UCA Universidad de Cádiz

Universidad de Cádiz

Proyectos fin de carrera de Ingeniería Técnica Industrial.

Electrónica Industrial.

Centro: ESCUELA POLITÉCNICA SUPERIOR DE ALGECIRAS

Titulación: Ingeniería Técnica Industrial. Electrónica Industrial.

Titulo: Estación meteorológica basada en sistema de adquisición de datos mediante PIC16F877/A e interfaz de usuario basado en entorno de programación gráfico Labview.

Autor: Alejandro J. Domínguez Hiniesta

Fecha: Febrero 2012

ESTACIÓN METEOROLÓGICA BASADA EN SISTEMA DE ADQUISICIÓN DE DATOS MEDIANTE PIC16F877/A E INTERFAZ DE USUARIO BASADO EN ENTORNO DE PROGRAMACIÓN GRÁFICO LABVIEW.

I.T.E.I. Ingeniería Técnica Industrial, esp. Electrónica Ind.

Febrero, 2012

Alumno: Alejandro J. Domínguez Hiniesta

Tutor: José Gabriel Ramiro Leo

ÍNDICE DE CONTENIDOS

ÍNDICE DE CONTENIDOS.

ÍNDICE DE FIGURAS.

ÍNDICE DE TABLAS.

1 INTRODUCCIÓN

1.1 Generalidades.

Desde siempre ha existido una estrecha relación entre el hombre y la climatología. La observación de los fenómenos climatológicos nos ha permitido actuar para adaptarnos al medio. Posiblemente el motivo más evidente sea el confort. Conocer la temperatura, la velocidad y la dirección del viento, la radiación solar, la humedad o la presión atmosférica puede sernos de mucha ayuda, pero existen otros muchos motivos de interés: información náutica y aeronáutica, estudios de viabilidad y mantenimiento para plantas solares y eólicas, etc.

El estudio de estas seis magnitudes con las técnicas y elementos de la industria actual supone la propuesta de este proyecto.

El presente Proyecto Final de Carrera se redacta con carácter de Trabajo para la obtención, por parte de quien lo suscribe, del título de Ingeniero Técnico en Electrónica Industrial. Si bien el trabajo versa sobre el diseño de una estación meteorológica, la finalidad del mismo no es otra que la formación y aplicación de técnicas y dispositivos altamente extendidos en la industria común, así como la demostración de las habilidades cognitivas adquiridas durante la formación académica universitaria. Por ello, se ha empleado un lenguaje de programación de bajo nivel para el PIC y se ha evitado utilizar sensores comerciales específicos para las medidas de las magnitudes meteorológicas sustituyéndose por dispositivos que cumplen con sus principios de funcionamiento básico.

1.2 Objetivos y planteamientos del trabajo.

El objetivo del presente proyecto es el diseño de una estación meteorológica para la observación, medición y registro de las magnitudes climatológicas más comunes. Para ello, se presenta un sistema de adquisición de datos mediante un microcontrolador PIC16F877/A y una interfaz de usuario basada en el entorno de programación gráfico LabView que permitirá la visualización, gestión y almacenamiento de datos de forma sencilla e intuitiva para el observador.

Las magnitudes a tratar son:

- ✓ Temperatura.
- ✓ Humedad.

- ✓ Presión Atmosférica.
- ✓ Velocidad del Viento.
- ✓ Dirección del Viento.
- ✓ Radiación Solar (Irradianza).

El planteamiento de trabajo para este proyecto pasa por los siguientes apartados:

- ✓ Estudio del microcontrolador.
- ✓ Estudio del lenguaje de programación ASSAMBLER (Ensamblador).
- ✓ Estudio, fabricación y montaje del equipo de programación del microcontrolador, basado en el modelo JDM.
- ✓ Estudio del entorno de programación gráfico LabView.
- ✓ Estudio de las características y funcionalidad de los sensores necesarios para el proyecto e implementación de los mismos según su principio de funcionamiento.
- ✓ Diseño y montaje del prototipo de pruebas de la estación meteorológica.
- ✓ Programación de la interfaz de usuario para la realización de pruebas y obtención de los objetivos marcados.

Este trabajo ha sido realizado por el alumno Alejandro Domínguez Hiniesta, bajo la supervisión del profesor D. José Gabriel Ramiro Leo, para la obtención del título de "Ingeniero Técnico en Electrónica Industrial"

1.3 Documentación (contenido documental).

El trabajo está constituido por los siguientes documentos:

- ✓ Introducción.
- ✓ Memoria.
- ✓ Esquemas y planos.
- ✓ Conclusiones y futuras líneas de trabajo.
- ✓ Bibliografía, artículos y direcciones web.
- ✓ Anexos.

2 MEMORIA

2.1 Introducción al microcontrolador.

Tengamos una ligera idea de lo que es un microcontrolador o no, es interesante saber que convivimos con ellos a diario. Tareas tan habituales como atender una llamada, conducir nuestro coche, lavar la ropa o escribir en el teclado de nuestro PC nos pone a mano de equipos gobernados por, al menos, un microcontrolador.

2.1.1 ¿Qué es un microcontrolador?

El microcontrolador es un circuito integrado programable que contiene todos los componentes necesarios para controlar el funcionamiento de una tarea determinada. Para ello dispone de los tres elementos básicos de una computadora: Memoria de almacenamiento (una memoria ROM en la que se almacena el programa, denominado firmware, que gobernará su funcionamiento y una memoria de acceso aleatorio RAM para almacenar datos). Unos puertos o líneas de entrada y salida mediante los cuales se comunicará con el entorno y una unidad lógica de control o unidad central de procesamiento CPU que coordina a todos estos bloques. En definitiva, se trata de un microprocesador con periféricos y memoria integrados en un único circuito integrado.

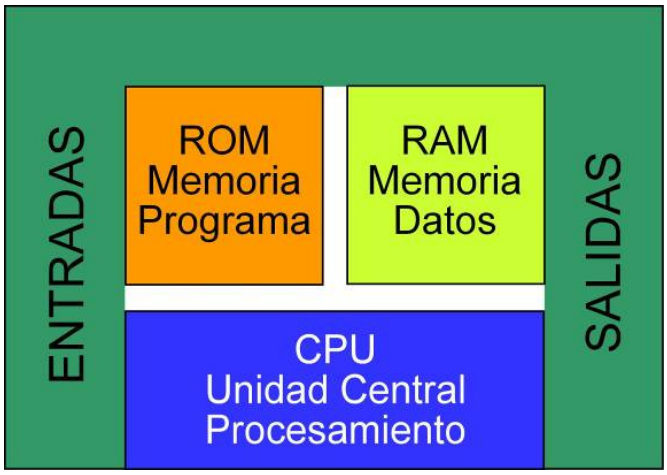


Ilustración 2.1: Esquema básico de un microcontrolador

Para dar respuesta a las necesidades de diseño, se ha creado una amplia gama de modelos en las que se han ido mejorando las características mediante la modificación del número de entradas y salidas, el aumento de las velocidades de trabajo, la incorporación de convertidores analógicos-digitales (ADC), moduladores de ancho de pulso (PWM), contadores (TIMERS), puertos de comunicación serial (USART), etc., ofreciendo al diseñador del sistema una multitud de posibilidades y características que permiten elegir el modelo más idóneo en función de las necesidades de cada proyecto.

2.1.2 El microcontrolador PIC

Fabricado por Microchip Technology Inc., el microcontrolador PIC (Programable Integrated Circuit) goza de una gran aceptación entre la comunidad de electrónicos, tanto a nivel de aficionado como a nivel profesional. Resumiendo brevemente los motivos de su popularidad podemos enumerar:

- ✓ Amplia gama de productos.

  Como se muestra en la figura 2 existen diferentes familias (buses de datos de 8, 16 y 32 bit de ancho) de dispositivos cuyas características se adecuan a las necesidades de los diferentes proyectos a abarcar.

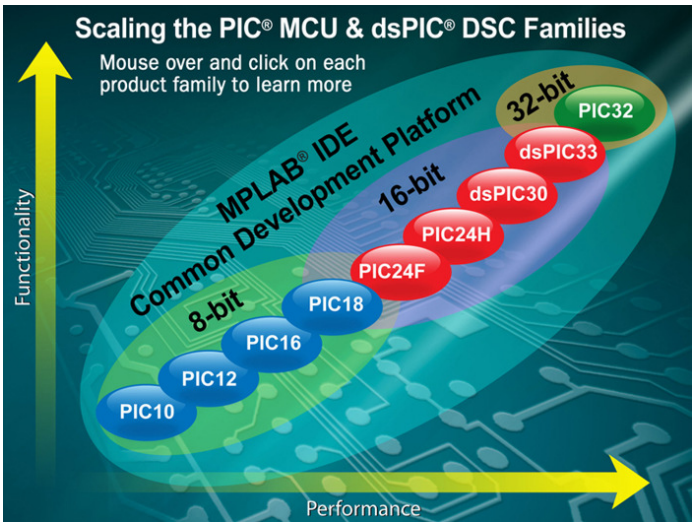


Ilustración 2.2: Familias de microcontroladores Microchip (www.microchip.com)

- ✓ Compatibilidad entre familias.

  La política de *Microchip Technology Inc.* en este respecto establece que los dispositivos de gama superior han de poder ejecutar programas diseñados para dispositivos de una gama inferior. Del mismo modo, el aprendizaje de uso de una gama inferior es válido para las sucesivas gamas superiores.

- ✓ Bajo coste.

  Su reducido coste le permite una alta divulgación, estableciéndolo como un elemento conocido y candidato a ser utilizado.

- ✓ Gran calidad y alta fiabilidad.

A pesar de su coste reducido, el PIC es un dispositivo de calidad, robusto y fiable que en condiciones normales y con un buen diseño del firmware puede permanecer trabajando en perfectas condiciones y con resultados fiables por mucho tiempo sin necesidad de mantenimiento.

- ✓ Consumo reducido

  Gracias a la tecnología CMOS los microprocesadores disponen de consumos muy reducidos. Para el caso del PIC empleado en este proyecto (16F877/A) los consumos típicos indicados en el datasheet son inferiores a 0,6mA para una alimentación de 3V y una frecuencia de trabajo de 4MHz.

- ✓ Tamaño reducido

  Su reducido tamaño y peso facilita su ubicación permitiendo reducir el tamaño y peso de los diseños. Esto supone un factor muy favorable particularmente en robótica y en equipos destinados a ser portados o a realizar desplazamientos.

- ✓ Documentación, software de programación y pruebas.

  A las características constructivas anteriormente descritas, hay que añadir la política de *Microchip Technology Inc.* de facilitar gratuitamente el acceso a información, software de programación y las herramientas de pruebas que pueden descargarse directamente de su página web, convirtiéndolo en un componente altamente extendido por su uso fácil, cómodo y rápido.

2.1.3 Microcontrolador PIC16F877.

El microcontrolador PIC16F877 de Microchip pertenece a la gama media de una familia de microcontroladores de 8 bits de bajo costo, con un procesador tipo RISC y segmentado, basado en una arquitectura tipo HARVARD y diseñado bajo tecnología CMOS.

La combinación de estas características de diseño da lugar a un dispositivo de bajo consumo, altamente eficiente en el uso de la memoria de datos y de programa, y por lo tanto en la velocidad de ejecución.

2.1.3.1 Arquitectura del microcontrolador PIC16F877.

La arquitectura hace referencia al modo en el que se diseña el funcionamiento de la CPU. Existen dos tipos de arquitecturas típicas a considerar. La arquitectura Von Neumann típica de los procesadores de propósito general y la arquitectura Harvard que es la adoptada por los microcontroladores PIC.

La gran diferencia entre ambas es que la arquitectura Von Neumann propone una única memoria donde se almacenan datos e instrucciones de programa, mientras que Harvard propone la separación de los datos y del programa haciendo uso de dos memorias diferenciadas que se comunican con la CPU mediante buses igualmente diferenciados.

A pesar de la dificultad que esto supone en su diseño por la necesidad de duplicar los buses de comunicación entre CPU y memorias, para determinadas aplicaciones en las que el flujo de instrucciones y de datos es más o menos el mismo, como es el caso de los microcontroladores PIC o dsPIC (controladores de señales digitales), la arquitectura Harvard repercute en ciertas mejoras respecto al rendimiento, evitando por ejemplo el denominado efecto "cuello de botella de Von Neumann" que consiste en la ralentización de la CPU debido a la saturación del bus de memoria que debe gestionar un gran número de datos e instrucciones de programa y potenciando de este modo el paralelismo en la ejecución de tareas.

2.1.3.2 Juego de instrucciones tipo RISC.

Atendiendo al tipo de instrucciones utilizadas por los procesadores, se pueden clasificar en los siguientes juegos de instrucciones:

- ✓ CISC: (Complex Instruction Set Computer) Computadores de juego de instrucciones complejo, que disponen de un repertorio de instrucciones elevado (unas 80), algunas de ellas muy sofisticadas y potentes, pero que como contrapartida requieren muchos ciclos de máquina para ejecutar las instrucciones complejas.

- ✓ RISC: (Reduced Instruction Set Computer) Computadores de juego de instrucciones reducido, en los que el repertorio de instrucciones es muy reducido (en nuestro caso 35), las instrucciones son muy simples y suelen

ejecutarse en un ciclo máquina. Además los RISC deben tener una estructura pipeline y ejecutar todas las instrucciones a la misma velocidad.

- ✓ SISC.(Specific Instruction Set Computer) Computadores de juego de instrucciones específico.

Podríamos decir que los procesadores tipo CISC, están basados en la microprogramación, es decir, cada instrucción es interpretada por un microprograma localizado en una sección de memoria en el circuito integrado del microprocesador. A su vez, las instrucciones compuestas se decodifican para ser ejecutadas por microinstrucciones almacenadas en una ROM interna. En esta tipología clásica las operaciones se realizan al ritmo de los ciclos de reloj, requiriéndose al menos un ciclo de reloj por instrucción.

En contra posición, la tipología RISC se basa en un conjunto de instrucciones muy reducidas de ejecución rápida que permiten a la CPU trabajar más rápido al utilizar menos ciclos de reloj para ejecutar las instrucciones. Además utiliza un sistema de direcciones no destructivas en RAM, esto significa que a diferencia de CISC, RISC después de realizar sus operaciones conserva en memoria los dos operandos y su resultado (en total tres direcciones), lo que facilita a los compiladores conservar llenos los 'pipelines' (conductos) de la CPU para utilizarlos concurrentemente y reducir la ejecución de nuevas operaciones y accesos a memoria. Y cada instrucción puede ser ejecutada en un solo ciclo de la CPU maximizando la velocidad y eficiencia del procesador.

Como contrapartida, la tipología de procesadores RISC precisa de compiladores que interpretan las instrucciones mientras que CISC las decodifica directamente mediante los microprogramas implementados en él.

2.1.3.3 Segmentación en la ejecución de instrucciones (“pipe-line”)

La segmentación es una técnica consistente en la descomposición de la ejecución de cada instrucción en varias etapas de segmentación (o segmento). Cada segmento actúa como un procesado diferente que entrega su salida como entrada del segmento siguiente. La finalización del primer segmento permite iniciar una segunda instrucción, permitiendo procesar diferentes instrucciones a la vez. Gracias a esta técnica puede ejecutarse una instrucción en un ciclo de instrucción o lo que es lo mismo cuatro ciclos de reloj, salvo en las instrucciones de salto que toman dos ciclos de instrucción.

Apoyándonos en la Ilustración 2.3y el ejemplo propuesto en la Tabla 2.1 reforzaremos el concepto de la segmentación.

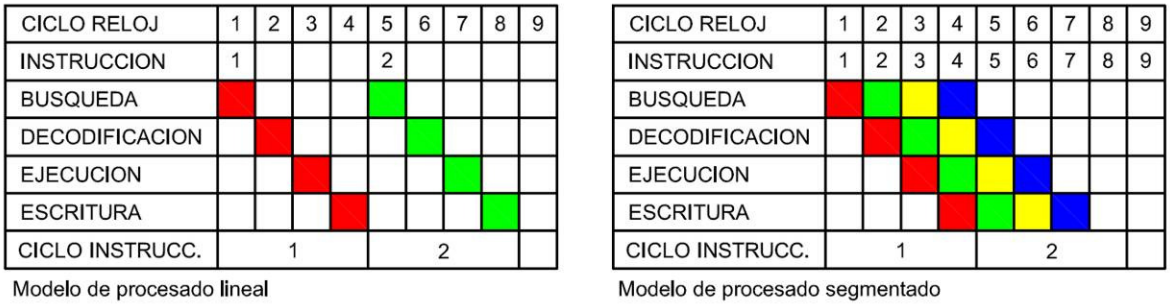


Ilustración 2.3: Comparativa entre el modelo de procesado lineal y segmentado.

Ciclo de Instrucción	1	2	3	4	5	6
Ejemplo: 1. MOVLW 5h 2. MOVWF PORTB 3. CALL AABB 4. MOVLW 6h	Busca 1					
		Ejecuta1				
		Busca2	Ejecuta2			
			Busca3	Ejecuta3		
				Busca4	Vacio por salto	
					Busca AABB	Ejecuta AABB

Tabla 2.1: Ejemplo de ejecución de instrucciones en procesador segmentado.

2.1.3.4 Tecnología CMOS.

Son muchas las ventajas de la tecnología CMOS (Complementary metal-oxide-semiconductor) aplicada a los circuitos integrados frente a la TTL (transistor-transistor logic), ya que permite rangos de tensión de alimentación entre 3V y 15V aproximadamente, a diferencia de la tecnología TTL que limita la alimentación a niveles comprendidos entre 4,75V y 5,25V. No obstante, como medida de protección ante el envejecimiento prematuro del dispositivo, se recomienda no alimentar con niveles de tensión superiores a 12V. Este generoso incremento en el rango de tensión de alimentación nos lleva a definir, respecto a estos, los niveles lógicos de tensión, estableciéndose el "0 lógico" (o nivel bajo) en las tensiones por debajo de un tercio de la tensión de alimentación (1/3Vcc) y el "1 lógico" (o nivel alto) a los valores por encima de los dos tercios de la tensión de alimentación (2/3Vcc).

En cuanto al consumo de un dispositivo CMOS frente a uno TTL podemos establecerlo en diez veces inferior mientras su capacidad de carga (cargabilidad a la salida de la puerta CMOS) es cuarenta veces superior a la TTL. Si a esto sumamos sus buenas características de inmunidad al ruido obtenemos la formula por la cual la tecnología CMOS está tan extendida en la industria.

En cualquier caso, no todo pueden ser ventajas. Si hablamos de velocidad de transmisión, la tecnología CMOS puede definirse como lenta a razón de 50MHz en algunos casos extremos, frente a los 250MHz de la serie TTLS (Long Scale) standart.

En cualquier caso, si la velocidad de transmisión no es un requisito indispensable, parece un interesante precio a pagar:

Ventajas CMOS	Inconvenientes CMOS
✓ Mayor rango de tensione de alimentación.	✓ Menor velocidad de transmisión.
✓ Mayor diferenciación entre niveles lógicos.	
✓ Menor consumo.	
✓ Mayor cargabilidad.	
✓ Mayor inmunidad al ruido.	

Tabla 2.2: Ventajas de la tecnología CMOS frente a TTL.

2.1.3.5 Características del microcontrolador PIC16F877.

CPU:

- ✓ Tecnología RISC.
- ✓ Sólo 35 instrucciones que aprender.
- ✓ Todas las instrucciones se ejecutan en un ciclo de reloj, excepto los saltos que requieren dos.
- ✓ Frecuencia de operación de 0 a 20 MHz (200 nseg de ciclo de instrucción).
- ✓ Opciones de selección del oscilador.

Memoria:

- ✓ Hasta 8k x 14 bits de memoria Flash de programa.
- ✓ Hasta 368 bytes de memoria de datos (RAM)
- ✓ Hasta 256 bytes de memoria de datos EEPROM
- ✓ Lectura/escritura de la CPU a la memoria flash de programa
- ✓ Protección programable de código
- ✓ Stack de hardware de 8 niveles

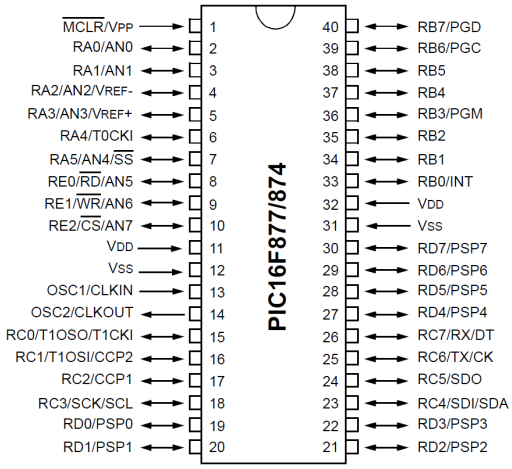


Ilustración 2.4: Esquema PIC16F877 en zócalo DIM 40 pines.

Reset e interrupciones:

- ✓ Hasta 14 fuentes de interrupción
- ✓ Reset de encendido (POR)
- ✓ Timer de encendido (PWRT)
- ✓ Timer de arranque del oscilador (OST)
- ✓ Sistema de vigilancia Watchdog timer.

Otros:

- ✓ Modo SLEEP de bajo consumo de energía
- ✓ Programación y depuración serie "In-Circuit" (ICSP) a través de dos patitas.
- ✓ Rango de voltaje de operación de 2.0 a 5.5 volts
- ✓ Alta disipación de corriente de la fuente: 25mA

- ✓ Rangos de temperatura: Comercial, Industrial y Extendido
- ✓ Bajo consumo de potencia:
  - Menos de 0.6mA a 3V, 4 Mhz
  - 20 µA a 3V, 32 Khz
  - menos de 1µA corriente de standby (modo SLEEP).

Periféricos:

Periférico	PIC16F873 PIC16F876	PIC16F874 PIC16F877	Características
3 a 5 Puertos paralelos	PortA,B,C	PortA, B,C,D,E	con líneas digitales programables individualmente
3 Timers	Timer0	Timer0	Contador/Temporizador de 8 bits con pre-escalador de 8 bits
	Timer1	Timer1	Contador/Temporizador de 16 bits con pre-escalador
	Timer2	Timer2	Contador/Temporizador de 8 bits con pre-escalador y post-escalador de 8 bits y registro de periodo
2 módulos CCP	Captura	Captura	16 bits, 1.5 nseg de resolución máxima
	Comparación	Comparación	16 bits, 200 nseg de resolución máxima
	PWM	PWM	10 bits
1 Convertidor A/D	AN0,...,AN4	AN0,...,AN7	de 10 bits, hasta 8 canales
Puertos Serie	SSP	SSP	Puerto Serie Síncrono
	USART/SCI	USART/SCI	Puerto Serie Universal
	ICSP	ICSP	Puerto serie para programación y depuración "in circuit"
Puerto Paralelo Esclavo	PSP	PSP	Puerto de 8 bits con líneas de protocolo

Tabla 2.3: Características de los periféricos de la familia 16F

2.1.3.6 ¿Cómo se graba el PIC16F877?.

Para llevar a cabo el proceso de grabado de un PIC necesitamos disponer de una tarjeta hardware comúnmente llamada programador y de un software. Si bien es cierto que el proceso de grabado puede realizarse tanto en paralelo como en serie, en nuestro caso nos centraremos en la programación serie, debido a que es la más habitual y la que he usado para este proyecto, no obstante, téngase en cuenta que existe también la programación en paralelo y la programación mediante USB, que es una variante de la modalidad serie.

La función del programador (o tarjeta de programación) es excitar los pines necesarios para configurar el PIC en modo programación, permitiendo que reciba el software que hemos creado y compilado para su función específica a través del pin destinado a la recepción de los datos.

En esta operación de grabado se involucran solo 5 pines, que según cada modelo pueden estar ubicados de diferente manera. Por ello es común encontrar tarjetas especificas para determinadas familias de PICs (en función del número de pines), o tarjetas con zócalos de diferentes tamaños que permiten la programación de varias familias de PICs. El conocimiento de las funciones de estos pines posibilita la implementación de esta circuitería en la placa final del dispositivo, permitiéndose la programación del dispositivo sin desmontarlo del circuito final de aplicación. Esta propiedad conocida como ICSP (In Circuit Serial Program) es muy cómoda para las opciones de mantenimiento y actualización de software.

Los pines involucrados son:

- ✓ VPP: Es el pin que alimentado a 12V establece al PIC en modo de programación. Es el único pin que trabaja con este nivel de tensión capaz de destruirlo si se aplica en cualquier otra patilla del dispositivo.
- ✓ VDD: Es el pin destinado a la alimentación negativa del dispositivo. En este caso se aplica tensión de referencia o masa.
- ✓ VSS: Es el pin destinado a la alimentación positiva del dispositivo. En este caso +5V.
- ✓ PGD: Es el pin destinado al intercambio de datos durante la programación serie.
- ✓ PGC: Es el pin destinado a la señal de reloj durante la programación serie.

Dejando de lado la posibilidad de programar sobre el circuito final, diremos que existen multitud de programadores comerciales a la venta tanto en tiendas de electrónica como en internet. El ejemplo más claro es el programador de Microchip PICStart Plus que permite conexión serie y USB y es perfectamente compatible con el software gratuito de programación, depuración y grabación MPLab (también de Microchip).

Por otra parte existe información libre y suficiente para la fabricación de diferentes modelos que se adapten a nuestras necesidades. El más extendido quizás sea el modelo JDM (iniciales de su creador Jens Dyekjær Madsen) o cualquiera de sus muchas variantes, con características básicamente similares. Es un circuito muy simple, pero que tiene una serie de ventajas que lo hacen muy interesante:

- ✓ Se conecta al puerto serie, disponible generalmente en cualquier PC.
- ✓ Existe software gratis para utilizarlo, incluso bajo DOS, LINUX y por supuesto Windows (incluido WindowsXP)
- ✓ Sirve para programar varios modelos de PICS (PIC12C5XX, 12C67X, 24CXX, 16C55X, 16C61, 16C62X, 16C71, 16C71X, 16C8X, 16F8X entre otros) y también para leer/escribir varios chips de memoria (24Cxx). Otros microcontroladores también pueden ser programados mediante un adaptador.
- ✓ Dispone del conector ICSP (In-Circuit Serial Programming) para la programación de microcontroladores sin necesidad de desmontarlos de su placa de circuito impreso.
- ✓ No necesitamos de una fuente de alimentación externa, ya que se alimenta directamente del puerto del PC.
- ✓ Su costo es muy bajo.

En cuanto al software, su función es establecer la comunicación síncrona con el PIC que permita la transmisión del software previamente programado y compilado para desarrollar la tarea específica a la que estará destinado el PIC. Este software permite la verificación de la grabación del dispositivo y la configuración de diversos permisos sobre el PIC que permitan posteriores lecturas, y comprobaciones del contenido de memoria del dispositivo.

Como ya se ha mencionado MPLab permite la programación, la depuración e incluso el grabado, si hacemos uso de ciertos grabadores, de los microcontroladores PIC y es suministrado gratuitamente por Microchip en su página oficial (www.microchip.com), no obstante existen otras alternativas igualmente gratuitas como WinPic800 (www.winpic800.com) o IcProg (www.ic-prog.com) que aunque no permiten la programación y la depuración resultan igualmente efectivas en el gravado de los dispositivos.

2.1.3.7 Programador y software empleados en este proyecto.

Para el caso particular de este proyecto se ha recurrido a un programador basado en JDM denominado MULTIPIC PROGRAMMER 5 Ver.2. de libre distribución. Es un derivado del programador JDM que puede encontrarse en la dirección web http://feng3.nobody.jp/en/pg5v2.html. Aunque carece de la programación ICSP presenta todas las características y ventajas del diseño original:

- ✓ Realiza la programación a través del puerto serie.
- ✓ Carece de alimentación externa ya que utiliza la alimentación del puerto serie.
- ✓ Es un diseño de bajo coste.

Ilustración 2.5: Programador MULTIPIC PROGRAMMER 5Ver.2 montado para este proyecto

Para el grabado del PIC he empleado el software WINPIC800, también de distribución libre. Su uso es sencillo y tras algunas nociones básicas sobre su interfaz se puede comenzar a usar con resultados satisfactorios. Veamos estas nociones básicas.

Al iniciar WINPIC800 se nos presenta la pantalla principal. En ella debemos poder identificar el programador, la familia y el modelo exacto del PIC a programar.

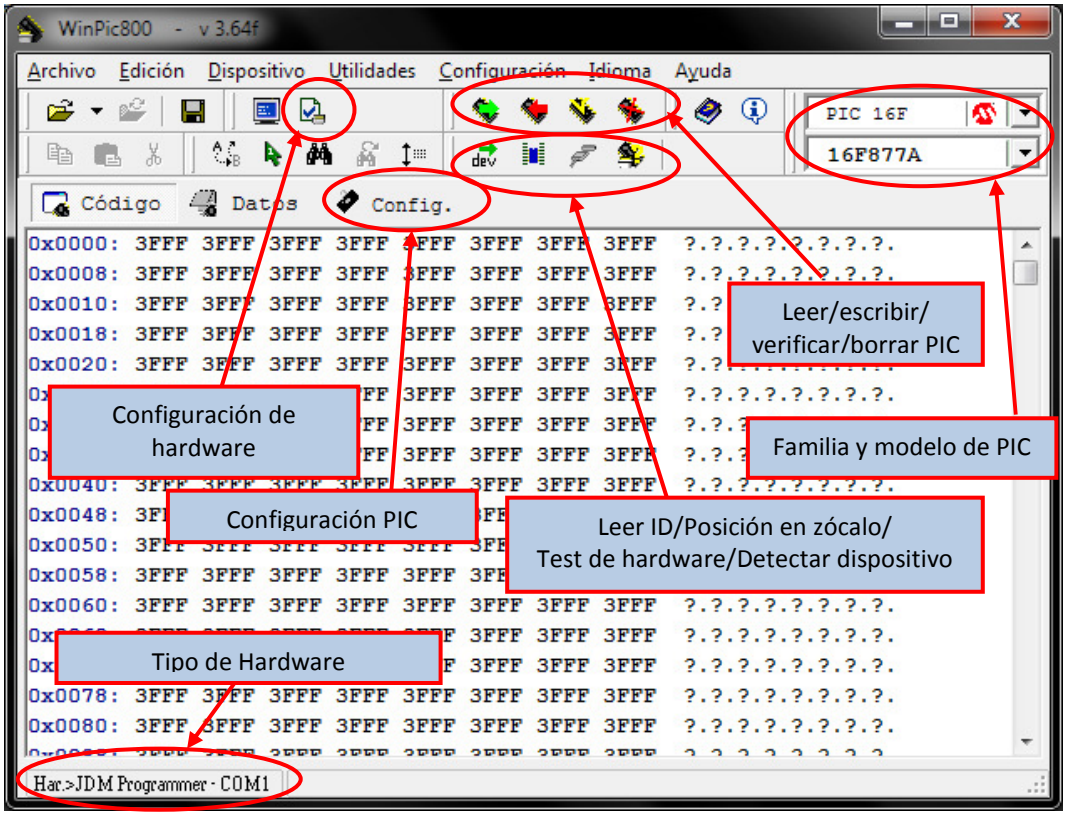


Ilustración 2.6: Captura de la pantalla principal del Software de grabación WINPIC800.

Si es la primera vez que lo ejecutamos, o cambiamos de modelo de PIC o de hardware para grabarlo, debemos configurarlo.

Para seleccionar la familia y el modelo del PIC, en la pantalla principal (Ilustración 2.6) desplegamos sus respectivos campos y seleccionamos el PIC con el que trabajaremos, en nuestro caso 16F877A. Ahora pulsaremos sobre el icono de configuración de hardware y seleccionamos el tipo de hardware que utilizaremos para la programación y el puerto donde lo conectamos, en nuestro caso será JDM programmer y el puerto COM1, como se muestra en la Ilustración 2.7. Una vez configurado el hardware de programación, debemos configurar las características del PIC.

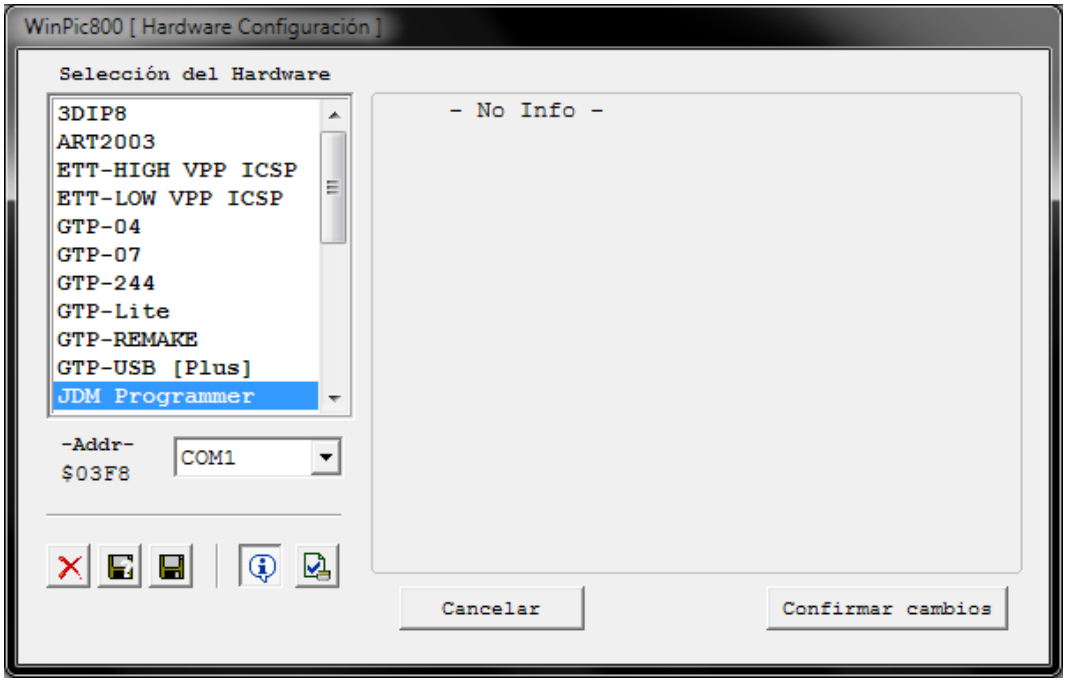


Ilustración 2.7: Captura de la pantalla de configuración de hardware del Software de grabación WINPIC800.

Estas características podemos programarlas bajo software, aun así haremos que coincidan tanto en software como en hardware.

Pulsamos sobre el icono de configuración del PIC para acceder a la pantalla representada en la Ilustración 2.8. En ella seleccionaremos las características de configuración del PIC respecto a los datos representados en la Tabla 2.4:

Oscilador	
LP	Oscilador de bajo consumo (32-200 KHz).
XT	Oscilador estándar (100 KHz – 4 MHz).
HS	Oscilador de alta velocidad (8- 20 MHz).
RC	Oscilador de bajo costo.
Protección de Código	
CP OFF.	Sin protección para el código
CP 0000h-00FFh	Protección para ½ del código (la parte alta).
CP 0000h-07FFh	Protección para ¾ del código (la parte alta).
CP 0000h-0FFFh	Protección total del código.
Bits de Configuración	
WDTEN	Activación del perro guardián.

PWRTEN	Activación del Timer de Conexión de Alimentación.
BOREN	Activación del Reset por Caída de Tensión.
LVP	Activación de la programación en Bajo Voltaje.
CPD	Activación del código de protección de la memoria EEPROM.
CP	Activación del permiso de escritura en la memoria FLASH.
DEBUG	Activación del Modo Depurador en Circuito.

Tabla 2.4: Opciones para la configuración del PIC.

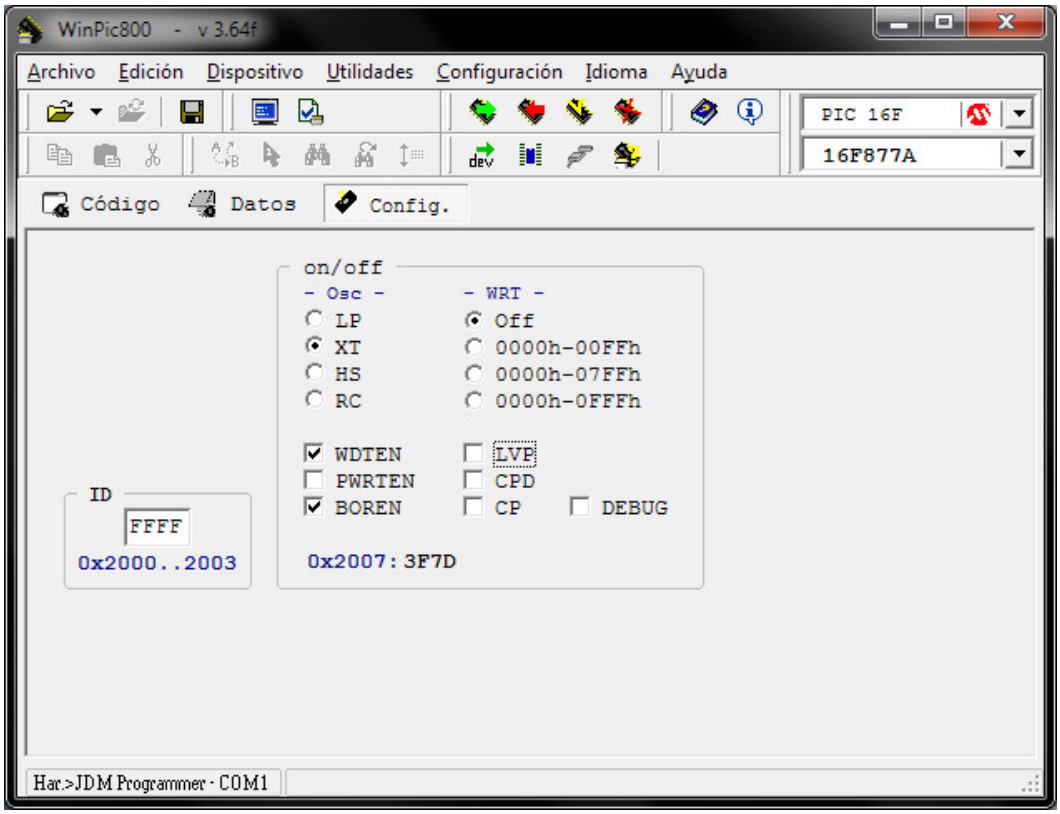


Ilustración 2.8: Captura de la pantalla de configuración de PIC del Software de grabación WINPIC800.

Una vez configurado WINPIC800 tan solo debemos abrir el archivo *.hex resultado de la compilación de nuestro programa y pulsar el icono de programar todo y esperar la confirmación de operación realizada con éxito.

2.2 Introducción a LabView.

2.2.1 ¿Qué es LabView?

LabView es un entorno de programación gráfico de la compañía National Instruments, usado por miles de ingenieros e investigadores para desarrollar sistemas sofisticados de medida, pruebas y control usando íconos gráficos e intuitivos y cables que parecen un diagrama de flujo. Ofrece una integración incomparable con miles de dispositivos de hardware y brinda cientos de bibliotecas integradas para análisis avanzado y visualización de datos, todo para crear instrumentación virtual.

La plataforma LabView es escalable a través de múltiples objetivos y sistemas operativos, desde su introducción en 1986 se ha convertido en un líder en la industria.

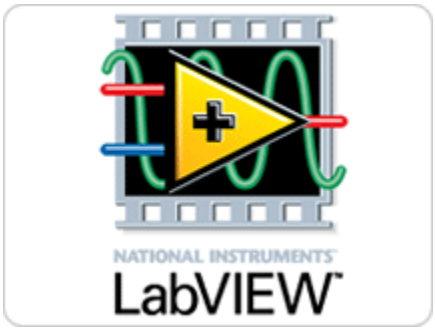


Ilustración 2.9: Logotipo LabView National Instruments.

Las aplicaciones de LabView engloban tareas como:

- ✓ Adquisición de datos y procesamiento de señales.
- ✓ Automatización de sistemas de pruebas y validación.
- ✓ Enseñanza e investigación académica.
- ✓ Control de instrumentos.
- ✓ Diseño de sistemas embebidos.

2.2.2 Ventajas en el uso de LabView.

LabView ofrece interesantes ventajas de uso. El lenguaje gráfico permite el desarrollo de programas relativamente complejos con cierta facilidad, permitiendo programar mediante algunos gráficos lo que en código de texto nos habría llevado muchas líneas de programa.

LabView está ideado para la creación de instrumentación virtual, de hecho sus programas se denominan VI (Virtual Instruments), aunque en él puede desarrollarse cualquier software que hiciésemos en código de texto. Este concepto plantea una gran ventaja, poder implementar instrumentación virtual o real (incluso de diferentes fabricantes) mediante el empleo de un único software adaptado a un proceso.

LabView permite la modularidad en la resolución de problemas. El hecho de poder ejecutar sub-VI's dentro del V.I principal potencia la reutilización de código tanto propio como compartido, de ahí la gran cantidad de bibliotecas disponibles. Todo esto desemboca a una gran comunidad de desarrolladores de código que ofrecen soporte para el aprendizaje y perfeccionamiento en el diseño de código.

2.2.3 Por qué he elegido este software.

Como ya hemos dicho, LabView es un entorno de programación gráfico altamente extendido en la industria. Desde sistemas SCADA (Supervisory Control And Data Acquisition) a equipos de diagnosis pasando por sistemas de registro de datos en laboratorios de investigación o equipamiento para aplicaciones biomédicas, LabView proporciona a ingenieros y técnicos una potente herramienta, adaptable a las características y necesidades específicas de cada proyecto.

La formación curricular en materia de programación supone para la carrera de un ingeniero un gran aporte en recursos para la resolución de problemas y en definitiva aptitudes favorables de cara a diversas oportunidades laborales.

2.3 Tarjeta de adquisición de datos.

La tarjeta de adquisición de datos tiene por objeto recabar los valores medidos por los sensores (transductores) y trasmitirlos por puerto serie al ordenador para su tratamiento y gestión. En su análisis descompondremos el conjunto en dos grupos fundamentales:

- ✓ Los sensores y sus circuitos de adaptación de señal.
- ✓ El PIC.

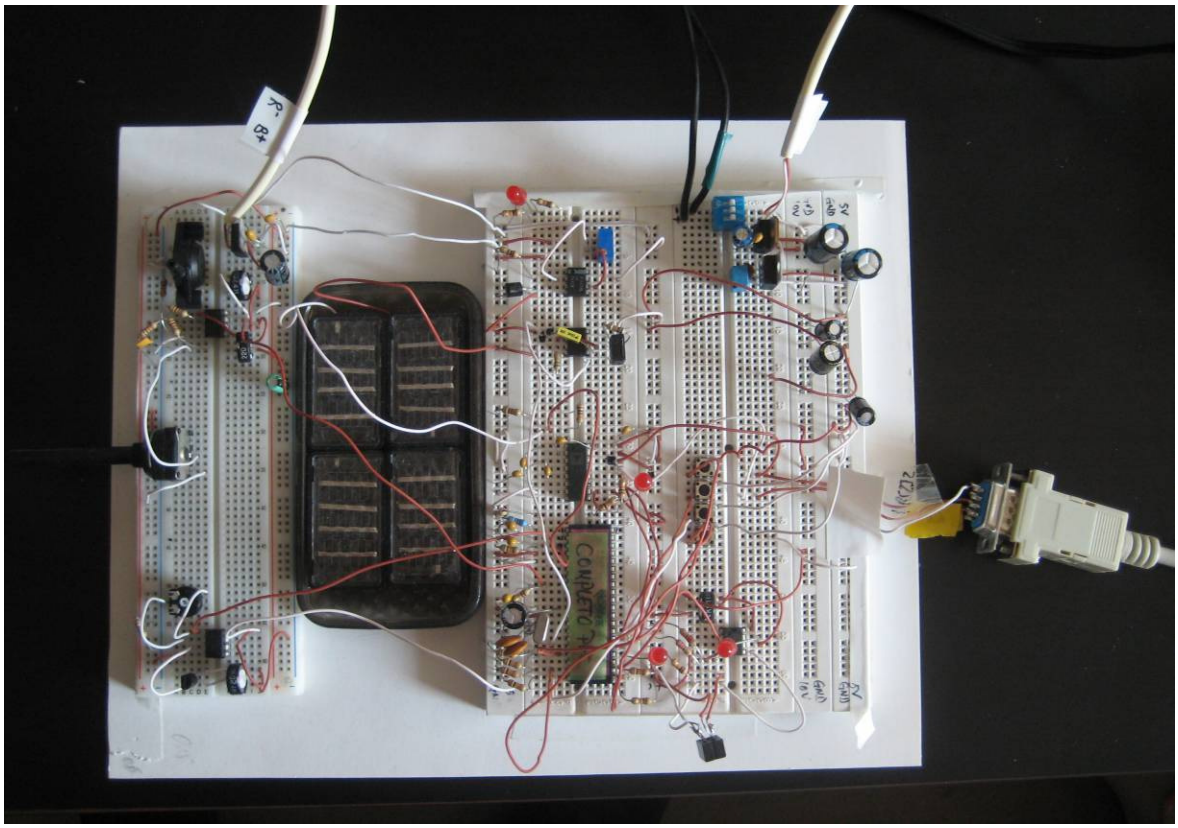


Ilustración 2.10: Prototipo de la tarjeta de adquisición de datos.

2.3.1 Los sensores y sus circuitos de adaptación de señal. Alternativas propuestas a los sensores comerciales mediante componentes discretos.

En virtud de mantener el carácter académico de este trabajo, se han analizado los principios básicos de los sensores necesarios para la medida de las magnitudes deseadas y se han implementado mediante el uso de componentes discretos, no recurriendo a sensores comerciales específicos para magnitudes típicamente meteorológicas.

Si bien es necesario tener un conocimiento previo sobre las posibilidades del microcontrolador para tener un objetivo claro en la proposición de alternativas a los sensores comerciales mediante componentes discretos, abordaremos en primer lugar el análisis de las alternativas propuestas, con objeto de poder hacer una exposición ordenada y sin saltos, haciendo un especial hincapié en la coexistencia de ambos apartados en el proceso de desarrollo del proyecto.

Las magnitudes que deseamos medir son:

- ✓ Temperatura. (ºC)
- ✓ Humedad. (%)
- ✓ Presión Atmosférica. (hPa.)
- ✓ Velocidad del Viento. (m/s)
- ✓ Dirección del Viento. (º)
- ✓ Radiación Solar (Irradianza). (W/m^2)

2.3.1.1 Alternativa propuesta para el Termómetro.

El termómetro será el sensor destinado a la medida de la temperatura del aire (temperatura ambiente). El componente discreto elegido para tal propósito es el circuito integrado LM35. Este sensor analógico de temperatura ofrece un voltaje de salida linealmente proporcional a la variación de temperatura, con una sensibilidad de +10mV/ºC y una exactitud garantizada de 0,5ºC. El sensor LM35 permite un rango de medida de {-55ºC a +150ºC} y carece de calibración. El conjunto de las características del sensor LM35 nos ofrece unas prestaciones óptimas para el objeto del presente proyecto.

Ec. 2.1:
$$Sensibilidad_{LM35}\left[\frac{mV}{^{o}C}\right] \equiv S = 10\left[\frac{mV}{^{o}C}\right] = 0.01\left[\frac{V}{^{o}C}\right]$$

Ec. 2.2:
$$V[V] = T[^{o}C]\cdot S\left[\frac{V}{^{o}C}\right] = T\cdot 0.01[V]$$

Para el caso particular de nuestro proyecto vamos a establecer que el sensor sea capaz de medir dentro del rango {-15ºC; +60ºC}. Según la Ec. 2.2 para el rango de temperaturas especificado, obtendremos un rango de tensión de {-0.15V;+0.60V}.

Puesto que el PIC no soporta tensiones negativas en sus puertos, debemos realizar un desplazamiento lineal sobre la curva característica de Voltaje-Temperatura para adaptarla a las características del PIC (Ilustración 2.11). Para ello, modificaremos la

referencia de masa del sensor, haciendo que para una temperatura de 0ºC, entregue 0.15V. La ecuación resultante para la función de transferencia final será:

Ec. 2.3:
$$V'[V] = T[^{\circ}C] \cdot S\left[\frac{V}{^{\circ}C}\right] + 0.15[V]$$

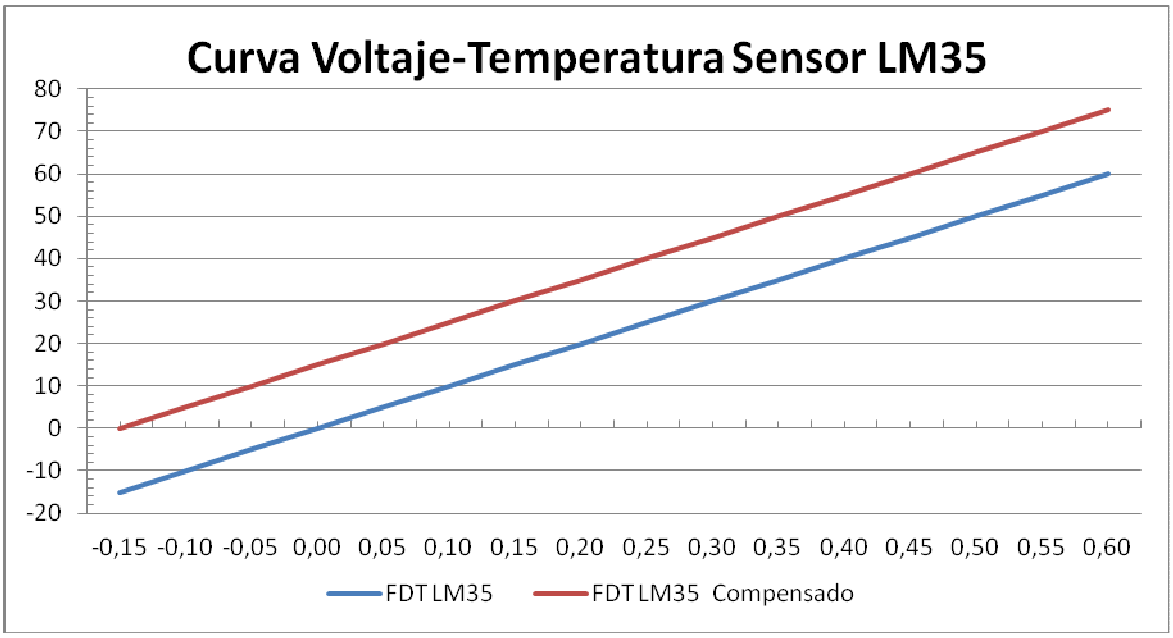


Ilustración 2.11: Curva característica Voltaje-Temperatura del sensor LM35

En consecuencia, para el rango de temperaturas deseado {-15ºC; +60ºC} obtendremos un rango de tensión de {0V; +0.75V}.

Una vez que tenemos una señal positiva para todo nuestro rango de medida, debemos adecuar la señal para su conversión por el ADC. Con objeto de maximizar la resolución de la conversión vamos a configurar el PIC para que trabaje con sus tensiones de referencia máximas y mínimas que corresponden a las tensiones de alimentación del PIC (V_{DD}=5V y V_{SS}=0V). Por tanto, amplificaremos la señal para conseguir una tensión de salida máxima de 5v. Definimos, en consecuencia, la ganancia del circuito adaptador mediante la ecuación:

Ec. 2.4:
$$\Delta V = \frac{Vo}{Vi} = \frac{V\max_{ADC}}{V'\max_{LM35}} = \frac{5V}{0.75V} = 6.67$$

Ahora tendríamos una curva característica conforme a la gráfica siguiente:

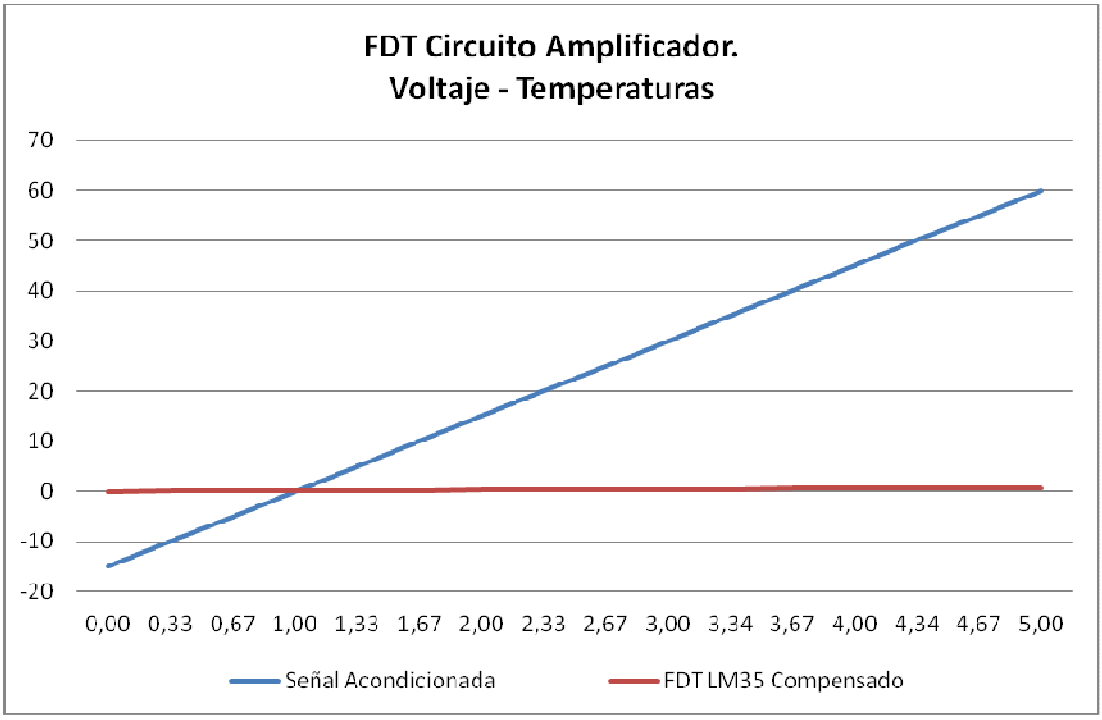


Ilustración 2.12: Curva característica Voltaje-Temperatura para el circuito amplificador.

Para llevar a cabo el circuito de adecuación, emplearemos un amplificador operacional (A.O.) en configuración no inversora. Las ventajas de uso del A.O. (ideal) son principalmente:

- ✓ Impedancia de entrada (Zi) infinita.
- ✓ Impedancia de salida (Zo) cero.
- ✓ Ganancia (G) Infinita.
- ✓ Tiempo de respuesta cero.

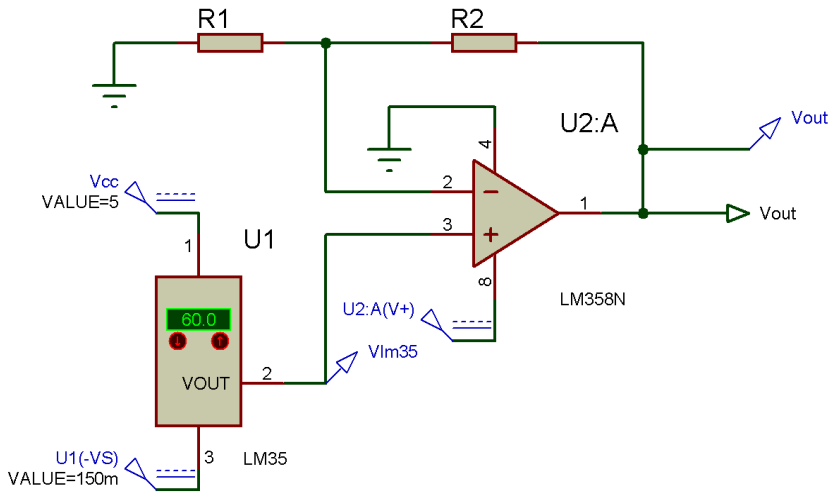


Ilustración 2.13: Esquema del circuito acondicionador para la señal de temperatura.

Para calcular los valores de las resistencias analizaremos el circuito. Considerando que se trata de un A.O. con realimentación negativa podemos establecer que:

Ec. 2.5: $$V^+ = V^-$$

Ec. 2.6: $$I^+ = I^- = 0$$

Así pues, analizamos el circuito y aplicando Ec. 2.6:

Ec. 2.7: $$\frac{0 - V^-}{R_1} = \frac{V^- - V_{out}}{R_2}$$

Aplicando Ec. 2.5:

Ec. 2.8: $$\frac{-V_{lm35}}{R_1} = \frac{V_{lm35} - V_{out}}{R_2}$$

Despejando V_{out} tendremos:

Ec. 2.9: $$V_{out} = \left(1 + \frac{R_2}{R_1}\right) \cdot V_{lm35}$$

Puesto que conocemos el valor deseado de ganancia (Ec. 2.4) dividimos Ec. 2.9 entre V_{lm35} para poder despejar la ganancia.

Ec. 2.10: $$\frac{V_{out}}{V_{lm35}} = \left(1 + \frac{R_2}{R_1}\right) = 6.67$$

Ec. 2.11: $$5{,}67 \cdot R_1 = R_2$$

Buscando una solución de compromiso, podemos establecer R_1=1kΩ

Ec. 2.12: $$\boxed{R_1 = 1k\Omega \quad ; \quad R_2 = 5{,}670k\Omega}$$

El circuito final quedará con los valores de R normalizados como:

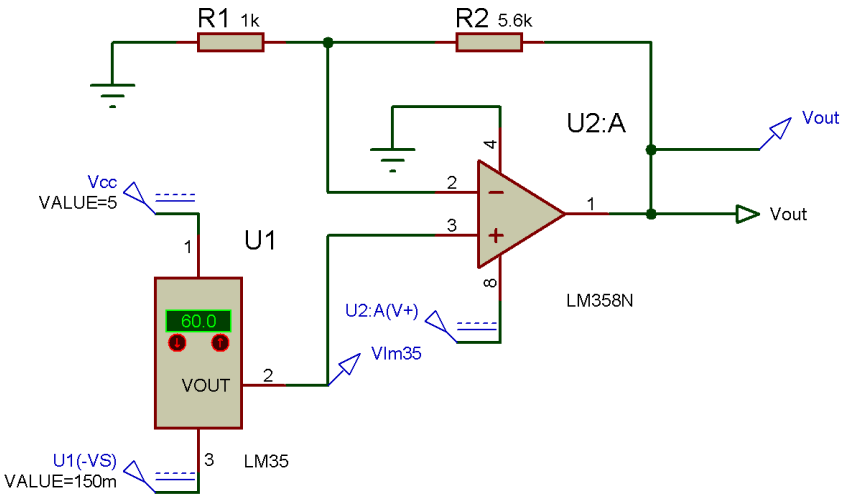
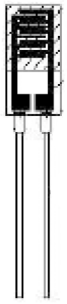


Ilustración 2.14: Circuito acondicionador final para la señal de temperatura.

2.3.1.2 Alternativa propuesta para el Higrómetro.

El higrómetro es el sensor destinado a la medida de vapor de agua contenido en el aire. A esta medida se le conoce como humedad en el aire. Esta magnitud puede representarse en valor absoluto, recibiendo en nombre de Humedad Absoluta (H.A.) o mediante su valor relativo (medida más común), recibiendo el nombre de Humedad relativa (HR). La Humedad Relativa, representa la relación porcentual entre la cantidad de vapor de agua real que contiene el aire y la que necesitaría contener para saturarse a idéntica temperatura.



Para realizar esta medida se ha propuesto la utilización de un sensor de humedad resistivo modelo HCZ-D5 de la compañía MULTICOMP. Se trata de un polímero sensible a la humedad sobre el que se disponen dos pistas conductoras enfrentadas. El polímero al absorber humedad aumenta su conductividad (por ende disminuye la resistencia) permitiendo medir la humedad absorbida por el polímero.

Ilustración 2.15: Imagen del sensor HCZ-D5 (datasheet HCZ-D5)

El sensor HCZ-D5 varía su resistencia conforme a la siguiente tabla:

Relative Humidity - Impedance - 25°C, 1 KHz, 1 V_{rms} (Sine wave)

% RH	20	30	40	50	60	70	80	90
Normal Value (KΩ)	6,300	1,400	310	87	31	11.8	4.8	2

Tabla 2.5: Respuesta en kΩ frente a diferentes valores de Humedad Relativa. (datasheet HCZ-D5)

La metodología para el sensado de la humedad relativa se basa en la configuración del integrado 555 como multivibrador astable. Esta configuración, permite generar una onda cuadrada continua en la que controlaremos el periodo de la señal, determinando tanto el tiempo en el que ésta permanece en estado alto, como en el que permanece en estado bajo. Estos tiempos vienen determinados por los valores de las resistencias R_1 y R_2 así como por el valor del condensador C_1 conforme a las expresiones:

Ec. 2.13: $$t_1 = \ln(2) \cdot (R_1 + R_2) \cdot C_1$$

Ec. 2.14: $$t_2 = \ln(2) \cdot R_2 \cdot C_1$$

Ec. 2.15: $$T = t_1 + t_2$$

Ec. 2.16: $$T = \ln(2) \cdot (R_1 + 2 \cdot R_2) \cdot C_1$$

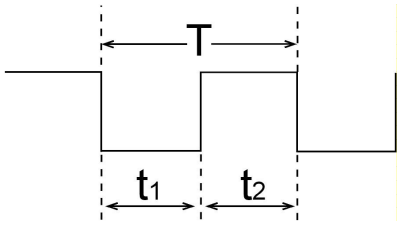


Ilustración 2.16: Onda modelo del multivibrador astable.

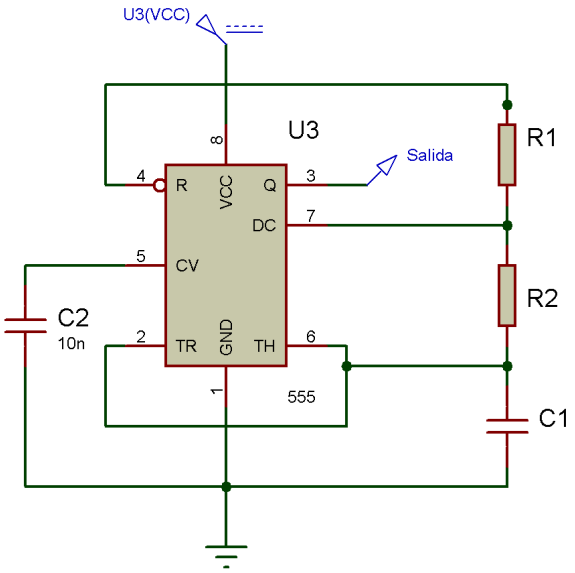


Ilustración 2.17: Esquema del integrado 555 en configuración de oscilador astable.

Vamos a utilizar el *TIMER1* y el *CCP2* del PIC para capturar el periodo de la señal producida por el oscilador. Sabiendo que trabajamos a 4Mhz, que hemos configurado el preescaler en 1:4 y que el *TIMER1* es un registro de 16 bits y que por tanto dispone de 2^{16} cuentas (65536 cuentas), calculamos los valores de los componentes del circuito para que la señal generada permanezca dentro de los límites de medida, tratando de optimizarlo al máximo:

Ec. 2.17:
$$T = \frac{1}{\frac{fosc}{4}} \cdot \frac{1}{preescaler} = 1\mu s \cdot 4 = 4\mu s$$

De esta ecuación obtenemos el valor de la frecuencia para el periodo mínimo y para el periodo máximo que podremos medir.

Ec. 2.18:
$$T_{\min} = 4\mu s \Rightarrow F = 250kHz$$

Ec. 2.19:
$$T_{\max} = 4\mu s \cdot 65536 = 0.262144\ s \Rightarrow F = 3{,}8147Hz$$

Nuestro sensor variará su resistencia en valores comprendidos entre 6300kΩ para una HR% de 20% y 2kΩ para una HR% de 80% (como se muestra en la Tabla 2.5). Así pues, conocidos los valores para R1, a partir de la Ec. 2.16 obtendremos:

Ec. 2.20:
$$T_{\max} = \ln(2)\cdot(R_{1\max} + 2R_2)C_1 \Rightarrow T_{\max} \approx 0.693(6300k\Omega + 2R_2)C_1$$

Ec. 2.21:
$$T_{\min} = \ln(2)\cdot(R_{1\min} + 2R_2)C_1 \Rightarrow T_{\min} \approx 0.693(2k\Omega + 2R_2)C_1$$

Los valores que optimizan estas ecuaciones responden a:

Ec. 2.22: $$R_2 = 47k\Omega; \quad C_1 = 68pF$$

Los periodos máximos y mínimos quedarán establecidos en:

Ec. 2.23: $$T_{max} \approx 0.693(6300k\Omega + 2 \cdot 47k\Omega)68pF = 301{,}31\mu s \Rightarrow f = 3318{,}83Hz$$

Ec. 2.24: $$T_{min} \approx 0.693(2k\Omega + 2 \cdot 47k\Omega)68pF = 4.52\mu s \Rightarrow f = 221{,}05kHz$$

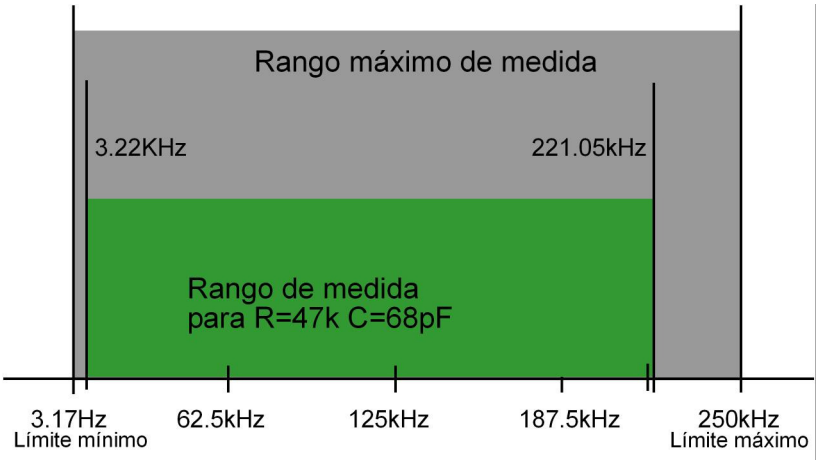


Ilustración 2.18: Acondicionamiento del rango de medida para el higrómetro

El circuito generador de la señal medible para establecer la humedad quedará entonces:

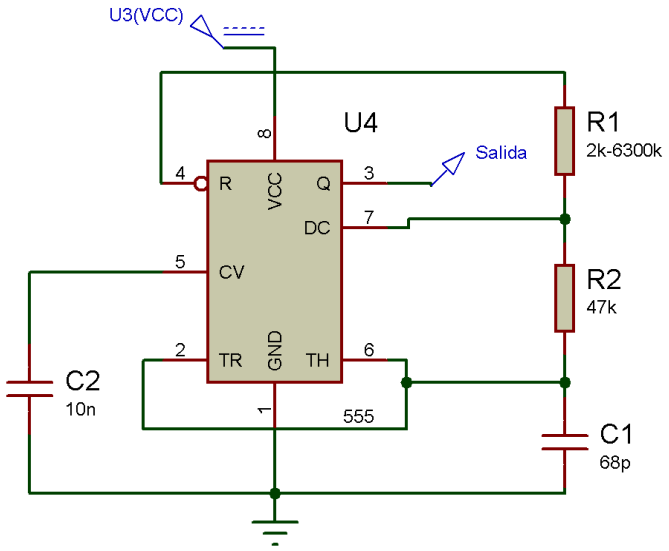
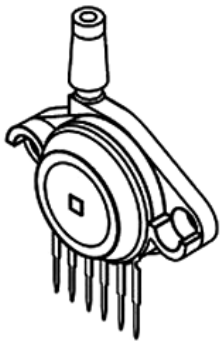


Ilustración 2.19: Esquema final del integrado 555 en configuración de oscilador astable.

2.3.1.3 Alternativa propuesta para el Barómetro.

El barómetro es el sensor destinado a la medida de la presión atmosférica. Para realizar esta medida se ha propuesto la utilización del sensor MPX4115AP de Motorola. Se trata de una piezoresistencia de silicio sensible a la presión, que proporciona una variación de tensión exacta y lineal, directamente proporcional a la presión que se le aplica.



PIN NUMBER			
1	V_{out}	4	N/C
2	Grd	5	N/C
3	V_S	6	N/C

Ilustración 2.20: Sensor de presión MPX4115AP y pines de conexionado. (datasheet MPX4115AP)

El sensor consta, de un diafragma monolítico de silicio para la medida del esfuerzo y de una fina película en una red de resistencias integradas en un chip. El chip se ajusta, calibra y compensa en temperatura por láser. El modelo MPX4115AP viene provisto de una caja o encapsulado tipo 867B-04, que dota al mismo de un puerto de conexión por si fuese preciso conectarlo a algún tipo de conducto.

El sensor MPX4115AP permite realizar la medida de la presión absoluta en un rango comprendido entre 0kPa y 115kPa, entregando una tensión proporcional entre 0V y 5V conforme a la Función de Transferencia siguiente:

Ec. 2.25: $$V_{OUT} = V_S \cdot (0.009 \cdot P - 0.095) \pm Error$$

A continuación se muestra la gráfica correspondiente a la FDT del sensor, también conocida como característica presión-tensión.

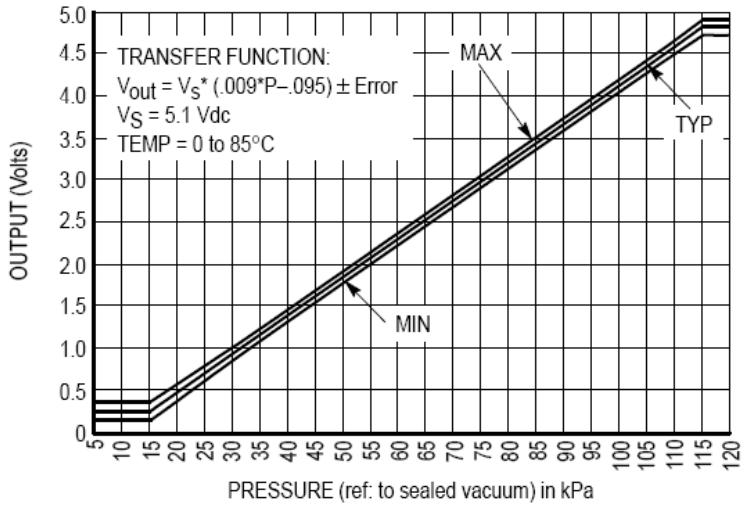


Ilustración 2.21: Característica Presión-Tensión del sensor de presión absoluta MPX4115AP.

Para el caso particular de este proyecto se pretende medir un rango de presiones de {915hPa - 1080hPa} lo que supone medir un rango de tensiones de 0,74V en el intervalo {3,60V – 4,34V}. Para optimizar la resolución del ADC, vamos a acondicionar el circuito para medir valores comprendidos entre {0V – 5V}. Para ello, desplazaremos el valor mínimo de tensión medible a 0V mediante un amplificador operacional en configuración de restador y posteriormente amplificaremos la señal a medir para un máximo de 5V mediante un amplificador operacional con realimentación negativa en configuración no inversora. Veamos el circuito y sus correspondientes cálculos:

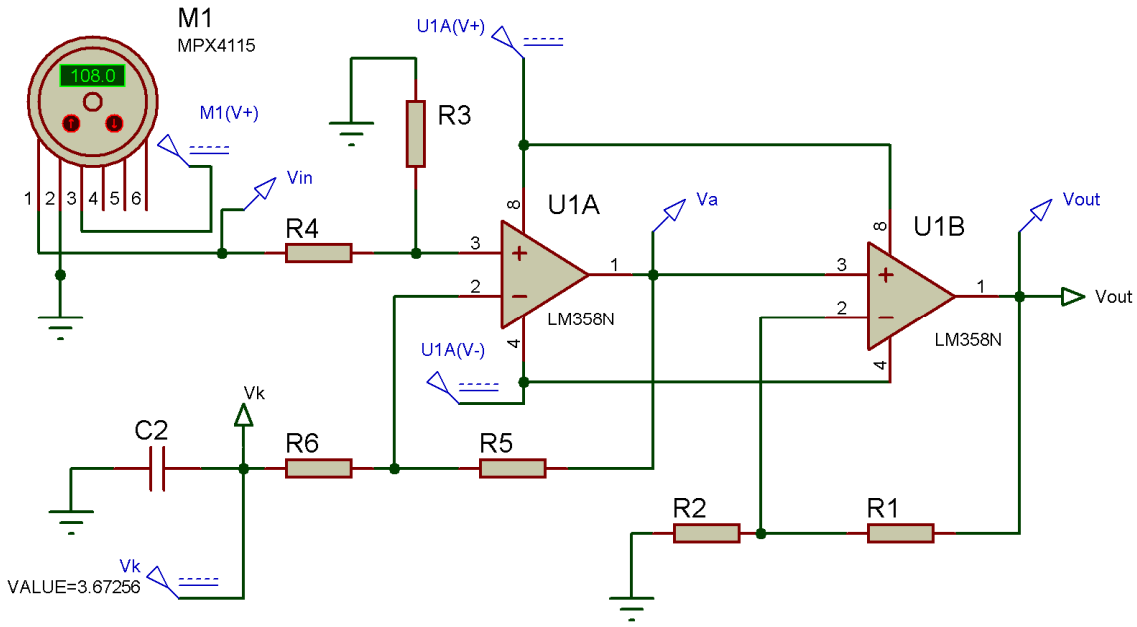


Ilustración 2.22: Circuito acondicionador del sensor MPX4115AP.

Etapa 1. Calibrado a cero. Correspondiente al A.O. U1A:

Ec. 2.26:

$$\frac{V_K - V^-}{R_6} = \frac{V^- - V_{1A}}{R_5}$$

Ec. 2.27:

$$\frac{V_{IN} - V^+}{R_4} = \frac{V^+ - 0}{R_3} \Rightarrow V^+ = \frac{R_3}{R_3 + R_4} V_{IN}$$

Considerando que se trata de un A.O. en con realimentación negativa, podemos considerar: $V^+ = V^-$, sustituyendo:

Ec. 2.28:
$$\frac{V_K - \frac{R_3}{R_3 + R_4} V_{IN}}{R_6} = \frac{\frac{R_3}{R_3 + R_4} V_{IN} - V_{1A}}{R_5}$$

Y despejando V_{1A}:

Ec. 2.29:
$$V_{1A} = \left(\frac{R_5}{R_6} + 1 \right) \frac{R_3}{R_3 + R_4} V_{IN} - \frac{R_5}{R_6} V_K$$

Resolvemos la ecuación teniendo en cuenta que:

Considerando $R_K = 1k\Omega \quad \forall K \in \{3;6\}$

Ec. 2.30:
$$\boxed{V_{1A} = V_{IN} - V_K}$$

Etapa 2. Amplificación. Correspondiente al A.O. U1B:

Ec. 2.31:
$$\frac{0 - V^-}{R_2} = \frac{V^- - V_{1B}}{R_1}$$

Puesto que se trata de un A.O. en realimentación negativa $V^- = V^+ = V_{1A}$

Ec. 2.32:
$$\frac{V_{1B}}{V_{1A}} = 1 + \frac{R_1}{R_2} \Rightarrow \quad \Delta v = 6.67 = 1 + \frac{R_1}{R_2}$$

De donde obtenemos por solución de compromiso:

$$\boxed{R_1 = 100\Omega \Rightarrow \quad R_2 = 576\Omega}$$

Quedando el circuito como se representa en la figura siguiente:

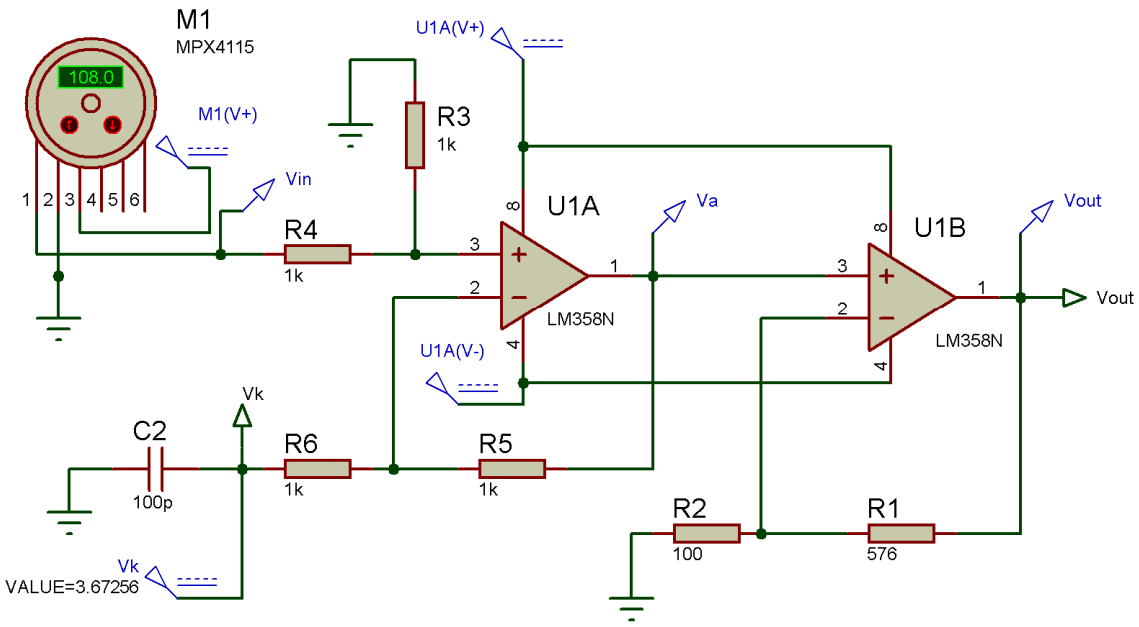


Ilustración 2.23: Circuito acondicionador del sensor MPX4115AP para el rango {915hPa -1080hPa}.

2.3.1.4 Alternativa propuesta para el Anemómetro.

El anemómetro es el sensor destinado a medir la velocidad instantánea del viento. Existen básicamente tres tipos de anemómetros:

- ✓ **Anemómetro de empuje:** Estos anemómetros están constituidos por una pala o bien por un cubo o esfera hueca, de muy poco peso que queda suspendido de un punto. A su exposición al viento, esta se desplaza dentro de un cuadrante determinando la fuerza que ejerce el viento sobre el objeto y así determinando la velocidad del viento.

- ✓ **Anemómetro de rotación:** Se trata de un eje solidario a una hélice o a unas cazoletas que giran a una velocidad proporcional a la velocidad del viento.

- ✓ **Anemómetro de compresión:** Este tipo de anemómetro está basado en el tubo de Pitot y está formado por dos pequeños tubos, uno de ellos con orificio frontal (que mide la presión dinámica) y lateral (que mide la presión estática), y el otro sólo con un orificio lateral. La diferencia entre las presiones medidas permite determinar la velocidad del viento.

Para este proyecto vamos a considerar un anemómetro de fabricación propia de tipo rotacional, constituido por tres cazoletas situadas perpendicularmente al eje de rotación formando ángulos de 120º entre sí.

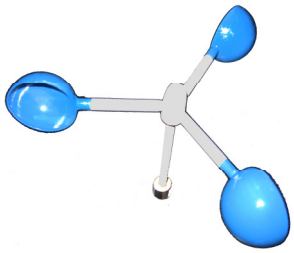


Ilustración 2.25: Anemómetro rotacional de cazoletas utilizado en este proyecto.

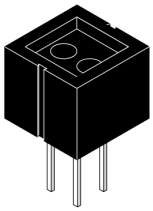


Ilustración 2.24: CNY70. Sensor emisor-receptor de infrarrojos.

El eje principal estará marcado de tal forma que mediante el empleo de un sensor óptico de luz infrarroja CNY70 registraremos el paso de la marca del eje por el sensor, generando una onda pulsatoria rectangular cuyo periodo determinará la velocidad angular del eje.

Conocida la distancia entre el centro de la cazoleta y el centro del eje principal podemos determinar que:

Ec. 2.33: $$v = w \cdot r$$

Ec. 2.34: $$w = 2\pi f = \frac{2\pi}{T}$$

De ambas ecuaciones deducimos:

Ec. 2.35:
$$v = \frac{2\pi r}{T}$$

La solución propuesta para la medida de la velocidad del viento, requiere un circuito capaz de detectar el paso de la marca del eje y amplificar la señal generada a un valor de 1 lógico para que sea detectado por el puerto CCP1 del PIC.

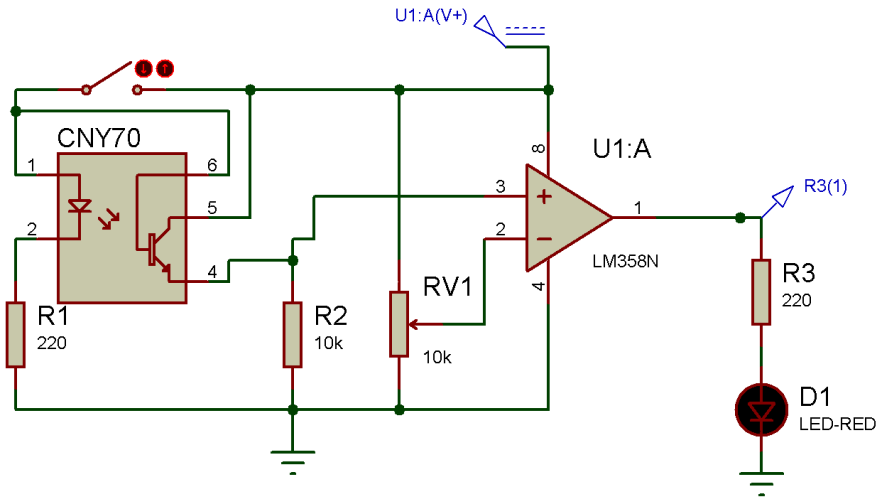


Ilustración 2.26: Modelado del circuito acondicionador para el anemómetro.

Nota: El modelo del CNY70 no corresponde al modelo real de dicho sensor óptico. Se trata de un modelo de optoacoplador genérico empleado para el circuito esquemático. En el elemento real, la unión entre las patillas 6 y 1 se produce internamente y el interruptor no es necesario ya que el transistor entra en saturación al recibir la reflexión del infrarrojo en la superficie reflectante del eje del anemómetro.

El PIC, por su parte, empleando el temporizador *TIMER1* determinará el tiempo transcurrido entre dos flancos de subida de la señal sensada, determinando el periodo de la onda y por tanto la velocidad del viento.

Para el circuito acondicionador se ha empleado un amplificador operacional en configuración comparador, con objeto de obtener a la entrada del CCP1 valores de 0V o 5V según el estado de corte o saturación, respectivamente, como consecuencia de la reflexión del infrarrojo del mismo encapsulado.

2.3.1.5 Alternativa propuesta para la Veleta.

El objeto de la veleta es determinar la dirección desde la que sopla el viento. En los sistemas de anemómetros rotacionales de hélice, aprovechando que este debe estar alineado con la ráfaga de viento, se emplea un sistema combinado en el que la veleta es el cuerpo del anemómetro.

En nuestro caso, emplearemos una veleta clásica de fabricación propia, en cuyo eje se dispondrá de manera solidaria un disco codificado que determinará su orientación. La codificación del disco se realizará en código gray, con objeto de que por cada cambio de posición tan solo se produzca el cambio de un bit.

Para determinar la posición de la veleta, se contemplarán 16 posiciones, coincidentes con las 16 principales posiciones de la rosa de los vientos, conforme a la metodología habitual para la identificación de vientos. Para ello, se alinearán 4 sensores CNY70 (2^4=16 posiciones) que serán enfrentados al disco codificado que indicará el posicionamiento de la veleta.

Las siguientes figuras muestran los 16 principales vientos considerados en nuestro proyecto y el disco codificado sobre el que se realizará la lectura.

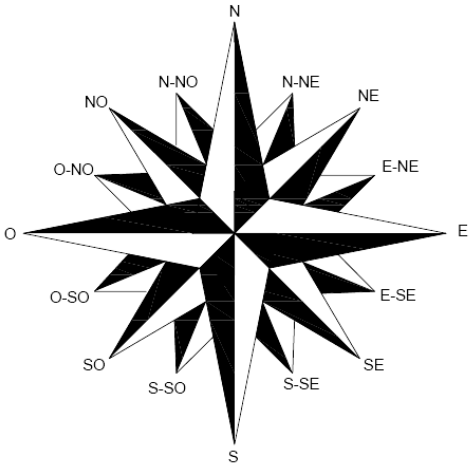


Ilustración 2.28: Rosa de los vientos de 16 puntos.

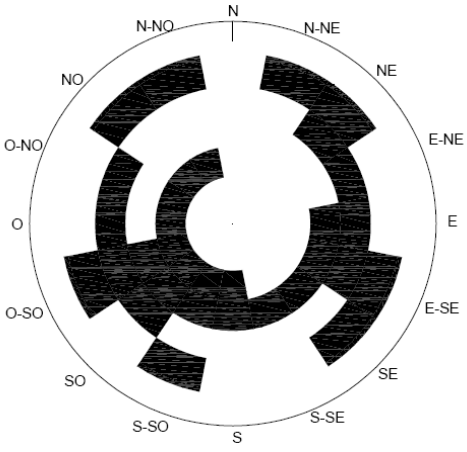


Ilustración 2.27: Disco codificado para determinar el posicionamiento de la veleta.

Los sensores empleados, como ya he comentado, son el mismo modelo que el empleado para el anemómetro. Puesto que las características de funcionamiento deseadas para el circuito acondicionador son las mismas (detectar marca o no y obtener por

comparación niveles lógicos alto o bajo) emplearemos el mismo circuito de la Ilustración 2.26 para cada uno de los cuatro sensores.

2.3.1.6 Alternativa propuesta para el Piranómetro.

Un Piranómetro o solarímetro es un instrumento meteorológico utilizado para medir la radiación solar incidente sobre la superficie de la tierra. Esta magnitud se mide en kilovatios por metro cuadrado. Fundamentalmente estos dispositivos funcionan bajo una tecnología de termopares. El efecto térmico producido por el sol sobre los termopares produce una pequeña fuerza electromotriz que al ser medida permite establecer una relación entre f.e.m. y radiación solar incidente.

Por otro lado, hay que considerar los piranómetros de efecto fotovoltaico. Como su nombre indica, estos piranómetros no funcionan bajo la tecnología de termopares sino, que emplean un fotodiodo que permite diferenciar el espectro solar por la frecuencia de la onda electromagnética, y así, mediante la medida de voltaje, conocer la radiación incidente. Estos piranómetros tienen el inconveniente de ser más sensibles a irregularidades y variaciones al carecer de la inercia térmica de los termopares.

En ambos sistemas, para garantizar el rango de longitud de onda medida, suelen emplearse filtros a modo de cúpulas o coberturas de vidrio que bloquean las longitudes de ondas no deseadas. Según su posicionamiento y los elementos de bloqueo antepuestos a su superficie de exposición a la radiación solar, estos elementos medirán:

- ✓ **Radiación Global**: El elemento se dispone sobre una superficie horizontal y mide la radiación directa del sol más la radiación difusa dispersada al atravesar la atmosfera.

- ✓ **Radiación incidente sobre superficies no horizontales**: El elemento se dispone sobre una superficie inclinada y el resultado de la medición es consecuencia de la suma de las radiaciones directa, difusa y reflejada para ese plano de inclinación.

- ✓ **Radiación global reflejada (Albedo):** El elemento se dispone en posición invertida respecto al plano horizontal y el resultado de su medida representa la radiación reflejada por la tierra.

- ✓ **Radiación difusa:** El elemento se dispone sobre un plano horizontal y se emplea una pantalla que bloquea la radiación directa, obteniéndose como resultado la radiación difusa dispersada como consecuencia de su entrada en la atmosfera.

Para el caso particular de este proyecto, se propone el uso de un conjunto de 4 células fotovoltaicas a modo de piranómetro fotovoltaico. El objetivo es medir la radiación solar global incidente sobre una superficie horizontal. Está magnitud conocida como Irradianza se medirá en W/m^2.

Las células fotovoltaicas escogidas corresponden al modelo C-0120 de CEBEK. Cada célula ofrece una tensión nominal de 0.45V y 100mA de corriente y se extiende sobre una superficie de $1125mm^2$ (45x25mm). El conjunto de células se han conexionado en serie para obtener una tensión nominal de 1,8V y una corriente nominal de 100mA.

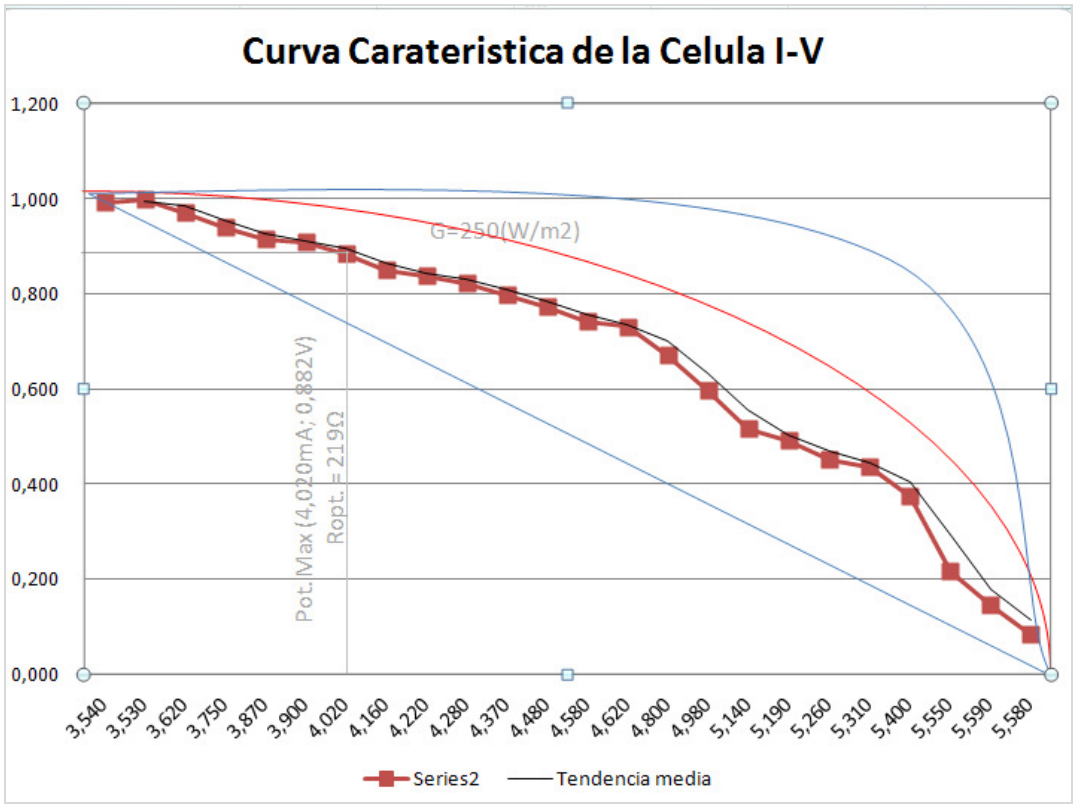


Ilustración 2.29: Curva característica Intensidad-Tensión de la célula fotovoltaica.

La curva característica de una célula fotovoltaica (tensión-intensidad) representa pares de tensión e intensidad en los que puede encontrarse funcionando la célula. Esta curva responde a la expresión:

Ec. 2.36:
$$I = I_{SC}\left(1 - e^{\frac{\varepsilon(V_{CO}-V)}{mKT}}\right)$$

Siendo:

ε: carga eléctrica del electrón 1.6021x10-19 C.
m: Parámetro constructivo de la célula (normalmente 1)
K: Constante de Boltzmann =1.3806504x10^{-23}J/K
T: Temperatura en grados Kelvin

Los valores característicos de esta gráfica son:

- ✓ Tensión a circuito abierto (V_{CO}): es el máximo valor de tensión en extremos de la célula y se da cuando esta no está conectada a ninguna carga.

- ✓ Corriente de cortocircuito (I_{SC}): definido como el máximo valor de corriente que circula por una célula fotovoltaica y se da cuando la célula está en cortocircuito.

- ✓ Punto de máxima potencia (PMP): Es el producto del valor de tensión máxima (V_M) e intensidad máxima (I_M) para los que la potencia entregada a una carga es máxima.

- ✓ Factor de forma(FF): Se define como el cociente de potencia máxima que se puede entregar a una carga entre el producto de la tensión de circuito abierto y la intensidad de cortocircuito, es decir:

Ec. 2.37:
$$FF = \frac{P_M}{V_{OC} \cdot I_{SC}} = \frac{V_M \cdot I_M}{V_{OC} \cdot I_{SC}}$$

- ✓ Rendimiento (η): Se define como el cociente entre la máxima potencia eléctrica que se puede entregar a la carga (PMP) y la irradiancia incidente (PL) sobre la célula, que es el producto de la irradiancia incidente G por el área de la célula S.

Ec. 2.38:
$$\eta = \frac{PM}{PL} = \frac{V_M \cdot I_M}{V_L \cdot I_L}$$

La curva característica I-V representada en la Ilustración 2.29 ha sido realizada empíricamente mediante la exposición del conjunto de células a una radiación lumínica

constante, soportando una carga variable de valor inicial 282Ω que se ha ido reduciendo hasta 14,5Ω.

Nota: El método normalizado para la realización de esta grafica es con una Irradianza de 1000W/m2 y una temperatura ambiente de 25ºC.

Realizando esta misma prueba bajo radiación solar, obtenemos los datos que se reflejan en la siguiente tabla:

PRUEBA EN CRTO. CTO	Superficie	0,013				
I (mA)	V (mv)	R(ohm)	Pot(W)	G(W/m²)	Pot(mV)	G(mW/m²)
7,40	1920	259,46	0,0142	1,0929	14,21	1092,92
8,00	1913	239,13	0,0153	1,1772	15,30	1177,23
9,10	1902	209,01	0,0173	1,3314	17,31	1331,40
10,30	1890	183,50	0,0195	1,4975	19,47	1497,46
11,50	1878	163,30	0,0216	1,6613	21,60	1661,31
12,90	1863	144,42	0,0240	1,8487	24,03	1848,67
14,40	1849	128,40	0,0266	2,0481	26,63	2048,12
17,30	1821	105,26	0,0315	2,4233	31,50	2423,33
20,10	1793	89,20	0,0360	2,7723	36,04	2772,25
23,50	1760	74,89	0,0414	3,1815	41,36	3181,54
25,10	1741	69,36	0,0437	3,3615	43,70	3361,47
27,00	1720	63,70	0,0464	3,5723	46,44	3572,31
29,10	1700	58,42	0,0495	3,8054	49,47	3805,38
31,30	1677	53,58	0,0525	4,0377	52,49	4037,70
32,90	1657	50,36	0,0545	4,1935	54,52	4193,48
36,30	1616	44,52	0,0587	4,5124	58,66	4512,37
50,30	1451	28,85	0,0730	5,6143	72,99	5614,25
68,00	1153	16,96	0,0784	6,0311	78,40	6031,08
75,40	962	12,76	0,0725	5,5796	72,53	5579,60
89,80	240	2,67	0,0216	1,6578	21,55	1657,85
91,50	86	0,94	0,0079	0,6053	7,87	605,31
91,60	56	0,61	0,0051	0,3946	5,13	394,58

Pmax	Gmax
0,0784	6,0311

Tabla 2.6: Tabla de resultados de la variación de la carga del circuito.

De estos datos obtenemos las gráficas características de Intensidad-Tensión, Potencia e Irradianza.

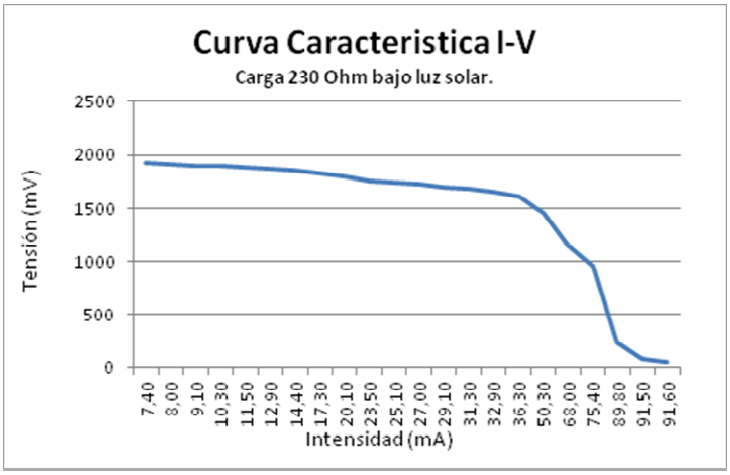


Ilustración 2.30: Curva caracteristica Corriente-Tension bajo luz solar.

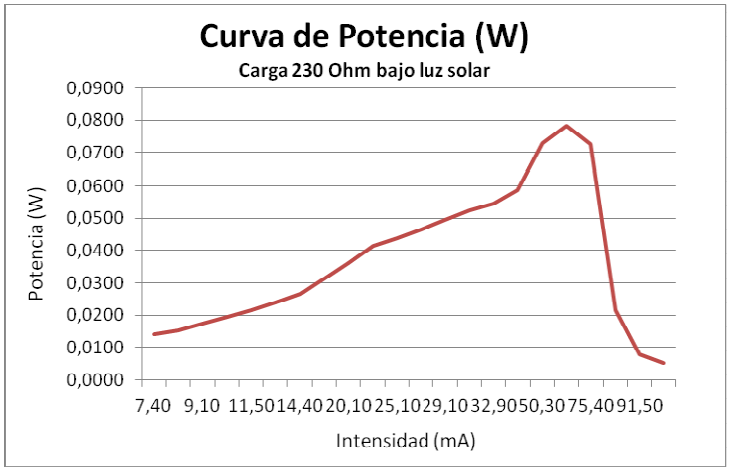


Ilustración 2.31: Curva característica de Potencia bajo luz solar.

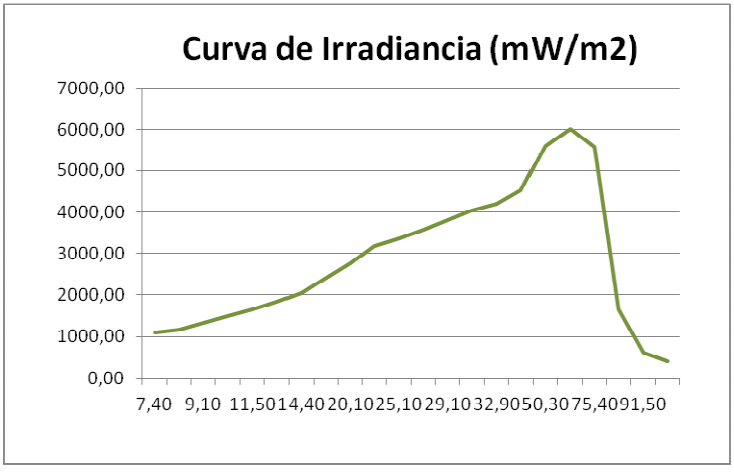


Ilustración 2.32: Curva característica de Irradianza bajo luz solar.

La metodología para medir la Irradianza en este proyecto consiste en medir la tensión generada por la placa como respuesta a la radiación solar global. Puesto que la carga del circuito es conocida, podemos determinar la intensidad que circula por ella y de este modo, la potencia generada por el conjunto de las células fotovoltaicas.

Ec. 2.39 **:** $$G\left[\frac{W}{m^2}\right]=\frac{P[W]}{S[m^2]}$$

Ec. 2.40: $$P[W]=U[V]\cdot I[A]=U[V]\frac{U[V]}{R[\Omega]}=\frac{U^2}{R}[W]$$

Como ya hemos definido antes, la Irradianza es la potencia por unidad de superficie por lo que como también conocemos la superficie total de las células ya tendríamos determinada la Irradianza.

Ec. 2.41: $$G\left[\frac{W}{m^2}\right]=\frac{P[W]}{S[m^2]}=\frac{U^2}{R\cdot S}\left[\frac{W}{m^2}\right]$$

Los valores críticos de la célula son sus puntos nominales de trabajo.

$$Vnom=1{,}8V$$

$$Inom=100mA$$

Para asegurar esta relación, se determina el valor de la resistencia mínima de carga a aplicar al circuito. Aplicando la ley de Ohm:

Ec. 2.42: $$R_{carga}\min=\frac{Vnom}{Inom}=\frac{1{,}8V}{100mA}=18\Omega$$

Emplearemos una resistencia de carga de 20Ω

$$R_{carga}=20\Omega$$

Puesto que el valor de señal obtenido será digitalizado por el ADC del PIC, debemos acondicionar la señal para realizar una lectura lo más precisa posible. Para ello amplificaremos el valor de tensión para obtener un rango de 0-5V.

Ec. 2.43: $$\Delta v=\frac{V_{out}}{V_{in}}=\frac{5V}{1{,}8V}=2{,}78$$

Este acondicionamiento se realizará mediante un amplificador operacional con realimentación negativa en configuración no inversora, tal y como se muestra en la Ilustración 2.33, por tanto $V^- = V^+ = V_{in}$:

Ec. 2.44:
$$\frac{0 - V^-}{R_1} = \frac{V^- - V_{out}}{R_2}$$

Resolviendo:

Ec. 2.45:
$$V_{out} = \left(1 + \frac{R_2}{R_1}\right) \cdot V_{in}$$

De donde:

Ec. 2.46:
$$\frac{V_{out}}{V_{in}} = \left(1 + \frac{R_2}{R_1}\right) = 2.78$$

Por tanto determinamos que:

$$\boxed{R_1 = 1k\Omega \Rightarrow \quad R_2 = 1.78k\Omega}$$

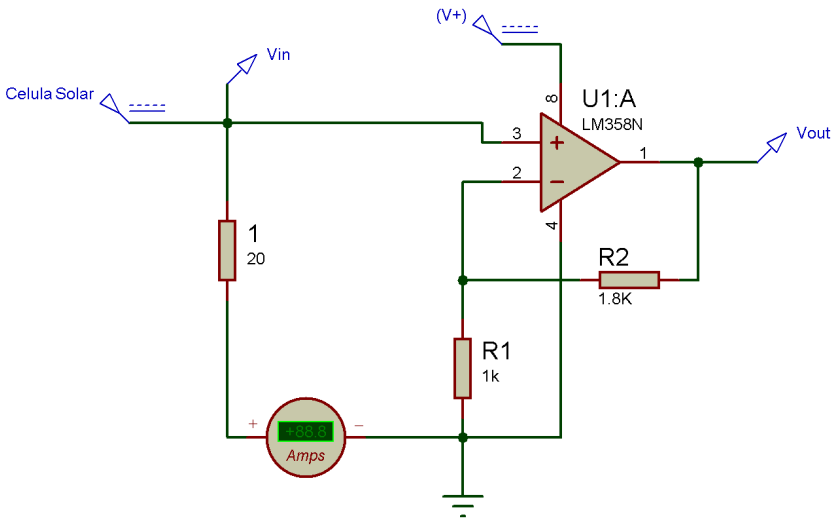
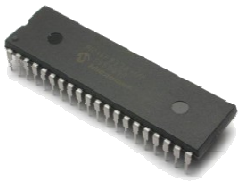


Ilustración 2.33: Circuito acondicionador para la medición de radiación solar.

La ecuación final por tanto quedará como:

Ec. 2.47:
$$G = \left(\frac{V_{OUT}}{\Delta V}\right)^2 \cdot \frac{1}{R \cdot S} = \left(\frac{V_{OUT}}{2.78}\right)^2 \cdot \frac{1}{20 \cdot 1{,}125 \cdot 10^{-3}} = 5.75 \cdot {V_{OUT}}^2 \left[\frac{W}{m^2}\right]$$

2.3.2 El PIC como elemento de la tarjeta de adquisición.



El PIC16F877 como elemento de la tarjeta de adquisición es un circuito integrado más. Para su funcionamiento no requiere apenas elementos externos. Basta con una alimentación positiva de 5Vdc en V_{SS} y una referencia a masa 0V en V_{DD}, una frecuencia de reloj que en nuestro caso es suministrada mediante un cristal de cuarzo de 4Mhz y dos condensadores de 33pF.Con estos requisitos y la patilla nº1 RESET/MCLR conectada a masa, el PIC puede comenzar a ejecutar su firmware previamente grabado.

A continuación se muestra el esquema básico de conexionado del PIC16F877 para su funcionamiento.

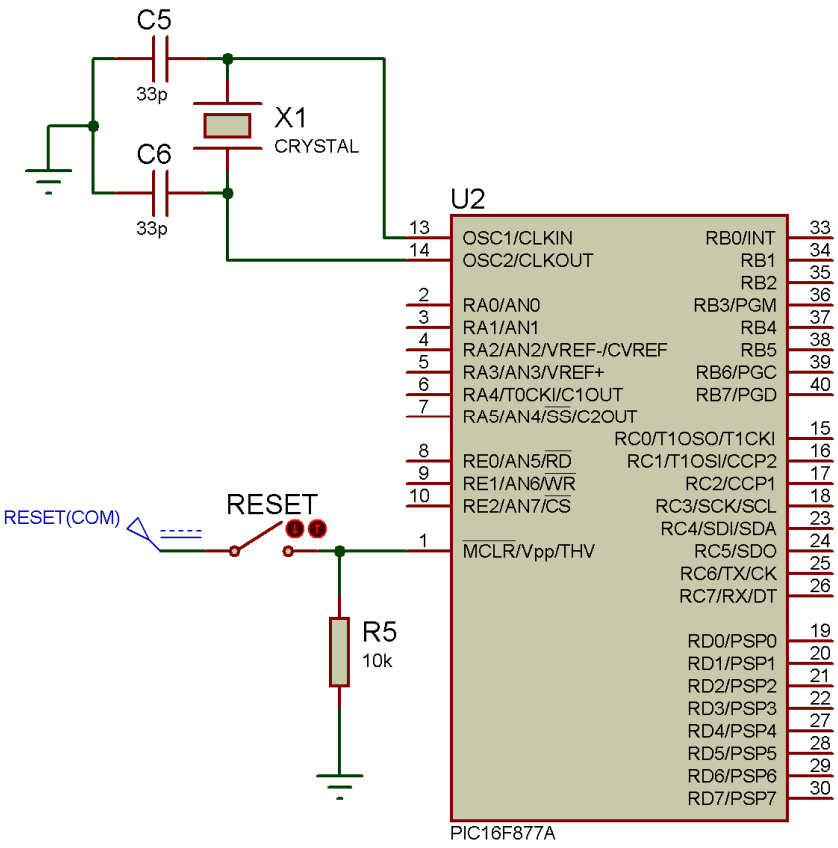
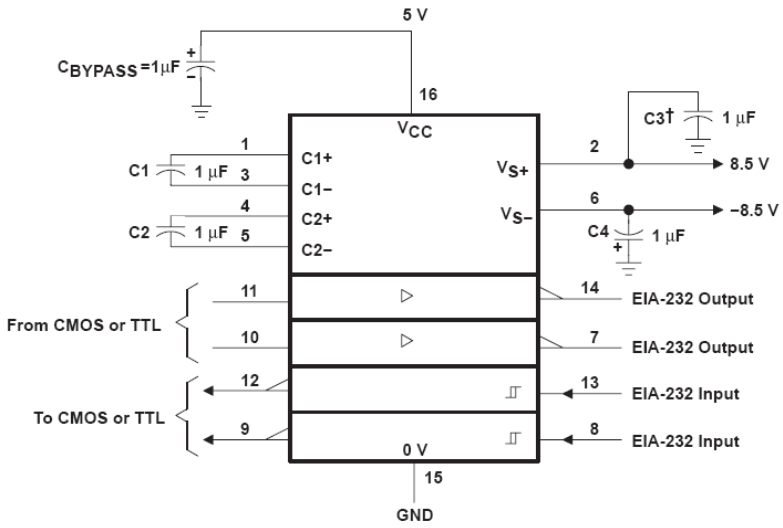


Ilustración 2.34: Esquema básico de conexionado del PIC16F877 para su funcionamiento.

En los puertos empleados como entradas, con objeto de establecer un nivel de masa estable y minimizar los posibles efectos de "falsos unos" es aconsejable el uso de

resistencias PULL-DOWN que mantengan el puerto a masa mientras la entrada no está a nivel alto. (En caso de que sean entradas a nivel bajo, es aconsejable hacer uso de resistencias PULL-UP).

Aunque el PIC está dotado de puerto serie, no es posible hacer una conexión directa entre el puerto serie del PIC y el puerto serie del PC. Esto es debido a que el PIC trabaja con niveles TTL de tensión y el PC trabaja con la norma RS232 que establece tensiones de ±15V para el puerto serie. Por ello es preciso emplear una interfaz que adapte ambos niveles de tensión en cada dispositivo, de lo contrario, los niveles de tensión del PC quemarían el puerto serie del PIC.



El elemento destinado a tal tarea es el integrado MAX232. Como se muestra en el esquema de la figura, con el uso de algunos condensadores junto con el integrado MAX232 podemos adaptar hasta dos entradas de entrada y dos de salida.

Ilustración 2.35: Integrado MAX232.Adaptador de niveles de tensión.

Realmente esto es todo lo necesario para trabajar con un PIC. Los circuitos de acondicionamiento de señal, son circuitos propios de la adquisición de señales analógicas, destinados a mejorar los rangos y características de la adquisición. Y en cuanto a las salidas, obviamente, cada una llevará su propio hardware asociado, en función de sus características, pero no por necesidad del PIC.

2.3.3 Elementos importantes en la configuración y programación del PIC.

Una vez analizados y establecidos los sensores que emplearemos, estamos en disposición de determinar el modelo de microprocesador que podemos emplear (recordemos el apunte que hicimos a este respecto sobre la simultaneidad de ambos

procesos de diseño), siendo el elegido el PIC16F877/A cuyas características cumplen con los requisitos mínimos establecidos por los sensores.

Periférico / Módulo	Necesidad	PIC16F877
Entradas Digitales	6 bits	25 bits analógicos
Entradas Analógicas	3 bits	+ 8 analógicos/digitales
CAD (Convertidor Analógico Digital)	3 canales	8 canales
CCP (Captura / Comparación / PWM)	2 módulos	2 módulos
Puerto de comunicaciones serie AUSART (Addressable Universal Synchronous Asynchronous Receiver Transmitter)	1 módulo	1 módulo

Tabla 2.7: Comparativa entre características deseadas y características reales del PIC

Antes de abordar la programación del firmware del PIC, es interesante hacer una explicación sobre los módulos o periféricos del PIC involucrados en el diseño con objeto de conocer su funcionamiento y configuración.

2.3.3.1 El Convertidor Analógico Digital (CAD)

El PIC16F877 dispone de un *CAD* multiplexado de 8 canales y 10 bits de resolución que podemos gobernar mediante el uso de cuatro registros:

Los registros de *ADCON0* y *ADCON1* son los registros mediante los cuales configuraremos y gobernaremos al *CAD*.

El diseñador, mediante la configuración de los bits (*PCFG3:PCFG0*) del registro *ADCON1* (Tabla 2.8), puede configurar hasta un total de 8 entradas analógicas y hacer uso de ellas multiplexando la entrada del *CAD* mediante el control de los bits (*CHS2:CHS0*) del registro *ADCON0* (Tabla 2.10).

ADCON1 REGISTER (ADDRESS 9Fh)

U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

Tabla 2.8: Registro ADCON1. (datasheet PIC16F877)

Véase en la Tabla 2.9 la configuración de entradas analógicas o digitales obtenida según los diferentes valores de los bits PCFx del registro *ADCON1*. Nótese que el convertidor contempla la posibilidad de configurar los pines *AN2* y *AN3* del PIC como

valores analógicos de referencia superior e inferior para la conversión. De no configurarse esta opción, el PIC tomará como valores de referencia interna los correspondientes a V_{SS} y V_{DD}.

PCFG3:PCFG0: A/D Port Configuration Control bits:

PCFG3: PCFG0	AN7(1) RE2	AN6(1) RE1	AN5(1) RE0	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AN0 RA0	VREF+	VREF-	CHAN/ Refs(2)
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	RA3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	RA3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	RA3	VSS	2/1
011x	D	D	D	D	D	D	D	D	VDD	VSS	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	RA3	RA2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	RA3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	RA3	RA2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	RA3	RA2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	RA3	RA2	1/2

A = Analog input D = Digital I/O

Tabla 2.9: Bits de control para la configuración de los puertos analógicos/digitales. (datasheet PIC16F877)

ADCON0 REGISTER (ADDRESS: 1Fh)

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

Tabla 2.10: Registro ADCON0. (datasheet PIC16F877)

La Tabla 2.11 muestra el canal seleccionado como entrada en el *CAD* (y el PIN correspondiente) en función de los bits de configuración del multiplexor.

CHS2	CHS1	CHS0	CANAL	PIN
0	0	0	Canal 0	RA0 / AN0
0	0	1	Canal 1	RA1 / AN1
0	1	0	Canal 2	RA2 / AN2
0	1	1	Canal 3	RA3 / AN3
1	0	0	Canal 4	RA5 / AN4
1	0	1	Canal 5	RE0 / AN5
1	1	0	Canal 6	RE1 / AN6
1	1	1	Canal 7	RE2 / AN7

Tabla 2.11: Bits de selección CHSx del canal del multiplexor.

Para poder ajustarnos a nuestras necesidades de muestreo, disponemos de la posibilidad de seleccionar la *frecuencia de reloj* que se empleará en la conversión mediante el *pre-escaler* de la *frecuencia de trabajo* del PIC. Para ello configuraremos los bits *ADCS1* y *ADCS0* del registro *ADCON0*. Disponiendo de las siguientes posibilidades:

ADCS1:0	FRECUENCIA	T_{AD}
00	$F_{OSC}/2$	$2*T_{OSC}$
01	$F_{OSC}/8$	$8*T_{OSC}$
10	$F_{OSC}/32$	$32*T_{OSC}$
11	F_{RC} (oscilador interno)	Oscilador RC interno

Tabla 2.12: Configuración de la frecuencia de trabajo del CAD.

El tiempo de conversión por bit se define como T_{AD}. Para una conversión de 10 bits el convertidor requiere un mínimo de $12*T_{AD}$. Para asegurar una correcta conversión, el reloj del convertidor debe ser configurado para asegurar un mínimo de 1,6 microsegundos/bit.

Los registros *ADRESH* y *ADRESL* son el destino del resultado de la conversión. Se trata de registros de 8 bits que representan el nibble alto (Hi) y bajo (Low) de la palabra de conversión. Este resultado puede justificarse a la izquierda (mostrando los 6 bits menos significativos como 0) o a la derecha (6 bits más significativos como 0) mediante el bit de configuración *ADMF* del registro *ADCON1*.

Para realizar la conversión del valor analógico, una vez configurado el reloj, la justificación y el canal, ha de habilitarse mediante el bit *ADON* del registro *ADCON0* y dejar $2T_{AD}$ antes de requerir el inicio de conversión mediante el bit *GO/DONE* del mismo registro. Durante la conversión *GO/DONE* se mantendrá en 1 y pasará a 0 cuando finalice, momento en el que podremos disponer de el valor de 10bits en los registros *ADRESx*.

2.3.3.2 CCP (Módulo Captura / Comparación / PWM).

Como ya hemos mencionado, el PIC16F877 viene dotado con módulos *CCP*. Cada módulo dispone de dos registros de 8 bits que pueden funcionar como:

- ✓ Registro de captura de 16 bits
- ✓ Registro de comparación de 16 bits
- ✓ Registro de parámetros del *PWM*.

Cada modo de trabajo opera idénticamente salvo por el evento de disparo. Los recursos de los que dispone el modulo *CCP* se muestran en la Tabla 2.13.

MODO DE TRABAJO CCP	ACTUACIÓN SOBRE EL TIMER
Comparación	*TIMER1*
Captura	*TIMER1*
PWM	*TIMER2*

Tabla 2.13: Recursos de *TIMER* para los modos de trabajo del *CCP*.

La interacción entre recursos cuando se hace uso de ambos módulos a la vez se resume en la Tabla 2.14.

MODO CCPX	MODO CCPY	INTERACCIÓN
Captura	Captura	La base de tiempo en *TIMER1* debe ser la misma
Captura	Comparación	La comparación debe ser configurada mediante el disparador de eventos especiales, el cual borra el *TIMER1*
Comparación	Comparación	Las comparaciones deben ser configuradas mediante el disparador de eventos especiales, el cual borra el *TIMER1*
PWM	*PWM*	Ambos *PWM* deben trabajar con la misma frecuencia
PWM	Captura	No existen restricciones de interacción
PWM	Comparación	No existen restricciones de interacción.

Tabla 2.14: Restricciones en el uso de recursos en el uso simultaneo de ambos módulos *CCP*.

El registro *CCPRx* de 16 bits (entendiendo por x como 1 para *CCP1* y 2 para *CCP2*) se compone de dos registros de 8 bits, *CCPRxH* que representan el nibble alto y *CCPRxL* que representa al nibble bajo.

Mediante el registro *CCPxCON* (ver Tabla 2.15) se controla el funcionamiento de los módulos *CCPx*.

Dado que en este proyecto en concreto, tan solo se hace uso de los módulos *CCP1* y *CCP2* en modo de Captura, limitaremos la explicación de su funcionamiento y configuración a este modo de funcionamiento.

CCP1CON REGISTER/CCP2CON REGISTER (ADDRESS: 17h/1Dh)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CCPxX	CCPxY	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

Tabla 2.15: Registro CCP1CON. (datasheet PIC16F877)

La función del *CCP* en modo Captura es identificar un determinado evento, configurado previamente mediante los bits *CCPxM3:CCPxM0* del registro *CCPxON*. Estos bits configuran el modo de funcionamiento (captura, comparación o *PWM*) y el evento a identificar a través del puerto *CCPx*, conforme se indica en la Tabla 2.16.

<CCPXM3: CCPXM0>	MODO DE TRABAJO	EVENTO
0000	Desactivado	Desactivado.
0100	Captura	Cada flanco de bajada.
0101	Captura	Cada flanco de subida.
0110	Captura	Cada 4 flancos de subida.
0111	Captura	Cada 16 flancos de subida.
1000	Comparación	Se activa la salida como resultado de la comparación (*CCPxIF*=1).
1001	Comparación	Se borra la salida como resultado de la comparación (*CCPxIF*=1).
1010	Comparación	Se genera la interrupción por software (*CCPxIF*=1; *CCPx* no esta afectado en este modo).
1011	Comparación	Activación de eventos especiales (*CCPxIF*=1; *CCPx* no se ve afectado). *CCP1* resetea *TMR1*; *CCP2* restablece *TMR1* e inicia una conversión A/D (si el módulo A / D está habilitado).
11xx	*PWM*	No requiere ningún evento.

Tabla 2.16: Bits de selección del modo de trabajo del *CCPx*.

Veamos entonces el proceso de trabajo del *CCP* en modo captura.

Se configurará el puerto *CCPx* como entrada y el *TIMER1* como temporizador, recordemos que el modo captura utiliza como recurso temporizador el *TIMER1* (ver Tabla 2.13) e introduciremos la señal que debe provocar el evento por el pin correspondiente al puerto *CCPx*. De esta forma, conforme a la configuración que hayamos establecido el PIC esperará el evento que produzca el disparo del *CCP* (un flanco de subida por ejemplo). En ese momento se producirá una interrupción que establecerá el bit de bandera *CCPxIF* a 1 y

almacenará el contenido del *TIMER1* (*TMR1H:TMR1L*) en el registro *CCPRx* (*CCPRxH:CCPRxL*).

2.3.3.3 Puerto de comunicaciones USART.

El puerto *USART* también conocido como *SCI* (interfaz de comunicación serie) del PIC16F877 puede ser configurado en los siguientes modos de trabajo:

- ✓ Asíncrono (full dúplex)
- ✓ Síncrono - Maestro (half dúplex)
- ✓ Síncrono - Esclavo (half dúplex)

Continuando con la metodología empleada en los periféricos anteriormente descritos, tan solo abordaremos la explicación sobre el sistema de comunicación asíncrono (full dúplex) utilizado en este proyecto.

En el modo de comunicación asíncrono, las transferencias de información se realizan sobre dos líneas *TX* (transmisión) y *RX* (recepción), al ritmo de una frecuencia controlada internamente por el *USART*. Al tratarse de una comunicación full-duplex las operaciones de recepción y transmisión pueden realizarse simultáneamente.

En el intercambio de información, cada palabra se trata de forma independiente a las demás conforme al formato estándar *NRZ* (NonReturn-to-Zero). Según este estándar los datos se envían en paquetes de 8 o 9 bits precedidos por un bit de inicio (*START*) y cerrando el envío con un bit de parada (*STOP*) a frecuencias normalizadas.

El funcionamiento del *USART* en modo asíncrono puede descomponerse en cuatro bloques funcionales:

- ✓ El generador de baudios
- ✓ La transmisión síncrona
- ✓ La recepción síncrona.
- ✓ El circuito de muestreo.

a) El Generador de Baudios.

El *USART* dispone de un *Baud Rate Generator* (generador de velocidad de transmisión) integrado, cuya función es generar a partir de la frecuencia del oscilador las velocidades estándares de transmisión. En el modo asíncrono, es posible ampliar la gama de frecuencias a "altas velocidades" mediante el valor del bit *BRGH*. Si el valor de *BRGH*=0 manejaremos la gama denominada "baja velocidad" y si su valor es 1 la denominada "alta velocidad". La configuración del bit afecta a la frecuencia conforme se muestra en la siguiente tabla:

BRGH=0 (baja velocidad)	**BRGH=1 (alta velocidad)**
$BaudRate=F_{OSC}/(64(x+1))$	$BaudRate=F_{OSC}/(16(x+1))$
X= valor comprendido entre 0 y 255 seleccionado en el registro *SPBRG*	

Tabla 2.17: Efecto del bit BRGH sobre la velocidad de transmisión.

El registro *SPBRG* como se ha indicado en la tabla anterior, contiene la variable mediante la cual configuraremos la velocidad de transmisión con la que deseamos trabajar. Este valor debe ser un valor entero y debe hacer referencia a una frecuencia de comunicación normalizada, por lo que no podremos tomar cualquier número. A continuación se muestra la Tabla 2.18 y la Tabla 2.19 obtenidas del datasheet del PIC16F877, donde se relacionan las velocidades de transmisión habituales con los valores de *SPBRG* y su porcentaje de error cuantificado.

BAUD RATE (K)	Fosc = 20 MHz			Fosc = 16 MHz			Fosc = 10 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	-	-	-	-	-	-	-	-	-
1.2	1.221	1.75	255	1.202	0.17	207	1.202	0.17	129
2.4	2.404	0.17	129	2.404	0.17	103	2.404	0.17	64
9.6	9.766	1.73	31	9.615	0.16	25	9.766	1.73	15
19.2	19.531	1.72	15	19.231	0.16	12	19.531	1.72	7
28.8	31.250	8.51	9	27.778	3.55	8	31.250	8.51	4
33.6	34.722	3.34	8	35.714	6.29	6	31.250	6.99	4
57.6	62.500	8.51	4	62.500	8.51	3	52.083	9.58	2
HIGH	1.221	-	255	0.977	-	255	0.610	-	255
LOW	312.500	-	0	250.000	-	0	156.250	-	0

BAUD RATE (K)	Fosc = 4 MHz			Fosc = 3.6864 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	0.300	0	207	0.3	0	191
1.2	1.202	0.17	51	1.2	0	47
2.4	2.404	0.17	25	2.4	0	23
9.6	8.929	6.99	6	9.6	0	5
19.2	20.833	8.51	2	19.2	0	2
28.8	31.250	8.51	1	28.8	0	1
33.6	-	-	-	-	-	-
57.6	62.500	8.51	0	57.6	0	0
HIGH	0.244	-	255	0.225	-	255
LOW	62.500	-	0	57.6	-	0

Tabla 2.18: Velocidades de transmisión en modo asíncrono para *BRGH*=0. (datasheet PIC16F877)

BAUD RATE (K)	Fosc = 20 MHz			Fosc = 16 MHz			Fosc = 10 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	-	-	-	-	-	-	-	-	-
1.2	-	-	-	-	-	-	-	-	-
2.4	-	-	-	-	-	-	2.441	1.71	255
9.6	9.615	0.16	129	9.615	0.16	103	9.615	0.16	64
19.2	19.231	0.16	64	19.231	0.16	51	19.531	1.72	31
28.8	29.070	0.94	42	29.412	2.13	33	28.409	1.36	21
33.6	33.784	0.55	36	33.333	0.79	29	32.895	2.10	18
57.6	59.524	3.34	20	58.824	2.13	16	56.818	1.36	10
HIGH	4.883	-	255	3.906	-	255	2.441	-	255
LOW	1250.000	-	0	1000.000		0	625.000	-	0

BAUD RATE (K)	Fosc = 4 MHz			Fosc = 3.6864 MHz		
	KBAUD	% ERROR	SPBRG value (decimal)	KBAUD	% ERROR	SPBRG value (decimal)
0.3	-	-	-	-	-	-
1.2	1.202	0.17	207	1.2	0	191
2.4	2.404	0.17	103	2.4	0	95
9.6	9.615	0.16	25	9.6	0	23
19.2	19.231	0.16	12	19.2	0	11
28.8	27.798	3.55	8	28.8	0	7
33.6	35.714	6.29	6	32.9	2.04	6
57.6	62.500	8.51	3	57.6	0	3
HIGH	0.977	-	255	0.9	-	255
LOW	250.000	-	0	230.4	-	0

Tabla 2.19: Velocidades de transmisión en modo asíncrono para *BRGH*=1. (datasheet PIC16F877)

b) La transmisión asíncrona.

Para poder explicar la transmisión asíncrona, haremos uso del diagrama de bloques que viene representado en el datasheet del PIC16F877.

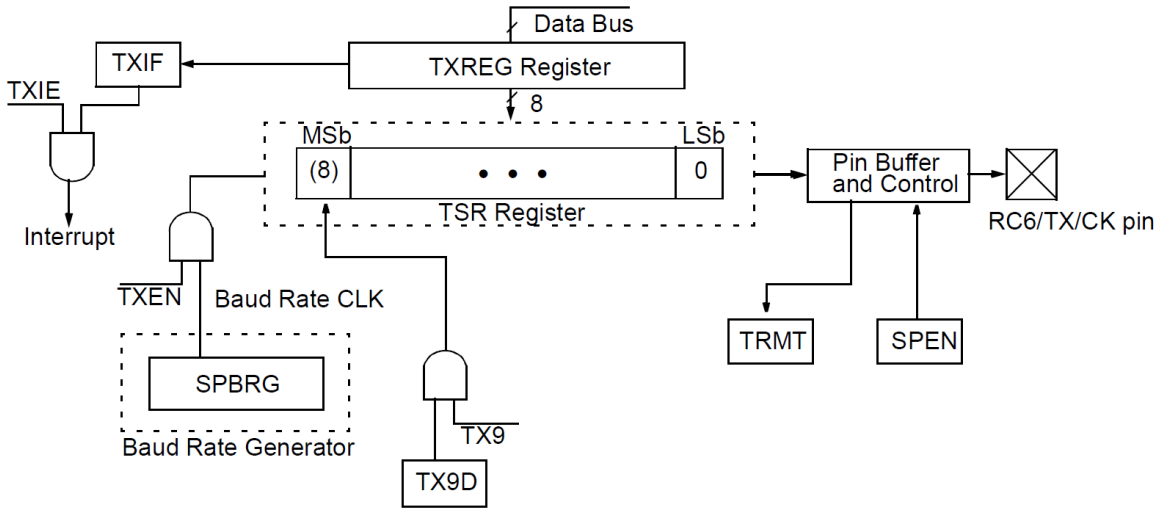


Ilustración 2.36: Diagrama de bloque de la transmisión asíncrona. (datasheet PIC16F877)

Atendiendo al diagrama de la ilustración, para llevar a cabo el proceso de una transmisión debemos actuar sobre 4 registros, que explicamos a continuación:

- ✓ RCSTA (RECEIVE STATUS AND CONTROL REGISTER).Tabla 2.20
- ✓ TXSTA (TRANSMIT STATUS AND CONTROL REGISTER).Tabla 2.21
- ✓ PIE1 (PERIPHERAL INTERRUPTS ENABLE REGISTER).Tabla 2.22
- ✓ PIR1(PERIPHERAL INTERRUPTS REGISTER).Tabla 2.23

RCSTA: RECEIVE STATUS AND CONTROL REGISTER (ADDRESS 18h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

Tabla 2.20: Registro de control y estado de la recepción. (datasheet PIC16F877)

El registro *RCSTA* es el destinado a la configuración de la recepción. No obstante, este registro se ve involucrado de igual manera en la transmisión ya que su 7º bit (*SPEN*) es el encargado de habilitar el puerto de comunicación serial.

TXSTA: TRANSMIT STATUS AND CONTROL REGISTER (ADDRESS 98h)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

Tabla 2.21: Registro de control y estado de la transmisión. (datasheet PIC16F877)

El registro *TXSTA* es donde se configura la transmisión en sí, aunque al igual que el *RCSTA* posee ciertos parámetros de configuración conjunta tanto para la transmisión como para la recepción. Las funciones sobre las que actúan los bits son las siguientes:

- ✓ Bit 0. *TX9D*: Contiene el valor de 9º bit de transmisión en caso de ser habilitado para su uso.
- ✓ Bit 1. *TRMT*: Se trata de un registro que actúa como bandera. Notifica el estado del *TSR* (registro de transmisión) almacenando un 0 cuando está lleno y un 1 cuando esta vacio.

✓ Bit 2. *BRGH*: selector de alta/baja velocidad en *BRG*. 1=alta velocidad. 0=baja velocidad.

✓ Bit3. Sin implementar.

✓ Bit 4. *SYNC*: Determina el sincronismo de la comunicación. 0=asíncrono. 1=síncrono.

✓ Bit 5. *TXEN*: Habilitador de la transmisión. 1=habilitado. 0=deshabilitado

✓ Bit 6. *TX9*: Habilitador del 9º bit de transmisión. 1=habilitado. 0=deshabilitado

✓ Bit 7. *CSRC*: Selector del origen de la señal de reloj. No afecta a en comunicación asíncrona. En comunicación síncrona 0=modo esclavo. La señal de reloj es de origen externo. 1=modo maestro. La señal de reloj es generada internamente en el *BRG*.

PIE1 REGISTER (ADDRESS 8Ch)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

Tabla 2.22: Registro de activación de interrupciones. (datasheet PIC16F877)

El registro *PIE1* está destinado a la activación de las interrupciones de los periféricos. Las funciones sobre las que actúan sus bits son:

✓ Bit 0. *TMR1IE*: Habilitación de la interrupción por desbordamiento del *TIMER1*. 0=deshabilitada. 1=habilitada.

✓ Bit 1. *TMR2IE*: Habilitación de la interrupción por desbordamiento del *TIMER2*. 0=deshabilitada. 1=habilitada.

✓ Bit 2. *CCP1IE*: Habilitación de la interrupción por evento en *CCP1*. 0=deshabilitada. 1=habilitada.

✓ Bit3. *SSPIE*: Habilitación de la interrupción producida por el puerto serie síncrono. 0=deshabilitada. 1=habilitada.

✓ Bit 4. *TXIE*: Habilitación de la interrupción por transmisión del puerto *USART*. 0=deshabilitada. 1=habilitada.

✓ Bit 5. *RCIE*: Habilitación de la interrupción por recepción del puerto USART. 0=deshabilitado. 1=habilitado.

✓ Bit 6. *ADIE*: Habilitación de la interrupción por finalización de la conversión en el ADC. 0=deshabilitado. 1=habilitado.

✓ Bit 7. *PSPIE*: Habilitación de la interrupción por envío o recepción del puerto paralelo. 0=deshabilitado. 1=habilitado.

PIR1 REGISTER (ADDRESS 0Ch)

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

Tabla 2.23: Registro de interrupciones de periféricos. (datasheet PIC16F877)

El registro *PIR1* contiene las banderas de las interrupciones habilitadas en el registro *PIE1* y que se detallan a continuación:

✓ Bit 0. *TMR1IF*: Interrupción por desbordamiento del *TIMER1*. 0=*TIMER1* no desbordado. 1=*TIMER1* desbordado (debe ser reseteado por software).

✓ Bit 1. *TMR2IF*: Interrupción por desbordamiento del *TIMER2*. 0=*TIMER2* no desbordado. 1=*TIMER2* desbordado (debe ser reseteado por software).

✓ Bit 2. *CCP1IF*: Interrupción por evento en *CCP1*. En modo captura, si *CCP1IF*=1 indica que se ha producido en el pin *CCP1* el evento programado y se ha realizado la captura del contenido de *TMR1* al *CCPR1*. Esta bandera debe ser reseteada por software. Si *CCP1IF*=0 indica que no se ha producido el evento deseado.

✓ Bit3. *SSPIF*: Interrupción producida por el puerto serie síncrono. Debe ser reseteada por software al retorno de la rutina de servicio.

✓ Bit 4. *TXIF*: Indicador de interrupción producida por transmisión. 0= indica que el búfer de transmisión está lleno. 1= Indica que el búfer de transmisión está vacío.

✓ Bit 5. *RCIF*: Indicador de interrupción por recepción. 0= indica que el búfer de recepción está vacío. 1= Indica que el búfer de recepción está lleno.

✓ Bit 6. *ADIF*: Interrupción producida por la finalización de la conversión en el *ADC*. 0=conversión no finalizada. 1=conversión finalizada.

✓ Bit 7. *PSPIF*: Interrupción por envío o recepción en el puerto paralelo. 0=no se ha producido ninguna operación de envío/recepción. 1=Se ha producido una operación de envío/recepción.

Una vez presentados los registros que intervienen en la transmisión, estamos en condiciones de desmembrar el proceso de transmisión. Toda la actividad de esta operación funciona alrededor del registro *TSR* (*Transmit Shift Register*). En primer lugar se permite la actividad en el puerto serie mediante la activación del bit *SPEN*. Posteriormente se activa la autorización de la trasmisión mediante el bit *TXEN* y se establecen los valores del *Baude Rate Generator*. Se configura la utilización del 9º bit de transmisión. Se configuran las interrupciones por transmisión. Una vez todo está configurado, se deposita el contenido del registro a transmitir en *TXREG* quien al llenarse activa la bandera *TXIF* que produce la interrupción por transmisión. El contenido de *TXREG* pasa entonces al *TSR* reseteando al bit *TXIF* y *TRMT* que actúa como bandera de estado. El *TSR* comienza a lanzar bit a bit y cuando este se vacía el *TRMT* vuelve a situarse en 1.

c) La recepción asíncrona

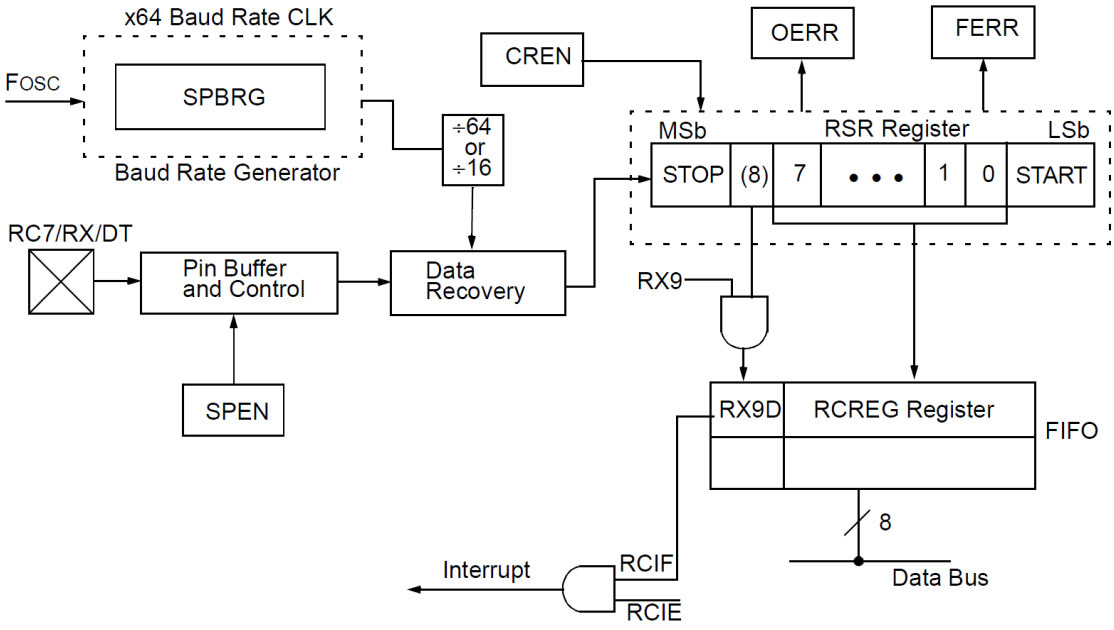


Ilustración 2.37: Diagrama de bloque de la recepción asíncrona. (datasheet PIC16F877)

Puesto que los registros que intervienen en este proceso son los mismos pasaremos directamente a describir el funcionamiento del mismo.

Lo principal es permitir la actividad del puerto serie mediante el bit *SPEN*, configurar las características del *Baud Rate Generator* y configuran las interrupciones por recepción *RCIE*. Posteriormente se selecciona si se desea recepción continua *CREN* y el uso del 9º bit de recepción. Una vez realizado esto, a la recepción de datos en el puerto RX, estos pasan al *RSR* y una vez completados los 8 bits pasan al *RCREG* quien al llenarse activa la bandera *RCIF* para que provoque la interrupción en caso de que hayan sido autorizadas.

d) El circuito de muestreo.

El circuito de muestreo actúa sobre la patilla de recepción *RC7/RX/DT* del PIC realizando un total de tres muestras para determinar mediante un circuito de mayoría si se trata de un nivel alto o bajo.

2.3.4 Programación del PIC.

A continuación se exponen los diagramas de flujo correspondientes a la programación del microcontrolador, así como su código de texto.

2.3.4.1 Diagramas de flujo.

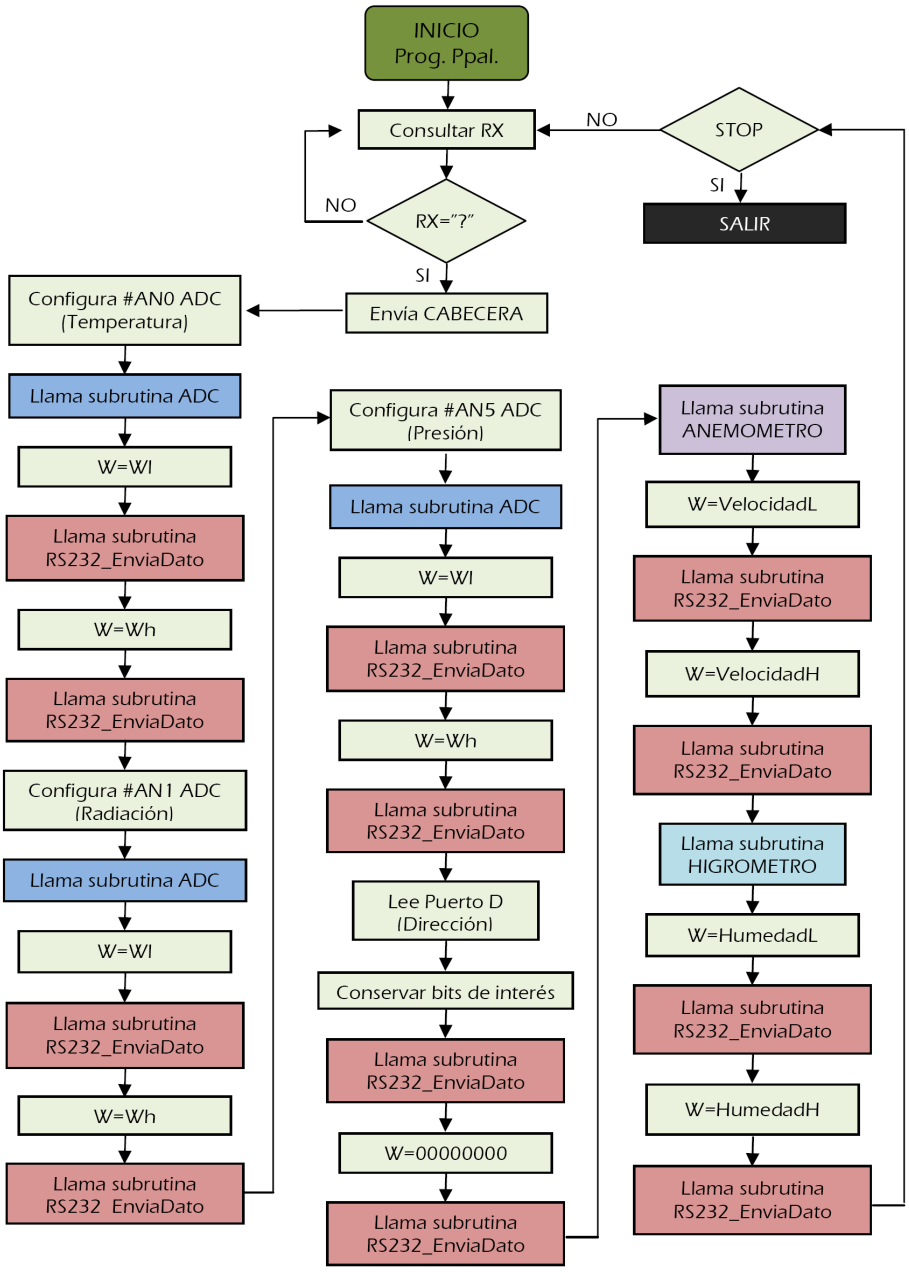


Ilustración 2.38: Diagrama de flujo del programa principal.

Los siguientes diagramas de flujo de las subrutinas en orden de aparición son:

La subrutina *ADC* encargada de gestionar el *Convertidor Analógico Digital* para obtener los resultados de la conversión:

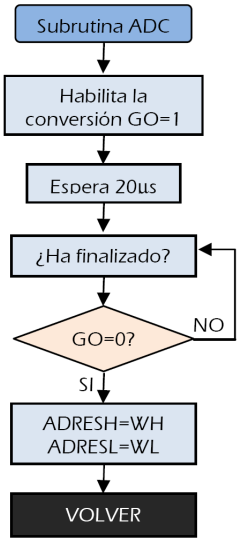


Ilustración 2.39: Diagrama de flujo de la subrutina ADC

La subrutina RS232_EnviaDato, encargada de enviar el contenido del registro *W* al *TXREG* para su transmisión.

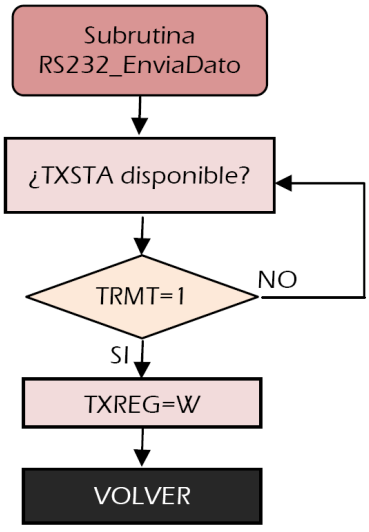


Ilustración 2.40: Diagrama de flujo de la subrutina RS232_EnviaDato

La subrutina Anemómetro, que gestiona la adquisición del periodo de tiempo transcurrido entre pulsos adquiridos por el pin *CCP1* como resultado del circuito acondicionador para la velocidad del viento.

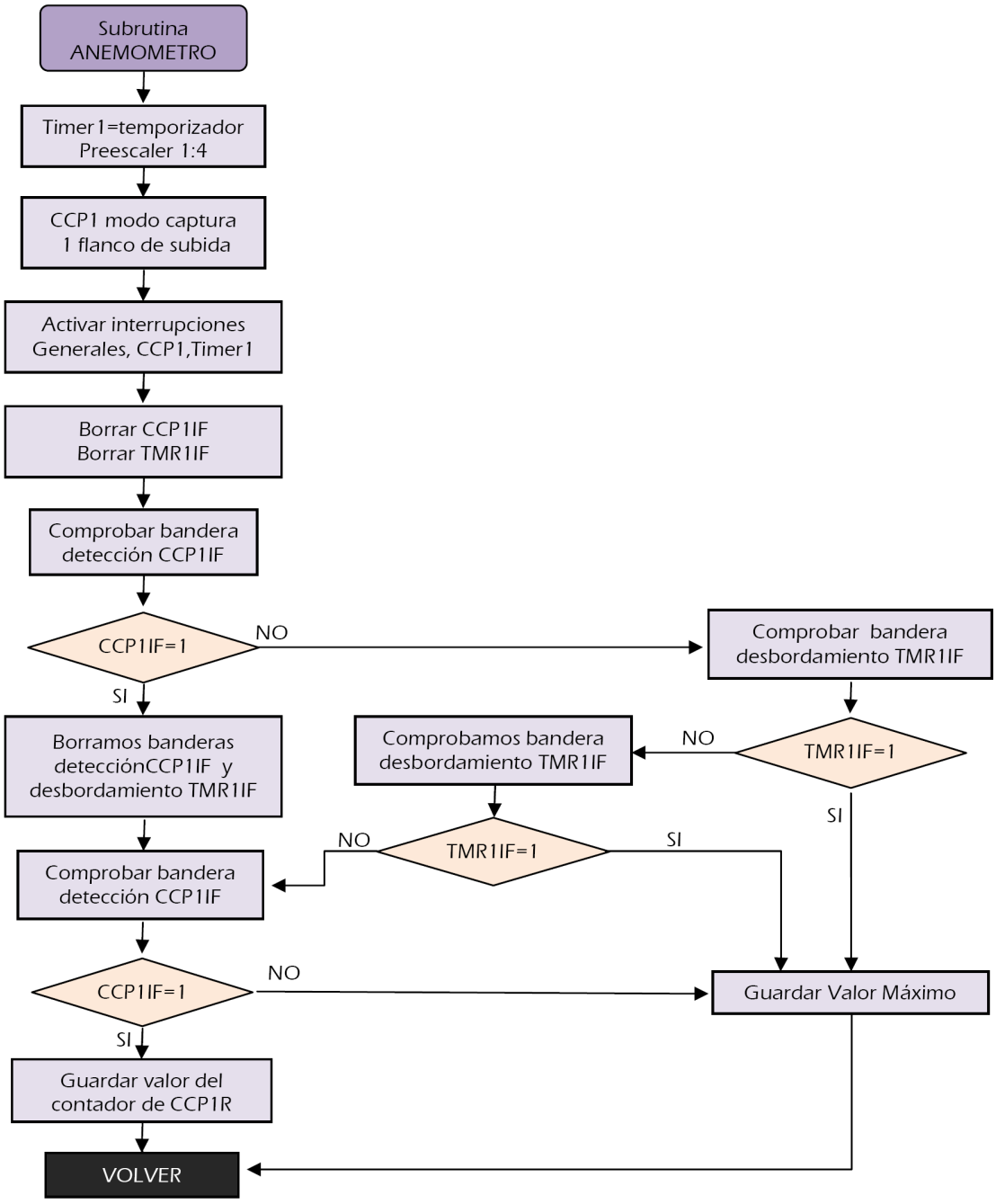


Ilustración 2.41: Diagrama de flujo de la subrutina Anemómetro.

La subrutina Higrómetro, que al igual que la subrutina Anemómetro gestiona la adquisición del periodo, en este caso, de una onda rectangular, resultado del circuito acondicionador para la medida de humedad, constituido por un oscilador astable mediante 555.

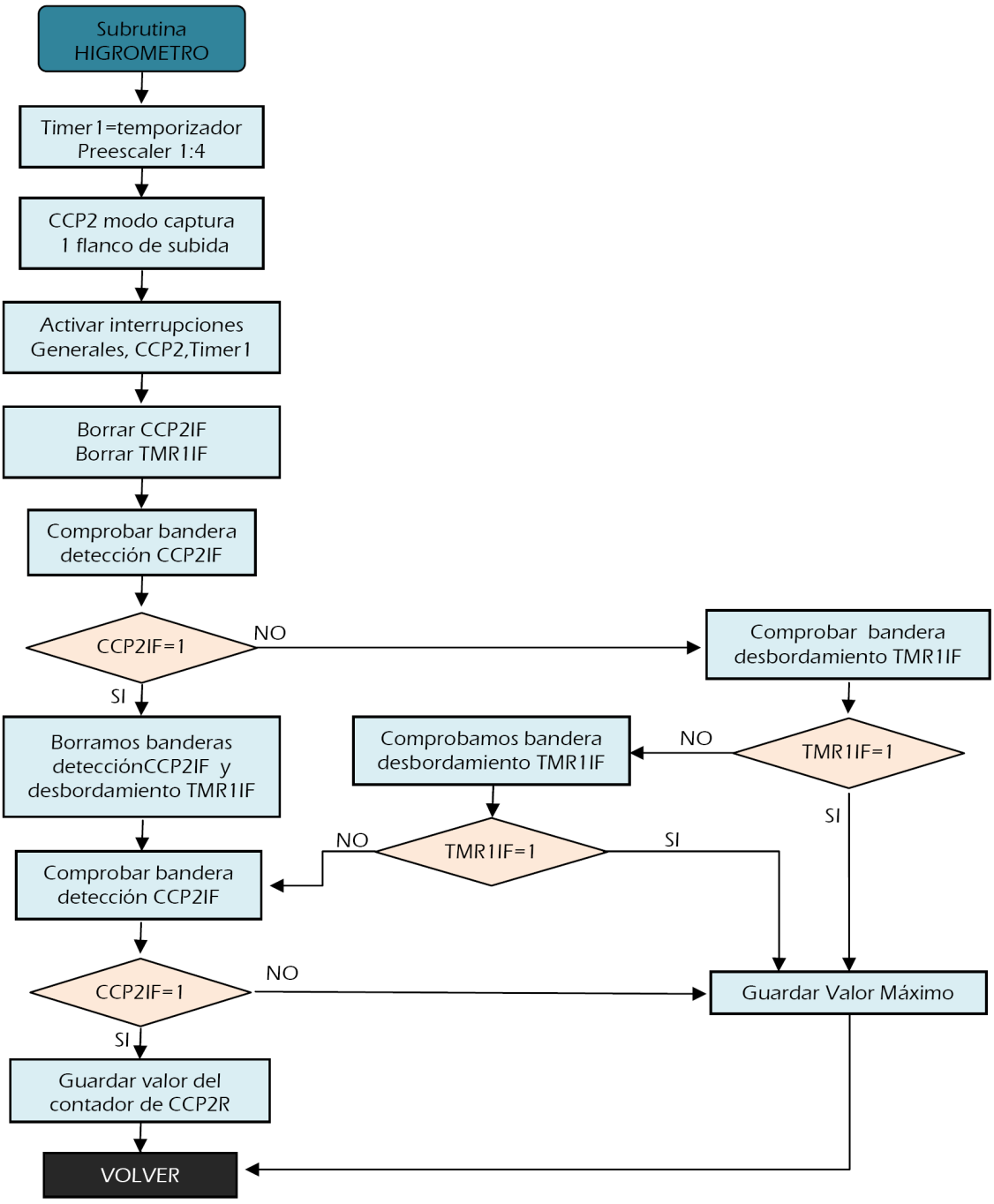


Ilustración 2.42: Diagrama de flujo de la subrutina Higrómetro.

2.3.4.2 Programa en ENSAMBLADOR.

```
;****  _____CÓDIGO DEFINITIVO PFC ESTACIÓN METEOROLÓGICA_____****
;*****************************************************************
;**Código DEFINITIVO para el PFC Estación Meteorológica.          **
;** Diseñado y probado por Alejandro Domínguez Hiniesta.           **
;*****************************************************************


        processor       16f877
        include         <p16f877.inc>
        __CONFIG        0x3F71

;**** Definición de variables ****
contador20us EQU 0x20
Wl      EQU 0X21
Wh      EQU 0X22
VelocidadL EQU 0x23
VelocidadH EQU 0x24
HumedadL EQU 0x25
HumedadH EQU 0x26


;**** Definiciones para el ensamblador ****

;**** Definición de macros ****
; Se envía el dato del registro W al PC mediante rs232.-
Putreg macro  Registro
        movfw Registro
        call            RS232_EnviaDato
        endm
;**** Fin definiciones de macros ****

;////////////////////////////////
```

```
;**** Inicio del Micro ****
Reset.
        org     0x00    ; Aquí comienza el micro.-
        goto    inicio
        org     0x04
        goto    INTER
inicio
;****CONFIGURACION DEL CAD*****
        movlw b'01000001'   ; A/D conversion a Fosc/8
        movwf ADCON0

;****DEFINICION DE PUERTOS E/S ANALOGICAS/DIGITALES****
        bsf     STATUS,RP0  ;  Acceso de memoria al banco 1
        bcf     STATUS,RP1
        movlw b'10001000'   ; A/D port AN0,AN1,AN4,AN5, VREF-, VREF+
        movwf ADCON1
        clrf    TRISB       ; PORTB como salida
        movlw b'10111111'
        movwf TRISC         ; RC7/RX entrada, RC6/TX salida, RC2/CCP1 entrada,
                            ;RC1/CCP2 entrada
        movlw b'11111111'   ; portA como entrada
        movwf TRISA
        movlw b'00001111'   ; RD0, RD1,RD2,RD3 como entradas
        movwf TRISD


;****CONFIGURACION DEL USART
        movlw   b'00100100' ; Configuración USART  TXEN=1, BRGH=1
        movwf   TXSTA       ; y activación de transmisión
        movlw   .25         ; 9600 baudios
        movwf   SPBRG
        bsf     PIE1,RCIE   ; Habilita interrupción en recepción
        bcf     STATUS,RP0  ; Acceso de memoria al banco 0
        movlw   b'10010000' ; Configuración del USART para recepción continua
        movwf   RCSTA       ; Puesta en ON
```

```
; ****COMIENZO DEL PROGRAMA PRINCIPAL.
        clrf    PORTA                   ; Limpiar salidas
        clrf    PORTB
        clrf    PORTD
        clrf    PORTE
        clrf      Wl
        clrf      Wh
inicio2
        ;Envío de carácter de inicio.
        movlw .255
        call RS232_EnviaDato
        call RS232_EnviaDato
        ;Toma de datos de temperatura.
        movlw b'01000001'
        movwf ADCON0                    ; selección del canal 0 (Temperatura Amb.)
        call adc
        movfw Wl
        call      RS232_EnviaDato
        movfw Wh
        call RS232_EnviaDato
        ;Toma de datos de radiación.
        movlw b'01001001'
        movwf ADCON0                    ; selección del canal 1 (Radiación Solar)
        call adc
        movfw Wl
        call      RS232_EnviaDato
        movfw Wh
        call RS232_EnviaDato
        ;Toma de datos de presión.
        movlw b'01101001'
        movwf ADCON0                    ;selección del canal 5 (Presión Barométrica)
        call adc
        movfw Wl
```

```
        call    RS232_EnviaDato
        movfw Wh
        call RS232_EnviaDato
        ;Toma de datos de dirección de viento.
        movfw PORTD             ;captura del puerto D (Veleta digital)
        andlw  b'00001111'      ;eliminamos la información de las líneas RD4 a RD7
        call    RS232_EnviaDato
        movlw b'00000000'
        call RS232_EnviaDato
        ;Toma de datos de Velocidad del viento.
        call anemometro
        movfw VelocidadL
        call    RS232_EnviaDato
        movfw VelocidadH
        call    RS232_EnviaDato
        ;Toma de datos de la Humedad Relativa.
        Call    higrometro
        movfw HumedadL
        call    RS232_EnviaDato
        movfw HumedadH
        call RS232_EnviaDato

        goto INTER


;*****************************************************************
;**_____________________________ SUBRUTINAS_____________________________**
;*****************************************************************
;**** Demora ****
demora_20us
                movlw 0x05 ;
                movwf contador20us      ; Iniciamos contador1.-
repeticion
                decfsz  contador20us,1  ; Decrementa Contador1.-
                goto    repeticion      ; Si no es cero repetimos ciclo.-
```

```
                return                  ; Regresa de la subrutina.-

;**** Se envía Dato ****
RS232_EnviaDato
        bsf     STATUS,RP0      ; Banco 1
        btfss   TXSTA,TRMT      ; chequea si está listo
        goto    $-1             ; Esperamos  a que se desocupe.-
        bcf     STATUS,RP0      ; Banco 0
        movwf TXREG             ; envía Dato.-
        return
;****Conversión ADC****
adc
        bsf     ADCON0,GO       ; habilita la conversión.
        call    demora_20us
_espera
        btfsc   ADCON0,GO       ; ¿ha finalizado la conversión?
        goto    _espera         ; si no ha finalizado vuelve a _adc
        bcf     STATUS,RP0
        bcf     STATUS,RP1
        movfw ADRESH            ; si ha finalizado guarda el valor de ADRESH en W
        movwf Wh
        bsf     STATUS, RP0
        movfw ADRESL
        bcf     STATUS, RP0
        movwf Wl
        return

;****Anemómetro****
anemometro
        movlw b'00100001'       ; Se selecciona TMR1, preescaler de 1/4, modo
                                ;temporizador.-
        movwf T1CON
        movlw b'00000101'       ; Se configura CCP modo captura cada flanco de
                                ;subida.-
        movwf CCP1CON
```

```
        bsf     INTCON, GIE         ; Activación de las interrupciones generales.
        bsf     STATUS,RP0
        bsf     PIE1, CCP1IE        ; Activación de la interrupción por CCP1
        bsf     PIE1, TMR1IE        ; Interrupción por desbordamiento del TIMER1
        bcf     STATUS,RP0

        bcf     PORTC,4
        bcf     PIR1,CCP1IF         ; Borramos bandera
        bcf     PIR1,TMR1IF         ; Borramos bandera de desbordamiento timer1
pto1
        btfss   PIR1,CCP1IF         ; Testeamos bandera.-
        goto    desborde1           ; No se activo. Comprobamos desbordamiento
        bcf     PIR1,CCP1IF         ; Se activo, la borramos.-
        clrf    TMR1H               ; Borramos Timer1.-
        clrf    TMR1L ;

pto2
        btfss   PIR1,CCP1IF         ; Volvemos a testear bandera.-
        goto    desborde2           ; No se activo, comprobamos desbordamiento
        movfw CCPR1L                ; Copiamos el valor capturado.-
        movwf VelocidadL
        movfw CCPR1H
        movwf VelocidadH
        goto    marcadoAne

desborde1
        btfss   PIR1,TMR1IF         ; Testeamos bandera desbordamiento TRM1.-
        goto    pto1
        movlw .254
        movwf VelocidadL
        movlw .255
        movwf VelocidadH
        bsf     PORTC,4
        goto    marcadoAne
```

```
desborde2
        btfss   PIR1,TMR1IF         ; Testeamos bandera desbordamiento TMR1.-
        goto    pto2
        movlw .254
        movwf VelocidadL
        movlw .255
        movwf VelocidadH
        bsf     PORTC,4
        goto    marcadoAne

marcadoAne
        return

;****Higrómetro****
higrometro
        movlw b'00100001'       ; Se selecciona TMR1, preescaler de 1/4, modo
                                ;temporizador.-
        movwf T1CON
        movlw b'00000101'       ; Se configura CCP modo captura cada flanco de
                                ;subida.-
        movwf CCP2CON
        bsf     INTCON, GIE     ; Activación de las interrupciones generales.
        bsf     STATUS,RP0
        bsf     PIE1, CCP2IE    ; Activación de la interrupción por CCP2
        bsf     PIE1, TMR1IE    ; Interrupción por desbordamiento del TIMER1
        bcf     STATUS,RP0

        bcf     PIR2,CCP2IF     ; Borramos bandera
        bcf     PIR1,TMR1IF     ; Borramos bandera de desbordamiento timer1
pto3
        btfss   PIR2,CCP2IF     ; Testeamos bandera.-
        goto    desborde3       ; No se activo. Comprobamos desbordamiento
        bcf     PIR2,CCP2IF     ; Se activo, la borramos.-
        clrf    TMR1H            ; Borramos Timer1.-
        clrf    TMR1L ;
```

```
pto4
        btfss   PIR2,CCP2IF         ; Volvemos a testear bandera.-
        goto    desborde4           ; No se activo, comprobamos desbordamiento
        movfw CCPR2L                ; Copiamos el valor capturado.-
        movwf HumedadL
        movfw CCPR2H
        movwf HumedadH
        goto    marcadoHigr

desborde3
        btfss   PIR1,TMR1IF         ; Testeamos bandera desbordamiento TRM1.-
        goto    pto3
        movlw .254
        movwf HumedadL
        movlw .255
        movwf HumedadH
        goto    marcadoHigr

desborde4
        btfss   PIR1,TMR1IF         ; Testeamos bandera desbordamiento TMR1.-
        goto    pto4
        movlw .254
        movwf HumedadL
        movlw .255
        movwf HumedadH
        goto    marcadoHigr

marcadoHigr
        return

;****Tratamiento de interrupción****
INTER   btfss   PIR1,RCIF       ; ¿Interrupción por recepción?
        goto    VOLVER          ; No, falsa interrupción
        bcf     PIR1,RCIF       ; Si, reponer flag
```

```
        movf   RCREG,W      ; Lectura del dato recibido
        sublw  a'?'
        btfss  STATUS,Z
        goto   INTER
        goto   inicio2
VOLVER  retfie

        END
```

2.4 Interfaz de usuario.

El interfaz de usuario pretende mostrar al usuario de una forma fácil y agradable los datos adquiridos a través de la tarjeta de adquisición, al mismo tiempo que realiza un registro en archivos independientes que permita posteriores consultas sobre los datos adquiridos.

A continuación se aborda la explicación de la interfaz de usuario, mediante la explicación de la funcionalidad del panel frontal y la programación del panel de diagrama de bloques de la interfaz.

2.4.1 El panel Frontal.

El panel frontal es la interfaz, propiamente dicha, con el operador. Representa el panel frontal de cualquier instrumento de medida, en el que pueden observarse o introducirse valores, o bien interactuar con pilotos de aviso y controles como ruletas o interruptores sin que deba tenerse en cuenta la programación interna del instrumento.

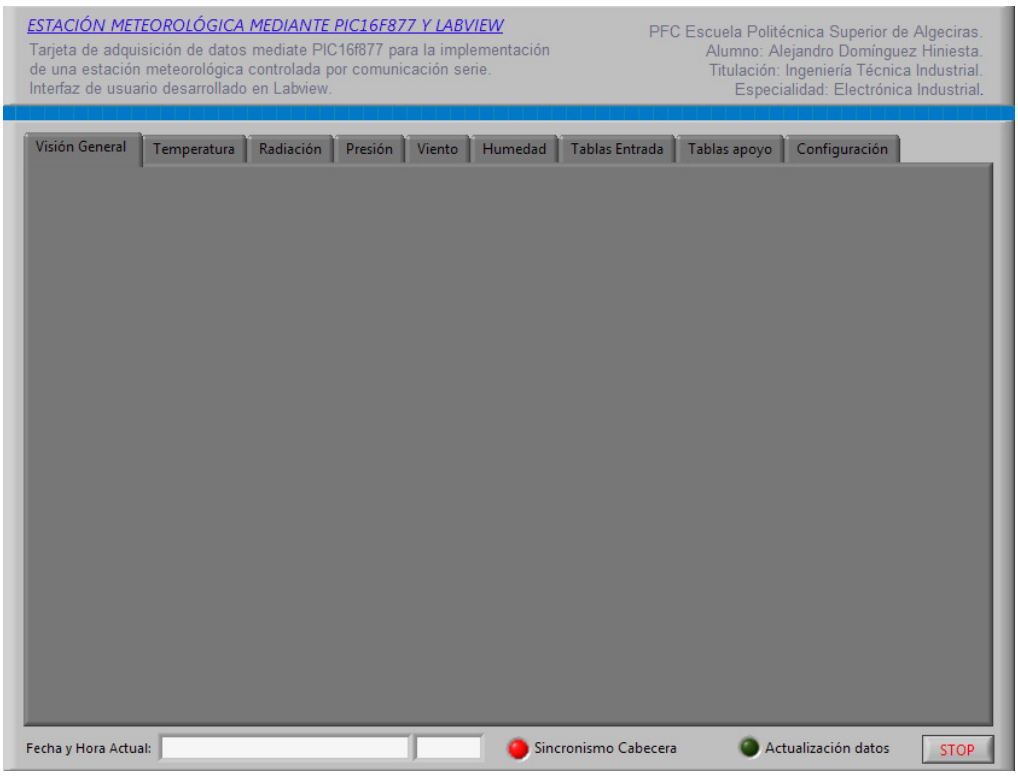


Ilustración 2.43: Captura del panel frontal de la interfaz de usuario.

En el presente proyecto, se pretende ofrecer una interfaz sencilla que muestre los valores capturados por nuestra tarjeta agrupados por magnitud. Para ello se ha diseñado una presentación por pestañas cuyos títulos corresponden a las magnitudes a medir. A demás, se han incluido unas pestañas adicionales que permiten controlar y comprender el correcto funcionamiento, así como establecer valores de configuración de la interfaz.

Al pie de la presentación, se muestran un conjunto de elementos de interés para todas las pestañas, como son la fecha y hora del sistema, así como dos pilotos indicadores, uno denominado *SINCRONISMO CABECERA* que se ilumina en verde mientras existe sincronismo con el envío de datos o en rojo cuando lo pierde y otro denominado *ACTUALIZACIÓN DE DATOS* que se ilumina en verde cuando se produce la entrega de la cola de datos generada en el bucle productor al bucle consumidor. Como en todo dispositivo, se ha instalado un pulsador de *STOP* que permite la detención del sistema.

A continuación se detallan cada una de las pestañas.

2.4.1.1 Visión General.

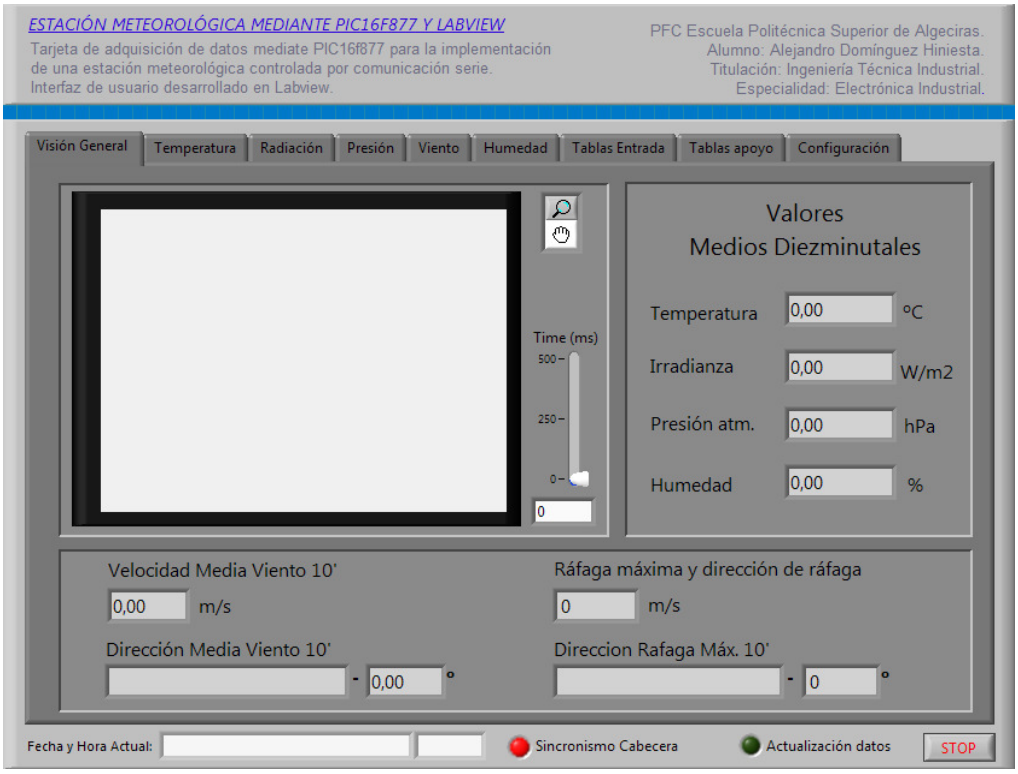


Ilustración 2.44: Captura de la pestaña Visión General del panel frontal.

En la pestaña de Visión General se ha tratado de representar todo aquello que querríamos saber de un vistazo rápido, como son los valores promediados de cada magnitud. A demás, se ha incorporado un campo en el que se representa la captura de una cámara que permite controlar la presencia y funcionamiento de los aparatos de viento. Este proyecto no contempla el almacenamiento de imágenes, tan solo el visionado en tiempo real del estado de los aparatos de viento.

Dentro del campo donde se representa la captura de la cámara, se ha dispuesto un control en forma de deslizador que introduce un tiempo de espera entre las capturas de la cámara. Esto repercute sobre el microcontrolador del PC liberando recursos, facilitando a los equipos más limitados la ejecución de las tareas.

2.4.1.2 Temperatura

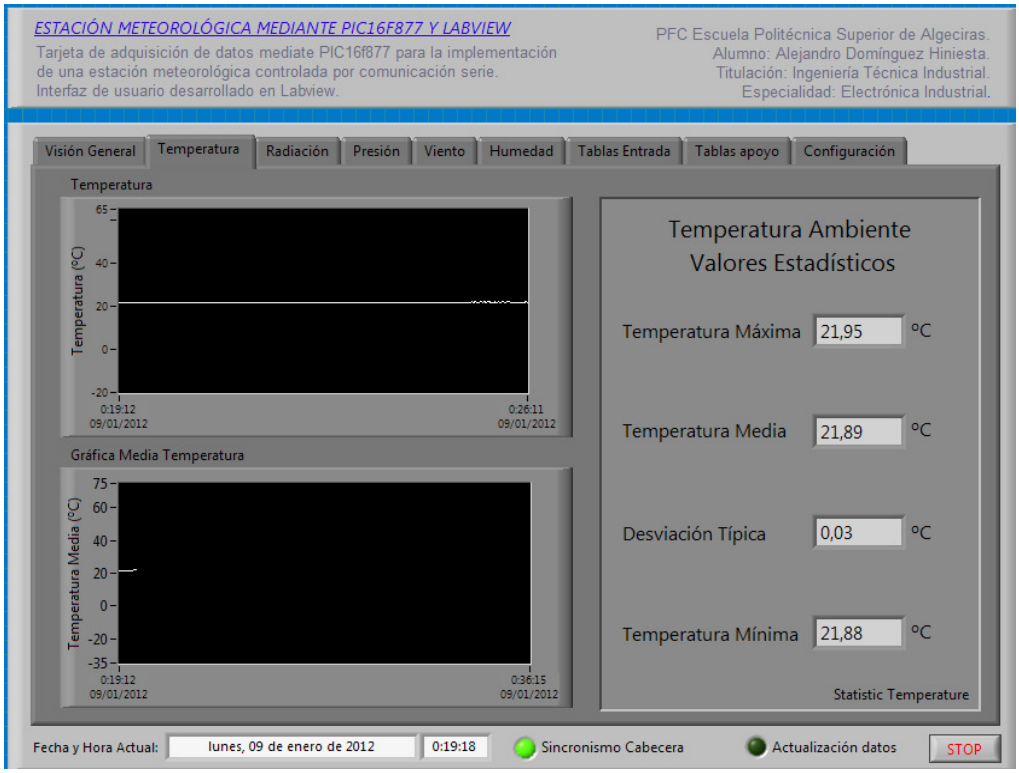


Ilustración 2.45: Captura de la pestaña Temperatura del panel frontal.

La pestaña de temperatura presenta al operador los valores estadísticos extraídos de la muestra diezminutal, tales como valor máximo, mínimo, desviación típica y valor medio. Así mismo, se presentan dos gráficas referentes a la temperatura. La primera situada más arriba, mostrará el conjunto de los valores capturados, la segunda, situada abajo, presentará el valor medio de la muestra.

2.4.1.3 Radiación

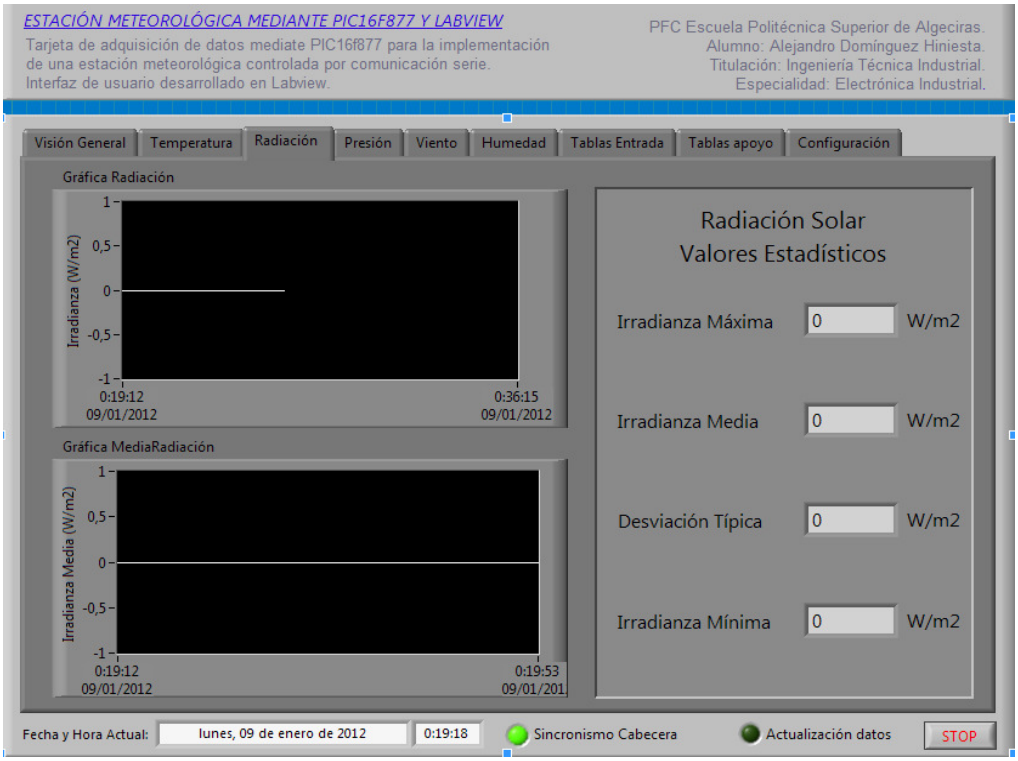


Ilustración 2.46: Captura de la pestaña Radiación del panel frontal.

Dentro de los objetivos del diseño de la interfaz, se encontraba el de mantener la uniformidad del aspecto de las diferentes pestañas. La pestaña Radiación mantiene la misma estructura que la pestaña de Temperatura, así como para todas las pestañas de magnitudes a excepción del viento, motivo por el cual no es necesario entrar en mayores detalles.

2.4.1.4 Presión

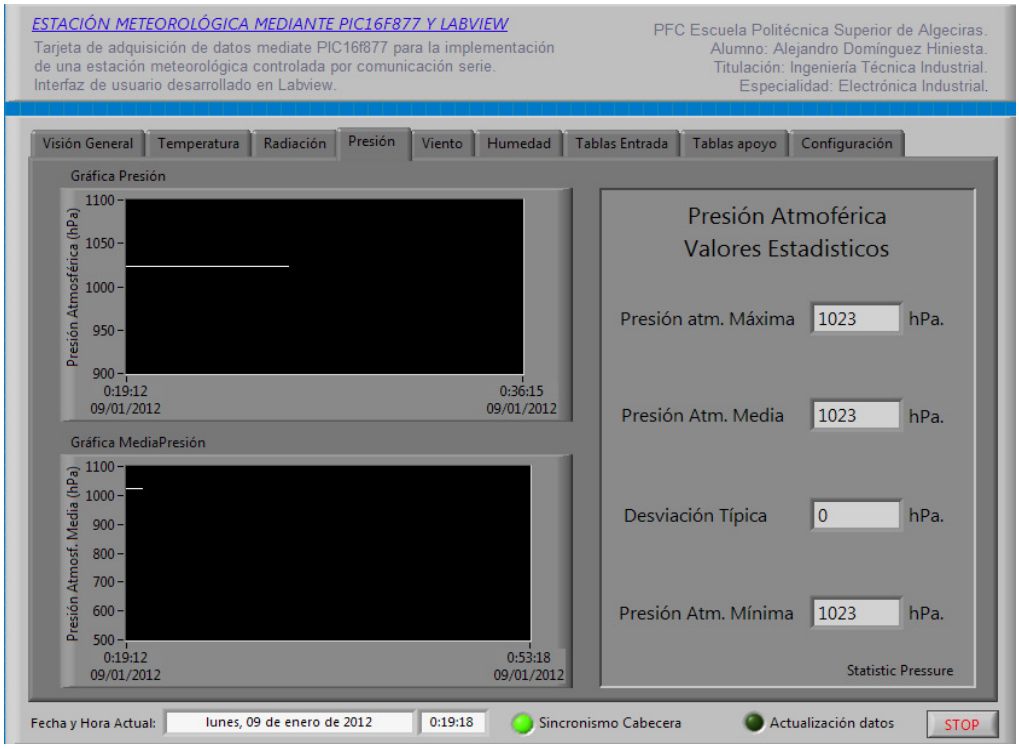


Ilustración 2.47: Captura de la pestaña Presión del panel frontal.

La pestaña Presión como puede verse comparando la Ilustración 2.55 y la Ilustración 2.56 también cumple con el objetivo de uniformidad de diseño.

2.4.1.5 Viento

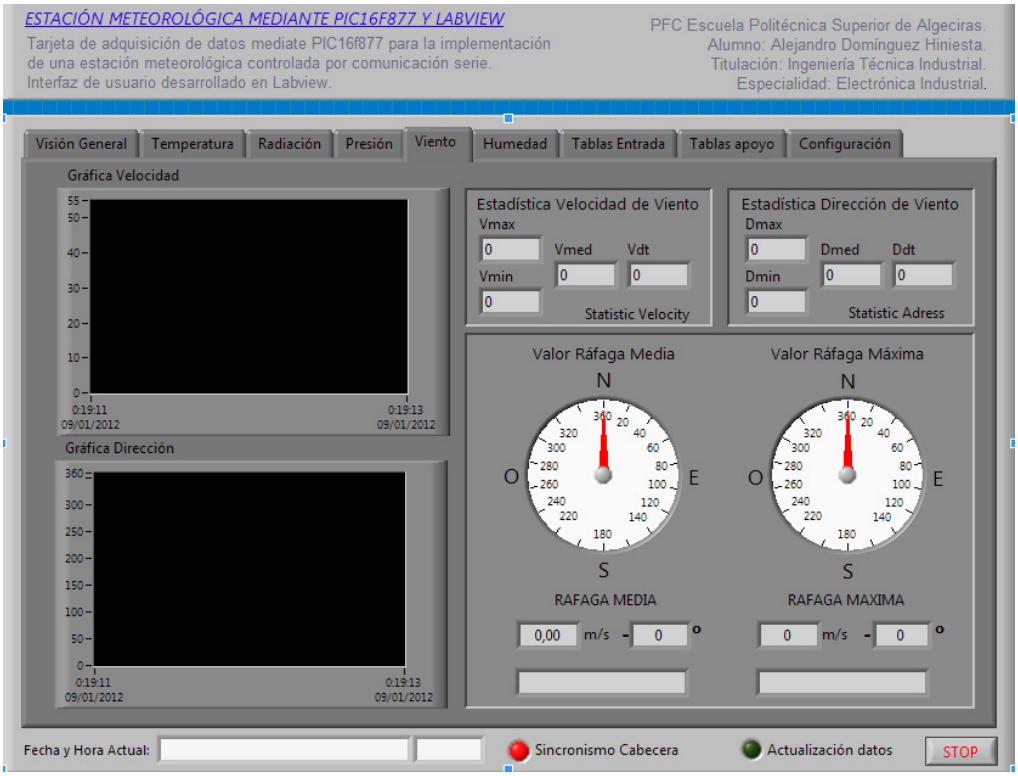


Ilustración 2.48: Captura de la pestaña Viento del panel frontal.

A diferencia del resto de pestañas de magnitud, la pestaña Viento presenta un diseño diferente. Esto está motivado por la cantidad de medidas que es preciso reflejar para poder observar las características del viento. Como puede verse en la Ilustración 2.48, no solo se reflejan los valores estadísticos considerados en el resto de magnitudes, sino que dentro de la misma pestaña se relacionan las magnitudes Velocidad y Dirección del viento.

Para terminar de completar esta pestaña Viento, a demás, se ha incluido la presencia de una tercera medida derivada de estas dos anteriores denominada Ráfaga Máxima. Esta medida identifica el pico máximo de velocidad de viento dentro de una muestra diezminutal y la dirección de procedencia del mismo. En consecuencia, con objeto de mantener la importancia de ambas medidas y poder compararlas, se han generado dos indicadores gráficos de apariencia de reloj que muestran las direcciones de origen de las ráfagas media y máxima así como sus direcciones expresadas en grados y en texto según los nombres de los puntos cardinales, junto con las intensidades de las mismas.

2.4.1.6 Humedad

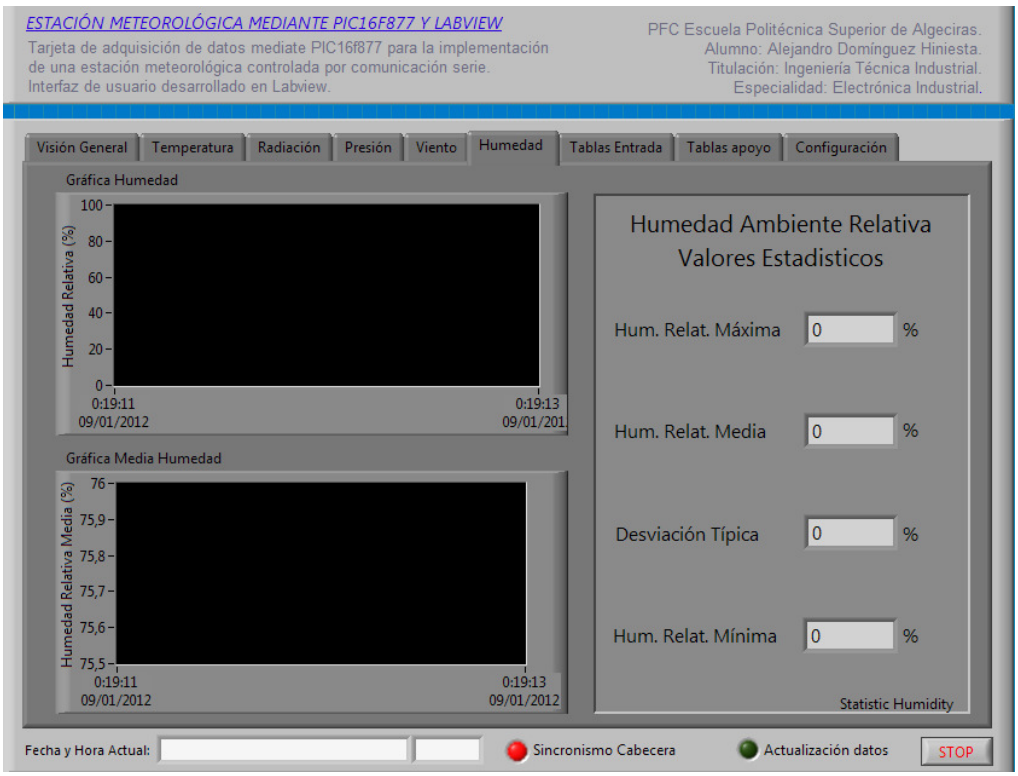


Ilustración 2.49: Captura de la pestaña Humedad del panel frontal.

La pestaña Humedad también cumple con el objetivo de uniformidad de diseño, siendo este semejante a los diseños establecidos para las pestañas Temperatura, Radiación y Presión.

2.4.1.7 Tablas de entrada

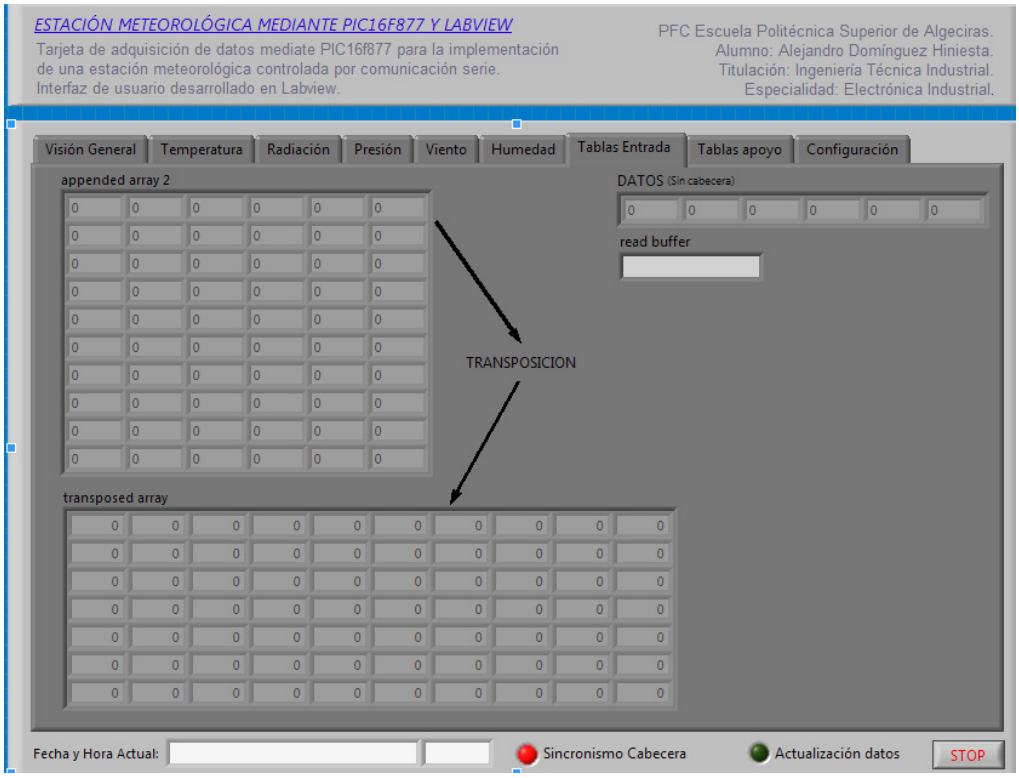


Ilustración 2.50: Captura de la pestaña Tablas Entrada del panel frontal.

La pestaña tablas de entrada constituye un apoyo en la comprobación del correcto funcionamiento del código de la interfaz y aunque no es necesario para la explotación del presente proyecto, se ha mantenido dada la gran utilidad ofrecida por esta herramienta en el desarrollo de la interfaz. En ella se representan dos fragmentos de los arrays constituidos por las medidas del PIC. Ambas tablas representan al mismo array en diferentes momentos del código. La tabla situada en la zona superior izquierda hace referencia al array *APPEND ARRAY 2* del *BUCLE PRODUCTOR* y en él puede comprobarse que la comunicación con el PIC es correcta y que los datos van accediendo a la *COLA DE DATOS*.

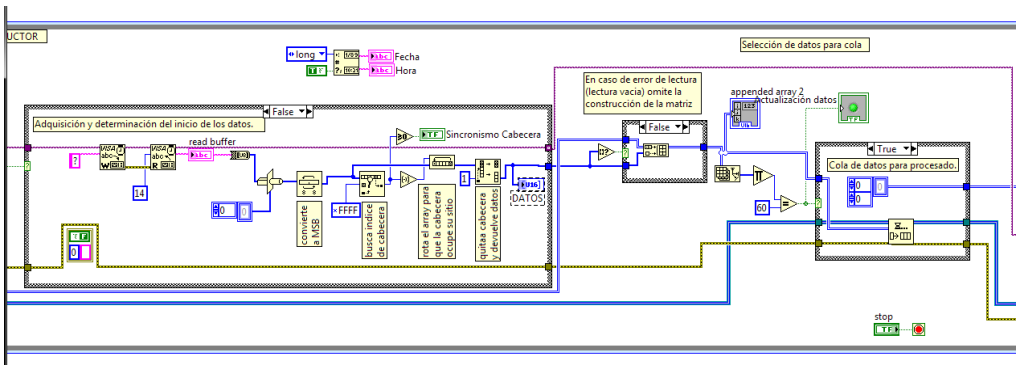


Ilustración 2.51: Bucle productor. Captura del diagrama de bloques.

La tabla situada abajo a la izquierda es la traspuesta del mismo array ya dentro del bucle *CONSUMIDOR*, y en él puede comprobarse que la *COLA DE DATOS* ha sido entregada al *CONSUMIDOR* para su tratamiento.

En la parte superior derecha encontramos un vector denominado *DATOS (SIN CABECERA)* que muestran el contenido de cada vector que se incluye en el array *APPEND ARRAY 2* una vez que ha sido despojado de la cabecera de control que determina el inicio de envió de datos. Y justo debajo de él se encuentra el *READ BUFFER* que es el indicador que muestra la lectura en bruto (formato string) que se hace del puerto serie.

2.4.1.8 Tablas de apoyo

La pestaña tablas de apoyo, al igual que la de tablas entrada, se ha mantenido dada su utilidad para desarrollo de la interfaz. En ella se representan dos fragmentos de los vectores de *VELOCIDAD Y DIRECCIÓN DEL VIENTO* empleados para determinar el proceso de cálculo de la *RÁFAGA MÁXIMA* y un fragmento del *ARRAY FINAL* cuyo contenido son las medidas realzadas, expresadas en sus unidades finales de interés.

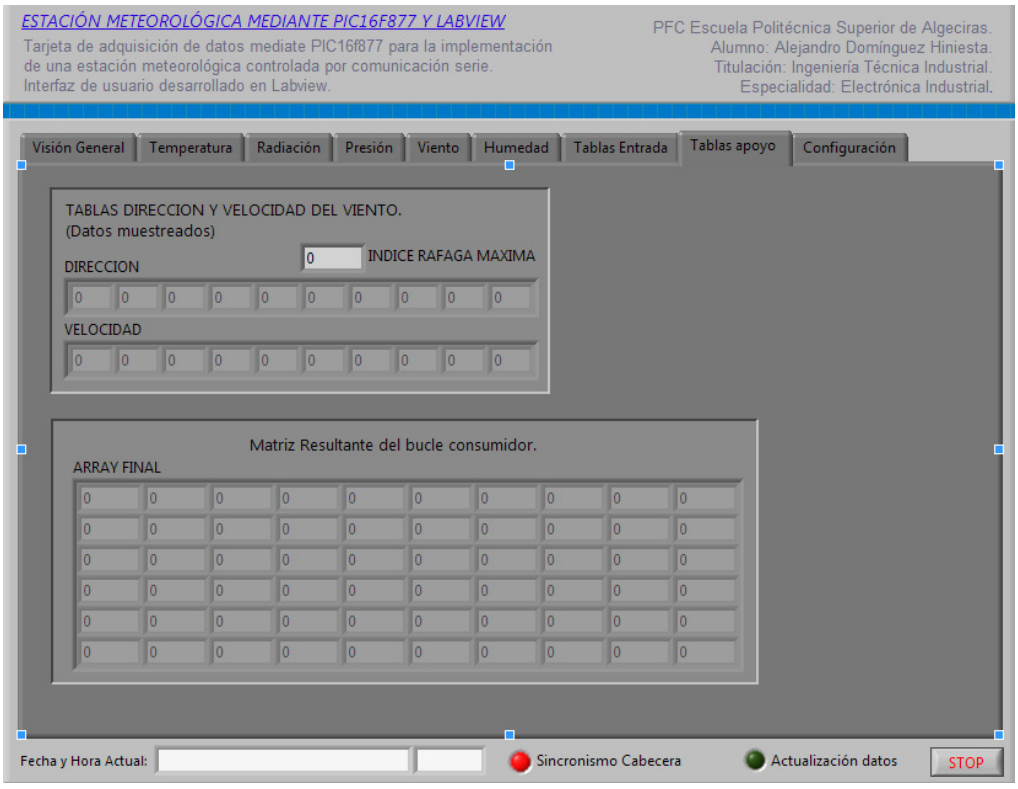


Ilustración 2.52: Captura de la pestaña Tablas Apoyo del panel frontal.

2.4.1.9 Configuración

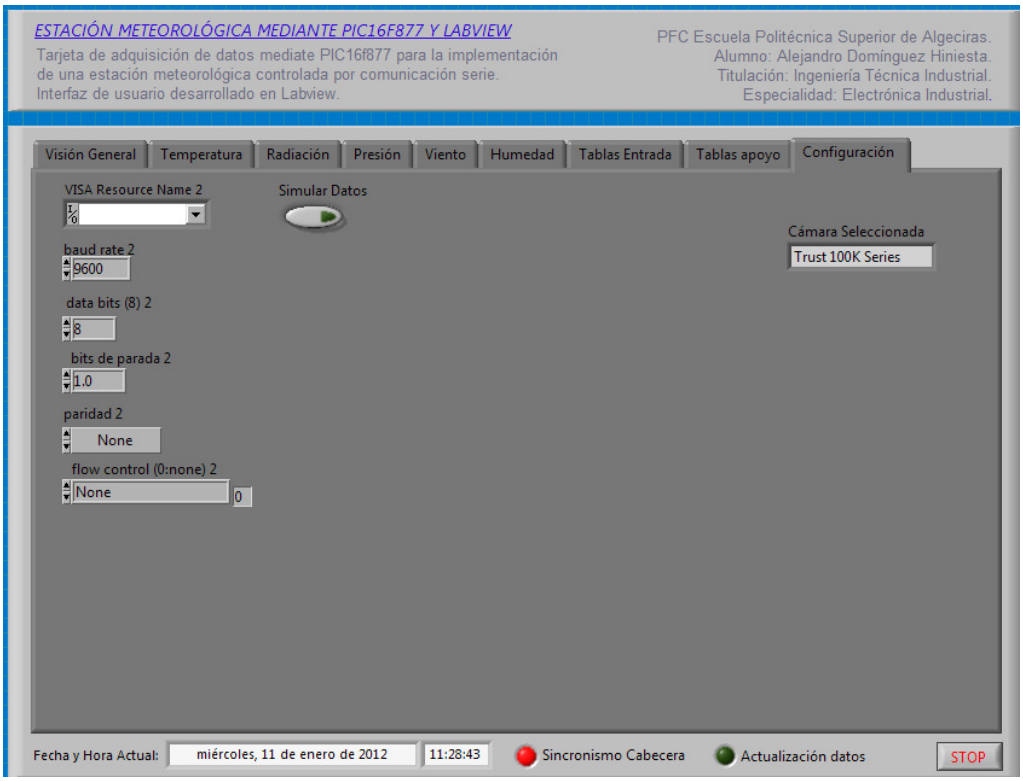


Ilustración 2.53: Captura de la pestaña Configuración del panel frontal.

La pestaña Configuración es la última de las pestañas, pero no por ello la menos importante. En ella se establecen los parámetros necesarios para establecer la comunicación con el puerto serie así como la opción de ordenar que se simulen los datos de entradas.

En un principio, esta pestaña contenía parámetros de configuración como la selección de los archivos de destino de los datos, la selección del numero de muecas del eje rotacional del anemómetro, la selección de la distancia de las cazoletas del anemómetro al eje e incluso la selección de la duración del tiempo de muestreo del sistema, pero finalmente se optó por que todas estas variables fuesen internas ya que se consideró que no eran susceptibles de sufrir cambios.

Se ha mantenido un campo que informa de la cámara seleccionada para la adquisición de datos.

2.4.2 El Diagrama de Bloques.

El diagrama de bloques en LabView representa la programación del conjunto funcional de la interfaz de usuario. En esta pantalla se establece la relación existente entre datos, constantes, indicadores, controles, funciones, estructuras y cada uno de los elementos que intervienen en la programación de la interfaz. Supone la parte no visible para el operador donde subyace la programación gráfica que determina qué ha de mostrarse en el panel frontal y como.

Se intentará dar una explicación global del código analizándolo por fragmentos funcionales.

2.4.2.1 Estructura productor/consumidor.

Este software está basado en una estructura de bucles Productor/Consumidor.

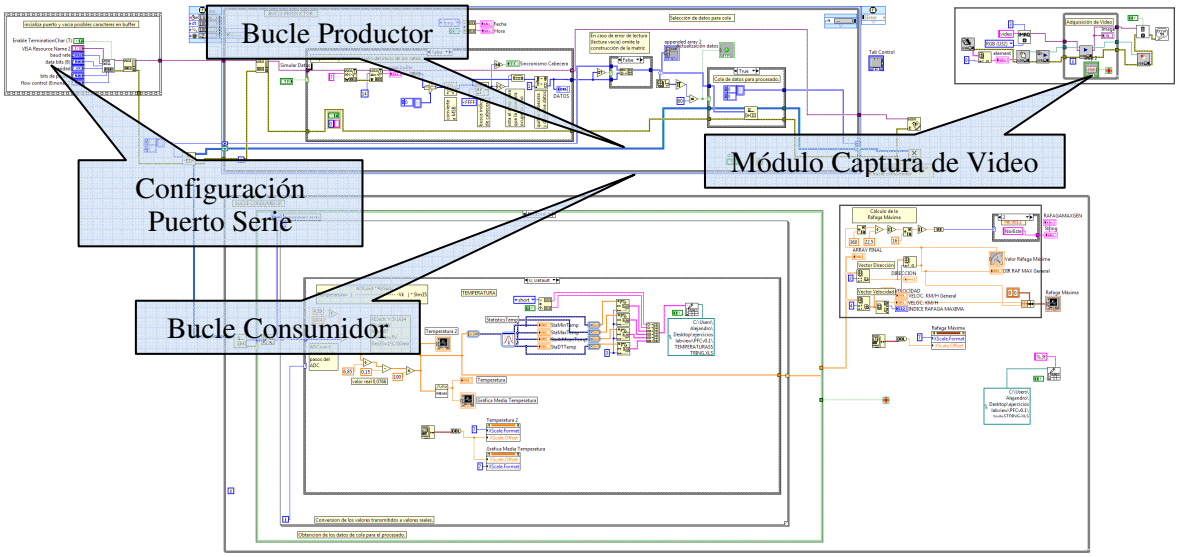


Ilustración 2.54: Captura del diagrama de bloques. Estructura del código.

Como su propio nombre indica, se dividen las actividades de producción de datos y consumición de los mismos, en dos (o más) bucles independientes conectados entre sí tan solo por la transferencia de datos. Esto permite poder adquirir datos al ritmo de la producción sin verse sometido al ritmo de tratamiento del bucle consumidor sin que este pueda hacerle perder el sincronismo con el proceso de adquisición.

Para empezar, nuestro VI recibe los datos a través del puerto serie, por tanto es preciso inicializar el puerto para poder utilizarlo.

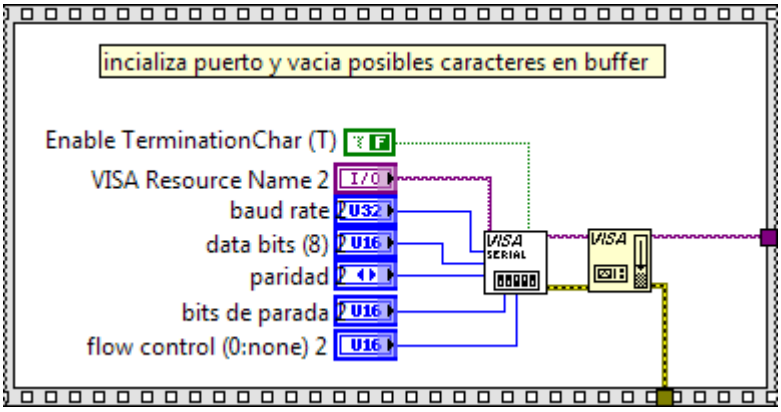


Ilustración 2.55: Inicialización del puerto serie. Captura del diagrama de bloques.

Mediante la función *VISA CONFIGURE SERIAL PORT*, perteneciente a las funciones *SERIAL PROTOCOL* de la paleta de funciones *CONECTIVITY*, definimos los parámetros bajo los cueles se establecerá la comunicación serie. La función *VISA FLUSH I/O BUFFER*, limpia el buffer de posibles valores almacenados en comunicaciones anteriores. A la salida de esta estructura *STAQUED SEQUENE* (secuencia de apilado) obtenemos la lectura de datos del puerto (línea morada) y la línea de errores (línea verdosa). Ambas líneas constituirán el medio de alimentación del bucle productor, como se muestra en la Ilustración 2.56.

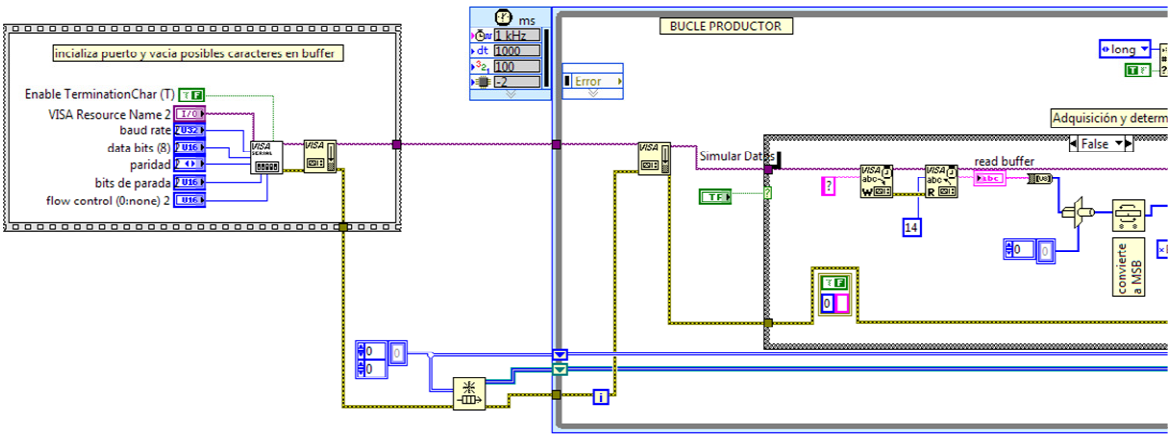


Ilustración 2.56: Enlace entre la secuencia de inicialización del puerto serie y el bucle productor.

Antes de entrar en el bucle productor (en la parte de abajo a la izquierda del mismo) vemos una función *OBTAIN QUEUE* (obtención de cola) que inicializa la *cola de datos* que posteriormente a la salida del bucle productor entregaremos al bucle consumidor.

2.4.2.2 El bucle productor.

El bucle productor, está constituido por una estructura *TIMED LOOP* (bucle temporizado) su funcionamiento es similar al de una estructura while, con la salvedad de que se ejecuta conforme a la frecuencia determinada por el programador. Esto permite mantener diferentes temporizaciones en el VI, sin que éstas se vean afectadas por los tiempos de ejecución de las demás. En nuestro caso, debemos muestrear los sensores cada segundo, es decir, ejecutaremos el bucle productor cada 1000ms.

Una vez dentro del bucle productor, encontramos una estructura *CASE* cuya función es ejecutar el código para adquirir los datos desde el puerto serie o bien, el código para realizar una simulación de posibles datos (por ejemplo para el caso de pruebas sin conexión a puerto serie) en función del valor del pulsador *SIMULAR DATOS* seleccionado por el operador en la pestaña de configuración del panel frontal.

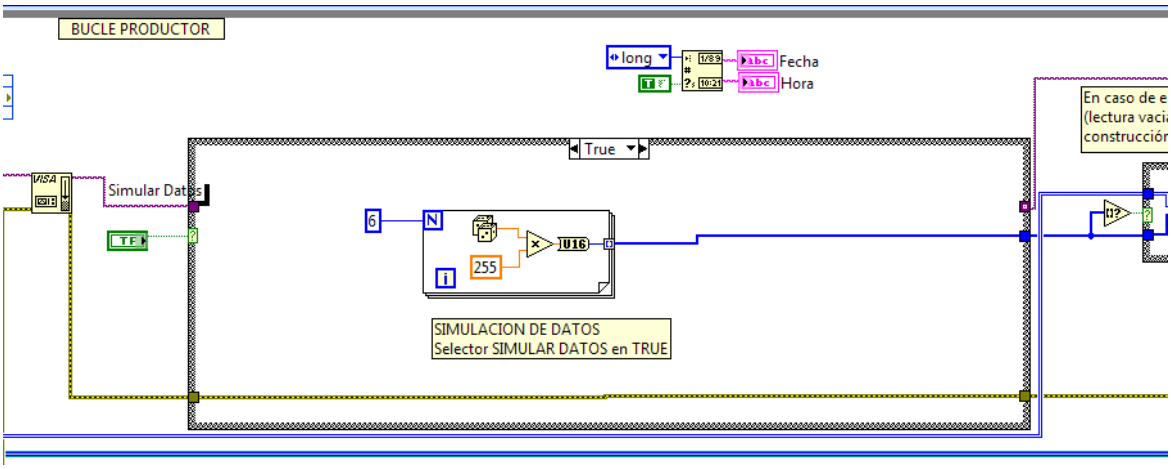


Ilustración 2.57: Estructura CASE para la selección del origen de datos. CASO DATOS SIMULADOS.

En el caso representado en la Ilustración 2.57, para la simulación de datos, encontramos un bucle *FOR* que genera 6 números aleatorios entre 0 y 255, correspondientes a una muestra por sensor. Como puede verse, la línea de datos proveniente del puerto serie se interrumpe a la entrada del *CASE TRUE* y no interactúa con la estructura *FOR*.

Estos datos saldrían de la estructura CASE en forma de un vector de 6 posiciones de 16 bits cada una, para ser tratados como un registro de datos adquiridos directamente de los sensores.

En caso de que el operador seleccionase que los datos no sean simulados (caso por defecto y de interés para este proyecto), se ejecutaría el código correspondiente al caso *FALSE,* mostrado en la Ilustración 2.58.

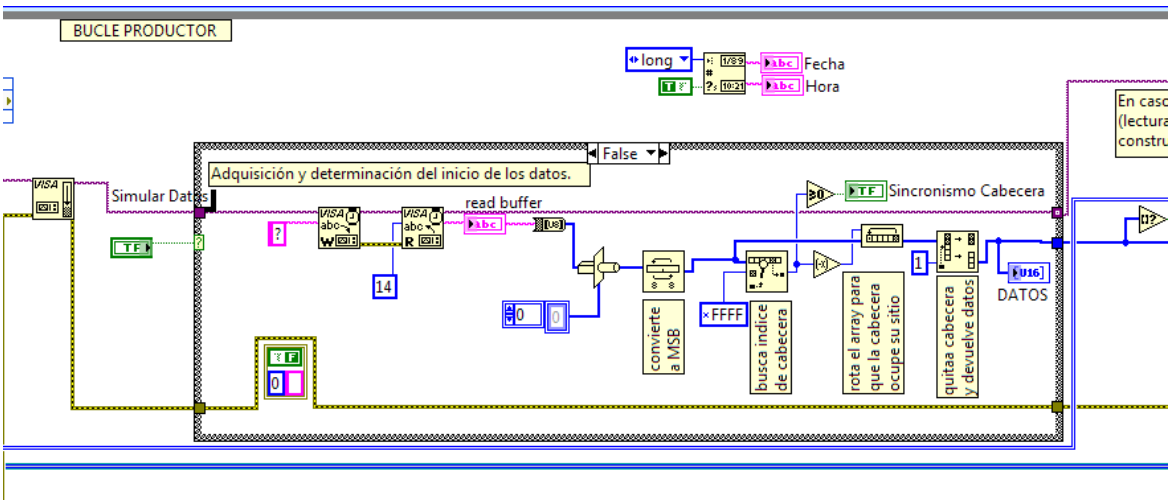


Ilustración 2.58: Estructura CASE para la selección del origen de datos. CASO DATOS PUERTO COM.

En este caso vemos que la línea de datos que provenía del puerto serie (línea morada) constituye una entrada de la estructura CASE. En realidad, esta línea actúa como un conducto de comunicación con el puerto serie, de la que podemos tanto leer como escribir, como veremos ahora.

Inicialmente, el PIC enviaba datos continuamente, y debía ser LabView quien los recuperase cuando fuese necesario. Este funcionamiento no era efectivo, por lo que se optó por hacer envíos de datos a demanda. Para ello, se planteó que el PIC iniciara la adquisición y realizase los envíos ante una consulta de LabView. En la Ilustración 2.58, vemos que primero encontramos la función VISA WRITE, mediante la cual escribimos en la línea de comunicación un "?". El PIC al detectar el carácter "?" inicia el ciclo de trabajo y envía el paquete de datos con una cabecera que indica el inicio del envío. Veamos como lo gestiona.

Una vez que el PIC envía el paquete de datos, formado por 14 bytes, LabView debe decodificarlo y tratarlo. De los 14 bytes, 12 son datos (cada sensor envía un registro de 16 bits, es decir 2 bytes) y 2 son palabra de inicio. Esta palabra de inicio es el valor decimal 65535 (valor máximo para una palabra de 16 bits, en hexadecimal xFFFF), que no puede darse en ningún momento salvo para iniciar el envío de datos, motivo por el cual en el software del PIC se ha de prever que este valor no se genere, de lo contrario generaríamos un error porque LabView consideraría que ha perdido la traza de los bytes restantes.

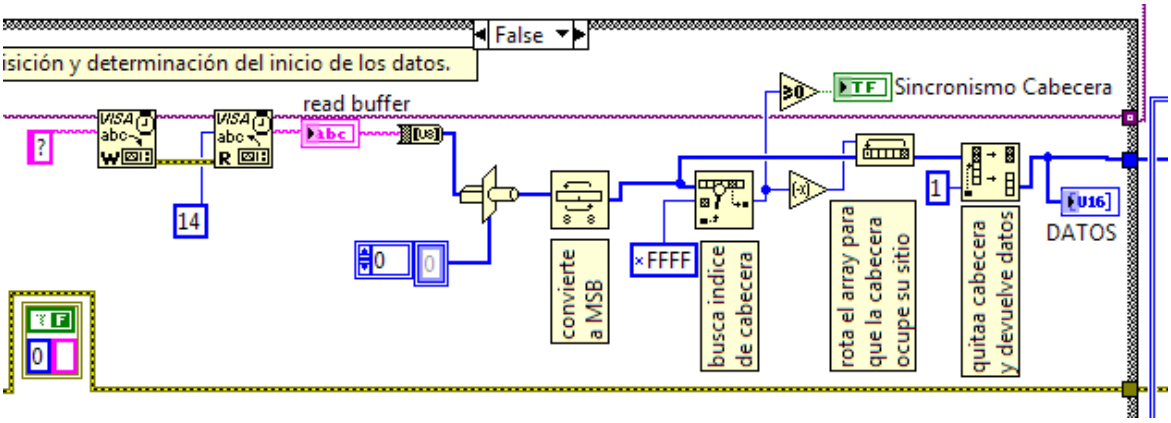


Ilustración 2.59: Tratamiento para la decodificación de los datos

Una vez realizada la consulta al PIC mediante "?", el PIC envía 14 bytes que son leídos como string (indicador rosa *READ BUFFER*) por la función *VISA READ*. Este paquete de datos se formatea en bytes (8bits) (línea gruesa azul) y se concatena en palabras de doble byte (16 bits). De aquí en adelante, no volveremos a interactuar con el puerto serie, por lo que abandonamos la línea de string morada correspondiente al puerto serie y comenzamos a seguir la línea gruesa azul que representa a un vector de una dimensión de números enteros.

Como nuestro PIC envía primero el byte alto (MSB) de cada registro, al concatenarlos hemos constituido una palabra LSB por lo que debemos intercambiar los bytes de cada palabra. Una vez correctamente constituidas las palabras, buscamos la posición en ese vector de dobles palabras (16 bits) de la cabecera (valor xFFFF) y comprobamos el índice de su posición. Luego rotamos el vector hacia el origen tantas veces como indique el índice de posición, de esta forma aseguramos que el vector inicia en la cabecera. Finalmente, eliminamos la primera palabra y obtenemos un vector de doble palabra con 6 posiciones, las 6 lecturas que buscábamos, en el orden que han sido enviadas.

Por fin tenemos la lectura de un segundo, pero necesitamos leer los datos por periodos de diez minutos. Esto supone:

Ec. 2.48: $$6\left(medidas/seg\right) x\, 10\,(\min) x\, 60\left(seg/\min\right) = 3600 medidas$$

Para ello debemos generar una matriz de 6 columnas (señales de sensor) y 600 filas (segundos), lo que supone un total de 3600 medidas. Veamos el código generado para ello en la Ilustración 2.60:

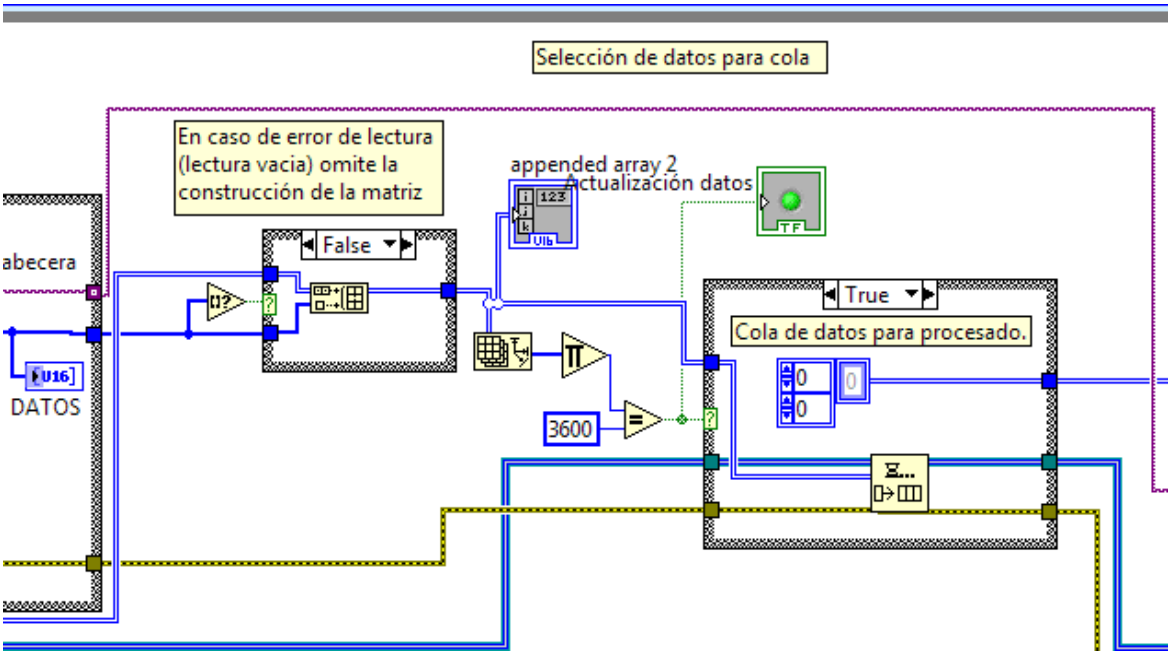


Ilustración 2.60: Código para generar una matriz de 3600 medidas.

Iniciamos la explicación de este tramo de código en la entrega del vector de 6 dobles palabras.

En primer lugar comprobamos que el vector no esté vacio, en ese caso añadimos el nuevo vector mediante la herramienta *BUILD ARRAY* al final de una matriz de dos dimensiones previamente inicializada que se encuentra dentro del caso *FALSE* de la estructura *CASE*. En el caso de que esté vacio, pasaríamos al caso *TRUE* de la estructura *CASE* que omite la inserción del vector a la matriz, esperando a que se cumpla la condición.

Una vez añadido, comprobamos el contenido de la matriz (*appended array2*) y multiplicamos el número de filas por el de columnas no vacías. Si este producto es igual a 3600 (es decir, si hemos conseguido 3600 medidas), ejecutamos el caso *TRUE* de la estructura *CASE* denominada *Cola de datos para procesado* y transferiríamos mediante la función *ENQUEUE ELEMENT* la matriz a la cola de datos. En caso contrario, ejecutaríamos el caso *FALSE* en el que no transmitimos nada y quedaríamos a la espera de que se cumpliese la condición de 3600 medidas.

El indicador booleano *Actualización datos* nos muestra mediante un piloto situado en el panel frontal cuando se produce la transferencia de la matriz a la cola de datos. Esta matriz de 600 filas x 6 columnas, distribuye los datos de cada sensor en columnas según el siguiente vector:

TEMPERATURA	RADIACIÓN	PRESIÓN ATM.	DIRECCIÓN V.	VELOCIDAD V.	HUMEDAD
TEMPERATURA1	RADIACIÓN1	PRESIÓN ATM.1	DIRECCIÓN V.1	VELOCIDAD V.1	HUMEDAD1
TEMPERATURA2	RADIACIÓN2	PRESIÓN ATM.2	DIRECCIÓN V.2	VELOCIDAD V.2	HUMEDAD2
…	…	…	…	…	…

Tabla 2.24: Vector indicador del orden de los datos adquiridos en la matriz almacenada en cola

A la salida del bucle productor, hemos conectado la función *RELEASE QUEUE* (Ilustración 2.61) que entrega el contenido de la cola a la función *DEQUEUE ELEMENT*, situado a la entrada del bucle consumidor, cuya función es extraer de la cola los elementos que previamente se han ido adquiriendo.

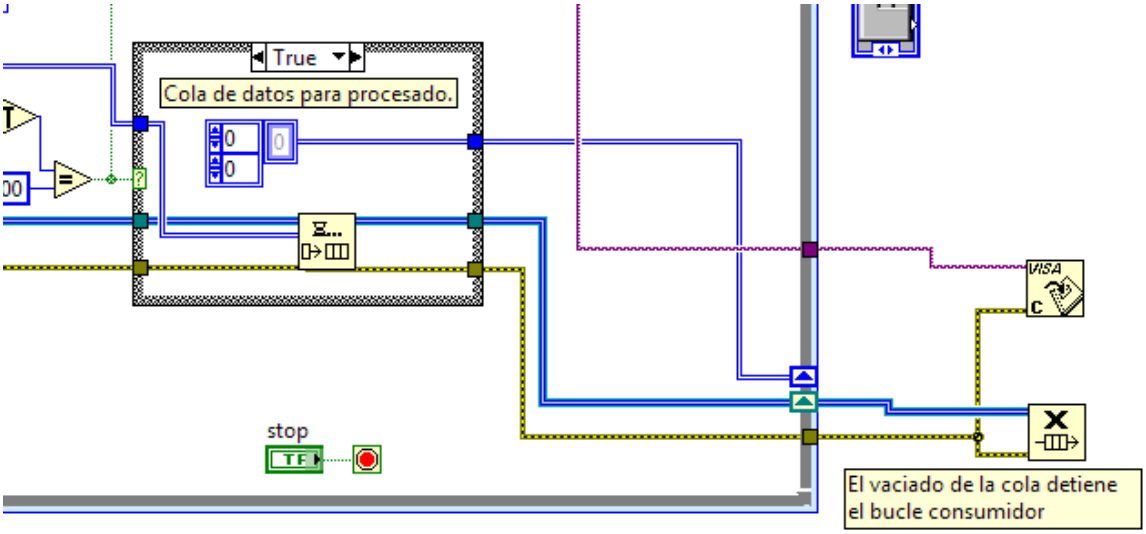


Ilustración 2.61: Detalle de la función *RELEASE QUEUE* a la salida del bucle productor.

2.4.2.3 El bucle consumidor

El bucle consumidor (Ilustración 2.62) presenta una arquitectura bien diferente. Se encuentra constituido por una estructura *WHILE* que se ejecuta siempre que no se produzca un error en la lectura de la cola de datos, ya que su control booleano de *STOP* se encuentra conectado a una constante que lo pararía en cualquier caso de error.

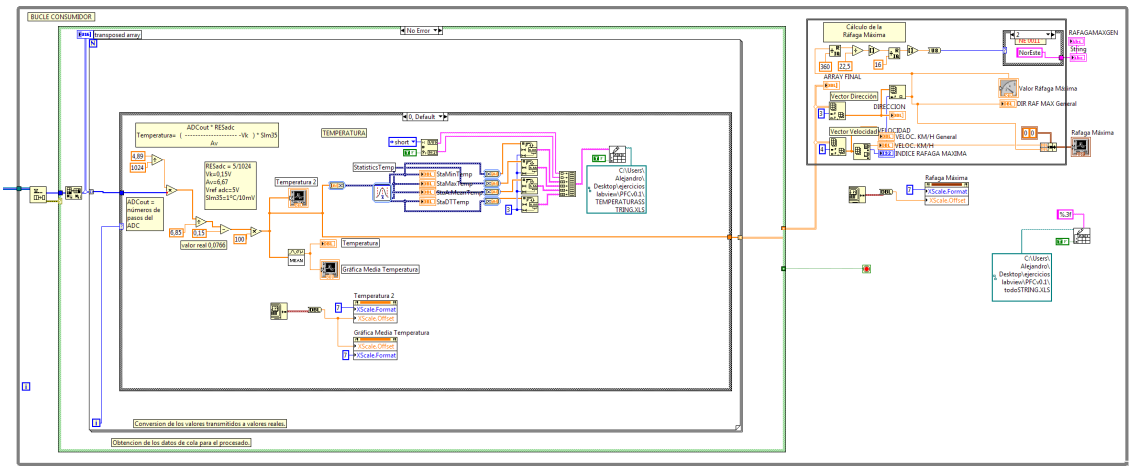


Ilustración 2.62: Bucle consumidor. Captura del diagrama de bloques.

Dentro de esta estructura se sitúa la función *DEQUEUE ELEMENT* comentada anteriormente, que extrae los datos de la cola y cuyo canal de error controla los casos de una estructura *CASE* alojada en su interior que gestiona el error de la cola de datos. Esta estructura posee dos casos posibles:

- ✓ *ERROR:* donde se encuentra la constante que detendría el bucle.
- ✓ *NO ERROR:* Donde se gestionan los datos adquiridos.

En el caso de no error, (Ilustración 2.63) mediante la función *TRANSPONSE 2D ARRAY*, transponemos la matriz que se ha almacenado en la cola de datos, obteniendo una matriz de 6 filas x 600 columnas en la que las medidas de cada sensor se distribuirán en filas, conforme se indica en la figura (Tabla 2.25).

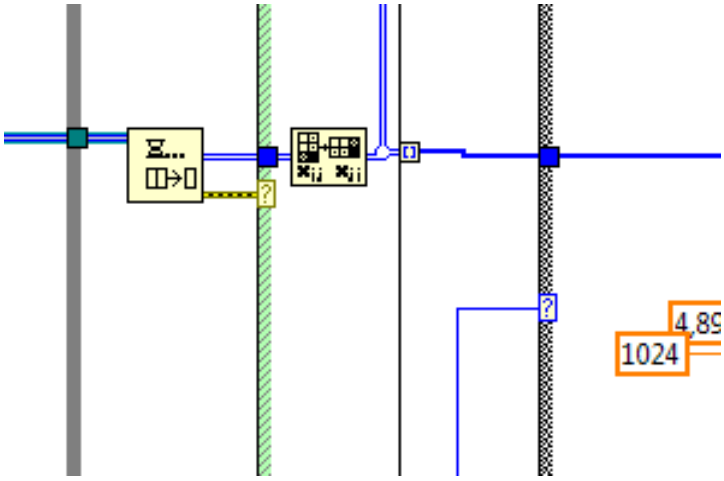


Ilustración 2.63: Detalle de la función *TRANSPONSE 2D ARRAY*

TEMPERATURA	…
RADIACIÓN	…
PRESIÓN	…
DIRECCIÓN V.	…
VELOCIDAD V.	…
HUMEDAD	…

Tabla 2.25: Vector indicador del orden de los datos adquiridos en la matriz almacenada en cola

2.4.2.4 Adecuación y gestión de los datos.

Ahora que obtenemos los datos de cada sensor en vectores-filas de 600 valores, lo que hacemos es controlar el estado de la estructura *CASE* mediante el índice de la fila con la que trabajamos, así podemos establecer la correspondencia entre casos y magnitudes tal que:

CASO 1	TEMPERATURA
CASO 2	RADIACIÓN SOLAR
CASO 3	PRESIÓN ATMOSFÉRICA
CASO 4	DIRECCIÓN DE VIENTO
CASO 5	VELOCIDAD DE VIENTO
CASO 6	HUMEDAD

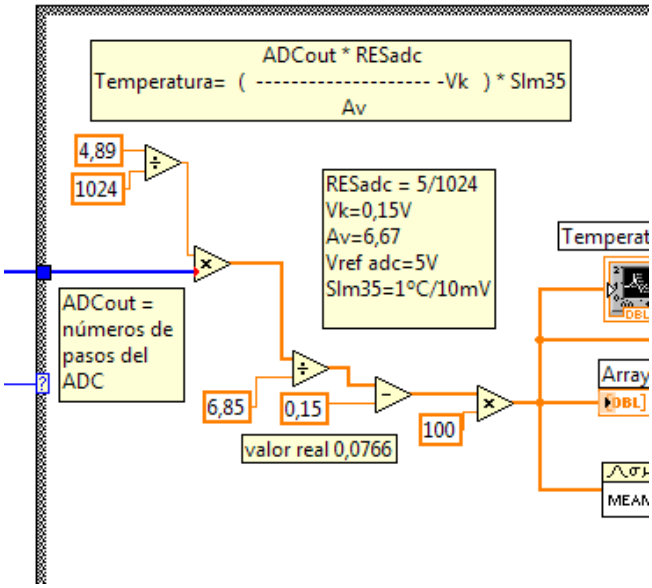
Tabla 2.26: Distribución de casos para la adecuación y gestión de datos del bucle consumidor según magnitudes.

Cada magnitud se gestiona aproximadamente del mismo modo. Primero se realiza una adecuación de la señal para que sea expresada en sus correspondientes unidades, luego se representa gráficamente y se obtienen los valores estadísticos deseados que a su vez son representados gráficamente y almacenados en archivos para su gestión posterior si fuese preciso.

Para entender el código desarrollado en su totalidad analizaremos las adecuaciones de unidades realizadas para cada magnitud y tan solo en una de ellas analizaremos la parte común correspondiente a la representación grafica, el cálculo de valores estadísticos y el almacenamiento en archivos.

Las adecuaciones de unidades de cada magnitud responden al cálculo inverso realizado para su acondicionamiento explicado en los correspondientes subapartados del apéndice 2.3.1 titulado “Los sensores y sus circuitos de adaptación de señal. Alternativas propuestas a los sensores comerciales mediante componentes discretos.”

2.4.2.4.1 Caso 1. Adecuación de las unidades de Temperatura.



NOTA: la tensión real estabilizada entregada al PIC es de 4.89V en lugar de 5V. En la práctica he tenido que hacer el ajuste a la tensión de referencia para obtener un valor más aproximado al resultado teórico.
Lo mismo sucede con las tolerancias de las resistencias empleadas para ajustar la ganancia.

Ilustración 2.64: Calculo del valor de temperatura correspondiente al valor de tensión medido en el ADC.

Este tramo de código, obtiene el valor de temperatura despejado de la ecuación de acondicionamiento e incorporando la conversión del ADC. La lectura sería, partiendo de la línea azul que representa la lectura del ADC:

Ec. 2.49:
$$ADC_{OUT} \cdot RES_{ADC} = Vin_{ADC}$$

Ec. 2.50:
$$RES_{ADC} = \frac{4.89\,V}{1024\,cuentas} = 4.77 \cdot 10^{-3}\,V/cuenta$$

Puesto que la tensión de entrada del ADC es la salida del circuito acondicionador, retomamos la ecuación de su FDT:

Ec. 2.9:
$$V_{out} = \left(1 + \frac{R_2}{R_1}\right) \cdot V_{lm35}$$

Despejando obtenemos:

Ec. 2.51:
$$V_{lm35} = \frac{V_{out}}{\Delta v} = \frac{ADC_{OUT} \cdot RES_{ADC}}{\left(1 + \frac{R_2}{R_1}\right)}$$

Ahora para calcular la temperatura despejamos de la FDT del sensor LM35. Recordemos que hicimos un desplazamiento de la señal para permitir la medida de valores negativos.

Ec. 2.52:
$$Temp[^{\circ}C] = (V_{lm35} - Vk) \cdot S$$

Nuestra ecuación final resultaría:

Ec. 2.53:
$$Temp[^{\circ}C] = \left(\frac{ADC_{OUT} \cdot RES_{ADC}}{\left(1 + \frac{R_2}{R_1}\right)} - Vk \right) \cdot S$$

Sustituyendo los valores teóricos calculados:

Ec. 2.54:
$$Temp[^{\circ}C] = \left(\frac{ADC_{OUT}[Cuentas] \cdot 4.77 \cdot 10^{-3}\left[\frac{V}{cuenta}\right]}{\left(1 + \frac{5.670k}{1k}\right)} - 0.15[V] \right) \cdot \frac{1^{\circ}C}{10mV}$$

2.4.2.4.2 Caso 2. Adecuación de las unidades de Radiación.

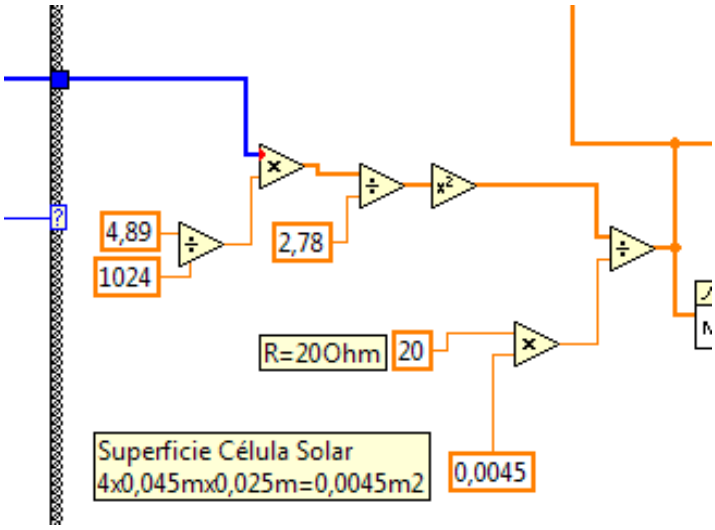


Ilustración 2.65: Calculo del valor de Irradianza para el valor de tensión medido por el ADC.

La magnitud a medir por el Piranómetro es la Irradianza que se expresa en W/m^2. Partiendo de la ecuación final del circuito de acondicionamiento, despejamos

Ec. 2.55:
$$G = \left(\frac{V_{OUT}}{\Delta V}\right)^2 \cdot \frac{1}{R \cdot S}\left[\frac{W}{m^2}\right]$$

Recordemos que la tensión de salida del circuito acondicionador es la entrada al ADC

Ec. 2.56:
$$ADC_{OUT} \cdot RES_{ADC} = Vin_{ADC} = V_{OUT}$$

Y que su resolución responde a la ecuación:

Ec. 2.57:
$$RES_{ADC} = \frac{4.89\,V}{1024\,cuentas} = 4.77 \cdot 10^{-3}\,V/cuenta$$

Por tanto, la ecuación programada para el cálculo de la Irradianza resulta:

Ec. 2.58:
$$G = \left(\frac{ADC_{OUT} \cdot RES_{ADC}}{\Delta V}\right)^2 \cdot \frac{1}{R \cdot S}$$

2.4.2.4.3 Caso 3. Adecuación de las unidades de Presión.

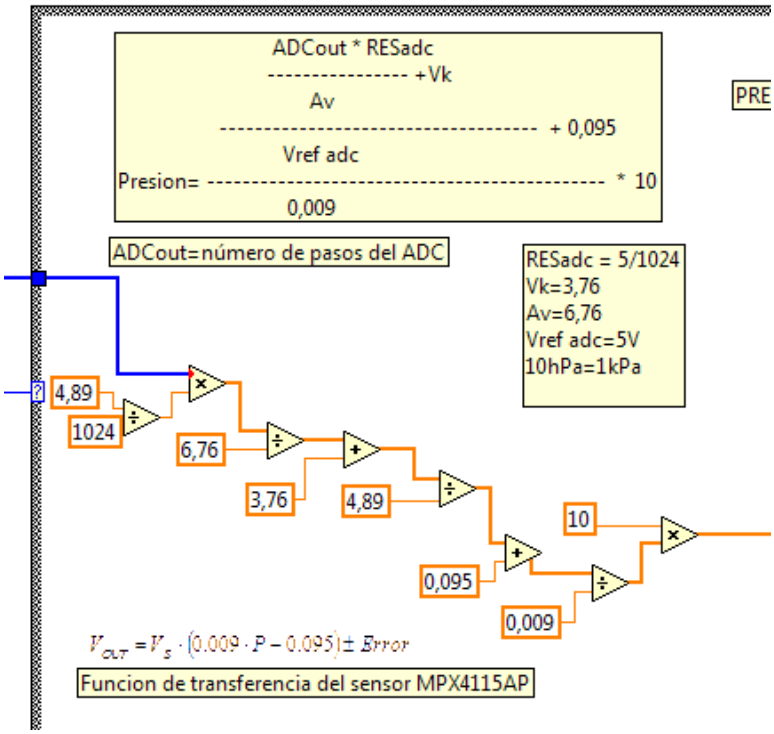


Ilustración 2.66: Calculo del valor de presión atmosférica conforme a la tensión medido por el ADC.

Retomando la función de transferencia del circuito acondicionador:

Ec. 2.59: $$\frac{V_{1B}}{V_{1A}} = 1 + \frac{R_1}{R_2} \Rightarrow \quad \Delta v = 6.67 = 1 + \frac{R_1}{R_2}$$

Ec. 2.60: $$V_{1A} = V_{IN} - V_K$$

Obtenemos que:

Ec. 2.61: $$V_{1B} = \left(1 + \frac{R_1}{R_2}\right) \cdot \left(V_{IN} - V_K\right)$$

Puesto que V_{IN} es el valor de salida del sensor MPX41145AP, sustituimos su FDT en la ecuación resultante de su circuito acondicionador:

Ec. 2.62: $$V_{IN} = Vout_{MPX\,4115\,AP} = V_{ref} \cdot \left(0.009 \cdot P - 0.0095\right)$$

Ec. 2.63: $$V_{1B} = \left(1 + \frac{R_1}{R_2}\right) \cdot \left[\left(V_{ref} \cdot \left(0.009 \cdot P - 0.0095\right)\right) - V_K\right]$$

Despejando el valor de la presión P:

Ec. 2.64: $$P = \frac{\dfrac{\dfrac{V_{1B}}{1 + \left(R_1 / R_2\right)} + V_K}{V_{ref}} + 0.095}{0.009}$$

Sustituyendo los valores teóricos y sabiendo que V_{1B} corresponde al valor de tensión de entrada al ADC:

$$V_{1B} = ADC_{OUT} \cdot RES_{ADC}$$

Ec. 2.65: $$P[kPa] = \frac{\dfrac{\dfrac{ADC_{OUT}[cuentas] \cdot 4.77 \cdot 10^{-3}\left[V / Cuenta\right]}{6.76} + 3.76[V]}{4.89[V]} + 0.095}{0.009}$$

Finalmente podemos obtener una ecuación reducida expresada en hPa.

Ec. 2.66: $$\boxed{P[hPa] = 0.16 \cdot ADC_{OUT} + 959.9}$$

2.4.2.4.4 Caso 4. Adecuación de las unidades de Dirección del Viento.

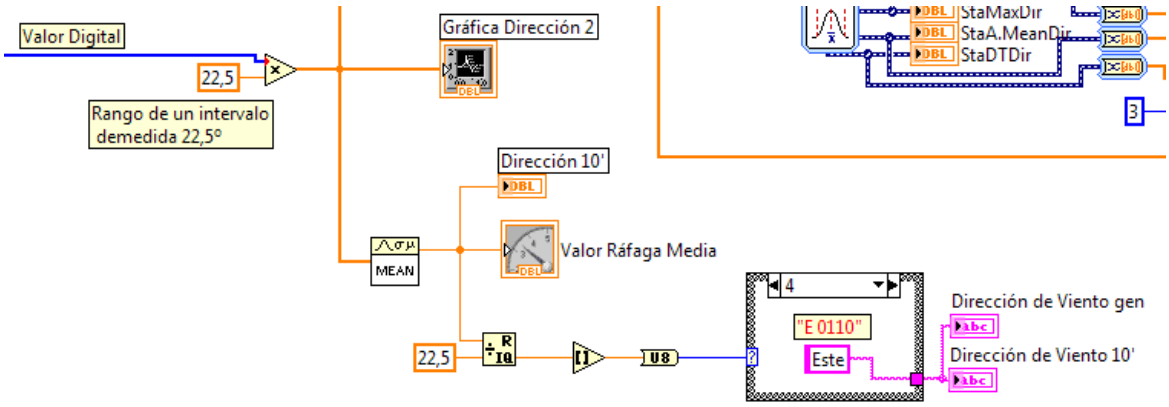


Ilustración 2.67: Cálculo de la dirección del viento correspondiente al valor digital muestreado.

En el caso de la dirección del viento, el valor adquirido es un valor digital. De entre los 360 grados de la circunferencia, mediremos 16 estados por lo que cada estado comprende un rango de 22.5º. Debido a que el valor final buscado es un valor medio, multiplicaremos el valor medido por el rango de cada medida obteniendo la posición angular de la medida.

POS	Denominación	Abreviatura	D3	D2	D1	D0	Angulo (º)
0	NORTE	N	0	0	0	0	0
1	NORTE-NORESTE	N-NE	0	0	0	1	22,5
2	NORESTE	NE	0	0	1	1	45
3	ESTE-NORESTE	E-NE	0	0	1	0	67,5
4	ESTE	E	0	1	1	0	90
5	ESTE-SURESTE	E-SE	0	1	1	1	112,5
6	SURESTE	SE	0	1	0	1	135
7	SUR-SURESTE	S-SE	0	1	0	0	157,5
8	SUR	S	1	1	0	0	180
9	SUR-SUROESTE	S-SO	1	1	0	1	202,5
10	SUROESTE	SO	1	1	1	1	225
11	OESTE-SUROESTE	O-SO	1	1	1	0	247,5
12	OESTE	O	1	0	1	0	270
13	OESTE-NOROESTE	O-NO	1	0	1	1	292,5
14	NOROESTE	NO	1	0	0	1	315
15	NORTE-NOROESTE	N-NO	1	0	0	0	337,5

Tabla 2.27: Asignación de los puntos cardinales a los casos de la estructura CASE.

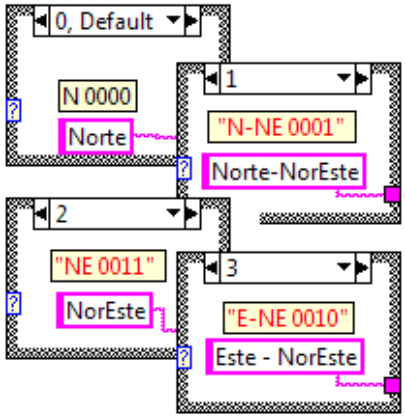


Ilustración 2.68: Ejemplos de algunos casos de la estructura.

En el caso de la dirección del viento he programado una cadena de texto que complementa la dirección angular con el nombre del correspondiente punto cardinal. Para ello, dividimos el valor medio de la dirección del viento obtenido para el periodo de muestreo diezminutal, entre la resolución de la medida angular (22,5º) determinando el posicionamiento teórico para la dirección angular media del viento en el periodo

diezminutal. Mediante el uso de este valor, controlamos los casos de una estructura *CASE* que indica mediante cadena de texto el origen de la ráfaga media (ver Tabla 2.27).

2.4.2.4.5 Caso 5. Adecuación de las unidades de Velocidad del Viento.

La velocidad del viento se calcula a partir del periodo que tarda el eje del anemómetro en completar una vuelta. Despejando la velocidad lineal de las ecuaciones descritas en el apartado del anemómetro (Ec. 2.35), podemos determinar su valore.

Ec. 2.35: $$v = \frac{2\pi r}{T}$$

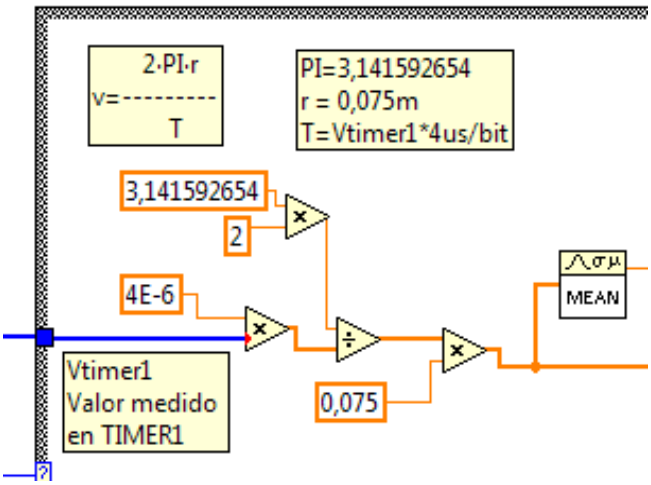


Ilustración 2.69: Cálculo de la dirección del viento correspondiente al valor digital muestreado.

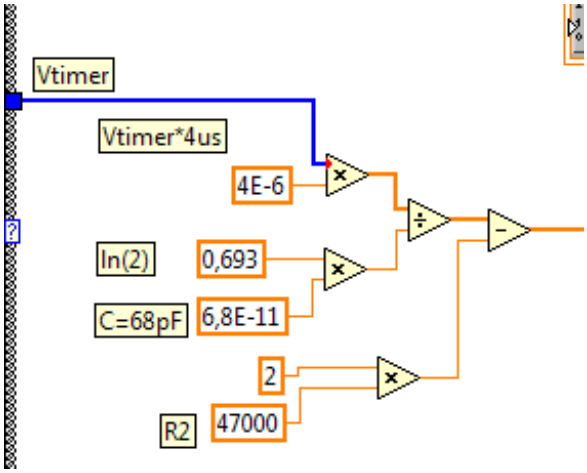
El periodo de la onda viene determinado por el producto del valor del contador V_{TIMER}[bits] por el periodo del oscilador del contador [4us]:

Ec. 2.67: $$T = V_{TIMER1} \cdot T_{OSC} = V_{TIMER1} \cdot T_{CICLO} \cdot PREESCALER = V_{TIMER1} 1\mu s \cdot 4$$

Ec. 2.68: $$v = \frac{2\pi r}{4\left[\mu s / bit\right] \cdot V_{TIMER1}[bits]}$$

Ec. 2.69: $$v = \frac{2\pi \cdot 0.075}{4 \cdot 10^{-6} \cdot V_{TIMER1}}$$

2.4.2.4.6 Caso 6. Adecuación de las unidades de Humedad.



El cálculo de la humedad atmosférica, se realiza de un modo muy similar al de la velocidad de viento. Se trata de calcular el periodo de una onda rectangular, a partir del valor almacenado en un registro de 16 bits que ejerce las funciones de contador. Lo primero por tanto es calcular a partir del número de bits contados el tiempo de duración de esa onda.

Ya que el preescaler utilizado es el mismo (1:4):

Ec. 2.67: $$T = V_{TIMER1} \cdot T_{OSC} = V_{TIMER1} \cdot T_{CICLO} \cdot PREESCALER = V_{TIMER1} 1\mu s \cdot 4$$

De la ecuación Ec. 2.16:

Ec. 2.16: $$T = \ln(2) \cdot (R_1 + 2 \cdot R_2) \cdot C_1$$

Ec. 2.70: $$V_{TIMER} \cdot 4\mu s = \ln(2) \cdot (R_1 + 2 \cdot R_2) \cdot C_1$$

Despejando R_1 obtenemos:

Ec. 2.71: $$R_1 = \frac{V_{TIMER} \cdot 4\mu s}{\ln(2) \cdot C_1} - 2 \cdot R_2$$

2.4.2.4.7 Elementos comunes. Representación grafica, cálculos de valores estadísticos y almacenamiento en archivos.

Como salida de cada uno de los procesos de adecuación de unidades, obtenemos un vector en el que se almacena el valor medido expresado en las unidades correspondientes a cada una de las magnitudes a medir. Estos vectores serán sobre los que se realizarán los cálculos estadísticos cuyos valores serán representados y almacenados en archivos. Estas operaciones se realizan con el mismo código, por lo que la explicación del mismo que a continuación se detalla es idéntica para cada magnitud, a excepción de los nombres de variables que intervienen, que para cada magnitud tendrán sus correspondientes nombres.

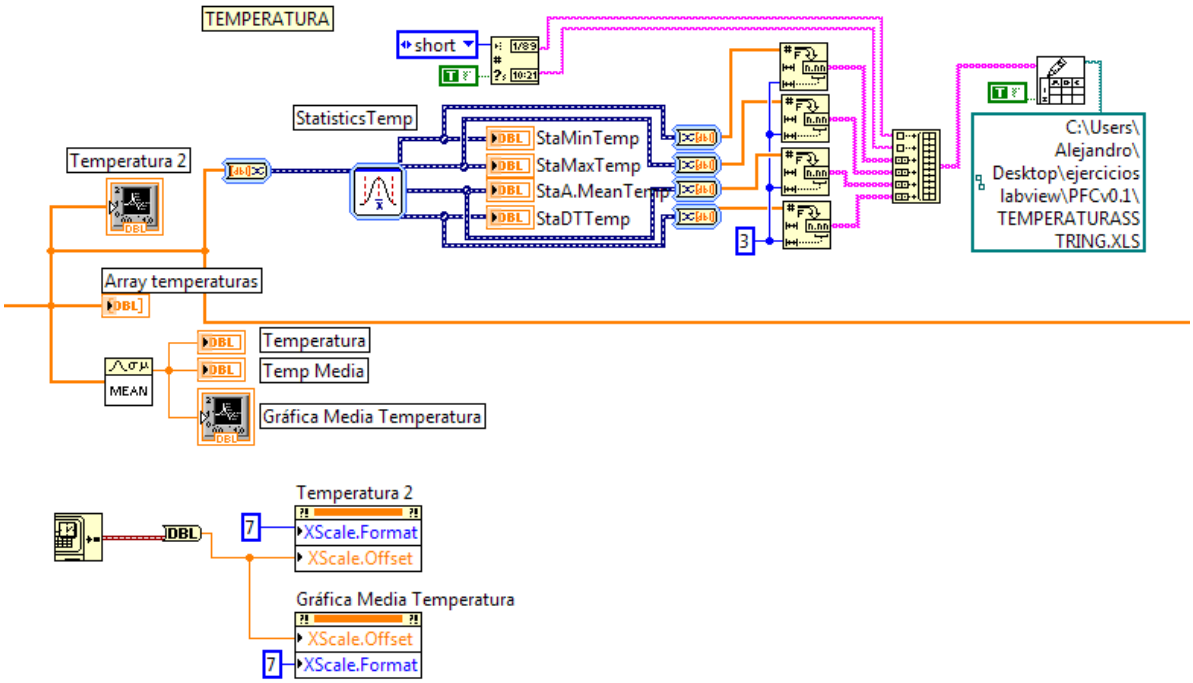


Ilustración 2.70: Cálculo estadístico, representación y almacenamiento de valores.

Basándonos en la Ilustración 2.70, vemos que la línea entrante desemboca en el vector (*ARRAY TEMPERATURAS*). Este vector es el comentado en el párrafo anterior que contiene los valores medidos expresados en las correspondientes unidades de la magnitud medida (en el caso particular de la ilustración, se trata de temperatura expresada en grados centígrados [ºC]). El contenido de este vector (los 600 valores medidos de temperatura) es representado mediante la herramienta *WAVEFORM CHARTS* titulada *TEMPERATURA2,* situada un poco más arriba en la ilustración.

Además, el vector *ARRAY TEMPERATURAS* es conducido a la herramienta *STATISTICS TEMP* que nos permite obtener los parámetros estadísticos que seleccionamos en su menú contextual. Para el objeto de este proyecto, se han seleccionado los parámetros de Máximo, Mínimo, Media y Desviación Típica.

De estas magnitudes estadísticas, hemos representado gráficamente la media aritmética que por comodidad se ha extraído de forma independiente para su representación mediante la función *MEAN*. Este valor medio es representado mediante un *WAVEFORM CHARTS* que he titulado *GRÁFICA MEDIA TEMPERATURA*.

Para finalizar, el vector *ARRAY TEMPERATURAS* sale por el lado derecho de la ilustración (línea gruesa de color naranja que representa array de números racionales) abandonando la estructura *CASE DE TRATAMIENTO DE MAGNITUDES*, el bucle *FOR* y finalmente la estructura CASE de NO ERROR permaneciendo solo dentro del la estructura del *BUCLE CONSUMIDOR*. El hecho de que la estructura *CASE DE TRATAMIENTO DE MAGNITUDES* entregue al bucle *FOR* su vector, hace que a la salida del bucle *FOR* obtengamos una matriz constituida por los seis vectores de magnitudes (matriz *ARRAY FINAL*) y que suponen la matriz final que contiene las medidas expresadas en sus correspondientes magnitudes.

En la zona baja de la ilustración se muestra un fragmento de código del que aún no se ha hablado. Se trata de una adecuación de los ejes para la representación grafica de los valores. Mediante la función *GET DATE/TIME IN SECONDS* y una función *PROPERTY NODE*, indicamos que el valor del ejes X sea expresado en formato fecha/hora.

2.4.2.4.8 Cálculo de la Ráfaga Máxima.

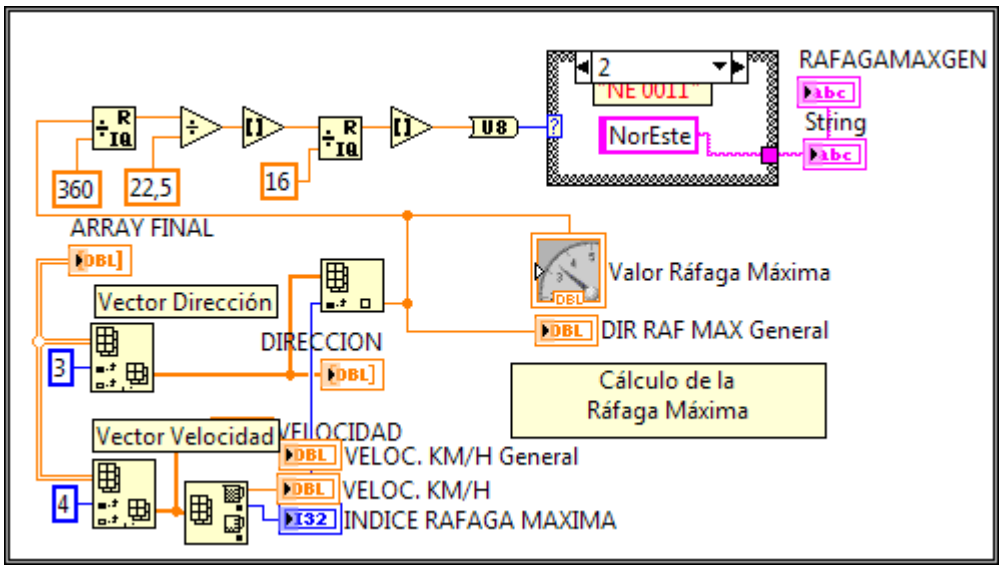


Ilustración 2.71: Cálculo de la ráfaga máxima de viento.

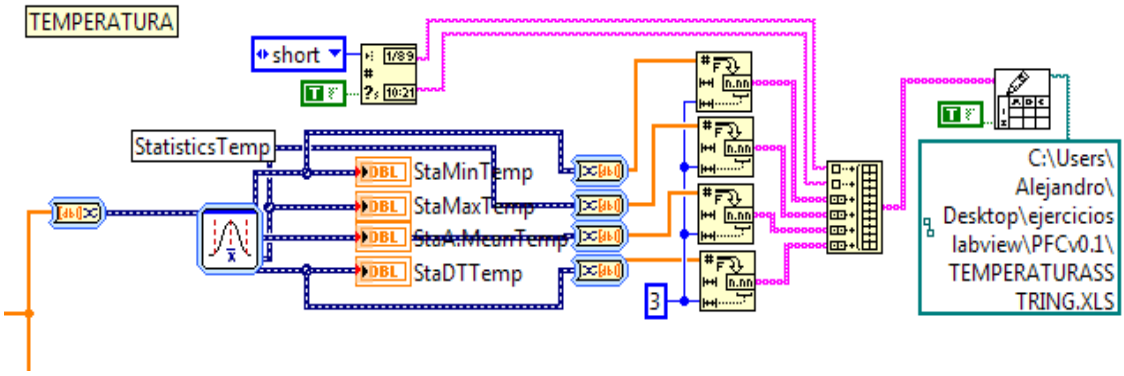
Puesto que los valores que se tratan referentes al viento son valores medios diezminutales, en ocasiones es preciso conocer cuando y de cuanta intensidad ha sido la máxima velocidad de viento. Esta medida recibe el nombre de Ráfaga Máxima.

Para calcular el valor de la ráfaga máxima, tomamos la matriz diezminutal de medidas *ARRAY FINAL* y extraemos en dos vectores independientes las filas 3 y 4 correspondientes a la dirección y velocidad del viento. De este último vector, buscamos y el valor máximo representándolo en el panel frontal como velocidad de ráfaga máxima e indicamos su índice de posición al vector de direcciones, extrayendo esta dirección y mostrándola en el panel frontal como dirección de ráfaga máxima.

En la Ilustración 2.71 puede verse el desarrollo de este código. Mediante la función *INDEX ARRAY* a la que le entregamos el índice que queremos aislar, extraemos los vectores indicados y luego con las funciones *ARRAY MAX & MIN* extraemos los valores máximos y sus correspondientes índices (con esta función también es posible extraer el valor mínimos del vector y su índice). Para representar la dirección obtenida se aplica el mismo procedimiento explicado para la adecuación de las unidades de Dirección del Viento.

2.4.2.4.9 Almacenamiento de datos en archivos.

Con objeto de poder analizar si fuese necesario las tendencias de las magnitudes meteorológicas adquiridas, los resultados estadísticos de la medidas diezminutales se constituirán en un vector junto con la hora y fecha de la toma. Este vector se almacenará en modo acumulativo en un archivo de nombre *<MAGNITUD>&STRING.XLS,* según la distribución FECHA, HORA, VALOR MINIMO, VALOR MAXIMO, MEDIA, DT.



A continuación, se muestra a modo de ejemplo un fragmento del archivo TEMPERATURASSTRING.xls

08/01/2012	22:57:03	21,321	21,321	21,321	0
08/01/2012	22:57:13	21,251	21,321	21,307	0,029
08/01/2012	22:57:23	21,321	21,391	21,335	0,029
08/01/2012	22:57:33	21,321	21,391	21,328	0,022
08/01/2012	22:57:43	21,321	21,391	21,328	0,022
08/01/2012	22:57:53	21,251	21,321	21,307	0,029
08/01/2012	22:58:03	21,321	21,391	21,328	0,022

Ilustración 2.72: Fragmento capturado del archivo TEMPERATURASSTRING.xls

2.4.2.4.10 Captura y representación de video.

El código diseñado para la captura de video, se ha ha ubicado fuera de los bucles de producción y consumo, por lo que su ejecución se realiza como la de un módulo independiente. Mediante la combinación de herramientas del menú *IMAQ USB* de la tableta de herramientas *VISION AND MOTION* y las de *IMAGE MANAGEMENT* de la tableta *VISION UTILITIES* principalmente puede establecerse el siguiente código.

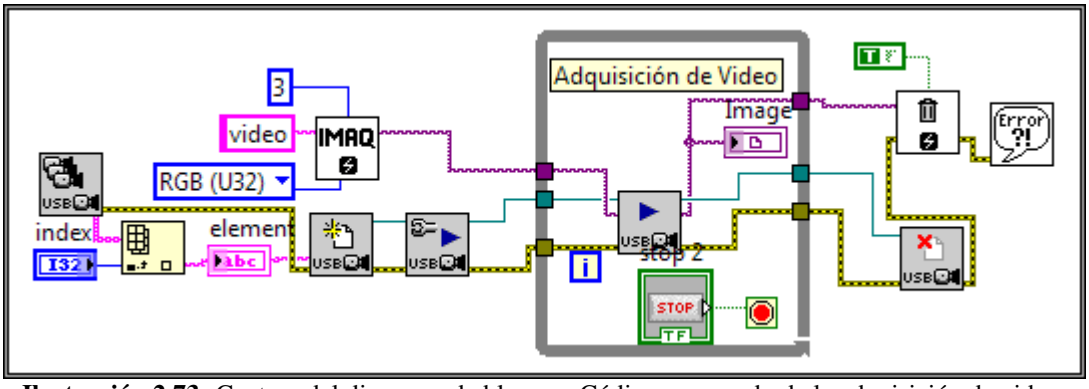


Ilustración 2.73: Captura del diagrama de bloques. Código encargado de la adquisición de video.

Partimos de la función *ENUMERATE CAMERAS*, esta función crea una lista de cámaras disponibles. Introducimos esta lista (para nuestro caso 1 sola cámara) en un vector y seleccionamos mediante el índice la cámara deseada (en nuestro caso concreto el índice 0). El indicador *ELEMENT* muestra el dispositivo seleccionado (en la pestaña del panel frontal *CONFIGURACION*). Inicializamos una sesión *IMAQ USB* para la captura de la cámara usb y generamos un espacio de memoria virtual para el almacenamiento de las imágenes mediante *IMAQ CREATE*. Configuramos e inicializamos la captura de las imágenes mediante *IMAQ USB GRAB SETUP*. A partir de este punto el proceso de inicialización de captura de imágenes a través del puesto USB está conpletado. Accedemos dentro del *BUCLE WHILE* donde la herramienta *IMAQ USB GRAB ADQUIRE* genera las imágenes y las muestra a través del indicador *IMAGE*. Para salir de este bucle de capturas es preciso detener el bucle *WHILE* mediante el booleano *STOP* para cerrar la sesión de captura y liberar la memoria virtual, de lo contrario se generaría un error que provoca el borrado intempestivo del espacio de memoria virtual, cerrando LabView. Por ello se ha dispuesto de un pulsador *STOP* junto a la imagen de cámara.

En el origen de este diseño, toda este código quedaba simplificado mediante una herramienta denominada *VISION ADQUISITION* de la paleta *VISION EXPRESS* perteneciente a una expansión de prueba de software denominada *IMAQ VISION*. Puesto que finalizó el periodo de evaluación se ha realizado el código alternativo que funciona sin limitaciones.

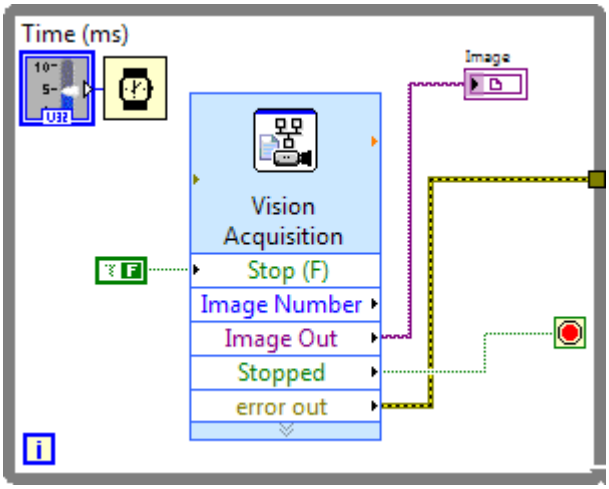
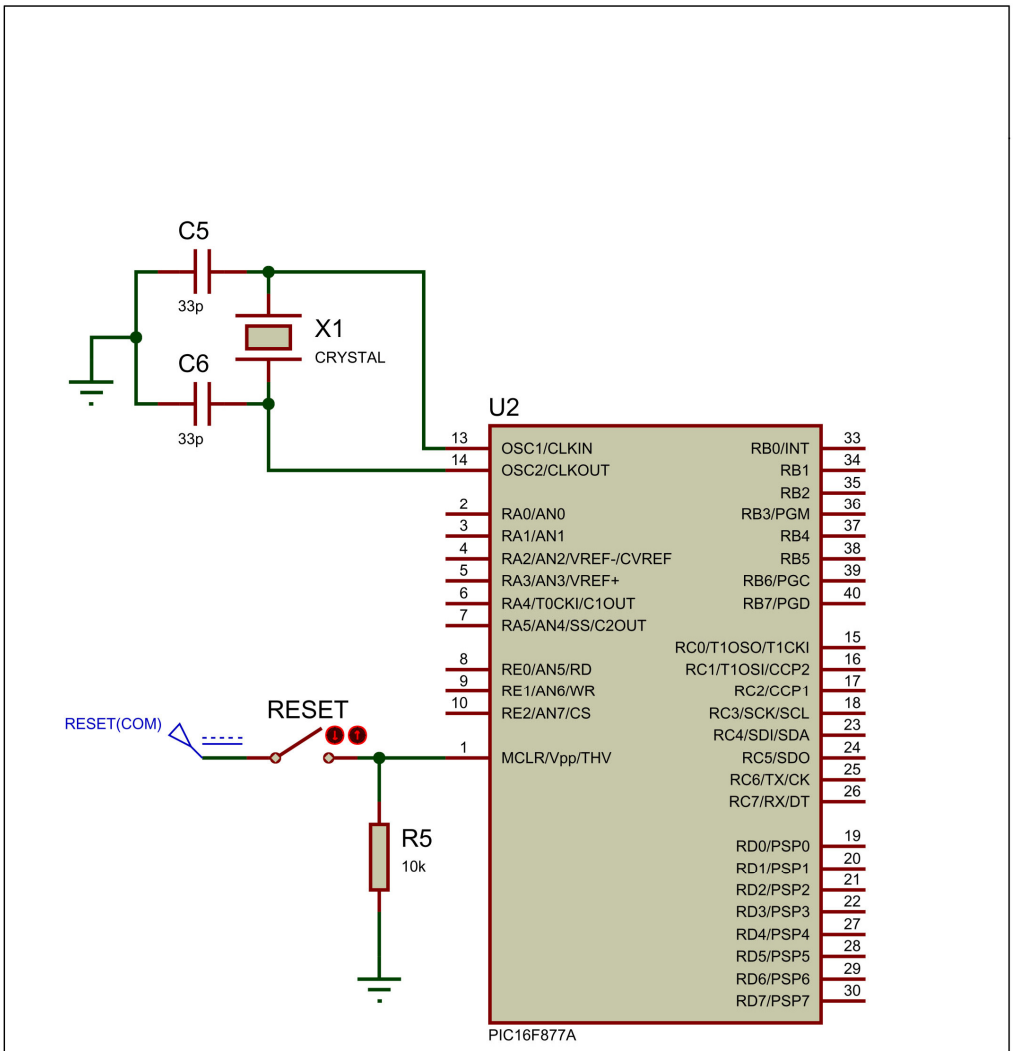
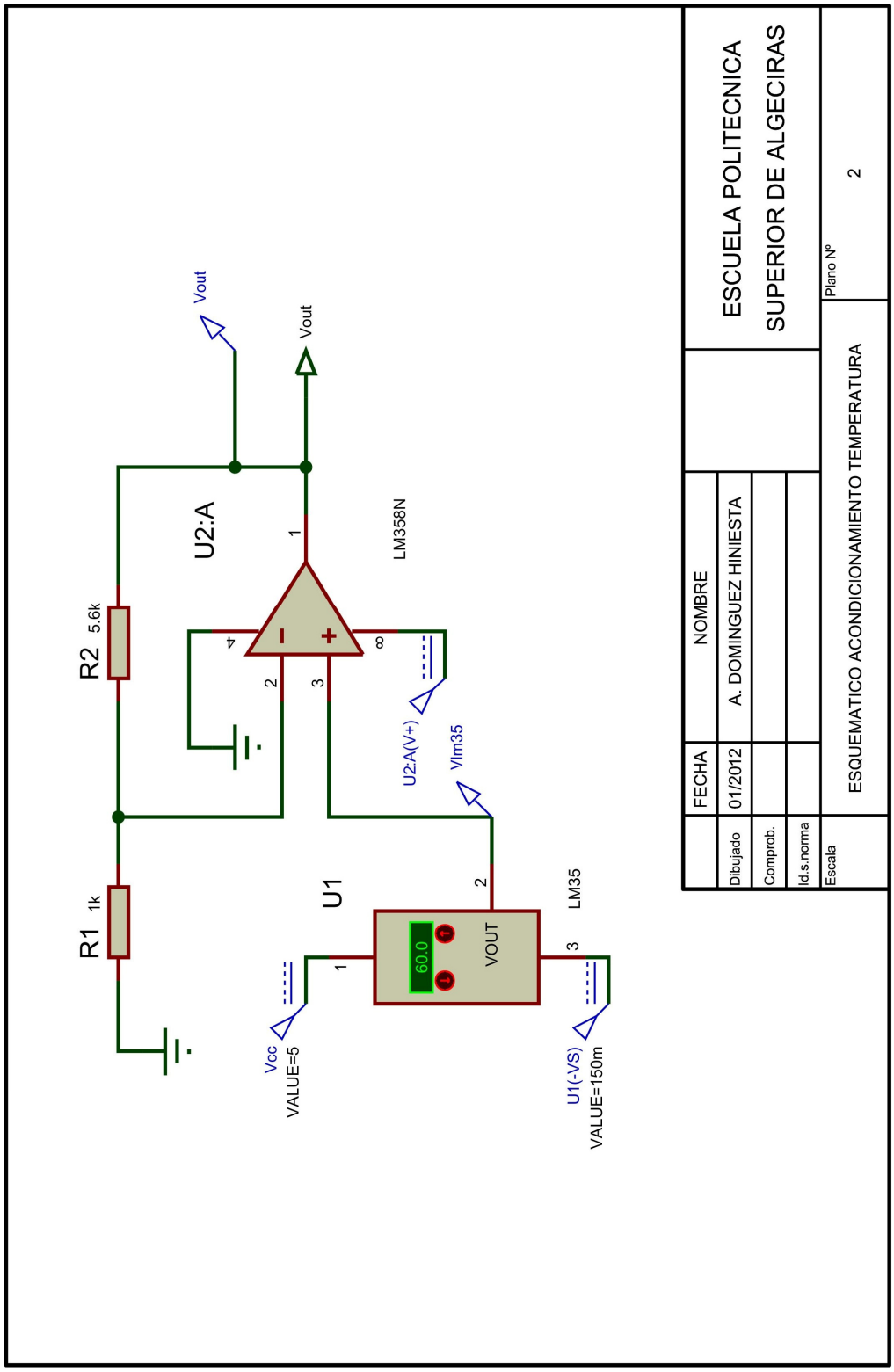


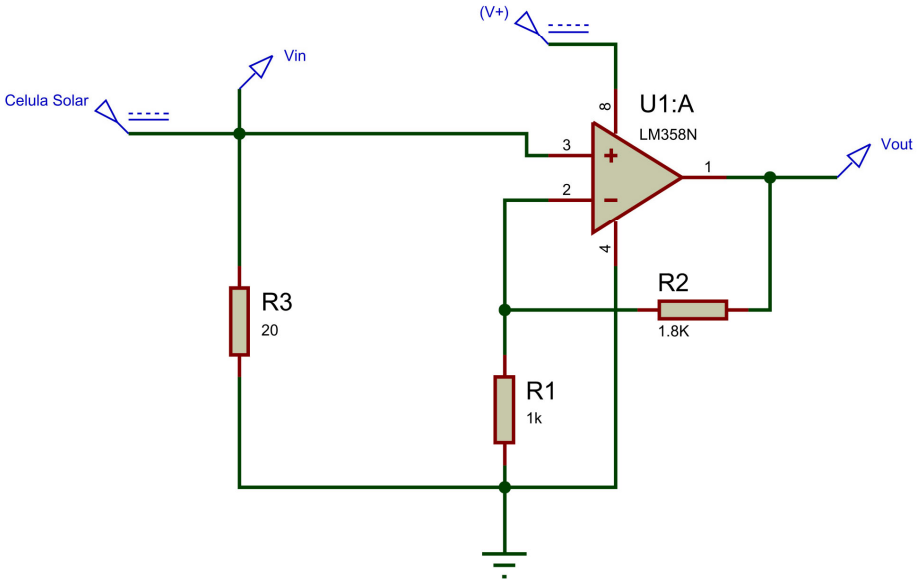
Ilustración 2.74: Código de captura de video USB realizado con la expansión IMAQ VISION

3 ESQUEMAS Y PLANOS

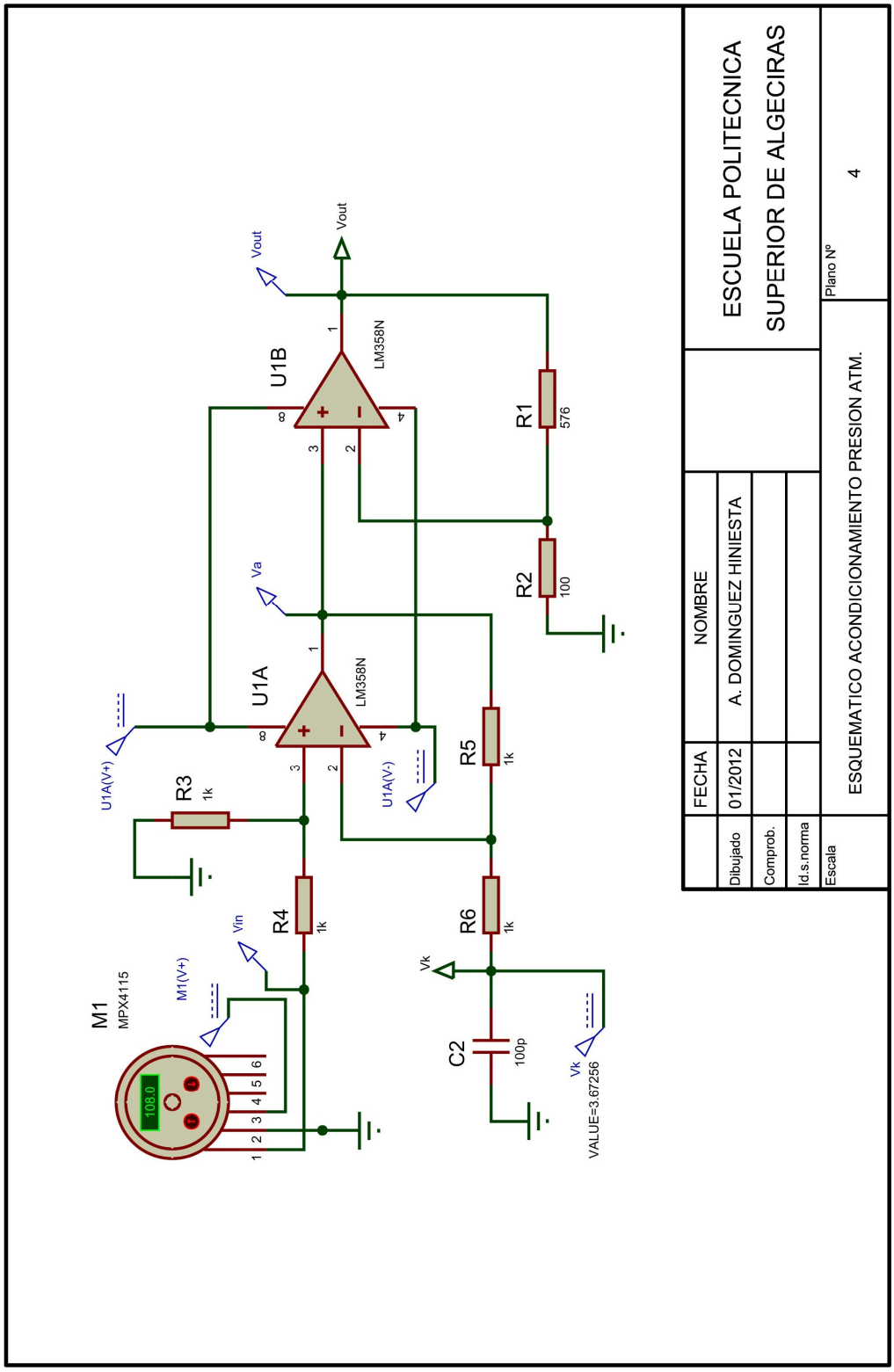


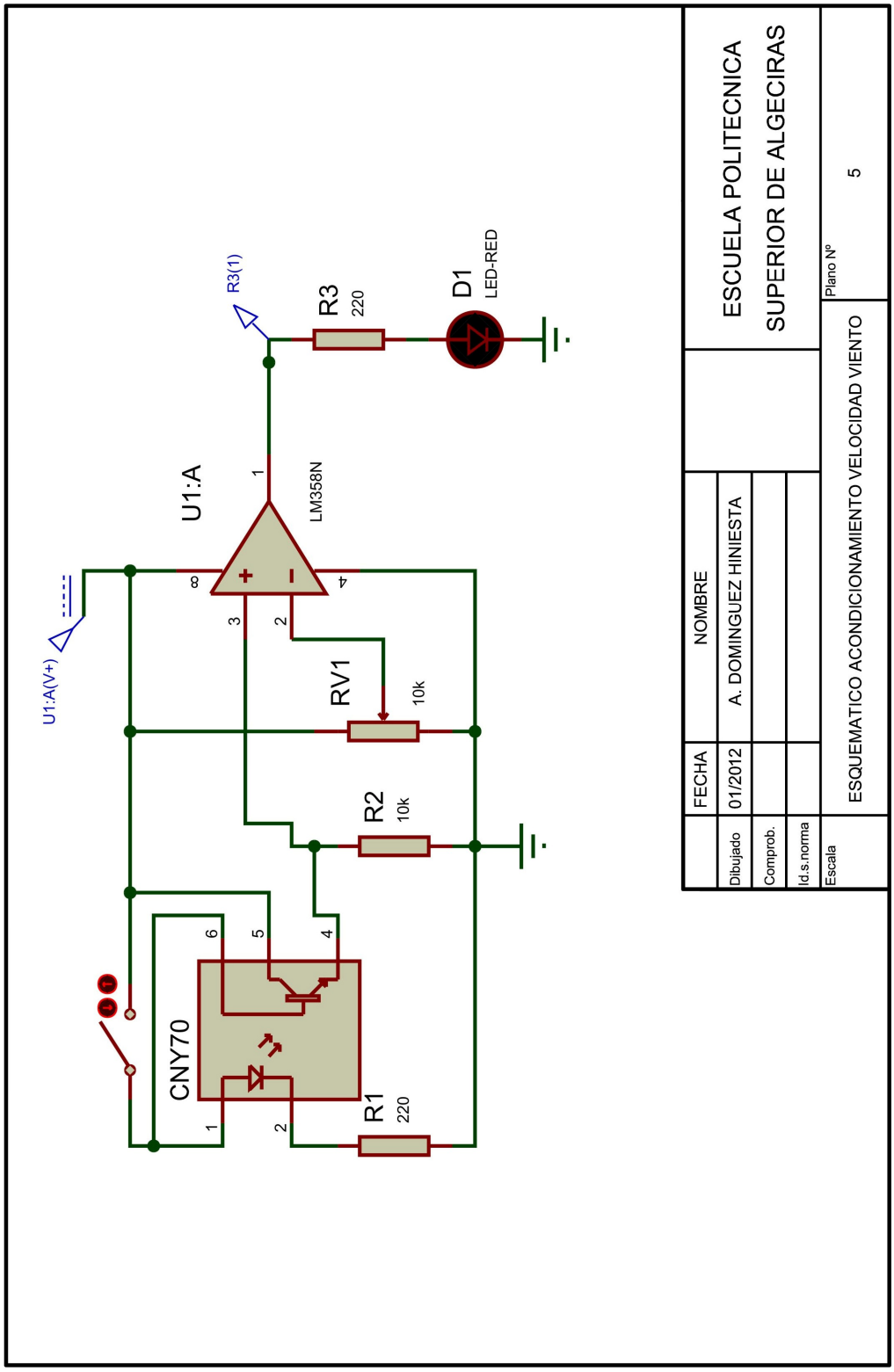
	FECHA	NOMBRE		ESCUELA POLITECNICA SUPERIOR DE ALGECIRAS
Dibujado	01/2012	A. DOMINGUEZ HINIESTA		
Comprob.				
Id.s.norma				
Escala	CONEXIONADO BASICO PIC16F877			Plano Nº 1

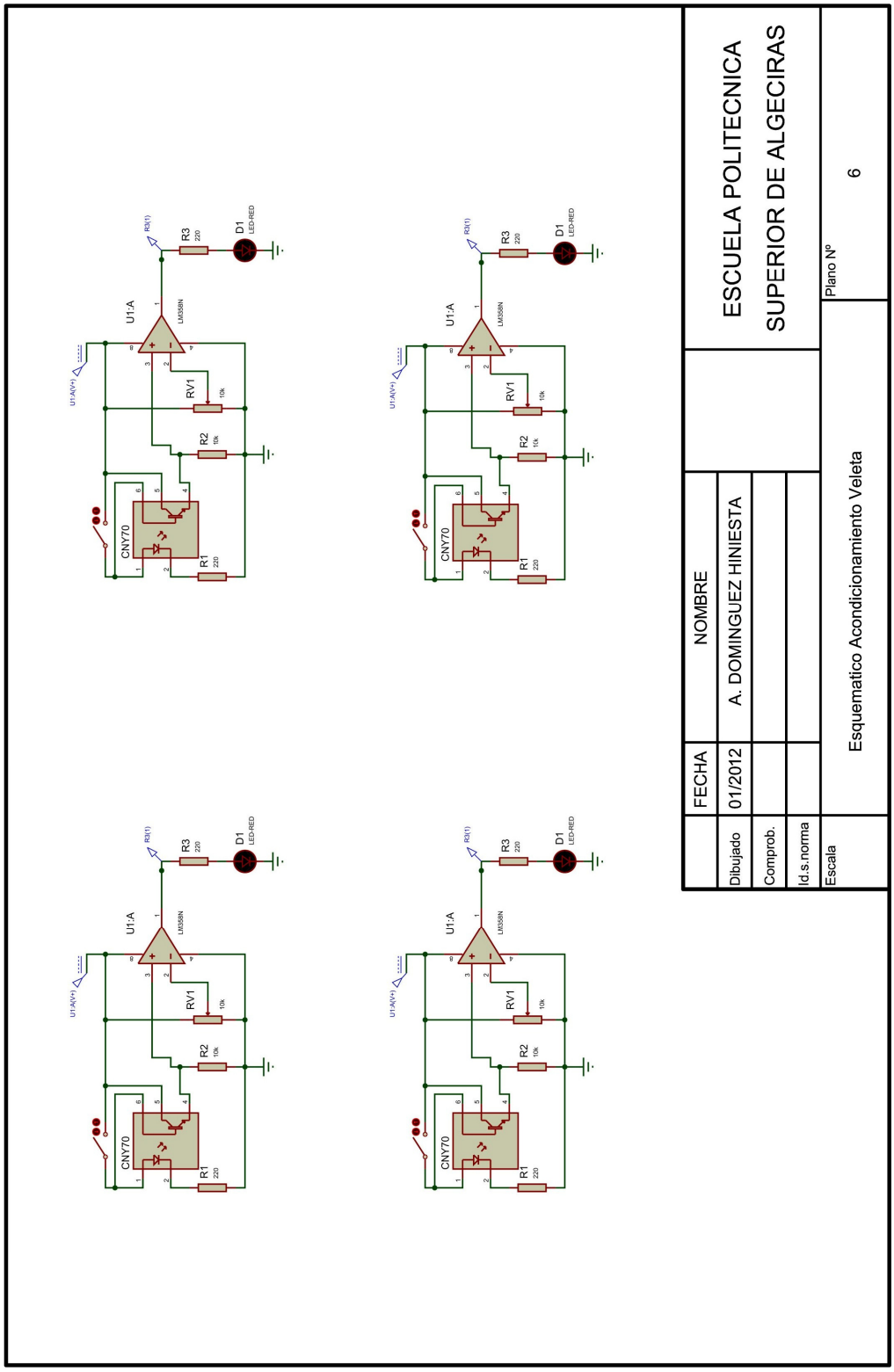

Vout
Vout
U2:A
LM358N
R2 5.6k
R1 1k
U1
LM35
VOUT
60.0
Vcc
VALUE=5
U2:A(V+)
Vlm35
U1(-VS)
VALUE=150m
ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
A. DOMINGUEZ HINIESTA
FECHA
01/2012
Dibujado
Comprob.
Id.s.norma
Escala
ESQUEMATICO ACONDICIONAMIENTO TEMPERATURA
Plano Nº
2

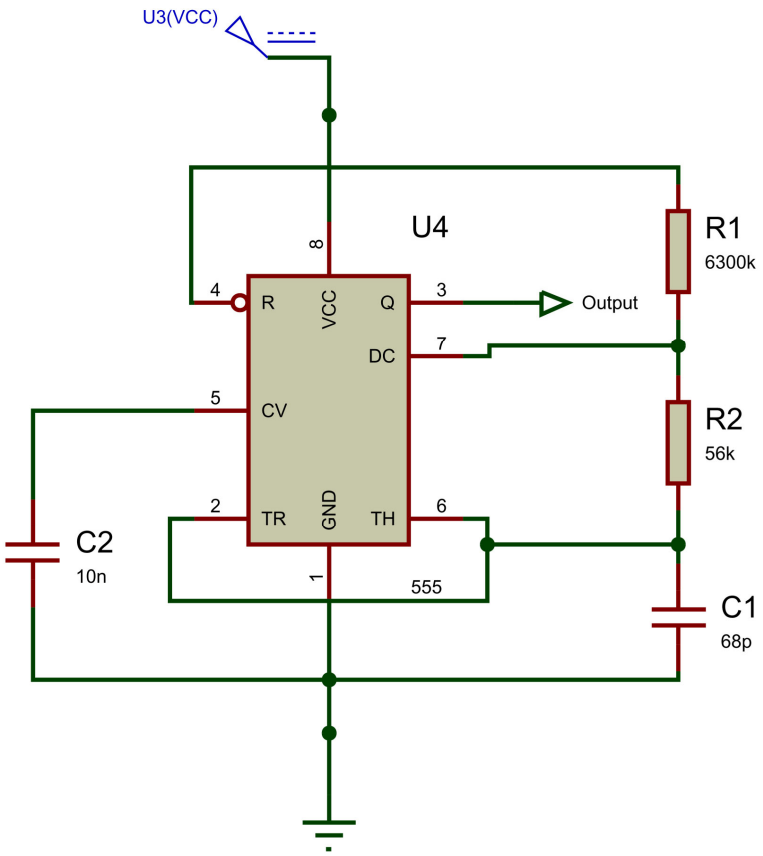


	FECHA	NOMBRE		ESCUELA POLITECNICA SUPERIOR DE ALGECIRAS
Dibujado	01/2012	A. DOMINGUEZ HINIESTA		
Comprob.				
Id.s.norma				
Escala	ESQUEMATICO ACONDICIONAMIENTO RADIACION			Plano Nº 3

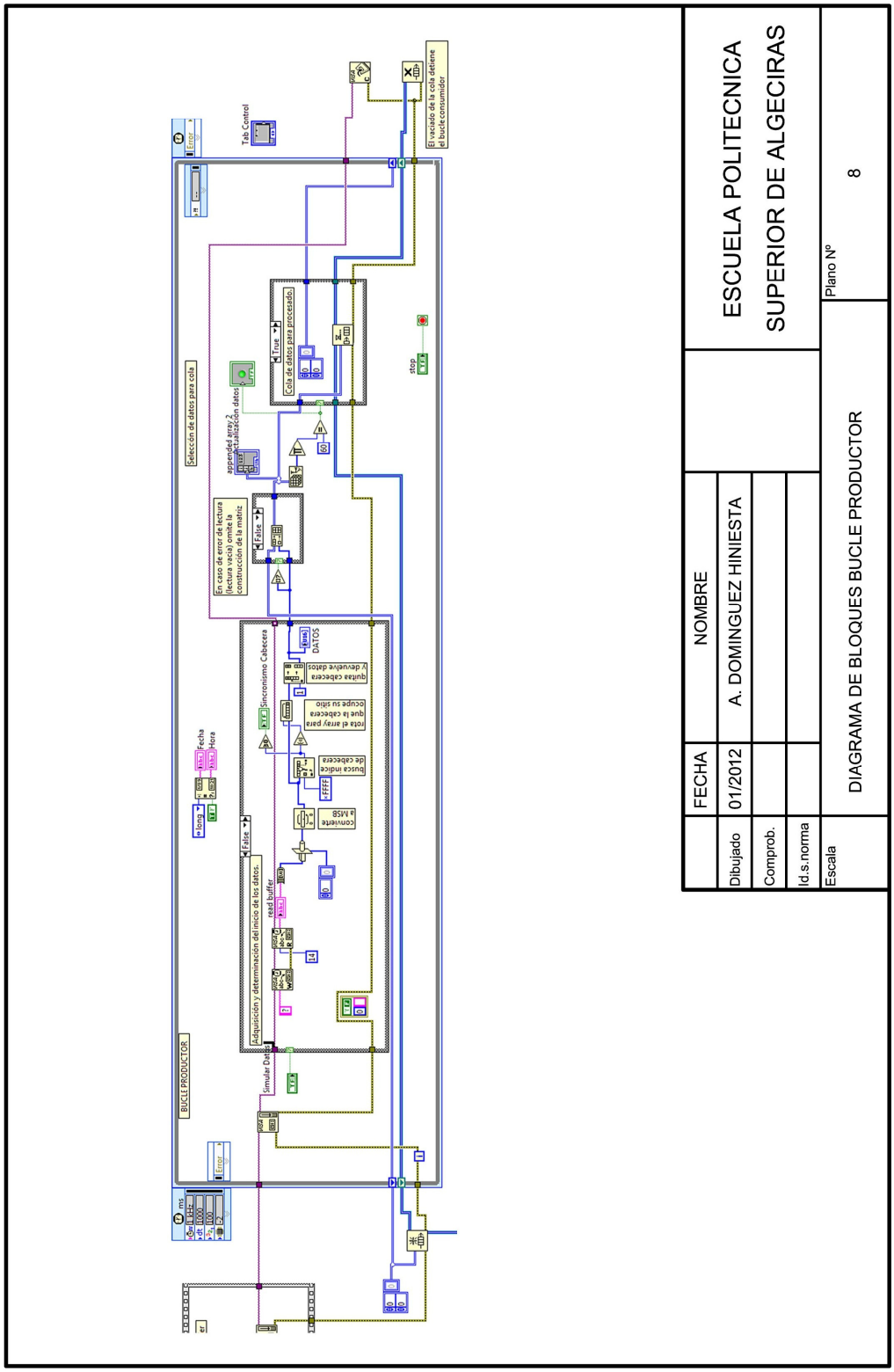

Vout
Vout
U1B
LM358N
R1
576
R2
100
Va
U1A
LM358N
U1A(V+)
U1A(V-)
R3
1k
R4
1k
R5
1k
R6
1k
Vin
M1
MPX4115
M1(V+)
108.0
Vk
C2
100p
Vk
VALUE=3.67256
ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
A. DOMINGUEZ HINIESTA
FECHA
01/2012
Dibujado
Comprob.
Id.s.norma
Escala
ESQUEMATICO ACONDICIONAMIENTO PRESION ATM.
Plano Nº
4


R3(1)
R3
220
D1
LED-RED
U1:A
LM358N
U1:A(V+)
RV1
10k
R2
10k
CNY70
R1
220
ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
FECHA
Dibujado
01/2012
A. DOMINGUEZ HINIESTA
Comprob.
Id.s.norma
Escala
ESQUEMATICO ACONDICIONAMIENTO VELOCIDAD VIENTO
Plano Nº
5

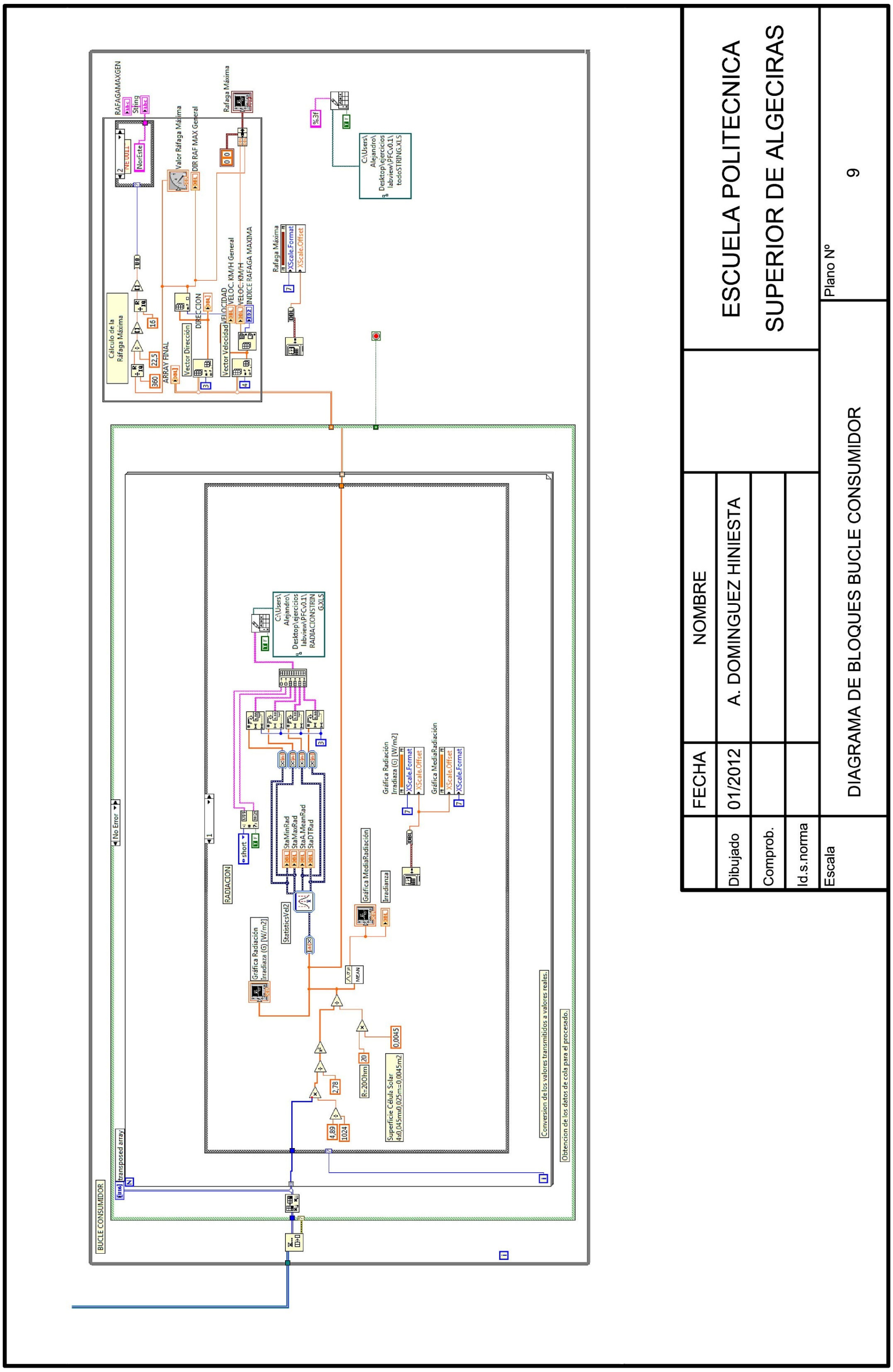

ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
FECHA
Dibujado
01/2012
A. DOMINGUEZ HINIESTA
Comprob.
Id.s.norma
Escala
Esquematico Acondicionamiento Veleta
Plano Nº
6
U1:A
LM358N
RV1
10k
R2
10k
R1
220
R3
220
D1
LED-RED
CNY70

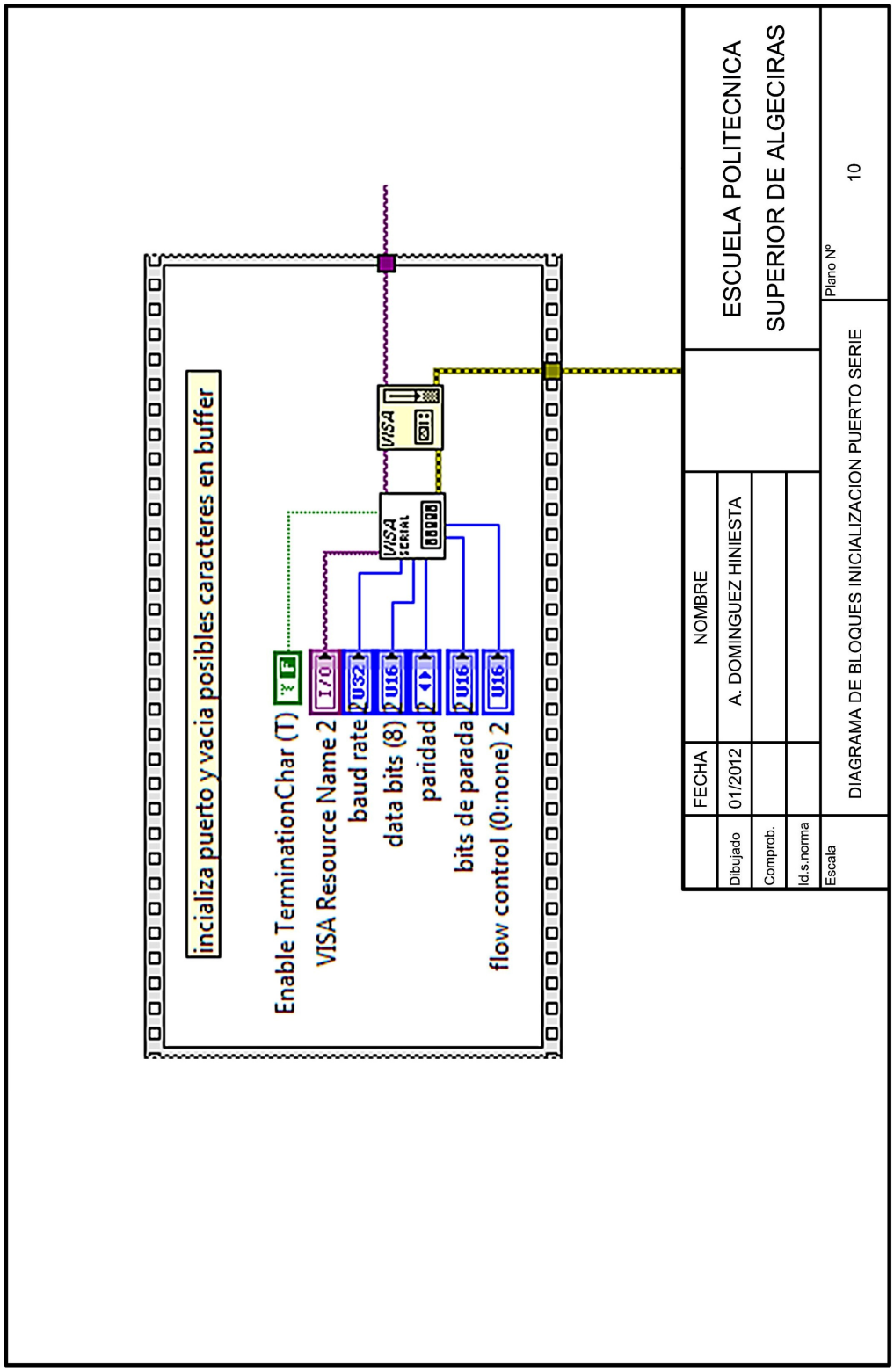


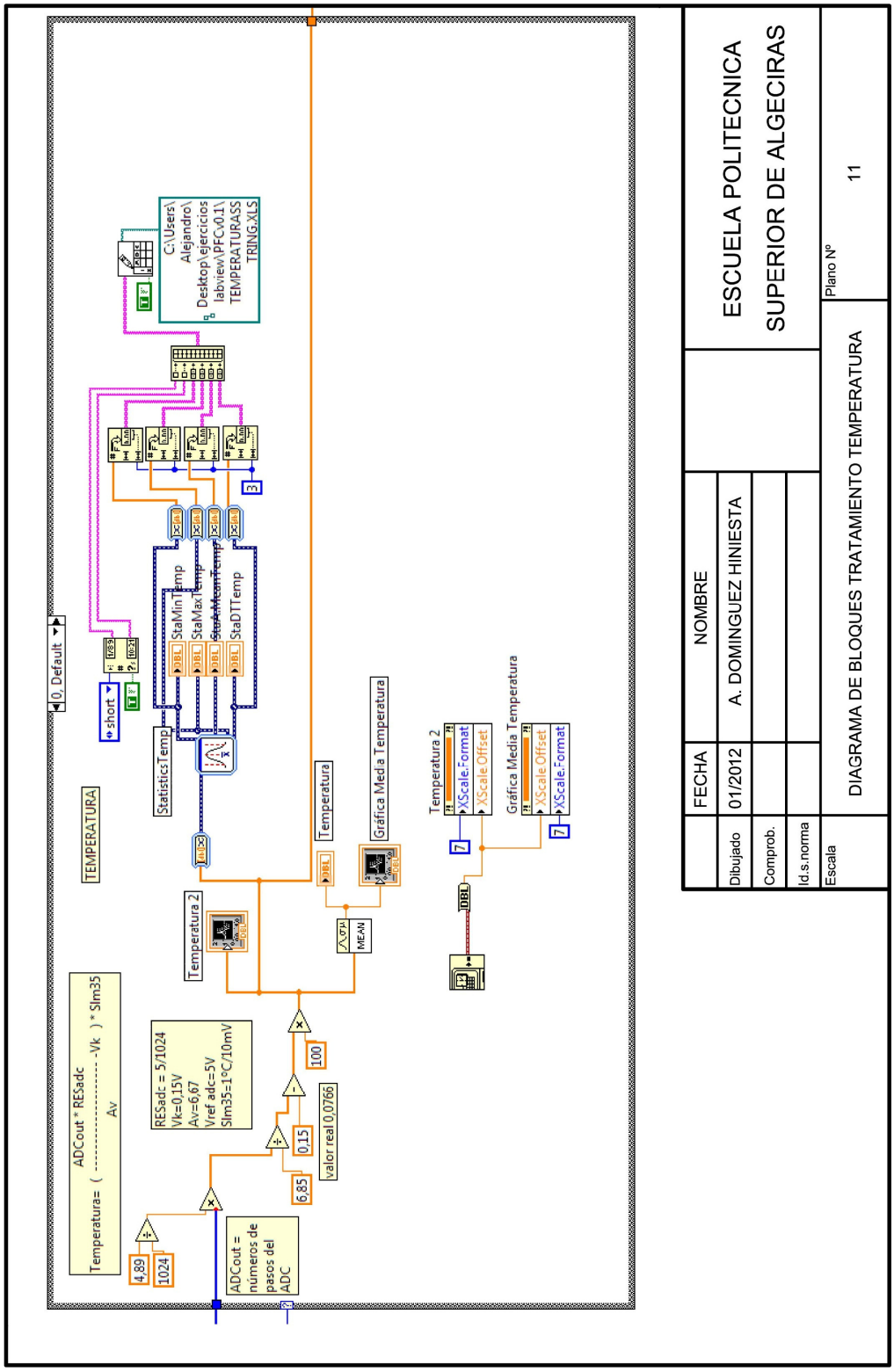
	FECHA	NOMBRE		ESCUELA POLITECNICA SUPERIOR DE ALGECIRAS
Dibujado	01/2012	A. DOMINGUEZ HINIESTA		
Comprob.				
Id.s.norma				
Escala	ESQUEMATICO ACONDICIONAMINETO HUMEDAD			Plano Nº 7

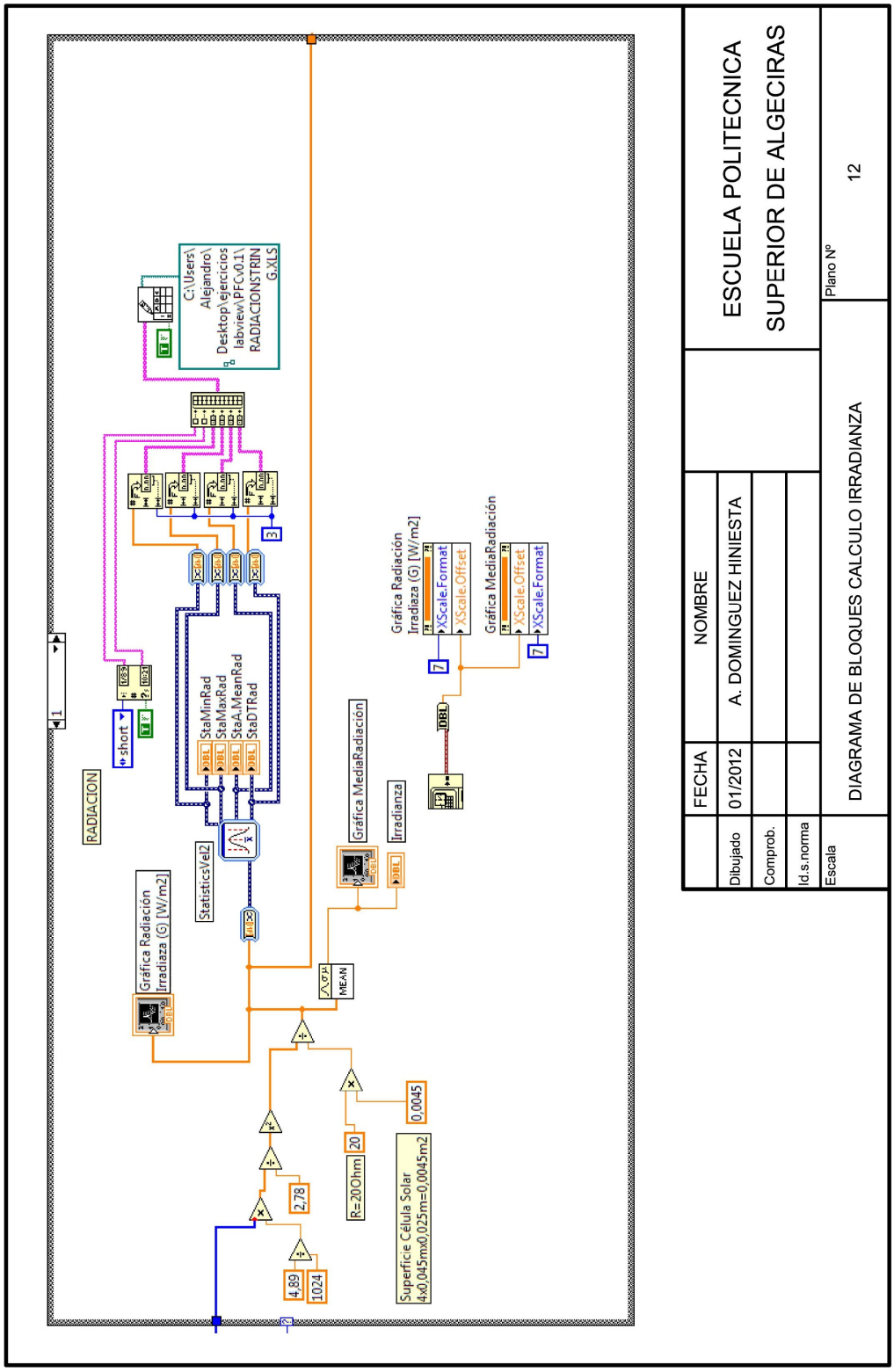


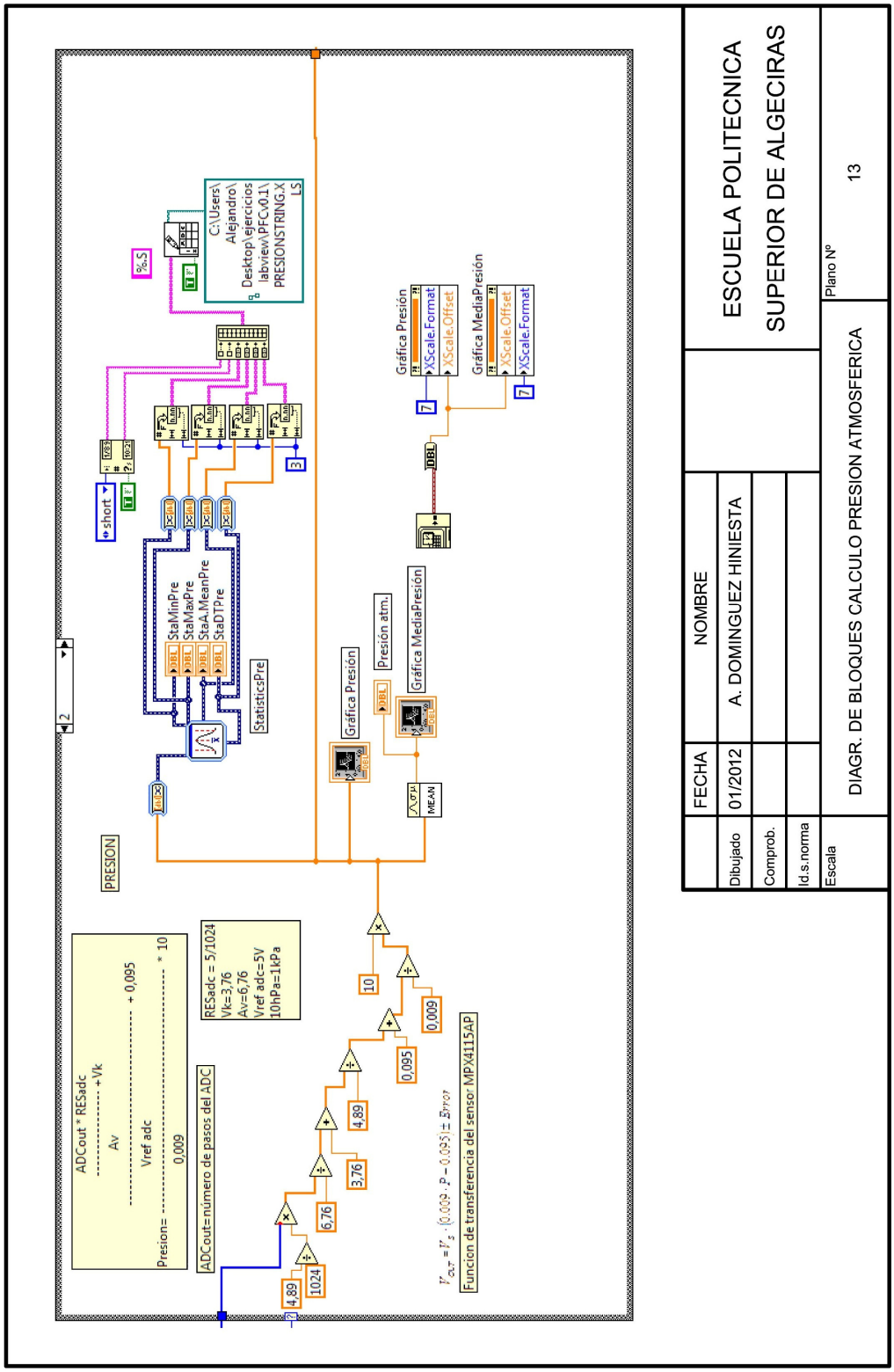
	FECHA	NOMBRE	ESCUELA POLITECNICA SUPERIOR DE ALGECIRAS
Dibujado	01/2012	A. DOMINGUEZ HINIESTA	
Comprob.			
Id.s.norma			
Escala	DIAGRAMA DE BLOQUES BUCLE PRODUCTOR		Plano Nº 8

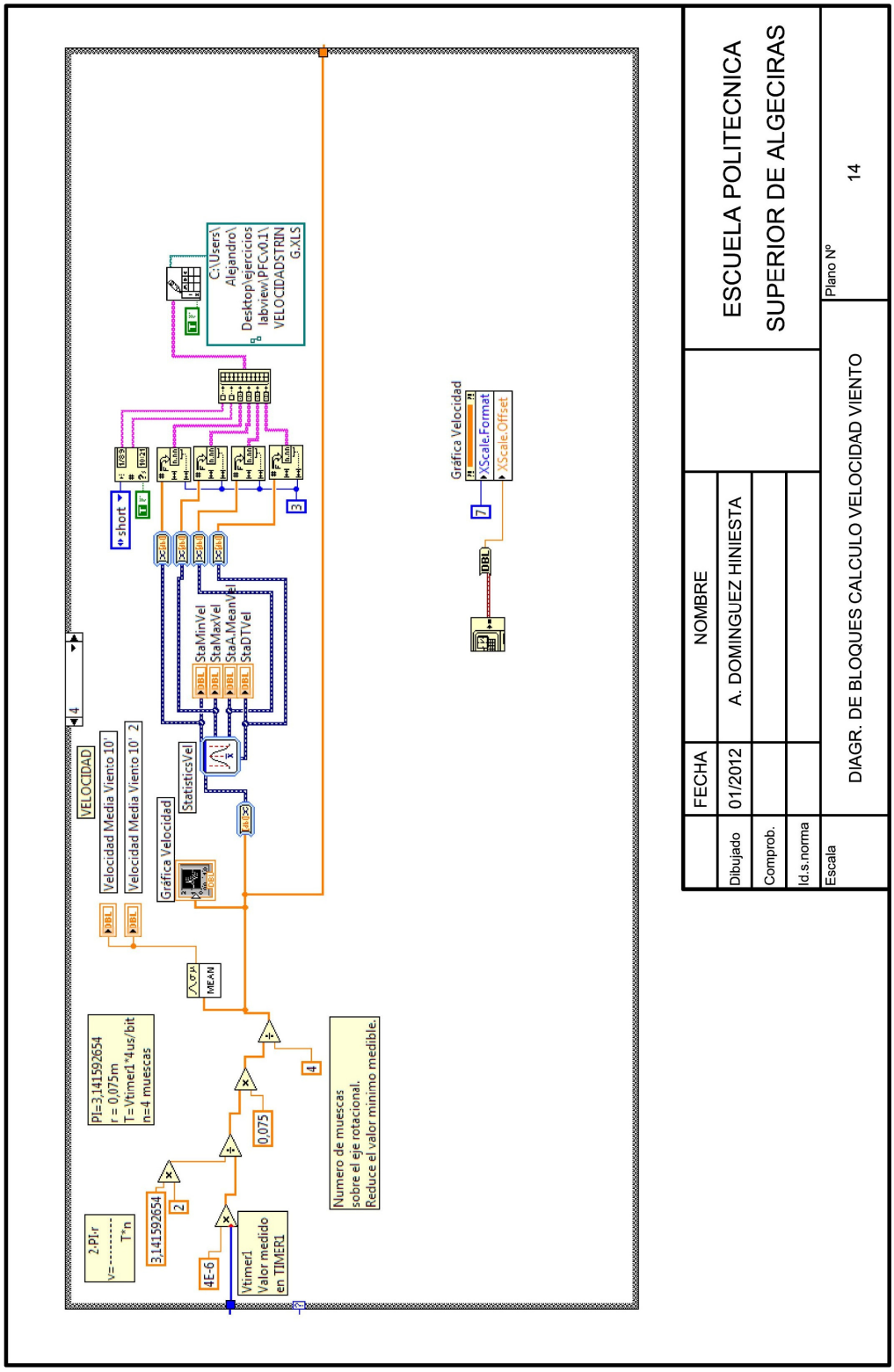

ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
FECHA
NOMBRE
Dibujado
01/2012
A. DOMINGUEZ HINIESTA
Comprob.
Id.s.norma
Escala
DIAGRAMA DE BLOQUES BUCLE CONSUMIDOR
Plano Nº
9

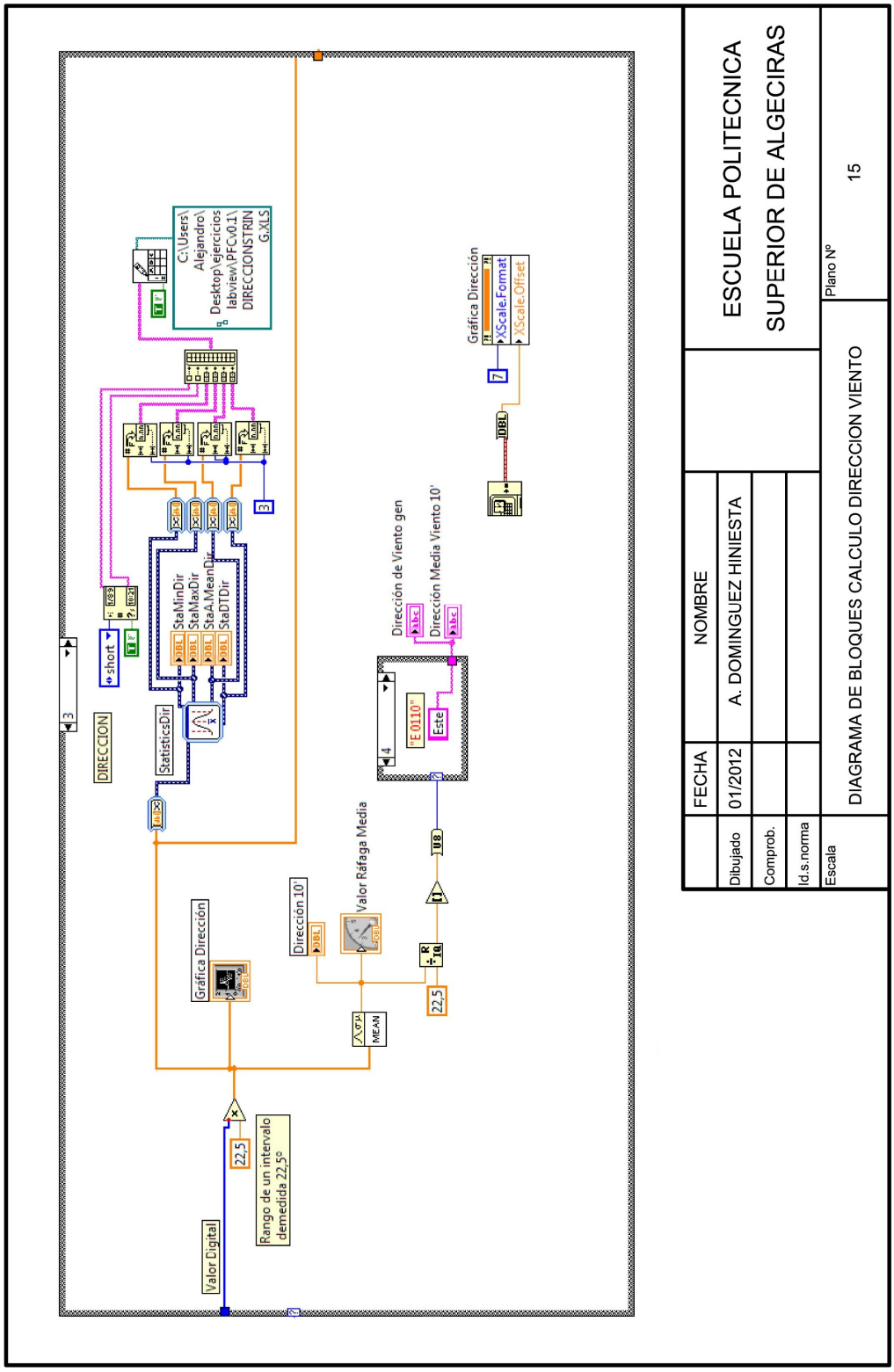

incializa puerto y vacia posibles caracteres en buffer
Enable TerminationChar (T)
VISA Resource Name 2
baud rate
data bits (8)
paridad
bits de parada
flow control (0:none) 2
I/O
U32
U16
U16
U16
VISA
VISA SERIAL
ESCUELA POLITECNICA SUPERIOR DE ALGECIRAS
NOMBRE
FECHA
Dibujado
A. DOMINGUEZ HINIESTA
01/2012
Comprob.
Id.s.norma
Escala
DIAGRAMA DE BLOQUES INICIALIZACION PUERTO SERIE
Plano Nº
10

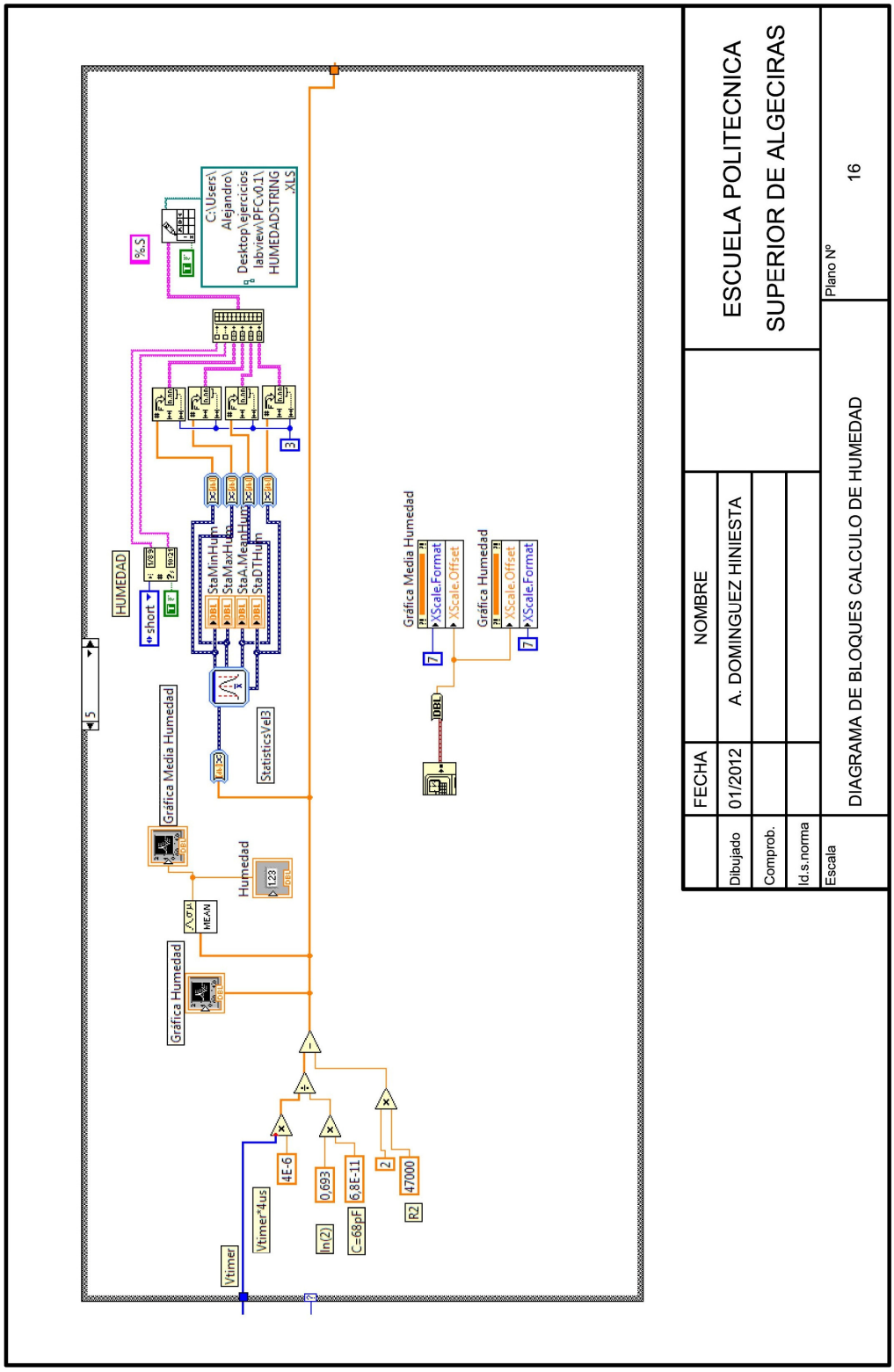

0, Default
TEMPERATURA
short
StatisticsTemp
StaMinTemp
StaMaxTemp
StaAvTemp
StaDTTemp
C:\Users\Alejandro\Desktop\ejercicios labview\PFC\v0.1\TEMPERATURASSTRING.XLS
Temperatura
Temperatura 2
Gráfica Media Temperatura
MEAN
Temperatura= (ADCout * RESadc / Av) - Vk) * Slm35
RESadc = 5/1024
Vk=0,15V
Av=6,67
Vref adc=5V
Slm35=1ºC/10mV
ADCout = números de pasos del ADC
4,89
1024
6,85
0,15
valor real 0,0766
100
XScale.Format
XScale.Offset
DBL
ESCUELA POLITECNICA SUPERIOR DE ALGECIRAS
NOMBRE
A. DOMINGUEZ HINIESTA
FECHA
01/2012
Dibujado
Comprob.
Id.s.norma
Escala
DIAGRAMA DE BLOQUES TRATAMIENTO TEMPERATURA
Plano Nº
11

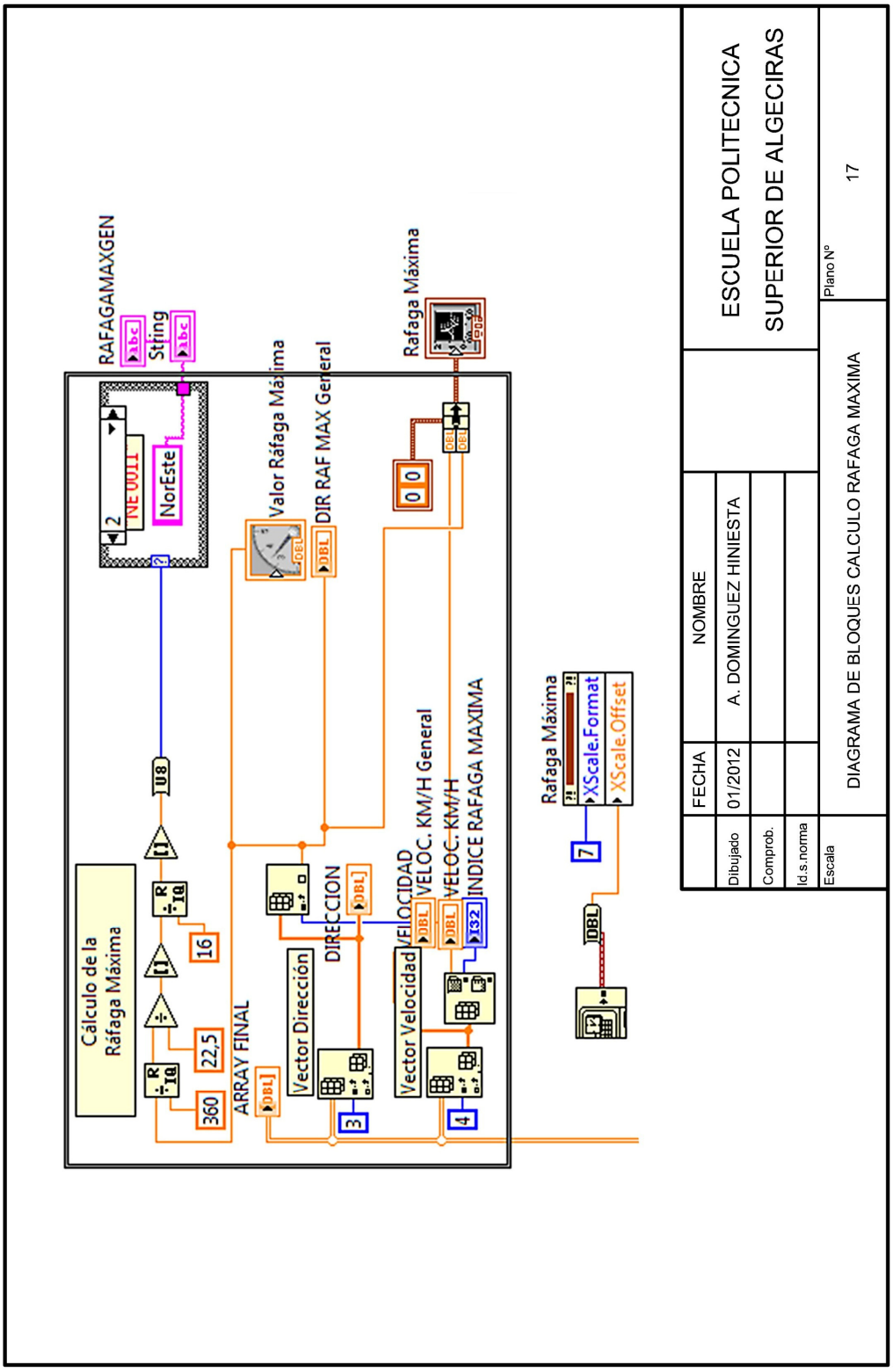

ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
A. DOMINGUEZ HINIESTA
FECHA
01/2012
Dibujado
Comprob.
Id.s.norma
Escala
DIAGRAMA DE BLOQUES CALCULO IRRADIANZA
Plano Nº
12
C:\Users\Alejandro\Desktop\ejercicios labview\PFC\v0.1\RADIACIONSTRIN G.XLS
RADIACION
short
StaMinRad
StaMaxRad
StaA.MeanRad
StaDTRad
StatisticsVel2
Gráfica Radiación Irradiaza (G) [W/m2]
Gráfica MediaRadiación
Irradianza
XScale.Format
XScale.Offset
MEAN
4,89
1024
2,78
R=20Ohm
20
0,0045
Superficie Célula Solar 4x0,045mx0,025m=0,0045m2

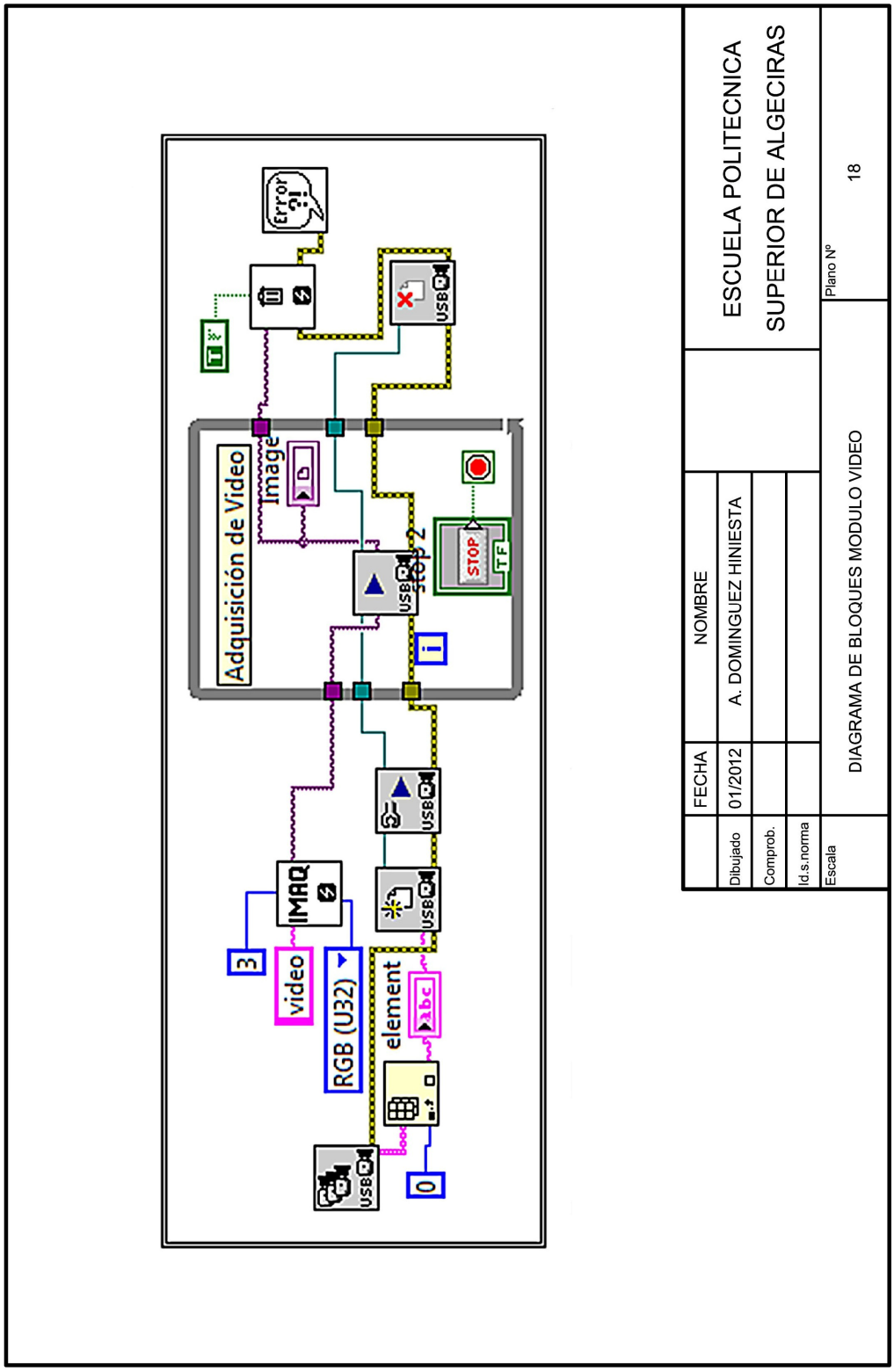

ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
A. DOMINGUEZ HINIESTA
FECHA
01/2012
Dibujado
Comprob.
Id.s.norma
Escala
DIAGR. DE BLOQUES CALCULO PRESION ATMOSFERICA
Plano Nº
13
PRESION
ADCout * RESadc
Av
Vref adc
Presion= 0,009 * 10
+ 0,095
+ Vk
ADCout=número de pasos del ADC
RESadc = 5/1024
Vk=3,76
Av=6,76
Vref adc=5V
10hPa=1kPa
Funcion de transferencia del sensor MPX4115AP
StaMinPre
StaMaxPre
StaA.MeanPre
StaDTPre
StatisticsPre
Gráfica Presión
Presión atm.
Gráfica MediaPresión
MEAN
short
%.5
C:\Users\Alejandro\Desktop\ejercicios labview\PFC v0.1\PRESIONSTRING.XLS
XScale.Format
XScale.Offset
DBL
4,89
1024
6,76
3,76
4,89
0,095
0,009
10

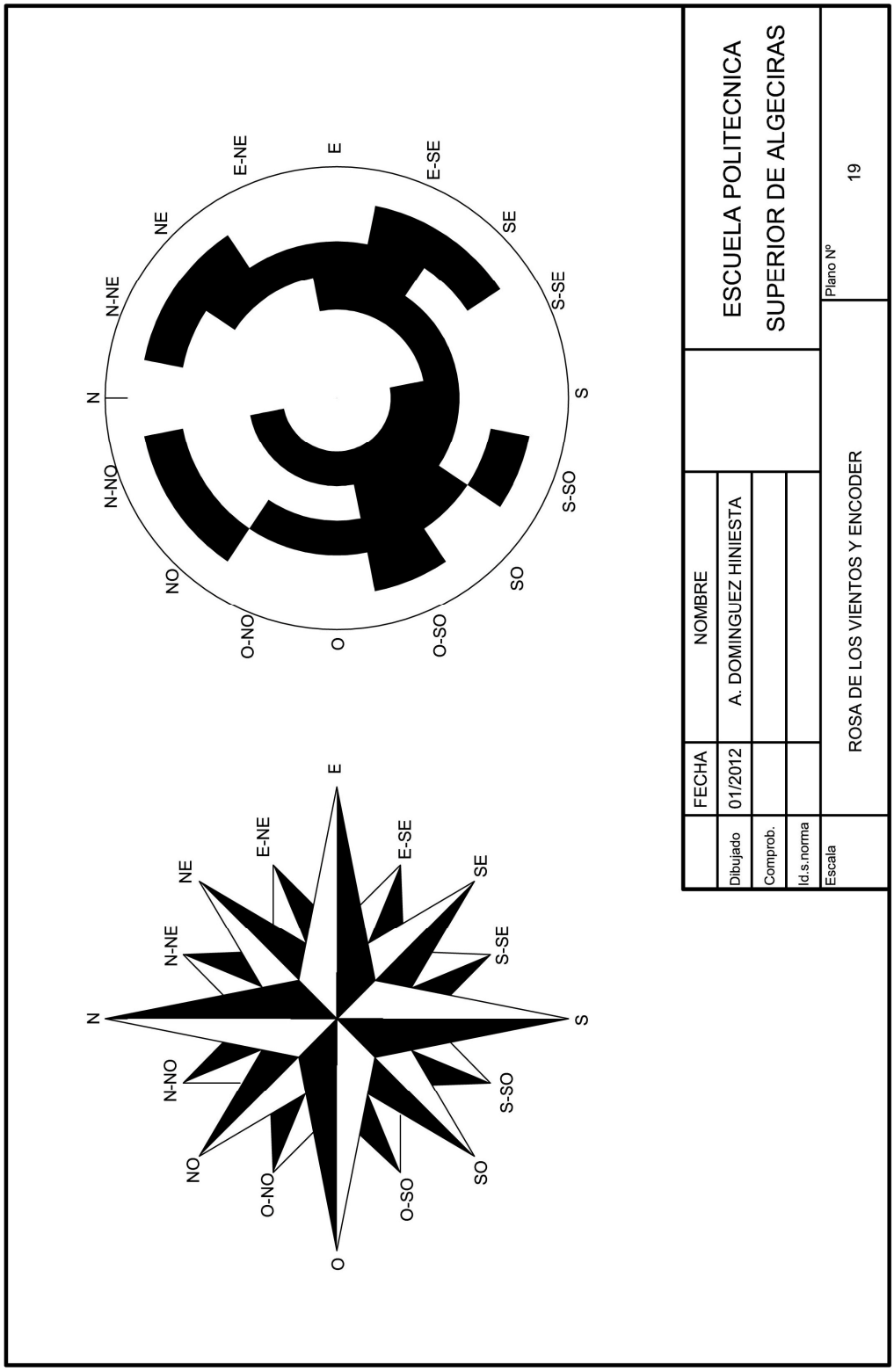

ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
FECHA
Dibujado
01/2012
A. DOMINGUEZ HINIESTA
Comprob.
Id.s.norma
Escala
DIAGR. DE BLOQUES CALCULO VELOCIDAD VIENTO
Plano Nº
14
VELOCIDAD
Velocidad Media Viento 10'
Velocidad Media Viento 10' 2
Gráfica Velocidad
StatisticsVel
StaMinVel
StaMaxVel
StaA.MeanVel
StaDTVel
Gráfica Velocidad
XScale.Format
XScale.Offset
short
C:\Users\Alejandro\Desktop\ejercicios labview\PFC\v01\VELOCIDADSTRING.XLS
PI=3,141592654
r = 0,075m
T=Vtimer1*4us/bit
n=4 muescas
2·PI·r
V= ---------
T*n
3,141592654
2
4E-6
Vtimer1
Valor medido en TIMER1
0,075
4
Numero de muescas sobre el eje rotacional.
Reduce el valor minimo medible.
MEAN


ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
A. DOMINGUEZ HINIESTA
FECHA
01/2012
Dibujado
Comprob.
Id.s.norma
Escala
DIAGRAMA DE BLOQUES CALCULO DIRECCION VIENTO
Plano Nº
15
C:\Users\Alejandro\Desktop\ejercicios labview\PFCv01\DIRECCIONSTRING.XLS
Gráfica Dirección
XScale.Format
XScale.Offset
DIRECCION
short
StaMinDir
StaMaxDir
StaA.MeanDir
StaDTDir
StatisticsDir
Dirección de Viento gen
Dirección Media Viento 10'
"E 0110"
Este
Dirección 10'
Valor Ráfaga Media
22,5
Rango de un intervalo demedida 22,5º
Valor Digital


ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
A. DOMINGUEZ HINIESTA
FECHA
01/2012
Dibujado
Comprob.
Id.s.norma
Escala
DIAGRAMA DE BLOQUES CALCULO DE HUMEDAD
Plano Nº
16
HUMEDAD
short
Gráfica Media Humedad
Gráfica Humedad
Humedad
StatisticsVel3
Vtimer
Vtimer*4us
4E-6
0,693
ln(2)
6,8E-11
C=68pF
2
47000
R2


RAFAGAMAXGEN
String
NorEste
Valor Ráfaga Máxima
DIR RAF MAX General
Rafaga Máxima
Cálculo de la Ráfaga Máxima
ARRAY FINAL
Vector Dirección
DIRECCION
Vector Velocidad
VELOCIDAD
VELOC. KM/H General
VELOC. KM/H
INDICE RAFAGA MAXIMA
Rafaga Máxima
XScale.Format
XScale.Offset
ESCUELA POLITECNICA SUPERIOR DE ALGECIRAS
NOMBRE
FECHA
Dibujado
01/2012
A. DOMINGUEZ HINIESTA
Comprob.
Id.s.norma
Escala
DIAGRAMA DE BLOQUES CALCULO RAFAGA MAXIMA
Plano Nº
17


Adquisición de Video
Image
STOP
TF
stop?
i
IMAQ
video
RGB (U32)
element
abc
3
0
ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
A. DOMINGUEZ HINIESTA
FECHA
01/2012
Dibujado
Comprob.
Id.s.norma
Escala
DIAGRAMA DE BLOQUES MODULO VIDEO
Plano Nº
18


N
N-NE
NE
E-NE
E
E-SE
SE
S-SE
S
S-SO
SO
O-SO
O
O-NO
NO
N-NO
N
N-NE
NE
E-NE
E
E-SE
SE
S-SE
S
S-SO
SO
O-SO
O
O-NO
NO
N-NO
ESCUELA POLITECNICA
SUPERIOR DE ALGECIRAS
NOMBRE
FECHA
Dibujado
01/2012
A. DOMINGUEZ HINIESTA
Comprob.
Id.s.norma
Escala
ROSA DE LOS VIENTOS Y ENCODER
Plano Nº
19

4 CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO

4.1 Conclusiones.

El objetivo principal y personal en la realización del presente proyecto final de carrera, pretendía el desarrollo de un sistema de adquisición de datos para magnitudes meteorológicas basados en dos tecnologías desconocidas para mí, la programación de microcontroladores y la programación bajo un entorno gráfico.

Aunque a nivel práctico, por su facilidad de aprendizaje y por la documentación existente al respecto habría sido más fácil desarrollar la programación del PIC bajo lenguaje C, ENSAMBLADOR ofrece mayores posibilidades de aprendizaje respecto a la tecnología de los microprocesadores. Al tratarse de un lenguaje de bajo nivel en el que las instrucciones están representadas por nemotécnicos, requiere de la comprensión de los procesos a realizar con un mayor detalle.

En cuanto a la programación en entorno gráfico, LabView representa una potente herramienta dentro del marco de la industria y la investigación. Sus aplicaciones se extienden desde bancos de pruebas y de trabajos en laboratorios hasta aplicaciones de uso o diagnostico a pie de campo, generalmente desarrolladas por el propio usuario quien diseña su herramienta conforme a sus necesidades especificas.

Poder haber aprendido las técnicas que me han permitido alcanzar los objetivos determinados para el desarrollo de este proyecto, constituye el mayor logro de los establecidos previos al inicio del mismo.

Aunque la exactitud del sistema no suponía uno de los retos del proyecto, esta se adecua bastante bien a los de otras estaciones meteorológicas comerciales, por lo que se consideran resultados óptimos para el diseño de un prototipo de muy bajo coste cuyo objetivo pretendía la propuesta de una alternativa basada en el conocimiento de los principios básicos de funcionamiento de un sensor meteorológico, como justificación de los conocimientos adquiridos.

4.2 Propuestas de mejora.

Las propuestas de mejoras pueden afrontarse de muy diferentes formas. Todo proyecto es susceptible de ser mejorado, máxime aún cuando se trata un trabajo con fines académicos y no comerciales que puedan verse limitados por los rendimientos económicos.

Considerando la vertiente tecnológica del hardware, existen ciertas posibilidades de mejoras como el desarrollo de la comunicación via bluethooth, wifi, GSM o satélite, en lugar de realizarla por puerto serie, así como dotarlo de un sistema de alimentación basado en energías renovables como la solar o la eólica que permitirían independizarlo de cualquier red de alimentación de energía. El conjunto de estas dos mejoras permitiría su evolución hacia el desarrollo de un sistema embebido autónomo capaz de ser instalado en cualquier ubicación sin necesidad de mantener un lazo físico de unión entre el puesto de control y la tarjeta de adquisición.

Atendiendo a la vertiente software, la posibilidad de programar el PIC en otros lenguajes más versátiles o el empleo de otros lenguajes de distribución gratuita para el desarrollo de la interfaz de usuario, constituyen las principales propuestas de mejoras para este proyecto.

Por otro lado, existen mejoras respecto a las magnitudes a medir. La incorporación de magnitudes que determinen la calidad del aire, permitiría su uso para la detección de vertidos tóxicos a la atmosfera en entornos industriales, control del la calidad de aire en ciudades, túneles y minas, así como aplicaciones de seguridad y confort en domótica.

5 BIBLIOGRAFÍA Y DIRECCIONES WEB

5.1 Bibliografía

MICROCONTROLADOR PIC16F84 DESARROLLO DE PROYECTOS 2ª EDICIÓN.
Enrique Palacios- Fernando Remiro – Lucas J. López.
Editorial RA-MA

MICROCONTROLADORES: FUNDAMENTOS Y APLICACIONES CON PIC.
Ramón Pallás Areny / Fernando E. Valdés Pérez
Editorial MARCOMBO

MICROCONTROLADORES PIC: DISEÑO PRÁCTICO DE APLICACIONES 1ª PARTE (3ª ED.)
Ignacio Angulo Martínez y José Mª Angulo Usategui.
Editorial MCGRAW-HILL

MICROCONTROLADORES PIC: DISEÑO PRACTICO DE APLICACIONES (2ª PARTE): PIC16F87X
José María Angulo Usategui, Susana Romero Yesa e Ignacio Angulo Martínez.
Editorial MCGRAW-HILL

LABVIEW BASICS I: INTRODUCTION COURSE MANUAL.
NATIONAL INSTRUMENTS

LABVIEW BASICS II: DEVELOPMENT COURSE MANUAL.
NATIONAL INSTRUMENTS

LABVIEW ENTORNO GRÁFICO DE PROGRAMACIÓN. (2ª EDICIÓN).
José Rafael Lajara Vizcaíno y José Pelegrí Sebastiá.
Editorial MARCOMBO

5.2 Documentos de consulta

Datasheet PIC16F877/A

Datasheet MAX232

Datasheet LM35

Datasheet CNY70

Datasheet MPX4115AP

Datasheet HCZ-D5

Datasheet CÉLULA SOLAR C-0120/C-0123

Datasheet 7805

Datasheet 7808

Datasheet 7810

Datasheet LM741

Datasheet LM358

5.3 Direcciones WEB

http://www.microchip.com

http://www.ni.com

http://www.datasheetcatalog.net/

http://forums.ni.com

https://www.onsemi.com/

http://www.todopic.net/foros

http:/www.forosdeelectronica.com

http://www.fadisel.es

http://www.meteosevilla.com

http://feng3.nobody.jp/en/pg5v2.html

6 ANEXOS

6.1 Datasheets

Datasheet LM35

Datasheet CNY70

Datasheet MPX4115AP

Datasheet HCZ-D5

Datasheet CÉLULA SOLAR C-0120/C-0123

Datasheet MAX232