del viento y precipitación de agua de lluvia diariamente. Estos datos se obtienen mediante unos sensores y son procesados por un microcontrolador.

El sistema de control electrónico está formado por dos partes: una parte está ubicado en el exterior, formada por los sensores, y una segunda en el interior que consta del microcontrolador.

Con los datos extraídos de los sensores cada media hora, el microcontrolador calcula un tiempo de riego adecuado para un crecimiento óptimo de la planta utilizando los recursos necesarios cada 24horas, como recomienda la Organización de las Naciones Unidas para la Agricultura y Alimentación (FAO), descritas en Allen et. Al(1998), usando el método Penman - Monteith.

2. DEFINICIÓN DEL PROYECTO

2.1. OBJETIVOS

Dada la necesidad de los agricultores no profesionales de emplear el menor tiempo posible en los riegos de los campos, el autor ha desarrollado un dispositivo que automatice este proceso. Se quiere conseguir un riego, según las necesidades en las épocas más calurosas y secas, y a la vez prescindir de él en tiempos más fríos y lluviosos para poder ahorrar en energía.

Éste sistema gestiona automáticamente el tiempo de riego. El cual, vendrá determinado por las condiciones meteorológicas, centrándose exclusivamente en la temperatura, humedad del aire, radiación solar, velocidad del viento y precipitación. El control de la gestión del riego proporciona al agricultor un óptimo crecimiento de la cosecha y al mismo tiempo un ahorro económico tanto en electricidad, ya que el riego se realiza mediante un grupo de presión, como en agua, debido a que solo se gasta lo necesario para el óptimo crecimiento de la planta.

El objetivo personal del autor para éste proyecto, es crear una programación fácil y fluida en el entorno del microcontrolador Arduino. Se desea poder gestionar cualquier tipo de señal de entradas y salidas de los sensores necesarios y así adquirir los datos para, mediante algoritmos o fórmulas, poder automatizar o gestionar un sistema deseado.

2.2. JUSTIFICACIÓN DEL PROYECTO

Es de conocimiento general que los recursos naturales no deben desperdiciarse. Por este motivo el proyecto solo utiliza los recursos necesarios para un correcto crecimiento de las plantas. Con éste sistema se consigue un ahorro de agua y electricidad, ya que se gestiona de manera automática y calcula con gran eficiencia los tiempos de riego con los datos proporcionados por los sensores.

Desde un punto de vista académico se deseaba experimentar con el microcontrolador Arduino, en concreto, el modelo Arduino UNO que es el último modelo disponible de la casa, con los requisitos para poder implementar éste proyecto. El microcontrolador tiene un gran potencial para poder desarrollar éste tipo de instalaciones y a muy bajo coste. El autor también tiene interés en el tipo de programación que utiliza éste microcontrolador y su manera de desarrollar los proyectos.

Otro aspecto importante a la hora de la elección de este proyecto es poder aumentar los conocimientos de programación y adquisición y tratado de datos, debido a que el nivel alcanzado aun no era el deseado, siendo una magnífica oportunidad para conseguir un dominio adecuado. Además, desde el punto de vista del autor, es fundamental para un técnico, gozar de un buen dominio de programación, ya que los dispositivos programables se encuentran en multitud de sistemas eléctricos y electrónicos.

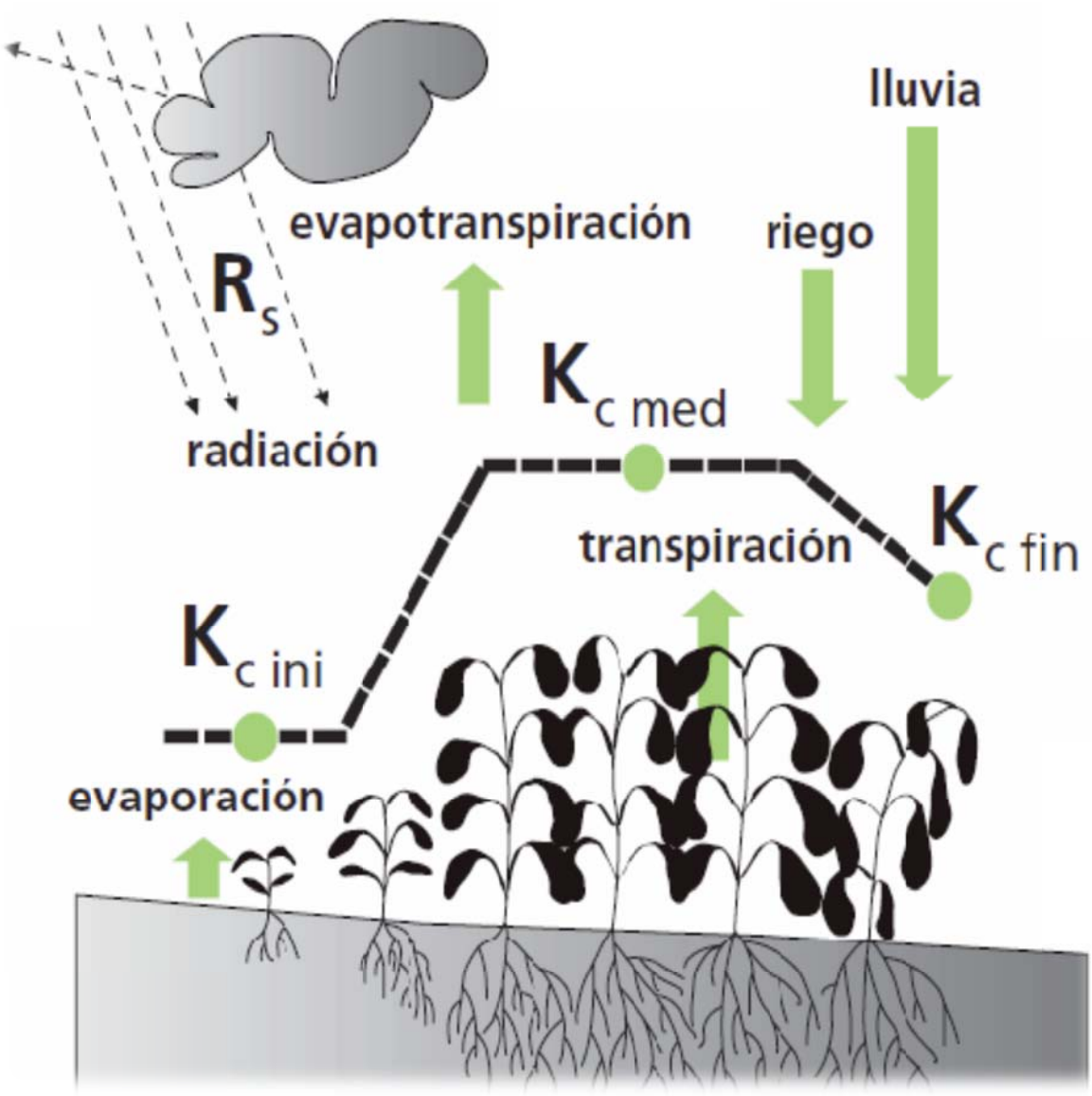
2.3. ALCANCE DEL PROYECTO

Para la realización de la estación de riego automatizada que debe ser empleada en una superficie agrícola de aproximadamente 100 , se hace imprescindible conocer la temperatura, la humedad, la radiación solar y la precipitación, para determinar la transpiración de la planta. La obtención de ésta información y su correcta gestión establece una serie de datos adicionales que se añaden al proyecto para generar unos óptimos tiempos de riego.

El proyecto incluye:

- Placa del microcontrolador Arduino UNO.
- Placa de simulación de sensores para comprobación de funcionamiento.
- Sensor de temperatura y humedad.
- Pluviómetro.
- Anemómetro.
- Piranómetro.
- Actuador.
- Amplificador de Piranómetro.
- Programación.

3. EVAPOTRANSPIRACIÓN DEL CULTIVO



3.1. Introducción a la Evapotranspiración.

En este proyecto se ha desarrollado un sistema de riego automático. Para este fin, deben considerarse las causas que producen las pérdidas de agua en los cultivos. El factor principal que debe controlarse es la evapotranspiración (ET). Esta puede estudiarse como una combinación de dos procesos independientes: la evaporación y la transpiración del cultivo.

3.1.1. Evaporación

La evaporación es el proceso por el cual el agua líquida se convierte en vapor de agua. Este proceso requiere dos tipos de energía: la radiación solar directa y, en menor grado, la temperatura ambiente del aire. La fuerza que disminuye el vapor de agua de una superficie evaporante es la diferencia entre la presión del vapor de agua en la superficie evaporante y la presión de vapor de agua de la atmosfera. A medida que ocurre la evaporación, el aire de alrededor se satura gradualmente y el proceso se vuelve cada vez más lento hasta detenerse completamente, si la humedad permanece alrededor de la hoja. El reemplazo del aire saturado por un aire más seco depende en gran medida de la velocidad del viento. Por lo tanto, la radiación, la temperatura del aire, la humedad atmosférica y la velocidad del viento son parámetros climatológicos a considerar al evaluar el proceso de la evaporación.

3.1.2. Transpiración

La transpiración consiste en la vaporización del agua líquida contenida en los tejidos de la planta y su posterior traslado hacia la atmosfera. Los cultivos pierden agua predominantemente a través de las estomas. Estos son unas pequeñas aberturas en la hoja de la planta, que realizan una función de conducción para que los gases y el vapor de agua puedan salir a la atmosfera.

El agua, junto con algunos nutrientes, es absorbida por las raíces y transportada a través de la planta. La vaporización ocurre dentro de la hoja, en los espacios intercelulares y el intercambio del vapor con la atmosfera es controlado por la abertura estomática. Casi toda el agua absorbida del suelo se pierde por transpiración y solamente una pequeña fracción se convierte en parte de los tejidos vegetales.

La transpiración, igual que la evaporación, depende del aporte de energía, del gradiente de presión del vapor y de la velocidad del viento. Para obtener un control de la transpiración se han de considerar: la radiación, la temperatura del aire, la humedad y el viento. Otro aspecto que influye en la transpiración es el contenido de agua del suelo y su capacidad de conducir el agua a las raíces, así como la salinidad del mismo y del agua de riego. La tasa de transpiración también es influida por las características y las distintas prácticas del cultivo y el medio donde se produce. El estado de desarrollo del cultivo también es un aspecto importante a considerar.

3.1.3. Evapotranspiración

La evaporación y la transpiración son procesos que ocurren simultáneamente. Aparte de la disponibilidad de agua en la superficie, la evaporación de un suelo cultivado es determinada principalmente por la fracción de radiación solar que llega a la superficie del suelo. Esta fracción disminuye a lo largo del ciclo del cultivo a medida que la vegetación del mismo proyecta mas sombra sobre el suelo. En las primeras etapas el agua se pierde principalmente por evaporación directa del suelo, pero con el desarrollo del cultivo y finalmente cuando este cubre totalmente el suelo, la transpiración se convierte en el principal motivo de la perdida de agua.

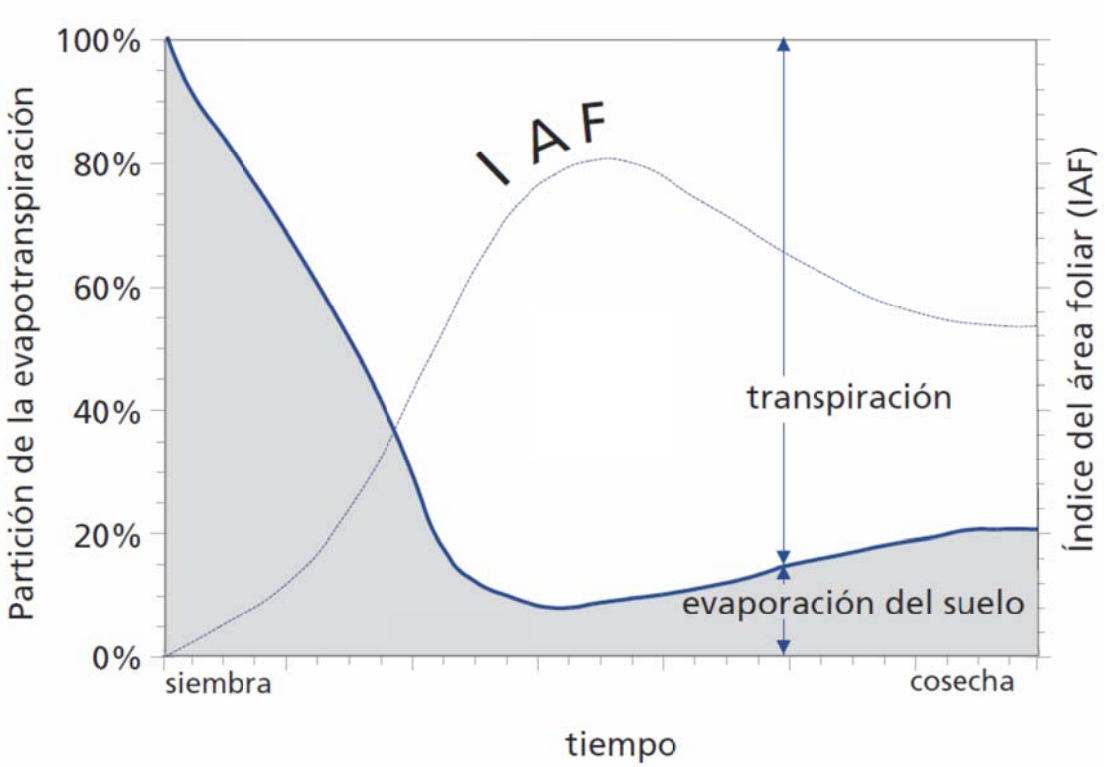


Figura 2: *Repartición de la evapotranspiración en evaporación y transpiración Durante el periodo de crecimiento de un cultivo anual*

En la figura 2 se puede observar como la evapotranspiración está dividida en sus dos componentes (evaporación y transpiración), en relación con el índice del área foliar (IAF). Este es una cantidad adimensional, que presenta el área de la hoja (lado superior solamente) por unidad de área de suelo debajo de ella. En el momento de la siembra, casi el 100% de la ET ocurre en forma de evaporación, mientras que cuando la cobertura vegetal es completa, más del 90% de la ET ocurre como transpiración.

3.1.4. Evapotranspiración del cultivo de referencia

El grado de evapotranspiración de una superficie de referencia, sin restricciones de agua, se conoce como evapotranspiración del cultivo de referencia (ETo). La superficie de referencia corresponde a un cultivo hipotético con unas características especificas Los elementos que influyen en la ETo son únicamente parámetros climáticos, pudiendo ser calculada mediante la obtención de las condiciones meteorológicas del lugar del cultivo. La ETo no tiene en cuenta el tipo de cultivo, ni las características del suelo.

3.1.5. Evapotranspiración del cultivo bajo condiciones estándar

La evapotranspiración del cultivo bajo condiciones estándar (ETc.), es la que tiene lugar cuando el cultivo se encuentra exento de enfermedades, con buena fertilización y que se desarrolla en superficies anchas y planas, bajo optimas condiciones de suelo y agua y en la que se logra la máxima productividad según las condiciones climatológicas de cada región. La ETc. cuantifica la cantidad de agua que necesita ser administrada al cultivo como riego o precipitación, mientras que la ET0 proporciona la cantidad de agua que pierde el cultivo en forma de evapotranspiración. La evapotranspiración del cultivo se calcula a partir de datos climáticos teniendo en cuenta los factores de la resistencia del cultivo, el albedo (fracción de radiación reflejada por la superficie terrestre) y la resistencia del aire según el método de Penman-Monteith. En este método se determina la estimación de la tasa de evapotranspiración del cultivo estándar de referencia (ETo). La relación ETc/ETo puede ser determinada para diferentes cultivos y es conocida como coeficiente del cultivo (Kc), y se utiliza para relacionar ETc a ETo de manera que ETc = Kc · ETo. Las diferencias en la anatomía de las hojas, características de los estomas, las propiedades aerodinámicas, e incluso el albedo, ocasionan que la evapotranspiración del cultivo difiera de la evapotranspiración del cultivo de referencia bajo las mismas condiciones climáticas. Otro aspecto a considerar son las diferentes fases en el crecimiento de un determinado cultivo, ya que el coeficiente de cultivo (Kc) cambia desde la siembra hasta la cosecha.

3.1.6. Estimación de la evapotranspiración

En la actualidad existen varios métodos para determinar la evapotranspiración, pero ninguno de ellos resulta ser una tarea fácil por el simple motivo que no es un parámetro que pueda medirse directamente ya que se requieren instrumentos específicos de lectura de los diversos parámetros físicos de los que el más fiable es el lisímetro que mide la cantidad de agua del suelo. De los distintos métodos existentes para determinar la evapotranspiración, los que más destacan son: balances de energía y microclimaticos (basados en el principio de conservación de la energía), balance de agua en el suelo (evaluación de los flujos de agua que entran y salen de la zona radicular del cultivo en un periodo de tiempo, figura 3), lisímetros (tanques aislados llenados con turba en los

que el cultivo se desarrolla. Se determina la perdida de agua por el cambio de masa), ET calculada con datos meteorológicos y ET estimada con el tanque de evaporación.

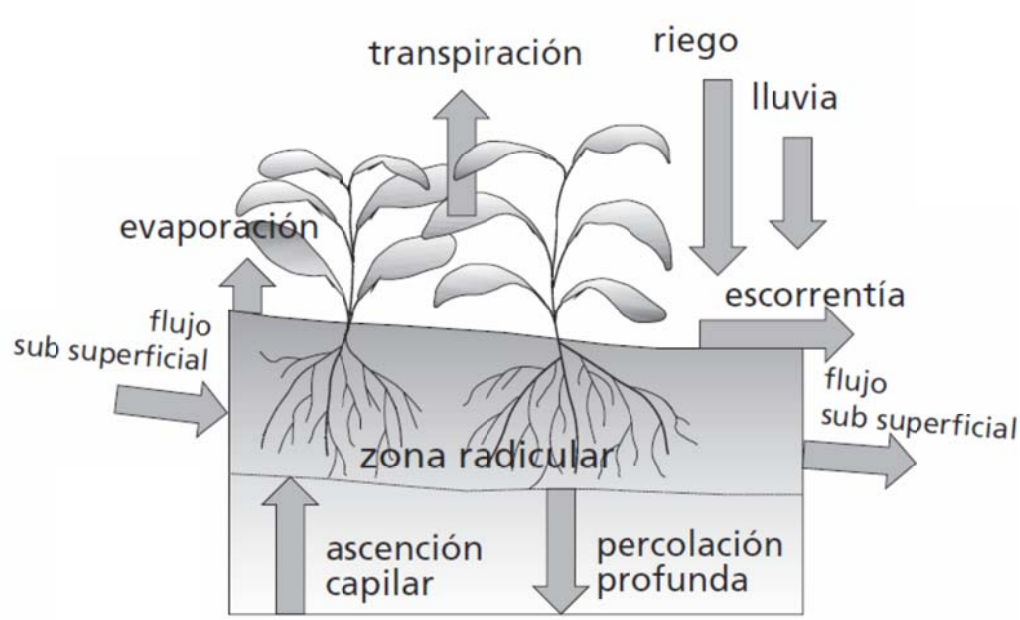


Figura 3: *Balance de agua en el suelo de la zona radicular.*

3.2. Ecuación de FAO Penman-Monteith

Éste capítulo pretende mostrar la importancia de contar con un método estándar y fiable para el cálculo de la evapotranspiración de referencia (ETo) con parámetros meteorológicos. El método de FAO Penman-Monteith se recomienda como el único método para determinar la evapotranspiración de referencia ETo. En este capítulo se describen el método, su derivación y los datos meteorológicos requeridos.

3.2.1. Introducción a la ecuación de FAO Penman-Monteith

La necesidad de realizar el riego optimo para las necesidades del cultivo, en distintos territorios, hizo que la FAO (Organización de las Naciones Unidas para la Agricultura y Alimentación) reuniera un conjunto de expertos e investigadores en riego, para poder realizar el cálculo de los requerimientos de agua de los cultivos y para elaborar recomendaciones sobre la revisión y la actualización de procedimientos a este respecto.

Los expertos se decantaron por el método combinado de Penman-Monteith como el nuevo método estandarizado para el cálculo de la evapotranspiración de referencia y aconsejo sobre los procedimientos para el cálculo de los parámetros que la

formula incluye. El método reduce las imprecisiones del método FAO Penman y produce globalmente valores más consistentes con datos reales de uso de agua de diversos cultivos.

$$ET_0 = \frac{0{,}408 \cdot \Delta \cdot (R_n - G) + \gamma \cdot \frac{900}{T+273} \cdot u_2 \cdot (e_s - e_a)}{\Delta + \gamma \cdot (1 + 0{,}34 \cdot u_2)}$$

Ecuación 1

Donde:

ET_0	Evapotranspiración de referencia (mm dia^{-1})
R_n	Radiación neta en la superficie del cultivo (MJ m^{-2} dia^{-1})
R_a	Radiación extraterrestre (MJ m^{-2} dia^{-1})
G	Flujo del calor de suelo (MJ m^{-2} dia^{-1})
T	Temperatura media del aire a 2 m de altura (°C)
u_2	Velocidad del viento a 2 m de altura (m s^{-1})
e_s	Presión de vapor de saturación (kPa)
e_a	Presión real de vapor (kPa)
$e_s - e_a$	Déficit de presión de vapor (kPa)
Δ	Pendiente de la curva de presión de vapor (kPa $°C^{-1}$)
□	Constante psicrométrica (kPa $°C^{-1}$)

Para el cálculo de la ecuación de FAO Penman-Monteith se emplean datos meteorológicos de radiación solar, temperatura del aire, humedad y velocidad del viento. Para garantizar una elevada precisión en el cálculo, los datos climáticos deben ser medidos o evaluados a 2 m de altura, sobre una superficie extensa de cultivo.

El valor resultante obtenido de esta ecuación basada en los parámetros meteorológicos no puede considerarse fiable al cien por cien, debido a que las etapas de adquisición para la obtención de los datos climáticos presentan unas ciertas desviaciones, la utilización de datos meteorológicos faltantes al igual que el error producido por el propio calculo, por el simple motivo de la simplificación y sencillez de la formula, ya que está pensada para que sea utilizada por el máximo número de personas sin importar los conocimientos en la materia. Los errores producidos en el cálculo expuesto anteriormente no presentan una desviación muy dispar de los datos reales, considerándose un error tolerable pero existente.

3.2.2. Datos meteorológicos para la estimación de la ET

Los métodos para calcular la evapotranspiración parten de datos meteorológicos que requieren varios parámetros climatológicos y físicos. Algunos de estos parámetros se miden directamente en estaciones meteorológicas. Otros parámetros se relacionan con los datos comúnmente medidos y se pueden obtener con la ayuda de relaciones directas o empíricas. Este capítulo presenta la medición y cómputo de los datos requeridos para el cálculo de la evapotranspiración de referencia por medio del método FAO Penman-Monteith. Diversos ejemplos ilustran los varios procedimientos del cálculo. También se presentan procedimientos para estimar datos faltantes.

3.2.2.1. Temperatura

La centralita de riego automático que se ha realizado en este proyecto, considera de rigurosa importancia el hecho que la temperatura del aire se mida dentro de un abrigo meteorológico a 2 m sobre la superficie, de acuerdo a los estándares de la Organización Meteorológica Mundial (OMM). La temperatura del aire se mide con un STA-5031 del fabricante Geónica instalado en la caseta meteorológica. Este sensor junto al sistema registra tanto la mínima como la máxima, y posteriormente al final del día se calcula el promedio.

Debido a la relación no lineal de la humedad con la temperatura, ambas incluidas en la ecuación FAO Penman-Monteith, la presión de vapor para cierto periodo se debe calcular como la media entre la presión de vapor bajo la temperatura máxima y la presión de vapor bajo la temperatura mínima del aire en ese periodo tal y como se puede observar en la ecuación 4. la temperatura máxima diaria del aire (Tmax) y la temperatura mínima diaria del aire (Tmin) son, respectivamente, la máxima y la mínima temperatura del aire observadas durante un periodo de 24 horas. La temperatura media diaria del aire (Tmedia) se emplea en la ecuación FAO Penman-Monteith solamente para calcular la pendiente de la curva de la presión de saturación de vapor (Δ) y del impacto de la densidad media del aire, debido a que las variaciones de temperatura en el valor de estos parámetros climáticos es pequeño. Tmedia para periodos de 24 horas se define como el promedio de las temperaturas máxima (Tmax) y mínima diaria (Tmin) en lugar del promedio de las mediciones horarias de temperatura.

3.2.2.2. Humedad del aire

Los factores más determinantes en la evaporación de agua en un cultivo son el aporte de energía del sol y el aire circundante. La diferencia entre la presión de vapor de agua en la superficie evapotranspirante y el aire circundante es el factor determinante en la variación del vapor. Tal y como se ha explicado anteriormente en zonas secas y calientes bien regadas, consumen una gran cantidad de agua debido a la energía proporcionada por el sol. Sucede lo contrario en zonas húmedas, donde la elevada humedad reduce la evapotranspiración aun teniendo un gran aporte de energía.

Humedad relativa.

La humedad relativa es el cociente entre la cantidad de agua que contiene el aire a una determinada temperatura y la cantidad que podría contener si estuviera saturado a la misma temperatura. El valor de la humedad relativa es adimensional y se expresa en forma de porcentaje. Aunque la presión real de vapor puede ser relativamente constante a lo largo del día, la humedad relativa fluctúa entre un valor máximo al amanecer y un valor mínimo a primeras horas de la tarde. La variación de la humedad relativa se produce porque la presión de saturación de vapor depende de la temperatura del aire. Como la temperatura del aire cambia durante el día, la humedad relativa también cambia sustancialmente.

Presión de vapor

Como es sabido, el vapor de agua es un gas y su presión influye a la presión atmosférica. La cantidad de vapor de agua en el aire está relacionada directamente con la presión parcial ejercida por ese vapor de agua y esta es por lo tanto una medida directa del contenido de vapor de agua del aire, esta presión se expresa en Pascal (Pa).

Cuando un volumen de aire se encuentra retenido sobre una superficie evaporante de agua, se alcanza un equilibrio entre las moléculas de agua que se incorporan al aire y las que vuelven a la fuente de agua. En ese momento, se considera que el aire está saturado puesto que no puede almacenar ninguna molécula de agua adicional. La presión correspondiente se llama presión de saturación de vapor. La cantidad de moléculas de agua que pueden almacenarse en el aire depende de la temperatura (T). Cuanta más alta es la temperatura del aire, mas alta es la capacidad de almacenar vapor de agua y más alta es la presión de saturación de vapor.

La presión real de vapor (ea) es la presión de vapor ejercida por el vapor de agua en el aire. Cuando el aire no se satura, la presión real de vapor será más baja que la presión de vapor de saturación. La diferencia entre la presión de saturación y la presión real de vapor se llama déficit de presión de vapor o déficit de saturación y es un indicador preciso de la capacidad real evaporativa del aire.

La presión de saturación del vapor puede ser calculada en función de la temperatura según la ecuación 2.

$$e^0(T) = 0.6108 \cdot \exp\left[\frac{17.27 \cdot T}{T+237.3}\right]$$

Ecuación 2

Donde:

e°(T) Presión de saturación de vapor a la temperatura del aire, T (kPa)

T Temperatura del aire (°C)

La problemática que presenta la no linealidad de la Ecuación 2 es que la presión media de saturación de vapor para un día, debe ser calculada como el promedio de la presión de saturación de vapor a la temperatura máxima media y la presión de saturación de vapor a la temperatura mínima media del aire y la humedad relativa para ese periodo tal y como se indica en la siguiente ecuación.

$$e_s = \frac{e^0(Tmax)+e^0(Tmin)}{2}$$

Ecuación 3

Donde:

E_s	Presión media de saturación de vapor (kPa)
$e°(T_{min})$	Presión de saturación de vapor a la temperatura mínima diaria (kPa)
$e°(T_{max})$	Presión de saturación de vapor a la temperatura máxima diaria (kPa)

Pendiente de la curva de presión de saturación de vapor (Δ)

Tal y como muestra el estudio de la FAO, mediante la ecuación 4, es necesario el cálculo de la pendiente de la curva que relación la presión de saturación de vapor con la temperatura, Δ. La pendiente de la curva (grafico 1) a una temperatura viene dada por.

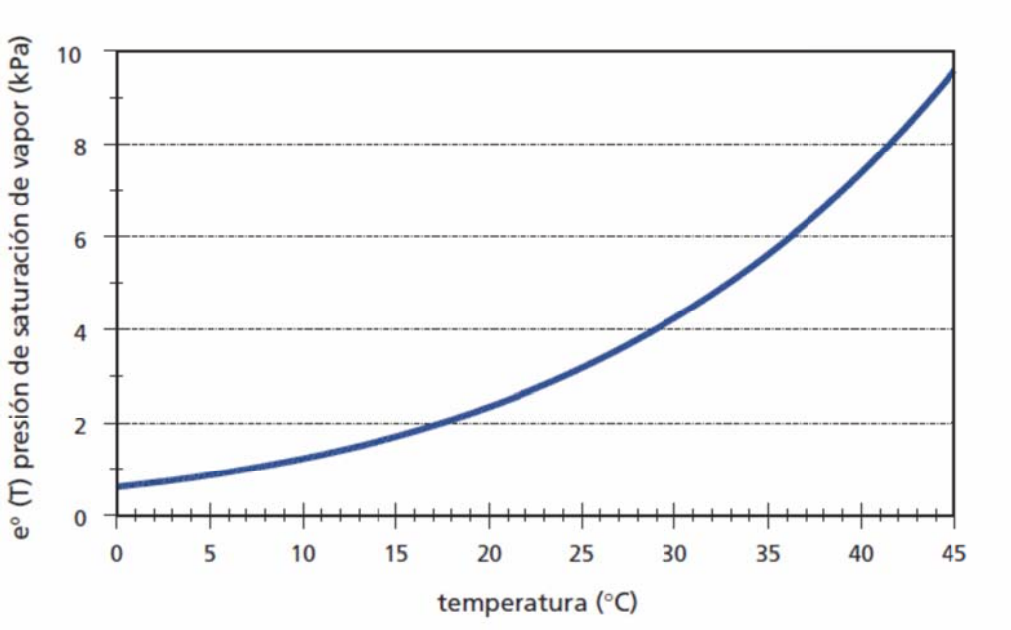


Grafico 1: *Presión de saturación de vapor en función de la temperatura.*

$$\Delta = \frac{4098 \cdot \left[0.6108 \cdot exp\left(\frac{17.27 \cdot T}{T+237.3}\right)\right]}{(T+237.3)^2}$$

Ecuación 4

Donde:

Δ	Pendiente de la curva de la presión de saturación de vapor a la temperatura del aire T (kPa °C^{-1})
T	Temperatura del aire (°C)

Presión real de vapor (e_a) en función de la humedad relativa

La presión real de vapor se puede derivar de la humedad relativa. Dependiendo de la fiabilidad de los datos de humedad, se pueden utilizar diversas ecuaciones, ya que si fuera el caso de disponer de valores de humedad relativa que presentasen error considerable en la medida, se utilizaría únicamente la humedad relativa máxima, pero no es el caso de este proyecto ya que el error producido por el sensor de humedad relativa es admisible.

$$e_a = \frac{e^0(Tmax) \cdot \frac{HRmax}{100} + e^0(Tmin) \cdot \frac{HR_{min}}{100}}{2}$$

Ecuación 5

Donde:

e_a	Presión real de vapor (kPa)
$e°(T_{min})$	Presión de saturación de vapor a la temperatura mínima diaria (kPa)
$e°(T_{max})$	Presión de saturación de vapor a la temperatura máxima diaria (kPa)
HR_{max}	Humedad relativa máxima (%)
HR_{min}	Humedad relativa mínima (%)

Déficit de presión de vapor (es - ea)

El déficit de presión de vapor es la diferencia entre la presión de saturación de vapor (es) y la presión real de vapor (ea) durante un determinado periodo. En el caso de ea, esta se computa similarmente con la aplicación de las ecuaciones 5, usando los promedios de cada periodo. El uso de la temperatura media del aire en lugar de Tmax y Tmin en la ecuación 2 produce una subestimación de es, con lo cual se obtiene un valor menor del déficit de presión y por lo tanto se hallan valores menores de ETo (subestimación de evapotranspiración del cultivo de referencia).

3.2.2.3. Radiación

La evapotranspiración está determinada por la cantidad de energía disponible para evaporar el agua. La radiación solar es la fuente más importante de energía en el planeta y puede cambiar grandes cantidades de agua líquida en vapor de agua. La cantidad potencial de radiación que puede llegar a una superficie evaporante viene determinada por su localización y época del año. Debido a las diferencias en la posición del planeta y a su movimiento alrededor del sol, esta cantidad potencial de radiación es diferente para cada latitud y para las diversas estaciones del año. La radiación solar real que alcanza la superficie evaporante depende de la turbidez de la atmosfera y de la presencia de nubes que reflejan y absorben cantidades importantes de radiación. Cuando se determina el efecto de la radiación solar en la evapotranspiración, se debe también considerar que no toda la energía disponible se utiliza para evaporar el agua. Parte de la energía solar se utiliza también para calentar la atmosfera el suelo. En la figura 4 se muestran los componentes que interviene en la radiación.

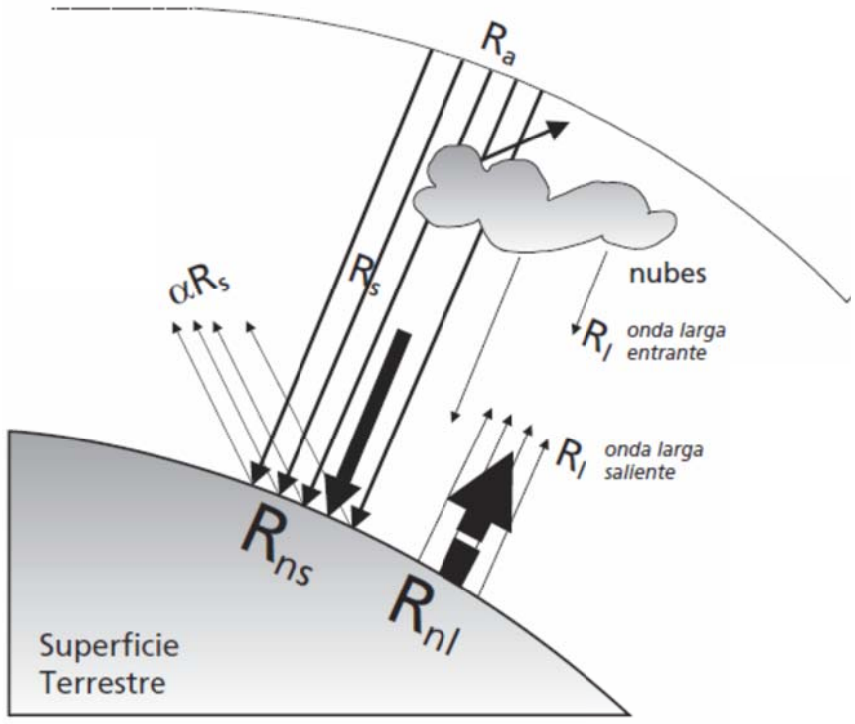


Figura 4: *Componentes de la radiación*

Radiación extraterrestre (R_a)

La radiación solar recibida en la parte superior de la atmosfera terrestre sobre una superficie horizontal se conoce como radiación extraterrestre, Ra. La energía de la radiación que recibe una superficie perpendicular a los rayos del sol en el extremo superior de la atmosfera terrestre, se llama constante solar, y tiene un valor aproximado de 0,082 MJ m-2 min-1. La intensidad local de la radiación, sin embargo, viene determinada por el ángulo entre la dirección de los rayos solares y la superficie de la

atmosfera. Este ángulo cambia durante el día y es diferente en diversas latitudes y en diversas épocas del año.

Si por ejemplo, el sol se encuentra directamente encima de la cabeza, el ángulo de incidencia es cero y la radiación extraterrestre es 0,082 MJ m-2 min-1. Así como las estaciones cambian, la posición del sol, la longitud del día y la radiación extraterrestre también cambian. De esta forma se puede afirmar que la radiación extraterrestre va en función de la latitud, la época del año y la hora del día, tal y como se puede observar en la ecuación siguiente:

$$R_a = \frac{24 \cdot 60}{\pi} \cdot G_{sc} \cdot d_r \cdot [\omega_s \cdot \sin\varphi \cdot \sin\delta + \cos\varphi \cdot \cos\delta \cdot \sin\omega_s]$$

Ecuación 6

Donde:

R_a	Radiación extraterrestre (MJ m^{-2} dia^{-1})
G_{sc}	Constante solar (0,082 MJ m^{-2} min^{-1})
d_r	Distancia relativa inversa Tierra-Sol (Ecuación 8)
ω_s	Ángulo de radiación a la puesta del sol (rad) (Ecuación 10).
φ	Latitud (rad) (Ecuación 7)
δ	Declinación solar (rad) (Ecuación 9).

Hay que tener en cuenta para el cálculo de la latitud tiene que estar expresada en radianes positivos para el hemisferio norte y negativo para el hemisferio sur. Para convertir de grados decimales a radianes se utiliza la siguiente ecuación:

$$(\text{Radianes}) = \frac{\pi}{180} \cdot (\text{grados decimales})$$

Ecuación 7

La distancia relativa inversa Tierra-Sol, dr y la declinación solar, están dadas por:

$$d_r = 1 + 0.033 \cdot \cos\left(\frac{2 \cdot \pi}{365} \cdot J\right)$$

Ecuación 8

$$\delta = 0.409 \cdot \sin\left(\frac{2 \cdot \pi}{365} \cdot J - 1.39\right)$$

Ecuación 9

Donde J es el número del día del año (máximo 365 días). El ángulo de radiación a la hora de la puesta del sol, ωs, viene expresada por:

$$\omega_s = \arccos(-\tan\varphi \cdot \tan\delta)$$

Ecuación 10

Radiación solar o de onda corta (R_s)

Es la cantidad de radiación que llega a un plano horizontal en la superficie terrestre. Si no se cuentan con mediciones directas, de esta magnitud, y dado que aproximadamente en un día despejado constituye un 75 % de la radiación extraterrestre (se pierde un 25 %), se puede aplicar la ecuación 11.

$$R_s = \left(0{,}25 + \frac{n \cdot \pi}{48 \cdot \omega_s}\right) \cdot R_a$$

Ecuación 11

Dónde:

Rs	Radiación solar o de onda corta (MJ m^{-2} dia^{-1})
n	Duración real de la insolación (horas)
ωs	Ángulo de radiación a la hora de la puesta del sol (rad) (Ecuación 10).
Ra	Radiación extraterrestre (MJ m^{-2} dia^{-1}) (Ecuación 6)

Radiación solar en un día despejado (R_{so})

Para el cálculo de la radiación solar en días despejados, en el caso en que no se puedan obtener las fracciones de la radiación solar extraterrestre que llega al campo, se emplea la (ecuación 17), donde se emplea la variable z para introducir la altitud de del campo del cultivo (m) y Ra como la radiación extraterrestre.

$$R_{so} = (0.75 + 2 \cdot 10^{-5}) \cdot R_a$$

Ecuación 12

Albedo (α) y radiación neta solar (R_{ns})

Una cantidad considerable de la radiación solar que llega a la superficie terrestre se refleja en ella. La fracción, α, de la radiación solar que es reflejada por la superficie se conoce como albedo. El albedo es muy variable de acuerdo al tipo de superficie y el ángulo de incidencia o la pendiente de la superficie terrestre. Este parámetro puede variar desde 0,95 para la nieve recientemente caída a 0,05 para un suelo desnudo y

húmedo. Una Cubierta verde de vegetación tiene un albedo de entre 0,20 a 0,25. Para el cultivo de referencia, se asume que α tiene un valor de 0,23.

Por otro lado, la radiación neta solar, Rns, es la fracción de la radiación solar Rs que no se refleja en la superficie, tal y como se puede ver en la ecuación 13.

$$R_{ns} = (1-\alpha)\, R_s.$$

Ecuación 13

Donde:

- R_{ns} Radiación neta solar o de onda corta (MJ m^{-2} dia^{-1})
- α Albedo o coeficiente de reflexión del cultivo, que es 0,23 para el cultivo hipotético de referencia (adimensional),
- R_s Radiación solar entrante (MJ m^{-2} dia^{-1}).

Radiación neta de onda larga (R_{nl})

Como es sabido, la radiación solar absorbida por la tierra se convierte en energía térmica. La tierra pierde esta energía por una serie de procesos, del que se puede destacar la emisión de radiación. Nuestro planeta, que tiene una temperatura mucho más baja que el sol, emite energía radiante con longitudes de onda más largas que el sol. Por este motivo, la radiación terrestre es conocida como radiación de onda larga. La radiación de onda larga emitida por el planeta es absorbida por la atmosfera o perdida hacia el espacio. La radiación de onda larga recibida por la atmosfera aumenta su temperatura. Por consiguiente, la atmosfera irradia también energía. Parte de la radiación emitida por la atmosfera se dirige nuevamente hacia la superficie terrestre. Por lo tanto, la superficie terrestre emite y recibe radiación de onda larga. La diferencia entre la radiación de onda larga entrante y saliente se llama radiación neta de onda larga, Rnl. Como la radiación saliente de onda larga es casi siempre mayor que la radiación entrante, Rnl representa una pérdida de energía. La ecuación empleada para el cálculo de la radiación neta de onda larga es la siguiente:

$$R_{nl} = \sigma \cdot \left[\frac{T^4_{maz,k} + T^4_{min,k}}{2}\right] \cdot \left(0.34 - 0.14 \cdot \sqrt{e_a}\right) \cdot \left(1.35 \cdot \frac{R_s}{R_{so}} - 0.35\right)$$

Ecuación 14

Donde:

- R_{nl} Radiación neta de onda larga (MJ m^{-2} dia^{-1})
  constante de Stefan-Boltzmann (4,903 x 10-9 MJ K^{-4} m^{-2} dia^{-1})
- $T_{max,K}$ temperatura máxima absoluta durante un periodo de 24 horas (K = °C + 273,16)
- $T_{min,K}$ temperatura mínima absoluta durante un periodo de 24 horas (K = °C + 273,16)

e_a Presión de vapor real (kPa)
R_s/R_{so} Radiación relativa de onda corta (valores _ 1,0)
R_s Radiación solar medida o calculada (Ecuación 11) (MJ m^{-2} dia^{-1})
R_{so} Radiación en un día despejado. (Ecuación 12) (MJ m^{-2} dia^{-1})

Radiación neta (R_n)

La radiación neta, Rn, no es nada más que la diferencia entre la radiación entrante y saliente de longitudes de onda cortas y largas (ecuación 6). Expresado de otra forma seria el equilibrio entre la energía absorbida, reflejada y emitida por la superficie terrestre o la diferencia de la radiación de onda corta entrante (Rns) y la radiación de onda larga saliente (Rnl). El valor de la radiación neta (Rn) es normalmente positiva durante el día y negativa durante la noche. El valor diario total para Rn es casi siempre positivo para 24 horas, excepto en condiciones extremas de latitudes muy elevadas.

$$R_n = R_{ns} - R_{nl}$$

Ecuación 15

Flujo de calor del suelo (G)

Para el riego diario del cultivo, el flujo de calor de debajo la superficie es muy pequeño y puede ser ignorado, tal y como nos indica el estudio FAO riego y drenaje, pero para entender mejor este concepto se explicara a continuación, ya que no deja de ser menos importante. En las estimaciones de evapotranspiración, se deben considerar todos los términos del balance energético, ya que la suma de la energía de calor sensible (H) mas el flujo de calor del suelo (G) y el flujo de calor latente (ET) tiene que ser igual a la radiación neta (Rn), tal y como se muestra en la ecuación 15. El flujo del calor del suelo, G, es la energía que se utiliza para calentar el suelo. G tiene valores positivos cuando el suelo se calienta y negativos cuando el suelo se enfría. Aunque el flujo calórico del suelo es pequeño comparado con Rn y puede ser objetivo de no consideración, la cantidad de energía ganada o perdida por el suelo en este proceso teóricamente debe restarse o agregarse a Rn para estimar la evapotranspiración.

$$R_n = G + H + \lambda ET$$

Ecuación 16

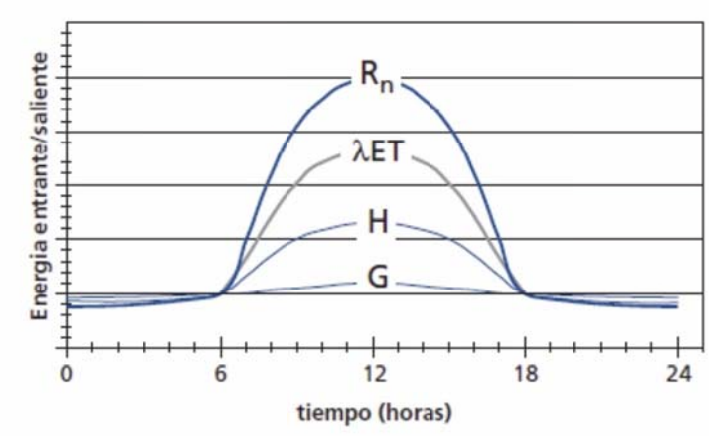


Grafico 2: *Balance de energía de una superficie agrícola*

3.2.2.4. Viento

El viento es considerado según el estudio de la FAO como un factor importante a la hora de obtener la evapotranspiración. Las principales características del viento son la dirección y la velocidad. Como la velocidad del viento en una localidad dada varia con el tiempo, es necesario expresarla como el promedio sobre un intervalo determinado de tiempo, que podría ser el promedio durante un día. La velocidad del viento se mide en metros por segundo (m s-1).

El encargado de medir la velocidad del viento es el anemómetro. Los anemómetros están formados por unas cazoletas o propulsores que giran sobre un eje gracias a la fuerza del viento. El número de vueltas efectuado durante un periodo de tiempo establecido como por ejemplo un día, proporcionaría el promedio de la velocidad del viento en un día.

Hay que tener en cuenta la altura a la que se mida el viento, ya que la fricción superficial tiende a reducir la velocidad del viento que se pretende medir, es decir que contra más elevado este la sonda mayor será la velocidad registrada. La altura empleada en la ecuación de la evapotranspiración es de 2 m de altura sobre la superficie. Si no se pudiera obtener la medida a la altura establecida, puede establecerse una relación logarítmica.

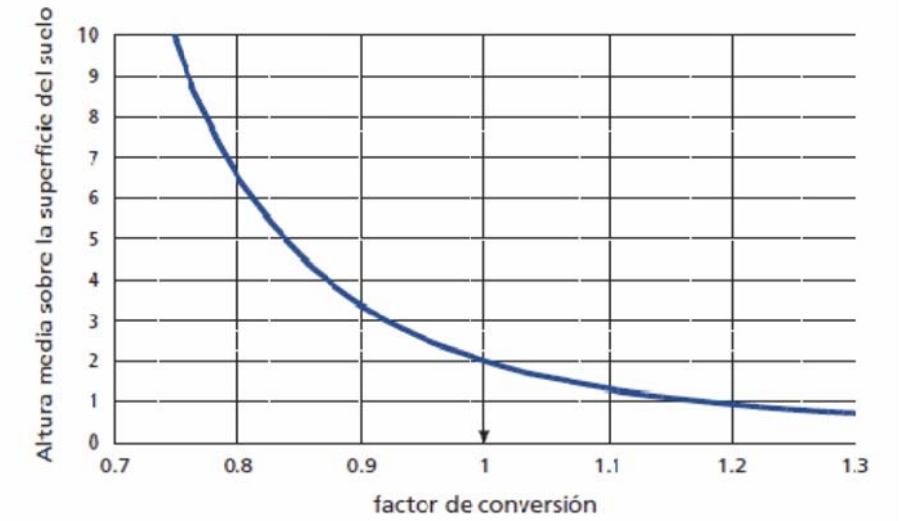


Grafico 3: *Factor de conversión para convertir velocidad del viento a una determinada altura*

3.2.2.5. Presión atmosférica

La atmosfera, que es la capa de aire que rodea a la Tierra, ejerce como cualquier otro fluido, una presión sobre los cuerpos que están en su interior. Esta presión es debida a las fuerzas de atracción entre la masa de la Tierra y la masa de aire y se denomina presión atmosférica**.**

La evaporación en altitudes elevadas ocurre en parte gracias a la baja presión atmosférica que se expresa con la constante psicrométrica. Este efecto es, sin embargo, pequeño y en los procedimientos del cálculo, el valor medio para una localidad es suficiente.

Este proyecto no dispone de un sensor de presión debido a que es un parámetro que influye poco y puede aplicarse una simplificación de la ley de los gases ideales a una temperatura estándar de 20°C (ecuación 17), que nos proporciona un valor muy próximo al real. Pero este hecho hará que se tenga que conocer e introducir la altitud en metros en el programa instalado en el ordenador para que el sistema calcule automáticamente la presión atmosférica.

$$P = 101.3 \cdot \left(\frac{293 - 0.0065 \cdot z}{293}\right)^{5.26}$$

Ecuación 17

Donde:

P Presión atmosférica (kPa)
z Elevación sobre el nivel del mar (m)

3.2.2.6. Calor latente de vaporación

La energía requerida para cambiar una masa unidad de agua líquida a vapor de agua bajo presión y temperatura constantes se denomina calor latente de vaporización, λ. El valor del calor latente de vaporización varía en función de la temperatura. Cuanto más elevada sea la temperatura, menos energía será requerida. Como el calor latente de vaporización varia un valor constante de 2,45 MJ kg-1 para la simplificación de la ecuación de FAO Penman-Monteith. Este valor corresponde al calor latente de vaporización a una temperatura del aire de alrededor de 20 °C.

3.2.2.7. Constante Psicrométrica

La constante psicrométrica, , es la siguiente:

$$\gamma = \frac{C_P \cdot P}{\varepsilon \cdot \lambda}$$

Ecuación 18

Donde:

	Constante psicrométrica (kPa °C^{-1})
P	Presión atmosférica (kPa)
	Calor latente de vaporización, 2,45 (MJ kg^{-1})
C_p	Calor especifico a presión constante, 1,013 x 10-3 (MJ kg^{-1}°C^{-1})
	Cociente del peso molecular de vapor de agua /aire seco = 0,622.

3.3. Evapotranspiración del Cultivo (ETc)

En este apartado se informara de las características e importancia del coeficiente del cultivo y su cálculo para determinar la evapotranspiración del cultivo bajo condiciones estándar (ETc). Las condiciones estándar de los cultivos son: que se desarrollen en campos extensos, bajo condiciones agronómicas excelentes y sin limitaciones de humedad en el suelo. La evapotranspiración de un cultivo es diferente a la del cultivo de referencia (ETo) ya que en este se tiene en cuenta las características propias del cultivo como la cobertura del suelo, propiedades de la vegetación y resistencia aerodinámica. Los efectos de las características que distinguen al cultivo del pasto están incorporados en el coeficiente del cultivo (Kc). En la metodología del coeficiente del cultivo, la evapotranspiración del cultivo se calcula multiplicando ETo por Kc. Se tiene que recalcar que el valor del Kc es siempre el mismo ya que sólo disponemos de lechugas, cuyo Kc es prácticamente 1 durante todo su crecimiento.

3.3.1. Factores que influyen en el coeficiente del cultivo (Kc)

Hay una serie de aspectos que influyen considerablemente y varían el coeficiente del cultivo que son: el tipo de cultivo, el clima, la evaporación del suelo y las etapas de crecimiento del cultivo.

Tipo de cultivo:

Las plantas tienen una serie de características propias que afectan al coeficiente de cultivo como puede ser: altura del cultivo, propiedades aerodinámicas, los estomas. Estas características anteriormente descritas pueden llegar a afectar como máximo entre un 15 a un 20 %. Puede verse un ejemplo en la figura 5, donde en función del tipo de cultivo puede variar el coeficiente del cultivo.

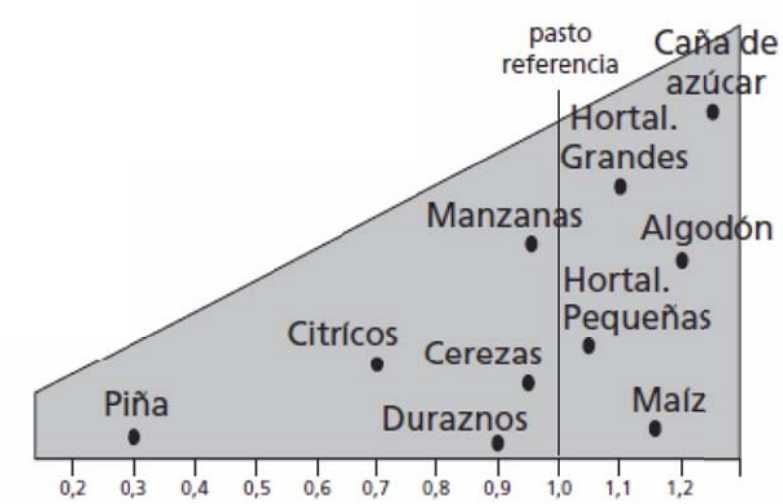


Figura 5: *Valores de Kc en función del tipo de cultivo.*

Clima:

En este proyecto únicamente se basa en el cultivo en condiciones estándar, como ya se ha comentado anteriormente con humedades relativas mínimas superiores al 45% y vientos inferiores a 2 m s-1. Está claro que dependiendo de la altura del cultivo y la velocidad del viento, afectara más o menos a la resistencia aerodinámica del pasto y por lo tanto los valores de coeficiente de cultivo.

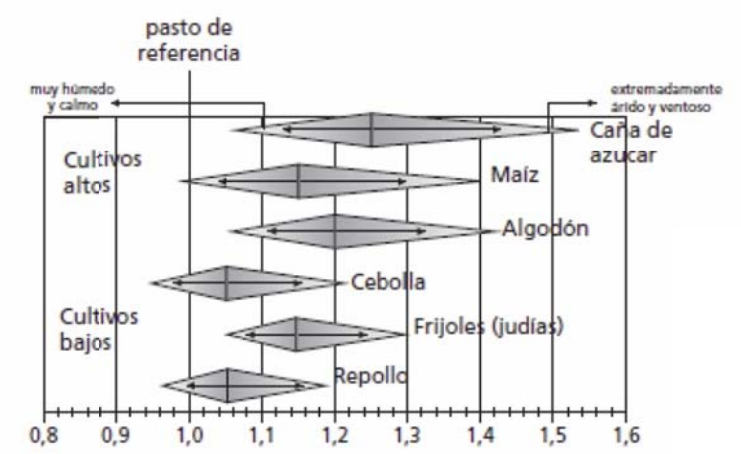


Figura 6: *Valores de Kc en función del clima.*

Evaporación del suelo:

La evaporación del suelo esta englobada en el coeficiente de cultivo, ya que contempla tanto la evaporación como la transpiración. El valor del coeficiente Kc para cultivos que cubren completamente el suelo refleja principalmente las diferencias en transpiración, debido a que la evaporación que ocurre en el suelo es relativamente pequeña. La evaporación aumenta cuando hace acto de presencia la lluvia o el cultivo es regado, se tiene que tener en cuenta que si el pasto esta germinando, la sombra producida por la masa foliar de este produce que la radiación solar, se proyecto directamente sobre el suelo aumentando consigo la evaporación. A modo de ejemplo se presenta la figura 8 donde en una superficie del suelo húmeda, en las condiciones de poca cobertura la evaporación puede llegar a superar la unidad, y si el cultivo está completamente desarrollado, la evaporación del suelo es casi inexistente.

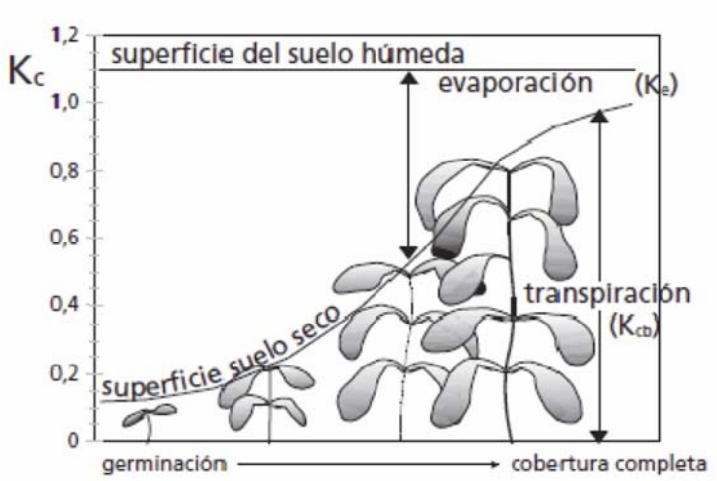


Figura 7: *Valores de Kc en función del estado de crecimiento del cultivo.*

Etapas de crecimiento del cultivo:

En el transcurso de este apartado ya se ha ido introduciendo que el crecimiento del cultivo influye en las condiciones del riego, ya que conlleva una variación de la altura y un aumento del área foliar. El crecimiento es computable en cuatro etapas:

-Inicial: está comprendida entre la fecha de siembra y el momento que el cultivo alcanza aproximadamente el 10% de cobertura del suelo. La longitud de la etapa inicial depende en gran medida del tipo de cultivo, la variedad del mismo, la fecha de siembra y del clima. El final de la etapa inicial ocurre cuando la vegetación cubre aproximadamente un 10% de la superficie del suelo. Para cultivos permanentes, la fecha de siembra es reemplazada por el momento en que aparecen las primeras hojas.

-Desarrollo del cultivo: La etapa de desarrollo del cultivo está comprendida desde el momento en que la cobertura del suelo es de un 10% hasta el momento de alcanzar la cobertura efectiva completa. Para una gran variedad de cultivos, el estado de cobertura completa ocurre al inicio de la floración. Para cultivos en hileras, donde en las hileras se cubra toda la superficie del suelo.

-Mediados de temporada: Esta etapa engloba el periodo de tiempo entre la cobertura hasta el comienzo de la madurez. La madurez, se puede visualizar mediante la aparición de un color amarillento en las hojas o la caída de algunas.

-Final de temporada: Esta es la última fase del desarrollo del cultivo, esta engloba desde la madurez hasta la recolección. La etapa inicial y final de temporada son las que tienen un coeficiente Kc más pequeño de todos.

Cálculo de la evapotranspiración del cultivo

Vista toda la parte teórica del coeficiente del cultivo, la evapotranspiración del cultivo ETc se calcula como el producto de la evapotranspiración del cultivo de referencia, ETo y el coeficiente del cultivo Kc:

$$ETc = Kc \cdot ETo$$

Ecuación 19

En éste proyecto se ha tomado como referencia un único tipo de cultivo, con un coeficiente del cultivo medio cercano a 1(exactamente 0,8966)

Cultivo	Kc ini	Kc med	Kc fin
Lechuga	0,7	1,00	0,95

3.4. Cálculos de la ETc para periodos de un día

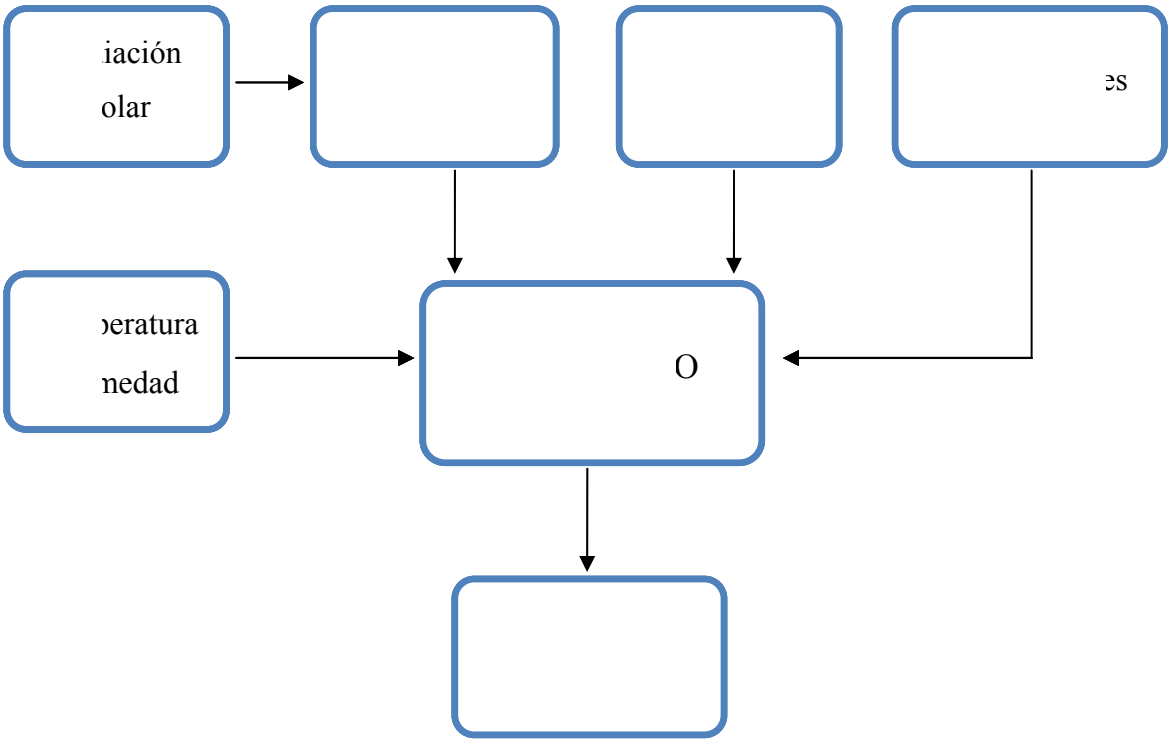
El cálculo de la ETo usando la ecuación de Penman-Monteith para periodos de tiempo de 24 horas proporcionara generalmente resultados precisos. Por tanto en éste proyecto será el método que se va a utilizar. Los datos meteorológicos requeridos son los siguientes:

- Temperatura del aire: Temperatura máxima (Tmax) y temperatura mínima (Tmin) diaria.
- Humedad del aire: el promedio diario de la presión real de vapor (ea) derivada de lecturas psicrometrícas, de la temperatura del punto de condensación o de los datos de humedad relativa.
- Velocidad del viento: valores promedio diarios para 24 horas de la velocidad diaria del viento medida a una altura de 2 m (u2).
- Radiación: Radiación neta diaria (Rn) medida o calculada de la radiación solar de onda corta y de la radiación de onda larga o de la duración real de las horas diarias de insolación (n). Como la magnitud del flujo diario de calor del suelo (G) debajo de la superficie de referencia es relativamente pequeña, esta puede ser ignorada en cálculos diarios. Para éste proyecto, se ha medido directamente la Radiación neta diaria(Rn)

4. DESCRIPCIÓN DE COMPONENTES

4.1. Esquema de bloques.

El proyecto contiene una parte de "*hardware*", del cual se puede ver el diagrama de bloques en la siguiente figura, en ella figuran todas las etapas necesarias para su funcionamiento.



En el esquema podemos ver el bloque de adquisición de datos (Sensores de temperatura, humedad, radiación solar, velocidad del viento y precipitaciones). El sensor de radiación solar, necesita de un amplificador (AC420) para poder mandar una señal de 0 a 5 voltios.

4.2. Microcontrolador Arduino UNO

4.2.1. PLACA ARDUINO UNO

Figura 8: *Placa del Microcontrolador Arduino UNO*

El microcontrolador que vamos a utilizar en éste proyecto es el Arduino. Se trata de una placa open hardware por lo que su diseño es libre de distribución y utilización.

En éste caso, tenemos la placa UNO, la cual es la última versión de la marca Arduino, y está basada en el microcontrolador ATMega328. Cuenta con 14 entradas digitales / pines de salida (de los cuales 6 pueden ser utilizados como salidas PWM), 6 entradas analógicas, un oscilador de cristal de 16 MHz, una conexión USB, un conector de alimentación, un encabezado ICSP y un botón de reinicio.

Contiene todo lo necesario para apoyar el microcontrolador, simplemente se conecta a un ordenador con un cable USB o el poder con un adaptador de CA a CC-o de la batería para empezar.

El Uno no utiliza el chip controlador FTDI USB a serie. En su lugar, cuenta con la Atmega8U2 programado como un convertidor de USB a serie.

Resumen:

-14 Entradas digitales

-14 Pines salida

-Posibilidad de 6 Salidas PWM

-6 Entradas analógicas

-Conexión USB.

4.2.2. ENTORNO DE DESARROLLO

Para programar la placa es necesario descargarse de la página web de Arduino el entorno de desarrollo (IDE). Se dispone de versiones para Windows y para MAC, así como las fuentes para compilarlas en LINUX. En la figura 10 se muestra el aspecto del entorno de programación. En el caso de disponer de una placa de USB es necesario instalar los drivers FTDI. Estos drivers vienen incluidos en el paquete de Arduino mencionado anteriormente. Existen en la web, versiones para distintos sistemas operativos.

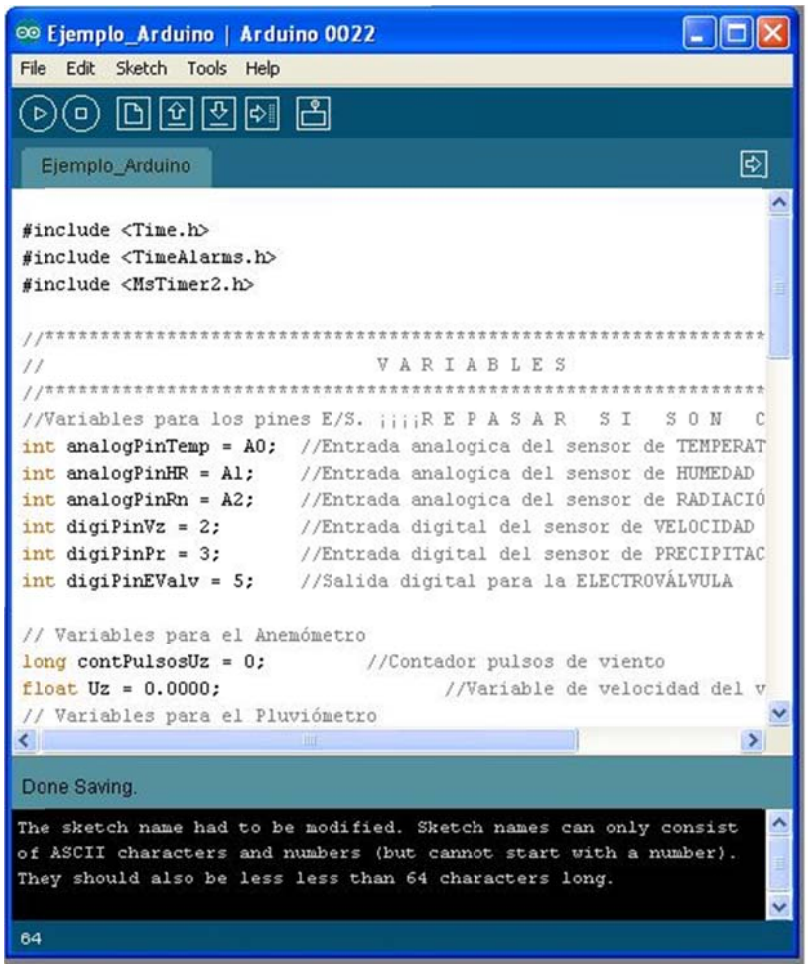


Figura 9: *Entorno de desarrollo.*

Lo primero que tenemos que hacer para comenzar a trabajar con el entorno de desarrollo de Arduino es configurar las comunicaciones entre la placa Arduino y el PC. Para ello deberemos abrir en el menú "*Tools*" la opción "*Serial Port*". En ésta opción deberemos seleccionar el puerto serie al que está conectada nuestra placa.

El primer paso para comprobar que todo lo que hemos hecho hasta ahora está bien y familiarizarnos con el interfaz de desarrollo, es abrir uno de los ejemplos. Se recomienda abrir el ejemplo "*Blink"*. Para ello debemos acceder a través del menú File → Sketchbook → Examples → Digital → Blink.

El ejemplo "*Blink"* lo único que hace es parpadear un LED que está colocado ene l pin número 13 de la placa. Vamos a ver qué hay que hacer para subir el programa a la placa Arduino. Primero comprobamos que el código fuente es el correcto. Para ello pulsamos el botón de verificación de código que tiene forma de triángulo inclinado 90 grados Figura 11. Si todo va bien deberá aparecer un mensaje en la parte inferior de la interfaz indicando "*Done compiling*". Una vez que el código ha sido verificado procederemos a cargarlo en la placa. Para ello tenemos que pulsar el botón reset de la placa Figura x e inmediatamente después pulsar el botón que comienza la carga Figura 11.

Durante la carga del programa, en la placa USB, se encenderán los LED que indican que se están enviando y recibiendo información por el puerto serie: TX/RX. Si todo se ha realizado correctamente debe aparecer el mensaje "*Done uploading"*. Ahora tan sólo queda esperar unos 8 segundos aproximadamente para comprobar que todo ha salido bien. Si el LED colocado en el pin 13 de la placa se enciende y se apaga cada segundo entonces todo ha ido bien.

Figura 10: *Barra de herramientas*

4.3. ADQUISICIÓN DE DATOS.

En éste proyecto vamos a utilizar los sensores meteorológicos de la marca GEONICA, ya que están especializados en éste tipo de proyectos, y sus productos son ideales para la adquisición de datos para la fórmula de la evapotranspiración.

Como los sensores, son de la misma marca, están pensados para colocarse en soportes y así poder resguardarse de las inclemencias del tiempo, y a su vez, poder registrar los datos sin ningún tipo de interferencia.

Dicho esto, a continuación se va a pasar a explicar diferentes tipos de sensores (Temperatura, Humedad, Radiación solar, Velocidad del viento y Precipitaciones), y las características de los escogidos. Las hojas de características de los diferentes sensores, se encuentran en el apartado de Anexos.

4.3.1. SENSOR DE PRECIPITACIÓN.

Definición de precipitación:

La precipitación se define como la caída de hidrometeoros del cielo que llegan a la superficie terrestre en forma de lluvia, nieve, granizo, llovizna… Cuando está en estado líquido se expresa en milímetros (mm) o litros por metro cuadrado (l/m2).

Aunque esté expresada en mm no es una unidad de longitud sino de volumen, ya que se basa en la cantidad de lluvia caída sobre una superficie de 1 m2, y la altura es la medición de precipitación en esa superficie en milímetros. Por ejemplo, 1 mm de precipitación significa que en una base de 1 m2 ha caído 1 litro de agua.

Pluviómetro:

Para este proyecto se ha elegido el Pluviómetro que ofrece GEONICA, el modelo 52202.

Figura 11: *Pluviómetro, Modelo 52202*

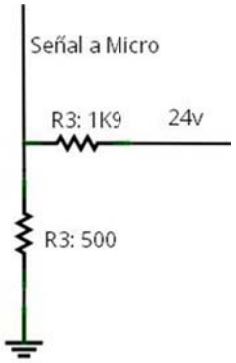
El diseño utiliza un mecanismo de balancín para la medida simple y eficaz de la precipitación. La geometría del cubo y materiales especialmente seleccionados para el máximo de salida de agua, reduciendo así la contaminación y errores.

El área de captación es de 200cm2, un diámetro de 18cm y 30cm de altura. Tiene una resolución de 0,1mm por pulso con una precisión de 2-3%.

La señal de salida es de 24AC/DC y 500mA, por lo que se necesita adecuar la salida a 5v. Para ello, crearemos un divisor de tensión con dos resistencias, de 1900Ω y de 500Ω.

5v = 10mA *500 Ω

Con lo que finalmente tendremos un circuito como el siguiente.



4.3.2. SENSOR DE VELOCIDAD DEL VIENTO.

Definición de viento: Se define como un fenómeno meteorológico que consiste en el movimiento del aire por la atmósfera ocasionado por fenómenos naturales.

Los movimientos del aire se deben a las acciones de la energía solar sobre la superficie y a las diferencias de presión entre las capas atmosféricas, provocadas por las diferentes temperaturas de la Tierra.

Anemómetro:

El sensor elegido es el Modelo 27106 de la casa GEONICA.

Figura 12: *Anemómetro, Modelo 57106*

La hélice de cuatro palas helicoidales ataca a un mini-tacómetro generador, proporcionando una salida directamente proporcional a la componente axial del viento. El cambio de polaridad significa cambio de velocidad del viento.

La salida del taco-generador es de 10 pulsos por revolución. Para 1800rpm tenemos una velocidad media medida de 8,8 m/s.

4.3.3. SENSOR DE TEMPERATURA Y HUMEDAD

Definición de temperatura: La temperatura es una magnitud referida a las nociones comunes de calor o frío.

Definición de Humedad relativa: Se define como la cantidad o porcentaje de vapor de agua que contiene el aire y que podría llegar a contener. Este parámetro no tiene unidades se mide en porcentaje.

El factor evaporación viene directamente relacionado con la temperatura y del nivel de saturación del aire. A mayor temperatura mayor evaporación, pero hay que tener en cuenta que si el nivel del aire esta cerca de la saturación por muy elevada que se la temperatura, no aumentará la humedad relativa.

Sensores de Temperatura y Humedad:
Los sensores elegidos vienen juntos en un mismo modelo: el STH – 5031

Figura 13: *Modelo STH-5031*

El rango de medida del sensor de temperatura es de -30º a 70ºC con una sensibilidad de 0,1ºC. La señal de salida es un circuito resistivo, de 0v a 1v.

La señal de salida del sensor de humedad es, como el de temperatura, de 0v a 1v.

4.3.4. SENSOR DE RADIACIÓN SOLAR

Definición de temperatura: La radiación solar es el conjunto de radiaciones electromagnéticas emitidas por el Sol. Ésta para alcanzar la superficie de la Tierra, ha de pasar la capa de ozono la cual absorbe la mayor parte de radiación. La magnitud para medirla es la irradiancia, que mide la energía por unidad de tiempo y área cuando alcanza la Tierra. La unidad para medir esta son los W/m².

Modelo LP02:

Figura 14: *Piranómetro, Modelo LP02*

LP02 es una alternativa moderna para superar el problema llamado "estrella" o "en blanco y negro" que implican una pobre estabilidad de la pintura reflectante blanca.

LP02 sirve para medir el flujo de radiación solar que es incidente sobre una superficie plana en W/m2 a partir de un campo de 180 grados de vista (también llamado "global" la radiación solar).

La señal de salida del LP02 es muy pequeña, 15 µV por cada W/m2, con lo que necesitamos de un modelo de amplificar ésta señal.

4.3.5. AMPLIFICADOR AC420

Para poder controlar debidamente la señal recibida por el piranómetro, tenemos que utilizar el amplificador AC420, que también ofrece la casa GEONICA.

Figura 15: *Amplificador AC420*

Según la hoja de características del amplificador, para calcular la ganancia interna, tenemos la siguiente fórmula:

$$\frac{}{}$$

Por lo tanto, si ponemos una R = 10 kOhm, tendremos una salida de 1mA por cada mV de entrada.

Así que si teniendo en cuenta que el rango máximo del piranómetro es de 2000w/m2, cuya salida es de 3mV, el amplificador mandará una señal al Arduino de 30mA, que con una resistencia de 166,6 ohmios (realmente serían dos en serie de 150ohm y 68ohm), obtendríamos 5v.

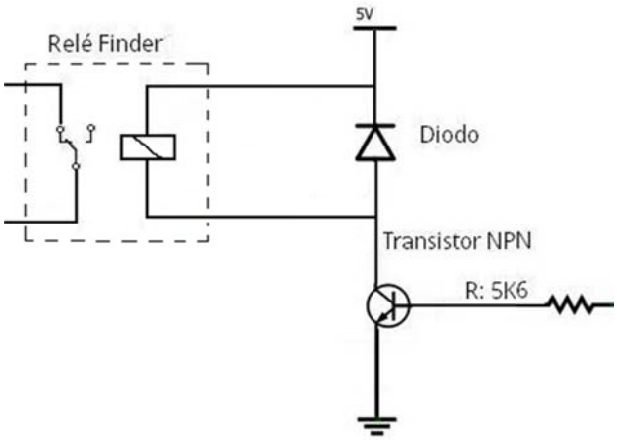
4.3.6. ACTUADOR PARA LA BOMBA DE RIEGO

Para garantizar una presión constante en el riego, se empleará una bomba de 4 bares de presión. Esta bomba está conectada a la red eléctrica y el fabricante indica una potencia de 1kW.

Para accionar la bomba es imprescindible un actuador, que en este caso será un mini-relé para circuito impreso de la del fabricante FINDER, el 40.51. Este dispositivo se caracteriza por disponer de una bobina de accionable por corriente de bajo consumo con tensión excitable de 5V. La corriente nominal soportada por el dispositivo es de 8A y el fabricante garantiza 10 millones de operaciones.

Figura 16: *Mini-Relé Finder 40.51*

Para el correcto funcionamiento del relé, utilizaremos un transistor NPN para un junto con un diodo y una resistencia de 5k6 Ω.

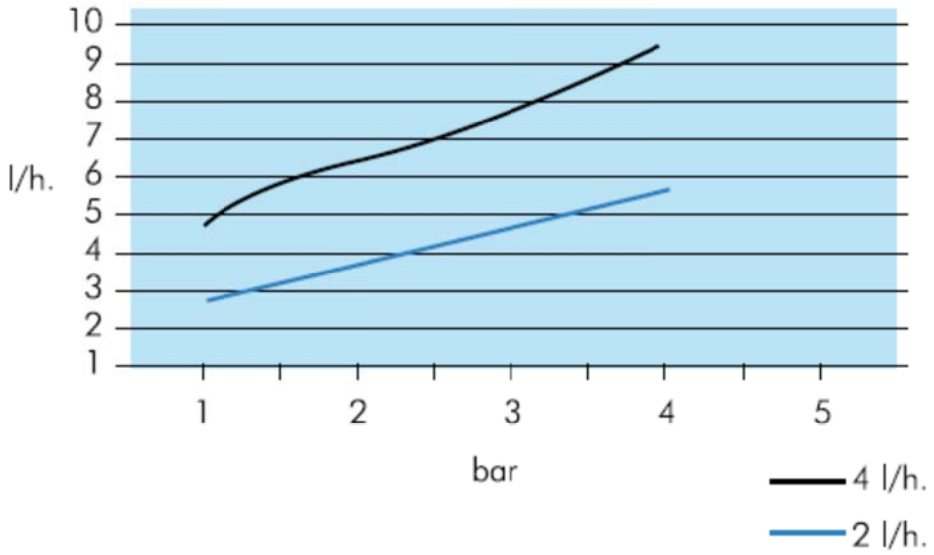


4.3.7. GOTEROS

Para el riego por goteo, se consideran los goteros de interlínea como los más adecuados para el riego.

En la huerta hay 483 goteros, separados 30cm uno de cada en cada línea de riego, de 16mm de diámetro y con un caudal de 4l/h.

Figura 17: *Gotero interliniar*



5. TIEMPOS DE RIEGO

En éste apartado se va a explicar el tiempo de riego necesario para que el riego sea el más eficaz y totalmente automático.

Los tiempos de riego se obtienen mediante el cálculo de la ETc que son las pérdidas de agua diarias y la dosis práctica de riego.

Una vez entendido el cálculo de la ETc se describe la forma para poder contabilizar el caudal real por metro cuadrado de la superficie agrícola en cuestión.

Por un lado se tiene de calcular el número de goteros por metro cuadrado según la ecuación 20 y con la ayuda de la figura 18.

$$Nº\ goteros/m^2 = \frac{1}{a \cdot b}$$

Ecuación 20

Donde:

a Distancia entre goteros de la misma línea (m)
b Distancia entre las distintas líneas de riego (m)

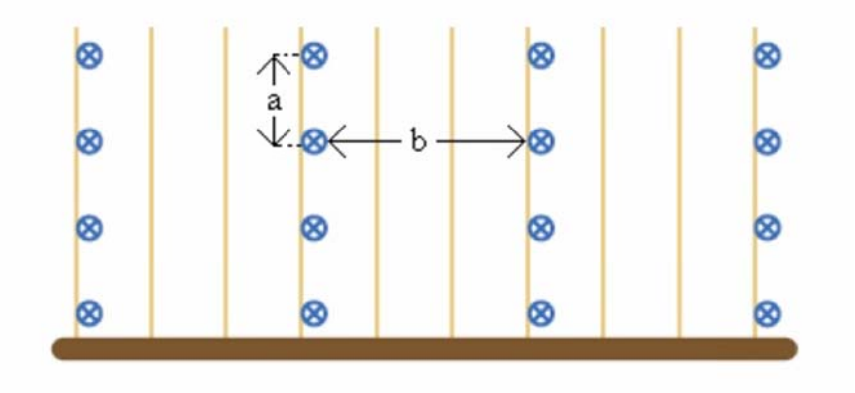


Figura 18: *Distancia entre los goteros de la huerta*

En éste proyecto, tenemos una distancia entre goteros de 30cm, y una separación entre líneas de riego de 60cm. Es decir,

a: 0,3 m
b: 0,6 m

Con lo que, según la ecuación nº 20, tenemos 5.55 goteros por metro cuadrado, y redondeando 6 goteros por metro cuadrado.

A continuación se procederá a realizar el producto entre el caudal real y el nº de goteros/ , obteniendo el caudal de riego por unidad de superficie, tal y como se presenta en la ecuación 21.

$$C_r = \frac{Nºgoteros}{m^2} \cdot caudal\ real$$

Ecuación 21

Donde:

C_r	Caudal de riego por unidad de superficie (l $m^{-2}h^{-1}$)
Caudal real	Caudal real de riego de un gotero (l h^{-1})

En ésta instalación tenemos unos goteros de la marca Plasgot, de 16mm de sección con un caudal de 4 L/h a una presión de 1 bar, con lo que el caudal de riego por superficie será: Cr = 22.22 L/h ·

Ahora se procede al cálculo del tiempo de riego que es el tiempo que el sistema de bombeo está en marcha para subministrar el agua perdida durante el día. Mediante la ecuación 23 se puede estimar el tiempo de riego diario.

$$T_r = \frac{ET_c}{C_r}$$

Ecuación 22

Donde

T_r	Tiempo de riego (h dia^{-1})
ET_c	Evapotranspiración del cultivo (mm dia^{-1})
C_r	Caudal de riego por unidad de superficie (l m^{-2} h^{-1})

6. CÓDIGO DE PROGRAMACIÓN

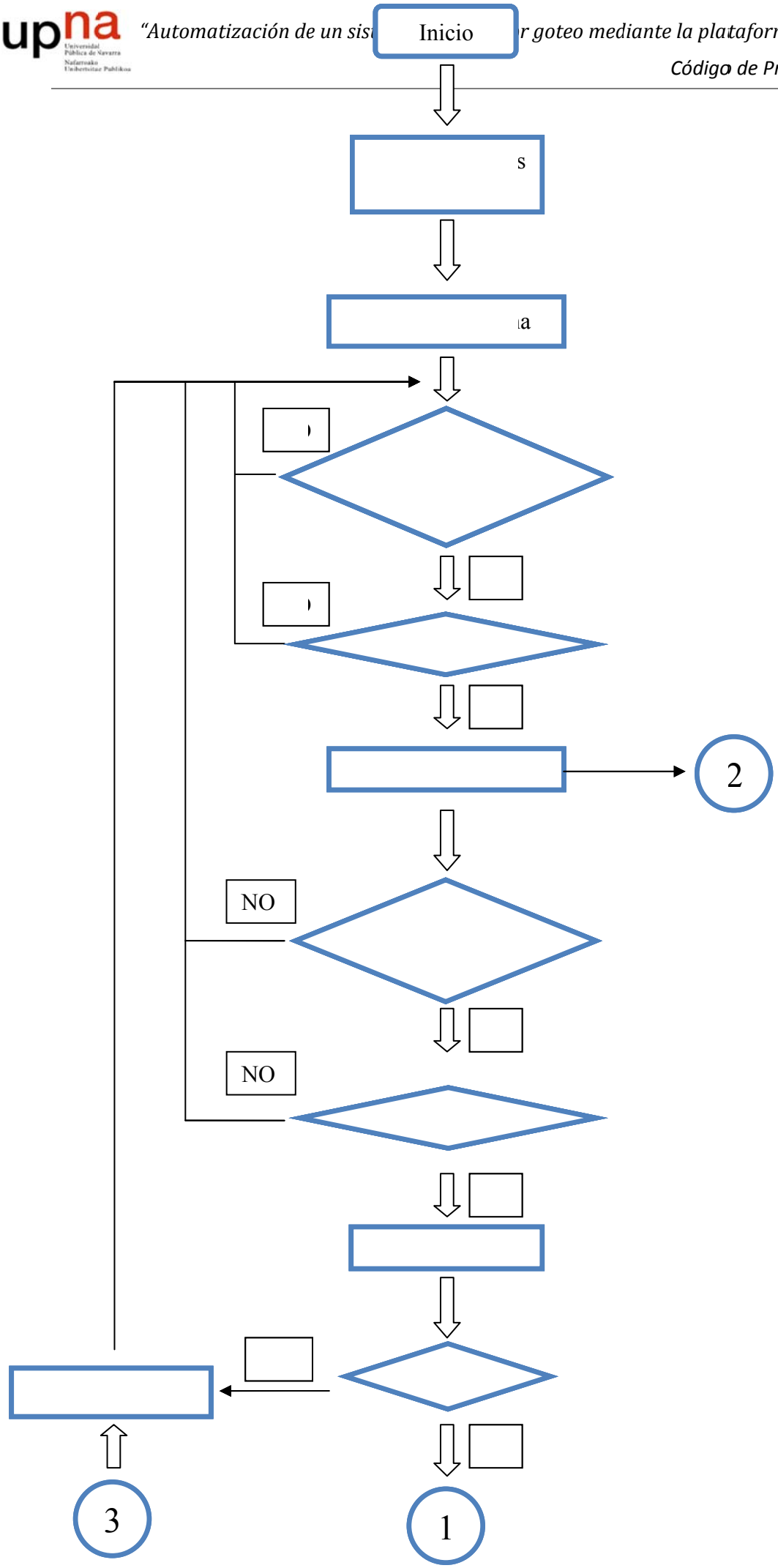
En éste apartado se desea hacer una guía para explicar la programación realizada para el riego automático. Se muestran los algoritmos de programación realizados y a continuación explicamos brevemente cómo está estructurado el código de programación.

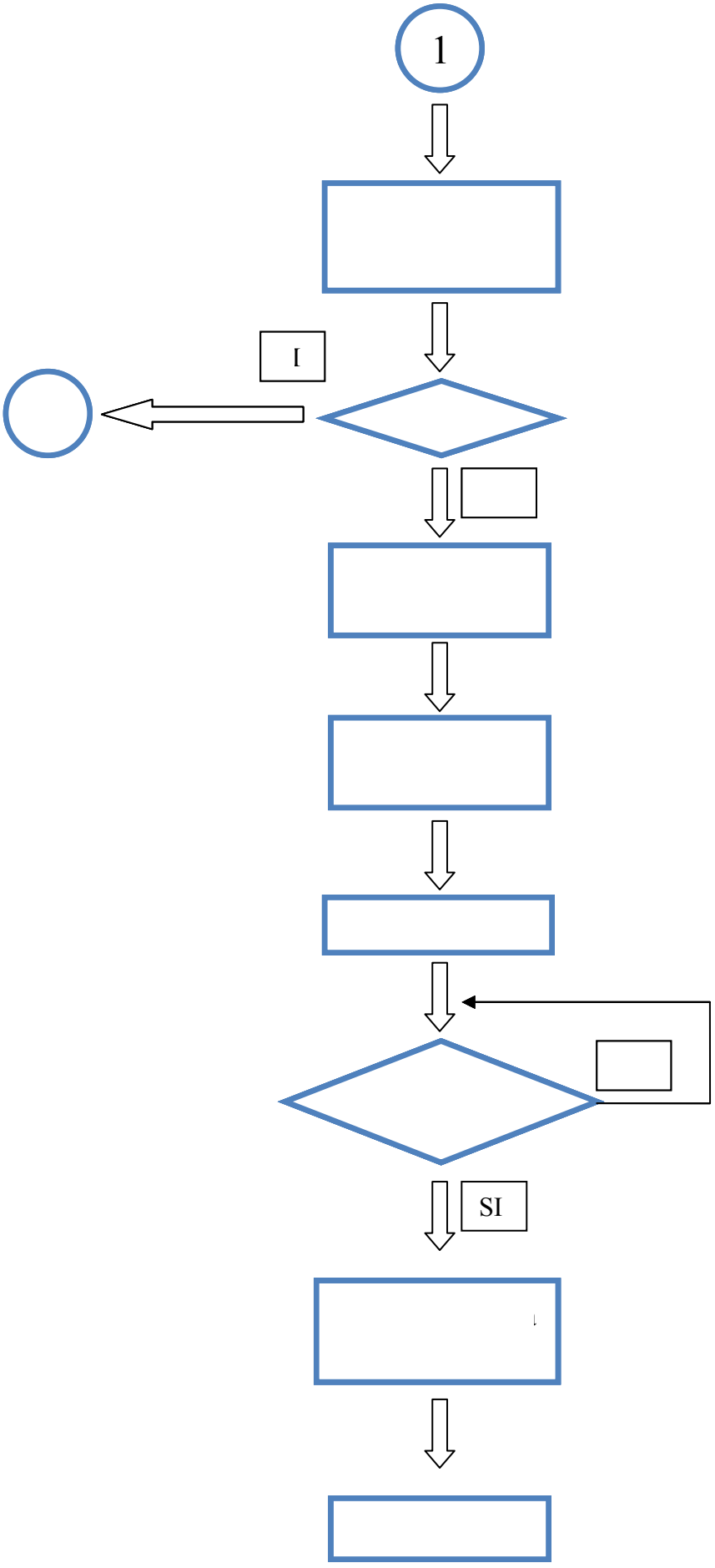
El programa se divide principalmente en 3 apartados. La estructura básica de la programación (declarar variables, setup( ) y loop( )), las Alarmas y las Funciones.

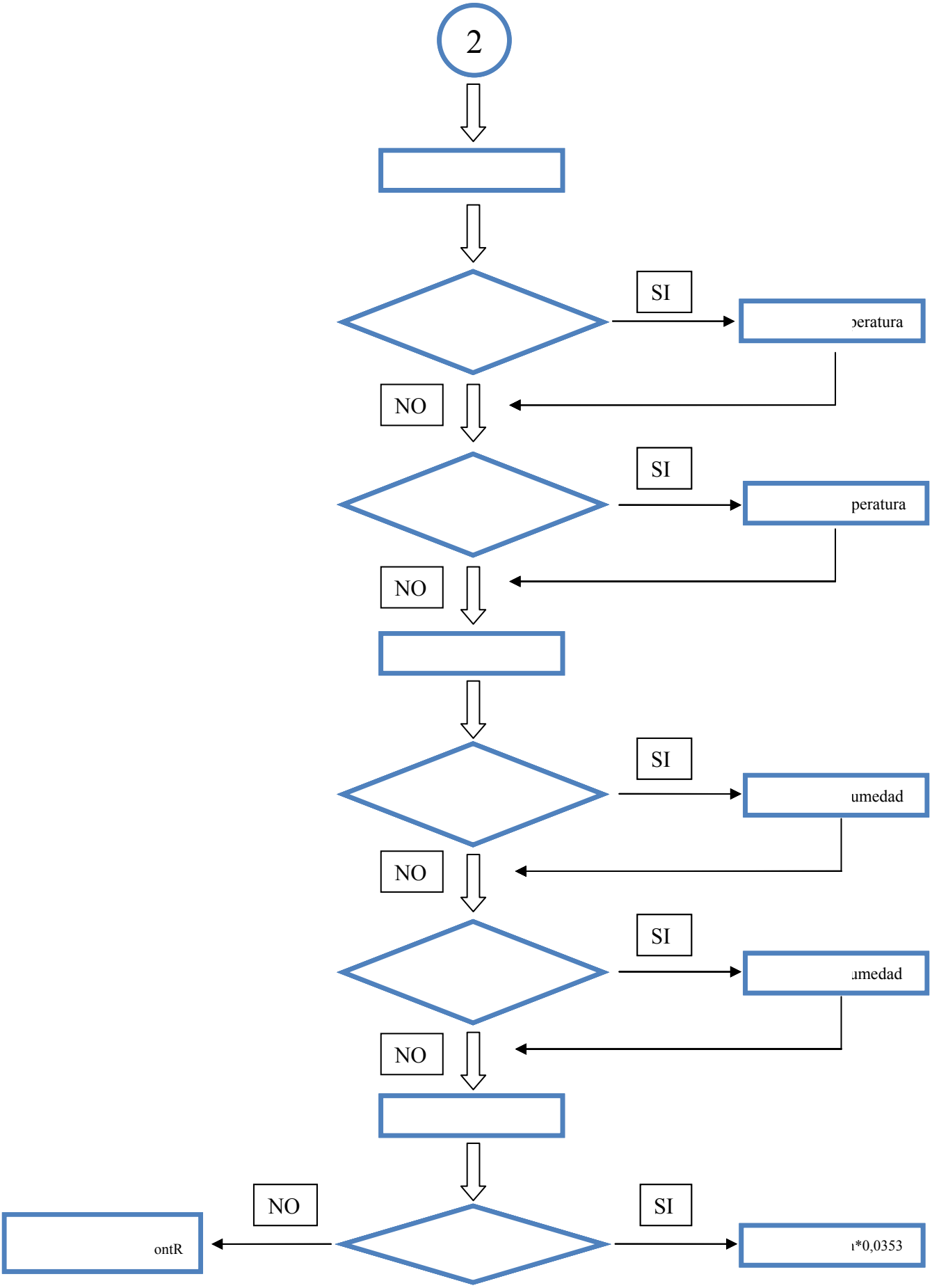
6.1. ALGORITMOS DE PROGRAMACIÓN

En éste apartado se va a explicar de forma breve y sencilla el código de programación. Para ello, comentaremos brevemente el código y para un completo entendimiento se muestran los algoritmos de programación.

La programación está basada en dos alarmas programadas. Una de ellas se activa cada media hora, y su función es la de recoger y almacenar los datos. La otra es que cada día a las 23.59 (ésta hora es programable), calcule con los datos adquiridos, el tiempo de riego estimado, y riegue durante ese tiempo.


Inicio
s
a
NO
NO
2
3
1


1
I
SI


2
SI
peratura
NO
SI
peratura
NO
SI
umedad
NO
SI
umedad
NO
NO
ontR
SI
*0,0353

6.2. RESUMEN DEL CÓDIGO DE PROGRAMACIÓN

6.2.1. ESTRUCTURA BÁSICA

Este apartado se divide en: Librerías, declaración de variables, void setup( ) y void loop( ).

6.2.1.1. LIBRERIAS

Para el correcto funcionamiento de la automatización, se han utilizado 3 librerías: Time.h, TimeAlarms.h y MsTimer2.h. Las cuales explicamos ahora, y añadimos su código en los anexos.

Time.h: Añade la funcionalidad de la hora normal de Arduino con o sin cronometraje de hardware externo. Permite que un sketch pueda obtener la hora y fecha: segundo, minuto, hora, día, mes y año. También proporciona el tiempo como un estándar de C, "time_t" y se puede calcular fácilmente según valores de tiempo compartido a través de diferentes plataformas.

En éste proyecto es la base del programa, ya que necesitamos saber en todo momento a qué hora, día y mes estamos. Es lo primero que se tiene que configurar al cargar el programa en la placa Arduino, para el correcto funcionamiento del código.

TimeAlarms.h: Es un complemento de la biblioteca de tiempo que hace que sea fácil realizar tareas en determinados momentos o después de intervalos específicos.

Las tareas programadas en un momento determinado del día se llaman "Alarms", tareas programadas después de un intervalo de tiempo transcurrido se llaman "Timers". Estas tareas se pueden crear para repetir o que se produzca una sola vez.

Se utiliza ésta librería, para realizar funciones repetitivas en el tiempo, como medir la temperatura, humedad, etc. cada media hora, y para regar cada 24h, mediante otra alarma.

MsTimer2.h: Es una interfaz pequeña y fácil de usar de la librería Timer2. Se llama MsTimer2 porque tiene una resolución de 1 milisegundo en el "Timer2".

Ésta librería sólo se utiliza para resolver el problema de la temporización del riego. Es decir sólo se inicia cuando comienza el riego, y se le asigna un valor en milisegundos, del tiempo que tiene que estar regando. Una vez pasado ese tiempo, se para.

6.2.1.2. VARIABLES

Una variable debe ser declarada y opcionalmente asignada a un determinado valor. En la declaración de la variable se indica el tipo de datos que almacenará (int, float, long…).

También puede ser declarada en el inicio del programa antes de setup( ), localmente a una determinada función e incluso dentro de un bloque como pueda ser un bucle. El sitio en el que la variable es declarada determina el ámbito de la misma. Una variable global es aquella que puede ser empleada en cualquier función del programa.

En éste proyecto se pueden dividir en 5 tipos de variables, las cuales pueden ser int, float o long. Normalmente para un rápido funcionamiento del código, se intenta no utilizar los tipos de variables float, ya que son muy pesados, y al juntarlos con los cálculos matemáticos, el programa se vuelve lento. Pero en éste proyecto, al tener que calcular una fórmula con datos precisos (hay valores que precisan de cuatro decimales de resolución), y no ser importante la velocidad de cálculo, se tienen que utilizar las variables de tipo float.

-E/S de los pines del Arduino. Variables que indican que pines del Arduino van a ser de entrada o de salida, y que nombre van a tener.

```
int analogPinTemp = A0; //Entrada analógica para la temperatura.
```

-Contadores. Tenemos dos tipos, los contadores de pulsos utilizados para medir la precipitación y la velocidad del viento, y los contadores utilizados en el programa para hacer medias de diferentes valores.

```
long contPulsosUz = 0; //Contador de pulsos de la velocidad del viento.
int contTemp = 0; //Contador de valores medios medidos de temperatura.
```

-Datos de Sensores. Son las variables que almacenan los datos obtenidos de los sensores (y re calculadas para la fórmula) y que se tienen que resetear cada vez que se riega.

```
int Rn = 0; // Rn es la variable que almacena la radiación solar.
int Uz = 0; //Uz la utilizamos para almacenar la velocidad del viento.
```

-Variables auxiliares para el cálculo de la ecuación de Penman-Monteith.

-Variables auxiliares para las funciones.

6.2.1.3. VOID SETUP( ) y LOOP( )

Estas dos funciones son la estructura básica de programación de Arduino. Es bastante simple y divide la ejecución en dos partes: setup y loop. Setup( ) constituye la preparación del programa y loop( ) es la ejecución. En la función setup( ) se incluye la declaración de variables y se trata de la primera función que ejecuta el programa. Esta función se ejecuta una única vez y es empleada para configurar el pinMode, e inicializar la comunicación serie. En éste proyecto también incluimos las alarmas y el set del reloj, mediante el setTime. La función loop( ) incluye el código a ser ejecutado continuamente, leyendo las entradas de la placa, salidas, etc. Como la repetición no se hace mediante el loop, sino que se consigue mediante las alarmas, en el void loop( ) sólo tendremos una línea de código, y es para que la librería de alarmas funcione correctamente seteandolas a 1 segundo.

Ejemplo void setup( ):

Figura 19: *Ejemplo de void setup( )*

Ejemplo de void loop( ):

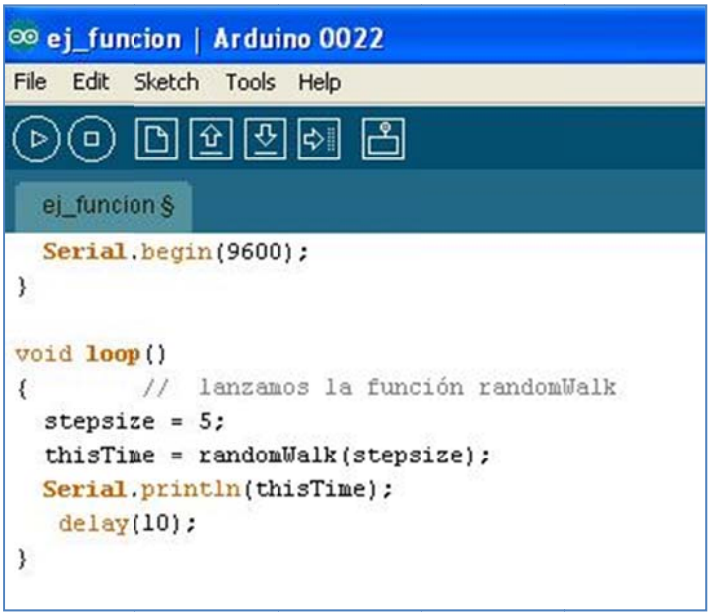


Figura 20: *Ejemplo de void loop( )*

6.2.2. LAS ALARMAS

Las alarmas son una parte fundamental de éste proyecto, ya que son las que crean la automatización del sistema. La función de estas alarmas es el de activar o llamar a alguna función en algún momento en el tiempo. El código en vez de apoyarse en el void loop( ) para realizar las operaciones repetitivas, utiliza la librería TimeAlarms.h para llamar a las funciones requeridas para la adquisición de datos, cada media hora, y otra cada 24 horas para el riego. Logrando así una precisión en el tiempo y sin miedo a perder segundos, por las interrupciones existentes.

Las alarmas se setean en el void setup( ), y son las siguientes:

- Alarm.alarmRepeat (23,59,59, ActivaRiego);
- Alarm.timerRepeat (1800, Fecha);

La primera es una alarma, para que se active el riego (en caso de que tenga que regar) todos los días a las 23:59 y 59 segundos.

La segunda, para repetir una alarma que active la función de adquisición de datos, cada 1800 segundos o lo que es lo mismo, cada media hora.

6.2.3. LAS FUNCIONES

Una función es un bloque de código identificado por un nombre y que es ejecutado cuando la función es llamada. La declaración de una función incluye en primer lugar el tipo de datos que devuelve la función (ej. int si lo que devuelve es un valor entero). Después del tipo de datos se especifica el nombre de la función y los parámetros de la misma. La siguiente función es empleada para realizar la lectura de la temperatura de un sensor que da una señal proporcional a la tª entre 0 y 1 voltios:

```
//************************************************************
//                    Entrada analógica de la TEMPERATURA
//************************************************************
//Mapeamos de 0 a 204, ya que la entrada será entre 0-1v.
int Termometro (){
  static int temperatura;
     sensorValueT = analogRead(analogPinTemp);              //
     temperatura = map (sensorValueT, 0, 204, -30, 70);    //
     return temperatura;
}
```

Figura 21: *Ejemplo de una función*

En el código de programación se puede diferenciar 5 tipos de funciones, que realizan diferentes operaciones: Adquisición de datos, almacenamiento de datos, proceso de datos (calcular la ecuación Penman-Monteith y el cálculo del tiempo de regado), activación/desactivación del actuador para el regado y el reseteo a cero, de las variables necesarias.

-Adquisición de datos: Como tenemos 5 sensores, utilizamos cinco funciones para recoger los datos enviados por los sensores. Éstas funciones son vacías (void), y lo que hacen es almacenar temporalmente los datos adquiridos en el momento en que son llamadas. Hay una excepción, y es la función Termometro( ), que devuelve el valor registrado por el termómetro en una variable entera (int).

Los sensores analógicos, son el de temperatura, humedad y radiación solar. Éstos se leen con unos pines que simulan ser analógicos, pero que en verdad son digitales y devuelven un valor entre 0 y 1023, para el piranómetro, y entre 0 y 204 para el termómetro e higrómetro. Para ello mapeamos las entradas según el output de cada sensor.

Las funciones son: Termometro( ), Higrometro( ) y Piranometro( ).

Los sensores de velocidad del viento y precipitaciones, son digitales, y envían pulsos. Por ello se han utilizado dos funciones que son llamadas al detectar una interrupción en los pines definidos y almacenan los pulsos contados. Así que éstas funciones son tanto de adquisición de datos, como almacenamiento.

Las funciones son: Pluviometro( ) y Anemometro( ).

-Almacenamiento de datos: Tenemos tres funciones que almacenan los datos. Las dos funciones comentadas en el apartado anterior, para la velocidad del viento y las precipitaciones, y la de los otros tres sensores.

Almacenamos la temperatura máxima, mínima y media. Humedad máxima y mínima, y la media de la radiación solar medida cada media hora.

La función es: DatosSensores( ).

-Proceso de datos: Se ha creado una función, que con los datos recogidos durante el día, calcula los milisegundos de riego necesario para las plantas, mediante la ecuación Penman-Monteith, a partir del valor de evapotranspiración diaria, y con la ecuación de riego explicado anteriormente.

La función es: Evapotranspiracion( ) y devuelve una variable float.

-Activación/Desactivación del actuador: Para activar el actuador y el circuito comience a regar, se necesita que se den varias condiciones. La primera de ellas, es que estemos en los meses definidos para el riego, que se han estimado desde abril a septiembre. La segunda, es que la temperatura a la hora del riego sea mayor que dos grados, para prevenir un riego con peligro de helada. Por último, comprobamos si se ha determinado un tiempo de riego mayor que cero, según los datos recogidos ese día.

Si se cumplen todas las condiciones, entonces activamos el actuador y a la vez, se setea un temporizador: el Timer2. El cual, activara la función de desactivar el actuador, en cuanto pase el tiempo de riego estimado. Una vez habilitado el riego, también llama a la función de resetear las variables, que explicamos a continuación.

Las funciones son: ActivaRiego( ) y DesactivaRiego( )

-Reseteo de variables: Esta función pone a cero a todas las variables que influyen en el cálculo del tiempo de riego. Son las variables que almacenan los datos de la temperatura, la humedad, velocidad del viento, precipitaciones y radiación solar.

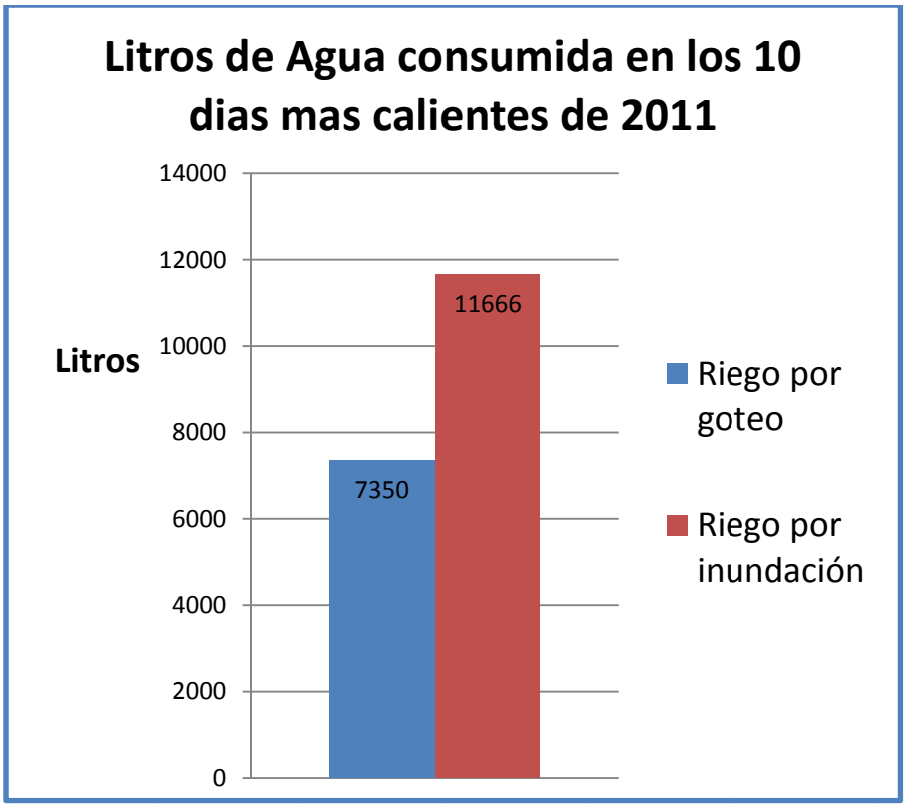
La función se llama: resetvariables( ).

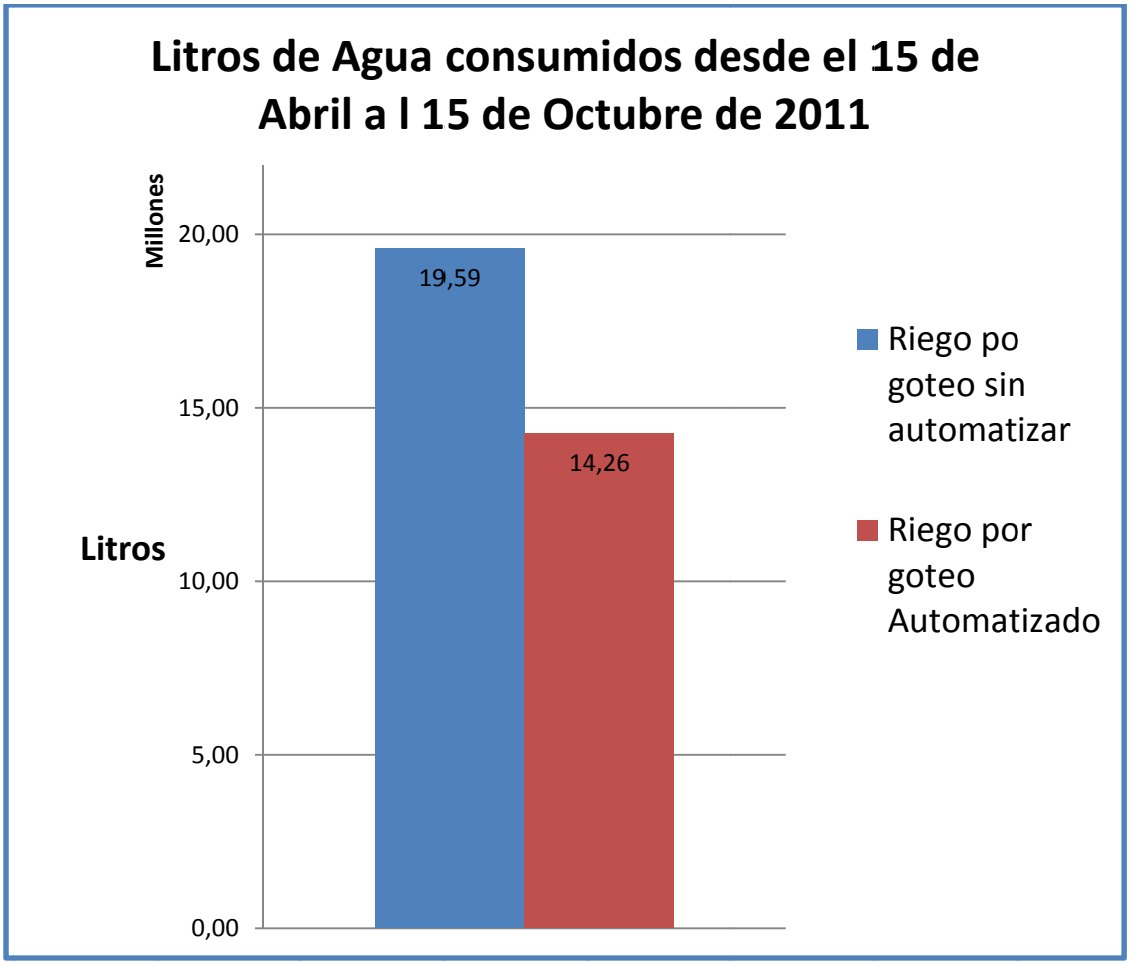
7. CONCLUSIONES

7.1. CONCLUSIONES FINALES

Este proyecto ha conseguido la optimización del riego de manera totalmente automática, mediante la aplicación del método Penman-Monteith, que asegura los mejores resultados del riego superficial, tal y como se había planteado.

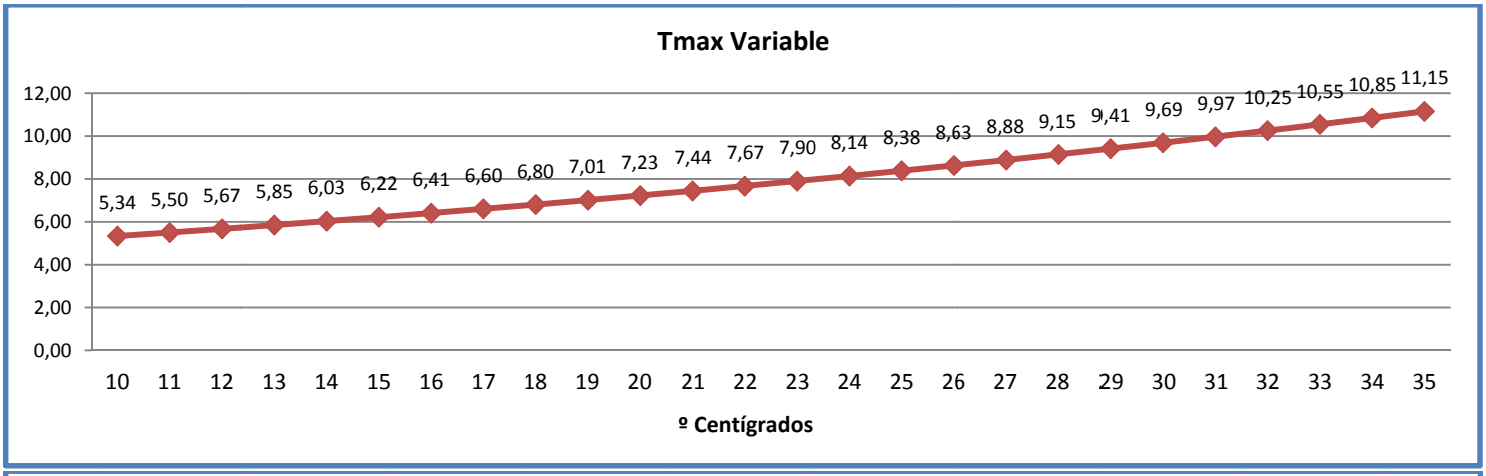
Con este sistema se ha conseguido un ahorro de agua de entre un 20% y un 30% dependiendo del criterio de riego del agricultor, si lo comparamos con un riego por goteo sin automatizar. Comparándolo con un riego por inundación, se consigue un ahorro de entre 35% y 40%.

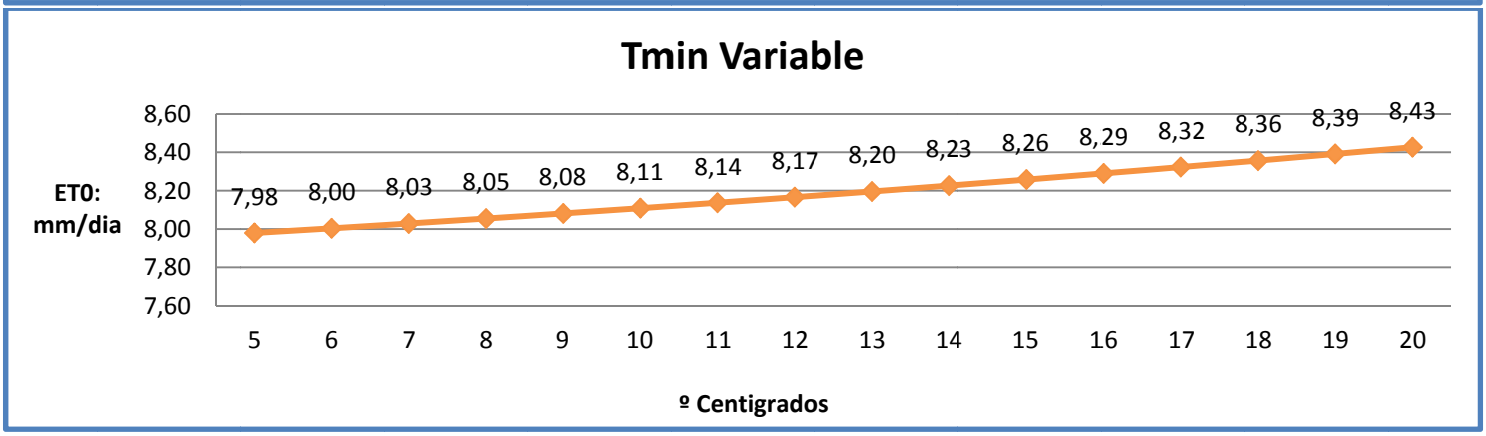


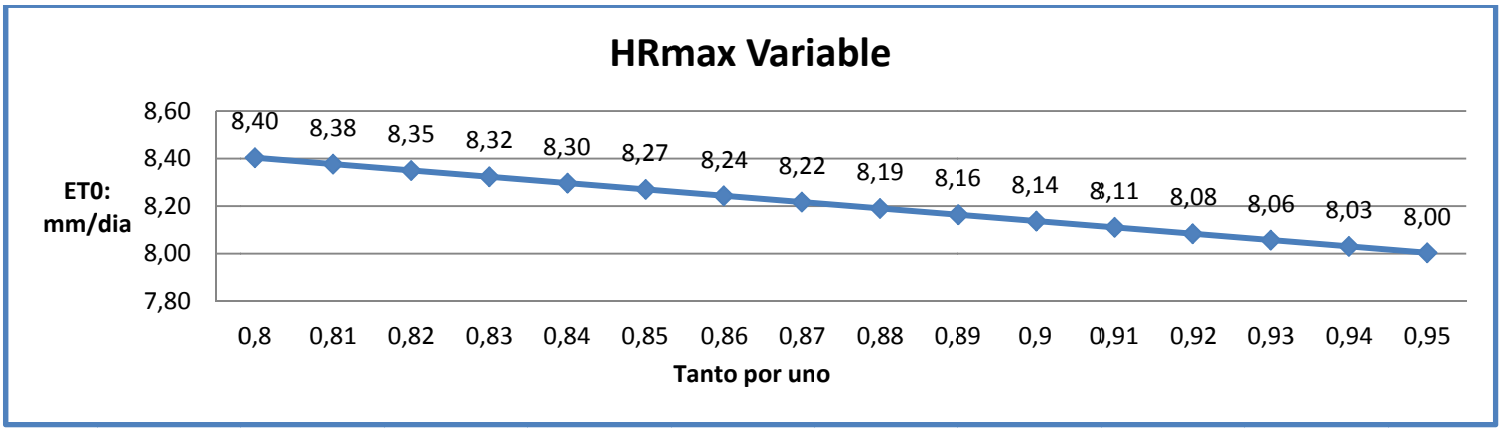


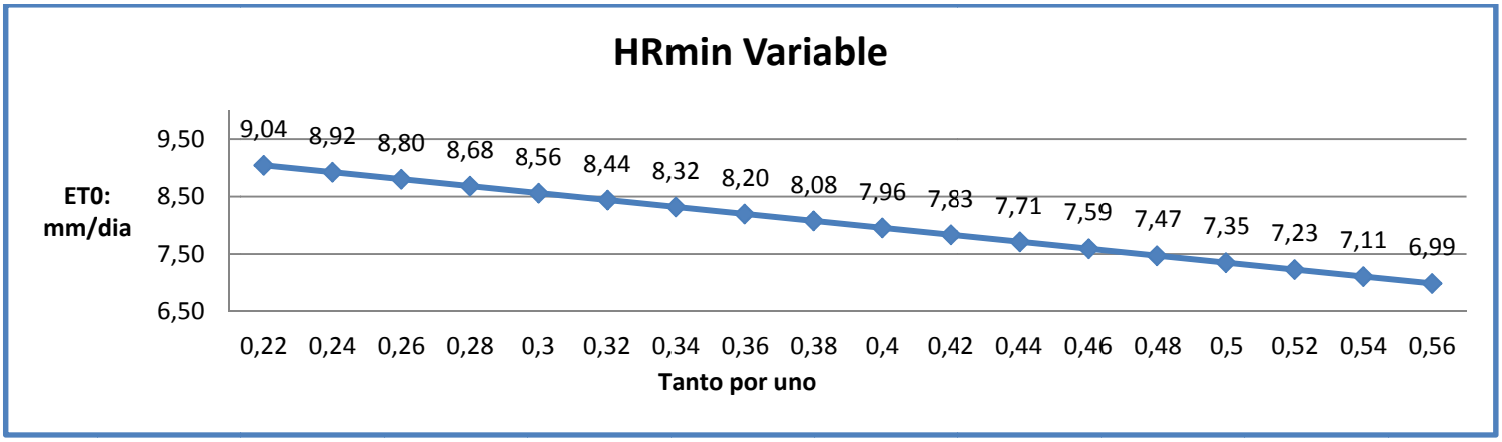
Además de un código de automatización, se ha realizado un código con diferentes valores meteorológicos, para poder obtener así datos estadísticos, y ver que sensores son los más influyentes en el tiempo de riego. Como era de esperar la temperatura máxima y radiación solar son los más influyentes. En cambio, la humedad y la velocidad del viento, no influencian tanto en el tiempo de riego, por lo que se podrían prescindir estos dos sensores, rebajando así los costes de la instalación.

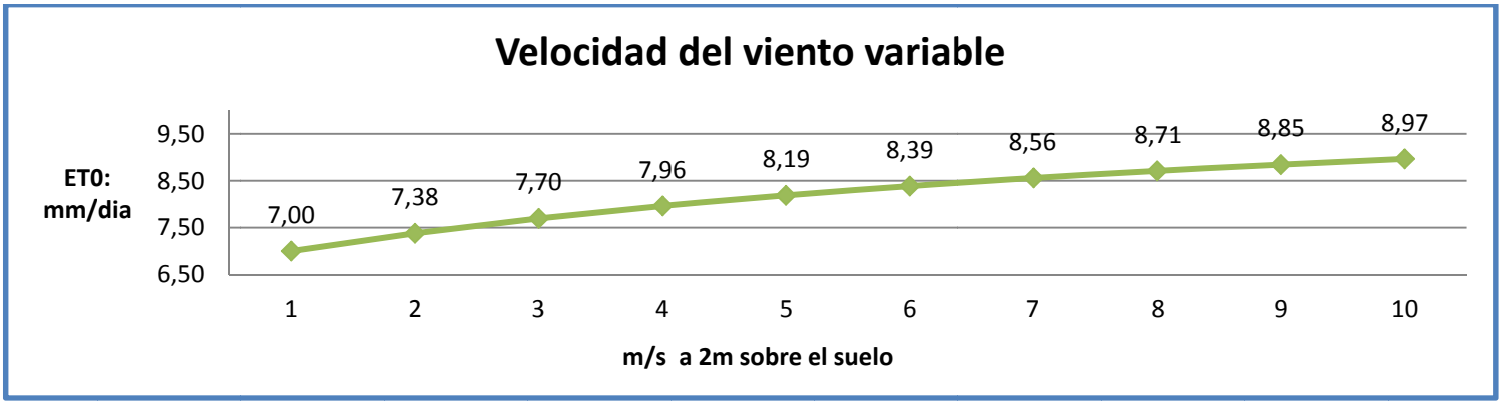
A continuación se va a presentar una serie de gráficos que presentan los diferentes valores de la ET0 según se va variando solo una variable (ej: se varía solo la temperatura máxima, y las demás, temperatura mínima, humedad máxima y mínima, velocidad del viento y radiación solar se mantienen constantes).

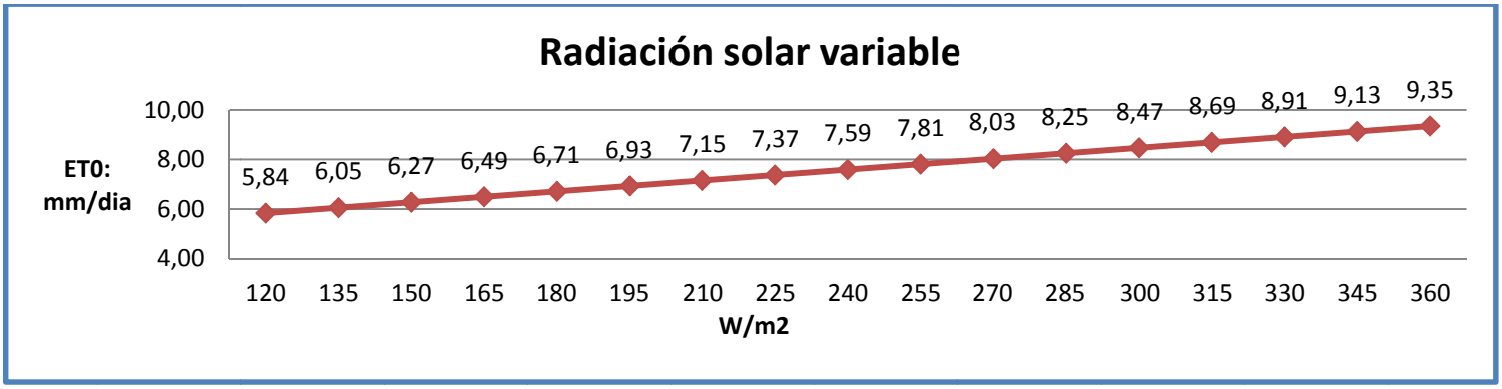

Tmax Variable
12,00
10,00
8,00
6,00
4,00
2,00
0,00
5,34 5,50 5,67 5,85 6,03 6,22 6,41 6,60 6,80 7,01 7,23 7,44 7,67 7,90 8,14 8,38 8,63 8,88 9,15 9,41 9,69 9,97 10,25 10,55 10,85 11,15
10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
º Centígrados


Tmin Variable
ETO: mm/dia
8,60
8,40
8,20
8,00
7,80
7,60
7,98 8,00 8,03 8,05 8,08 8,11 8,14 8,17 8,20 8,23 8,26 8,29 8,32 8,36 8,39 8,43
5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
º Centigrados


HRmax Variable
ETO: mm/dia
8,60
8,40
8,20
8,00
7,80
8,40 8,38 8,35 8,32 8,30 8,27 8,24 8,22 8,19 8,16 8,14 8,11 8,08 8,06 8,03 8,00
0,8 0,81 0,82 0,83 0,84 0,85 0,86 0,87 0,88 0,89 0,9 0,91 0,92 0,93 0,94 0,95
Tanto por uno







En opinión del autor de este proyecto, se ha podido trabajar en el aspecto que se deseaba potenciar, que era el de la programación. Debido al número de horas dedicadas a este fin, se han logrado unos resultados de acuerdo con las expectativas iniciales y, por otra parte, se ha conseguido una familiarización con el uso de un dispositivo con el cual no se había operado anteriormente.

7.2. ASPECTOS A CONSIDERAR EN UN FUTURO

Los puntos a mejorar en un futuro si se desea seguir con éste proyecto, serían el de la ampliación de componentes, como por ejemplo una pantalla LCD para poder visualizar diferentes datos. Se podría implementar un circuito para poder elegir diferentes tipos de riegos dependiendo de los cultivos. Añadiendo una memoria, se podría almacenar datos.

En caso de una ampliación de tipos de cultivos, se tendría que añadir las electroválvulas correspondientes. Con el relé actual podríamos abrir perfectamente la bomba de presión y con las electroválvulas controlar el riego específico para cada planta.

8. BIBLIOGRAFÍA

[1]Página oficial del microcontrolador Arduino: www.arduino.cc/es/

[2]Página para descargar software para dibujar los esquemas electrónicos: http://fritzing.org/projects/

[3]Página catálogo de Datasheet, para el microcontrolador utilizado por el Arduino: http://www.datasheetcatalog.net/es/datasheets_pdf/A/T/M/E/ATMEGA168.shtml

[4]Archivo descargable de la evapotranspiración del cultivo según la FAO: ftp://ftp.fao.org/docrep/fao/009/x0490s/x0490s00.pdf

[5]Página del departamento de Desarrollo Rural, Industria, Empleo y Medio Ambiente - Gobierno de Navarra, para adquisición de datos meteorológicos:

http://meteo.navarra.es/estaciones/estacion.cfm?IDestacion=405

[6]Artículo de las necesidades hídricas en cultivos agrícolas:

http://meteo.navarra.es/estaciones/estacion.cfm?IDestacion=405

[7]Catálogo de la marca Plasgot para electroválvulas:

http://www.plasgot.com/ELECTROVALVULAS.HTM

[8]Catálogo de la marca Plasgot para goteros:

http://www.plasgot.com/GOTEROS.HTM

[9]Catálogo de la casa Geonica para sensores de temperatura y humedad:

http://www.geonica.com/prod/80/138/Sensores-Meteorologicos-e-Hidrologicos/Temperatura-y-Humedad-del-Aire/index.html

[10]Catálogo de la casa Geonica para sensores de radiación solar:

http://www.geonica.com/prod/83/141/Sensores-Meteorologicos-e-Hidrologicos/Radiacion-solar-tipo-fotovoltaico/index.html

[11]Catálogo de la casa Geonica para sensores de precipitaciónes:

http://www.geonica.com/prod/82/140/Sensores-Meteorologicos-e-Hidrologicos/Precipitacion/index.html

[12]Catálogo de la casa Geonica para sensores de velocidad del viento:

http://www.geonica.com/prod/79/137/Sensores-Meteorologicos-e-Hidrologicos/Viento/index.html

9. ANEXOS

9.1. ANEXO I: LIBRERIAS

9.1.1. Librería Time

```
/* DateStrings.cpp
 * Definitions for date strings for use with the Time library
 *
 * No memory is consumed in the sketch if your code does not call any of the string methods
 * You can change the text of the strings, make sure the short strings are each exactly 3 characters
 * the long strings can be any length up to the constant dt_MAX_STRING_LEN defined in Time.h
 *
 */

#include <avr/pgmspace.h>
#include "Time.h"

// the short strings for each day or month must be exactly dt_SHORT_STR_LEN
#define dt_SHORT_STR_LEN  3 // the length of short strings

static char buffer[dt_MAX_STRING_LEN+1];  // must be big enough for longest string and the
terminating null

char monthStr1[] PROGMEM = "January";
char monthStr2[] PROGMEM = "February";
char monthStr3[] PROGMEM = "March";
char monthStr4[] PROGMEM = "April";
char monthStr5[] PROGMEM = "May";
char monthStr6[] PROGMEM = "June";
char monthStr7[] PROGMEM = "July";
char monthStr8[] PROGMEM = "August";
char monthStr9[] PROGMEM = "September";
char monthStr10[] PROGMEM = "October";
char monthStr11[] PROGMEM = "November";
char monthStr12[] PROGMEM = "December";

PGM_P monthNames_P[] PROGMEM =
{
   "",monthStr1,monthStr2,monthStr3,monthStr4,monthStr5,monthStr6,
         monthStr7,monthStr8,monthStr9,monthStr10,monthStr11,monthStr12
};

char monthShortNames_P[] PROGMEM = "ErrJanFebMarAprMayJunJulAugSepOctNovDec";

char dayStr0[] PROGMEM = "Err";
char dayStr1[] PROGMEM = "Sunday";
char dayStr2[] PROGMEM = "Monday";
char dayStr3[] PROGMEM = "Tuesday";
char dayStr4[] PROGMEM = "Wednesday";
char dayStr5[] PROGMEM = "Thursday";
```

```
char dayStr6[] PROGMEM = "Friday";
char dayStr7[] PROGMEM = "Saturday";

PGM_P dayNames_P[] PROGMEM = {
dayStr0,dayStr1,dayStr2,dayStr3,dayStr4,dayStr5,dayStr6,dayStr7};
char dayShortNames_P[] PROGMEM = "ErrSunMonTueWedThrFriSat";

/* functions to return date strings */

char* monthStr(uint8_t month)
{
   strcpy_P(buffer, (PGM_P)pgm_read_word(&(monthNames_P[month])));
          return buffer;
}

char* monthShortStr(uint8_t month)
{
  for (int i=0; i < dt_SHORT_STR_LEN; i++)
    buffer[i] = pgm_read_byte(&(monthShortNames_P[i+ (month*dt_SHORT_STR_LEN)]));
  buffer[dt_SHORT_STR_LEN] = 0;
  return buffer;
}

char* dayStr(uint8_t day)
{
  strcpy_P(buffer, (PGM_P)pgm_read_word(&(dayNames_P[day])));
  return buffer;
}

char* dayShortStr(uint8_t day)
{
  uint8_t index = day*dt_SHORT_STR_LEN;
  for (int i=0; i < dt_SHORT_STR_LEN; i++)
    buffer[i] = pgm_read_byte(&(dayShortNames_P[index + i]));
  buffer[dt_SHORT_STR_LEN] = 0;
  return buffer;
}
```

9.1.2. Librería TimeAlarms

```
extern "C" {
#include <string.h> // for memset
}

#include <WProgram.h>
#include "TimeAlarms.h"
#include "Time.h"

#define IS_ONESHOT  true   // constants used in arguments to create method
```

```
#define IS_REPEAT   false


//**************************************************************
//* Alarm Class Constructor

AlarmClass::AlarmClass()
{
 Mode.isEnabled = Mode.isOneShot = 0;
 Mode.alarmType = dtNotAllocated;
 value = nextTrigger = 0;
 onTickHandler = NULL;  // prevent a callback until this pointer is explicitly set
}

//**************************************************************
//* Private Methods


void AlarmClass::updateNextTrigger()
{
 if( (value != 0) && Mode.isEnabled )
 {
  time_t time = now();
  if( dtIsAlarm(Mode.alarmType) && nextTrigger <= time )   // update alarm if next trigger is not yet in
the future
  {
   if(Mode.alarmType == dtExplicitAlarm ) // is the value a specific date and time in the future
   {
    nextTrigger = value;  // yes, trigger on this value
   }
   else if(Mode.alarmType == dtDailyAlarm)  //if this is a daily alarm
   {
    if( value + previousMidnight(now()) <= time)
    {
     nextTrigger = value + nextMidnight(time); // if time has passed then set for tomorrow
    }
    else
    {
     nextTrigger = value + previousMidnight(time);  // set the date to today and add the time given in
value
    }
   }
   else if(Mode.alarmType == dtWeeklyAlarm)  // if this is a weekly alarm
   {
    if( (value + previousSunday(now())) <= time)
    {
     nextTrigger = value + nextSunday(time); // if day has passed then set for the next week.
    }
    else
    {
```

```
        nextTrigger = value + previousSunday(time);  // set the date to this week today and add the time
given in value
       }
     }
     else  // its not a recognized alarm type - this should not happen
     {
       Mode.isEnabled = 0;  // Disable the alarm
     }
   }
   if( Mode.alarmType == dtTimer)
   {
     // its a timer
     nextTrigger = time + value;  // add the value to previous time (this ensures delay always at least Value
seconds)
   }
  }
  else
  {
    Mode.isEnabled = 0;  // Disable if the value is 0
  }
}
```

9.1.3. Librería: MsTimer2

```
#include <MsTimer2.h>

unsigned long MsTimer2::msecs;
void (*MsTimer2::func)();
volatile unsigned long MsTimer2::count;
volatile char MsTimer2::overflowing;
volatile unsigned int MsTimer2::tcnt2;

void MsTimer2::set(unsigned long ms, void (*f)()) {
        float prescaler = 0.0;

#if defined (__AVR_ATmega168__) || defined (__AVR_ATmega48__) || defined
(__AVR_ATmega88__) || defined (__AVR_ATmega328P__) || (__AVR_ATmega1280__)
        TIMSK2 &= ~(1<<TOIE2);
        TCCR2A &= ~((1<<WGM21) | (1<<WGM20));
        TCCR2B &= ~(1<<WGM22);
        ASSR &= ~(1<<AS2);
        TIMSK2 &= ~(1<<OCIE2A);

        if ((F_CPU >= 1000000UL) && (F_CPU <= 16000000UL)) {        // prescaler set to 64
                TCCR2B |= (1<<CS22);
                TCCR2B &= ~((1<<CS21) | (1<<CS20));
                prescaler = 64.0;
        } else if (F_CPU < 1000000UL) {  // prescaler set to 8
                TCCR2B |= (1<<CS21);
```

```
                TCCR2B &= ~((1<<CS22) | (1<<CS20));
                prescaler = 8.0;
        } else { // F_CPU > 16Mhz, prescaler set to 128
                TCCR2B |= ((1<<CS22) | (1<<CS20));
                TCCR2B &= ~(1<<CS21);
                prescaler = 128.0;
        }
#elif defined (__AVR_ATmega8__)
        TIMSK &= ~(1<<TOIE2);
        TCCR2 &= ~((1<<WGM21) | (1<<WGM20));
        TIMSK &= ~(1<<OCIE2);
        ASSR &= ~(1<<AS2);

        if ((F_CPU >= 1000000UL) && (F_CPU <= 16000000UL)) {        // prescaler set to 64
                TCCR2 |= (1<<CS22);
                TCCR2 &= ~((1<<CS21) | (1<<CS20));
                prescaler = 64.0;
        } else if (F_CPU < 1000000UL) {  // prescaler set to 8
                TCCR2 |= (1<<CS21);
                TCCR2 &= ~((1<<CS22) | (1<<CS20));
                prescaler = 8.0;
        } else { // F_CPU > 16Mhz, prescaler set to 128
                TCCR2 |= ((1<<CS22) && (1<<CS20));
                TCCR2 &= ~(1<<CS21);
                prescaler = 128.0;
        }
#elif defined (__AVR_ATmega128__)
        TIMSK &= ~(1<<TOIE2);
        TCCR2 &= ~((1<<WGM21) | (1<<WGM20));
        TIMSK &= ~(1<<OCIE2);

        if ((F_CPU >= 1000000UL) && (F_CPU <= 16000000UL)) {        // prescaler set to 64
                TCCR2 |= ((1<<CS21) | (1<<CS20));
                TCCR2 &= ~(1<<CS22);
                prescaler = 64.0;
        } else if (F_CPU < 1000000UL) {  // prescaler set to 8
                TCCR2 |= (1<<CS21);
                TCCR2 &= ~((1<<CS22) | (1<<CS20));
                prescaler = 8.0;
        } else { // F_CPU > 16Mhz, prescaler set to 256
                TCCR2 |= (1<<CS22);
                TCCR2 &= ~((1<<CS21) | (1<<CS20));
                prescaler = 256.0;
        }
#endif

        tcnt2 = 256 - (int)((float)F_CPU * 0.001 / prescaler);

        if (ms == 0)
                msecs = 1;
```

```
	else
		msecs = ms;

	func = f;
}

void MsTimer2::start() {
	count = 0;
	overflowing = 0;
#if defined (__AVR_ATmega168__) || defined (__AVR_ATmega48__) || defined
(__AVR_ATmega88__) || defined (__AVR_ATmega328P__) || (__AVR_ATmega1280__)
	TCNT2 = tcnt2;
	TIMSK2 |= (1<<TOIE2);
#elif defined (__AVR_ATmega128__)
	TCNT2 = tcnt2;
	TIMSK |= (1<<TOIE2);
#elif defined (__AVR_ATmega8__)
	TCNT2 = tcnt2;
	TIMSK |= (1<<TOIE2);
#endif
}

void MsTimer2::stop() {
#if defined (__AVR_ATmega168__) || defined (__AVR_ATmega48__) || defined
(__AVR_ATmega88__) || defined (__AVR_ATmega328P__) || (__AVR_ATmega1280__)
	TIMSK2 &= ~(1<<TOIE2);
#elif defined (__AVR_ATmega128__)
	TIMSK &= ~(1<<TOIE2);
#elif defined (__AVR_ATmega8__)
	TIMSK &= ~(1<<TOIE2);
#endif
}

void MsTimer2::_overflow() {
	count += 1;

	if (count >= msecs && !overflowing) {
		overflowing = 1;
		count = 0;
		(*func)();
		overflowing = 0;
	}
}

ISR(TIMER2_OVF_vect) {
#if defined (__AVR_ATmega168__) || defined (__AVR_ATmega48__) || defined
(__AVR_ATmega88__) || defined (__AVR_ATmega328P__) || (__AVR_ATmega1280__)
	TCNT2 = MsTimer2::tcnt2;
#elif defined (__AVR_ATmega128__)
	TCNT2 = MsTimer2::tcnt2;
```

```
#elif defined (__AVR_ATmega8__)
        TCNT2 = MsTimer2::tcnt2;
#endif
        MsTimer2::_overflow();
}
```

9.2. ANEXO II: CÓDIGO DE PROGRAMACIÓN

```
#include <Time.h>
#include <TimeAlarms.h>
#include <MsTimer2.h>

//*****************************************************************
//                              V A R I A B L E S
//*****************************************************************
//Variables para los pines E/S.
int analogPinTemp = A0;  //Entrada analogica del sensor de TEMPERATURA
int analogPinHR = A1;    //Entrada analogica del sensor de HUMEDAD
int analogPinRn = A2;    //Entrada analogica del sensor de RADIACIÓN SOLAR
int digiPinEValv = 5;    //Salida digital para la BOMBA PRESION

// Variables para el Anemómetro
long contPulsosUz = 0;        //Contador pulsos de viento
float Uz = 0.0000;            //Variable de velocidad del viento a 2m

// Variables para el Pluviómetro
int contPulsosPr = 0;         //Contador pulsos de Precipitaciones
float nPrdia = 0.0000;        //Variable de las precipitaciones diarias (en mm)

// Variables para el Termómetro
int sensorValueT = 0;         //Varieble de lectura del sensor temperatura
float Temperatura1 = 0.0000;  //Variable almacenamiento del Termometro
int Tmax = -30.0000;          //Variable de la temperatura maxima
int Tmin = 70.0000;           //Variable de la temperatura minima
float Tmed = 0.0000;          //Variable de la temperatura media
int temp = 0;                 //Variable de la temperatura actual

// Variables para el Higrómetro
int sensorValueHR = 0;        //Variable de lectura del sensor de humedad
float HRmax = 0.0000;         //Variable de humedad relativa maxima
float HRmin = 100.0000;       //Variable de humedad relativa minima
```

```
// Variables para el Piranómetro
int sensorValueRn = 0;        //Variable de lectura del sensor de radiación solar
float Rn = 0.0000;            //Variable de radiacion solar neta
int contRn =0;                //Variable contadora de Rn

// Variables para utilizar en Evapotranspiración
float exp1 = 0.0000;
float exp2 = 0.0000;
float delta = 0.0000;         //Variable de pend. Presion de vapor
float vdelta = 0.0000;        //Variable delta
float vUz = 0.0000;           //Variable Uz
float vy = 0.0000;            //Variable Y
float vTmed = 0.0000;         //Variable de Tmed
float dpv = 0.0000;           //Variable deficit de vapor
float es = 0.0000;
float ea = 0.0000;
float expmax = 0.0000;
float expmin = 0.0000;
float e0tmax = 0.0000;
float e0tmin = 0.0000;
float e0 = 0.0000;
float ET0 = 0.0000;           //Variable de Evapotranspiracion de Referencia
float ET01 = 0;
float Tiemporiego = 0;
float TiemporiegoMin =0;
boolean Fechacorrecta = 0;

//**********************************************************************
//                                    Void setup ()
//**********************************************************************


void setup () {

  pinMode(analogPinTemp, INPUT);
  pinMode(analogPinHR, INPUT);
  pinMode(analogPinRn, INPUT);
  pinMode(digiPinEValv, OUTPUT);

  setTime(0,1,0,1,12,11);                    // Setea el reloj a las 12.01 del 1 de Diciembre
de 2011
  Alarm.alarmRepeat(23,59,59, ActivaRiego);          // Alarma de ABR, MAY, JUN, JUL,
AGOST y SEP (Verano) 8pm. Para regar
  Alarm.timerRepeat(1800, Fecha);                // Alarma repetitiva cada 30mins
(30min * 60seg = 1800seg) Para adquisicion datos
```

```
 attachInterrupt(1, Pluviometro, FALLING);    // Interrupcion para contar pulsos del
pluviometro con el falnco negativo
 attachInterrupt(0, Anemometro, FALLING);     // Interrupcion para contar pulsos del
anemometro con el falnco negativo

}


//*********************************************************************
//                                    Void loop ()
//*********************************************************************

void loop() {
 Alarm.delay(1000); // wait one second between clock display
 }


//*********************************************************************
//*                    ------------------------------------------------           *
//*                   |   L E C T U R A  D E  S E N S O R E S   |                  *
//*                    ------------------------------------------------           *
//*********************************************************************

//*********************************************************************
//                        ALMACENAMIENTO DE DATOS DE LOS SENSORES                *
//*********************************************************************

void DatosSensores(){
 Temperatura1 = Termometro ();
   if (Temperatura1 > Tmax) {Tmax = sensorValueT;};
   if (Temperatura1 < Tmin) {Tmin = sensorValueT;};
   Tmed = (Tmax + Tmin) / 2;
 Higrometro ();
   if (sensorValueHR > HRmax) {HRmax = sensorValueHR;}; //si la entrada es mayor que
la HRmax, almacenarla en HRmax
   if (sensorValueHR < HRmin) {HRmin = sensorValueHR;}; //si la entrada es menor que
la HRmin, almacenarla en HRmin

 Piranometro ();
   if (contRn == 0) {Rn = sensorValueRn * 0,0353;}
   else {Rn = (Rn + sensorValueRn * 0,0353) / contRn;};
   contRn++;   //Hacemos la media de la radiación solar
}

void Datoscont(){
```

```
  Uz = contPulsosUz * 8.8 / (1800 * 600 * 24);  //Los pulsos guardados en un dia los
convertimos en m/s.
  nPrdia = contPulsosPr / 10;              //Convertimos los pulsos en mm/dia.
}

//*****************************************************************
//                          Entrada digital de las PRECIPITACIONES                *
//*****************************************************************

void Pluviometro(){
   contPulsosPr++;   // Contaremos los pulsos en 24 horas y luego deduciremos las
Precipitaciones
}

//*****************************************************************
//                          Entrada digital de las Velocidad Viento                *
//*****************************************************************

void Anemometro (){
   contPulsosUz++;  // Contaremos los pulsos en 24 horas y luego deduciremos la
Velocidad media
}

//*****************************************************************
//                          Entrada analógica de la TEMPERATURA                *
//*****************************************************************
//Mapeamos de 0 a 204, ya que la entrada será entre 0-1v en vez de 0-5v
float Termometro (){
 static float temperatura;
   sensorValueT = analogRead(analogPinTemp);          //Leemos el valor del pin
asignado al Termometro
   temperatura = map (sensorValueT, 0, 204, -30, 70);  // Mapeamos el valor obtenido
de 0v - 5v -> 0 - 1023. Es decir entre -30º y 70º
   return temperatura;
}

//*****************************************************************
//                          Entrada analógica de la HUMEDAD                *
//*****************************************************************
//Mapeamos de 0 a 204, ya que la entrada será entre 0-1v en vez de 0-5v
void Higrometro (){
  sensorValueHR = analogRead(analogPinHR);
  sensorValueHR = map (sensorValueHR, 0, 204, 0, 100);
}
```

```
//*****************************************************************
//                    Entrada analógica de la RADIACIÓN SOLAR      *
//*****************************************************************
//
void Piranometro () {
   sensorValueRn = analogRead(analogPinRn);
   sensorValueRn = map (sensorValueRn, 0, 1023, 0, 2000);
}

//*****************************************************************
//                    Fórmula de la Evapotranspiración             *
//*****************************************************************

float Evapotranspiracion () { //Devuelve el el tiempo de riego en milisegundos.

  static float ET0Tiemporiego;
  // Llamamos a la recogida de datos, para saber: Tmax, Tmin, Tmed, HRmax, HRmin,
Rn, Uz y nPrdia.
  // Fórmula de la pendiente de la PRESIÓN DE VAPOR
  Tmed = (Tmax + Tmin)/2.0000;
  exp1 = Tmed + 237.3000;
  exp2 = (17.2700 * Tmed)  ;
  e0 = 0.6108 * exp (exp2 / exp1);
  delta = (4098 * e0) / sq(exp1);
  // Fórmulas V
  vUz = 1.0000 + 0.3400 * Uz;
  vdelta = delta / (delta + 0,0640 * vUz);
  vy = 0.0666 / (delta + 0,0640 * vUz);
  vTmed = Uz * 900.0000 / (Tmed + 273.3000);
  // Fórmulas DEFICIT PRESIÓN DE VAPOR
  expmax = 17.2700 * Tmax / (Tmax + 237.3000);
  expmin = 17.2700 * Tmin / (Tmin + 237.3000);
  e0tmax = 0.6108 * exp (expmax);
  e0tmin = 0.6108 * exp (expmin);
  es = (e0tmax + e0tmin)/2.0000;
  ea = (e0tmax * HRmin + e0tmin * HRmax)/2.0000;
  dpv = es - ea;

  ET0 = (Rn * 0.0353 * vdelta + vTmed * dpv * vy) ;
  // A la Evapotranspiración de referencia le restamos el nivel de agua recogido en un
dia.
  ET01 = ET0 - nPrdia;
  // Cálculo del tiempo de riego necesario segun ET01, en minutos
  TiemporiegoMin = ET01 / 22.2100 * 60; //
  // Cálculo del tiempo de riego necesario segun ET01, en milisegundos
```

```
 ET0Tiemporiego = TiemporiegoMin * 60000;

 return ET0Tiemporiego;
}

//*********************************************************************
//                ACTIVACIÓN/DESACTIVACION SALIDA ACTUADOR                *
//*********************************************************************

//-    Vamos a regar solo si estamos en los meses calurosos: Desde Abril a Septiembre
void Fecha () {
 if (month() >= 4 && month() <= 9 ) {
   DatosSensores();
   Fechacorrecta = 1;
 }
}
void ActivaRiego () {
 temp = Termometro ();
 if (temp >= 2 && Fechacorrecta == 1) {
   Tiemporiego = Evapotranspiracion ();
    if (Tiemporiego > 0){
      digitalWrite(digiPinEValv, HIGH);
      MsTimer2::set(Tiemporiego, DesactivaRiego);
      MsTimer2::start();
    }
 }
 resetvariables();
}

void DesactivaRiego () {
 digitalWrite(digiPinEValv, LOW);
 MsTimer2::stop();
}
//*********************************************************************
//                                RESETEO DE LAS VARIABLES                *
//*********************************************************************
void resetvariables(){
     Uz = 0;              //Reseteamos la Velocidad del viento
     Tmed = 0;            //Reseteamos la Temperatura media
     HRmax = 0;           //Reseteamos la Humedad relativa máxima
     HRmin = 0;           //Reseteamos la Humedad relativa mínima
     Rn = 0;              //Reseteamos la Radiación solar
     contRn = 0;          //Reseteamos el contador de Rn
     contPulsosUz = 0;    //Reseteamos el contador de pulsos de Uz
     contPulsosPr = 0;    //Reseteamos el contador de pulsos de Pr
```

```
    Fechacorrecta = 0;          //Reseteamos la marca de la fecha correcta para riego
}
```

9.3. ANEXO III: HOJA DE CARACTERÍSTICAS

Precipitaciones:

GEONICA EARTH SCIENCES

Brochure nº 9737.0031

MODEL 52202
TIPPING BUCKET RAIN GAUGE

The Tipping Bucket Rain Gauge meets the specifications of the World Meteorological Organization (WMO).

The design uses a proven tipping bucket mechanism for simple and effective rainfall measurement.
The bucket geometry and material are specially selected for maximum water release, thereby reducing contamination and errors.

Catchment area of 200 cm2 and measurement resolution of 0.1 mm meet the recommendations of the WMO. Leveling screws and bull-seye level are built-in for easy and precise adjustment in the field. Measured precipitation is discharged through a collection tube for verification of total rainfall.

Model 52202 is heated for operation in cold temperatures. An unheated version, 52203, is available for use in moderate climates. To discourage birds from perching on the funnel rim, accessory bird wire assembly may be attached to the gauge.

SPECIFICATIONS:

Size:
18 cm dia. x 30 cm high, (39 cm high with mounting base)

Catchment Area:
200 cm2

Resolution:
0.1 mm per tip
0.2 mm per tip (optional)

Accuracy:
2% up to 25 mm/hr
3% up to 50 mm/hr

Output:
Magnetic reed switch (N.O.), rating 24VAC/DC 500mA

Operating Temperature:
-20°C to +50°C (heated)

Power:
18 Watts for heater only

Mounting:
Clamp for 1" (1.34" dia.) iron pipe
or 3 bolts on 160mm dia. circle

Other:
Leveling adjustment, thermostatic control for heater,intake screen

Ordering information	Model
TIPPING BUCKET RAIN GAUGE (HEATED)	52202
TIPPING BUCKET RAIN GAUGE (UNHEATED)	52203
BIRD WIRE ASSEMBLY	52250

GEONICA, S.A. - Alejandro Rodriguez, nº 22 - 28039 Madrid - Spain Tel. +34 91 450 51 18 Fax +34 91 459 46 14 e-mail: info@geonica.com www.geonica.com

Radiación Solar

GEONICA EARTH SCIENCES

Brochure n° 9737.0034

MODEL LP02
SOLAR RADIATION SENSOR

LP02 is a solar radiation sensor that can be applied for most common solar radiation observations. It complies with the latest ISO and WMO standards. The scientific name of this instrument is pyranometer. LP02 is a modern alternative for the so-called "star" or "black and white" pyranometers overcoming the problem of poor stability of the white reflective paint.

INTRODUCTION

LP02 serves to measure the solar radiation flux that is incident on a plane surface in W/m2 from a 180 degrees field of view (also called "global" solar radiation). Working completely passive, using a thermopile sensor, LP02 generates a small output voltage proportional to this flux. Contrary to photodiode-based- and "black and white" instruments LP02 has a spectrally flat response across the full solar spectrum.

Using LP02 is easy. For readout one only needs an accurate voltmeter that works in the millivolt range. To calculate the radiation level the voltage must be divided by the sensitivity; a constant that is supplied with each individual instrument. LP02 can directly be connected to most commonly used datalogging systems.

LP02 can be used for general meteorological observations, building physics, climate- and solar collector testing. A common application is for outdoor solar radiation measurements as part of a meteorological station. This application requires horizontal levelling; levelling feet (7) and a level (11) are included. The LP02 cable can easily be installed or replaced by the user.

Applicable standards are ISO 9060 and 9847, WMO (World Meteorological Organisation), and ASTM E824-94. LP02 can also be used for stability estimations according to EPA (EPA-454/R-99-005).

LP02 SPECIFICATIONS

ISO classification	: second class
Spectral range	: 305 to 2800 nm
Sensitivity (nominal)	: 15 µV/ W.m-2
Temperature range	: -40 to +80 oC
Range	: 0 to 2000 W.m-2
Temperature dependence	:< 0.1%/°C
Calibration traceability	: WRR

OPTIONS

Additional cable length x metres (add to 5m),
AMF 01 Albedometer Fixture (used with 2 x LP02)
AC100 / AC420 amplifiers, LI 18 hand held readout

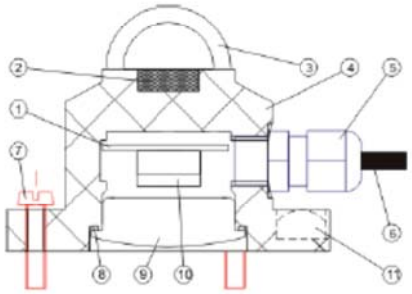


Figure 1 LP02 solar radiation sensor. **(1)** printed circuit board, **(2)** sensor, **(3)** glass dome, **(6)** cable, standard length 5 m, **(10)** screwed cable connection, **(9)** access for cable connection/ replacement.

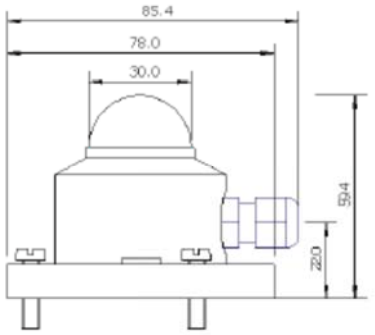


Figure 2 LP02 dimensions in mm. Standard cable length is 5 m. Cable can be installed / replaced by the user.

GEONICA, S.A. - Alejandro Rodriguez, n° 22 - 28039 Madrid - Spain Tel. +34 91 450 51 18 Fax +34 91 459 46 14 e-mail: info@geonica.com www.geonica.com

Amplificador AC420:

Hukseflux
Thermal Sensors

AC100 & AC420

HIGH ACCURACY MILLIVOLT AMPLIFIERS

AC100 and AC420 are high accuracy amplifiers that are specially designed for use with thermopile sensors. The primary application is with heat flux sensors, pyranometers, net-radiometers and pyrheliometers.

The specifications of both amplifiers are such that they can be used with the highest class of this type of instruments, while ensuring that no accuracy is lost. Also the amplifiers contain protection against transients (useful in meteorological applications). In this sense, AC100 and AC420 are unique. AC100 has a voltage output, AC420 has a 4 to 20 mA current output.
While the current output in some applications has the advantage of extra quality assurance for cable breaks (current will go to zero), it has the disadvantage that it cannot be used with negative input signals; so use with sensors that produce signals that are both positive and negative, such as net-radiometers and heat flux sensors in meteorological situations, is not possible.

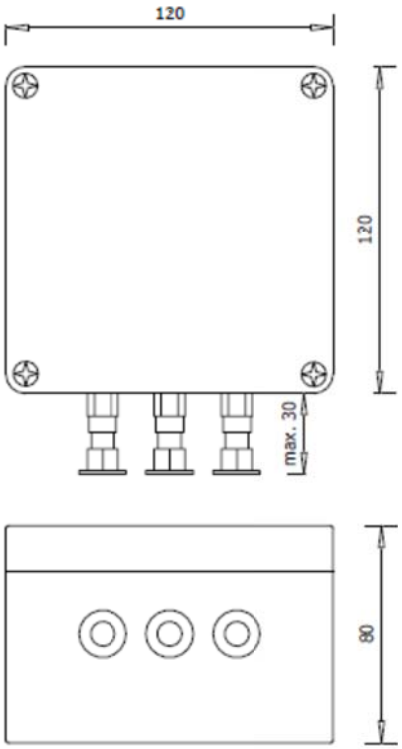


Figure 1 *AC100 and AC 420 dimensions. all dimensions are in millimeters.*

AC100 & AC420 FEATURES & BENEFITS

- Extremely low zero offset and range drift over the entire temperature range.
- Adaptable sensitivity, by do-it-yourself mounting of ordinary metal film resistors
- Possibility of feeding through extra wires, e.g. from temperature sensors or more voltage signals that are led through the same cable as the signal that must be amplified.
- Can be used with the highest accuracy lowest signal sensors without loss of accuracy.
- Suitable for outdoor installation.

OPTIONS

MA 220 mains adapter for 110/ 220 VAC

See also NAM01 nanovolt amplifier

SPECIFICATIONS

See table 1 on the next page

	AC100	AC420
Typical sensors:	Heat flux plates, pyranometers net –radiometers, pyrgeometers pyrheliometers, albedometers. Connectors for wiring of additional temperature sensors are available inside AC100 housing	Pyranometers, pyrheliometers albedometers, heat flux plates in applications with only one-way heat flux. Connectors for wiring of additional temperature sensors are available inside AC420 housing
Minimal full scale input:	+/- 3 mV	0 to + 3 mV
Maximum full scale input:	1000 mV	1200 mV
Output:	Voltage between +3 and –3 Volt	4 –20 mA current loop (to 3.5 mA)
Standard Amplification (A):	200	1 mA/mV
Do-it –yourself adjustment of A by resistor R:	A = (2*(50.000/R)+1 A from 1 to 1000. R can be composed of 1 to 3 separate resistors in series.	10 mV full scale, using 4 to 14 mA range R = 100* full scale input A from 16 mA/5 mV to 16 mA/1200 mV. R can be composed of 1 to 3 separate resistors in series.
Example of R calculation:	R of 1000 ohm plus 10 ohm gives an A of 100.	Heat flux sensor UT03 delivers 4.56 mV an 1000 W/m2 heat flux. R of 456 ohms delivers 4 mA at zero input and 14 mA at 1000 W/m². AC 420 is only to be used if the input voltage does not reverse / change sign.
Specifications of R:	1%, 50 ppm metal film resistor	1%, 50 ppm metal film resistor
Input impedance:	1 Mohm	1 Mohm
Ambient temperature range:	-20 to + 50 °C	-20 to + 50 °C
Temperature range for storage:	-30 to + 70 °C	-30 to + 70 °C
Zero drift at output:	< 0.05 mV/°C	< 0.25 m A/°C
Range drift at output:	< 20 ppm/°C	< 20 ppm/°C
Initial accuracy:	0.1%	0.1%
Power supply:	8 – 24 VDC	10 – 24 VDC
Supply current:	< 20 mA	n.a.
Loop voltage to output ratio:	n.a.	< 10 mA / V
Output impedance:	< 10 ohm	n.a.
Maximum load:	< 1 mA	n.a.
Response time:	< 1 s	< 1 s
Connection:	Swivels for cables from 4 to 6 mm diameter	Swivels for cables from 4 to 6 mm diameter
Input protection:	Protected against static discharge, reverse power	Protected against static discharge, reverse power

Table 1 *AC100 and AC420 specifications*

Version 0615 (page 2 of 2) Hukseflux reserve the right to alter specifications without prior notice
HUKSEFLUX™ THERMAL SENSORS . WWW.HUKSEFLUX.COM . INFO@HUKSEFLUX.COM

Temperatura y Humedad:

GEONICA EARTH SCIENCES

Brochure nº 9737.0027

5031 SERIES
AIR TEMPERATURE AND RELATIVE HUMIDITY SENSORS

The Temperature and Relative Humidity probes **5031 Series** are ready for direct connection to the meteorological stations METEODATA-2000C & 3000C Series.

The following versions are avilable:


Model 41003 Multi-Plate Radiation Shield


Model 43502 Aspirated Radiation Shield

Modelo STH-5031: Air Temperature and Relative Humidity combined sensor. Both sensors are mounted on a Nylon body with connector.
This combined sensor is normally installed inside a naturally ventilated radiation shield Model 41003 or alternatively inside a motor aspirated protector Model 43502, which has to be powered at 12-14 VDC (AC adaptor optional)

Model STA-5031: Air Temperature sensor mounted on a cylindrical Nylon body with connector. This sensor is normally installed inside a naturally ventilated radiation shield Model 41003 or alternatively inside a motor aspirated protector Model 43502.

Model STS-5031: Soil Temperature sensor, installed inside a short stainless steal cylindrical sheath mounted on a PVC body for thermal isolation. The cable output is waterproof.

Model STT-5031: Water Temperature sensor installed inside a short stainless steel cylindrical sheath, mounted on a PVC body. The probe and cable connection is submergible up to 20 meters water column.

Model SH-5031: Air Humidity sensor mounted on a Nylon body with connector. This sensor is normally installed inside a naturally ventilated radiation shield Model 41003 or alternatively inside a motor aspirated protector Model 43502.

SPECIFICATIONS:

	Temperature	Humidity
Sensor Type:	Resistor	Capacitive Polymer
Measuring range:	-30º to +70ºC (other ranges available)	0-100% HR
Accuracy	± 0,1ºC	± 3% (0-90% HR) ± 5% (90-98% HR)
Output signal	Resistive circuit	0-1V

GEONICA, S.A. - Alejandro Rodriguez, nº 22 - 28039 Madrid - Spain Tel. +34 91 450 51 18 Fax +34 91 459 46 14 e-mail: info@geonica.com www.geonica.com

Velocidad del viento:

GEONICA EARTH SCIENCES

Brochure nº 9737.0060

MODEL 27106 PROPELLER ANEMOMETER
MODEL 27005 GILL UVW ANEMOMETER

The Propeller Anemometer is a precision air speed measuring instrument.
The four blade helicoid propeller drives a miniature tach-generator.
Output voltage is directly proportional to the axial component of the wind speed.
Signal polarity indicates direction of propeller rotation. The Propeller Anemometer is also available with single or dual direction photo-chopper transducer.

The Propeller Anemometer is especially suited for measuring the vertical wind componenet. Since normal wind direction seldom exceeds ±30° from horizontal, propeller response can be calibrated to follow the co-sine law within 3% over this range.

The Gill UVW Anemometer measures the three orthogonal vectors of the wind: along wind component "U", across wind component "V", and vertical wind component "W".

The three Propeller Anemometer sensors are mounted at right angles on a common mast. Each sensor measures the wind component parallel with its axis of rotation. Propeller response as a function of wind angle approximates the ccsine law, allowing true wind velocity and direc-tion to be calculated.
The UVW Anemometer is designed for maximum sensitivity at lower wind speeds. Optional carbon fiber thermoplastic (CFT) propellers are available for applications requiring greater range and durability. Rain and dust are excluded from the precision bearings by a continuous duty air blower that keeps the instrument under a slight positive pressure.

Ordering information	Model
PROPELLER ANEMOMETER w/08274 EPS propeller	
With standard tach-generator transducer	27106
With optional single-direction photochopper	*ADD SUFFIX "D"*
With optional two-direction photochopper	*ADD SUFFIX "F"*
Optional 08254 CFT propeller	*ADD SUFFIX "T"*
UVW ANEMOMETER w/08274 EPS propellers	27005
Optional 08254 CFT propellers	*ADD SUFFIX "T"*
Optional 12 VDC blower (500 mA)	*ADD SUFFIX "J"*
SPARE PROPELLERS	
22 cm dia. Expanded Polystyrene Propeller (EPS)	08274
20 cm dia. Carbon Fiber Thermoplastic Propeller (CFT)	08254

SPECIFICATIONS:

08274 Expanded Polystyrene Propeller (EPS):

Diameter: 22 cm
Pitch: 29.4 cm wind passage per revolution
Range: 0-25 m/s (55 mph)
Threshold: 0.3 m/s (0.6 mph)*
Distance constant: 1.0 m (3.2 ft)*

08254 Carbon Fiber Thermoplastic Propeller (CFT):

Diameter: 20 cm
Pitch: 30.0 cm wind passage per revolution
Range: 0-35 m/s (80 mph)
Threshold: 0.4 m/s (0.8 mph)*
Distance constant: 2.1 m (6.9 ft)*

Signal output:

Standard tach-generator transducer - analog DC voltage proportional to wind component. Polarity indicates rotation direction.
1800 RPM (500 mV) = 8.8 m/s (19.7 mph).
Optional photochopper transducer - Voltage pulse with frequency proportional to wind component. 10 pulses per revolution. 1800 RPM (300 Hz) = 8.8 m/s (19.7 mph)

Power Requirement:

24 VAC/1.2 W for blower motor in UVW Anemome-ter (separate 11.5/230V transformer supplied). Propeller Anemometer with tachgenerator is self-powered. Optional photochopper transducer requires 5-15 VDC at 11 mA.

Dimensions:

UVW Anemometer: *overall height - 107 cm (42 in), each sensor projects 41 cm (16 in), base diameter 16 cm (6.2 in), mounting 34 mm (1.34 in) diameter (standard 1 in pipe).*
Propeller Anemometer: *overall length with mounting 43 cm (17 in), housing diameter 2.5 cm (1 in), mounts on standard 3/4 in pipe.*

Weight:

UVW Anemometer: *3.6 kg (7.9 lbs), shipping weight 8.2 kg (18 lbs)*
Propeller Anemometer: *0.5 kg (1.2 lbs)*

**Nominal values determined in accordance with ASTM stan-dard procedures.*
Specifications subject to change without notice.

GEONICA, S.A. - Alejandro Rodriguez, nº 22 - 28039 Madrid - Spain Tel. +34 91 450 51 18 Fax +34 91 459 46 14 e-mail: info@geonica.com www.geonica.com

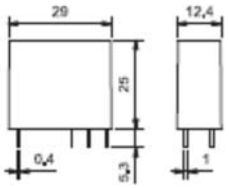
Actuador para la bomba:

finder — **Serie 40 - Mini-relé para circuito impreso enchufable 8 - 10 - 16 A**

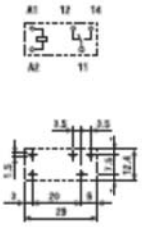
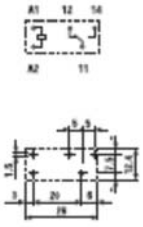
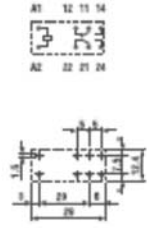
Características

Relé con 1 o 2 contactos
40.31 - 1 contacto 10 A (pas 3.5 mm)
40.51 - 1 contacto 10 A (pas 5 mm)
40.52 - 2 contactos 8 A (pas 5 mm)
Montaje en circuito impreso
- directo o en zócalo
Montaje en carril de 35 mm (EN 60715)
- en zócalos con bornes a platina o de conexión rápida

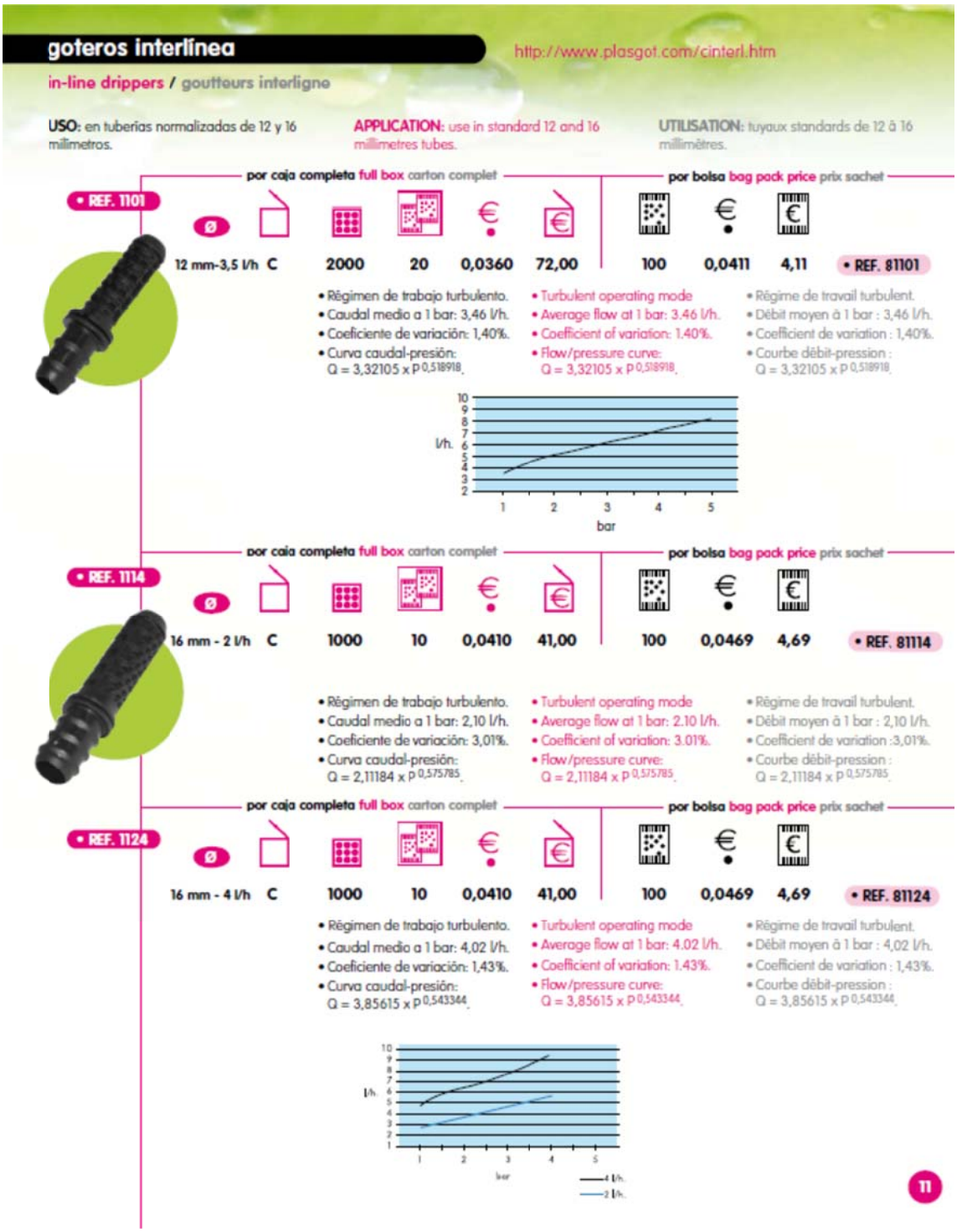
- Bobina DC (estándar o sensible) y bobina AC
- Contactos sin Cadmio
- 8 mm, 6 kV (1.2/50 µs) entre bobina y contactos
- UL Listing (combinaciones relé/zócalo)
- Estanco al flux: RT II estándar, (disponible en versión RT III)
- Zócalos serie 95
- Modulos de señalización y protección CEM
- Modulos temporizados serie 86



Para Cargas de Motores y "Pilot Duty" Homologadas por UL ver "Información Técnica General" página V

	40.31	40.51	40.52
			
	• Reticulado 3.5 mm • 1 contacto 10 A • Montaje en circuito impreso o en zócalo serie 95	• Reticulado 5 mm • 1 contacto 10 A • Montaje en circuito impreso o en zócalo serie 95	• Reticulado 5 mm • 2 contactos 8 A • Montaje en circuito impreso o en zócalo serie 95
			
	Vista parte inferior	Vista parte inferior	Vista parte inferior
Características de los contactos			
Configuración de contactos	1 contacto conmutado	1 contacto conmutado	2 contactos conmutados
Corriente nominal/Máx. corriente instantánea A	10/20	10/20	8/15
Tensión nominal/Máx. tensión de conmutación V AC	250/400	250/400	250/400
Carga nominal en AC1 VA	2500	2500	2000
Carga nominal en AC15 (230 V AC) VA	500	500	400
Motor monofásico (230 V AC) kW	0.37	0.37	0.3
Capacidad de ruptura en DC1: 30/110/220 VA	10/0.3/0.12	10/0.3/0.12	8/0.3/0.12
Carga mínima conmutable mW (V/mA)	300 (5/5)	300 (5/5)	300 (5/5)
Material estándar de los contactos	AgNi	AgNi	AgNi
Características de la bobina			
Tensión nominal V AC (50/60 Hz)	6 - 12 - 24 - 48 - 60 - 110 - 120 - 230 - 240		
de alimentación (U_N) V DC	5 - 6 - 7 - 9 - 12 - 14 - 18 - 21 - 24 - 28 - 36 - 48 - 60 - 90 - 110 - 125		
Potencia nominal en AC/DC/DC sens. VA (50 Hz)/W/W	1.2/0.65/0.5	1.2/0.65/0.5	1.2/0.65/0.5
Campo de funcionamiento AC	(0.8....1.1)U_N	(0.8....1.1)U_N	(0.8....1.1)U_N
DC/DC sensible	(0.73....1.5)U_N/(0.73....1.75)U_N	(0.73....1.5)U_N/(0.73....1.75)U_N	(0.73....1.5)U_N/(0.73....1.75)U_N
Tensión de mantenimiento AC/DC	0.8 U_N /0.4 U_N	0.8 U_N /0.4 U_N	0.8 U_N /0.4 U_N
Tensión de desconexión AC/DC	0.2 U_N /0.1 U_N	0.2 U_N /0.1 U_N	0.2 U_N /0.1 U_N
Características generales			
Vida útil mecánica AC/DC ciclos	10 · 10^6/20 · 10^6	10 · 10^6/20 · 10^6	10 · 10^6/20 · 10^6
Vida útil eléctrica con carga nominal AC1 ciclos	200 · 10^3	200 · 10^3	100 · 10^3
Tiempo de respuesta: conexión/desconexión ms	7/3 - (12/4 sensible)	7/3 - (12/4 sensible)	7/3 - (12/4 sensible)
Aislamiento entre bobina y contactos (1.2/50 µs) kV	6 (8 mm)	6 (8 mm)	6 (8 mm)
Rigidez dieléctrica entre contactos abiertos V AC	1000	1000	1000
Temperatura ambiente °C	-40....+85	-40....+85	-40....+85
Categoría de protección	RT II**	RT II**	RT II**
Homologaciones (según los tipos)			

Goteros:



goteros interlínea http://www.plasgot.com/cinterl.htm

in-line drippers / goutteurs interligne

USO: en tuberías normalizadas de 12 y 16 milímetros. **APPLICATION:** use in standard 12 and 16 millimetres tubes. **UTILISATION:** tuyaux standards de 12 à 16 millimètres.

	por caja completa full box carton complet					por bolsa bag pack price prix sachet			
	Ø	□	▦	€	€	▦	€	€	
• REF. 1101	12 mm-3,5 l/h	C	2000	20	0,0360	72,00	100	0,0411	4,11	• REF. 81101
• REF. 1114	16 mm - 2 l/h	C	1000	10	0,0410	41,00	100	0,0469	4,69	• REF. 81114
• REF. 1124	16 mm - 4 l/h	C	1000	10	0,0410	41,00	100	0,0469	4,69	• REF. 81124

- Régimen de trabajo turbulento.
- Caudal medio a 1 bar: 3,46 l/h.
- Coeficiente de variación: 1,40%.
- Curva caudal-presión: $Q = 3{,}32105 \times P^{0{,}518918}$.

- Turbulent operating mode
- Average flow at 1 bar: 3.46 l/h.
- Coefficient of variation: 1.40%.
- Flow/pressure curve: $Q = 3{,}32105 \times P^{0{,}518918}$.

- Régime de travail turbulent.
- Débit moyen à 1 bar : 3,46 l/h.
- Coefficient de variation : 1,40%.
- Courbe débit-pression : $Q = 3{,}32105 \times P^{0{,}518918}$.

- Régimen de trabajo turbulento.
- Caudal medio a 1 bar: 2,10 l/h.
- Coeficiente de variación: 3,01%.
- Curva caudal-presión: $Q = 2{,}11184 \times P^{0{,}575785}$.

- Turbulent operating mode
- Average flow at 1 bar: 2.10 l/h.
- Coefficient of variation: 3.01%.
- Flow/pressure curve: $Q = 2{,}11184 \times P^{0{,}575785}$.

- Régime de travail turbulent.
- Débit moyen à 1 bar : 2,10 l/h.
- Coefficient de variation :3,01%.
- Courbe débit-pression : $Q = 2{,}11184 \times P^{0{,}575785}$.

- Régimen de trabajo turbulento.
- Caudal medio a 1 bar: 4,02 l/h.
- Coeficiente de variación: 1,43%.
- Curva caudal-presión: $Q = 3{,}85615 \times P^{0{,}543344}$.

- Turbulent operating mode
- Average flow at 1 bar: 4.02 l/h.
- Coefficient of variation: 1.43%.
- Flow/pressure curve: $Q = 3{,}85615 \times P^{0{,}543344}$.

- Régime de travail turbulent.
- Débit moyen à 1 bar : 4,02 l/h.
- Coefficient de variation : 1,43%.
- Courbe débit-pression : $Q = 3{,}85615 \times P^{0{,}543344}$.

11

10. PRESUPUESTO

En este capítulo se harán dos presupuestos. El primero muestra los componentes explicados anteriormente en ésta memoria, y cuyo precio es demasiado elevado para llevar a cavo este proyecto. Por eso se incluye un segundo presupuesto, con unos sensores menos precisos, pero que reducirán un 96% el presupuesto total, siendo mas asequible la instalación del proyecto.

De éstos materiales sólo se han comprado los necesarios para crear la placa de pruebas, ya que con ella se puede simular a los otros sensores.

PRESUPUESTO 1:

Material	*Unidades*	*Precio (€)*
Arduino UNO	**1**	**30€**
Relé Finder serie 40	**1**	**4,68**
Cableado	10m	**15,73**
Sensor Temperatura y Humedad	**1**	**525**
Sensor Radiación Solar	**1**	**800**
Amplificador AC420	**1**	**250**
Sensor Viento	**1**	**570**
Sensor Precipitaciones	**1**	**550**
Componentes Placa pruebas	**1**	**6,29**
TOTAL		**2751,7 €**

PRESUPUESTO 2:

Material	*Unidades*	*Precio (€)*
Arduino UNO	**1**	**30€**
Relé Finder serie 40	**1**	**4,68**
Cableado	10m	**15,73**
Sensor Temperatura y Humedad	**1**	**6,97**
Fotorresistencia LDR	**1**	**1,85**
Sensor Viento	**1**	**29**
Sensor Precipitaciones	**1**	**8,50**
Componentes Placa pruebas	**1**	**6,29**
TOTAL		**103,02 €**

Como podemos ver la diferencia entre los dos presupuestos es enorme (de un 96%), ya que la primera tiene los mejores dispositivos existentes en el mercado, y la segunda los componentes básicos para poder montar y probar el montaje del proyecto.

11. PLANOS

11.1. ESQUEMA ELECTRÓNICO