# ESCUELA POLITÉCNICA NACIONAL

## FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

## DISEÑO, CONSTRUCCIÓN E IMPLEMENTACIÓN DE UN SISTEMA DE TELEMETRÍA PARA UN VEHÍCULO DE KARTING

**PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES**

**JOSÉ DANIEL MORALES BEJARANO**
**jdannymoralesb@gmail.com**

**DAVID GONZALO VARGAS BURGOS**
**david.vargas.burgos@gmail.com**

**DIRECTOR: ING. EDWIN GUILLERMO NIETO RIOS**
**edwin.nieto@epn.edu.ec**

**Quito, agosto 2012**

# DECLARACIÓN

Nosotros, David Gonzalo Vargas Burgos y José Daniel Morales Bejarano declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

David Gonzalo Vargas Burgos José Daniel Morales Bejarano

# CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por David Gonzalo Vargas Burgos y José Daniel Morales Bejarano, bajo mi supervisión.

Ing. Edwin Nieto
DIRECTOR DEL PROYECTO

# DEDICATORIA

Agradezco a Dios por haberme permitido llegar a culminar esta etapa importante y por haber siempre bendecido mi camino.

Quiero dedicar este trabajo a mis padres Luis y Marcia, quienes con su esfuerzo me han apoyado y guiado a lo largo de mi vida. La culminación de este proyecto es mi manera de agradecerles por ello.

Le dedico de igual manera este trabajo a mi esposa por haber estado a mi lado y con su amor motivarme y ayudarme a continuar en los momentos que las cosas se presentaban difíciles.

Deseo agradecer a los padres de mi esposa Víctor y Marcela por su preocupación y apoyo para que pueda culminar este proyecto.

También quiero agradecer a mis mejores amigos Marco Oquendo, Daniel Magno,y Fernando Navarrete por su ayuda especialmente en la etapa de pruebas del proyecto.

En especial deseo agradecer a Daniel Morales porque gracias a la dedicación y entrega al proyecto pudimos obtener el mejor resultado.

David G. Vargas Burgos

# DEDICATORIA

Antes que nada deseo agradecer a Dios, ya que con su mano invisible ha corregido mi camino cuando lo he necesitado, y me ha permitido enfrentar cada uno de los retos que hacen ahora de mi lo que soy.

A mis padres, quienes me guiaron y protegieron desde pequeño, y ahora puedo compartir con ellos este título.

A mis hermanos, quienes me han respaldado en cada paso que he dado.

A Gabriel Morales, mi hijo, quien es la razón más grande para superarme día a día y a su mamá Mily Ramírez, quien desde que nos conocimos ha estado a mi lado.

A David Vargas, mi compañero de tesis, con quien hemos superado esta etapa de nuestras vidas, y ha sido siempre un apoyo no solo en este proyecto.

Al Ingeniero Edwin Nieto, Director del proyecto de titulación por sus grandiosos consejos y su valioso tiempo entregado al proyecto.

José Daniel Morales

# ÍNDICE DE CONTENIDOS

**CAPITULO 1** ..... **12**

**1. INTRODUCCION Y ANTECEDENTES** ..... 12

***1.1. TELEMETRÍA*** ..... ***12***

**1.1.1 LA TELEMETRÍA EN COMPETENCIAS DEPORTIVAS** ..... 12

**1.1.2 TRANSMISIÓN Y PROCESAMIENTO DE LOS DATOS GENERADOS POR EL VEHÍCULO** ..... 13

**1.1.3 DISEÑO E INSTALACIÓN A MEDIDA DE LOS REQUERIMIENTOS** ..... 14

***1.2 EL VEHÍCULO DE KARTING*** ..... ***15***

**1.2.1 CHASIS** ..... 15

**1.2.2 BANDEJA** ..... 15

**1.2.3 BARRAS** ..... 16

**1.2.4 DIRECCIÓN** ..... 16

**1.2.5 EJE POSTERIOR** ..... 17

**1.2.6 FRENOS** ..... 17

**1.2.7 MOTOR** ..... 18

**1.2.7.1 Motor de dos tiempos** ..... 18

*1.2.7.1.1 Primer tiempo* ..... 19

*1.2.7.1.2 Segundo tiempo* ..... 19

*1.2.7.1.3 Sistema de Ignición* ..... 20

*1.2.7.1.4 Bujía* ..... 20

*1.2.7.1.5 Bobina* ..... 21

*1.2.7.1.6 Tanque de combustible* ..... 22

*1.2.7.1.7 Filtro de Admisión* ..... 22

*1.2.7.1.8 Carburador* ..... 22

*1.2.7.1.9 Transmisión* ..... 23

*1.2.7.1.10 Piñón* ..... 23

*1.2.7.1.11 Cadena* ..... 23

*1.2.7.1.12 Corona* ..... 23

***1.3 SISTEMAS TELEMÉTRICOS EXISTENTES EN EL MERCADO*** ..... ***24***

**1.3.1 SOLUCIONES DE SOFTWARE** ..... 25

1.3.1.1 **Software “TELEMETRIA. F1 Challenge 99-02”** ..... 25

**1.3.1.2 Software “u-Lab y u-Graph” de KMT - Messtechnik & Telemetrie [4]** ..... 26

**1.3.1.3 Soluciones de hardware** ..... 26

**1.3.1.4 Línea de productos de KMT - Messtechnik & Telemetrie** ..... 26

**1.3.1.5 Sistema de telemetría 4/8 canales para ruedas "CT4/8-Wheel"** ..... 26
**1.3.1.6 Medidor de revoluciones por minuto del motor "RPM-8000-PRO"** ..... 27
**1.3.1.7 Sistema de transmisión – recepción "CT2 mini"** ..... 28
**1.3.1.8 Sistema de transmisión – recepción TEL1/40k TTL** ..... 28

**CAPITULO 2** ..... **29**
**2 GENERALIDADES** ..... 29
***2.1. POSICIÓN DEL VEHÍCULO EN LA PISTA*** ..... ***29***
**2.1.1. SISTEMA GPS** ..... 29
**2.1.1.1. Introducción** ..... 29
**2.1.1.2. Estructura del sistema GPS** ..... 30
*2.1.1.2.1. Satélites* ..... 31
*2.1.1.2.2. Control terrestre* ..... 32
*2.1.1.2.3. Receptores* ..... 32
**2.1.2. FUNCIONAMIENTO DEL GPS** ..... 34
**2.1.2.1. Señales de navegación de los satélites** ..... 35
**2.1.3. TIPOS DE ENCENDIDO DE LOS RECEPTORES GPS** ..... 37
**2.1.3.1. Warm Start** ..... 37
**2.1.3.2. Cold Start** ..... 38
**2.1.3.3. Autolocate** ..... 38
**2.1.3.4. Search in the sky** ..... 38
**2.1.4. SEÑALES DEL GPS** ..... 38
**2.1.4.1. Coarse/Acquisition (C/A)** ..... 39
**2.1.4.2. Precise (P)** ..... 39
**2.1.5. FUENTES DE ERROR** ..... 39
***2.2. MICROPROCESADORES*** ..... ***40***
**2.2.1. MICROCONTROLADOR PIC** ..... 41
**2.2.1.1. Características del microcontrolador PIC** ..... 41
**2.2.2. MICROCONTROLADOR AVR** ..... 42
**2.2.2.1. Características del microcontrolador AVR** ..... 43
***2.3. COMUNICACIÓN POR RADIOFRECUENCIA*** ..... ***44***
**2.3.1. BANDAS DE FRECUENCIA DE USO LIBRE** ..... 44
**2.3.1.1. Modulación** ..... 44
*2.3.1.1.1. Modulación ASK (Amplitude Shift key)* ..... 45
*2.3.1.1.2. Modulación FSK (frequency shift keying)* ..... 46
*2.3.1.1.3. Modulación GFSK (Gaussian Frequency Shift Keying)* ..... 47
*2.3.1.1.4. Modulación PSK (Phase Shift Keying)* ..... 47

**CAPITULO 3..... 49**

**3. DISEÑO Y CONSTRUCCIÓN DEL SISTEMA TELEMETRICO** ..... 49

***3.1. CARACTERÍSTICAS DEL MÓDULO DE TELEMETRIA .....49***

***3.2. ELEMENTOS DEL SISTEMA DE TELEMETRIA .....50***

**3.2.1. MICROCONTROLADOR** ..... 50

**3.2.2. DISPOSITIVO DE COMUNICACIÓN INALAMBRICA** ..... 51

**3.2.2.1. Aplicaciones** ..... 52

**3.2.3. SISTEMA DE POSICIONAMIENTO** ..... 53

**3.2.4. HMI** ..... 55

**3.2.4.1. Software de control LABVIEW** ..... 55

**3.2.4.2. Programación** ..... 56

3.2.4.2.1. **Panel Frontal** ..... 56

3.2.4.2.2. **Diagrama de bloques** ..... 56

**3.2.4.3. Ventajas** ..... 57

**3.2.4.4. Aplicaciones** ..... 57

**3.2.4.5. Diseño de software** ..... 58

*3.2.4.5.1. Ventana Selección* ..... 58

*3.2.4.5.2. Nueva pista* ..... 59

*3.2.4.5.3. Nueva Carrera* ..... 62

*3.2.4.5.4. Carrera Guardada* ..... 64

**3.2.5. MEDICIÓN DE TEMPERATURA** ..... 66

**3.2.5.1. Sensor de temperatura DS18B20 1- Wire.** ..... 67

*3.2.5.1.1. Protocolo 1-WIRE* ..... 68

**3.2.5.2. Ventajas** ..... 68

**3.2.5.3. Comunicación a nivel de bits** ..... 69

**3.2.5.4. Utilización del sensor de temperatura DS18B20** ..... 70

**3.2.6. MEDICIÓN DE REVOLUCIONES POR MINUTO** ..... 72

**3.2.6.1. Conversor LM2917** ..... 73

**3.2.7. ACONDICIONAMIENTO DEL SENSOR DE R.P.M** ..... 75

**3.2.8. MEDICIÓN DE NIVEL DE COMBUSTIBLE** ..... 77

**3.2.8.1. Acondicionamiento del Sensor de nivel** ..... 79

**3.2.9. DISEÑO DEL TRANSCEPTOR EN PITS** ..... 80

**3.2.10. DISEÑO DE LA TRAMA DE DATOS** ..... 81

**3.2.11. CONSTRUCCIÓN DEL MÓDULO DE TELEMETRÍA.** ..... 82

**CAPITULO 4** ..... **85**

**4. PRUEBAS Y RESULTADOS** ..... 85

***4.1. PRUEBA DE FUNCIONAMIENTO DEL RECEPTOR GPS*** ..... ***85***

***4.2. PRUEBA DE FUNCIONAMIENTO DEL MEDIDOR DE LAS R.P.M DEL VEHÍCULO*** ..... ***88***

***4.3. PRUEBA DE FUNCIONAMIENTO DEL BLOQUE DE MEDICIÓN DE NIVEL DE COMBUSTIBLE*** ..... ***89***

***4.4. PRUEBA DE FUNCIONAMIENTO DE LA MEDICIÓN DE VELOCIDAD*** ..... ***91***

***4.5. PRUEBA DE FUNCIONAMIENTO DEL HMI REALIZADO CON LABVIEW 2009*** ..... ***93***

***4.6. PRUEBA DE FUNCIONAMIENTO DEL MEDIDOR DE TEMPERATURA*** ..... ***96***

**CAPITULO 5** ..... 98

**5. CONCLUSIONES Y RECOMENDACIONES** ..... 98

***5.1. CONCLUSIONES*** ..... ***98***

***5.2. RECOMENDACIONES*** ..... ***99***

**REFERENCIAS BIBLIOGRÁFICAS** ..... **101**

**ANEXO A** ..... **102**

**ANEXO B** ..... **103**

**ANEXO C** ..... **104**

**ANEXO D** ..... **105**

**ANEXO E** ..... **106**

# RESUMEN

El capítulo uno muestra una breve introducción sobre los sistemas telemétricos su uso en competencia, y los sistemas que se pueden encontrar en el mercado.

El capítulo dos presenta los dispositivos y demás elementos que vamos a utilizar para la realización de nuestro proyecto, nos da una breve descripción de sus características, sus usos más frecuentes así como ventajas y desventajas de los mismos.

El capítulo tres muestra la etapa de diseño, en la cual partimos desde las necesidades que deben de cumplir cada una de las etapas hasta el análisis de las señales que ingresan y las que salen de las mismas.

El capítulo cuatro nos presenta las pruebas que se realizaron al sistema, habiendo pruebas para las etapas físicas individuales, para la HMI, y para el sistema funcionando en conjunto.

El capítulo cinco son las conclusiones a las que llegamos después de realizar este proyecto y damos recomendaciones para la realización del mismo, aplicables claro a trabajos similares.

# PRESENTACIÓN

El trabajo descrito a continuación más que un tratado sobre telemetría o mecánica pretende ser una guía para los técnicos de las carreras de electrónica quienes están interesados en implementar diseños para el campo automotriz.

Por medio del presente proyecto se describen las pruebas que se hicieron a los sensores, que nosotros como electrónicos conocemos, orientadas a su uso en automotores, además mediante un procedimiento metódico y claramente descrito mostramos la realización del sistema telemétrico desde cero hasta su implementación sobre vehículos de competencia.

Esperamos que el presente trabajo sea de utilidad para las personas interesadas en incursionar en aplicaciones de electrónica para vehículos automotores.

# CAPITULO 1

# 1. INTRODUCCION Y ANTECEDENTES

## 1.1. TELEMETRÍA

La telemetría es un procedimiento para medir magnitudes físicas y químicas de forma remota al lugar donde estas se producen. De esta manera se puede adquirir datos sobre el funcionamiento de un equipo, dispositivo, u otra clase de fuente remota a través de cable, radio u otro medio. La telemetría es ampliamente utilizada para monitoreo y a veces control en grandes sistemas industriales como plantas químicas, textiles, petroleras; en el control y monitoreo a distancia de naves espaciales; e incluso en competencias deportivas para autos para realizar el monitoreo del funcionamiento del mismo y poder dar instrucciones tanto a mecánicos como al piloto, dentro de estas competencias se encuentran el rally, la fórmula 1, y el karting. En el presente proyecto nos centraremos en el monitoreo a distancia de vehículos de karting.

### 1.1.1 LA TELEMETRÍA EN COMPETENCIAS DEPORTIVAS

La importancia de la telemetría en competencias deportivas de autos ha llegado al punto que los diferentes equipos dependen tanto de estos sistemas como de los fabricantes de motores o neumáticos. La dependencia radica en que la información adquirida de la telemetría mientras el vehículo está en competencia permite evaluar el funcionamiento del mismo ante situaciones no controladas, lo que facilita la creación de sistemas de simulación y emulación más apegados a la realidad; también aporta en el desarrollo de futuros sistemas; también permite analizar fallas y/o rendimiento de los vehículos de karting y aplicar las correcciones necesarias, en nuestro caso permitirá que los mecánicos encuentren con rapidez fallas en el vehículo de karting y puedan poner el vehículo a punto con menos trabajo.

La mayor parte de la información disponible sobre telemetría en competencias deportivas se encuentra orientada a fórmula 1, sin embargo muchos criterios

pueden ser ampliables a otras competencias realizando los cambios apropiados -de aquí se desprende una de las necesidades de elaborar un sistema hecho a medida de los requerimientos, de eso se hablará mas adelante-. Nuestro proyecto involucra adaptar estos conceptos al caso del karting. Dentro de los cambios a realizar se encuentran la cantidad de sensores usados y el tipo de motor con el que vamos a trabajar. El motor usado en karting es un motor de dos tiempos mientras que en fórmula 1 se usan motores a cuatro tiempos.

En fórmula 1 se usan sensores para medir cerca de 100 parámetros entre los que cuentan temperatura del motor, presión aplicada sobre cada pedal –acelerador y freno- , ángulo de giro de las ruedas, presión de aire de las ruedas, velocidad en tiempo real, revoluciones del motor, por darnos una idea. En el presente proyecto se pretende monitorear la posición del vehículo en la pista, así como su velocidad, temperatura del motor, nivel de combustible, combustible usado por vuelta, y la velocidad del motor. Parámetros como la velocidad del vehículo y el combustible usado por vuelta serán evaluados en forma indirecta.

Luego de medidos los diferentes parámetros estos serán transmitidos a los “pits” -la base de operaciones del equipo- para ser procesados y analizados y así los técnicos del equipo tomen las decisiones necesarias.

### 1.1.2 TRANSMISIÓN Y PROCESAMIENTO DE LOS DATOS GENERADOS POR EL VEHÍCULO

Los datos generados por los sensores serán enviados a los “pits” a través de un módulo transmisor ubicado en el vehículo. En los pits se encuentra instalado el sistema de recepción de la señal. Los técnicos pueden observar la información recibida en sus monitores. Basados en la información, los técnicos le comunican al piloto que acciones debe tomar, y así se completa el proceso. Ver fig. 1.1.

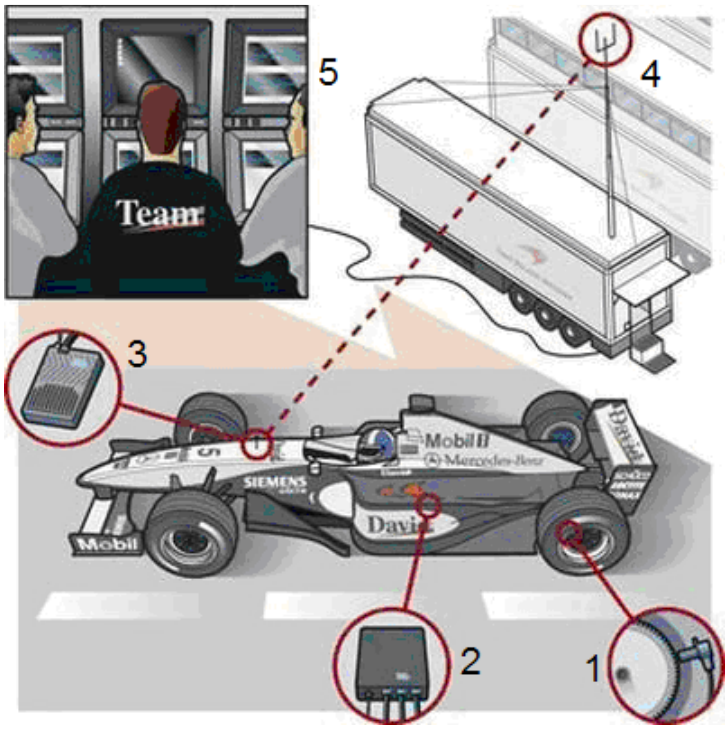


**Figura 1.1.** Ejemplo de transmisión de datos a pits mediante telemetría en fórmula 1. Partes del sistema: 1. Sensor instalado en una rueda, 2. Módulo de acondicionamiento de los sensores, 3. Módulo transmisor de información, 4. Antena del sistema de recepción, 5. Técnicos observando la información recibida. [1]

### 1.1.3 DISEÑO E INSTALACIÓN A MEDIDA DE LOS REQUERIMIENTOS

Por diferentes factores el diseñador se ve obligado a crear sistemas telemétricos propietarios, dentro de esos factores quizás los más importantes son:

- No se puede interferir en los sistemas electrónicos existentes. El dueño no desea que se le “meta mano” a lo que está previamente instalado.

- No existen módulos orientados a alguna aplicación específica.

- Los módulos existentes en el mercado son muy costosos y a veces sobredimensionados para las necesidades reales.

## 1.2 EL VEHÍCULO DE KARTING

Es un vehículo terrestre monoplaza sin techo o cockpit, sin suspensiones y con o sin elementos de carrocería. Posee cuatro ruedas no alineadas que están en contacto con el suelo, las dos delanteras ejerciendo el control de dirección y las dos traseras conectadas por un eje de una pieza que transmiten la potencia de un motor monocilíndrico.

Las partes principales de un vehículo de karting son: el chasis (comprendida la carrocería), los neumáticos y el motor.

### 1.2.1 CHASIS

Es la parte principal de la estructura del automóvil que soporta la carrocería y las partes mecánicas y está formado por un conjunto de tubos de acero soldados entre si para conformar una estructura rígida.

La figura 1.2. muestra la forma del chasis y sus partes constitutivas.

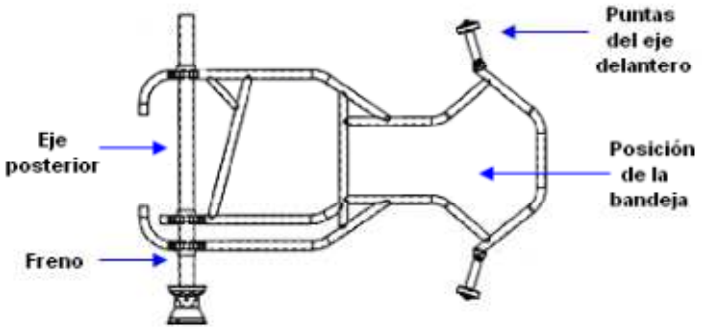


**Figura 1.2.** Chasis de vehículo de karting

### 1.2.2 BANDEJA

Está colocada sobre el extremo delantero del chasis hasta el comienzo del asiento y está atornillada al chasis.

Se la utiliza como apoya pies, protección y soporte para el depósito del combustible, el cual se coloca entre las piernas del piloto bajo la columna de dirección.

La forma que la bandeja tenga depende exclusivamente de los reglamentos de acuerdo a la categoría de la competición a la cual pertenezca el vehículo.
El material para la construcción generalmente es de aluminio de 1 o 2 mm. de espesor aunque actualmente también se utiliza plástico y otros materiales.

### 1.2.3 BARRAS

Es un conjunto de tubos que se colocan en el bastidor y sirven para dar estabilidad y soporte a otros elementos que van sujetos sobre ellos.
La defensa sujeta al chasis tiene el nombre de barra estabilizadora del eje delantero y sirve principalmente para proteger los pies del piloto y para sostener los faldones del vehículo, la defensa posterior sirve igualmente para protección y adicionalmente como agarre para la salida del vehículo.
El vehículo también dispone de protecciones laterales las cuales sirven para proteger al piloto contra golpes.

### 1.2.4 DIRECCIÓN

Esta conformada por el volante, la columna de dirección también llamada eje del volante, palanca del eje de dirección y la barra de dirección.
Todo este conjunto de partes funcionan de tal manera que si el volante gira en una dirección la barra de dirección apoyada en el eje de dirección iría en sentido contrario por lo cual las ruedas irían en dirección que el volante gire, así como se muestra en la figura 1.3.

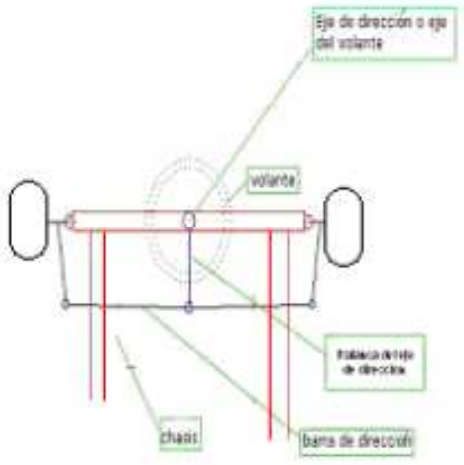


**Figura 1.3**. Esquema de la dirección del vehículo

### 1.2.5 EJE POSTERIOR

Consiste en una barra de acero cilíndrica con un diámetro comprendido entre 20 y 45 mm. la cual, puede ser hueca o maciza y su grosor depende de la categoría a la que pertenece el vehículo.

En los automóviles de karting sin cambios, el eje posterior es muy importante ya que es el único soporte del freno.

El eje posterior se encuentra unido al chasis del vehículo mediante dos o tres apoyos que cuentan con rodamientos los cuales permiten que la barra gire con mucha facilidad.

### 1.2.6 FRENOS

Consiste en discos, los cuales van colocados en el eje posterior, los vehículos de karting que poseen marchas adicionalmente poseen discos de frenos en sus llantas delanteras.

Las pastillas que actúan sobre los discos van posicionadas en conjunto con los discos sobre el chasis.

Los frenos utilizados pueden ser accionados por medio del cable o por una bomba hidráulica en caso de ser frenos hidráulicos.

El freno montado en el eje posterior se muestra en la figura 1.4.

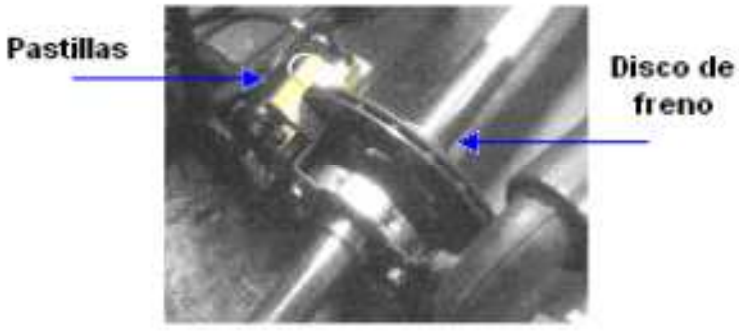


**Figura 1.4.** Freno montado en el vehículo

### 1.2.7 MOTOR

El motor que se utiliza es un motor de dos tiempos cuyo cilindraje y características depende de la categoría a la que pertenece el vehículo.

Los motores pueden ser monocilíndricos para las categorías de 80 cc. a 100 cc. y bicilíndricos, de dos cilíndricos, para los motores que poseen cambios cuyos cilindraje es de 250 cc.

Los motores monocilíndricos se refrigeran mediante el aire aunque en la actualidad existen vehículos que como refrigerante utilizan agua, como es el caso de los motores que poseen caja de cambios.

La figura 1.5. muestra la imagen del motor de un vehículo de karting.

**Figura 1.5.** Motor refrigerado por aire para vehículo de karting

#### 1.2.7.1 Motor de dos tiempos

Es un motor de combustión interna que realiza las cuatro etapas del ciclo termodinámico (admisión, compresión, expansión y escape) en dos movimientos lineales del pistón o una vuelta del cigüeñal.

No posee válvulas y cada dos tiempos hay una carrera de trabajo, lo que significa que cada revolución del motor produce un impulso.

Consta de un cilindro que en su interior aloja el hueco por donde se traslada de forma lineal el pistón o émbolo.

El cilindro se ubica en una caja de aluminio llamada cárter.

El cigüeñal es la pieza que recibe el movimiento de vaivén del pistón por la biela y lo transforma en el movimiento rotatorio del cigüeñal.

El funcionamiento del motor de dos tiempos es de la siguiente manera:

#### *1.2.7.1.1 Primer tiempo*

La bujía inicia la explosión de la mezcla de aire y gasolina previamente comprimida. Como consecuencia de la presión del gas caliente, baja el pistón y realiza trabajo. También cierra el canal de admisión A, comprime la mezcla abajo en el cárter, un poco más tarde abre el canal U y el canal de Escape E. Bajo la compresión adquirida el gas inflamable fresco fluye del cárter por el canal U hacia la cámara de explosión y empuja los gases de combustión hacia el tubo de escape. Así el cilindro se llena con mezcla fresca.

La figura 1.6. muestra el interior del cilindro durante el inicio del primer tiempo.

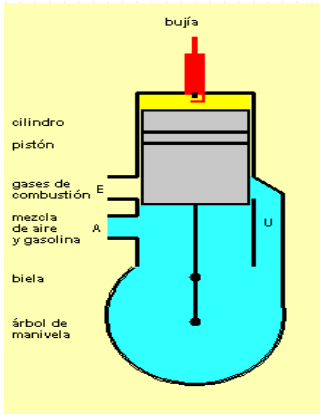


**Figura 1.6.** Motor dos tiempos (Primer tiempo) [1]

#### *1.2.7.1.2 Segundo tiempo*

El émbolo vuelve a subir y cierra primero el canal U, después el canal de escape E. Comprime la mezcla, se abre el canal de admisión A y llena el cárter con la mezcla nueva preparada por el carburador.

A continuación la figura 1.7. muestra el interior del cilindro durante el segundo tiempo.

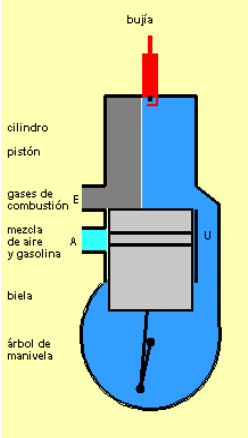


**Figura 1.7.** Motor de dos tiempos (Segundo tiempo) [1]

*1.2.7.1.3 Sistema de Ignición*

El sistema de ignición de un vehículo de karting es muy básico ya que solo está conformado por: la bujía y la bobina.

*1.2.7.1.4 Bujía*

Una bujía es el elemento que produce el encendido de la mezcla de combustible y aire en los cilindros, mediante una chispa, dentro del cilindro de un motor de combustión interna.

La bujía recibe un voltaje comprendido entre 10 y 30 mil voltios procedentes de la bobina, lo cual genera una chispa entre el electrodo central de la bujía y el terminal que está conectado a tierra.

Se utiliza una bujía por cilindro.

Existen en el mercado muchas variedades de bujías siendo las de iridio las que generan mayor potencia y permiten obtener explosiones más efectivas.

La chispa se genera en la parte (A) en la figura 1.8. El resistor se ha incluido para evitar el ruido generado por las chispas de alto voltaje.

Además hay también bujías especializadas para competencia en donde los materiales de construcción y la posición de los electrodos es lo más importante. La bujía a ser utilizada tiene mucho que ver con la temperatura de la cavidad de explosión del motor, ya que dependiendo del material y las dimensiones del electrodo es la potencia de la chispa generada por la bujía.

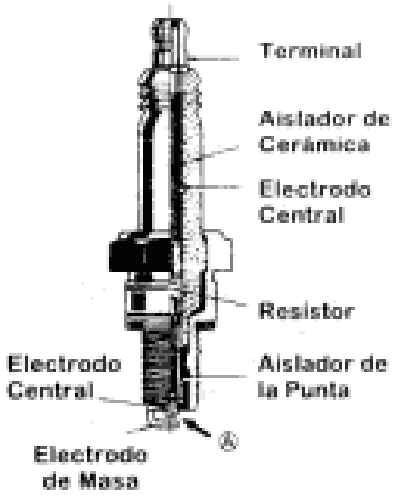


**Figura 1.8.** Bujía [2]

*1.2.7.1.5 Bobina*

Este es el dispositivo que permite entregar el alto voltaje a la bujía para producir la chispa de encendido dentro del cilindro.

Consiste en una bobina de bajo voltaje y una bobina de alto voltaje que están conectadas formando un autotransformador. La bobina de bajo voltaje está montada alrededor de un núcleo magnético que se encuentra en el motor, el cual al girar mientras el motor está encendido excita a la bobina, esta bobina induce un voltaje a la bobina de alto voltaje.

La figura 1.9. y 1.10. muestran las bobinas instaladas en el motor de un vehículo de karting.

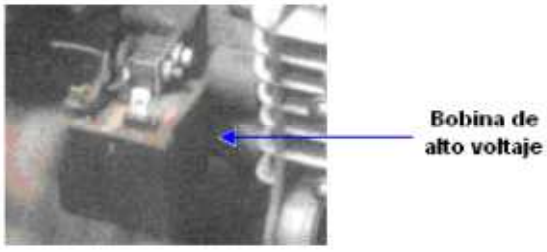


**Figura 1.9.** Bobina de alto voltaje

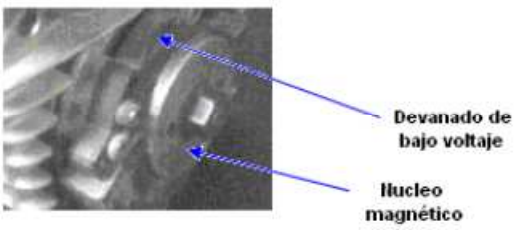


**Figura 1.10**. Bobina de bajo voltaje

*1.2.7.1.6 Tanque de combustible*

Está ubicado bajo la columna de dirección en medio de las piernas del conductor. De acuerdo al reglamento el material de construcción deberá ser de plástico y debe poseer un sistema antiderrames el cual deberá funcionar inclusive cuando el tanque se encuentre en posición invertida.

*1.2.7.1.7 Filtro de Admisión*

Es un dispositivo de plástico pequeño que tiene la finalidad de almacenar aire. Este dispositivo varía la carburación, razón por la cual se utiliza en las pruebas de clasificación. De acuerdo al reglamento, el filtro debe ser de una marca homologada.

La figura 1.11. muestra el filtro de admisión para un motor de un vehículo de karting.

**Figura 1.11.** Filtro de admisión

*1.2.7.1.8 Carburador*

El carburador es el dispositivo que hace la mezcla de aire-combustible en los motores de gasolina. A fin de que el motor funcione más económicamente y obtenga la mayor potencia de salida, es importante que la gasolina esté en las mejores condiciones. A fin de hacer una mezcla óptima de aire-combustible los mecánicos de los automóviles de karting regulan este dispositivo en los entrenamientos, para cada competencia.

La gasolina en este tipo de vehículos es mezclada con aceite, utilizando 8 onzas de aceite sintético por cada galón de combustible.

La figura 1.12. muestra el carburador de un vehículo de karting.

**Figura 1.12.** Carburador de un vehículo de karting

*1.2.7.1.9 Transmisión*

El sistema de transmisión es lo que permite que la potencia generada por el vehículo se convierta en movimiento del vehículo.

La transmisión de un vehículo de karting está conformada por tres elementos que son: la cadena, el piñón y la corona.

El sistema de transmisión se muestra en la figura 1.13

*1.2.7.1.10 Piñón*

El piñón es el elemento de salida de potencia del motor hacia el eje trasero, cuanto más dientes tenga mayor velocidad proporcionará al vehículo de karting pero esto produce un efecto que reduce la aceleración del vehículo.

*1.2.7.1.11 Cadena*

La cadena es una serie de eslabones que transmite el movimiento del piñón a la corona, ya que no se utilizan tensores en los vehículos de karting la dimensión de la cadena debe ser la adecuada al tamaño del piñón y de la corona.

*1.2.7.1.12 Corona*

Es la rueda dentada que está al eje y que recibe, a través de la cadena, la potencia del motor. El número de dientes que debe poseer la corona depende del tipo de circuito.

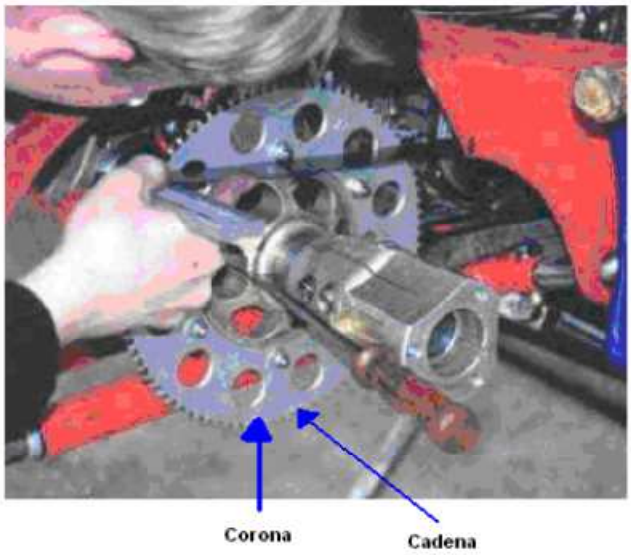


**Fig. 1.13** Sistema de transmisión de un vehículo de karting

## 1.3 SISTEMAS TELEMÉTRICOS EXISTENTES EN EL MERCADO

Dada la importancia que la telemetría ha adquirido, muchos fabricantes de productos de control han desarrollado sus propias soluciones para ofrecerlas al mercado. Dichas soluciones constan de software para procesar, presentar y analizar la información; y hardware para la transmisión y recepción de la información así como para manejar los diferentes sensores que se usan.

No se ha encontrado productos orientados específicamente a la telemetría de vehículos de karting, pero como ya se dijo con anterioridad algunas ideas se pueden tomar para cumplir con el propósito de este proyecto. Como parte de sistemas existentes, mostramos a continuación algunas soluciones tanto de software como de hardware que se encuentran en el mercado, orientadas tanto para competencia como para vehículos de uso doméstico

### 1.3.1 SOLUCIONES DE SOFTWARE

Como soluciones de software para realizar telemetría se puede utilizar casi cualquier software para control y monitoreo como por ejemplo LabView o InTouch; también se pueden realizar aplicaciones en Visual Basic, C++ o cualquier otro software para programación.

A continuación presentamos ejemplos de software orientado exclusivamente a telemetría. En el primer caso es un programa orientado exclusivamente a telemetría de competencias deportivas de autos, y en el segundo caso sirve para telemetría de cualquier clase de sistema.

#### 1.3.1.1 Software “TELEMETRIA. F1 Challenge 99-02”

Este software está orientado a pilotos avanzados de fórmula 1, que están familiarizados con la frecuente preparación del motor para competencia. Este software permite evaluar el tiempo ocupado por vuelta, además que permite observar la posición del vehículo en la pista, su velocidad y las revoluciones por minuto, del motor además que permite analizar la fuerza G – permite analizar como el peso del vehículo se distribuye cuando se frena, se acelera o toma las curvas por acción de la inercia-. Ver fig. 1.14.

**Figura1.14.** Software telemetría F1 mostrando la pista y la velocidad en función del tiempo

#### 1.3.1.2 Software “u-Lab y u-Graph” de KMT - Messtechnik & Telemetrie [4]

Es un software orientado a la adquisición y análisis de datos bajo el entorno de Windows 95/98/NT además que permite graficar y aplicar transformada rápida de Fourier –FFT- a las imágenes. Ver fig.1.15. y fig.1.16.

#### 1.3.1.3 Soluciones de hardware

Como soluciones de hardware (sensores, transmisores, receptores, acondicionadores de señal), se pueden utilizar dispositivos existentes orientados a aplicaciones industriales, pero al igual que con el software existen soluciones orientadas al sector automotriz y algunas orientadas exclusivamente a vehículos de competencia.

Existen varios fabricantes que ofrecen soluciones de hardware para telemetría. A continuación mostramos algunas de estas soluciones del fabricante “KMT - Messtechnik & Telemetrie”.

#### 1.3.1.4 Línea de productos de KMT - Messtechnik & Telemetrie

KMT - Messtechnik & Telemetrie ofrece una línea muy completa de transmisores y receptores orientados a telemetría de vehículos con fines tanto industriales como deportivos.

#### 1.3.1.5 Sistema de telemetría 4/8 canales para ruedas “CT4/8-Wheel”

Este sistema permite medir la temperatura en el disco del freno, provee una resolución de 12 bits y puede usar como sensores termocuplas tipo K, y sensores capacitivos. El sistema incluye un módulo receptor con su antena, los cables de alimentación y los cables para los sensores. Ver figuras 1.15. y 1.16.

**Figura 1.15.** Módulo CT4/8 - Wheel instalado en una rueda

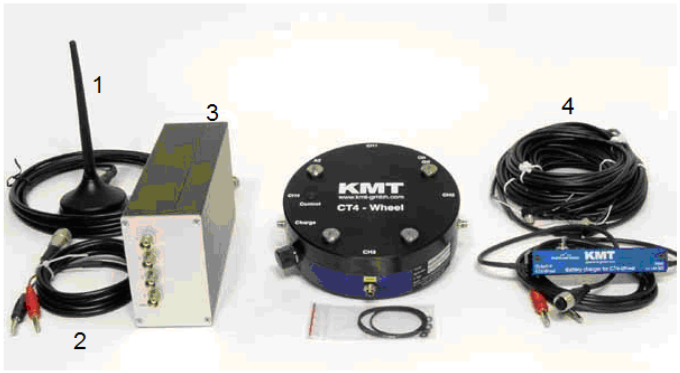


**Figura 1.16.** Módulo incluyendo: 1. Antena receptora, 2. Cables de alimentación, 3. Receptor, 4. Cables para conectar los sensores.

### 1.3.1.6 Medidor de revoluciones por minuto del motor "RPM-8000-PRO"

Este es un dispositivo práctico que permite medir las revoluciones por minuto (RPM) del motor del auto sin necesidad de conectar un sensor adicional, simplemente se conecta a los contactos de la fosforera o de la batería del vehículo y nos dará la medida. Ver fig. 1.17.

**Figura 1.17.** Módulo RPM-8000-PRO

### 1.3.1.7 Sistema de transmisión – recepción “CT2 mini”

Este es un conjunto de un transmisor y su respectivo receptor. El transmisor acepta en sus entradas sensores de presión, fuerza, temperatura y voltaje para maquinas giratorias y la transmite hasta máximo 500 m dependiendo de las antenas usadas. Ver fig. 1.18.

**Figura 1.18.** Transmisor y receptor con su antena

### 1.3.1.8 Sistema de transmisión – recepción TEL1/40k TTL

Estos módulos constan de un par transmisor y receptor, transmite de 40 a 40000 pulsos TTL/seg a una distancia de hasta 500 m y utilizando 4 canales diferentes. Ver fig. 1.19.

**Figura 1.19.** Transmisor y receptor TEL1/40k TTL

# CAPITULO 2

# 2 GENERALIDADES

De acuerdo a lo propuesto en el proyecto, en el presente capitulo se abordará una breve descripción de las tecnologías a aplicarse, mostrando sus características y funcionamiento, y así poder comprender la importancia de estas en la realización del proyecto.

## 2.1. POSICIÓN DEL VEHÍCULO EN LA PISTA

Uno de los objetivos generales planteados en el presente proyecto es el monitoreo del vehículo de karting en la pista.

Para alcanzar dicho objetivo se ha visto que la manera más conveniente es el uso de un receptor GPS debido a que en la actualidad representa la solución más económica, con menores requerimientos de hardware adicional y mayor facilidad de implementación por las dimensiones que poseen los receptores.

A continuación se realizará una breve descripción del sistema GPS.

### 2.1.1. SISTEMA GPS

#### 2.1.1.1. Introducción

Desde el año de 1959 con el lanzamiento del satélite Vanguard, de uso militar, se comenzó a investigar sobre la manera de localizar un punto cualquiera de la Tierra con la ayuda de la transmisión de señales de radio desde el espacio. En el año de 1993 el Departamento de Defensa de los Estados Unidos de América, basado en los resultados recogidos de la investigación con el satélite *Vanguard* puso en funcionamiento el sistema de localización por satélite conocido

por las siglas en inglés GPS *(Global Positioning System)* ó Sistema de Posicionamiento Global.

En el principio del proyecto, en las transmisiones de los satélites GPS estaban programados errores de cálculo codificados para limitar la aplicación solo a uso militar y así se mantuvo hasta el año 2000 en donde se dio apertura a su libre uso y aplicación en cualquier actividad de la vida civil.

El sistema GPS permite conocer en tiempo real la posición y la altura de un objeto situado en cualquier punto de la Tierra en todo momento, con una precisión de incluso pocos centímetros, sin importar si se encuentra fijo o en movimiento.

La primera prueba exitosa del sistema GPS como instrumento de ayuda a la navegación, la realizó el trasbordador espacial *Discovery*. Actualmente los satélites GPS pertenecen a una segunda generación denominada *Block* II.

### 2.1.1.2. Estructura del sistema GPS

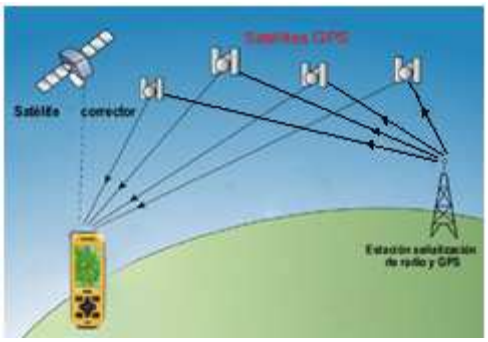


**Figura 2.1.** Estructura del Sistema GPS [1]

En la figura 2.1 se puede observar que el sistema GPS está conformado por tres partes que son los satélites, los receptores y el control terrestre.

A continuación se describirá las partes que conforman al sistema GPS para que se pueda entender de mejor manera su funcionamiento y la importancia de cada una de estas.

*2.1.1.2.1. Satélites*

El sistema se compone de 27 satélites de órbita terrestre media de los cuales 3 son de respaldo, situados a una altura 20169 kilómetros de distancia de la Tierra. Los satélites se encuentran distribuidos en seis órbitas polares diferentes, es decir, están distribuidos en seis planos orbitales de cuatro satélites cada uno y se encuentran sincronizados de tal forma que cubren toda la superficie terrestre como lo indica la figura 2.2

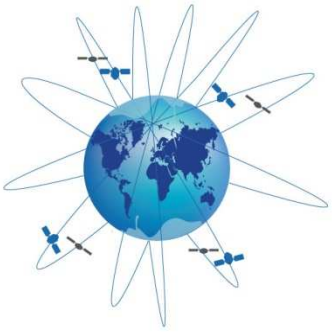

**Figura 2.2.** Orbita Polar

Una órbita polar es una órbita en la que el satélite se desplaza en dirección norte – sur y como el planeta Tierra gira en dirección este, el satélite puede cubrir todas las combinaciones posibles de latitud y longitud.

Cada uno de los satélites que conforman la red mide cinco metros de largo y pesa aproximadamente 860Kg. La energía eléctrica que requieren para su funcionamiento la adquieren a partir de dos paneles compuestos de celdas solares y el respectivo sistema de baterías de respaldo. Adicionalmente están equipados con un transmisor de señales codificadas de alta frecuencia (1575,42

MHz y 1227,6MHz ), un sistema de computación y un reloj atómico de cesio, con una exactitud tal que se atrasa un segundo cada 30 mil años.

Cada satélite realiza dos vueltas completas en cada día sideral, un día sideral es equivalente a 23 horas 56 minutos y 4 segundos, es decir a una velocidad aproximada de 11300 kilómetros por hora.

#### *2.1.1.2.2. Control terrestre*

El monitoreo y control de los satélites que conforman el sistema GPS se ejerce desde diferentes estaciones terrestres que pertenecen a la Fuerza Aérea de los Estados Unidos ubicadas en Hawai, Kwajelein (Islas Marshall en el Pacífico Occidental), Isla Ascensión (costa oeste de África), Diego García (sur de la India) y Colorado Springs (Estados Unidos), además de estaciones de monitoreo operadas por la National Geospatial-Intelligence Agency (NGA). La información de seguimiento es enviada a la estación de control maestro de Colorado Springs.

Las estaciones rastrean la trayectoria orbital de los satélites que son afectadas por la atracción de la Tierra, la Luna y el Sol e introducen las correcciones necesarias a las señales de radio que son afectadas por la ionosfera.

#### *2.1.1.2.3. Receptores*

Los receptores GPS son los instrumentos que detectan, decodifican y procesan la señal que obtienen de los satélites para determinar la posición y la altura a la que se encuentra situados.

En general los receptores GPS poseen una antena, sintonizada a las frecuencias transmitidas por los satélites; procesadores de recepción, un reloj de alta estabilidad como por ejemplo un reloj de cristal, adicionalmente pueden incluir un display para presentar la información de la ubicación y la velocidad del usuario.

Cuando se solicita una posición, el receptor que se utiliza para ello necesita por lo menos la señal de 3 satélites de la red, los cuales envían señales indicando la posición y el reloj de cada de uno de ellos. La posición que ocupan los satélites en sus respectivas órbitas facilita que el receptor GPS reciba, de forma constante y simultánea, las señales de por lo menos 6 u 8 satélites, para que esto sea posible los receptores poseen un cierto número de canales, es decir el número de satélites que puede monitorear simultáneamente, en un principio el número de canales era limitado a 4 o 5, sin embargo los receptores de última generación son capaces de monitorear entre 12 y 20 canales simultáneamente.

El receptor calcula el retraso de cada una de las señales y con esto puede determinar la distancia al satélite. Con las distancias y las coordenadas a las cuales se encuentra cada satélite, el receptor obtiene las coordenadas del punto en donde se encuentra.

Algunos receptores GPS pueden retransmitir la información de posición a un PC u otro dispositivo usando el protocolo NMEA 0183, o el NMEA 2000, los cuales son abiertos para el uso del público en general.

Los receptores se pueden interconectar con otros dispositivos usando interfaces como conexión serial, USB o Bluetooth entre otros.

### 2.1.2. FUNCIONAMIENTO DEL GPS

Los sistemas GPS basan su funcionamiento en el principio de la triangulación, con el que se conoce una posición, permitiendo incluso rastrear y ubicar el origen de una transmisión por ondas de radio.

Desde el momento que el receptor GPS detecta una señal de radiofrecuencia transmitida por un satélite desde su órbita, se genera una esfera virtual o imaginaria que envuelve al satélite. El propio satélite actuará como centro de la esfera cuya superficie se extenderá hasta el punto o lugar donde se encuentre situada la antena del receptor; por tanto, el radio de la esfera será igual a la distancia que separa al satélite del receptor. A partir de ese instante el receptor GPS medirá las distancias que lo separan como mínimo de dos satélites más. Para ello tendrá que calcular el tiempo que demora cada señal en viajar desde los satélites hasta el punto donde éste se encuentra situado y realizar los correspondientes cálculos matemáticos.

Para medir el momento a partir del cual el satélite emite la señal y el receptor GPS la recibe, es necesario que tanto el reloj del satélite como el del receptor estén perfectamente sincronizados.

Como se mencionó anteriormente el satélite utiliza un reloj atómico de cesio sumamente exacto pero el receptor posee uno normal, en su mayoría de cuarzo, el cual no es tan preciso. Para sincronizar con exactitud el reloj del receptor, el satélite emite cada cierto tiempo una señal digital o patrón de control. El receptor utiliza los tiempos de llegada para calcular la distancia a cada satélite, del cual determina la posición del receptor usando geometría y trigonometría. Este resultado sería preciso si la señal viajase por el vacio pero en este caso

atraviesan la masa gaseosa que compone la atmósfera, además existen otros factores que pueden influir también algo en el desplazamiento de la señal, como son las condiciones atmosféricas locales y el ángulo existente entre el satélite y el receptor. Para corregir los efectos de todas esas variables, el receptor se sirve de complejos modelos matemáticos que guarda en su memoria y así los resultados de los cálculos los complementa después con la información adicional que recibe también del satélite, lo que permite mostrar la posición con mayor exactitud.

#### 2.1.2.1. Señales de navegación de los satélites

Los satélites continuamente envían señales de navegación que consisten en una trama de datos que ocupa 1500 bits la cual es transmitida en cinco partes de 300 bits cada una.

Las tramas se transmiten cada 30 segundos a una velocidad de transmisión de 50 bps. La figura 2.3 describe como están conformadas las tramas del mensaje de navegación de los satélites.

Los primeros 18 segundos de transmisión contienen datos orbitales y temporales. En los primeros 6 segundos se transmiten las correcciones del reloj y la salud o estado del satélite y en los siguientes 12 segundos se transmiten datos de información orbital, también llamados efemérides.

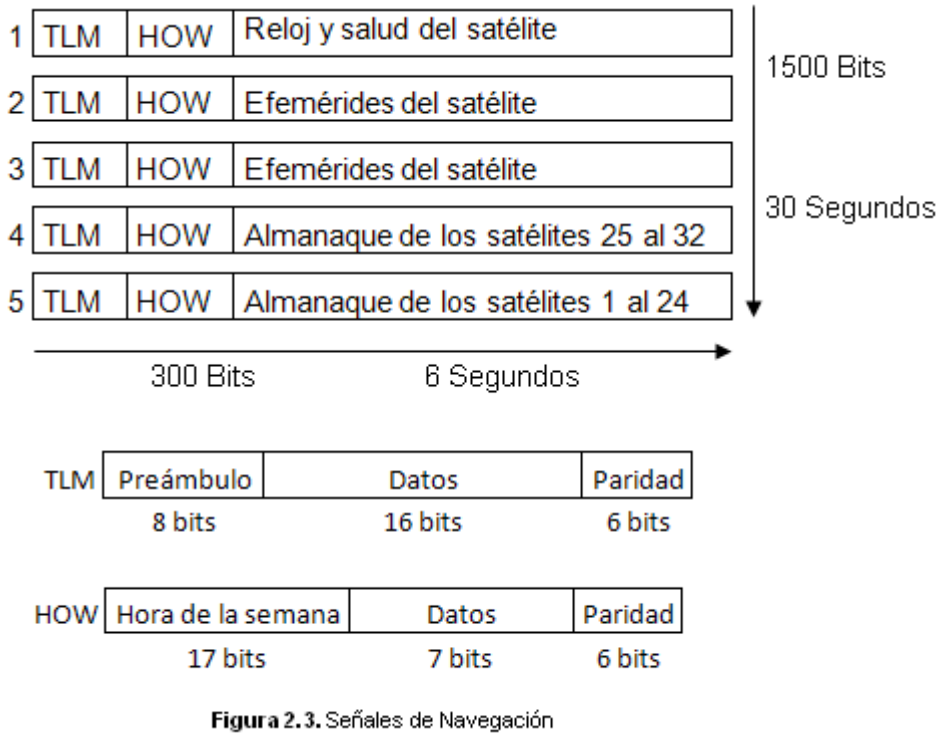


**Figura 2.3.** Señales de Navegación

La Efemérides es la trama que envían los satélites con su ubicación exacta y es actualizada por las estaciones de seguimiento de la Tierra. Esta información es actualizada en un tiempo típico de 4 horas o en 6 horas cuando se trabaja en condiciones de funcionamiento anormales.

En el cuarto bloque de bits de 6 segundos se transmite información del estado del satélite llamada también salud del satélite y otros parámetros de navegación y en el quinto bloque se transmiten el almanaque de un satélite.

El almanaque es un conjunto de valores que permiten predecir la órbita y la posición de todos los satélites de una manera aproximada. Cada satélite envía de forma individual los datos correspondientes al almanaque de todos los demás satélites. El tiempo de adquisición de señal en el arranque de un receptor puede ser mejorado significativamente si dispone del almanaque actualizado ya que con esa información el receptor puede predecir que satélites están disponibles en ese

momento.

La trama completa de la señal de navegación se retransmite cada 12 minutos y medio.

La palabra telemetría (TLM) que describe la figura 2.3 está destinada a medir la distancia con el satélite. El preámbulo de 8 bits siempre es 10001011.

La palabra HOW (Hand Over Word) permite identificar la hora a la que ha sido enviado el último bit de la subtrama.

### 2.1.3. TIPOS DE ENCENDIDO DE LOS RECEPTORES GPS

El comportamiento de los receptores GPS al momento del encendido dependerán de la información de las señales de navegación que tengan almacenadas en la memoria. Para empezar a predecir los receptores necesitan los datos de tiempo, posición y el almanaque.

En función de los datos almacenados el arranque de los receptores GPS puede ser:

#### 2.1.3.1. Warm Start

Es decir arranque en caliente, este tipo de arranque es posible cuando el receptor tiene datos aproximados de tiempo, posición, almanaque y el efemérides conocido es reciente. En situaciones de arranque warm start la adquisición de datos puede tomar menos de 30 segundos.

#### 2.1.3.2. Cold Start

Es decir arranque en frio, este tipo de arranque es posible cuando se conocen datos aproximados de tiempo y posición, el almanaque pero se desconoce el efemérides de por lo menos un satélite de los cuatro necesarios para poder obtener una ubicación. El tiempo de la adquisición de datos puede tomar más de 60 segundos.

#### 2.1.3.3. Autolocate

Este tipo de arranque es posible cuando se desconocen totalmente las variables de tiempo, efemérides y posición pero se conoce parte o todo el almanaque. Es común que suceda esto cuando el receptor ha estado apagado en largos periodos de tiempo o se haya variado mucho la posición. El tiempo de adquisición de datos es equiparable con el arranque en frio.

#### 2.1.3.4. Search in the sky

En este tipo de arranque se desconocen todas las señales de navegación. En este caso la adquisición de datos puede tomar de 10 a 15 minutos.

### 2.1.4. SEÑALES DEL GPS

Los satélites del sistema GPS transmiten una señal en polarización circular derecha RHCP (Right Hand Cirularly Polarized) en las bandas L1 de microonda a una frecuencia de 1575,42 MHz y L2 a una frecuencia de 1227,6 MHz

La banda L es emitida a una potencia suficiente para tener un nivel de recepción superior a los -160 dBW para L1 y de -166dBW para L2.

Los receptores pueden identificar las señales de los distintos satélites del sistema

GPS, aunque transmitan a la misma frecuencia, porque utilizan un código de división de acceso múltiple (CDMA).

Los códigos CDMA de espectro expandido que utilizan los satélites GPS son:

#### 2.1.4.1. Coarse/Acquisition (C/A)

Es una secuencia de datos de una duración de un milisegundo que está conformada por un código pseudo-aleatorio de 1023 bits conocido como el código PRN y una señal de saltos de fase bipolar (BPSK) en la frecuencia de 1,023 MHz que modula a la portadora L1. Cada satélite tiene su propio (C/A)

#### 2.1.4.2. Precise (P)

Es un código que trabaja a 10,23 Mchips/s que se repite solo una vez por semana. La portadora L2 es modulada por este código.

Existe una versión de este código cuyos datos son encriptados y se lo conoce como P(Y).

### 2.1.5. FUENTES DE ERROR

El sistema GPS es susceptible a la presencia de errores que van desde ± 1 metro hasta ± 30 metros. Los errores a los cuales es susceptible el sistema GPS pueden ser relativos al Satélite, a la propagación de la señal y relativos a los receptores.

La tabla 2.1 muestra cuantitativamente como pueden influenciar los errores en la precisión de un receptor GPS.

| TIPO DE ERROR | ERROR DE PRECISION |
|---|---|
| Refracción ionosférica | ± 5 metros |
| Error de parámetros orbitales | ± 2,5 metros |
| Error del reloj del satélite | ± 2 metros |
| Efecto multipath | ± 1 metro |
| Refracción troposférica | ± 0,5 metros |
| Error en cálculos numéricos | ± 1 metro |

**Tabla 2.1** Errores en el sistema GPS

## 2.2. MICROPROCESADORES

En el presente proyecto ha surgido la necesidad de utilizar una serie de sensores que realizarán funciones especificas las cuales serán detalladas en el capitulo siguiente. Estos sensores generaran información que se deberá procesar y transmitir hacia el receptor que se encontrará en pits[1].

El dispositivo capaz de realizar las tareas de cálculo, procesamiento y transmisión serial de información es el microprocesador.

En el mercado hay una vasta cantidad de microprocesadores y microcontroladores, pero los que consideraremos para la realización del proyecto por sus características y por el conocimiento que se posee de su manejo son los PIC's fabricados por Microchip Technology Inc y los microcontroladores AVR fabricados por ATMEL.

A continuación se hará una breve descripción de cada uno de los microcontroladores anteriormente mencionados de modo que se pueda realizar una comparación entre ellos y de esa forma determinar cual se adapta de mejor manera a las necesidades a cubrir en el presente proyecto.

[1] Es el término utilizado para hacer referencia al lugar en que los automóviles que compiten pueden detenerse para realizar ajustes mecánicos, cambios de llantas o carga de combustible.

### 2.2.1. MICROCONTROLADOR PIC

El microcontrolador PIC (Peripheral Interface Controller) fue desarrollado a mediados de la década del 70 y diseñado con la capacidad de procesar palabras de 8 bits de datos.

Los microcontroladores PIC poseen una arquitectura RISC, razón por la cual pueden procesar una instrucción en 4 ciclos de reloj.

La arquitectura RISC es parte del siguiente capítulo en donde será descrita con mayor detalle.

La memoria de programa que poseen estos microcontroladores dependerá del modelo de PIC que se adquiera, pero se los puede encontrar en un rango que va desde 512 a 32000 palabras, y cada palabra podrá ser de 12, 14 o 16 bits, variable que también dependerá del tipo de PIC.

Para la programación de un microcontrolador PIC el fabricante distribuye el software llamado MPLAB el cual maneja lenguaje assembler pero también se puede encontrar software que permite la programación de estos microcontroladores con lenguaje basic.

En la actualidad son muchas las aplicaciones en las cuales se puede emplear un microcontrolador PIC ya que ahora por ejemplo incluyen módulos de comunicación serial e incluso módulos para el control de motores.

#### 2.2.1.1. Características del microcontrolador PIC

Actualmente son muchas las mejoras de hardware que se han realizado en el microcontrolador PIC y las cuales, como se dijo anteriormente, dependerán del tipo que se utilice pero haciendo un resumen de las más importantes se pueden mencionar las siguientes:

- Temporizadores de 8 y 16 bits.
- Conversores analógico-digital de 8, 10 y12 bits.
- Módulos que permiten el manejo de protocolos ethernet, CAN e Irda.
- Capacidad para utilizar la interfaz USB.
- Periféricos para el control de motores.
- Memorias flash con capacidad desde 256 bytes hasta 256 kbytes.

### 2.2.2. MICROCONTROLADOR AVR

El microcontrolador AVR es fabricado por ATMEL. Este microcontrolador es capaz de realizar una instrucción por cada ciclo de reloj por la optimización de la arquitectura RISC que posee, la cual le permite decodificar una instrucción mientras está realizando otra.

Posee 32 registros de 8 bits, lo sobresaliente en el microcontrolador AVR es que los registros de entrada y salida y la memoria de datos comparten un espacio de datos unificados, a diferencia del microcontrolador PIC el stack se ubica en este espacio de memoria unificado, y no está limitado a un tamaño fijo.

La diferencia entre los demás microcontroladores AVR son los tamaños de memoria RAM y ROM, en donde el más pequeño es el Tiny AVR ATtiny11 con 1KB de memoria flash, sin memoria RAM y con 8 pines; el más grande es el XMEGA ATxmega con 384 KB de memoria flash y 100 pines.

La programación de un AVR al igual que el PIC se la puede hacer con software que maneja leguaje assembler pero también se puede encontrar software que maneja lenguaje basic o inclusive leguaje C que fue el lenguaje para el cual el fabricante lo diseñó

#### 2.2.2.1. Características del microcontrolador AVR

Las características más sobresalientes del microcontrolador AVR son:

- Temporizadores de 8 y 16 bits
- Conversores A/D de 16 bits
- Poseen un reloj interno capaz de generar señales de 1MHz, 4MHz y 8MHz sin hardware adicional
- Manejo de interfaz USB
- Soportan Zigbee

Después de la experiencia adquirida con el manejo y analizar las diferentes características y prestaciones de los microcontroladores PIC y AVR se pueden realizar las siguientes comparaciones:

- El microcontrolador AVR presenta mayor inmunidad al ruido.
- Un microcontrolador AVR de 40 pines es más económico que un PIC de 40 pines con características similares.
- Mayor experiencia en manejo de microcontroladores AVR que microcontroladores PIC

Las características que poseen los microcontroladores AVR se ajustan de mejor manera para la realización del proyecto principalmente por su alta inmunidad al ruido ya que en el caso de nuestro proyecto el circuito va a ubicarse bastante cerca al motor del vehículo.

## 2.3. COMUNICACIÓN POR RADIOFRECUENCIA

Para la realización del presente proyecto se planteó transmitir la información de forma inalámbrica utilizando un rango de frecuencias que va de 200Mhz a 487Mhz.

En el mercado se puede encontrar una variedad de transmisores que trabajan a una frecuencia de 433MHz y se diferencian por la potencia de transmisión, tipo de modulación que utilizan y el número de canales que poseen.

A continuación se hará una descripción de las comunicaciones de radio frecuencia.

### 2.3.1. BANDAS DE FRECUENCIA DE USO LIBRE

Desde el año de 1985 la FCC (Federal comunications comission) liberó el uso del espectro radioeléctrico autorizando su uso sin licencia para transmisiones que posean una potencia que no exceda el valor de un vatio.

Las frecuencias liberadas fueron las siguientes: 313 Mhz, 434Mhz, 868 Mhz, 2,4Ghz y 5.8 Ghz; estas dos últimas actualmente son bastante usadas por las redes inalámbricas.

#### 2.3.1.1. Modulación

La modulación es la técnica mediante la cual se transporta información sobre una señal portadora, que generalmente es una sinusoide. El uso de la técnica de modulación permite el mejor uso del canal de transmisión y protege a la señal de la interferencia y ruido ambiental.

Las técnicas de modulación digital que se pueden usar son:

- Modulación por desplazamiento de amplitud
- Modulación por desplazamiento de fase o de fase múltiple
- Modulación por desplazamiento de frecuencia
- Modulación de amplitud en cuadratura

A continuación se hará una descripción de las técnicas de modulación que más se utilizan en los transmisores y módulos disponibles en el mercado.

*2.3.1.1.1. Modulación ASK (Amplitude Shift key)*

Es una técnica de modulación en donde la señal moduladora es una señal digital, entonces la señal a transmitir únicamente tendrá dos valores de amplitud al ser transmitida o que equivale a encender o apagar la señal portadora como se puede observar en la figura2.5 :

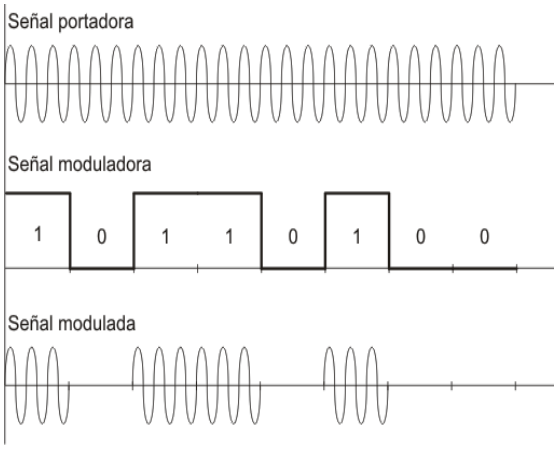


**Figura 2.5** Modulación ASK

El valor de la señal modulada está dado por la siguiente ecuación:

$$V(t) = Vp \cdot Vm \cdot sen(2\pi \cdot fp \cdot t)$$

En donde:

Vp → Valor pico de la señal portadora

Vm → Valor de la señal Moduladora

Fp → Frecuencia de la señal portadora

Entre las desventajas que tiene esta técnica de modulación  al igual que en la modulación AM es que este tipo de modulación es muy sensible al ruido atmosférico y a que no permite el uso de ninguna técnica para aumentar la velocidad de transmisión.

Una ventaja es que los módulos que utilizan esta técnica tienen un bajo costo.

La técnica de modulación ASK es muy utilizada en sistemas de transmisión en fibra óptica.

*2.3.1.1.2. Modulación FSK (frequency shift keying)*

Es una técnica de modulación por desplazamiento de frecuencia. Es equivalente a la modulación FM pero más sencilla porque por ser digital la señal moduladora solo habrá una variación entre dos valores., es decir el bit 1L será representado por una frecuencia y el bit 0L será representado por una frecuencia de otro valor como se puede observar en la figura 2.6:

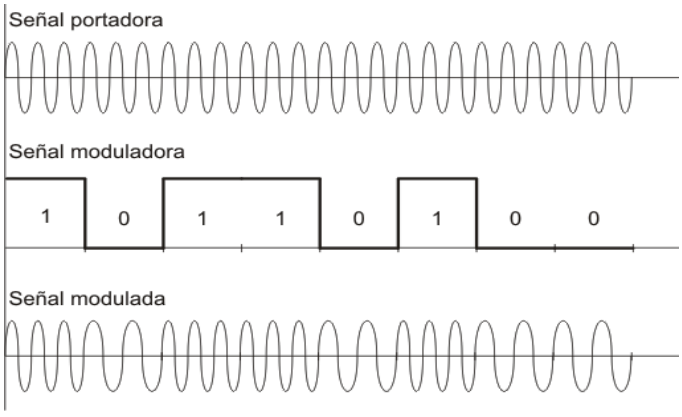


**FIGURA 2.6** Modulación FSK

El valor de la señal modulada está dado por la siguiente ecuación:

Cuando la señal moduladora tiene un valor 0 lógico

$$V(t) = Vp \cdot sen(2\pi \cdot f1 \cdot t)$$

Cuando la señal moduladora tiene un valor 1 lógico

$$V(t) = Vp \cdot sen(2\pi \cdot f2 \cdot t)$$

Los valores de f1 y f2 son desplazamientos de frecuencia de magnitudes iguales pero opuestas a la frecuencia de la señal portadora.

Entre las desventajas de esta técnica de modulación es la dificultad de su implementación razón por la cual los módulos que la usan tienen un costo más elevado.

Una ventaja es que al no variar su amplitud es menos vulnerable al ruido y no posee tantos errores como en la modulación ASK.

*2.3.1.1.3. Modulación GFSK (Gaussian Frequency Shift Keying)*

Esta técnica de modulación es una mejora de FSK. La diferencia está en que la información pasa por un filtro gausiano antes de ser modulada la señal. Esto permite tener un espectro más pequeño en la señal modulada lo que ayuda a obtener mayor velocidad de transmisión.

*2.3.1.1.4. Modulación PSK (Phase Shift Keying)*

Es una técnica de modulación por desplazamiento de fase. En esta técnica de modulación ocurre una variación en la fase cada vez que la señal digital de entrada varía entre 1L o 0L.

Al igual que en la técnica de modulación FSK esta posee una amplitud constante lo que la hace más segura contra el ruido.

En la siguiente figura 2.7 se puede observar esta técnica de modulación.

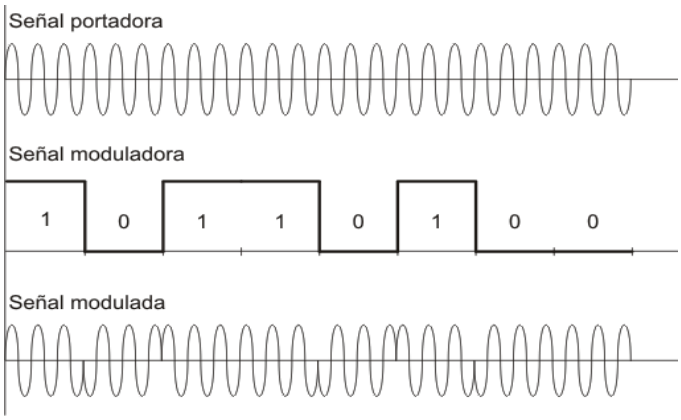


**Figura 2.7** Modulación PSK

El valor de la señal modulada está dado por la siguiente ecuación:

$$V(t) = Vp \cdot Vm \cdot cos(2\pi \cdot f \cdot t)$$

En donde:

Vm → Valor pico de la señal modulada

El valor de Vm es de 1 para 1 lógico y -1 para 0 lógico.

# CAPITULO 3

# 3. DISEÑO Y CONSTRUCCIÓN DEL SISTEMA TELEMETRICO

El sistema de telemetría diseñado permite transmitir la información generada en el vehículo de karting en el momento que se requiera hacia una PC que con el software, también diseñado para el sistema, procesará dicha información.

En el presente capitulo se diseñará y determinará los elementos que componen al sistema.

## 3.1. CARACTERÍSTICAS DEL MÓDULO DE TELEMETRIA

El dispositivo será diseñado para realizar las siguientes tareas:

- Obtener los datos de la posición del vehículo en la pista.
- Medir continuamente el nivel del combustible existente en el tanque del vehículo
- Tomar medidas de la temperatura a la que se encuentra el motor del vehículo.
- Realizar el conteo de las revoluciones por minuto del motor del vehículo
- Transmitir los datos generados por el vehículo hacia una computadora para procesarlos y visualizarlos en un HMI.

Para la realización de las tareas descritas, el diseño se basa en un microcontrolador, que será el núcleo que reciba y envíe la información en el tiempo necesario. La posición en la pista la obtendrá a partir de un GPS que estará todo el tiempo conectado y de esa forma entregue la posición más exacta posible. La transmisión la hará utilizando un transceptor de radiofrecuencia, el cual será descrito más adelante.

En la figura 3.1 se puede observar cómo está conformado el sistema de telemetría.

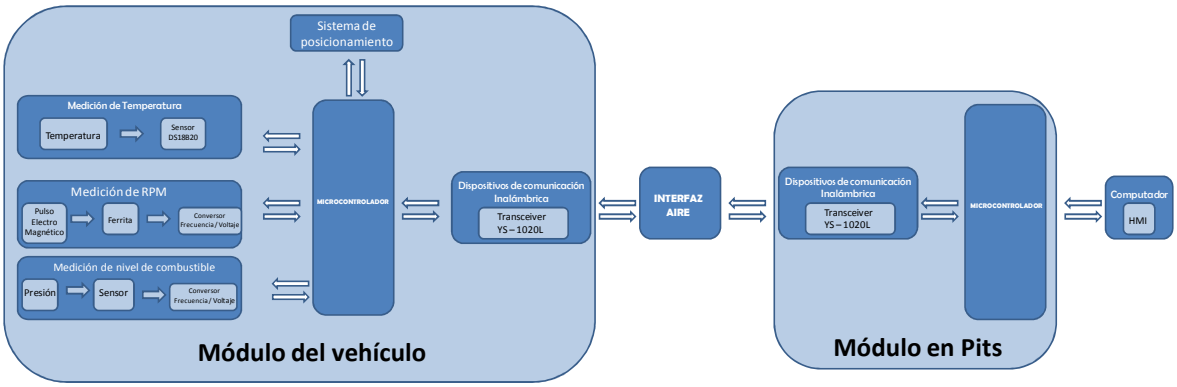


**Figura 3.1** Diagrama de bloques del sistema de telemetría

## 3.2. ELEMENTOS DEL SISTEMA DE TELEMETRIA

### 3.2.1. MICROCONTROLADOR

El microcontrolador es el cerebro de la aplicación ya que procesará la información obtenida del GPS, y los distintos sensores que serán descritos más adelante. Adicionalmente la información procesada la transmitirá de forma serial a un transceptor que estará conectado a uno de los puertos, de modo que la información procesada llegue hasta el módulo transceptor conectado a la PC.

Se determinó como una buena alternativa para el proyecto utilizar el microcontrolador AVR ATMEGA 164P debido a su bajo costo, la facilidad de encontrarlo en el mercado y el conocimiento de su manejo en comparación con los microcontroladores PIC también existentes en el mercado.

Adicionalmente el microcontrolador AVR ATMEGA 164P posee dos puertos para comunicación serial necesarios en la implementación del proyecto para la comunicación con el receptor GPS y el transceptor RF.

El microcontrolador AVR ATMEGA 164P posee una memoria RAM de 16Kbytes, memoria de programa de 128Kbytes y una memoria EEPROM de 4Kbytes. Memoria suficiente para la aplicación que se está implementando, ya que se

utilizarán aproximadamente 912 bytes de la memoria de programa y 82 bytes de la memoria de datos.

El microcontrolador posee un reloj interno capaz de generar 4, 8 o 16MHz con la circuitería externa sugerida por el fabricante la cual se encuentra en el manual del chip, pero al ser un reloj resistivo-capacitivo no es exacto y por esa razón para la aplicación que se está implementado se utilizará un cristal externo de 11.0592 MHz.

Características adicionales se las puede encontrar en el manual del chip adjunto en el anexo A.

### 3.2.2. DISPOSITIVO DE COMUNICACIÓN INALAMBRICA

De acuerdo a lo planteado para la realización del presente proyecto, se consideró el uso de módulos que trabajen en un rango de frecuencia comprendido entre 200 y 470MHz.

Los dispositivos de comunicación inalámbrica trabajarán a la intemperie y estarán instalados en el vehículo y en la computadora que estará ubicada en pits.

La pista de competencia Dos Hemisferios, lugar donde se realizan las competencias, posee una distancia aproximada de 1,5 km entre el vehículo ubicado en el punto más alejado de la pista y la computadora ubicada en pits.

El dispositivo de comunicación YS-1020L con una frecuencia de trabajo de 433MHz. es uno de los transceptores disponibles en el mercado cuyas características de potencia, alcance y costo, cumplen con los requerimientos para el proyecto.

Como se menciona en el párrafo anterior, módulo YS-1020L es un transceptor, es decir tiene integrado un transmisor y un receptor, razón por la cual utiliza comunicación half dúplex. El módulo está diseñado para un funcionamiento en interiores o exteriores.

Las principales características del módulo YS-1020L son:

- Utiliza modulación GFSK (GaussianFrecuency Shift keying)
- Funciona en un rango de temperaturas comprendido entre -3,5 y 75 ºC.
- Potencia de transmisión: 27 dBm
- Comunicación serial: 1200/2400/4800/9600/19200/38400bps
- Frecuencia portadora: 433 MHz multicanal
- Voltaje de trabajo: 5 V DC
- Alcance: ≤2Km (BER=$10^{-3}$ a 9600bps) y ≤3Km (BER=$10^{-3}$ a 200bps)
- Impedancia: 50Ù

En la figura 3.2 se puede observar el transceptor YS-1020L.

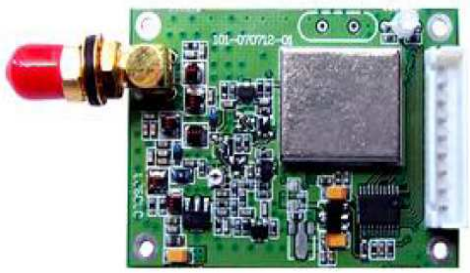

**Figura 3.2.** Transceptor YS-1020L

La distribución de pines del módulo se la puede encontrar en el manual dado por el fabricante, el cual se lo puede encontrar adjunta en el anexo B.

#### 3.2.2.1. Aplicaciones

Una de las razones, además de la frecuencia de trabajo, por las cuales se utiliza el módulo YS-1020L en el presente proyecto es por las aplicaciones para las cuales fue diseñado.

Las aplicaciones recomendadas por el fabricante son:

- Sistemas de telemetría
- Adquisición inalámbrica de datos para sistemas SCADA
- Display de LED’s inalámbricos
- Recolección de datos para sistemas inalámbricos de automatización de hogares
- Monitoreo remoto de sistemas
- Competiciones y eventos deportivos
- Recolección de datos en redes punto multipunto

### 3.2.3. SISTEMA DE POSICIONAMIENTO

El sistema de telemetría diseñado en el presente proyecto utiliza un sistema de posicionamiento conformado por un receptor GPS el cual será el encargado de entregar la posición del vehículo en la pista.

En el mercado se puede encontrar una variedad extensa de receptores GPS. Las características buscadas en el receptor GPS para el proyecto son:

- Salida serial de datos de posicionamiento.
- Bajo consumo de energía.

El sistema de telemetría diseñado utiliza un receptor GPS universal de la fábrica Deluo el cual posee las siguientes características:

**Figura 3.3** Receptor GPS Deluo

- 12 canales de recepción, que le permite recibir señales de 12 satélites simultáneamente.
- Frecuencia de recepción: 1575.42MHz.
- UART de 1 canal.
- Reloj de tiempo real interno (RTC).
- Convertidor A/D de 10 bits.
- RAM de 72 KB.
- ROM de 288 KB.

El receptor tiene una carcasa protectora que lo hace resistente al agua y a diferentes condiciones climáticas. Se llama receptor GPS universal porque la salida de datos de posicionamiento es serial a 4800 bps, por tanto puede utilizarse en muchas aplicaciones incluida la interacción con microcontroladores.
Posee un conector universal PS2. La distribución de pines se encuentra en el manual adjunto en el anexo C.

El consumo de corriente del receptor es relativamente bajo, un promedio de 80mA, se produce un pico de consumo de corriente de alrededor de 120mA cuando el dispositivo trata de encontrar los satélites, esto puede durar varios segundos e incluso minutos dependiendo de la cantidad de satélites a la vista.
Cuando el GPS se encuentra enganchado con tres satélites o más se tiene una precisión de ± 2m a ±30m. La identificación de estos satélites se guarda en la memoria no volátil del receptor, cada vez que se enciende el GPS éste trata de enlazarse usando la información de dicha memoria. Si se enciende nuevamente después de haber recorrido una distancia grande desde el último punto donde fue activado, el tiempo que se tarda en conectarse con los satélites se incrementará notablemente llegando incluso a los 15 minutos.

### 3.2.4. HMI

HMI (Human Machine Interface) es el término utilizado para referirse a la interfaz de usuario, es decir el medio con el cual el usuario se comunica con el equipo, ya sea mediante software o hardware.

En la actualidad se pueden encontrar algunos paquetes de software que permiten la adquisición de datos, procesamiento se señales, control de procesos ya que poseen herramientas que permiten realizar operaciones matemáticas avanzadas.

La facilidad que presenta el software de control LABVIEW gracias a la cantidad de herramientas que posee son la razón para que se haya escogido este paquete de software en el presente proyecto.

#### 3.2.4.1. Software de control LABVIEW

LabView (Laboratory Virtual Instrumentation Engineering Workbench) es un programa que fue creado en 1976 por National Instruments. Su diseño original fue para correr en los sistemas operativos de los computadores MAC pero actualmente tiene soporte para Windows, Linux y Mac.

Las aplicaciones de este software en la ingeniería son para la realización de pruebas, control y diseño utilizando programación en un lenguaje llamado lenguaje G debido a que la programación se la realiza con diagramas de bloques. La figura 3.4 muestra un ejemplo, en donde se observa la interfaz del programa así como la programación con bloques.

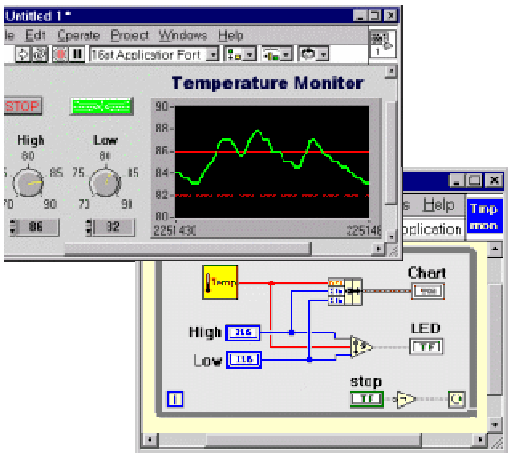


**Figura 3.4.** Programa de ejemplo realizado en LabView

### 3.2.4.2. Programación

La programación en LabView se dibuja y no se escribe, está herramienta gráfica ha sido diseñada de forma que simplifique el trabajo del programador. El software trae bloques y ciertas subrutinas prediseñadas con el fin de reducir el tiempo de creación de un proyecto. Los bloques son llamados VI (Virtual instruments), es decir instrumentos virtuales, están conformados por tres partes que son el panel frontal, el diagrama de bloques y los conectores.

#### 3.2.4.2.1. Panel Frontal

Es la interfaz con el usuario, sirve para interactuar con el usuario permitiéndole variar en tiempo real, las diferentes entradas de un proyecto y de ese modo saber la salida y así predecir el comportamiento de un sistema, en caso de que se esté realizando una prueba. En la figura 3.5 se puede observar un ejemplo del panel frontal de una aplicación.

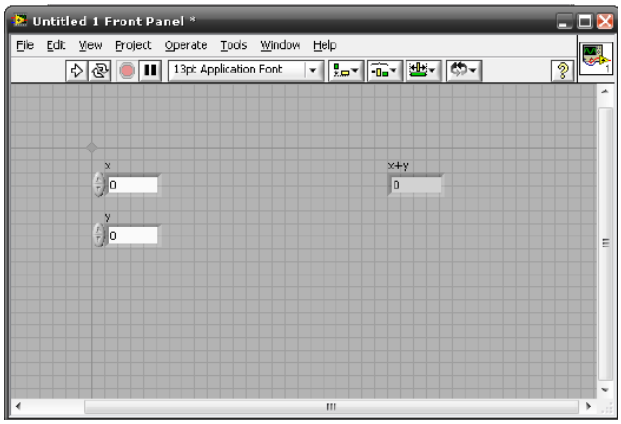


**Figura 3.5** Panel Frontal de LABVIEW

#### 3.2.4.2.2. Diagrama de bloques

Es el esquema del programa. Aquí el programador define la funcionalidad de cada una de las partes en el proyecto y es tan simple como arrastrar y posicionar en el lugar de trabajo el ícono que representa un instrumento o una operación según sea el caso.

La figura 3.6 muestra un ejemplo de Diagrama de bloques en Labview.

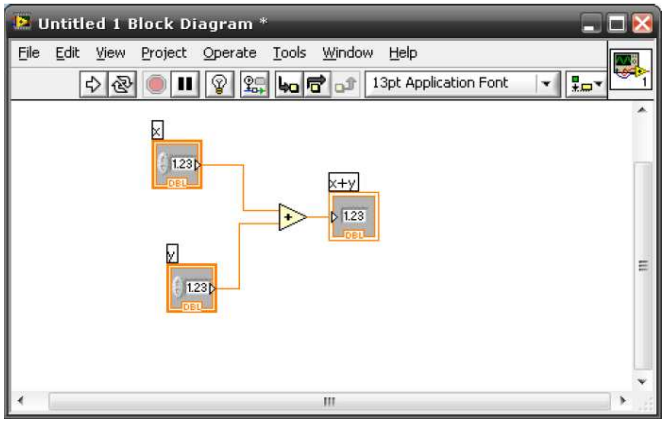


**Figura 3.6** Diagrama de Bloques en LABVIEW

Los bloques que conforman la programación en Labview se los puede unir con líneas llamadas conectores como se puede observar en la figura 3.6.

### 3.2.4.3. Ventajas

Además de ser un software de fácil manejo, inclusive para personas con poco conocimiento, permite la operación de proyectos con un alto grado de complejidad, que con otro tipo de lenguaje de programación sería imposible de realizarlo.

El paquete de software LabView permite el manejo de interfaces de comunicaciones como son los puertos seriales, puertos paralelos, USB, bluetooth, Irda, así como también permite el manejo de TCP/IP.

LabView está diseñado también para interactuar con otros paquetes computacionales como son: .Net, simulink de Matlab, Autocad entre otros.

También permite la visualización y manejo de gráficas con datos dinámicos y en tiempo real.

Posee adicionalmente herramientas gráficas para el procesamiento digital de señales.

### 3.2.4.4. Aplicaciones

Las aplicaciones de esta herramienta gráfica son sumamente extensas debido a la versatilidad que presenta para generar proyectos.

Entre las aplicaciones más importantes:

- Adquisición de datos
- Control de maquinaría e instrumentos
- Automatización en industrias
- Domótica
- Predicción de comportamiento de maquinaría

Una de las aplicaciones más importantes fue el uso de este software para controlar al acelerador de partículas más grande del mundo, perteneciente a la Organización Europea para la investigación nuclear (CERN), que fue construido en la frontera franco - suiza y con el cual se han realizado pruebas desde el 2008.

#### 3.2.4.5. Diseño de software

El software desarrollado en Labview tendrá dos funciones específicas en el presente proyecto:

- Interfaz gráfico con el usuario permitiendo visualizar la información enviada por los módulos.
- Almacenará los datos recibidos.

##### *3.2.4.5.1. Ventana Selección*

Es la ventana de menú del programa en donde se presentará las opciones siguientes para que el usuario pueda llamarlas:

- Nueva pista
- Nueva Carrera
- Carrera Guardada
- Terminar: Cierra la aplicación.

La figura 3.7 muestra el aspecto del menú Selección.

**Figura 3.7** Menu Selección

Los bloques que conforman el menú de selección se pueden observar en la figura 3.7.

**Figura 3.7** Diagrama de bloques del Menú Selección

### *3.2.4.5.2. Nueva pista*

Permite al usuario crear una nueva pista a partir de los datos obtenidos por el GPS que se encuentra en cada vehículo.

El usuario podrá observar la ventana que se muestra en la figura 3.8.

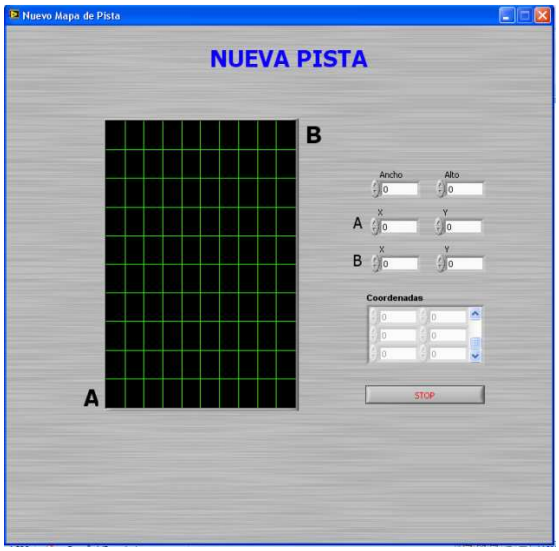


**Figura 3.8** Ventana de nueva pista

En la figura 3.9 se puede observar el diagrama de bloques del VI Nueva Pista. El bloque de Coordenadas son los datos de posición enviados por los receptores GPS instalados en cada vehículo. Los datos de posición son almacenados formando un arreglo de dos columnas X y Y. El valor de la variable N varía según el número de puntos almacenados.

El usuario debe ingresar previamente el ancho y largo aproximado de la pista, el punto inferior izquierdo y el punto superior derecho; estos valores son almacenados en los bloques correspondientes como se puede observar en la figura 3.9. Todos estos datos ingresan al sub VI “1” en donde son convertidos en puntos sobre el área de dibujo y pueden ser vistos por el usuario en la ventana Nueva Pista.

El Bloque A en conjunto con el Bloque B recuperan los datos del puerto serial para almacenarlos en la variable “Read Buffer”. De la trama obtenida se seleccionan los segundos, con los que obtiene el bloque Coordenadas.

Mediante el Bloque C con los datos ingresados se construye el archivo de texto con las coordenadas y se grafica la pista sobre el VI.

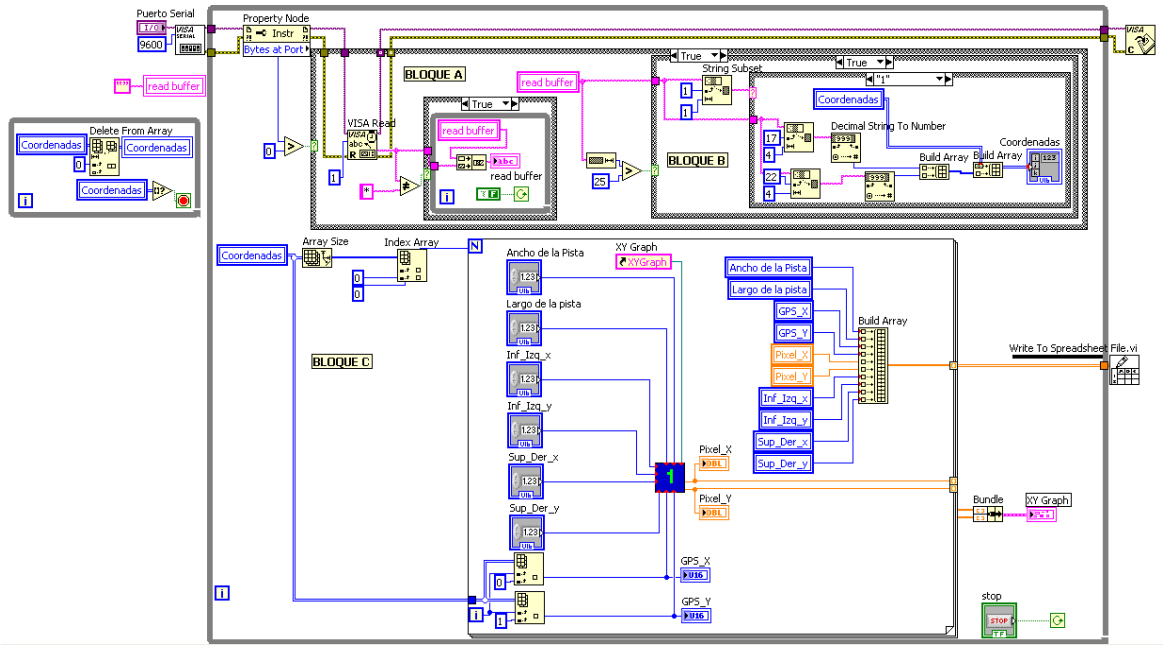


**Figura 3.9** Diagrama de bloques del VI Nueva Pista

El sub VI “1” es el encargado de acondicionar los datos de posición entregados por cada módulo instalado en los vehículos, de acuerdo al tamaño aproximado de la pista haciendo una relación entre el valor de coordenada ingresada y un punto sobre el area de dibujo.

El diagrama de bloques del sub VI “1” se puede observar en la figura 3.10.

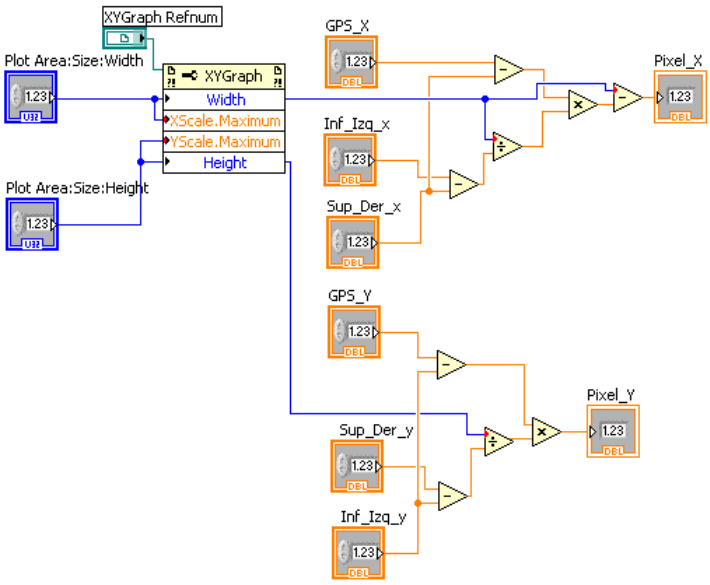


**Figura 3.10** Diagrama de Bloques del sub VI “1”

### *3.2.4.5.3. Nueva Carrera*

Permite al usuario visualizar en tiempo real la información generada en cada vehículo, la cual es enviada por los módulos instalados en los mismos. Esta información será almacenada en un archivo de texto.

Al seleccionar Nueva Carrera en el menú de selección el usuario podrá observar la figura 3.11.

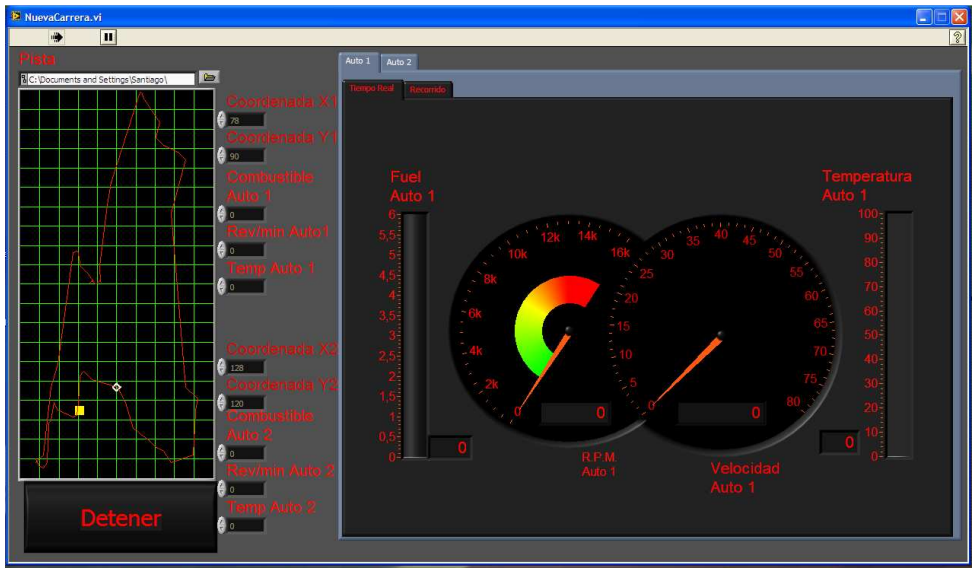


**Figura 3.11** Ventana de Nueva Carrera

Los diagramas de bloques que conforman el VI Nueva Carrera se puede observar a continuación en las figuras 3.12, 3.13 y 3.14.

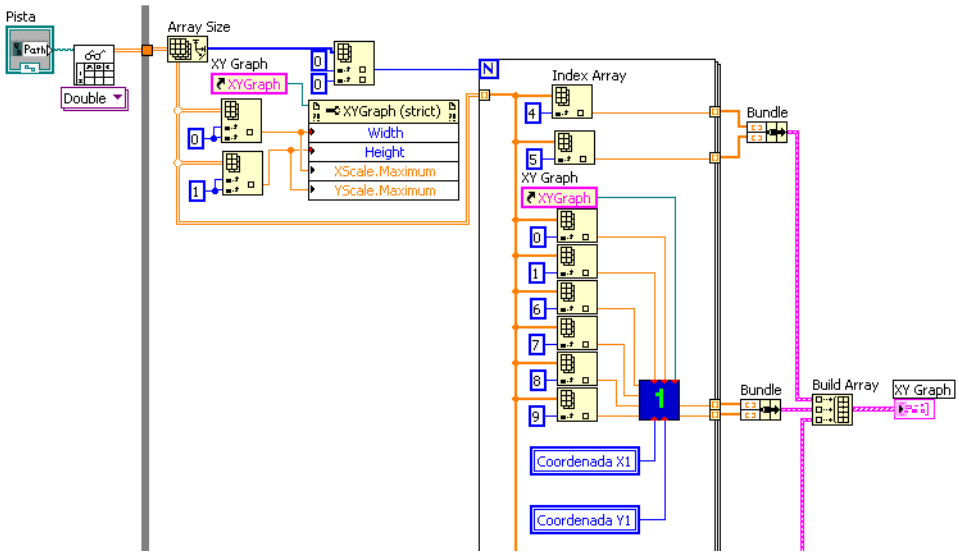


**Figura 3.12** Diagrama de Bloques VI Nueva Carrera

El diagrama de bloques de la figura 3.12 almacena las posiciones enviadas por cada vehículo y las posiciona sobre la pista entregando de esa forma una posición relativa del vehículo en la pista.

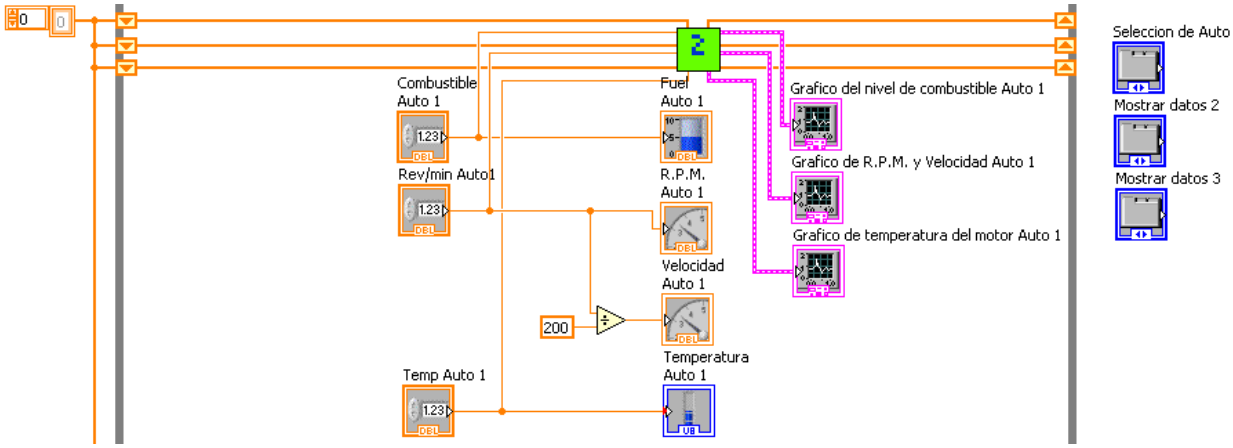


**Figura 3.13** Diagrama de Bloques VI Nueva Carrera

El diagrama de bloques de la figura 3.13 es la etapa del VI “Nueva Carrera” en donde se van almacenando y mostrando al usuario los datos generados por los dos vehículos.

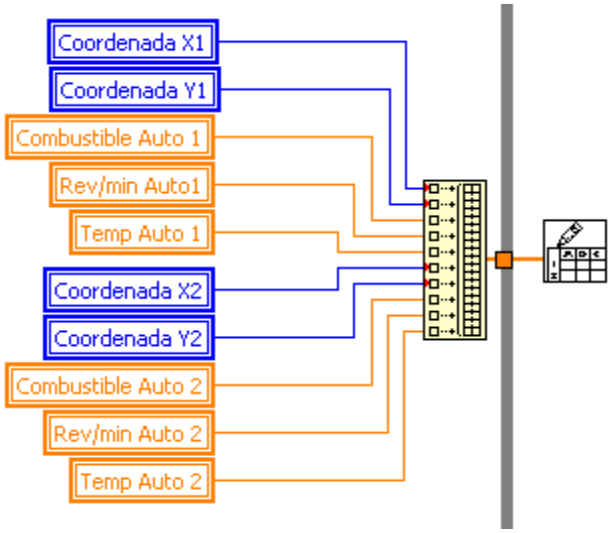


**Figura 3.14** Diagrama de Bloques VI Nueva Carrera

La figura 3.14 muestra los bloques mediante los cuales los datos, entregados por los vehículos, se almacenan en un archivo de texto para su futura reproducción o análisis.

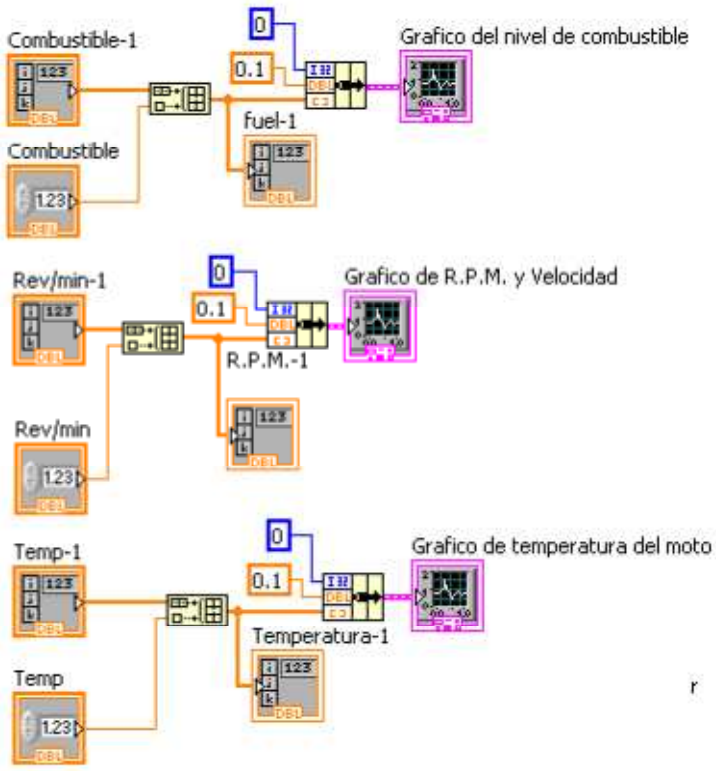


**Figura 3.15** Diagrama de bloques del Sub-VI "2"

La figura 3.15 es el diagrama de bloques del Sub VI "2" (Ver figura 3.13) el cual permitirá observar las gráficas en función del tiempo de las variables de velocidad y R.P.M, nivel de combustible y temperatura del motor presentes en la selección "Nueva Carrera" y "Carrera Guardada".

### *3.2.4.5.4. Carrera Guardada*

Permite simular una carrera en base a datos previamente almacenados para su respectivo análisis.

Al seleccionar el usuario en el menú selección "Carrera Guardada" podrá observar la interfaz que muestra la figura 3.16 en donde se puede observar adicionalmente las figuras generadas por el sub – VI "2".

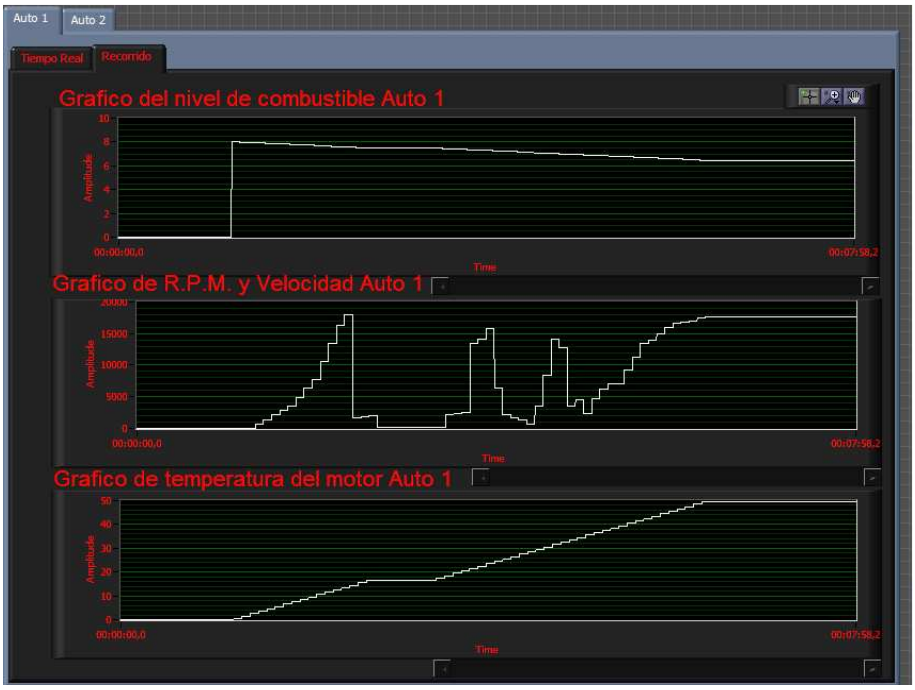


**Figura 3.16** Gráficos de las variables almacenadas en la carrera

El diagrama de bloques para esta selección se muestran en las figuras 3.17 y 3.18. Los cuales constituyen los bloques en donde el programa carga los datos a partir de dos archivos, el de los datos generados por los vehículos y el archivo de la pista.

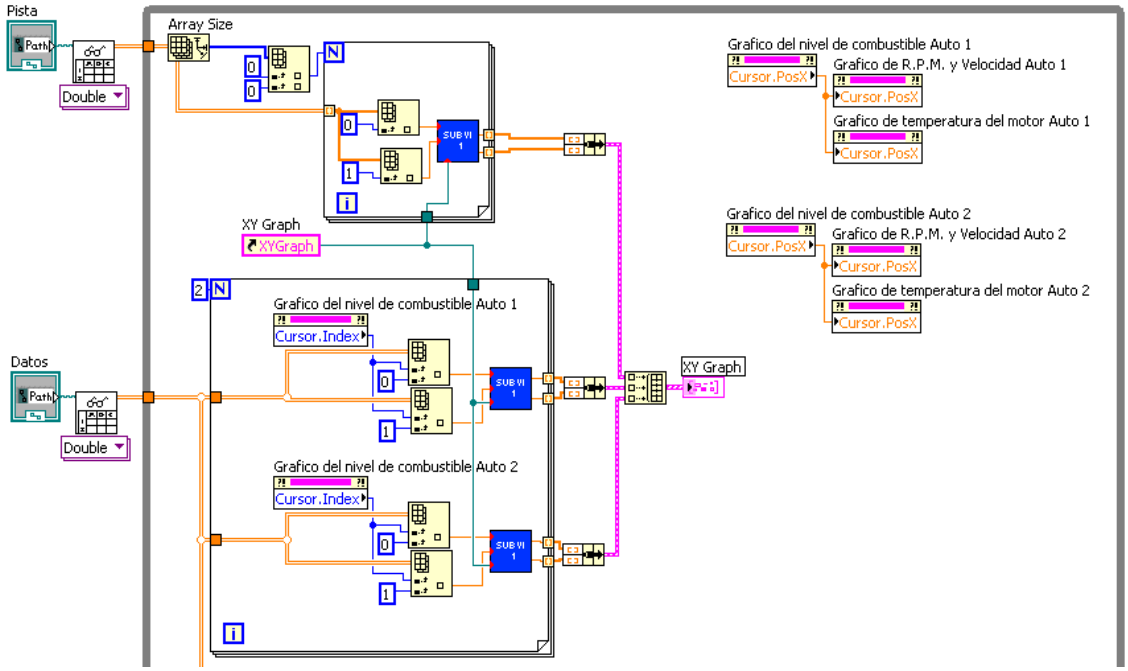


**Figura 3.17** Diagrama de bloques del VI Carrera Guardada

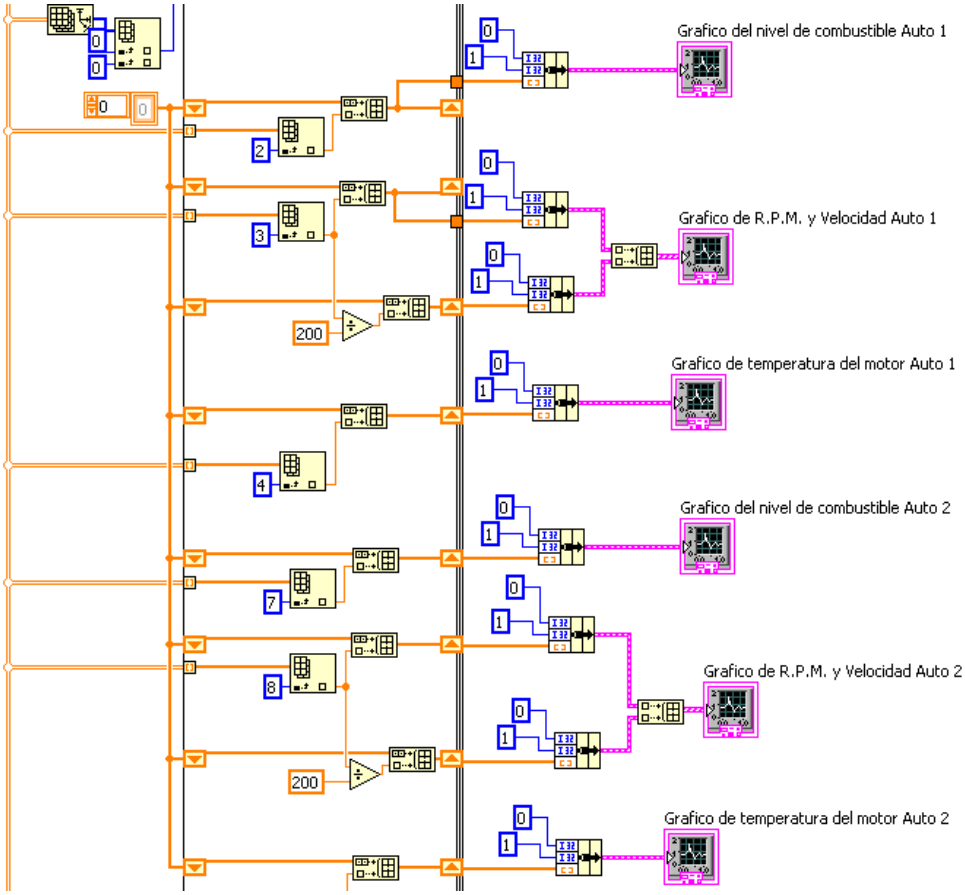


**Figura 3.18** Diagrama de bloques del VI carrera guardada

La interfaz permitirá que el usuario con el cursor pueda posicionarse sobre cualquier punto de la pista obteniendo los valores de nivel de combustible, temperatura del motor, R.P.M y velocidad, tomados para el punto que el usuario desea observar.

### 3.2.5. MEDICIÓN DE TEMPERATURA

Como parte de la realización del proyecto se encuentra el diseño del sistema para le medición de la temperatura del motor del vehículo de karting. La figura 3.19 muestra el diagrama de bloques del sistema de medición de temperatura.

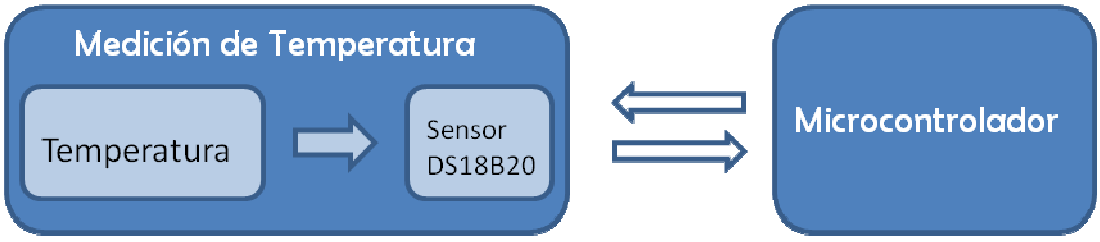


**Figura 3.19** Diagrama de bloques del sistema de medición de temperatura

La medición de la temperatura del motor del vehículo se la tomará del block[2] del motor como se observa en la figura 3.20.

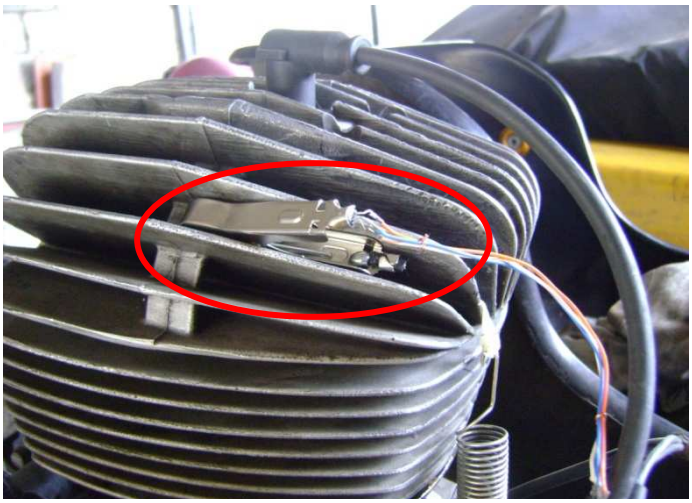

**Figura 3.20** Ubicación del sensor en el motor

El sensor DS18B20 de temperatura utiliza un solo pin del microcontrolador debido a que maneja el protocolo 1-wire.

A continuación se hará una descripción del sensor y el protocolo.

### 3.2.5.1. Sensor de temperatura DS18B20 1- Wire.

El sensor a utilizarse para tomar la temperatura del motor del vehículo es el DS18B20 fabricado por Maxim.

Posee una precisión de 9 a 12 bits, que en temperatura equivale hasta 0.0625ºC, con un rango de operación de -55ºC hasta 125ºC, rango dentro del cual está la temperatura generada por el motor del vehículo. Para la aplicación que se va a realizar se utilizarán 9 bits, de modo que se tendrá una precisión de 0.5℃.

Adicionalmente los datos binarios entregados por el sensor son independientes del voltaje de polarización. El voltaje de polarización recomendado por el fabricante es de 3V hasta 5.5V.

La figura 3.21 es una imagen del encapsulado que se utilizará en el sistema de medición de temperatura.

[2] Llamado también bloque del motor, es una pieza de hierro, o aleaciones, que aloja los cilindros, cigüeñal y demás elementos del motor.

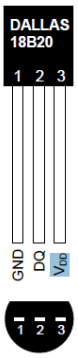


**Figura 3.21.** Sensor de temperatura DS18B20

#### *3.2.5.1.1. Protocolo 1-WIRE*

Es un protocolo de comunicaciones que permite realizar transmisión y recepción de señales entre un equipo maestro y varios esclavos en forma serial asincrónica y bidireccional, para ello usa solo un cable junto con la tierra. Por ello se dice que solo requiere de una terminal de entrada/salida compartida entre varios equipos 1-Wire, los cuales tienen una única identificación de direccionamiento, la cual consiste en un número de 64 bits que está grabada en la memoria ROM. El primer byte representa el código de la familia del tipo del dispositivo, los siguientes 6 bytes suministran 48 bits que son usados para generar el número único de cada dispositivo, generando así $2^{48}$ posibles identificaciones de dispositivos, lo que se convierte en una ventaja al hacer más fácil la selección de los dispositivos conectados en la red; finalmente el byte más significativo contiene un CRC con un valor basado en los datos de los 7 primeros bytes, con lo que el maestro puede determinar si una dirección fue leída sin errores.

### 3.2.5.2. Ventajas

Los dispositivos que usan el protocolo 1-Wire tienen las siguientes ventajas:

- Pueden ser formateados como diskettes, así como el acceso a los archivos es de una forma aleatoria sin afectar a los otros archivos guardados en dichos dispositivos.
- Puede funcionar en modo de alimentación parásita, lo que significa que toma energía del bus de datos mientras este se encuentra en uno lógico. Dicha energía es almacenada en un capacitor embebido para posteriormente ser utilizada para su propia alimentación cuando el bus se encuentra transportando datos.

### 3.2.5.3. Comunicación a nivel de bits

El dispositivo maestro inicia y controla la comunicación con los equipos esclavos. La forma de onda es similar a una modulación por ancho de pulso, debido a que los datos son transmitidos con pulsos anchos para los 0 lógicos y pulsos angostos para los 1 lógicos a través de slots de tiempo.

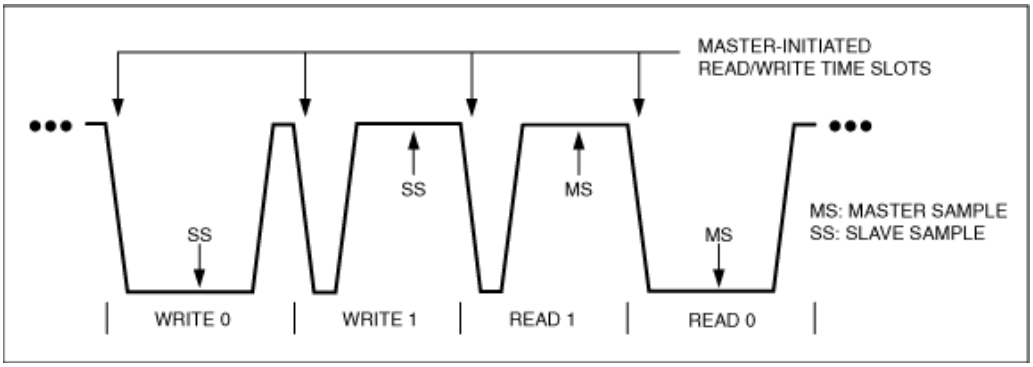


***Figura 3.22*** *Lectura/escritura de bits de datos con muestreo del maestro y esclavo*

Los comandos y datos son enviados bit por bit para formar bytes, empezando con el bit menos significante (LSB) la lectura o escritura de la información. El maestro y esclavo pueden ser transmisores y receptores de datos pero solo en una dirección al mismo tiempo, por lo que la comunicación es half duplex. Un sistema de reloj no es requerido debido a que cada elemento esclavo 1-Wire tiene internamente un oscilador que se sincroniza con el del maestro cada vez que en la línea de datos haya un flanco de bajada.

El Protocolo 1-Wire es una secuencia de transiciones de información, la que se desarrolla siguiendo el siguiente orden:

- Inicialización
- Comandos y funciones de ROM
- Comando y funciones de control y memoria
- Transferencia de bytes o datos

### 3.2.5.4. Utilización del sensor de temperatura DS18B20

Para la utilización del sensor DS18B20 que es un sensor digital de temperatura, el mismo que a parte de una resistencia de pull-up de 4.7k no necesita circuitería adicional, su manejo se hace mediante secuencias digitales con protocolo 1-wire, las mismas que se enviarán y recibirán mediante el microcontrolador. La figura 3.23 permite observar el circuito recomendado por el fabricante.

El diagrama 3.1 permite observar el funcionamiento y la interacción del sensor DS18B20 con el microcontrolador.

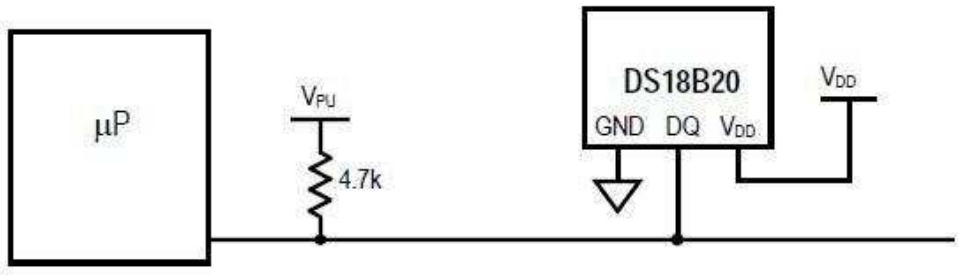


**Figura 3.23** Circuito para el sensor DS18B20

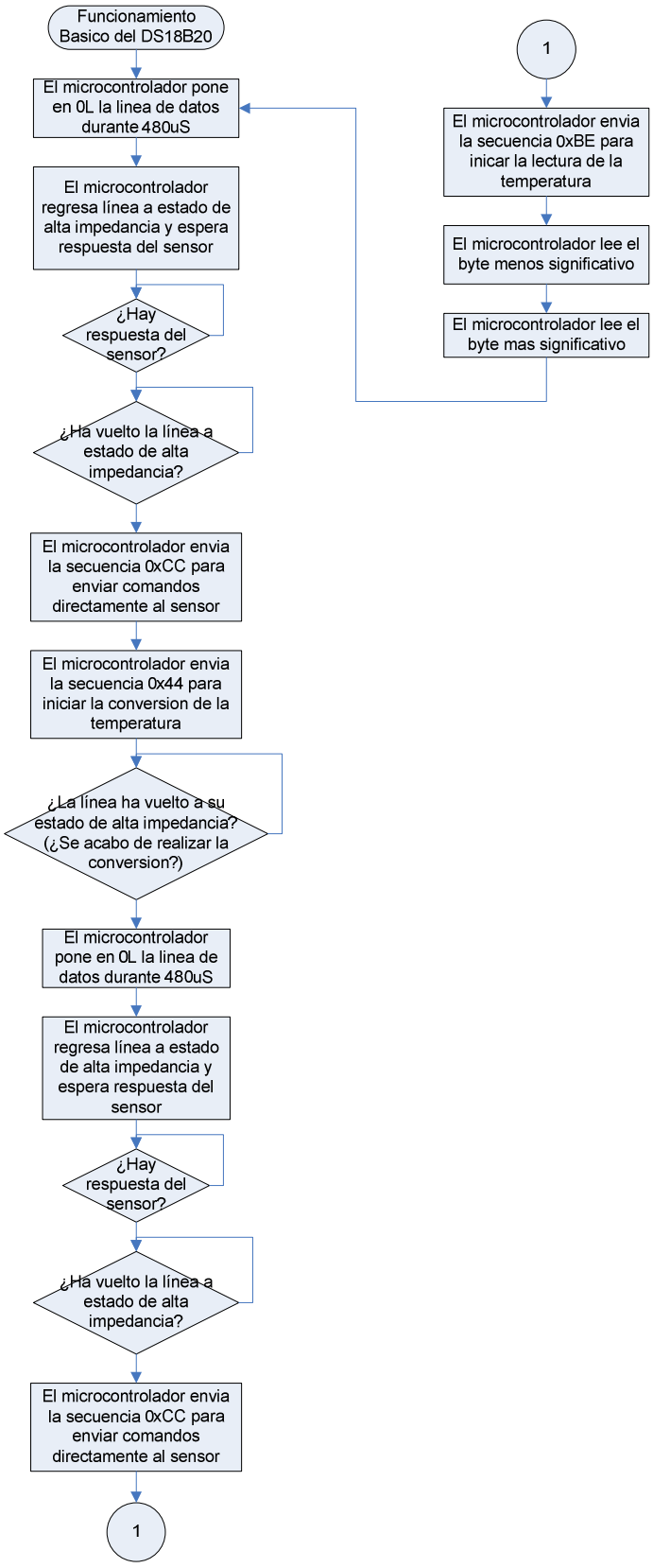


**Diagrama 3.1** Diagrama de flujo del medidor de temperatura

### 3.2.6. MEDICIÓN DE REVOLUCIONES POR MINUTO

El siguiente paso en la realización del presente proyecto se trata de la medición de las revoluciones por minuto del motor del vehículo.

La figura 3.24 muestra el diagrama de bloques del sistema diseñado para la adquisición de las revoluciones por minuto del motor.

**Figura 3.24** Diagrama de Bloques del sistema de medición de R.P.M.

Para obtener esta variable el sistema diseñado utilizará un sensor compuesto por una bobina con núcleo ferromagnético (ferrita) el cual estará ubicado sobre el cable de la bujía como se puede observar en la figura 3.25.

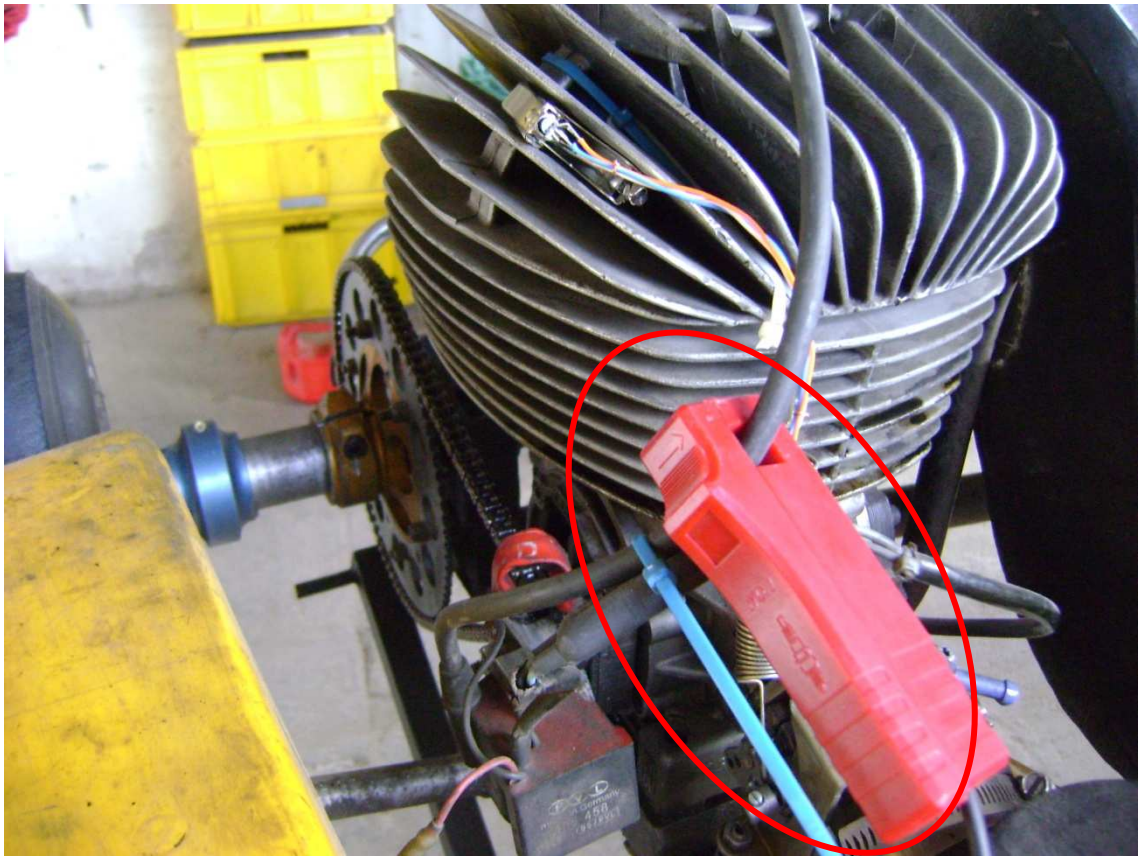

**Figura 3.25** Ubicación del sensor del sistema de medición de R.P.M

La señal entregada por el sensor tiene componentes de voltaje DC y AC. La componente AC puede llegar alcanzar una frecuencia máxima de 300Hz que corresponde a 18000 R.P.M. La figura 3.26 muestra la forma de la señal vista en el osciloscopio.

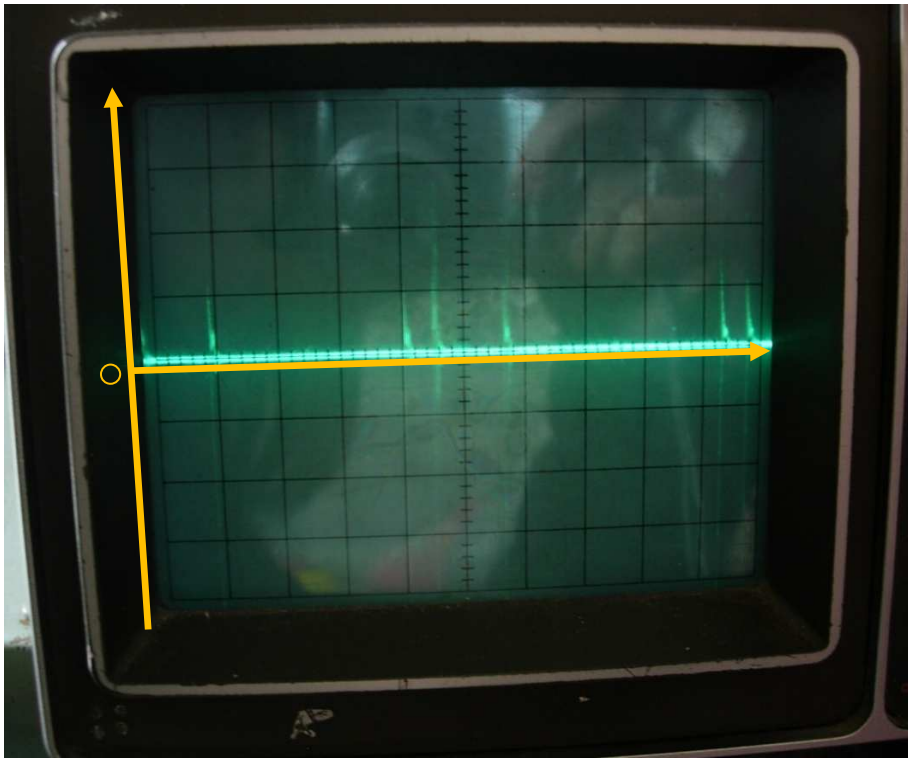

**Figura 3.26** Señal Obtenida a la salida del sensor para la medición de R.P.M

Para apreciar la señal observada en la figura 3.26 las escalas en el osciloscopio fueron establecidas de la siguiente manera: En el eje Y a 1V por división y en el eje X a 1ms por división.

Para el acondicionamiento de la señal obtenida se utilizará el conversor de frecuencia en voltaje LM2917.

### 3.2.6.1. Conversor LM2917

Es un conversor diseñado para utilizar poca circuitería externa. La señal de entrada puede variar entre 0 y 30KHz y entrega una salida de voltaje proporcional a la frecuencia siendo 0V. para 0Hz. La salida de voltaje corresponde a la siguiente expresión:

$$V_0 = F_{in} \times V_{cc} \times R_1 \times C_1 \times K$$

**Ecuación 3.1**

En donde:

$V_{cc}$ es el voltaje de polarización

Fin es la frecuencia de la señal de entrada

La frecuencia máxima se la obtiene de la siguiente expresión:

$$F_{max} := \frac{I_2}{C_1 \cdot Vcc}$$

**Ecuación 3.2**

En donde:

$F_{max}$ es la frecuencia máxima de la señal de entrada

$I_2$ es la corriente en el pin 2

La frecuencia máxima de la señal obtenida para la aplicación que se está diseñando es de 300Hz, como se mencionó anteriormente.

De las expresiones anteriores se obtienen los valores de $C_1$ y $R_1$.

El valor de $I_2$ se la obtiene de la gráfica de $I_2$ vs Fuente de voltaje la cual se la puede observar en la figura 3.27. Curvas de trabajo adicionales se las puede encontrar en el manual del conversor adjunto en el Anexo D.

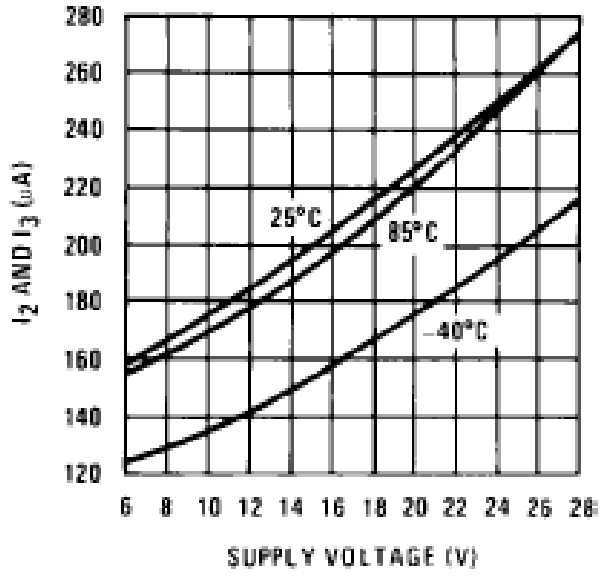


**Figura 3.27** Gráfica de $I_2$ vs Fuente de Voltaje

La distribución de pines del conversor se la puede observar en la figura 3.28

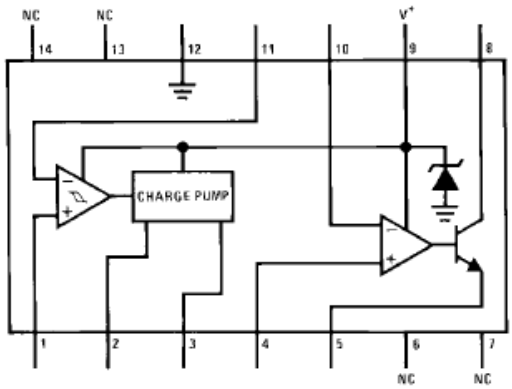


**Figura 3.28** Distribución de pines conversor LM2917

### 3.2.7. ACONDICIONAMIENTO DEL SENSOR DE R.P.M

El circuito del conversor fue diseñado para una señal de entrada con una frecuencia máxima de 300Hz para la cual se obtendrá un voltaje máximo de salida de 4.5V.

Los cálculos para el diseño que se presentan a continuación fueron obtenidos a partir de las ecuaciones 3.1 y 3.2. El voltaje de polarización a utilizarse es de 12V, pero en los cálculos se utiliza 7,5V debido a un diodo zener en el interior del conversor LM2917 usado para regular el voltaje de polarización.

Dado:

fmax=300Hz Vcc=7,5V $I_2$=160µA K=1 $V_{omax}$=4V

$C_1$= 7.11 x $10^{-8}$ F

$R_1$=2.539 x $10^4$ Ω

Los valores estándar de resistencia y capacitancia seleccionados son:

$R_1$= 24 kΩ

$C_1$= 70 nF

En la figura 3.29 se puede observar el circuito dado por el fabricante en donde se incluyen los valores obtenidos de los elementos en el diseño.

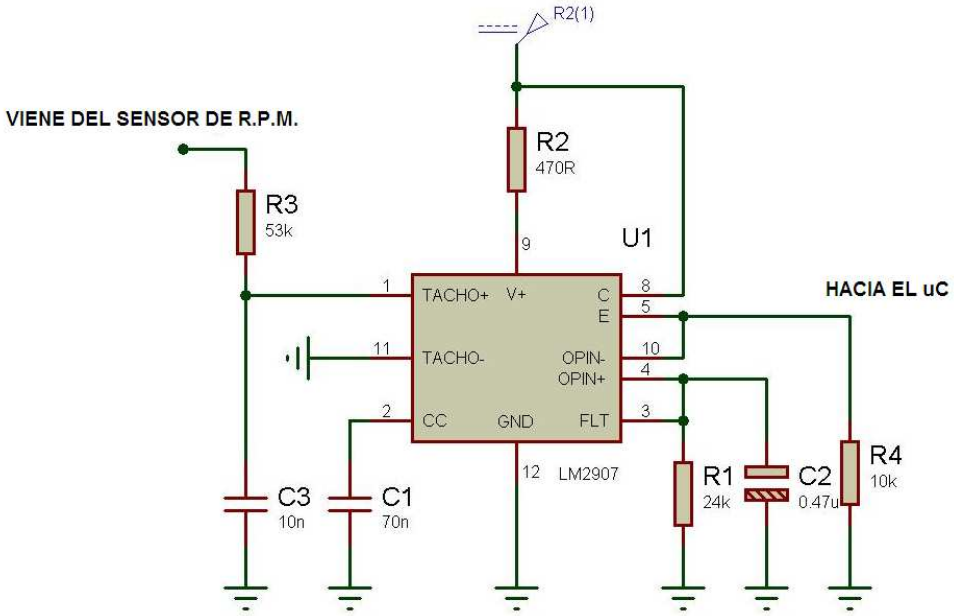


**Figura 3.29** Acondicionamiento del sensor de R.P.M

El valor de $C_2$ se determinó a partir de la ecuación 3.3 para que con una variación de frecuencia desde 0 hasta 300 Hz y considerando una constante de tiempo de dos cientos milisegundos se obtenga una salida de voltaje comprendida entre cero y 4.4V ($V_{max}$), el cálculo se muestra a continuación:

$\tau := R_1 \cdot C_2$ Ecuación 3.3

$5 \cdot \tau := 1 \cdot sec$

$$\tau = 1s$$

$$C_2 = \frac{0.2s}{R_1} = 8.33 \times 10^{-6} F$$

El valor estándar de $C_2$ seleccionado es de 10 uF.

Considerando a R3 y C3 un filtro pasabajos de primer orden y asumiendo que C3=0.01uF se obtiene R3 de la siguiente manera:

$C_3 := 0.01 \cdot \mu F$

$$R_3 := \frac{1}{2 \cdot \pi \cdot 300 \cdot Hz \cdot C_3} = 5.305 \times 10^4 \, \Omega$$

Por lo tanto el valor estándar escogido para R3 es de 53KΩ.

### 3.2.8. MEDICIÓN DE NIVEL DE COMBUSTIBLE

La figura 3.30 permite observar los bloques que conforman el sistema de medición de nivel de combustible.

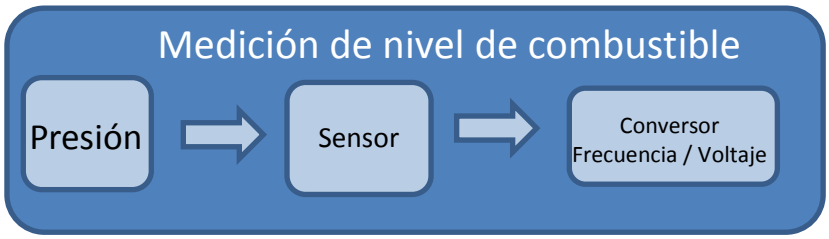


**Figura 3.30.** Diagrama de Bloques del sistema de medición del Nivel de Combustible

El sistema de medición de nivel de combustible utiliza un sensor de nivel de agua utilizado en lavadoras Samsung. La señal entregada por el sensor es una señal con componente AC y DC, en donde el valor aproximado de la componente continua es de 5V y el voltaje pico pico aproximado de la componente alterna está alrededor de 100mV.

La componente alterna de la señal posee una frecuencia que varía entre 20KHz y 25KHz razón por la cual se utilizará el conversor de frecuencia a voltaje LM2917 de manera similar que en el sistema de medición de R.P.M. La figura 3.28 muestra la forma de la señal obtenida a la salida del sensor. Las escalas del osciloscopio estaban establecidas en 0.2V por división en el eje vertical, 20µs en el eje horizontal y tenía activado el acoplamiento AC.

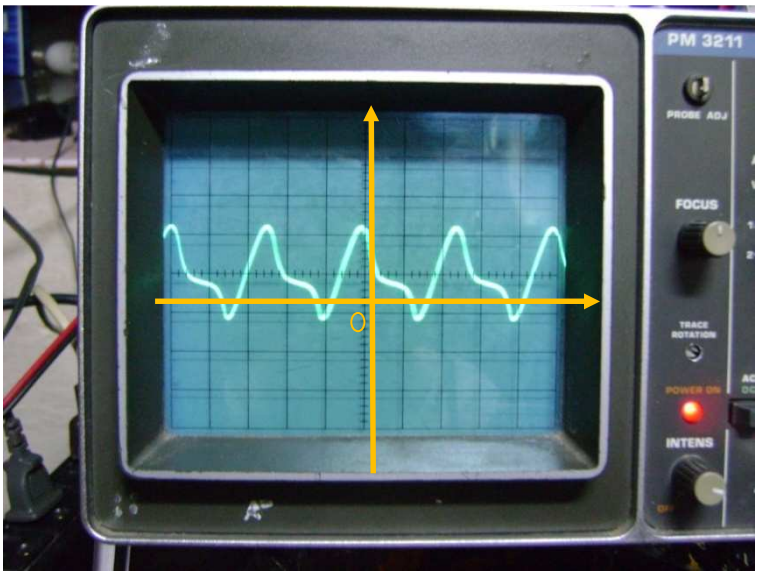


**Figura 3.31** Salida del sensor de Nivel

Para poder trabajar adecuadamente con el sensor de nivel fue necesario realizarle algunas pruebas de laboratorio en donde se pudo obtener la tabla 3.1.

| Altura (cm) | Voltaje (V) |
|---|---|
| 4.2 | 3.97 |
| 5.4 | 3.95 |
| 6.9 | 3.96 |
| 7.6 | 3.95 |
| 8.7 | 3.94 |
| 9.9 | 3.91 |
| 11 | 3.9 |
| 12 | 3.88 |
| 13.2 | 3.87 |
| 14.2 | 3.87 |
| 15.2 | 3.86 |
| 16.4 | 3.83 |
| 17.6 | 3.82 |
| 18.8 | 3.82 |
| 20 | 3.81 |
| 21 | 3.8 |
| 22 | 3.79 |
| 23.2 | 3.78 |
| 24.6 | 3.77 |
| 26.2 | 3.76 |
| 28.7 | 3.71 |

**Tabla 3.1** Datos de comportamiento del circuito de medición de nivel de combustible

La prueba consistió en llenar un tanque de combustible de 5 litros en porciones de 200cm$^3$. La frecuencia variaba mínimamente en la prueba razón por la cual se tomo en la salida de voltaje del conversor de frecuencia a voltaje LM2917.

La figura 3.32 muestra la curva característica del bloque de medición de nivel de combustible.

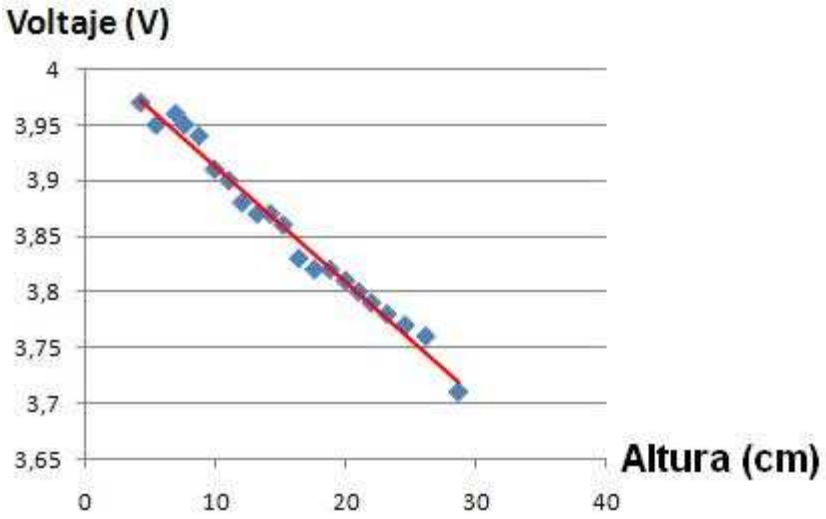


**Figura 3.32** Curva característica del bloque de medición de nivel de combustible

La curva característica del bloque de medición de combustible es casi lineal, razón por la cual al interpolar los puntos se puede obtener el valor de la pendiente de la misma.

### 3.2.8.1. Acondicionamiento del Sensor de nivel

La circuitería externa del sensor fue diseñada para una señal con una frecuencia máxima de 25KHz para la cual se obtendrá un voltaje de salida máximo de 4.41V.

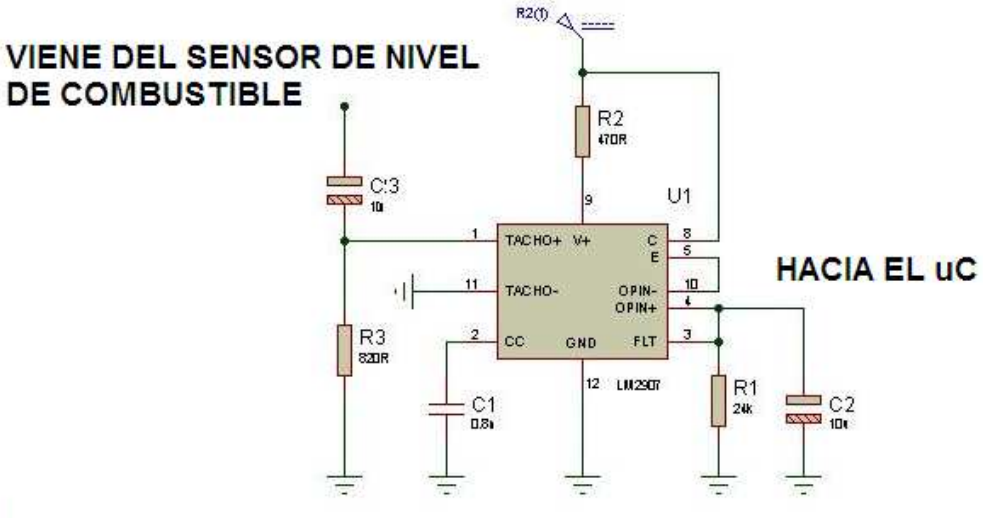


**Figura 3.33** Acondicionamiento del Sensor de nivel

La circuitería externa del conversor al igual que para el sistema de obtención de R.P.M. fue obtenida a partir del circuito recomendado por el fabricante.

El condensador $C_3$ y la resistencia $R_3$ funcionan como un filtro pasa alto para eliminar la componente continua de la señal y dar paso únicamente a la señal alterna.

El valor de $C_2$ se determinó a partir de la expresión 3.3 para que con una variación desde 20KHz hasta 25KHz se obtenga una salida de voltaje comprendida entre 0 y 4.41 ($V_{max}$). Se diseñó para una constante de tiempo de 15 segundos considerando el movimiento que tendrá el combustible debido a las vibraciones producidas por el vehículo cuando este se encuentre en competencia. Además se considera que el nivel de combustible es una variable que generalmente no cambia rápidamente.

Dado:

$f_{max}$= 25 KHz  Vcc=7,5V  $I_2$=160uA  K=1

Utilizando las expresiones 3.1 y 3.2 se obtiene los siguientes valores estándar:

$C_1$=1nF  $R_1$=24KΩ

Con la expresión 3.3 y la constante de tiempo de 15 segundos se obtiene un valor de capacitancia de $C_2$= 100uF

### 3.2.9. DISEÑO DEL TRANSCEPTOR EN PITS

El receptor en pits es el encargado de obtener la información generada por cada uno de los vehículos solicitándola cada cierto tiempo o cuando el usuario la requiera para que la información sea procesada por la PC y el usuario la pueda observar mediante la HMI realizada en Labview descrita anteriormente.

El diagrama de bloques del transceptor se lo puede observar en la figura 3.34.

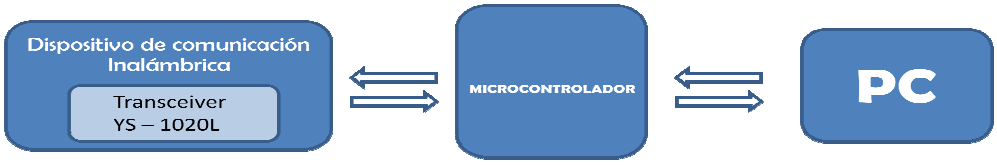


**Figura 3.34** Diagrama de Bloques del transceptor

El esquema del circuito del receptor en pits se lo puede observar en la figura 3.35 el cual básicamente consta de un microcontrolador y dos interfaces para comunicarse con los transceptores y la PC.

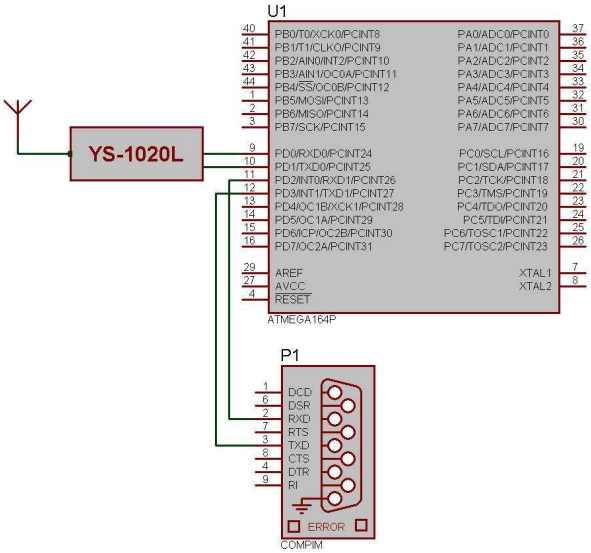


**Figura 3.35** Esquema del circuito del transceptor en pits

### 3.2.10. DISEÑO DE LA TRAMA DE DATOS

Los datos a enviarse entre los módulos instalados en los vehículos hacia el transceptor instalado en la PC de pits consta de: 8 bits de dirección, 174 bits de datos y 8 bits utilizados para realizar control de errores.

La figura 3.36 muestra la trama de datos.

| Dirección | , | Temperatura | , | Nivel de Combustible | , | R.P.M | , | Longitud | , | Latitud | , | CRC5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 8 | 24 | 8 | 24 | 8 | 24 | 8 | 32 | 8 | 32 | 8 | 8 |

**Figura 3.36** Trama de datos

Los datos se enviarán separados por comas de modo que la trama total enviada será de 200 bits, los cuales se transmitirán a una velocidad de 9600bps. Cada trama será enviada tres veces, una a continuación de otra creando un sistema de redundancia de modo que el receptor compare entre las tres tramas recibidas y asi mediante un esquema de mayoría gana, recuperar la trama original.

La tabla 3.2 muestra la respuesta al utilizar el esquema de mayoría gana.

| F2 | F1 | F0 | **R** |
|---|---|---|---|
| 0 | 0 | 0 | **0** |
| 0 | 0 | 1 | **0** |
| 0 | 1 | 0 | **0** |
| 0 | 1 | 1 | **1** |
| 1 | 0 | 0 | **0** |
| 1 | 0 | 1 | **1** |
| 1 | 1 | 0 | **1** |
| 1 | 1 | 1 | **1** |

**Tabla 3.2**

Con el mapa de Karnaugh resolvemos el sistema obtenido en la tabla 3.2.

| F0 \ F2 F1 | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| **0** | O | O | 1 | 0 |
| **1** | 0 | 1 | 1 | 1 |

**Tabla 3.3** Mapa de Karnaugh

La respuesta al sistema es la expresión 3.4

$$R=F_2\,F_0 + F_1\,F_0 + F_1\,F_2$$

**Ecuación 3.4**

### 3.2.11. CONSTRUCCIÓN DEL MÓDULO DE TELEMETRÍA.

La tarjeta que conforma el módulo de telemetría instalado en el vehículo se puede observar en la figura 3.37 y la tabla 3.4 muestra en detalle los elementos que la conforman.

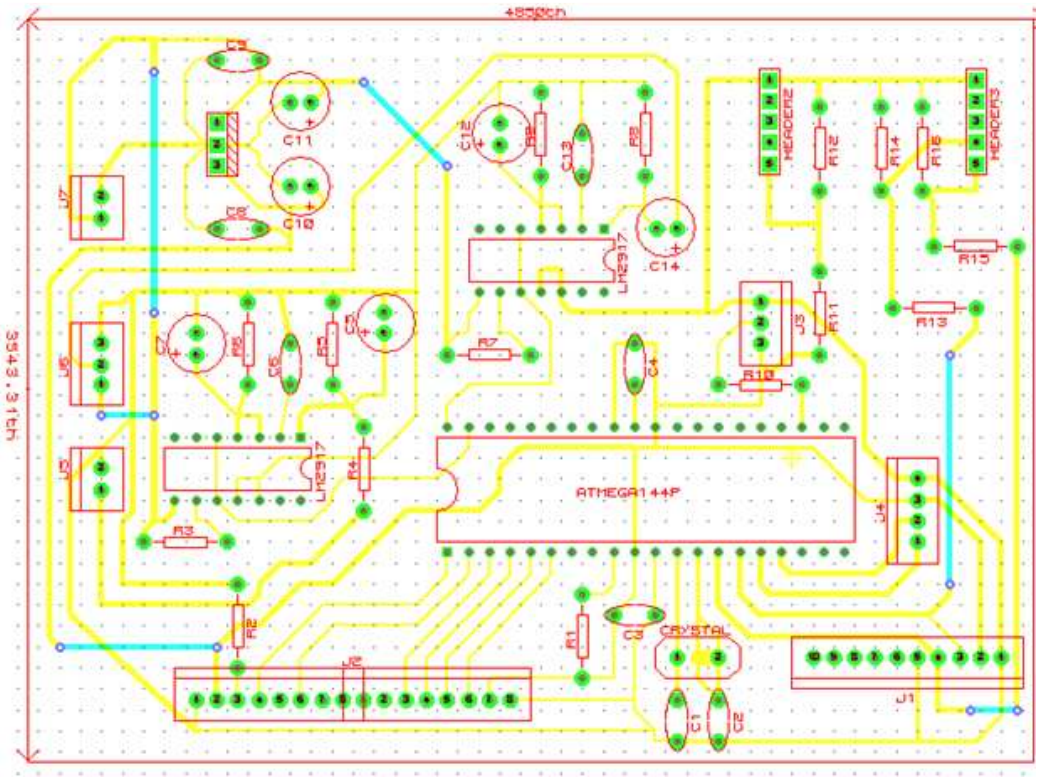


**Figura 3.37** Imagen del circuito impreso del módulo instalado en el vehículo

| Capacitores | Resistencias |
|---|---|
| C1 = 22p | R1 = 10k |
| C2 = 22p | R2 = 2.2k |
| C3 = 100n | R3 = 470 |
| C4 = 100n | R4 = 10k |
| C5 = 0.01u | R5 = 22k |
| C6 = 47n | R6 = 39k |
| C7 = 100u | R7 = 470 |
| C8 = 100n | R8 = 22k |
| C9 = 100n | R9 = 27k |
| C10 = 10u | R10 = 4.7k |
| C11 = 10u | R11 = 1.8k |
| C12 = 10u | R12 = 3.3k |
| C13 = 1n | R13 = 1.8k |
| C14 = 1u | R14 = 3.3k |
| | R15 = 1.8k |
| | R16 = 3.3k |
| | |
| Microcontrolador | Atmega 164p |
| Cristal | 1MHz |

**Tabla 3.4** Elementos de la tarjeta del módulo de telemetría

El voltaje de polarización como se dijo anteriormente es de 12V razón por la cual sobre ningún elemento caerá un voltaje mayor. Actualmente la tarjeta posee capacitores con tolerancia de 25V ya que en el mercado hay mucha disponibilidad de este tipo de capacitores.

La figura 3.38 es una imagen del módulo instalado en el vehículo de karting antes de la realización de las pruebas.

**Figura 3.38** Módulo Instalado en el vehículo

Como se puede observar en la figura 3.38 el módulo de telemetría va instalado en la parte posterior del asiento del piloto de modo que para el piloto sea imperceptible su presencia.

# CAPITULO 4

# 4. PRUEBAS Y RESULTADOS

En el presente capítulo se presentarán los resultados de las pruebas de funcionamiento del módulo diseñado para el sistema telemétrico. Las pruebas consisten en evaluar el funcionamiento de cada uno de los bloques que conforman el hardware así como el funcionamiento del HMI realizado en Labview 2009.

## 4.1. PRUEBA DE FUNCIONAMIENTO DEL RECEPTOR GPS

En esta prueba se coloca el receptor GPS en la pista ubicada cerca de la Mitad del Mundo la cual posee línea de vista con los satélites.

Para obtener los datos provenientes del GPS se utiliza el programa Hyperterminal configurando la comunicación con los parámetros que se pueden observar en la tabla 4.1.

| Parámetros | Valores |
|---|---|
| Velocidad de Tx. | 4800 bps |
| Bits de datos | 8 |
| Paridad | Ninguna |
| Bit de parada | 1 |

**Tabla 4.1**

Las sentencias NMEA entregadas al receptor GPS Deluo son $GPGSA, $GPGSV y $GPGMC se pueden observar en la figura 4.1. y cuyo detalle se lo puede observar en el anexo E.

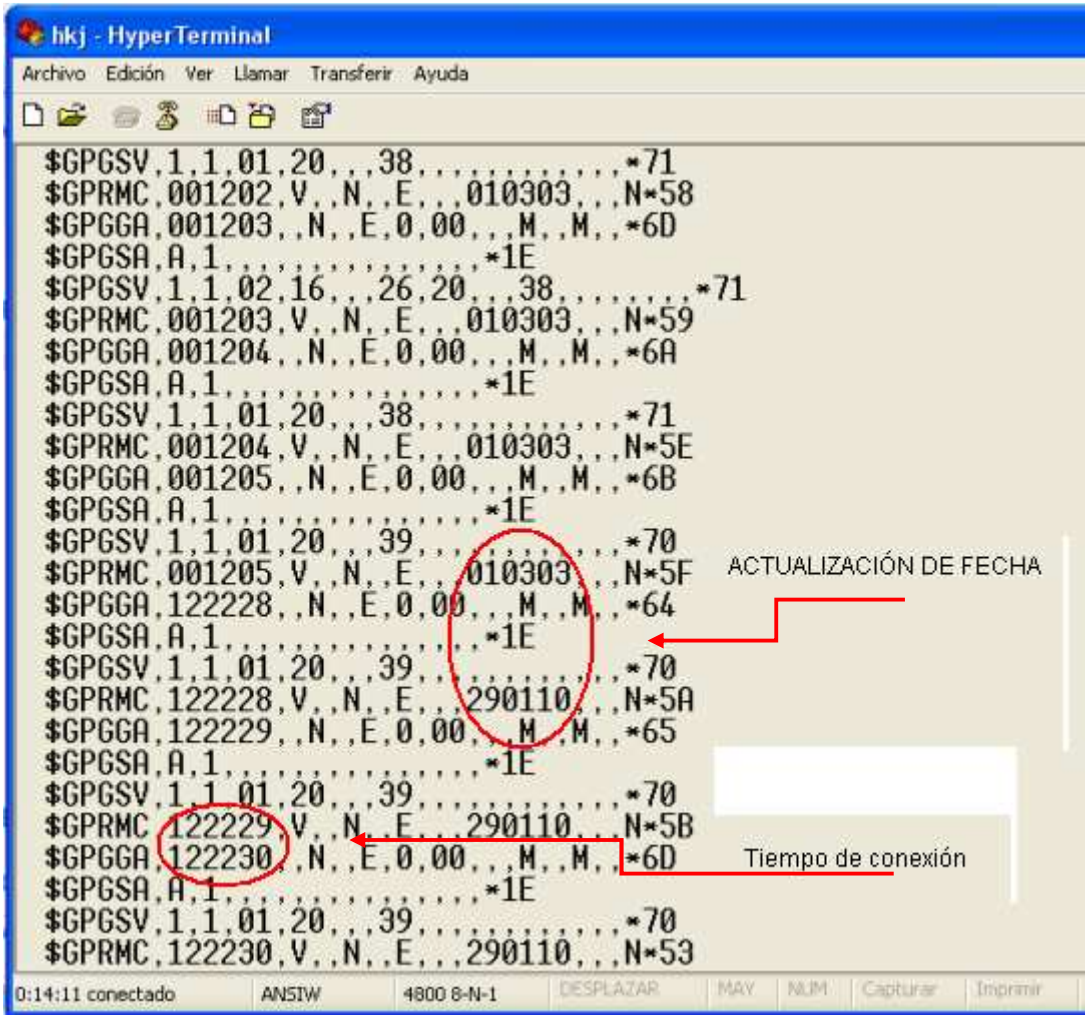


**Figura 4.1** Sentencias NMEA entregadas por el receptor GPS

El receptor GPS tardó aproximadamente doce minutos en poder llegar a FIX después del arranque en frio.

La sentencia que se utiliza para adquirir los datos de posición es la sentencia NMEA $GPRMC que contiene la hora, fecha, posición y el estado de la información. El receptor se configuró de modo que entregue solo esta sentencia y en intervalos de un segundo

Para la prueba del receptor de GPS se diseñó un VI en Labview para adquirir las tramas del mismo y almacenarlas en un archivo de texto.

La figura 4.2 muestra el VI diseñado para la prueba.

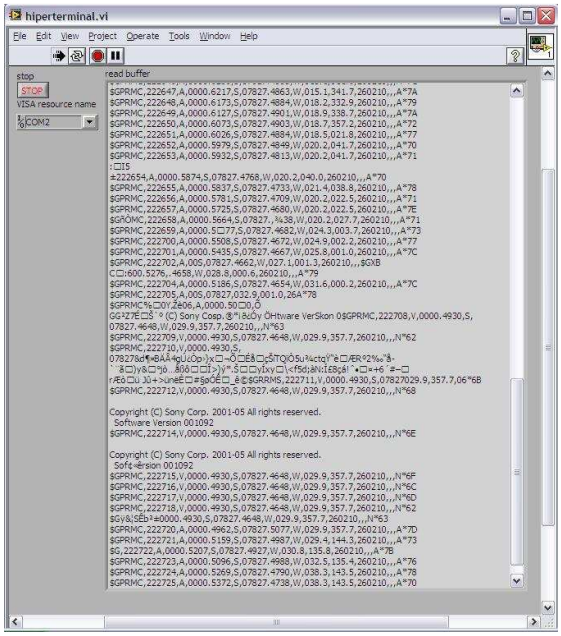


**Figura 4.2** VI para pruebas del receptor GPS

El archivo de texto generado en la prueba entregó la forma de un circuito de la pista “Dos Hemisferios” la cual se puede observar en la figura 4.3.

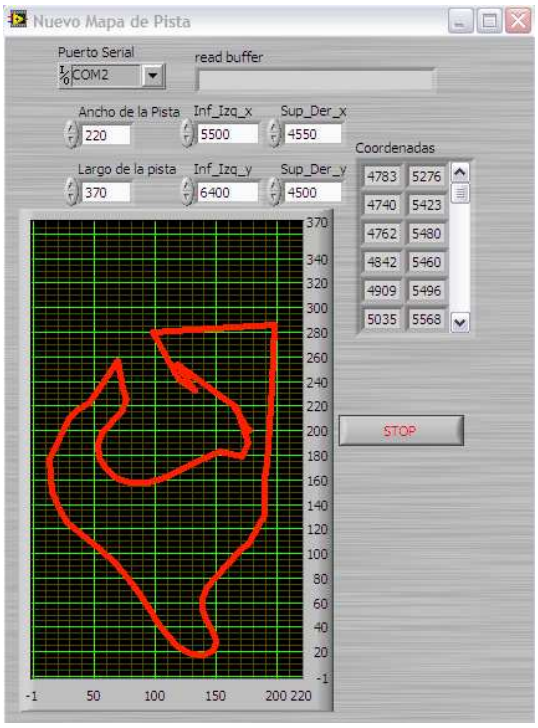


**Figura 4.3** Contorno del Kartódromo “Dos Hemisferios”

## 4.2. PRUEBA DE FUNCIONAMIENTO DEL MEDIDOR DE LAS R.P.M DEL VEHÍCULO

Para la realización de la prueba del medidor de R.P.M. se instalaron en el vehículo el módulo de telemetría diseñado en el presente proyecto y un dispositivo de adquisición de temperatura, velocidad y R.P.M de marca ALFANO. La tabla 4.2 contiene los valores comparativos entre el módulo diseñado para el proyecto y el dispositivo de adquisición de datos ALFANO.

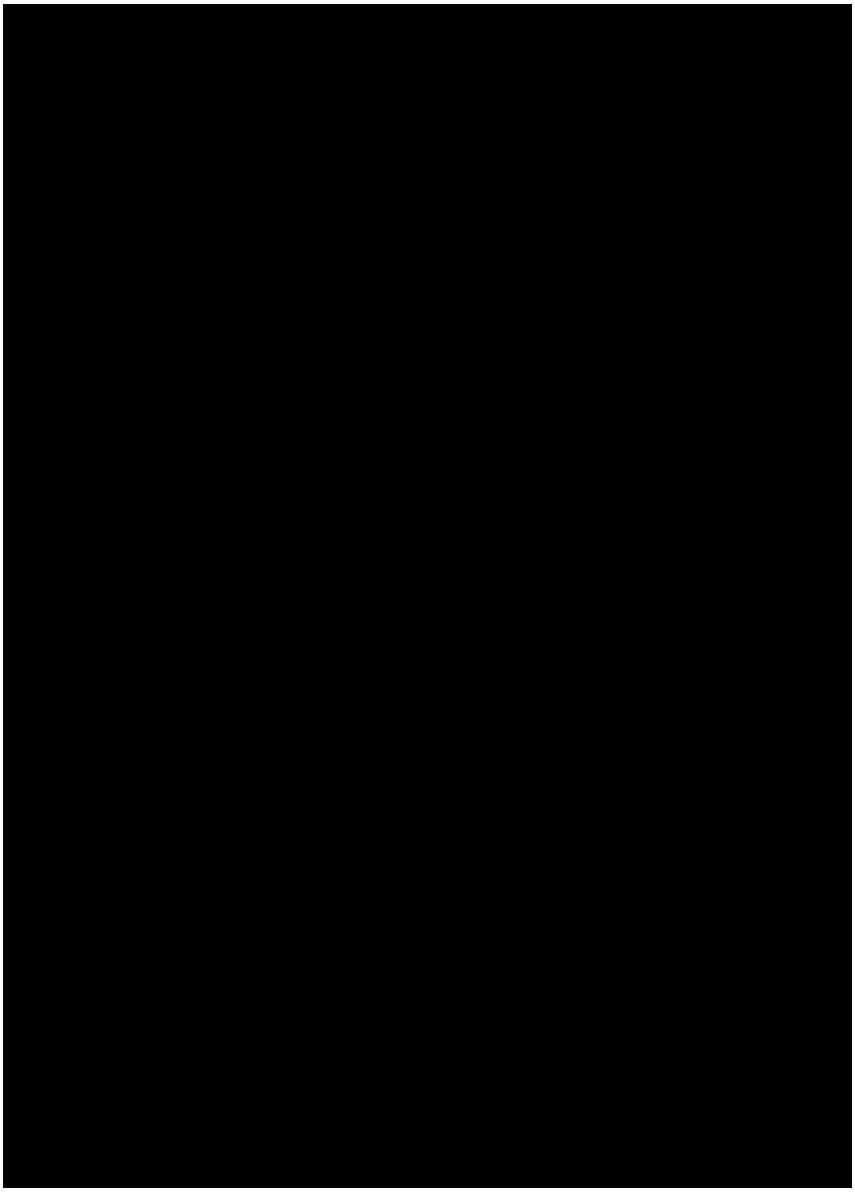

**Tabla 4.2** Mediciones comparativas de R.P.M

La figura 4.4 permite observar las mediciones comparativas del los datos de R.P.M adquiridos en la prueba realizada.

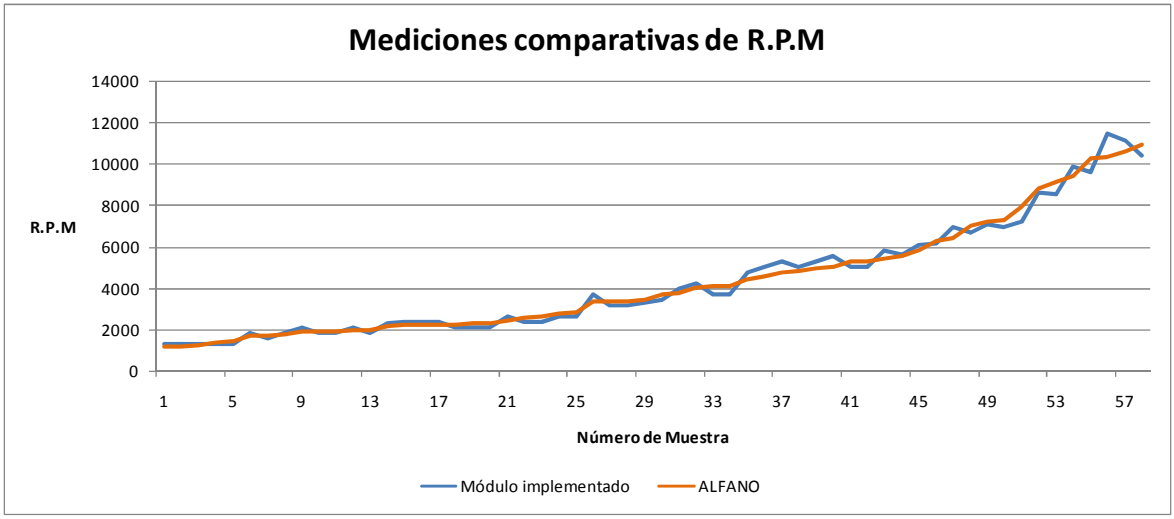


**Figura 4.4** Mediciones Comparativas de R.P.M

De los valores obtenidos en la tabla 4.2 se obtiene que el valor promedio del error es de 6.91% valor que no es crítico considerando que es una variable que en el tiempo varia constantemente; además que para el caso más crítico (12000R.P.M.) representan ±720 R.P.M de error, que en comparación con los instrumentos de aguja existentes en el mercado, los cuales tienen una precisión de 200 R.P.M, representa solamente tres líneas de error.

## 4.3. PRUEBA DE FUNCIONAMIENTO DEL BLOQUE DE MEDICIÓN DE NIVEL DE COMBUSTIBLE

Para la realización de la prueba del sistema de medición de combustible se instaló el sensor en un envase de 5 litros dentro del cual se fue variando la cantidad del líquido hasta llenarlo en su totalidad. Con la prueba se tomaron los valores de la altura del líquido y la salida de voltaje del circuito. Dicho valores se encuentran en la tabla 4.3 con la que se obtuvo la gráfica de comportamiento del circuito.

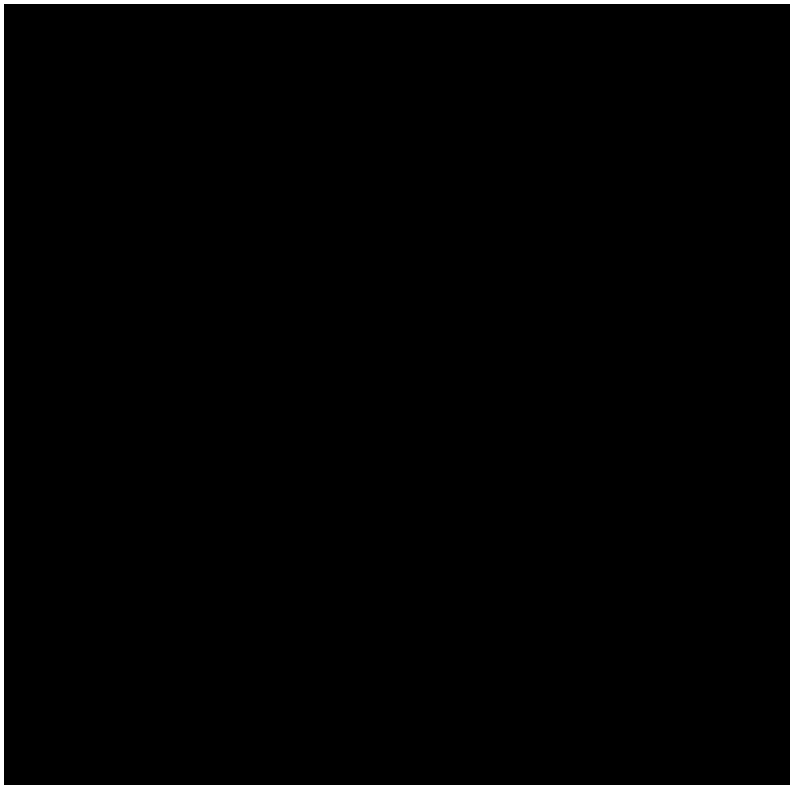

**Tabla 4.3** Mediciones comparativas de nivel de combustible

En la tabla 4.3 se obtiene que el error promedio es del 4,33%. El valor obtenido es aceptable debido a que no representa la cantidad necesaria de combustible para dar una vuelta en el circuito más extenso de la pista “Dos Hemisferios”.

La figura 4.5 permite observar las mediciones comparativas de nivel de combustible obtenidos en la prueba realizada.

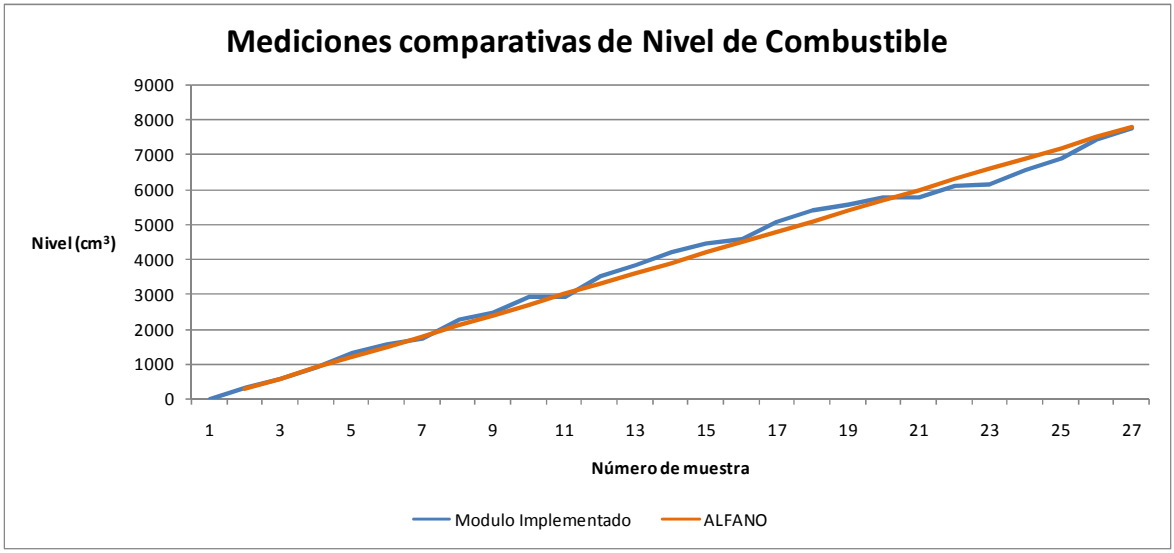


**Figura 4.5** Mediciones comparativas de nivel de combustible

## 4.4. PRUEBA DE FUNCIONAMIENTO DE LA MEDICIÓN DE VELOCIDAD

La prueba de medición de velocidad se la realizó en conjunto con las mediciones de R.P.M de modo que se las comparó de igual manera con el medidor de velocidad del dispositivo ALFANO. Los resultados comparativos de la prueba se pueden observar en la tabla 4.4.

| Número de Muestra | Modulo Implementado | ALFANO |
|---|---|---|
| 1 | 5 | 5 |
| 2 | 5 | 5 |
| 3 | 5 | 5 |
| 4 | 5 | 5 |
| 5 | 5 | 5 |
| 6 | 7 | 6 |
| 7 | 7 | 6 |
| 8 | 6 | 6 |
| 9 | 7 | 7 |
| 10 | 7 | 7 |
| 11 | 7 | 7 |
| 12 | 8 | 7 |
| 13 | 8 | 8 |
| 14 | 8 | 8 |
| 15 | 9 | 8 |

| Número de Muestra | Modulo Implementado | ALFANO |
|---|---|---|
| 16 | 10 | 9 |
| 17 | 8 | 9 |
| 18 | 9 | 9 |
| 19 | 8 | 9 |
| 20 | 9 | 9 |
| 21 | 9 | 9 |
| 22 | 9 | 10 |
| 23 | 9 | 10 |
| 24 | 12 | 11 |
| 25 | 12 | 11 |
| 26 | 10 | 11 |
| 27 | 11 | 12 |
| 28 | 13 | 12 |
| 29 | 13 | 12 |
| 30 | 14 | 13 |

| Número de Muestra | Modulo Implementado | ALFANO |
|---|---|---|
| 31 | 14 | 13 |
| 32 | 14 | 15 |
| 33 | 14 | 15 |
| 34 | 15 | 16 |
| 35 | 19 | 17 |
| 36 | 19 | 17 |
| 37 | 16 | 17 |
| 38 | 19 | 18 |
| 39 | 18 | 19 |
| 40 | 22 | 20 |
| 41 | 19 | 20 |
| 42 | 20 | 21 |
| 43 | 19 | 21 |
| 44 | 20 | 22 |

| Número de Muestra | Modulo Implementado | ALFANO |
|---|---|---|
| 45 | 21 | 23 |
| 46 | 25 | 23 |
| 47 | 22 | 24 |
| 48 | 26 | 24 |
| 49 | 23 | 24 |
| 50 | 27 | 26 |
| 51 | 26 | 28 |
| 52 | 30 | 29 |
| 53 | 32 | 30 |
| 54 | 29 | 31 |
| 55 | 36 | 33 |
| 56 | 39 | 35 |
| 57 | 37 | 39 |
| 58 | 43 | 47 |

**Tabla 4.4** Mediciones comparativas de Velocidad

La figura 4.6 permite observar las mediciones comparativas de velocidad obtenidos en la prueba realizada.

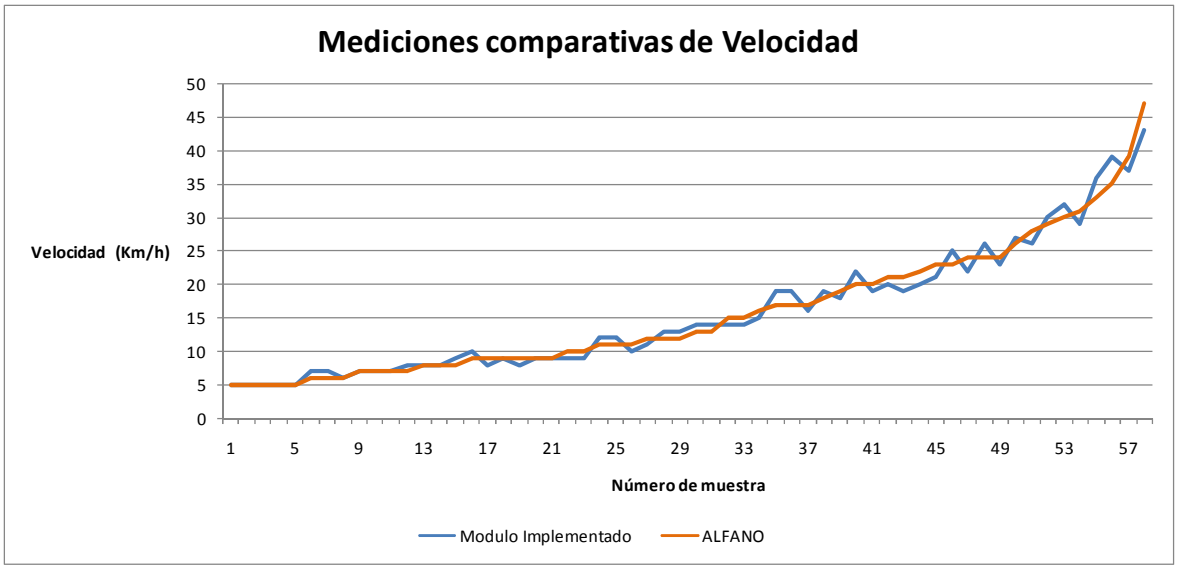


**Figura 4.6** Mediciones comparativas de Velocidad

De los datos que se encuentran en la tabla 4.4 se obtiene que el error promedio es del 6,56%. El valor obtenido de error se encuentra dentro del margen esperado ya que los velocímetros que se encuentran ampliamente difundidos en el mercado poseen un error del 10%.

## 4.5. PRUEBA DE FUNCIONAMIENTO DEL HMI REALIZADO CON LABVIEW 2009

Para la realización de la prueba, la PC cuya ubicación será en el área de pits, se conecta al módulo diseñado para transmisión y recepción. El funcionamiento del programa requiere tener pre instalado Labview 2009 en la PC.

Para empezar a correr la aplicación el usuario deberá seleccionar el ícono que se puede observar en la figura 4.7.

**Figura 4.7** Ícono de inicio de la aplicación

Después de seleccionar el ícono de la figura 4.7 se iniciará el programa mostrando al usuario la pantalla con el menú de Selección.

La figura 4.8 muestra la pantalla del menú de “Selección”.

**Figura 4.8** Pantalla de inicio

A continuación el usuario deberá seleccionar la actividad a realizarse, la cual para la prueba es generar una nueva pista. La figura 4.9 muestra la pantalla generada por el VI “Nueva Pista”.

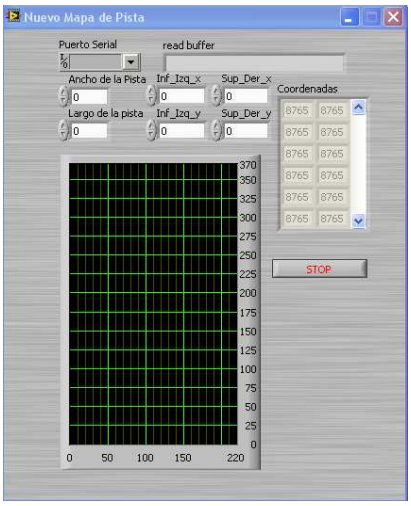


**Figura 4.9** Pantalla del VI “Nueva Pista”

A continuación el usuario deberá ingresar los valores del punto superior derecho, el punto inferior izquierdo, largo y ancho aproximados de la pista.

Después de ingresados los datos solicitados se inicia la adquisición de datos transmitidos por el módulo instalado en el vehículo. Como resultado a la adquisición de los puntos enviados por el módulo se obtiene la gráfica de la pista en el VI “Nueva Carrera”. La figura 4.10 muestra el VI “Nueva Pista” después de la adquisición de datos.

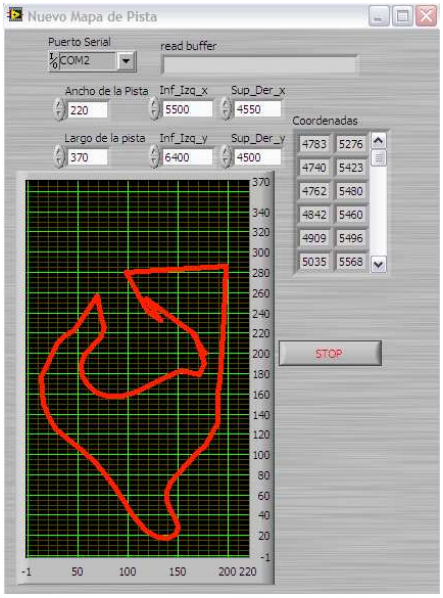


**Figura 4.10** Pista Obtenida

Después de haber generado la pista, el usuario en el menú de selección de la figura 4.8 escoge “Nueva Carrera”. Después de realizada la selección se abre una pantalla que se

puede observar en la figura 4.10 y se procede a cargar la pista para empezar guardar los datos generados por los módulos instalados en los vehículos durante la competencia.

Este VI permite al usuario ver los datos de velocidad, temperatura, R.P.M, cantidad de combustible y posición en la pista de los dos vehículos. Para observar el estado de cada vehículo simplemente se selecciona las pestañas “Auto 1” y “Auto 2”.

La figura 4.11 permite observar la pestaña “recorrido”, en donde se van almacenando los datos generados en la carrera y se van generando gráficos del consumo de combustible, variación de temperatura, R.P.M y velocidad en función del tiempo.

**Figura 4.11** VI “Carrera Nueva”

**Figura 4.12** Pestaña de Recorrido

La última opción del menú selección es “Carrera guardada”. Al escoger esta opción se generarán las mismas pantallas de las figuras 4.11 y 4.12 en donde se podrá cargar una carrera que se haya guardado para su correspondiente análisis.

## 4.6. PRUEBA DE FUNCIONAMIENTO DEL MEDIDOR DE TEMPERATURA

Como se pudo observar en el capítulo de diseño, el sensor de temperatura digital DS18B20 está ubicado con una pinza cercano a la bujía que es el punto más caliente del motor. Se realizó una prueba para comparar los valores de temperatura del motor, de igual manera que en las R.P.M con el módulo ALFANO.

La tabla 4.5 permite observar los valores comparativos obtenidos y el error.

| Número de muestra | Modulo Implementado | ALFANO | Número de muestra | Modulo Implementado | ALFANO |
|---|---|---|---|---|---|
| 1 | 18.5 | 20 | 23 | 48 | 49 |
| 2 | 18.5 | 20 | 24 | 48 | 49 |
| 3 | 18.5 | 20 | 25 | 48 | 51 |
| 4 | 19.5 | 20 | 26 | 48.5 | 50 |
| 5 | 24.5 | 26 | 27 | 48.5 | 49 |
| 6 | 26 | 27 | 28 | 49 | 47 |
| 7 | 27.5 | 28 | 29 | 53 | 54 |
| 8 | 27.5 | 28 | 30 | 56 | 57 |
| 9 | 27.5 | 28 | 31 | 58 | 59 |
| 10 | 28.5 | 30 | 32 | 59 | 60 |
| 11 | 29.5 | 31 | 33 | 59 | 60 |
| 12 | 30.5 | 32 | 34 | 59 | 58 |
| 13 | 31 | 33 | 35 | 62.5 | 64 |
| 14 | 36 | 37 | 36 | 62.5 | 63 |
| 15 | 38.5 | 39 | 37 | 62.5 | 62 |
| 16 | 43 | 42 | 38 | 62 | 63 |
| 17 | 44 | 45 | 39 | 64 | 65 |
| 18 | 44 | 45 | 40 | 66 | 65 |
| 19 | 45.5 | 46 | 41 | 66 | 67 |
| 20 | 46.5 | 48 | 42 | 67 | 68 |
| 21 | 47 | 48 | 43 | 67.5 | 68 |
| 22 | 47 | 48 | 44 | 68 | 69 |

**Tabla 4.5** Mediciones de temperatura

La figura 4.13 permite observar las mediciones comparativas de temperatura obtenidos en la prueba realizada.

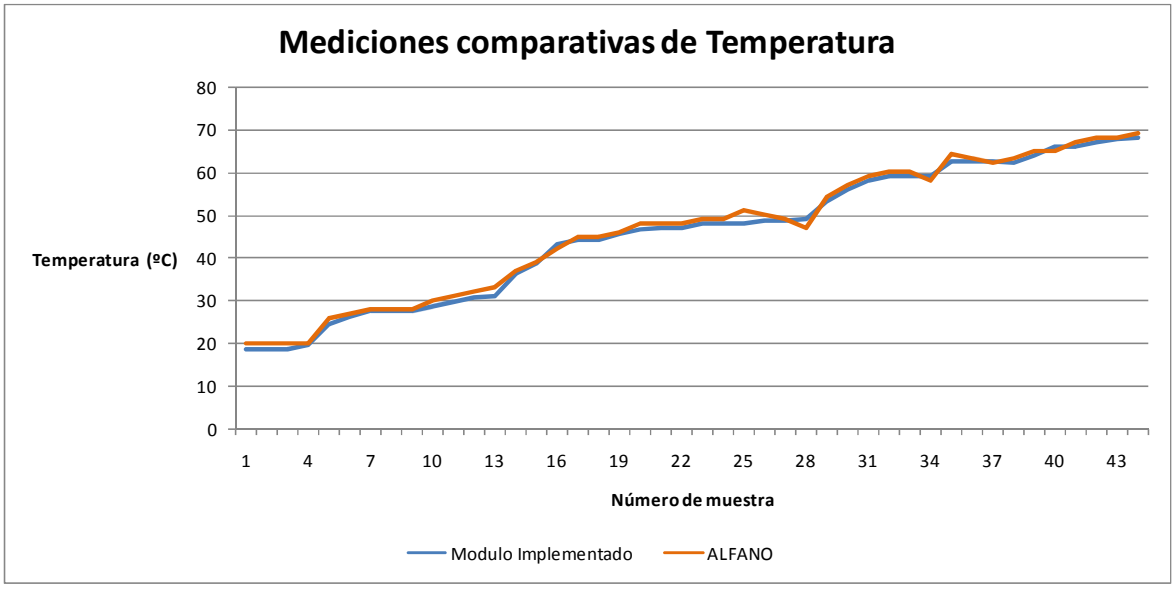


**Figura 4.13** Mediciones comparativas de temperatura

Al analizar los valores del error obtenido en las mediciones de temperatura se puede observar que existe un error promedio del 2,47%, el cual es un valor que se encuentra dentro de un margen aceptable ya que en caso de que el motor alcanzara una temperatura crítica para su funcionamiento, el dispositivo diseñado permite alertar tempranamente a los mecánicos.

# CAPITULO 5

# 5. CONCLUSIONES Y RECOMENDACIONES

## 5.1. CONCLUSIONES

Como conclusiones del proyecto realizado se pueden observar las siguientes:

- La utilización de sensores con salida digital, como el DS18B20 que se utilizó para medir temperatura, facilitó la realización del diseño ya que no fue necesario utilizar circuitos de acondicionamiento.

- El sensor de temperatura puede ser instalado en cualquier tipo de vehículo de karting ya que durante el período de pruebas se observó que la temperatura, que depende del cilindraje del motor, no sobrepasa los 100℃ que es una temperatura menor a la temperatura límite del módulo diseñado, razón por la cual el módulo implementado puede ser instalado en cualquier tipo de vehículo de karting.

- El módulo instalado en el vehículo no necesita tener línea de vista con el módulo ubicado en pits para que se realice la transmisión debido a que la frecuencia que se utiliza “rodea” los obstáculos, pero se pudo observar que la velocidad de transmisión en la que se generaban menos errores fue a 9600bps.

- Para mejorar la calidad en la transmisión de los datos se utilizó un esquema de CRC5 y de mayoría gana. El uso de estos métodos de detección de errores en conjunto con las características de los transceptores YS-1020L permiten obtener la información con mínimos errores.

- El módulo implementado puede ser instalado en cualquier tipo de vehículo de karting, ya que utiliza sensores no invasivos de modo que no hay la posibilidad de causar ningún tipo de afectaciones de funcionamiento sobre el vehículo en el cual se encuentra instalado.

## 5.2. RECOMENDACIONES

- Se debe tener especial cuidado con los valores de voltaje y corriente que manejan los sensores para evitar que ingrese ruido al sistema, lo que incluye tener una fuente independiente del sistema eléctrico del vehículo con circuitos de filtro calibrados para la frecuencia de funcionamiento del motor.

- Para realizar mediciones de frecuencia es recomendable la utilización de un conversor de frecuencia a voltaje como el LM2917 ya que si se utiliza únicamente el microcontrolador para realizar las mediciones de frecuencia el tiempo requerido en el procesamiento de otras tareas puede causar inestabilidad en las medidas realizadas.

- Se recomienda utilizar sensores y elementos que se puedan adquirir en el mercado local ya que al importarlos como el caso de los transceptores utilizados aumentan costos y en caso de daño el tiempo de reemplazo sería relativamente alto.

- Es recomendable la utilización de sensores específicos para la aplicación deseada. En la medición de nivel de combustible se utilizó un sensor de nivel de una lavadora para reducir costos en la implementación lo que causó inconvenientes para lograr que funcione de acuerdo a nuestras necesidades.

- Los módulos diseñados e implementados en el presente proyecto pueden ser adaptados para otro tipo de competencias en donde la distancia entre el módulo móvil y el fijo no sea mayor a 600 metros.

- Es recomendable ubicar los sensores de temperatura entre la bujía y el motor ya que es ese el punto más caliente del motor.

# REFERENCIAS BIBLIOGRÁFICAS

[1] http://secciondeportiva.com/2008/03/04/la-telemetria-en-la-formula-1/

[2] http://www.k-wz.de/vmotor/z_omotors.html

[3] http://www.automotriz.net/tecnica/conocimientos-basicos-11.htm

[4] http://www.kmt-gmbh.com/

[5] http://www.asifunciona.com/electronica/af_gps/af_gps_7.htm

[6] http://www.e-global.es/b2b-blog/wp-images/graficos/GPS.jpg [1]

[7] http://personal.redestb.es/jatienza/gps/datos.htm

[8] http://www.jcea.es/artic/gps-definiciones.htm

[9] http://books.google.com.ec/books?id=iTW7BBKScHsC&pg=RA1-PA748&lpg=RA1PA748&dq=tramas+satelite+GPS&source=bl&ots=k2zAWQjIxJ&sig=OEw8aAP3o3sy6UBDydFBqJOsnOQ&hl=es&ei=CEkIStv6NMertgewoOSGBw&sa=X&oi=book_result&ct=result&resnum=3#PRA2PA149,M1

[10] http://www.pdaexpertos.com/foros/viewtopic.php?t=50102

[11] http://en.wikipedia.org/wiki/Global_Positioning_System

[12] http://www.isa.cie.uva.es/gps/GPSerrores.html

[13] http://www.explicame.org/index2.php?option=com_content&do_pdf=1&id=19

[14] http://es.wikipedia.org/wiki/Microcontrolador

[15] http://www.datasheetcatalog.net/es/datasheets_pdf/A/T/M/E/ATMEGA16-16AC.shtml

[16] http://bieec.epn.edu.ec:8180/dspace/bitstream/123456789/949/6/T10712CAP1.pdf

[17] http://www.monografias.com/trabajos12/microco/microco.shtml

[18] http://es.wikipedia.org/wiki/LabVIEW

[19] http://en.wikipedia.org/wiki/LabVIEW

[20] http://www.yishi.net.cn/product_detail.asp?id=47&fid=139

[21] http://www.textoscientificos.com/redes/modulacion/ask

[22] http://www.textoscientificos.com/redes/modulacion/psk

[23] http://www.textoscientificos.com/redes/modulacion/FSK

# ANEXO A

# MICROCONTROLADOR ATMEGA164P

# ANEXO B

# TRANSCEPTOR YS1020L

# ANEXO C

# RECEPTOR GPS DELUO

# ANEXO D

# CONVERSOR LM2917

# ANEXO E

# CODIGO DEL PROGRAMA DEL MICROCONTROLADOR

```
        .include        "m164pdef.inc"

.equ    lcd_ddr         =       ddrb
.equ    lcd_port        =       portb                   ;LCD Data port
.equ    lcd_in          =       pinb

.equ    lcd_rs          =       0                       ;LCD Register Select
.equ    lcd_en          =       1                       ;LCD Enable
.equ    lcd_d4          =       2
.equ    lcd_d5          =       3
.equ    lcd_d6          =       4
.equ    lcd_d7          =       5


.def    N_l             =       r0                      ;usado como byte menos significativo del dividendo
                                                        ;para la subrutina de division
.def    N_m             =       r1                      ;usado como segundo byte menos significativo del
                                                        ;dividendo para la subrutina de division
.def    N_h             =       r2                      ;usado como byte mas significativo del dividendo
                                                        ;para la subrutina de division
.def    D_l             =       r3                      ;usado como byte menos significativo del divisor
                                                        ;para la subrutina de division
.def    D_m             =       r4                      ;usado como segundo byte menos significativo del
                                                        ;divisor para la subrutina de division
.def    D_h             =       r5                      ;usado como byte mas significativo del divisor
                                                        ;para la subrutina de division
.def    Q_l             =       r6                      ;usado como byte menos significativo del cociente
                                                        ;para la subrutina de division
.def    Q_m             =       r7                      ;usado como segundo byte menos significativo del
                                                        ;cociente para la subrutina de division
.def    Q_h             =       r8                      ;usado como byte mas significativo del cociente
                                                        ;para la subrutina de division
.def    R_l             =       r9                      ;usado como byte menos significativo del residuo
                                                        ;para la subrutina de division
.def    R_m             =       r10                     ;usado como segundo byte menos significativo del
                                                        ;residuo para la subrutina de division
.def    R_h             =       r11                     ;usado como byte mas significativo del residuo
                                                        ;para la subrutina de division
.def    X_l             =       r12                     ;usado como byte menos significativo del registro
                                                        ;auxiliar para la subrutina de division
.def    X_m             =       r13                     ;usado como segundo byte menos significativo del
                                                        ;registro auxiliar para la subrutina de division
.def    X_h             =       r14                     ;usado como byte mas significativo del registro
                                                        ;auxiliar para la subrutina de division
.def    i               =       r15                     ;usado como contador de iteraciones para la subrutina
                                                        ;de division
.def    temp0           =       r16                     ;usado como auxiliar para todo el programa en general
.def    temp1           =       r17                     ;usado como auxiliar para todo el programa en general
.def    temp2           =       r18                     ;usado como auxiliar para todo el programa en general
.def    temp3           =       r19                     ;usado como auxiliar para todo el programa en general
.def    temp4           =       r20                     ;usado como auxiliar para todo el programa en general
.def    temp5           =       r21                     ;usado como auxiliar para todo el programa en general
.def    temp6           =       r22                     ;usado como auxiliar para todo el programa en general
.def    timer1_desbordamiento   =       r23             ;usado para contar cuantas veces se desborda el timer1
.def    estado_temp     =       r24                     ;usado para ver el estado del proceso de obtencion de
                                                        ;la temperatura



.equ    Skip_Rom_cmd        =       0xcc                ;comando para acceder directamente a la pizarra del ds18b20
;.equ   Alarm_Search_cmd    =       0xec
.equ    Convert_T_cmd       =       0x44                ;comando para iniciar la adquisicion del valor de la
                                                        ;temperatura por el ds18b20
.equ    Write_Scratchpad_cmd=       0x4e                ;comando escribir alarmas y configuracion en el ds18b20
.equ    Read_Scratchpad_cmd =       0xbe                ;comando para iniciar la lectura de la pizarra del ds18b20
.equ    Th_18b20_cte        =       0x00                ;valor para la alarma de alta temperatura en el ds18b20
.equ    Tl_18b20_cte        =       0x00                ;valor para la alarma de baja temperatura en el ds18b20
.equ    config_18b20_cte    =       0x1f                ;palabra de configuracion del ds18b20

.equ    mSeg40              =       40000               ;constante para un retardo de 40 milisegundos
.equ    mSeg30              =       30000               ;constante para un retardo de 30 milisegundos
.equ    mSeg10              =       10000               ;constante para un retardo de 10 milisegundos
.equ    uSeg480             =       117                 ;constante para un retardo de 480 microsegundos
.equ    uSeg50              =       9                   ;constante para un retardo de 50 microsegundos
.equ    uSeg20              =       2                   ;constante para un retardo de 20 microsegundos
.equ    uSeg10              =       1                   ;constante para un retardo de 10 microsegundos

.dseg
lcd_status:         .byte       1                       ;display on/off, cursor on/off, blinks on/off==> 0b00001dcb
lcd_salida:         .byte       1                       ;valor que va al puerto
temperatura:        .byte       2                       ;aqui se almacena el codigo de 16bits que devuelve del ds18b20
decimal:            .byte       1                       ;valor decimal para que se imprima en la pantalla lcd
unidades:           .byte       1                       ;valor de las unidades que se imprime en la pantalla lcd
decenas:            .byte       1                       ;valor de las decenas que se imprime en la pantalla lcd
centenas:           .byte       1                       ;valor de las centenas que se imprime en la pantalla lcd
miles:              .byte       1                       ;valor de las unidades de mil que se imprime en la pantalla lcd
decenas_mil:        .byte       1                       ;valor de las decenas de mil que se imprime en la pantalla lcd
period_l:           .byte       1                       ;byte menos significativo de tres bytes que representan
                                                        ;el periodo de las rpm
period_m:           .byte       1                       ;segundo byte menos significativo de tres bytes que
                                                        ;representan el periodo de las rpm
period_h:           .byte       1                       ;byte mas significativo de tres bytes que representan
                                                        ;el periodo de las rpm
adc_val:            .byte       2                       ;reservo 64 bytes para almacenar los valores del adc
adc_suma:           .byte       2                       ;valor donde se almacena la suma de los valores
                                                        ;previamente obtenidos del adc
adc_promedio:       .byte       2                       ;reservo un byte para el valor promedio de las
                                                        ;256 muestras del adc
adc_i:              .byte       1                       ;contador que permite verificar cuantas muestras
                                                        ;del adc se han sumado
adc_nivel:          .byte       1
adc_rpm:            .byte       1


rx0_buffer:         .byte       1                       ;bytes recibidos por el puerto 0
tx0_buffer:         .byte       64                      ;bytes a ser transmitidos por el puerto 0

rx1_buffer:         .byte       70                      ;bytes recibidos por el puerto 0
rx1_i:              .byte       2

tx1_buffer:         .byte       1                       ;bytes a ser transmitidos por el puerto 0
```

```
;*****************************************************************************
;*********************************Inicio del programa*************************
;*****************************************************************************

.cseg
                .org        0x0000                      ;primera direccion, donde el micro comienza a leer
                jmp         reset


                ;.org        0x0018                     ;direccion para la interrupcion de captura del timer1
                ;jmp             timer1_capt


                .org        0x001a                      ;direccion para la interrupcion de captura del timer1
                jmp         timer1_compa


                .org        0x001e                      ;direccion para la interrupcion de desboramiento
                                                        ;del timer1
                jmp         timer1_ovf


                .org        0x0020                      ;direccion para la interrupcion de comparacion del
                                                        ;timer0
                jmp         timer0_compa


                .org        0x0028                      ;direccion para la interrupcion por recepcion por
                                                        ;el puerto serial0
                jmp         usart0_rx                   ;vaya a la subrituna de recepcion del puerto 0


                .org        0x002c                      ;direccion para la interrupcion por transmision de
                                                        ;byte completa por el puerto serial0
                jmp         usart0_tx                   ;vaya a la subrutina de transmision del puerto 0


                .org        0x0030                      ;direccion para la interrupcion de final del conversion del adc
                jmp         adc_do


                .org        0x0038                      ;direccion para la interrupcion por recepcion por el
                                                        ;puerto serial0
                jmp         usart1_rx                   ;vaya a la subrituna de recepcion del puerto 1
                ;Configuracion del funcionamiento del uC
                .org        0x003e
reset:          ldi         temp0,high(ramend)          ;inicializo el puntero del stack
                out         sph,temp0
                ldi         temp0,low(ramend)
                out         spl,temp0


                ;Configuracion de los puertos de I/O
                ser         temp0
                out         ddrb,temp0                  ;configuro el puerto b como salida
                clr         temp0
                out         ddra,temp0                  ;configuro el puerto a como entrada
                ldi         temp0,0b11110000
                out         ddrc,temp0                  ;configuro el puerto c la mitad como entrada y la
                                                        ;otra mitad como salida
                ldi         temp0,0b10111010
                out         ddrd,temp0                  ;configuro el puerto d algunas lineas como entradas
                                                        ;y otras como salidas
                ldi         temp0,255                   ;_____________________________
                out         lcd_port,temp0              ;___configuro el puerto donde esta conectado el lcd como salida

setup_lcd:      call        lcd_init                    ;subrutina par ainicializar la pantalla lcd
                call        lcd_cursor_off              ;apago el cursor del lcd
                ldi         temp0,0x80
                call        lcd_command                 ;situo el cursor sobre el primer caracter de la primera linea
                ldi         temp0,'T'
                call        lcd_senddata                ;escribo la letra T
                ldi         temp0,':'
                call        lcd_senddata                ;escribo dos puntos
                ldi         temp0,' '
                call        lcd_senddata                ;escribo un espacio

                ldi         temp0,0x8a                  ;situo el cursor sobre el decimo primer caracter de la primera linea
                call        lcd_command
                ldi         temp0,'N'
                call        lcd_senddata                ;escribo la letra N
                ldi         temp0,':'
                call        lcd_senddata                ;escribo dos puntos
                ldi         temp0,' '
                call        lcd_senddata                ;escribo un espacio

                ldi         temp0,0xc0
                call        lcd_command                 ;ubico el cursor sobre el primer caracter de la segunda linea
                ldi         temp0,'R'
                call        lcd_senddata                ;escribo la letra R
                ldi         temp0,'P'
                call        lcd_senddata                ;escribo la letra P
                ldi         temp0,'M'
                call        lcd_senddata                ;escribo la letra M
                ldi         temp0,':'
                call        lcd_senddata                ;escribo dos puntos
                ldi         temp0,' '
                call        lcd_senddata                ;escribo un espacio


setup_18b20:    call        reset_1W                    ;envio nua señal de reset al sensor ds18b20
                call        wait_pres                   ;espero el pulso de respuesta del sensor ds18b20
                call        Write_Scratchpad            ;envio un comando para escribir en la pizarra
                call        Th_18b20                    ;escribo el valor de la alarma para la temperatura superior
                call        Tl_18b20                    ;escribo el valor de la alarma para la temperatura inferior
                call        config_18b20                ;escribo la palabra de configuracion en el sensor


                ;Configuracion del USART0               ;este USART se conecta con el tranceiver de RF
setup_usart0:   clr         temp0
                sts         ubrr0h,temp0                ;;configuro la velocidad de la
                                                        ;Vtx = 4800bps--> UBRR = 12 para 1MHz en modo de baja velocidad
                ldi         temp0,12
                sts         ubrr0l,temp0
                clr         temp0
                ldi         temp0,0b00000010
                sts         ucsr0a,temp0                ;No habilito doble velocidad ni comunicacion multiprocesor
                ldi         temp0,0b11011000
                sts         ucsr0b,temp0                ;Habilito interrupciones de Rx y Tx, habilito la Rx y Tx,
                                                        ;configuro comunicacion para 8 bits de datos
                ldi         temp0,0b00000110
                sts         ucsr0c,temp0                ;Selecciono comunicacion asincronica, sin bits de paridad,
                                                        ;1 bit de parada, 8 bits de datos, polarizacion por flanco
                                                        ;negativo para el reloj
```

```
                ;Configuracion del USART1                   ;este USART se conecta con el GPS
setup_usart1:   clr         temp0
                sts         ubrr1h,temp0                    ;configuro la velocidad de la
                                                            ;Vtx = 4800bps--> UBRR = 12 para 1MHz en modo de baja velocidad
                ldi         temp0,12
                sts         ubrr1l,temp0
                clr         temp0
                sts         ucsr1a,temp0                    ;No habilito doble velocidad ni comunicacion
                                                            ;multiprocesor
                ldi         temp0,0b01001000
                sts         ucsr1b,temp0                    ;Habilito interrupciones de Rx y Tx,
                                                            ;habilito la Rx y Tx, configuro comunicacion
                                                            ;para 8 bits de datos
                ldi         temp0,0b00000110
                sts         ucsr1c,temp0                    ;Selecciono comunicacion asincronica,
                                                            ;sin bits de paridad, 1 bit de parada,
                                                            ;8 bits de datos, polarizacion por flanco
                                                            ;negativo para el reloj

                ;Inicializacion del GPS
GPS_init:       ldi         zh,high(CLR_*2)
                ldi         zl,low(CLR_*2)
                call        usart1_tx

                call        wait_40m

                ldi         zh,high(CLR_*2)
                ldi         zl,low(CLR_*2)
                call        usart1_tx

                call        wait_40m

                ldi         zh,high(OI10*2)
                ldi         zl,low(OI10*2)
                call        usart1_tx

                call        wait_40m

                ldi         zh,high(OI10*2)
                ldi         zl,low(OI10*2)
                call        usart1_tx

                call        wait_40m

                ldi         zh,high(NC*2)

                ldi         zl,low(NC*2)
                call        usart1_tx

                call        wait_40m

                ldi         zh,high(OI1*2)
                ldi         zl,low(OI1*2)
                call        usart1_tx

                call        wait_40m

                ldi         zh,high(OI1*2)
                ldi         zl,low(OI1*2)
                call        usart1_tx

                ;Configuracion del USART1                   ;este USART se conecta con el tranceiver de RF
                ldi         temp0,0b10010000
                sts         ucsr1b,temp0                    ;Habilito interrupciones de Rx, habilito la Rx,
                                                            ;configuro comunicacion para 8 bits de datos
                ldi         temp0,0b00000110
                sts         ucsr1c,temp0                    ;Selecciono comunicacion asincronica, sin bits
                                                            ;de paridad, 1 bit de parada, 8 bits de datos,
                                                            ;polarizacion por flanco negativo para el reloj


                ;Configuracion del ADC
setup_adc:      ldi         temp0,0b00100000            ;configuro el ADC para referencia externa, ajuste a
                                                        ;la izquierda, y el canal0
                sts         admux,temp0
                ldi         temp0,0b11001111            ;habilito el ADC, habilito la interrupcion del ADC,
                                                        ;uso un preescaler de 128
                sts         adcsra,temp0
                clr         temp0                       ;uso free-running mode en caso de habilitar el bit
                                                        ;ADATE en el adcsra
                sts         adcsrb,temp0
                ldi         temp0,0x03                  ;apago el modo digital de todos los pines del puerto A
                sts         didr0,temp0


                ;Configuracion del TIMER0
setup_timer0:   ldi         temp0,0b00000010                ;
                out         tccr0a,temp0                    ;configuro el timero para modo ctc
                ldi         temp0,0b00000101
                out         tccr0b,temp0                    ;selecciono el prescaler de 1024
                ldi         temp0,244
                out         ocr0a,temp0                     ;el valor a ser comparado es 244-->con prescaler
                                                            ;de 1024 produce interrupcion cada 250mseg aprox.
                ldi         temp0,0b00000010
                sts         timsk0,temp0                    ;habilito interrupcion por comparacion para el
                                                            ;timer0


                ;Configuracion del TIMER1
setup_timer1:   ldi         temp0,0b00000000                ;configuracion por default del registro tccr1a
                sts         tccr1a,temp0
                ldi         temp0,0b00001101                ;0b11000001;seleciono interrupcion con flanco
                                                            ;positivo y reloj sin prescaler
                sts         tccr1b,temp0
                ldi         temp0,0b00000010                ;habilito las interrupciones por captura y por
                                                            ;desbordamiento
                sts         timsk1,temp0
                ldi         temp0,high(100)
                sts         ocr1ah,temp0
                ldi         temp0,low(100)
                sts         ocr1al,temp0

set_values:         ldi         temp0,high(rx1_buffer)
                    sts         rx1_i,temp0
                    ldi         temp0,low(rx1_buffer)
                    sts         rx1_i+1,temp0

int_on:             sei                                     ;enciendo las interrupciones

;*************************************************************************************
;*********************************Programa Principal**********************************
;*************************************************************************************

main:
                ;jmp             main

                mov         temp0,estado_temp       ;estado_temp contiene el estado si debo iniciar la conversion,
                                                    ;si debo leer o si debo imprimir los datos en el lcd
                lsr         temp0
                brcc        main
                cli
                cpi         estado_temp,1           ;si estado_temp es 1 voy a iniciar la conversion
                breq        start_temper
                cpi         estado_temp,3           ;si estado_temp es 3 voy a leer el resultado
                breq        read_temper
                jmp         int_on                  ;si estado temp es cualquier valor enciendo las interrupciones
```

```
start_temper:       inc         estado_temp
                    call        reset_1W                ;doy un pulso de reset
                    call        wait_pres               ;espero el pulso de presencia
                    call        Skip_Rom                ;envio un comando para acceder directamente al ds18b20
                    call        Convert_T               ;inicio la conversion
                    jmp         int_on                  ;voy a encender las interrupciones y luego al main

read_temper:        clr         estado_temp             ;borro el estado_temp
                    call        signal_check            ;reviso si el sensor acabo la conversion
                    call        Reset_1W                ;doy un pulso de reset
                    call        wait_pres               ;espero el pulso de presencia
                    call        Skip_Rom                ;muevo 0xcc a W
                    call        Read_Scratchpad         ;envio comando para iniciar la lectura de la temperatura dle ds18b20
                    call        read_1W                 ;subrutina para leer un byte
                    sts         temperatura,temp0       ;almaceno el byte menos significativo en la sram
                    call        read_1W                 ;subrutina para leer un byte
                    sts         temperatura+1,temp0     ;almaceno el byte mas significativo en la sram
                    call        SigVal                  ;acomoda los valores para obtener signo en R_Signo,
                                                        ;temperatura entera en R_Temp y fracciones en R_Frac

                    ;/
                    ;lds            temp0,period_l          ;obtengo el valor
                    ;mov            D_l,temp0               ;del periodo
                    ;lds            temp0,period_m          ;y lo voy
                    ;mov            D_m,temp0               ;almacenando
                    ;lds            temp0,period_h          ;en el
                    ;mov            D_h,temp0               ;divisor
                    ;ldi            temp0,0x0f              ;guardo
                    ;mov            N_h,temp0               ;el valor de
                    ;ldi            temp0,0x42              ;un millon
                    ;mov            N_m,temp0               ;en el
                    ;ldi            temp0,0x40              ;dividendo
                    ;mov            N_l,temp0
                    ;call       division                ;llamo a la subrutina de division
                    ;/

                    call        lcd_datos               ;imprimo los valores en el lcd
                    jmp         int_on
;-------------------------------------------------------------------------------------

timer1_capt:        sbi         portc,6                         ;pongo 1L en portc,6
                    push        temp0                           ;guardo el contenido de temp0
                    push        temp1                           ;guardo el contenido de temp1
                    lds         temp0,icr1l                     ;recupero el valor
                    lds         temp1,icr1h                     ;del periodo
                    sts         period_l,temp0                  ;de la señal
                    sts         period_m,temp1                  ;y lo almaceno
                    sts         period_h,timer1_desbordamiento      ;en la sram
                    clr         timer1_desbordamiento               ;borro los
                    clr         temp0                           ;contadores del periodo
                    sts         tcnt1h,temp0                    ;para iniciar
                    ldi         temp0,0x1e      ;0x1f           ;una nueva medida
                    sts         tcnt1l,temp0                    ;de periodo en el siguiente ciclo
timer1_capt_exit:   pop         temp1                           ;recupero el valor previo de temp1
                    pop         temp0                           ;recupero el valor previo de temp0
                    cbi         portc,6                         ;pongo 0L en portc,6
                    reti
;-------------------------------------------------------------------------------------

timer1_ovf:         inc         timer1_desbordamiento           ;incremento el byte mas significativo del periodo
                    reti
;-------------------------------------------------------------------------------------

timer1_compA:       push        temp0
                    ldi         temp0,high(rx1_buffer)
                    sts         rx1_i,temp0
                    ldi         temp0,low(rx1_buffer)
                    sts         rx1_i+1,temp0
                    pop         temp0
                    reti
;-------------------------------------------------------------------------------------

timer0_compa:       sbi         portc,7                         ;pongo 1L en portc,7
                    inc         estado_temp                     ;cambio el estado_temp para continuar con
                                                                ;el rpoceso de medida de temperatura
                    cbi         portc,7                         ;pongo 0L en portc,7
                    reti
;-------------------------------------------------------------------------------------

usart0_rx:
                    reti
;-------------------------------------------------------------------------------------

usart0_tx:          push        temp0                           ;almaceno el valor de temp0
                    ld          temp0,x+                        ;recupero el contenido del buffer de transmision del usart0
                    cpi         temp0,0x00                      ;verifico si es cero
                    breq        usart0_tx_exit                  ;si es cero salgo
                    sts         udr0,temp0                      ;si es diferente lo envio y salgo
usart0_tx_exit:     pop         temp0                           ;recupero el valor previo de temp0
                    reti                                        ;retorno de la interrupcion
;-------------------------------------------------------------------------------------

usart1_rx:          push        xh
                    push        xl
                    push        temp0
                    lds         xh,rx1_i
                    lds         xl,rx1_i+1
                    lds         temp0,udr1
                    st          x+,temp0
                    sts         rx1_i,xh
                    sts         rx1_i+1,xl
                    clr         temp0
                    sts         tcnt1h,temp0
                    sts         tcnt1l,temp0
                    pop         temp0
                    pop         xl
                    pop         xh
                    reti
;-------------------------------------------------------------------------------------

usart1_tx:          lpm         temp1,z+                        ;recupero el contenido del buffer de transmision del usart0
                    cpi         temp1,0x00                      ;verifico si es cero
                    breq        usart1_tx_exit                  ;si es cero salgo, es decir si se acabo el
                                                                ;string a ser transmitido
                    sts         udr1,temp1                      ;si es diferente lo envio y salgo
usart1_tx_loop:     lds         temp1,ucsr1a
                    sbrs        temp1,6
                    jmp         usart1_tx_loop
                    call        wait_10m
                    jmp         usart1_tx
usart1_tx_exit:     lds         temp1,ucsr1a
                    sts         ucsr1a,temp1
                    ret                                         ;retorno de la interrupcion
;-------------------------------------------------------------------------------------

adc_do:             push        temp0                           ;almaceno el contenido del registro temp0
                    push        temp1                           ;almaceno el contenido del registro temp1
                    push        temp2                           ;almaceno el contenido del registro temp2
                    push        temp3                           ;alamceno el contenido del registro temp3
                    lds         temp0,adch                      ;muevo el valor del adc hacia temp0
                    sts         adc_val,temp0                   ;guardo el valor del adc en la sram
                    clr         temp1                           ;borro temp1
                    lds         temp2,adc_suma                  ;sumo el valor actual
```

```
                    lds         temp3,adc_suma+1            ;con los vlaores previos
                    add         temp2,temp0                 ;almacenados en adc_suma
                    adc         temp3,temp1                 ;y en adc_suma+1
                    sts         adc_suma,temp2              ;alamceno el valor presente
                    sts         adc_suma+1,temp3            ;en adc_suma y adc_suma+1
                    lds         temp0,adc_i                 ;verifico el contador
                    inc         temp0                       ;de sumas realizadas
                    sts         adc_i,temp0                 ;si no se han realizado 256 sumas
                    brne        adc_do_exit                 ;salgo, caso contrario
                    sts         adc_promedio,temp3          ;encuentro el promedio
                    lds         temp0,admux
                    cpi         temp0,0b00100000            ;configuro el ADC para referencia externa,
                                                            ;ajuste a la izquierda, y el canal0
                    breq        store_nivel
store_rpm:          sts         adc_rpm,temp3
                    ldi         temp0,0b00100000
                    sts         admux,temp0
                    jmp         adc_set
store_nivel:        sts         adc_nivel,temp3
                    ldi         temp0,0b00100001
                    sts         admux,temp0
adc_set:            clr         temp0                       ;y borro el temp0 para
                    sts         adc_suma,temp0              ;para inniciar de nuevo
                    sts         adc_suma+1,temp0            ;el proceso de sumas
adc_do_exit:        lds         temp0,adcsra                ;reinicio
                    ori         temp0,0b01000000            ;el proceso
                    sts         adcsra,temp0                ;de conversion
                    pop         temp3                       ;recupero el valor previo de temp3
                    pop         temp2                       ;recupero el valor previo de temp2
                    pop         temp1                       ;recupero el valor previo de temp1
                    pop         temp0                       ;recupero el valor previo de temp0
                    reti
;-------------------------------------------------------------------------------------

reset_1W:           cbi         portc,1                     ;bus va a nivel bajo
                    sbi         ddrc,1                      ;maneja el trisA
                    call        wait_480u                   ;retardo de 480uSeg
                    cbi         ddrc,1                      ;bus va a nivel alto por pull-up, maneja el trisA
                    ret
;-------------------------------------------------------------------------------------

wait_pres:          sbic        pinc,1                      ;se queda esperando que el bus vaya a 0l
                    jmp         wait_pres
rise_edge:          sbis        pinc,1                      ;se queda esperando que el bus vaya a 1l
                    jmp         rise_edge
                    ret
;-------------------------------------------------------------------------------------

config_18b20:       ldi         temp0,config_18b20_cte      ;subrutina para configurar el ds18b20
                    call        write_1W
Tl_18b20:           ldi         temp0,Tl_18b20_cte          ;subrutina para setear la alarma de baja
                                                            ;temperatura en el ds18b20
                    jmp         write_1W
Th_18b20:           ldi         temp0,Th_18b20_cte          ;subrutina para setear la alarma de alta
                                                            ;temperatura en el ds18b20
                    jmp         write_1W
Write_Scratchpad:   ldi         temp0,Write_Scratchpad_cmd  ;subrutina para escribir un valor en la pizarra
                    jmp         write_1W
Read_Scratchpad:    ldi         temp0,Read_Scratchpad_cmd   ;muevo 0xbe en W, para leer la pizarra
                    jmp         write_1W                    ;envio el dato en forma serial
Convert_T:          ldi         temp0,Convert_T_cmd         ;muevo el valor 0x44 a W
                    jmp         write_1W                    ;envio el dato en forma serial
Skip_Rom:           ldi         temp0,Skip_Rom_cmd          ;muevo 0xcc a W
Write_1W:           ldi         temp1,0x08
start_bit_w:        sbi         ddrc,1                      ;maneja el ddrc
                    call        wait_10u                    ;retardo de 10uSeg
                    ror         temp0
                    brcc        write_slot
                    cbi         ddrc,1                      ;bus va a nivel alto por pull-up, maneja el trisA
write_slot:         call        wait_50u
                    cbi         ddrc,1                      ;bus va a nivel alto por pull-up, maneja el trisA
                    nop                                     ;espera 1 useg
                    dec         temp1
                    brne        start_bit_w
                    call        wait_50u
                    ret
;-------------------------------------------------------------------------------------

read_1W:            ldi         temp1,0x08                  ;uso temp1 para contar 8 ciclos (bits)
                    clr         temp0                       ;borro temp1
start_bit_r:        sbi         ddrc,1                      ;doy pulso de inicio
                    call        wait_10u                    ;retardo de 10uSeg
                    cbi         ddrc,1                      ;bus va a nivel alto por pull-up, maneja el trisA
                    call        wait_10u                    ;retardo de 10uSeg
                    clc                                     ;borro el carry
                    sbic        pinc,1                      ;leo el bit enviado por el ds18b20
                    sec                                     ;pongo en 1l el carry
                    ror         temp0                       ;roto temp0 para el siguiente bit
                    call        wait_20u                    ;retado de 20uSeg
                    call        wait_10u                    ;retardo de 10uSeg
                    dec         temp1                       ;decremento el contador de bits
                    brne        start_bit_r                 ;cominezo el proceso de nuevo
                    ret
;-------------------------------------------------------------------------------------

signal_check:       sbis        pinc,1                      ;0 en proceso, 1 hecha la conversion
                    jmp         signal_check
                    call        wait_10u                    ;retardo de 10uSeg
                    ret
;-------------------------------------------------------------------------------------

SigVal:             lds         temp0,temperatura           ;recupero los
                    lds         temp1,temperatura+1         ;valores de la temperatura
                    ldi         temp2,0x03                  ;uso temp3 como contador
Recorre:            lsr         temp1                       ;roto hacia la derecha los
                    ror         temp0                       ;valores de la temperatura
                    dec         temp2                       ;hasta obtener un valor de 8 bits
                    brne        Recorre                     ;7 bits de entero y un decimal
                    sts         temperatura,temp0           ;almaceno el valor final de temperatura
                    ret                                     ;retorno de la subrutina
;-------------------------------------------------------------------------------------

wait_40m:           ldi         yh,high(mSeg40)             ;subrutina de retardo de 40mSeg
                    ldi         yl,low(mSeg40)              ;copio la constante correspondiente
                    jmp         wait_us                     ;salto al retardo
wait_30m:           ldi         yh,high(mSeg30)             ;subrutina de retardo de 30mSeg
                    ldi         yl,low(mSeg30)              ;copio la constante correspondiente
                    jmp         wait_us                     ;salto al retardo
wait_10m:           ldi         yh,high(mSeg10)             ;subrutina de retardo de 10mSeg
                    ldi         yl,low(mSeg10)              ;copio la constante correspondiente
                    jmp         wait_us                     ;salto al retardo
wait_480u:          ldi         yh,high(uSeg480)            ;subrutina de retardo de 480uSeg
                    ldi         yl,low(uSeg480)             ;copio la constante correspondiente
                    jmp         wait_us                     ;salto al retardo
wait_50u:           ldi         yh,high(uSeg50)             ;subrutina de retardo de 50uSeg
                    ldi         yl,low(uSeg50)              ;copio la constante correspondiente
                    nop                                     ;retardo de ajuste
                    nop                                     ;retardo de ajuste
                    jmp         wait_us                     ;salto al retardo
wait_20u:           ldi         yh,high(uSeg20)             ;subrutina de retardo de 20uSeg
                    ldi         yl,low(uSeg20)              ;copio la constante correspondiente
                    jmp         wait_us                     ;salto al retardo
wait_10u:           ldi         yh,high(uSeg10)             ;subrutina de retardo de 10uSeg
                    ldi         yl,low(uSeg10)              ;copio la constante correspondiente
wait_us:            sbiw        y,1
```

```
                brne    wait_us
                ret
;-------------------------------------------------------------------------------

division:       clr     R_h                     ;borro el valor del residuo_h
                clr     R_m                     ;borro el valor del residuo_m
                clr     R_l                     ;borro el valor del residuo_l
                clr     Q_h                     ;borro el valor del cuociente_h
                clr     Q_m                     ;borro el valor del cuociente_m
                clr     Q_l                     ;borro el valor del cuociente_l
                clr     X_h                     ;borro el valor del auxiliar_h
                clr     X_m                     ;borro el valor del auxiliar_m
                clr     X_l                     ;borro el valor del auxiliar_l
                ldi     temp0,25                ;uso i como contador de iteraciones
                mov     i,temp0
division_loop:  dec     i                       ;decremento i
                breq    division_exit
                lsl     N_l                     ;desplazo hacia
                rol     N_m                     ;la izquierda
                rol     N_h                     ;tanto el dividendo
                rol     R_l                     ;como el residuo
                rol     R_m                     ;para
                rol     R_h                     ;luego
                mov     X_l,R_l                 ;actualizar el
                mov     X_m,R_m                 ;registro X con
                mov     X_h,R_h                 ;los valores de los residuos
                sub     X_l,D_l                 ;le resto los divisores
                sbc     X_m,D_m                 ;a los registros
                sbc     X_h,D_h                 ;X
                brcs    carry_set               ;y verifico si hay carry
carry_reset:    mov     R_h,X_h                 ;si no hay carry
                mov     R_m,X_m                 ;actualizo los residuos
                mov     R_l,X_l                 ;con el nuevo valor de X
                sec                             ;pongo el carry en 1L
                jmp     set_Q                   ;y voy a escribir los cocientes
carry_set:      clc                             ;si habia carry borro el carry
set_Q:          rol     Q_l                     ;roto los cocientes
                rol     Q_m                     ;hacia la izquierda
                rol     Q_h                     ;y comienzo
                jmp     division_loop           ;otro ciclo
division_exit:  ret                             ;hasta retorna de la subrutina
;-------------------------------------------------------------------------------

bin2bcd:        ;Esta Subrutina cambia un numero de 16 bits a BCD sin realizar divisiones (restas)
                ;suscesivas ahorrando tiempo al uControlador
                  push    temp2                 ;almaceno el contenido de temp2 en el stack
                  push    temp3                 ;almaceno el contenido de temp3 en el stack
                  push    temp4                 ;almaceno el contenido de temp4 en el stack
                  push    temp5                 ;almaceno el contenido de temp5 en el stack
                  push    temp6                 ;almaceno el contenido de temp6 en el stack

                  clr     temp2                 ;borro el contenido de temp2
                  clr     temp3                 ;borro el contenido de temp3
                  clr     temp4                 ;borro el contenido de temp4
                  clr     temp5                 ;borro el contenido de temp5

                  lsl     temp0                 ;roto hacia
                  rol     temp1                 ;la izquierda
                  rol     temp2                 ;el numero
                  rol     temp3                 ;almacenado en
                  rol     temp4                 ;temp1:temp0

                  lsl     temp0                 ;roto hacia
                  rol     temp1                 ;la izquierda
                  rol     temp2                 ;el numero
                  rol     temp3                 ;almacenado en
                  rol     temp4                 ;temp1:temp0

  bin2bcd_loop:   push    temp5                 ;guardo el contador de desplazamiento en el stack
                  lsl     temp0                 ;roto hacia
                  rol     temp1                 ;la izquierda
                  rol     temp2                 ;el numero
                  rol     temp3                 ;almacenado en
                  rol     temp4                 ;temp1:temp0

  mil_back:       ldi     temp6,0x30            ;almaceno 0x30 en temp6 por si necesito sumar
                                                ;ese valor luego con las unidades de mil
                  mov     temp5,temp3           ;muevo miles:centenas a temp5
                  andi    temp5,0xf0            ;dejo solo las unidades de mil
                  cpi     temp5,0x50            ;verifico si exceden el valor de 5
                  brlo    cent_back             ;si no es mayor que cinco voy a verificar las centenas
                  add     temp5,temp6           ;si exceden el valor de 5 les sumo 3
                  andi    temp3,0x0f            ;borro el valor anterior de las unidades de mil
                  or      temp3,temp5           ;almaceno el nuevo valor de las unidades de mil

  cent_back:      ldi     temp6,0x03            ;almaceno 0x03 en temp6 por si necesito
                                                ;sumar ese valor luego con las centenas
                  mov     temp5,temp3           ;muevo miles:centenas a temp5
                  andi    temp5,0x0f                ;dejo solo las centenas
                  cpi     temp5,0x05            ;verifico si exceden el valor de 5
                  brlo    dec_back              ;si no es mayor que cinco voy a verificar
                                                ; las decenas
                  add     temp5,temp6           ;si exceden el valor de 5 les sumo 3
                  andi    temp3,0xf0            ;borro el valor anterior de las centenas
                  or      temp3,temp5           ;almaceno el nuevo valor de las centenas

  dec_back:       ldi     temp6,0x30            ;almaceno 0x30 en temp6 por si necesito
                                                ;sumar ese valor luego
                  mov     temp5,temp2           ;muevo decenas:unidades a temp5
                  andi    temp5,0xf0            ;dejo solo las decenas
                  cpi     temp5,0x50            ;verifico si exceden el valor de 5
                  brlo    uni_back              ;si no es mayor que cinco voy a verificar
                                                ; las unidades
                  add     temp5,temp6           ;si exceden el valor de 5 les sumo 3
                  andi    temp2,0x0f            ;borro el valor anterior de las decenas
                  or      temp2,temp5           ;almaceno el nuevo valor de las decenas


  uni_back:       ldi     temp6,0x03            ;almaceno 0x03 en temp6 por si necesito
                                                ; sumar ese valor luego con las unidades
                  mov     temp5,temp2           ;muevo decenas:unidades a temp5
                  andi    temp5,0x0f            ;dejo solo las unidades
                  cpi     temp5,0x05            ;verifico si exceden el valor de 5
                  brlo    bin2bcd_check         ;si no es mayor que cinco voy a verificar
                                                ; si ya he rotado 15 veces el numero
                  add     temp5,temp6           ;si exceden el valor de 5 les sumo 3
                  andi    temp2,0xf0            ;borro el valor anterior de las unidades
                  or      temp2,temp5           ;almaceno el nuevo valor de las unidades


  bin2bcd_check:  pop     temp5                 ;recupero del stack el valor del contador
                                                ; de desplazamientos
                  inc     temp5                 ;incremento el contador
                  cpi     temp5,13              ;verfico si ya he rotado 15 veces = 13
                                                ; del lazo + 2 antes de entrar al lazo
                  brsh    bin2bcd_exit          ;si ya son 15 veces voy al final de la
                                                ; subrutina
                  jmp     bin2bcd_loop          ;si es menor a 15 veces voy al inicio
                                                ;del lazo y realizo el proceso de nuevo
  bin2bcd_exit:
                  lsl     temp0                 ;roto hacia
                  rol     temp1                 ;la izquierda
                  rol     temp2                 ;el numero a ser
                  rol     temp3                 ;convertido a BCD
                  rol     temp4                 ;para completar 16 desplazamientos
```

```
                    brlo        dec_back                    ;si no es mayor que cinco voy a verificar
                                                            ; las decenas
                    add         temp5,temp6                 ;si exceden el valor de 5 les sumo 3
                    andi        temp3,0xf0                  ;borro el valor anterior de las centenas
                    or          temp3,temp5                 ;almaceno el nuevo valor de las centenas

dec_back:           ldi         temp6,0x30                  ;almaceno 0x30 en temp6 por si necesito
                                                            ;sumar ese valor luego
                    mov         temp5,temp2                 ;muevo decenas:unidades a temp5
                    andi        temp5,0xf0                  ;dejo solo las decenas
                    cpi         temp5,0x50                  ;verifico si exceden el valor de 5
                    brlo        uni_back                    ;si no es mayor que cinco voy a verificar
                                                            ; las unidades
                    add         temp5,temp6                 ;si exceden el valor de 5 les sumo 3
                    andi        temp2,0x0f                  ;borro el valor anterior de las decenas
                    or          temp2,temp5                 ;almaceno el nuevo valor de las decenas


uni_back:           ldi         temp6,0x03                  ;almaceno 0x03 en temp6 por si necesito
                                                            ; sumar ese valor luego con las unidades
                    mov         temp5,temp2                 ;muevo decenas:unidades a temp5
                    andi        temp5,0x0f                  ;dejo solo las unidades
                    cpi         temp5,0x05                  ;verifico si exceden el valor de 5
                    brlo        bin2bcd_check               ;si no es mayor que cinco voy a verificar
                                                            ; si ya he rotado 15 veces el numero
                    add         temp5,temp6                 ;si exceden el valor de 5 les sumo 3
                    andi        temp2,0xf0                  ;borro el valor anterior de las unidades
                    or          temp2,temp5                 ;almaceno el nuevo valor de las unidades


bin2bcd_check:      pop         temp5                       ;recupero del stack el valor del contador
                                                            ; de desplazamientos
                    inc         temp5                       ;incremento el contador
                    cpi         temp5,13                    ;verfico si ya he rotado 15 veces = 13
                                                            ; del lazo + 2 antes de entrar al lazo
                    brsh        bin2bcd_exit                ;si ya son 15 veces voy al final de la
                                                            ; subrutina
                    jmp         bin2bcd_loop                ;si es menor a 15 veces voy al inicio
                                                            ;del lazo y realizo el proceso de nuevo
bin2bcd_exit:
                    lsl         temp0                       ;roto hacia
                    rol         temp1                       ;la izquierda
                    rol         temp2                       ;el numero a ser
                    rol         temp3                       ;convertido a BCD
                    rol         temp4                       ;para realizar el desplazamiento
                    mov         temp0,temp2                 ;muevo decenas:unidades a temp0
                    andi        temp0,0x0f                  ;elimino las decenas
                    ;ori            temp0,0x30              ;transformo a ASCII
                    sts         unidades,temp0              ;almaceno en la SRAM
                    andi        temp2,0xf0                  ;elimino las unidades
                    swap        temp2                       ;intercambio nibles
                    ;ori            temp2,0x30              ;transformo a ASCII
                    sts         decenas,temp2               ;almaceno en la SRAM

                    mov         temp0,temp3                 ;muevo miles:centeas a temp0
                    andi        temp0,0x0f                  ;elimino las unidades de mil
                    ;ori            temp0,0x30              ;transformo a ASCII
                    sts         centenas,temp0              ;almaceno en la SRAM
                    andi        temp3,0xf0                  ;elimino las centenas
                    swap        temp3                       ;intercambio nibles
                    ;ori            temp3,0x30              ;transformo a ASCII
                    sts         miles,temp3                 ;almaceno en la SRAM

                    ;ori            temp4,0x30              ;transformo las decenas de mil a ASCII
                    sts         decenas_mil,temp4           ;almaceno en la SRAM

                    pop         temp6                       ;recupero el contenido de temp6 en el stack
                    pop         temp5                       ;recupero el contenido de temp5 en el stack
                    pop         temp4                       ;recupero el contenido de temp4 en el stack
                    pop         temp3                       ;recupero el contenido de temp3 en el stack
                    pop         temp2                       ;recupero el contenido de temp2 en el stack
                    ret
;-------------------------------------------------------------------------------------------

lcd_datos:          ldi         xh,high(tx0_buffer)
                    ldi         xl,low(tx0_buffer)
                    ldi         temp1,0x30
                    lds         temp0,temperatura
                    lsr         temp0
                    brcc        binario_cambio
                    ldi         temp1,0x35
binario_cambio:     sts         decimal,temp1
                    clr         temp1
                    call        bin2bcd
                    ldi         temp0,0x83
                    call        lcd_command
                    call        imprima_lcd
imprima_decimas:
                    ldi         temp0,'.'
                    st          x+,temp0
                    call        lcd_senddata
                    lds         temp0,decimal
                    st          x+,temp0
                    call        lcd_senddata

                    ldi         temp0,','
                    st          x+,temp0

                    ldi         temp0,' '
                    call        lcd_senddata
                    ldi         temp0,' '
                    call        lcd_senddata


imprima_rpm:        lds         temp0,adc_rpm   ;mov            temp0,Q_l
                    clr         temp1           ;mov            temp1,Q_m
                    call        bin2bcd
                    ldi         temp0,0xc5
                    call        lcd_command
                    call        imprima_lcd

                    ldi         temp0,','
                    st          x+,temp0

                    ldi         temp0,' '
                    call        lcd_senddata
                    ldi         temp0,' '
                    call        lcd_senddata


imprima_nivel:      lds         temp0,adc_nivel ;adc_promedio
                    clr         temp1
                    call        bin2bcd
                    ldi         temp0,0x8d
                    call        lcd_command
                    call        imprima_lcd

                    ldi         temp0,0x0a          ;estas lineas se han agregado
                    st          x+,temp0            ;estas lineas se han agregado
                    ldi         temp0,0x0d          ;estas lineas se han agregado
                    st          x+,temp0            ;estas lineas se han agregado
                    ldi         temp0,0x00          ;estas lineas se han agregado
                    st          x+,temp0            ;estas lineas se han agregado
```

```
                    ldi         temp0,','
                    st          x+,temp0
                    lds         temp0,rx1_buffer+14
                    st          x+,temp0
                    ldi         temp0,','
                    st          x+,temp0
                    lds         temp0,rx1_buffer+34
                    st          x+,temp0
                    lds         temp0,rx1_buffer+35
                    st          x+,temp0
                    lds         temp0,rx1_buffer+36
                    st          x+,temp0
                    lds         temp0,rx1_buffer+37
                    st          x+,temp0
                    ldi         temp0,','
                    st          x+,temp0
                    lds         temp0,rx1_buffer+21
                    st          x+,temp0
                    lds         temp0,rx1_buffer+22
                    st          x+,temp0
                    lds         temp0,rx1_buffer+23
                    st          x+,temp0
                    lds         temp0,rx1_buffer+24
                    st          x+,temp0
                    ldi         temp0,0x0a
                    st          x+,temp0
                    ldi         temp0,0x0d
                    st          x+,temp0
                    ldi         temp0,0x00
                    st          x+,temp0

                    ldi         temp0,' '
                    call        lcd_senddata
                    ldi         temp0,' '
                    call        lcd_senddata

                    lds         temp0,tx0_buffer
                    sts         udr0,temp0

                    ldi         xh,high(tx0_buffer+1)
                    ldi         xl,low(tx0_buffer+1)

                    ret
imprima_lcd:
                    lds         temp0,decenas_mil
                    cpi         temp0,0x00
                    brne        imprima_dec_mil

                    lds         temp0,miles
                    cpi         temp0,0x00
                    brne        imprima_miles

                    lds         temp0,centenas
                    cpi         temp0,0x00
                    brne        imprima_centenas

                    lds         temp0,decenas
                    cpi         temp0,0x00
                    brne        imprima_decenas
                    jmp         imprima_unidades

imprima_dec_mil:
                    ori         temp0,0x30
                    st          x+,temp0
                    call        lcd_senddata

imprima_miles:
                    lds         temp0,miles
                    ori         temp0,0x30
                    st          x+,temp0
                    call        lcd_senddata
imprima_centenas:
                    lds         temp0,centenas
                    ori         temp0,0x30
                    st          x+,temp0
                    call        lcd_senddata
imprima_decenas:
                    lds         temp0,decenas
                    ori         temp0,0x30
                    st          x+,temp0
                    call        lcd_senddata
imprima_unidades:
                    lds         temp0,unidades
                    ori         temp0,0x30
                    st          x+,temp0
                    call        lcd_senddata
                    ret
;***********************************Subrutinas del LCD*********************************

;*************************************inicializacion***********************************

lcd_init:           call        lcd_reset
                    clr         temp0
                    sts         lcd_status,temp0
                    ldi         temp0,0x28              ;Function set: 4bits, 2 Line, 8-bit, 5x7 dots
                    call        lcd_command             ;envie el comando
                    call        lcd_on
                    call        lcd_cursor_on
                    call        lcd_cls
                    call        lcd_entry
                    ldi         temp0,0x80
                    call        lcd_command
                    ret                                 ;Return from routine
;*************************************Reseteo del LCD**********************************

lcd_reset:          ldi     temp0,255
                    out     lcd_port,temp0
                    call    wait_40m
                    ldi     temp0,0x03
                    cbi     lcd_port,LCD_rs         ;Selected command register
                    bst     temp0,3
                    bld     temp1,lcd_d7
                    bst     temp0,2
                    bld     temp1,lcd_d6
                    bst     temp0,1
                    bld     temp1,lcd_d5
                    bst     temp0,0
                    bld     temp1,lcd_d4
                    out     lcd_port,temp1
                    sbi     lcd_port,lcd_en         ;Enable H->L
                    cbi     lcd_port,lcd_en
                    call    wait_30m
                    sbi     lcd_port,lcd_en         ;Enable H->L
                    cbi     lcd_port,lcd_en
                    call    wait_10m
                    sbi     lcd_port,lcd_en         ;Enable H->L
                    cbi     Lcd_port,lcd_en
                    call    wait_10m
                    ldi     temp0,0x02              ;indico 4 lineas de datos
                    cbi     lcd_port,lcd_rs         ;Selected command register
                    bst     temp0,3
                    bld     temp1,lcd_d7
                    bst     temp0,2
```

```
                    bld     temp1,lcd_d6
                    bst     temp0,1
                    bld     temp1,lcd_d5
                    bst     temp0,0
                    bld     temp1,lcd_d4
                    out     lcd_port,temp1
                    sbi     lcd_port,lcd_en         ;Enable H->L
                    cbi     Lcd_port,lcd_en
                    call    wait_10m
                    ret
;************************************Comandos del LCD************************************

lcd_on:             lds     temp0,lcd_status
                    ori     temp0,0b00001100
                    sts     lcd_status,temp0
                    jmp     lcd_command

lcd_off:            lds     temp0,lcd_status
                    andi    temp0,0b11111011
                    sts     lcd_status,temp0
                    jmp     lcd_command

lcd_cursor_on:      lds     temp0,lcd_status
                    ori     temp0,0b00001010
                    sts     lcd_status,temp0
                    jmp     lcd_command

lcd_cursor_off:     lds     temp0,lcd_status
                    andi    temp0,0b11111101
                    sts     lcd_status,temp0
                    jmp     lcd_command

lcd_blink_on:       lds     temp0,lcd_status
                    ori     temp0,0b00001101
                    sts     lcd_status,temp0
                    jmp     lcd_command

lcd_blink_off:      lds     temp0,lcd_status
                    andi    temp0,0b11111110
                    sts     lcd_status,temp0
                    jmp     lcd_command

lcd_entry:          ldi     TEMP0,0X06              ;Entry mode, auto increment with no shift, puede
                                                    ;tener valores de 0x04, 0x05, 0x06, 0x07
                    jmp     lcd_command
lcd_home:           ldi     TEMP0,0X02
                    jmp     lcd_command

lcd_shift_r:        ldi     temp0,0x1c
                    jmp     lcd_command

lcd_shift_l:        ldi     temp0,0x18
                    jmp     lcd_command

lcd_cursor_r:       ldi     temp0,0x14
                    jmp     lcd_command

lcd_cursor_l:       ldi     temp0,0x10
                    jmp     lcd_command

lcd_cls:            ldi     TEMP0,0X01              ;Clear LCD
                    jmp     lcd_command
;************************************Envio de comandos al LCD************************************

LCD_command:        clr     temp1
                    cbi     lcd_port,lcd_rs         ;Selected command register
                    bst     temp0,7
                    bld     temp1,lcd_d7
                    bst     temp0,6
                    bld     temp1,lcd_d6
                    bst     temp0,5
                    bld     temp1,lcd_d5
                    bst     temp0,4
                    bld     temp1,lcd_d4
                    out     lcd_port,temp1
                    sbi     lcd_port,lcd_en         ;Enable H->L
                    nop
                    nop
                    cbi     lcd_port,lcd_en
                    bst     temp0,3
                    bld     temp1,lcd_d7
                    bst     temp0,2
                    bld     temp1,lcd_d6
                    bst     temp0,1
                    bld     temp1,lcd_d5
                    bst     temp0,0
                    bld     temp1,lcd_d4
                    out     lcd_port,temp1
                    nop
                    nop
                    cbi     lcd_port,lcd_en
                    ldi     yh,high(2000)
                    ldi     yl,low(2000)
                    call    wait_us                 ;Wait for LCD to process the command
                    ret                             ;Return from command routine
;************************************Envio de datos al LCD************************************

LCD_senddata:       clr     temp1
                    bst     temp0,7
                    bld     temp1,lcd_d7
                    bst     temp0,6
                    bld     temp1,lcd_d6
                    bst     temp0,5
                    bld     temp1,lcd_d5
                    bst     temp0,4
                    bld     temp1,lcd_d4
                    out     lcd_port,temp1          ;mueve  a lcd_port los 4 bits mas significativos
                    sbi     lcd_port,lcd_rs         ;Selected data register
                    sbi     lcd_port,lcd_en         ;Enable H->L
                    nop
                    nop
                    cbi     Lcd_port,lcd_en
                    bst     temp0,3
                    bld     temp1,lcd_d7
                    bst     temp0,2
                    bld     temp1,lcd_d6
                    bst     temp0,1
                    bld     temp1,lcd_d5
                    bst     temp0,0
                    bld     temp1,lcd_d4
                    out     lcd_port,temp1          ;mueve  a lcd_port los 4 bits menos significativos
                    sbi     lcd_port,lcd_rs         ;Selected data register
                    sbi     lcd_port,lcd_en         ;Enable H->L
                    nop
                    nop
                    cbi     lcd_port,lcd_en
                    ldi     yh,high(2000)
                    ldi     yl,low(2000)
                    call    wait_us                 ;Wait for LCD to process the command
                    ret                             ;Return from busy routine
;************************************Envio de strings al LCD************************************

lcd_print_sub:      lpm     temp0,z+
```

```
lcd_print_sub:      lpm     temp0,z+
                    cpi     temp0,0
                    breq    lcd_print_exit
                    rcall   lcd_senddata            ;send first char
                    rjmp    lcd_print_sub           ;jump back to send the next character
lcd_print_exit:     ret                             ;End of routine
;*****************************************************************************************
;**                                                                                     **
;*****************************************Tablas*****************************************
;**                                                                                     **
;*****************************************************************************************
CLR_:
.db                 "@CLR",13,10,0,0
OI1:
.db                 "@OI 1",13,10,0
OI5:
.db                 "@OI 5",13,10,0
OI10:
.db                 "@OI 10",13,10,0,0
NC:
.db                 "@NC 00001000",13,10,0,0
```