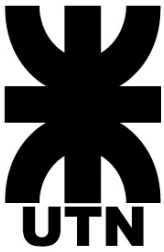
IVÁN TALIJANCIC



UNIVERSIDAD TECNOLÓGICA NACIONAL
Facultad Regional Reconquista

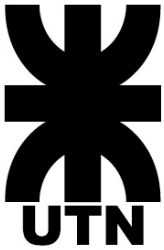
SISTEMA DE TELEMETRÍA REMOTO PARA SUB ESTACIONES TRANSFORMADORAS

Proyecto Final presentado en cumplimiento de las exigencias de la Carrera de Ingeniería Electromecánica de la Facultad Regional Reconquista, realizado por el estudiante Iván Talijancic.

Asesor: Prof: Ing. Alejandro Gregoret

Reconquista, Santa Fe
República Argentina
Año 2015

IVÁN TALIJANCIC



UNIVERSIDAD TECNOLÓGICA NACIONAL
Facultad Regional Reconquista

SISTEMA DE TELEMETRÍA REMOTO PARA SUB ESTACIONES TRANSFORMADORAS

Reconquista
Año 2015

AGRADECIMIENTOS

A mis padres y familia por acompañarme en cada decisión y desafío que asumo en mi vida.

A la Universidad Tecnológica Nacional y más específicamente a la Facultad Regional Reconquista, por brindarme la posibilidad de formarme como Ingeniero Electromecánico.

A los ingenieros: Alejandro Gregoret, Claudio Starna e Ivan Silvestri, quienes me permitieron realizar mis Prácticas Profesionales Supervisadas en su empresa ControlARG y cumplir el sueño de combinar mi vocación, y profesión con lo que amo. Brindándome todo lo necesario para cumplimentar mis prácticas con éxito y culminar mi carrera con la realización del presente proyecto final de carrera.

A todos y cada uno de los docentes que se mostraron predispuestos y desinteresados para evacuar mis dudas y consultas y colaborar en mi formación profesional.

A todas aquellas personas que han colaborado conmigo de alguna u otra manera a lo largo de esta etapa de mi vida.

DEDICATORIA

Dedico este esfuerzo personal y este logro académico y profesional, en primera instancia a mis padres, quienes me dieron todo lo que estuvo a su alcance y más para que pueda alcanzar mi sueño de ser ingeniero.

Y en segunda a instancia al resto de mi familia, hermanas, tíos y abuela quienes fueron fundamentales y estuvieron siempre presente durante el largo camino que tuve que recorrer.

ÍNDICE GENERAL

1 MEMORIA DESCRIPTIVA

1.1 Delimitación del tema

Este proyecto propone diseñar un sistema de telemetría remoto de para subestaciones transformadoras que consiste en un conjunto de módulos funcionales, ver FIGURA 1.3 y 1.4.

Desde una perspectiva general el sistema puede desglosarse en tres grandes módulos o bloques funcionales:

1. Módulo de Adquisición de Datos.
2. Módulo de Comunicación.
3. Módulo de Gestión de Datos o Servidor de Almacenamiento de Datos.

Ver FIGURAS 1.1 y 1.2.

1.1.1 Módulo de Medición y Adquisición de Datos:

Es el módulo encarga de sensar las diferentes magnitudes de la subestación transformadora que nos interesa censar, a este conjunto de variables a medir y adquirir, las separamos en:

1.1.1.1 Parámetros Eléctricos Funcionales:

Este módulo consiste en un multi-medidor, que nos permitirá sensar los parámetros eléctricos fundamentales en el funcionamiento de una subestación distribuidora, como ser:

- Potencia Aparente
- Potencia Activa
- Potencia Reactiva
- Corrientes
- Tensiones
- Cos(ϕ)
- Energía Activa y Reactiva

Para llevar a cabo estas mediciones el mismo requiere de transformadores de tensión y transformadores de corriente.

Es posible instalar más de un módulo de medición en cada SET (Sub Estación Transformadora), lo que permitirá, no solamente analizar los parámetros del transformador, sino también los parámetros eléctricos de cada uno de los alimentadores en baja tensión que surjan de cada SET.

1.1.1.2 Parámetros Físicos Funcionales:

Dentro de este conjunto de parámetros, nos encontramos con los siguientes:

- Temperatura de Cuba (alarma y desconexión, en caso de que el trasformador disponga de la protección).
- Nivel de Aceite (alarma y desconexión, en caso de que el transformador disponga de la protección).
- Estado del Relé Buchholz (alarma y desconexión, en caso de que el transformador disponga de la protección).
- Informe de la Temperatura y temperatura ambiente (medida por medio de una sonda Pt100).
- Informe del estado de los seccionadores de línea.

Es importante destacar que estos parámetros físicos funcionales son fundamentales a la hora de efectuar una operación segura del transformador, permitiendo hacer uso del mismo dentro de los rangos admisible de diseño de la maquina (especificados por su fabricante), a modo de evitar averías y no atentar con la vida útil del mismo.

1.1.2 Módulo de Comunicación:

1.1.2.1 Comunicación con el servidor

El sistema dispondrá de diferentes medios de comunicación con el servidor central.

Uno de ellos es a través de un modem GSM/GPRS que se encargará de tomar los datos, adquiridos por el módulo anteriormente descripto, y enlazarlos al servidor.

Este módulo de comunicación, utilizará el protocolo TCP/IP para enviar los datos referentes a la telemetría de la subestación transformadora y hará uso del sistema de SMS (Short Menssage Service), para hacer aviso de alarmas. Como ser, en el caso, de que alguno de los parámetros físicos funcionales del transformador se encuentre fuera de los umbrales seguros/normales de funcionamiento, el sistema comunicará acerca de dicha anomalía haciendo uso del servicio de SMS.

En caso de estar ubicado el sistema fuera del área de cobertura de celulares o redes urbanas, también será posible conectar al servidor central por equipos inalámbricos (como ser enlace de radiofrecuencia).

El otro de los medios de comunicación con el que contamos es a través de una conexión cableada mediante Ethernet (también conocido como estándar IEEE 802.3). Se hará uso de este medio en los casos en los que no sea necesario utilizar una comunicación inalámbrica y para funciones de configuración del equipo.

1.1.2.2 Bus Posterior

El bus posterior, comprende la comunicación con los diferentes periféricos ver FIGURAS 1.3 y 1.4. Esta comunicación se implementa sobre el protocolo I2C (Inter-Integrated Circuit).

Se opta por emplear este protocolo para manejar el bus posterior o interno del sistema, debido a las facilidades que el mismo brinda a la hora de manejar muchos esclavos (dispositivos pertenecientes a la red de comunicación que responden a las peticiones del maestro) en tareas de escritura, así como lectura de los mismos.

1.1.3 Servidor:

El mismo consistirá en un Servidor que se encarga de recibir los datos enviados por el módulo de comunicación (modem 3G/GPRS) y/o red Ethernet y almacenarlos en una base de datos para luego servir de datos al sistema SCADA (Sistema de Control y Adquisición de Datos).

El SCADA representará el estado en tiempo real de la red de distribución urbana, permitiendo conocer en cada instante los estados de alarma y sobrecarga de las mismas y tomar acciones en consecuencia.

2 Esquema del sistema en su totalidad

2.1 Utilizando comunicación GSM/GPRS:

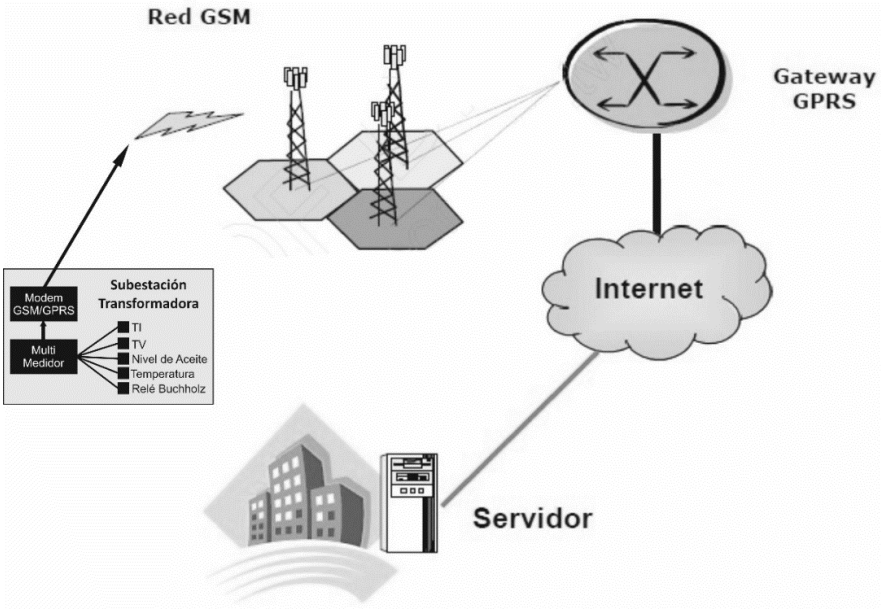


FIGURA 1.1 – Esquema del sistema utilizando conexión GPRS con el servidor.

2.2 Utilizando una conexión Ethernet con el servidor:

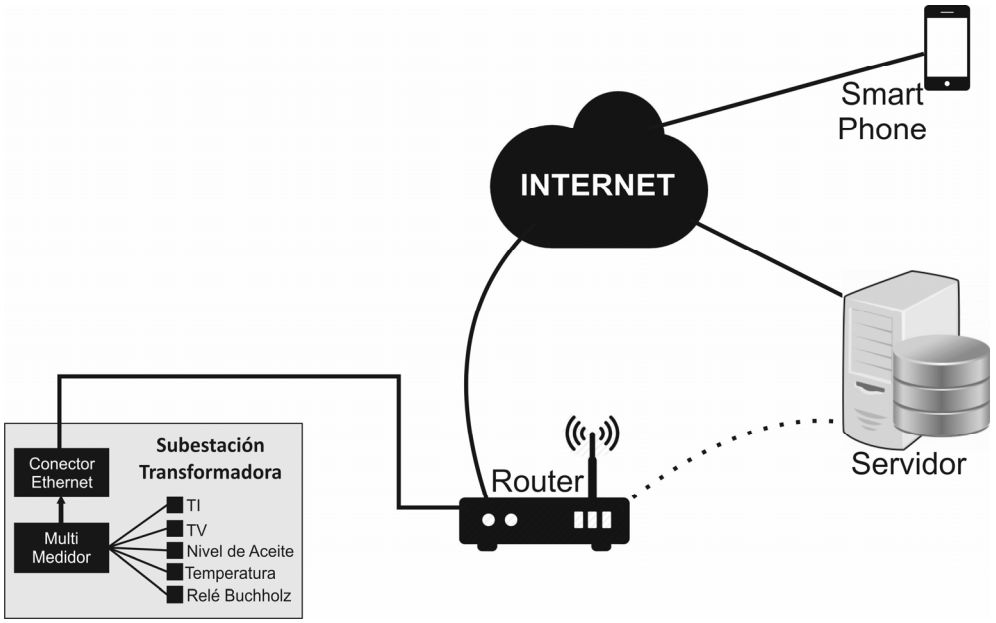


FIGURA 1.2 – Esquema del sistema haciendo uso de una conexión cablead Ethernet

2.3 Esquema del Concepto Modular del Sistema

En el diseño del sistema se plantea un concepto modular, en donde tengamos un "cerebro o CPU" que estará encargado de comunicarse con los periféricos, como medidores de energía, GPS, modem GSM/GPRS, módulos de entradas/salidas digitales y entradas analógicas y en donde cada uno de estos periféricos pueda ir anexándose al CPU, según las necesidades de cada caso.

Como mínimo el sistema contara con el CPU y un módulo de medición de energía.

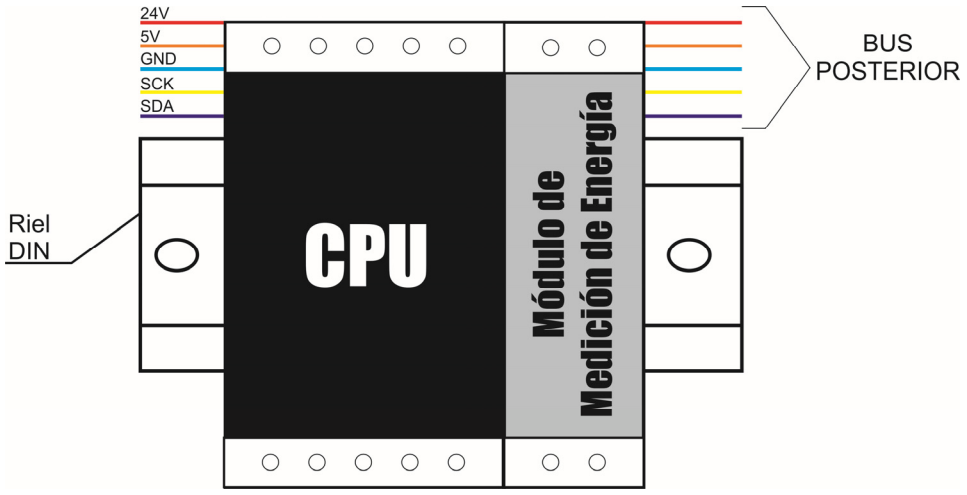


FIGURA 1.3 – Esquema modular del sistema en su configuración mínima

Cabe destacar que en su configuración mínima el equipo no es capaz de manejar una comunicación GPRS/GSM, para ello necesita del anexado del módulo de comunicación GPRS/GSM. En la configuración básica la comunicación con el servidor se logra mediante Ethernet, ver FIGURA 1.2.

La idea es que mediante la configuración modular del equipo, el usuario pueda adaptar el sistema a su aplicación particular, por ejemplo en la siguiente imagen vemos, como además de los dos módulos básicos se anexaron el módulo de comunicación GSM/GPRS y el módulo GPS.

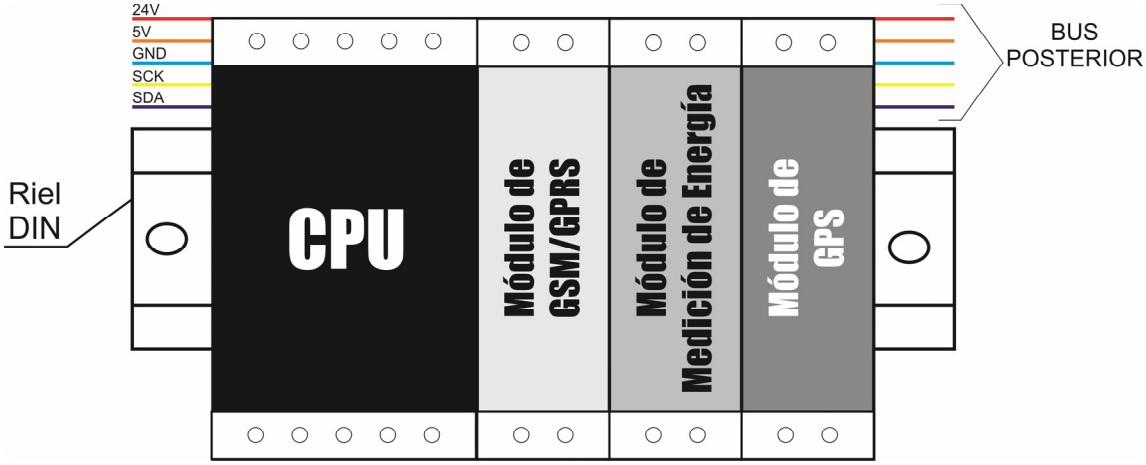


FIGURA 1.4 – Esquema Modular del Sistema

2 INTRODUCCIÓN

2.1 Definición del Problema

El problema que se plantea en el presente proyecto final de carrera, consiste en el desarrollo tanto del hardware, como del software de los distintos módulos funcionales que integran un sistema de telemetría remoto para subestaciones transformadoras, como anteriormente se explicó.

Tomando como punto de partida al CPU del sistema y continuando por los demás módulos:

- Módulo de Medición de Energía.
- Módulo GSM/GPRS
- Módulo GPS

Logrando en todos los casos obtener un prototipo funcional que permita poner a prueba los programas planteados y demostrar si la idea en general es viable.

2.2 Objetivo

Como resultado del desarrollo de este proyecto final de carrera, se obtendrán prototipos funcionales de una serie de módulos que integran un sistema de telemetría remoto para subestaciones transformadoras, *que pretende solucionar el problema concreto y real que se presenta a la hora de conocer el estado, en tiempo real, de dichas instalaciones.* El cuál permitirá, en primera instancia y como objetivo a corto plazo:

- Determinar perfiles de tensión que permitan caracterizar las cargas.
- En caso de sobre carga (sobre corriente), dar aviso de la señal de alarma por medio de SMS al encargado de mantenimiento.
- En caso de elevarse excesiva y peligrosamente la temperatura, dar aviso de la señal de alarma por medio de SMS al encargado de mantenimiento.

Mientras que como objetivo a largo plazo, si pensamos en la potencial inserción del sistema, en una Smart Grid, la implementación del mismo brindaría las siguientes ventajas:

- Registro en tiempo real de cada uno de los alimentadores en baja tensión y transformadores.
- Disminución de cortes intempestivos (al conocer el estado térmico de los transformadores, se puede prevenir su desconexión por temperatura).
- Si es necesario efectuar cortes programados, preverlos con anticipación de modo de minimizar los inconvenientes.

- Disminuir los cortes por sobrecarga, al poder informar a los usuarios que disminuyan el consumo en alimentadores comprometidos.
- Disminuir los tiempos de reparaciones, pues el sistema estará vinculado a la guardia, informando en cada instante del estado de los diferentes alimentadores.
- Reordenar los consumos con datos reales, permitiendo derivar cargas de un transformador a otro.
- Disminuir pérdidas técnicas y no técnicas.
- Protección de los transformadores.
- Contar con datos que permitan orientar las decisiones operativas hacia un uso más eficiente de recursos y un mejoramiento en la calidad del servicio de distribución de energía eléctrica.

2.3 Justificación

La justificación del desarrollo se ve reflejada en el gran número de ventajas que la creación de un sistema tal traería aparejado, solucionando una problemática real, que se puede observar claramente a nivel regional y en un ámbito más extenso.

2.4 Organización del Proyecto

El proyecto está estructurado en capítulos, dentro de los cuales se presenta en primera instancia un capitulo que expone los fundamentos teóricos básicos en los cuales se sustenta el desarrollo del presente proyecto final de carrera, luego una serie de capítulos que describen en detalle el proceso de desarrollo de cada uno de los módulos que componen el sistema en su totalidad, un capítulo se dedicó a explicar el protocolo de comunicación del bus posterior (I2C), así como desarrollar una breve exposición acerca de las experimentaciones efectuadas y los resultados obtenidos, en otro capítulo se explicó las utilidades del Stack TCP/IP de Microchip utilizado, para comprender las utilidades que el mismo presenta desde el punto de vista de la aplicación práctica y por último se expone un presupuesto y breve análisis económico del proyecto.

Como anexo se presentan todos los catálogos, hojas de datos (Data Sheets), notas de aplicación y la bibliografía consultada.

Además de los planos complementarios y un DVD con información de extensión para aquellos que estén interesados en profundizar en algún tema en particular de los diferentes desarrollados a lo largo del proyecto. En el mismo DVD se presentan video en los cuales se muestran experiencias de laboratorio desarrolladas para elaborar el proyecto.

3 FUNDAMENTOS TEÓRICOS

3.1 Introducción

El desarrollo de mi proyecto final consiste, como anteriormente se describió en el diseño de un módulo de comunicación y sus módulos de periféricos, los cuales son parte integral de un sistema de telemetría remoto de subestaciones distribuidoras de energía eléctrica. Para llevar a cabo este cometido me base en dispositivos electrónicos particulares, conocidos como microcontroladores, los mismos son en palabras simples circuitos integrados que pueden ser programados para llevar a cabo tareas específicas. A continuación se explicara con mayor detalle el funcionamiento de los microcontroladores, su arquitectura y programación. Por último se describirán los microcontroladores utilizados en el proyecto, sus características fundamentales y el porqué de su elección.

3.2 ¿Qué es un Microcontrolador?

FIGURA 3.1 – Imagen Microcontrolador PIC16F883

Un microcontrolador (abreviado μC, UC o MCU) es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento, memoria y periféricos de entrada/salida.

Algunos microcontroladores pueden utilizar palabras de 8 bits y funcionan a velocidad de reloj con frecuencias tan bajas como 4 kHz, con un consumo de baja potencia (mW o microvatios). Por lo general, tendrá la capacidad de mantenerse a la espera de un evento como pulsar un botón; así, el consumo de energía durante el estado de reposo (reloj de la CPU y los periféricos de la mayoría) puede ser sólo de nanovatios, lo que hace que muchos de ellos sean muy adecuados para aplicaciones con batería de larga duración. Otros microcontroladores

pueden servir para roles de rendimiento crítico, donde sea necesario actuar más como un procesador digital de señal (DSP), con velocidades de reloj y consumo de energía más altos.

Cuando es fabricado el microcontrolador, no contiene datos en la memoria ROM. Para que pueda controlar algún proceso es necesario generar o crear y luego grabar en la ROM (la cual puede ser flash o eeprom) o equivalente del microcontrolador algún programa, el cual puede ser escrito en lenguaje ensamblador u otro lenguaje para microcontroladores, sin embargo, para que el programa pueda ser grabado en la memoria del microcontrolador, debe ser codificado en sistema numérico hexadecimal que es finalmente el sistema que hace trabajar al microcontrolador cuando éste es alimentado con el voltaje adecuado y asociado a dispositivos analógicos y discretos para su funcionamiento.

La situación actual en el campo de los microcontroladores se ha alcanzado gracias al desarrollo de la tecnología de fabricación de los circuitos integrados. Este desarrollo ha permitido construir centenas de miles de transistores en un chip. Esto fue una condición previa para la fabricación de un microprocesador. Las primeras microcomputadoras se fabricaron al añadirles periféricos externos, tales como memoria, líneas de entrada/salida, temporizadores u otros. El incremento posterior de la densidad de integración permitió crear un circuito integrado que contenía tanto al procesador como periféricos. *Así es cómo fue desarrollada la primera microcomputadora en un solo chip, denominada más tarde microcontrolador*. [1]

3.2.1 ¿Microcontrolador y Microprocesador?

Es sencillo dar por sentado que son lo mismo un microcontrolador y un microprocesador y esto no es cierto, ya que difieren uno del otro en muchos sentidos. La primera y la más importante diferencia es su funcionalidad.

Para utilizar al microprocesador en una aplicación real, se debe conectar con componentes tales como memoria o buses de transmisión de datos. Aunque el microprocesador se considera una máquina de computación poderosa, no está preparado para la comunicación con los dispositivos periféricos con los cuales se conecta. Para que el microprocesador se comunique con algún periférico, se deben utilizar circuitos especiales.

Por otro lado, un microcontrolador está diseñado de manera tal que tenga todos los componentes integrados en un solo chip, de modo tal que no necesite de otros componentes especializados para su aplicación, porque todos los circuitos necesarios, ya se encuentran incorporados. Ver FIGURA 3.2

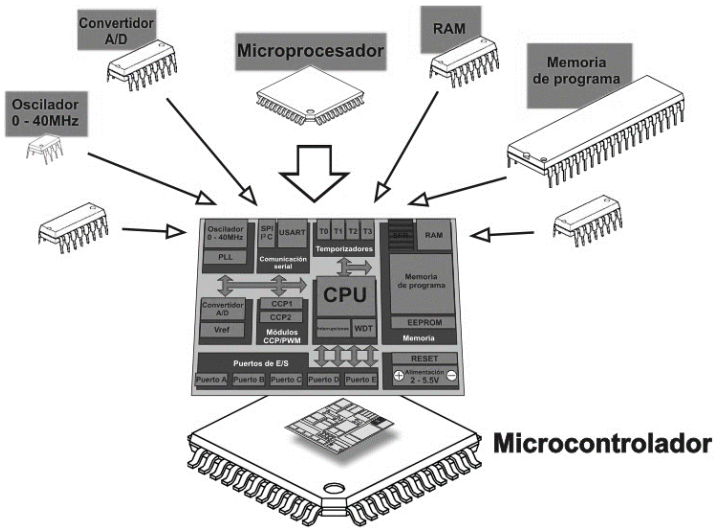


FIGURA 3.2 – Esquema funcional de un Microcontrolador

3.3 Conceptos Fundamentales

3.3.1 Unidad Central de Procesamiento (Central Proces Unit - CPU)

Como indica su nombre, esto es una unidad que controla todos los procesos dentro del microcontrolador. Consiste en varias unidades más pequeñas, de las cuales, las más importantes son:

- *Decodificador de instrucciones* es la parte que decodifica las instrucciones del programa y acciona otros circuitos basándose en esto. El "Juego de Instrucciones (Ver 3.3.3)" hace referencia a la capacidad de esta unidad componente de la CPU.
- *Unidad lógica aritmética (Arithmetical Logical Unit - ALU)* realiza todas las operaciones matemáticas y lógicas sobre datos.
- *Acumulador o registro de trabajo.* Es un registro SFR (ver 3.3.3 y FIGURA 3.4) estrechamente relacionado con el funcionamiento de la ALU. Es utilizado para almacenar todos los datos sobre los que se debe realizar cada operación (sumar, mover).

FIGURA 3.3 – Unidad Central de Procesos

3.3.2 Registro

Un registro o una celda de memoria es un circuito eléctrico, capaz de memorizar/guardar el estado de un byte.

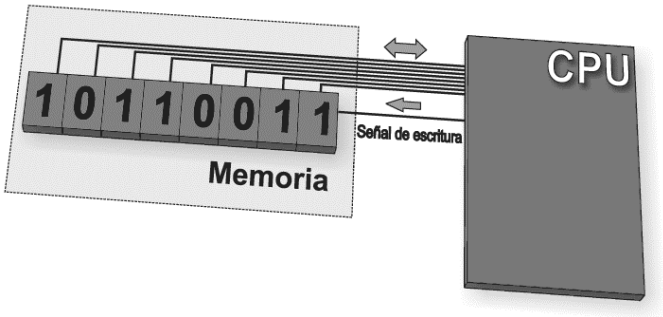


FIGURA 3.4 – Esquema de un registro de memoria

3.3.3 Registros SFR

A diferencia de los registros que no tienen ninguna función especial y predeterminada, cada microcontrolador dispone de un número de registros de funciones especiales (SFR), con la función predeterminada por el fabricante. Sus bits están conectados a los circuitos internos del microcontrolador tales como temporizadores, convertidores A/D, osciladores entre otros. Esto significa que directamente manejan el funcionamiento de estos circuitos, o sea del microcontrolador. Imagínese ocho interruptores que manejan el funcionamiento de un circuito pequeño dentro del microcontrolador. Los registros SFR hacen exactamente lo mismo.

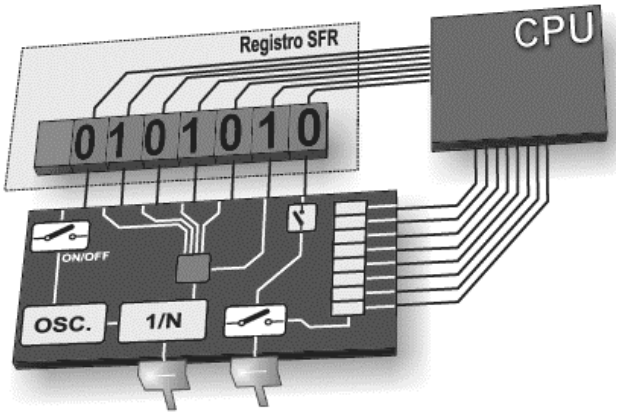


FIGURA 3.5 – Registros de Funciones Especiales

3.3.4 Unidad de Memoria

La unidad de memoria es la parte del microcontrolador utilizada para almacenar los datos. La manera más fácil de explicarlo es compararlo con un armario grande con muchos cajones. Si marcamos los cajones claramente, será fácil acceder a cualquiera de sus contenidos al leer la etiqueta en la parte delantera del cajón.

De manera similar, cada dirección de memoria corresponde a una dirección de memoria. El contenido de cualquier dirección se puede leer y escribir.

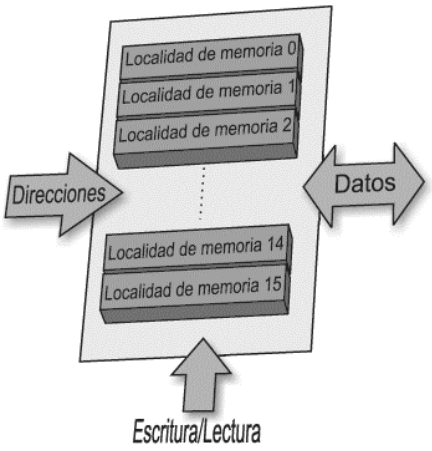


FIGURA 3.6 – Unidad de Memoria

Hay varios tipos de memorias dentro del microcontrolador:

3.3.4.1 Memoria ROM

La memoria ROM (Read Only Memory o Memoria de Solo Lectura) se utiliza para *guardar permanentemente* el programa que se está ejecutando. El tamaño de programa que se puede escribir depende del tamaño de esta memoria. Los microcontroladores actuales

normalmente utilizan el direccionamiento de 16 bits, que significa que son capaces de direccionar hasta 64 Kb de memoria, o sea 65535 localidades.

3.3.4.2 Memoria RAM

La memoria RAM (Random Access Memory o Memoria de Acceso Aleatorio) es un tipo de memoria que tiene la particularidad de perder el contenido almacenado, al desconectar la fuente de alimentación. Se utiliza para almacenar temporalmente los datos y los resultados inmediatos creados y utilizados durante el funcionamiento del microcontrolador. Por ejemplo, si el programa ejecuta la adición (de cualquier cosa) es necesario tener un registro que representa lo que se llama "suma" en vida cotidiana. Con tal propósito, uno de los registros de la RAM es denominado "suma" y se utiliza para almacenar los resultados de la adición.

3.3.5 Arquitectura Interna

El concepto de arquitectura interna del microcontrolador, hace referencia al modo en que se estructura el intercambio de datos entre la CPU y la memoria del mismo.

Todos los microcontroladores actuales utilizan uno de dos modelos básicos de arquitectura denominados Harvard y von-Neumann.

3.3.5.1 Arquitectura de Von-Neumann

Los microcontroladores que utilizan la arquitectura von- Neumann disponen de un solo bloque de memoria y de un bus de datos de 8 bits. Como todos los datos se intercambian por medio de estas 8 líneas, este bus está sobrecargado, y la comunicación por sí misma es muy lenta e ineficaz. *La CPU puede leer una instrucción o leer/escribir datos de/en la memoria*. Los dos procesos no pueden ocurrir a la vez puesto que las instrucciones y los datos utilizan el mismo bus.

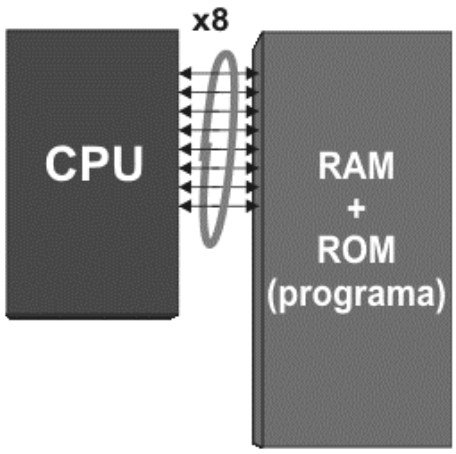


FIGURA 3.7 - Arquitectura de Von-Neumann

3.3.5.2 Arquitectura de Harvard

Los microcontroladores que utilizan esta arquitectura disponen de dos buses de datos diferentes. Uno es de 8 bits de ancho y conecta la CPU con la memoria RAM. El otro consiste en varias líneas (12, 14 o 16) y conecta a la CPU y la memoria ROM. Por consiguiente, la CPU puede leer las instrucciones y realizar el acceso a la memoria de datos a la vez.

Esto permite subsanar muchos de los inconvenientes que se presentan con la arquitectura anteriormente descripta. Brindándonos como resultado de ellos microcontroladores más rápidos, eficientes y eficaces a la hora de ejecutar las instrucciones de programa.

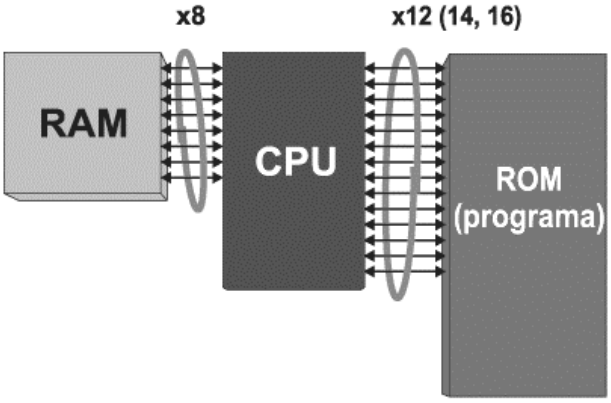


FIGURA 3.8 – Arquitectura de Harvard

3.3.6 Juego de Instrucciones

El nombre colectivo de todas las instrucciones que puede entender el microcontrolador es llamado Juego de Instrucciones. Los fabricantes aceptan cualquiera de los dos enfoques descritos a continuación:

3.3.6.1 RISC (Reduced Instruction Set Computer) - Computadora con Juego de Instrucciones Reducidas

En este caso la idea es que el microcontrolador reconoce y ejecuta sólo operaciones básicas (sumar, restar, copiar etc...) Las operaciones más complicadas se realizan al combinar éstas (por ejemplo, multiplicación se lleva a cabo al realizar adición sucesiva). *Es como intentar explicarle a alguien con pocas palabras cómo llegar al aeropuerto en una nueva ciudad.* Sin embargo, no todo es tan oscuro. Además, el microcontrolador es muy rápido así que no es posible ver todas las "acrobacias" aritméticas que realiza. El usuario sólo puede ver el resultado final de todas las operaciones. Por último, no es tan difícil explicar dónde está el aeropuerto si se utilizan las palabras adecuadas tales como: a la derecha, a la izquierda, el kilómetro etc.

3.3.6.2 CISC (Complex Instruction Set Computer) - Computadoras con un juego de instrucciones complejo

¡CISC es opuesto a RISC! Los microcontroladores diseñados para reconocer más de 200 instrucciones diferentes realmente pueden realizar muchas cosas a alta velocidad. No obstante, uno debe saber cómo utilizar todas las posibilidades que ofrece un lenguaje tan rico, lo que no es siempre tan fácil.

3.3.7 Oscilador

Los pulsos uniformes generados por el oscilador permiten el funcionamiento armónico y síncrono de todos los circuitos del microcontrolador. El oscilador se configura normalmente de tal manera que utilice un cristal de cuarzo o resonador cerámico para estabilización de frecuencia. Además, puede funcionar como un circuito autónomo (como oscilador RC). *Es importante decir que las instrucciones del programa no se ejecutan a la velocidad impuesta por el mismo oscilador sino varias veces más despacio.* Eso ocurre porque cada instrucción se ejecuta en varios ciclos del oscilador. En algunos microcontroladores se necesita el mismo número de ciclos para ejecutar todas las instrucciones, mientras que en otros el tiempo de ejecución no es el mismo para todas las instrucciones.

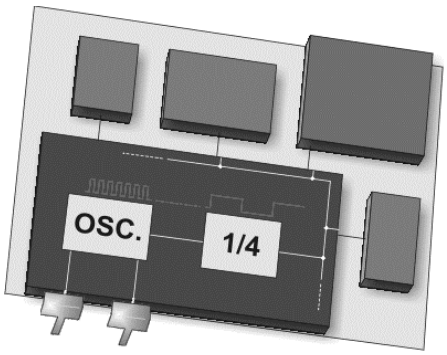


FIGURA 3.9 - Oscilador

3.3.8 Puertos de Entrada/Salida (E/S)

Para hacer útil un microcontrolador, hay que conectarlo a un dispositivo externo, o sea, a un periférico. Cada microcontrolador tiene uno o más registros (denominados puertos) conectados a los pines en el microcontrolador. ¿Por qué se denominan como puertos de entrada/salida? Porque usted puede cambiar la función de cada pin como quiera. Por ejemplo, usted desea que su dispositivo encienda y apague tres LEDs y que simultáneamente monitoree el estado lógico de 5 sensores o botones de presión. Uno de los puertos debe estar configurado de tal manera que haya tres salidas (conectadas a los LEDs) y cinco entradas (conectadas a los

sensores). *Eso se realiza simplemente por medio de software, lo que significa que la función de algún pin puede ser cambiada durante el funcionamiento.*

Una de las características más importantes de los pines de entrada/salida (E/S) es la corriente máxima que pueden entregar/recibir. En la mayoría de los microcontroladores la corriente obtenida de un pin es suficiente para activar un LED u otro dispositivo de baja corriente *(10-20mA)*.

Cada puerto de E/S normalmente está bajo el control de un registro SFR especializado, lo que significa que cada bit de ese registro determina el estado del pin correspondiente en el microcontrolador. Por ejemplo, al escribir un uno lógico (1) a un bit del registro de control (SFR), el pin apropiado del puerto se configura automáticamente como entrada. Eso significa que el voltaje llevado a ese pin se puede leer como 0 o 1 lógico. En caso contrario, al escribir 0 al registro SFR, el pin apropiado del puerto se configura como salida. Su voltaje (0V o 5V) corresponde al estado del bit apropiado del registro del puerto (0 o 1 lógico, respectivamente).

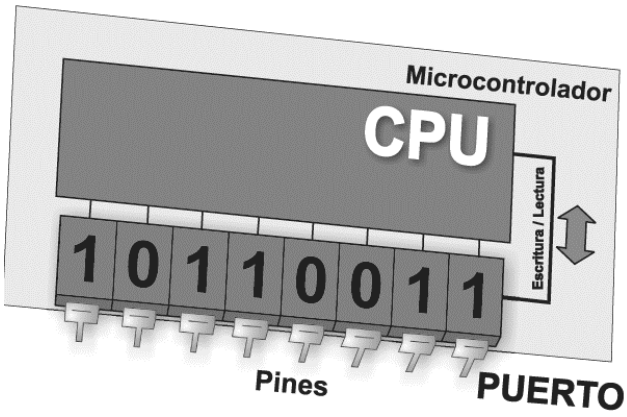


FIGURA 3.10 – Puerto de E/S

3.3.9 Interrupción

La mayoría de programas utilizan interrupciones durante su ejecución. El propósito del microcontrolador generalmente consiste en reaccionar a los cambios en su entorno. En otras palabras, cuando ocurre algo, el microcontrolador reacciona de alguna manera... Por ejemplo, al apretar el botón del mando a distancia, el microcontrolador lo registra y responde al comando cambiando de canal, subiendo o bajando el volumen etc. Si el microcontrolador pasará la mayoría del tiempo comprobando varios botones sin parar - las horas, los días, esto no sería nada práctico.

Por lo tanto, el microcontrolador "aprendió un truco" durante su evolución. En vez de seguir comprobando algún pin o bit, el microcontrolador deja su "trabajo de esperar" a un "experto" que reaccionará sólo en caso de que suceda algo digno de atención.

La señal que informa al procesador central acerca de tal acontecimiento se denomina INTERRUPCIÓN.

3.3.10 Comunicación Serie

Hoy en día, la mayoría de los microcontroladores llevan incorporados varios sistemas diferentes para la comunicación en serie, como un equipamiento standard. Cuál de estos sistemas se utilizará en un caso concreto, depende de muchos factores, de los cuales, los más importantes son:

¿Con cuántos dispositivos el microcontrolador tiene que intercambiar los datos?

¿Cuál es la velocidad del intercambio de datos obligatoria?

¿Cuál es la distancia entre los dispositivos?

¿Es necesario transmitir y recibir los datos simultáneamente?

Una de las cosas más importantes en cuanto a la comunicación en serie es el Protocolo que debe ser estrictamente observado. Es un conjunto de reglas que se aplican obligatoriamente para que los dispositivos puedan interpretar correctamente los datos que intercambian mutuamente. Afortunadamente, los microcontroladores se encargan de eso automáticamente.

La velocidad de transmisión serial (baud rate) es el término utilizado para denotar el número de bits transmitidos por segundo [bps]. *Fíjese que este término se refiere a bits, y no a bytes*. El protocolo normalmente requiere que cada byte se transmita junto con varios bits de control. Eso quiere decir que un byte en un flujo de datos serial puede consistir en 11 bits. Por ejemplo, si velocidad de transmisión serial es 300 bps un máximo de 37 y un mínimo de 27 bytes se pueden transmitir por segundo.

Los sistemas de comunicación serial, más utilizados son:

3.10.1 I2C (Inter Integrated Circuit) - Circuito Inter-Integrado

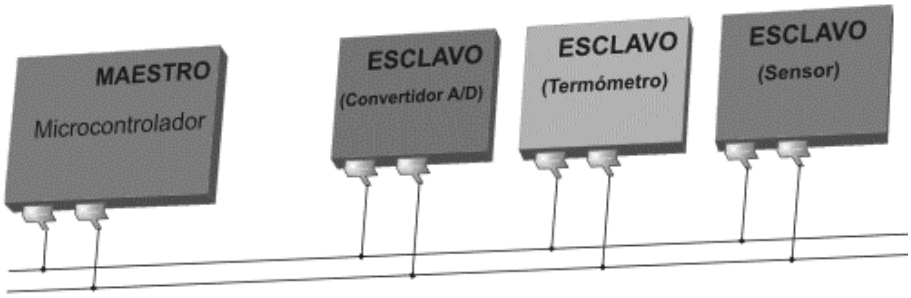


FIGURA 3.11 – Bus I2C

I2C es un sistema para el intercambio de datos serial entre microcontroladores y diversos periféricos, así como otros microcontroladores. Se utiliza cuando la distancia entre ellos es corta (el receptor y el transmisor están normalmente en la misma placa de circuito impreso). La conexión se establece por medio de dos líneas - una se utiliza para transmitir los datos, mientras que la otra se utiliza para la sincronización (la señal de reloj). Como se muestra en la figura, un dispositivo es siempre el principal (master - maestro), el que realiza el direccionamiento de un chip subordinado (slave - esclavo) antes de que se inicie la comunicación. De esta manera un microcontrolador puede comunicarse con 112 dispositivos diferentes. La velocidad de transmisión serial es normalmente 100 Kb/seg (el modo estándar) o 10 Kb/seg (modo de velocidad de transmisión baja). Recientemente han aparecido los sistemas con la velocidad de transmisión serial 3.4 Mb/sec. La distancia entre los dispositivos que se comunican por el bus I2C está limitada a unos metros.

Este es el protocolo de comunicación que utilizo en mi proyecto para llevar a cabo las comunicaciones en lo que denominaos bus posterior, es decir las comunicaciones entre el módulo de comunicación (o CPU del sistema) y los diferentes periféricos, como son placas de entradas y salidas digitales (GPIO – General Propouse Input Ouput), entradas analógicas, GPS y medidor/es de energía.

3.10.2 SPI (Serial Peripheral Interface bus) - Bus Serial de Interfaz de Periféricos

Un bus serial de interfaz de periféricos es un sistema para la comunicación serial que utiliza hasta cuatro líneas (normalmente solo son necesarias tres) - para recibir los datos, para transmitir los datos, para sincronizar y (opcional) para seleccionar el dispositivo con el que se comunica. Es de funcionamiento Full-Duplex, lo que significa que permite el envío y recepción simultánea de datos.

La velocidad de transmisión máxima es mayor que en el sistema de conexión I2C.

Este es el protocolo de comunicación que utilizo en mi proyecto para comunicar el micro del módulo de comunicación con el integrado que maneja la comunicación Ethernet y también para comunicarse con una tarjeta de memoria microSD.

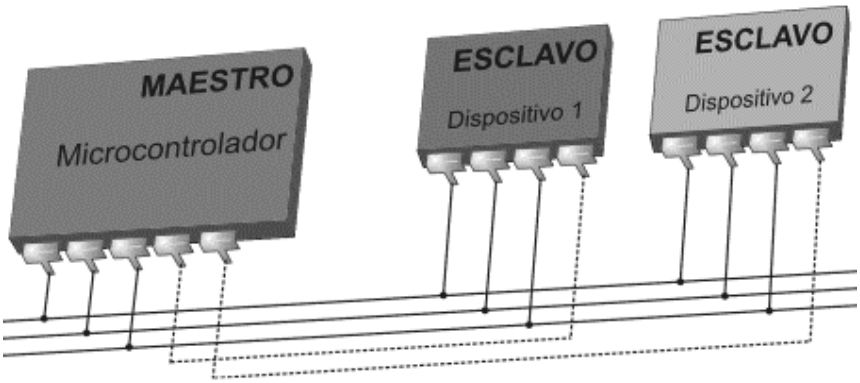


FIGURA 3.12 – Bus SPI

3.10.3 UART (Universal Asynchronous Receiver/Transmitter) - Transmisor-Receptor Asíncrono Universal

Este tipo de conexión es asíncrona, lo que significa que no se utiliza una línea especial para transmitir la señal de reloj. En algunas aplicaciones este rasgo es crucial (por ejemplo, en mandar datos a distancia por RF o por luz infrarroja). Puesto que se utiliza sólo una línea de comunicación, tanto el receptor como el transmisor reciben y envían los datos a la misma velocidad, la cual ha sido predefinida para mantener la sincronización necesaria. Esto es una manera simple de transmitir datos, puesto que básicamente representa una conversión de datos de 8 bits de paralelo a serial. La velocidad de transmisión no es alta, es hasta 1 Mbit/sec.

Este es el protocolo que utilizo en el desarrollo del módulo de comunicación, para conectar el microcontrolador con el modem GSM/GPRS.

3.4 Microcontroladores PIC

Los PIC, de Microchip, son una opción más dentro del vasto mercado de microcontroladores. La popularidad de estos micros radica en su alta disponibilidad en el mercado y bajo precio.

El fabricante ha procurado una difusión exhaustiva de información relativa a sus productos, lo cual ha traído como consecuencia un proliferado uso de este tipo de microcontroladores. Algunos de los profesionales y aficionados que los utilizan difunden sus desarrollos e inventos por Internet lo cual ha promovido su uso. *Muchos consideran que los PIC son los más fáciles de programar.*

Por otro lado, se han desarrollado una serie de herramientas de bajo costo por parte de terceros (empresas, profesionales y aficionados), como son programadores, software, etc., que facilitan el uso y programación de estos dispositivos.

Compiladores de C y Basic están disponibles para programar a los PIC, lo cual facilita muchísimo el desarrollo de aplicaciones, acortando los tiempos y disminuyendo los costos de los proyectos.

Por estos motivos es que para llevar a cabo el desarrollo del presente proyecto he utilizado microcontroladores PIC, de la firma Microchip.

3.5 Microcontroladores Utilizados en el Proyecto:

Básicamente, fueron dos los microcontroladores los utilizados. Un microcontrolador de alta gama *dsPIC33FJ128GP204*, que hace las veces de cerebro o CPU de todo el módulo de comunicación, manejando las comunicaciones en el bus posterior, así como la comunicación remota con el servidor en el cual se almacenan los datos y otro de gama media PIC16F883 para los módulos de periféricos.

En las siguientes páginas se muestran las características principales de cada uno de estos microcontroladores, las mismas son extracciones de las hojas de datos, para conocer con más detalle las características dirigirse al Anexo N° II o al DVD adjunto al presente informe.

3.5.1 dsPIC33FJ128GP204

MICROCHIP

dsPIC33FJ32GP302/304, dsPIC33FJ64GPX02/X04, and dsPIC33FJ128GPX02/X04

16-bit Digital Signal Controllers (up to 128 KB Flash and 16K SRAM) with Advanced Analog

Operating Conditions

- 3.0V to 3.6V, -40ºC to +150ºC, DC to 20 MIPS
- 3.0V to 3.6V, -40ºC to +125ºC, DC to 40 MIPS

Clock Management

- 2% internal oscillator
- Programmable PLL and oscillator clock sources
- Fail-Safe Clock Monitor (FSCM)
- Independent Watchdog Timer
- Low-power management modes
- Fast wake-up and start-up

Core Performance

- Up to 40 MIPS 16-bit dsPIC33F CPU
- Single-cycle MUL plus hardware divide

Advanced Analog Features

- 10/12-bit ADC with 1.1Msps/500 ksps rate:
  - Up to 13 ADC input channels and four S&H
  - Flexible/Independent trigger sources
- 150 ns Comparators:
  - Up to two Analog Comparator modules
  - 4-bit DAC with two ranges for Analog Comparators

Input/Output

- Software remappable pin functions
- 5V-tolerant pins
- Selectable open drain and internal pull-ups
- Up to 5 mA overvoltage clamp current/pin
- Multiple external interrupts

System Peripherals

- 16-bit dual channel 100 ksps Audio DAC
- Cyclic Redundancy Check (CRC) module
- Up to five 16-bit and up to two 32-bit Timers/Counters
- Up to four Input Capture (IC) modules
- Up to four Output Compare (OC) modules
- Real-Time Clock and Calendar (RTCC) module

Communication Interfaces

- Parallel Master Port (PMP)
- Two UART modules (10 Mbps)
  - Supports LIN 2.0 protocols
  - RS-232, RS-485, and IrDA® support
- Two 4-wire SPI modules (15 Mbps)
- Enhanced CAN (ECAN) module (1 Mbaud) with 2.0B support
- I^2C module (100K, 400K and 1Mbaud) with SMbus support
- Data Converter Interface (DCI) module with I^2S codec support

Direct Memory Access (DMA)

- 8-channel DMA with no CPU stalls or overhead
- UART, SPI, ADC, ECAN, IC, OC, INT0

Qualification and Class B Support

- AEC-Q100 REVG (Grade 0 -40ºC to +150ºC)
- Class B Safety Library, IEC 60730, VDE certified

Debugger Development Support

- In-circuit and in-application programming
- Two program breakpoints
- Trace and run-time watch

Packages

Type	SPDIP	SOIC	QFN-S	QFN	TQFP
Pin Count	28	28	28	44	44
I/O Pins	21	21	21	35	35
Contact Lead/Pitch	.100"	1.27	0.65	0.65	0.80
Dimensions	.285x.135x1.365"	7.50x2.05x17.9	6x6x0.9	8x8x0.9	10x10x1

Note: All dimensions are in millimeters (mm) unless specified.

dsPIC33FJ32GP302/304, dsPIC33FJ64GPX02/X04, AND dsPIC33FJ128GPX02/X04 PRODUCT FAMILIES

The device names, pin counts, memory sizes, and peripheral availability of each device are listed below. The following pages show their pinout diagrams.

TABLE 1: dsPIC33FJ32GP302/304, dsPIC33FJ64GPX02/X04, AND dsPIC33FJ128GPX02/X04 CONTROLLER FAMILIES

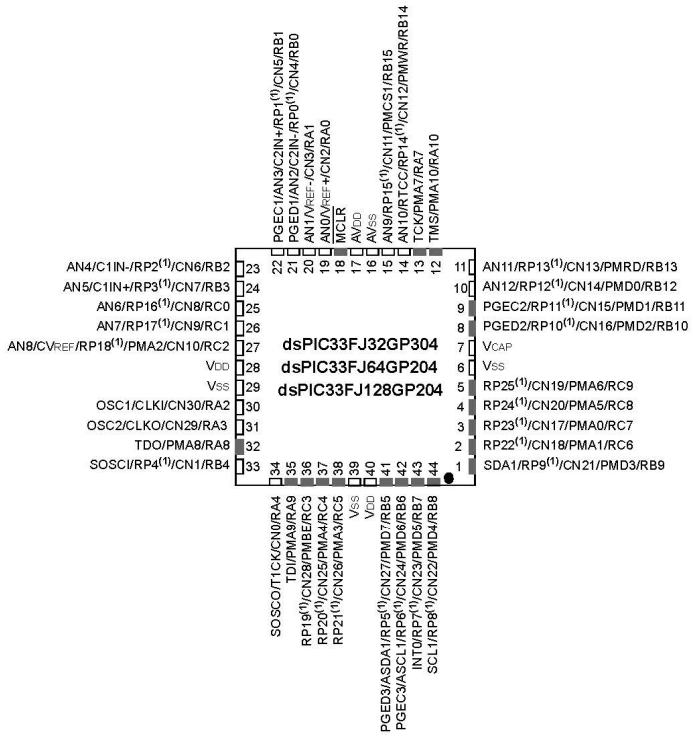
Device	Pins	Program Flash Memory (Kbyte)	RAM (Kbyte)(1)	Remappable Peripheral									RTCC	I²C™	CRC Generator	10-bit/12-bit ADC (Channels)	16-bit Audio DAC (Pins)	Analog Comparator (2 Channels/Voltage Regulator)	8-bit Parallel Master Port (Address Lines)	I/O Pins	Packages
				Remappable Pins	16-bit Timer(2)	Input Capture	Output Compare Standard PWM	Data Converter Interface	UART	SPI	ECAN™	External Interrupts(3)									
dsPIC33FJ128GP804	44	128	16	26	5	4	4	1	2	2	1	3	1	1	1	13	6	1/1	11	35	QFN TQFP
dsPIC33FJ128GP802	28	128	16	16	5	4	4	1	2	2	1	3	1	1	1	10	4	1/0	2	21	SPDIP SOIC QFN-S
dsPIC33FJ128GP204	44	128	8	26	5	4	4	1	2	2	0	3	1	1	1	13	0	1/1	11	35	QFN TQFP
dsPIC33FJ128GP202	28	128	8	16	5	4	4	1	2	2	0	3	1	1	1	10	0	1/0	2	21	SPDIP SOIC QFN-S
dsPIC33FJ64GP804	44	64	16	26	5	4	4	1	2	2	1	3	1	1	1	13	6	1/1	11	35	QFN TQFP
dsPIC33FJ64GP802	28	64	16	16	5	4	4	1	2	2	1	3	1	1	1	10	4	1/0	2	21	SPDIP SOIC QFN-S
dsPIC33FJ64GP204	44	64	8	26	5	4	4	1	2	2	0	3	1	1	1	13	0	1/1	11	35	QFN TQFP
dsPIC33FJ64GP202	28	64	8	16	5	4	4	1	2	2	0	3	1	1	1	10	0	1/0	2	21	SPDIP SOIC QFN-S
dsPIC33FJ32GP304	44	32	4	26	5	4	4	1	2	2	0	3	1	1	1	13	0	1/1	11	35	QFN TQFP
dsPIC33FJ32GP302	28	32	4	16	5	4	4	1	2	2	0	3	1	1	1	10	0	1/0	2	21	SPDIP SOIC QFN-S

Note 1: RAM size is inclusive of 2 Kbytes of DMA RAM for all devices except dsPIC33FJ32GP302/304, which include 1 Kbyte of DMA RAM.
2: Only four out of five timers are remappable.
3: Only two out of three interrupts are remappable.

Pin Diagrams (Continued)

44-Pin QFN(2)

■ = Pins are up to 5V tolerant

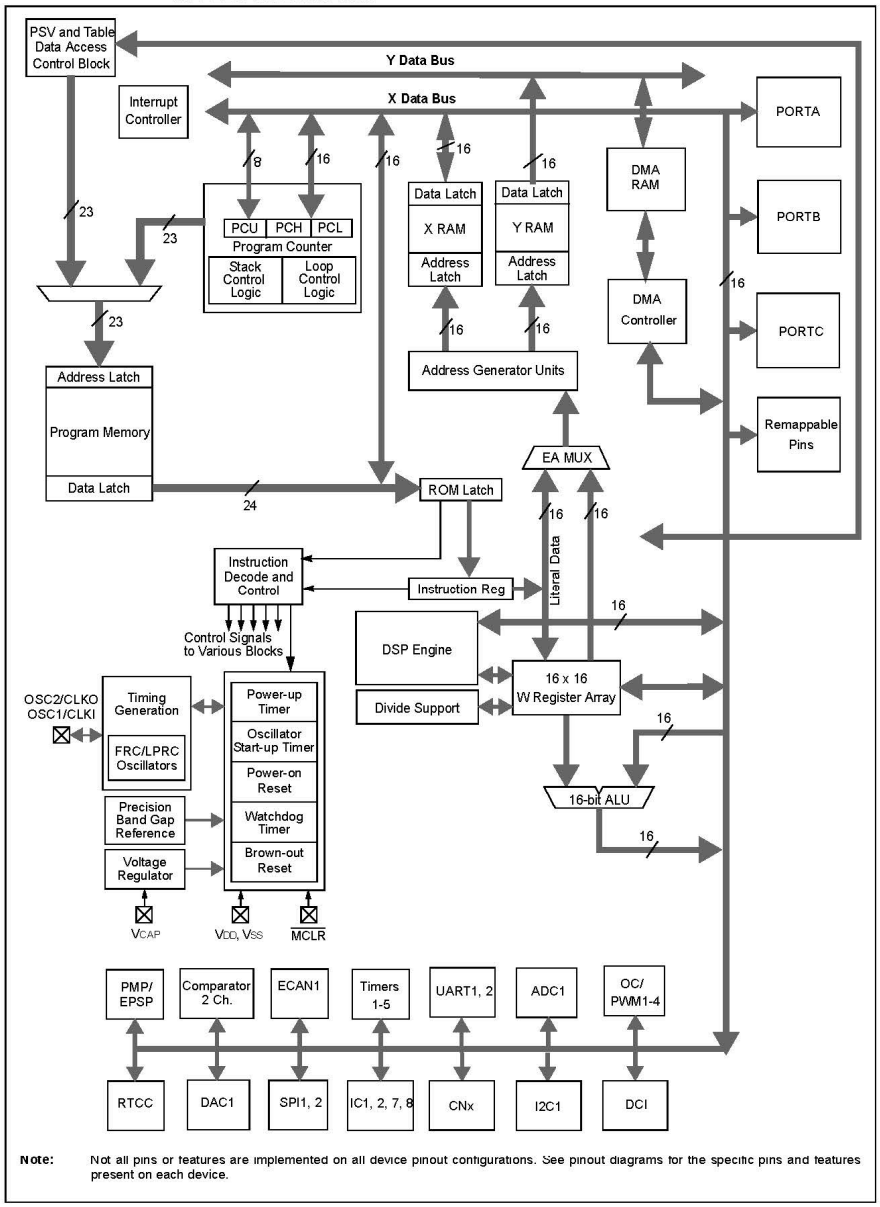


Note 1: The RPx pins can be used by any remappable peripheral. See Table 1 in this section for the list of available peripherals.

2: The metal plane at the bottom of the device is not connected to any pins and is recommended to be connected to VSS externally.

dsPIC33FJ32GP302/304, dsPIC33FJ64GPX02/X04, AND dsPIC33FJ128GPX02/X04

FIGURE 1-1: dsPIC33FJ32GP302/304, dsPIC33FJ64GPX02/X04, AND dsPIC33FJ128GPX02/X04 BLOCK DIAGRAM



Note: Not all pins or features are implemented on all device pinout configurations. See pinout diagrams for the specific pins and features present on each device.

3.5.2 PIC16F883

MICROCHIP

PIC16F882/883/884/886/887

28/40/44-Pin Flash-Based, 8-Bit CMOS Microcontrollers with nanoWatt Technology

High-Performance RISC CPU:

- Only 35 instructions to learn:
  - All single-cycle instructions except branches
- Operating speed:
  - DC – 20 MHz oscillator/clock input
  - DC – 200 ns instruction cycle
- Interrupt capability
- 8-level deep hardware stack
- Direct, Indirect and Relative Addressing modes

Special Microcontroller Features:

- Precision Internal Oscillator:
  - Factory calibrated to ±1%
  - Software selectable frequency range of 8 MHz to 31 kHz
  - Software tunable
  - Two-Speed Start-up mode
  - Crystal fail detect for critical applications
  - Clock mode switching during operation for power savings
- Power-Saving Sleep mode
- Wide operating voltage range (2.0V-5.5V)
- Industrial and Extended Temperature range
- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Brown-out Reset (BOR) with software control option
- Enhanced low-current Watchdog Timer (WDT) with on-chip oscillator (software selectable nominal 268 seconds with full prescaler) with software enable
- Multiplexed Master Clear with pull-up/input pin
- Programmable code protection
- High Endurance Flash/EEPROM cell:
  - 100,000 write Flash endurance
  - 1,000,000 write EEPROM endurance
  - Flash/Data EEPROM retention: > 40 years
- Program memory Read/Write during run time
- In-Circuit Debugger (on board)

Low-Power Features:

- Standby Current:
  - 50 nA @ 2.0V, typical
- Operating Current:
  - 11 μA @ 32 kHz, 2.0V, typical
  - 220 μA @ 4 MHz, 2.0V, typical
- Watchdog Timer Current:
  - 1 μA @ 2.0V, typical

Peripheral Features:

- 24/35 I/O pins with individual direction control:
  - High current source/sink for direct LED drive
  - Interrupt-on-Change pin
  - Individually programmable weak pull-ups
  - Ultra Low-Power Wake-up (ULPWU)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (CV_{REF}) module (% of V_{DD})
  - Fixed voltage reference (0.6V)
  - Comparator inputs and outputs externally accessible
  - SR Latch mode
  - External Timer1 Gate (count enable)
- A/D Converter:
  - 10-bit resolution and 11/14 channels
- Timer0: 8-bit timer/counter with 8-bit programmable prescaler
- Enhanced Timer1:
  - 16-bit timer/counter with prescaler
  - External Gate Input mode
  - Dedicated low-power 32 kHz oscillator
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Enhanced Capture, Compare, PWM+ module:
  - 16-bit Capture, max. resolution 12.5 ns
  - Compare, max. resolution 200 ns
  - 10-bit PWM with 1, 2 or 4 output channels, programmable “dead time”, max. frequency 20 kHz
  - PWM output steering control
- Capture, Compare, PWM module:
  - 16-bit Capture, max. resolution 12.5 ns
  - 16-bit Compare, max. resolution 200 ns
  - 10-bit PWM, max. frequency 20 kHz
- Enhanced USART module:
  - Supports RS-485, RS-232, and LIN 2.0
  - Auto-Baud Detect
  - Auto-Wake-Up on Start bit
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Master Synchronous Serial Port (MSSP) module supporting 3-wire SPI (all 4 modes) and I^2C™ Master and Slave Modes with I^2C address mask

 Preliminary

Pin Diagrams – PIC16F882/883/886, 28-Pin PDIP, SOIC, SSOP

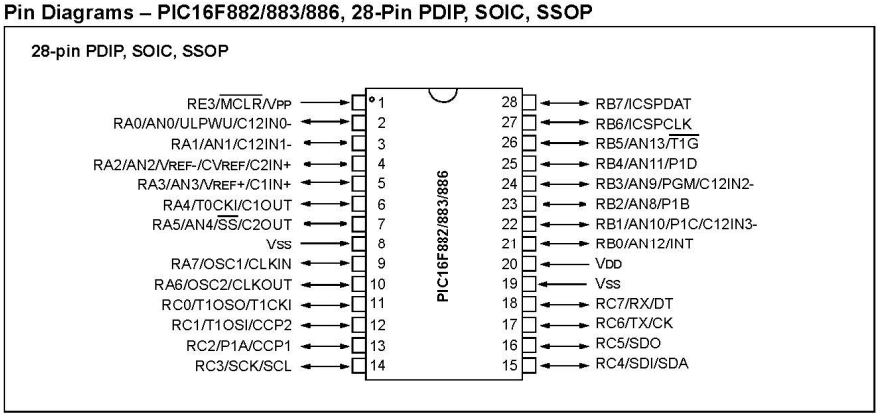


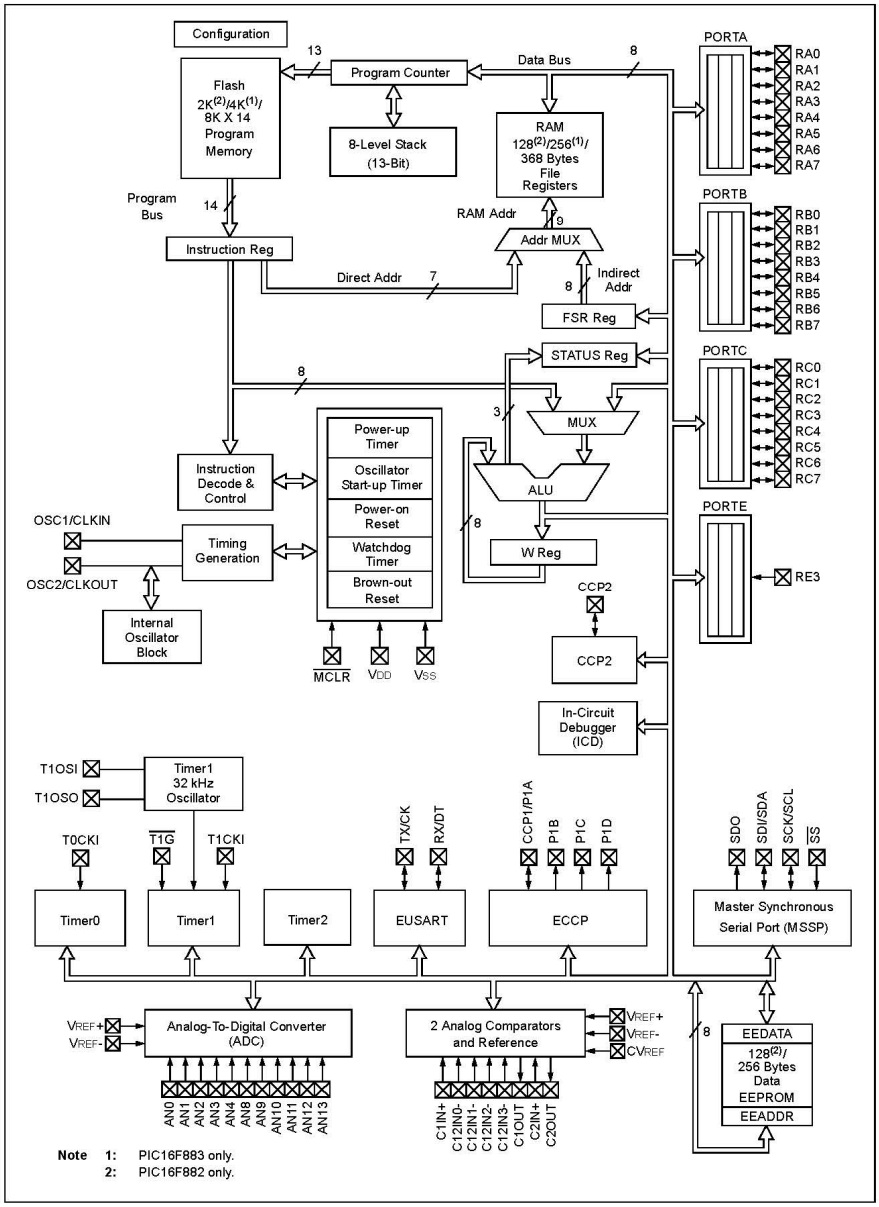
TABLE 1: PIC16F882/883/886 28-PIN SUMMARY (PDIP, SOIC, SSOP)

I/O	Pin	Analog	Comparators	Timers	ECCP	EUSART	MSSP	Interrupt	Pull-up	Basic
RA0	2	AN0/ULPWU	C12IN0-	—	—	—	—	—	—	—
RA1	3	AN1	C12IN1-	—	—	—	—	—	—	—
RA2	4	AN2	C2IN+	—	—	—	—	—	—	VREF-/CVREF
RA3	5	AN3	C1IN+	—	—	—	—	—	—	VREF+
RA4	6	—	C1OUT	T0CKI	—	—	—	—	—	—
RA5	7	AN4	C2OUT	—	—	—	SS	—	—	—
RA6	10	—	—	—	—	—	—	—	—	OSC2/CLKOUT
RA7	9	—	—	—	—	—	—	—	—	OSC1/CLKIN
RB0	21	AN12	—	—	—	—	—	IOC/INT	Y	—
RB1	22	AN10	C12IN3-	—	P1C	—	—	IOC	Y	—
RB2	23	AN8	—	—	P1B	—	—	IOC	Y	—
RB3	24	AN9	C12IN2-	—	—	—	—	IOC	Y	PGM
RB4	25	AN11	—	—	P1D	—	—	IOC	Y	—
RB5	26	AN13	—	T1G	—	—	—	IOC	Y	—
RB6	27	—	—	—	—	—	—	IOC	Y	ICSPCLK
RB7	28	—	—	—	—	—	—	IOC	Y	ICSPDAT
RC0	11	—	—	T1OSO/T1CKI	—	—	—	—	—	—
RC1	12	—	—	T1OSI	CCP2	—	—	—	—	—
RC2	13	—	—	—	CCP1/P1A	—	—	—	—	—
RC3	14	—	—	—	—	—	SCK/SCL	—	—	—
RC4	15	—	—	—	—	—	SDI/SDA	—	—	—
RC5	16	—	—	—	—	—	SDO	—	—	—
RC6	17	—	—	—	—	TX/CK	—	—	—	—
RC7	18	—	—	—	—	RX/DT	—	—	—	—
RE3	1	—	—	—	—	—	—	—	Y(1)	MCLR/VPP
—	20	—	—	—	—	—	—	—	—	VDD
—	8	—	—	—	—	—	—	—	—	VSS
—	19	—	—	—	—	—	—	—	—	VSS

Note 1: Pull-up activated only with external MCLR configuration.

PIC16F882/883/884/886/887

FIGURE 1-1: PIC16F882/883/886 BLOCK DIAGRAM



Note 1: PIC16F883 only.
2: PIC16F882 only.

4 SOFTWARES UTILIZADOS

4.1 Introducción

Para llevar a cabo la elaboración del presente proyecto se utilizaron diversas herramientas de software que permitieron la elaboración de los diferentes programas que se desarrollaron para los microcontroladores, así como aplicaciones de escritorio, para ordenadores que se usaron como asistentes en el desarrollo.

4.2 CCS

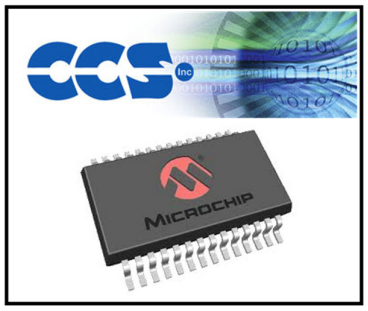


FIGURA 4.1 – Logo CCS

CCS Compiler es una muy útil herramienta de desarrollo de aplicaciones embebidas, que nos permite escribir y compilar programas en leguaje C y cuenta con la ventaja de tener un gran número de librerías, o aplicaciones pre construidas, que nos ayudan a disminuir en gran medida el tiempo de desarrollo.

En la siguiente imagen se muestra una captura de pantalla de la interface del programa.

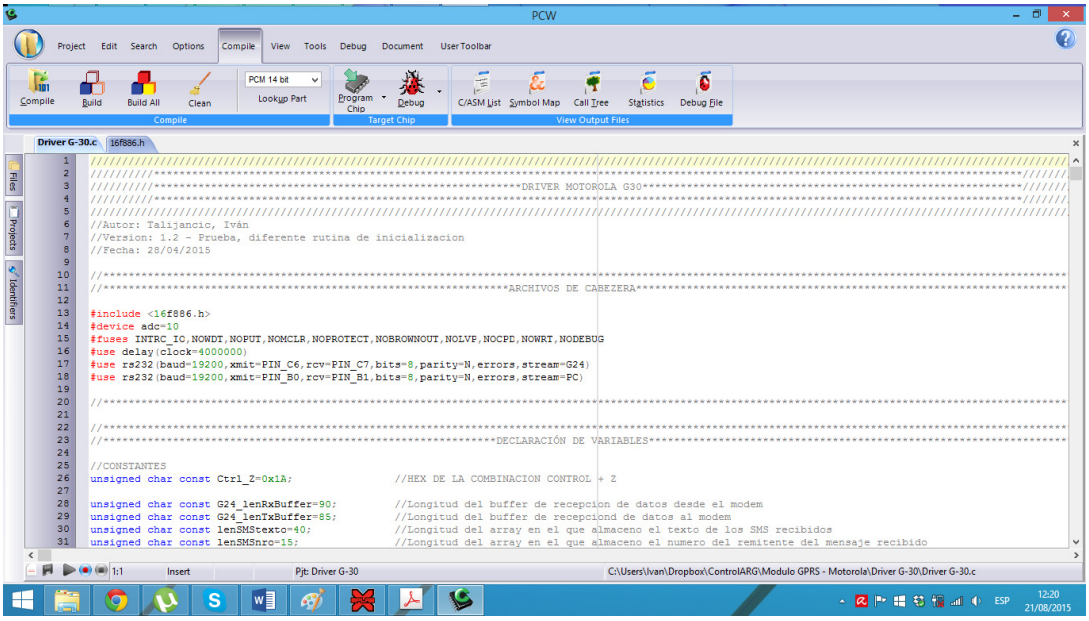


FIGURA 4.2 – Captura de Pantalla CCS

Esta fue la herramienta con la cual me inicié en el mundo de la programación de microcontroladores en leguaje C, ya que se presenta, en primera instancia, como la opción más atractiva a la hora de comenzar a programar, permitiéndonos conseguir considerable avances y resultados en períodos de tiempo relativamente cortos y focalizándonos solo en el software sin tener que tener un extenso conocimiento acerca del hardware de los micros. [2]

4.3 MPLABX

MPLAX es el entorno de desarrollo que nos proporciona el fabricante de microcontroladores Microchip para trabajar con sus productos. Es también un software de programación y compilación, en el cual podemos trabajar con múltiples lenguajes de codificación a saber, C, C++, ensamblador, entre otros.

Tiene la desventaja de exigirnos un menor nivel de abstracción del hardware del microcontrolador que queremos programar por lo cual resulta un entorno de desarrollo menos atractivos y agradable a la hora de comenzar a desarrollar aplicaciones embebidas con microcontroladores PIC. Pero una vez que se manejan estos conocimientos se presenta como la alternativa de excelencia para implementar nuestros programas, exigiéndonos un mayor compromiso intelectual, pero permitiéndonos un mayor entendimiento del funcionamiento del microcontrolador, así como de nuestros programas.

Esta es la herramienta en la cual se desarrollaron las versiones definitivas de los diferentes programas que se fueron creando en el desarrollo del presente proyecto. En la siguiente imagen se muestra la interface del programa.

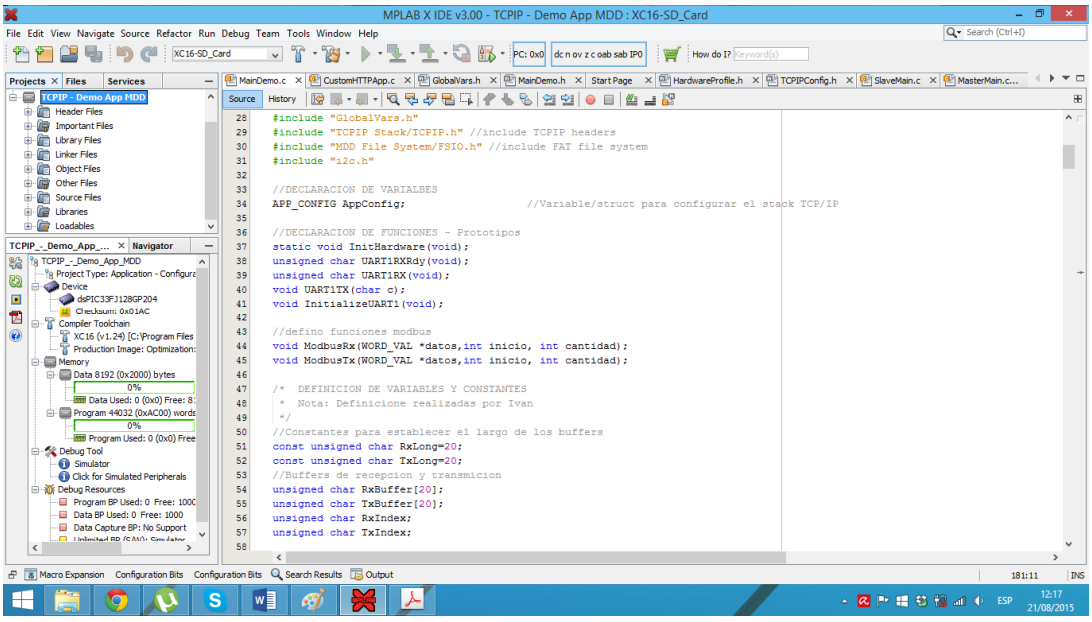


FIGURA 4.3 – Captura de Pantalla MPLAX

4.4 EAGLE

Eagle 6.2 es el programa que se utilizó para el diseño de los circuitos impresos, también llamados placas. El mismo es una plataforma de diseño, que nos permite plantear el esquemático del circuito (plano funcional del mismo) y posteriormente ejecutar el ruteo (diseño de las pistas de la placa) de dicho circuito. Es una herramienta esencial para realizar este trabajo, que cuenta con una muy extensa librería de componentes y además es de gran uso y difusión a nivel mundial, por ende es muy fácil encontrar en internet material de ayuda, así como librerías de componentes desarrolladas por particulares, lo cual facilita muchísimo la tarea de diseño de los circuitos impresos. Todos los circuitos impresos que se desarrollaron en el presente trabajo se hicieron con este programa.

A continuación se muestra capturas de pantalla de la interface del programa, en sus dos entornos de trabajo, esquemático y board o placa.

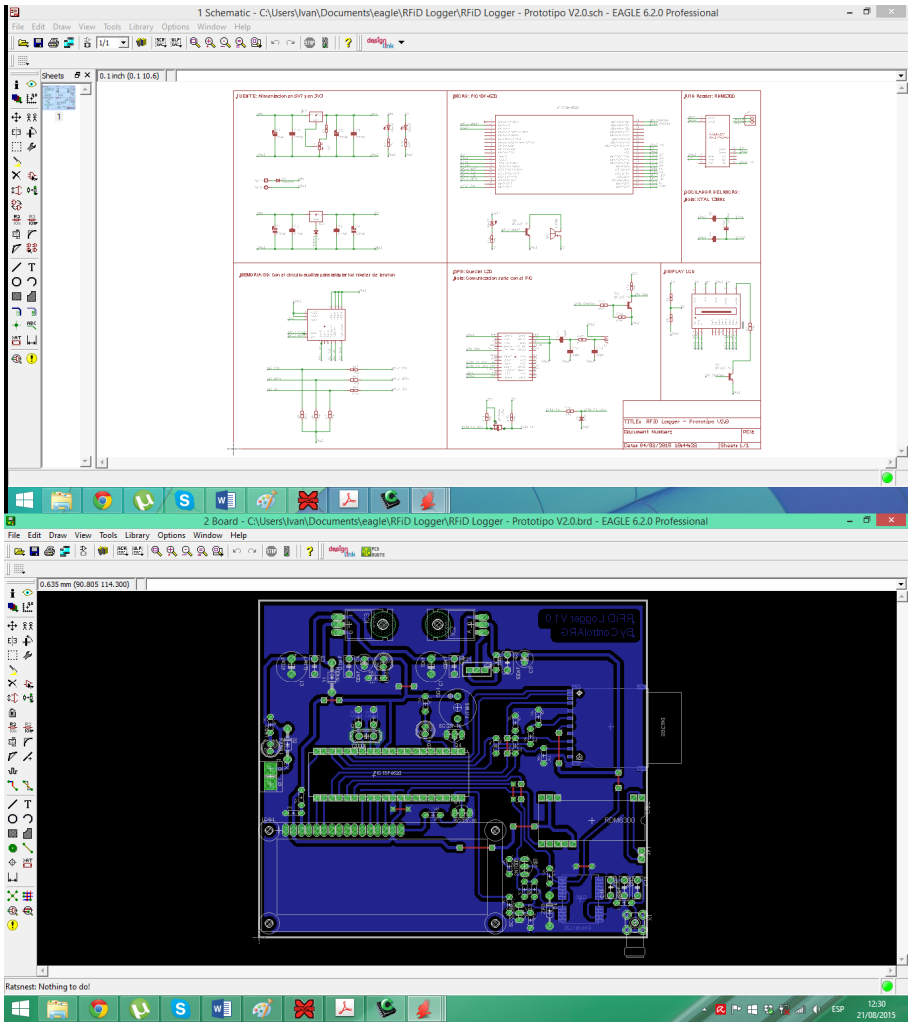


FIGURA 4.4 – Captura de Pantalla Eagle 6.2

4.5 LabVIEW

FIGURA 4.5 – Logo LabVIEW

LabVIEW (acrónimo de Laboratory Virtual Instrumentation Engineering Workbench) es una plataforma y entorno de desarrollo para diseñar sistemas, con un lenguaje de programación visual gráfico. Recomendado para sistemas hardware y software de pruebas, control y diseño, simulado o real y embebido, pues acelera la productividad. El lenguaje que usa se llama lenguaje G, donde la G simboliza que es lenguaje Gráfico.

Este programa fue creado por National Instruments (1976) para funcionar sobre máquinas MAC, salió al mercado por primera vez en 1986. Ahora está disponible para las plataformas Windows, UNIX, MAC y GNU/Linux.

Los programas desarrollados con LabVIEW se llaman Instrumentos Virtuales, o VIs, y su origen provenía del control de instrumentos, aunque hoy en día se ha expandido ampliamente no sólo al control de todo tipo de electrónica (Instrumentación electrónica) sino también a su programación embebida, comunicaciones, matemáticas, etc. Un lema tradicional de LabVIEW es: "La potencia está en el Software", que con la aparición de los sistemas multinúcleo se ha hecho aún más potente. Entre sus objetivos están el reducir el tiempo de desarrollo de aplicaciones de todo tipo (no sólo en ámbitos de Pruebas, Control y Diseño) y el permitir la entrada a la informática a profesionales de cualquier otro campo. LabVIEW consigue combinarse con todo tipo de software y hardware, tanto del propio fabricante -tarjetas de adquisición de datos, PAC, Visión, instrumentos y otro Hardware- como de otros fabricantes.

Esta herramienta de software es la que elegí como soporte para el desarrollo de los programas de control del modem gsm/gprs.

En el anexo I – Laboratorios de LabVIEW se presentan unas capturas de pantalla en las que se aprecia el VI creado y utilizado para probar las rutinas de manejo del modem GSM/GPRS. [3]

5 CPU

5.1 Introducción

Como se explicó anteriormente, en el contexto del sistema de telemetría en su totalidad, llamamos CPU (o cerebro) al micro que se encarga de gestionar las diferentes comunicaciones que se desarrollan en el sistema. Una remota con el servidor y otra local que se desarrolla en segundo plano, por lo que denominamos bus posterior.

La comunicación remota consiste en manejar/controlar un modem GSM/GPRS embebido y la comunicación del bus local consiste en encuestar a los diversos periféricos que integran el sistema, como ser: medidores de energía, gps, módulos de entradas y salidas, valiéndose para efectuar dicha tarea del protocolo I2C.

Por último el CPU deberá tener la capacidad de funcionar como un servidor web embebido, teniendo la capacidad de cargar una página web desde una tarjeta SD, utilizando para esto una conexión Ethernet. Esta última funcionalidad es de mucha utilidad porque nos permite mostrar variables en tiempo real, a través de un simple explorador web, así como ingresar parámetros de configuración del equipo por el mismo medio.

5.2 Diseño de la placa

En el diseño de la placa de prototipo del CPU se tuvo como premisa contar con todos los componentes de hardware que sean necesarios para efectuar todas las tareas anteriormente descriptas, el resultado del diseño se ve plasmado en los planos N°1 - CPU Esquemático, N°2 - CPU Componentes y N°3 – CPU Board, en las siguientes imágenes podemos ver la placa del prototipo del CPU, fabricada:

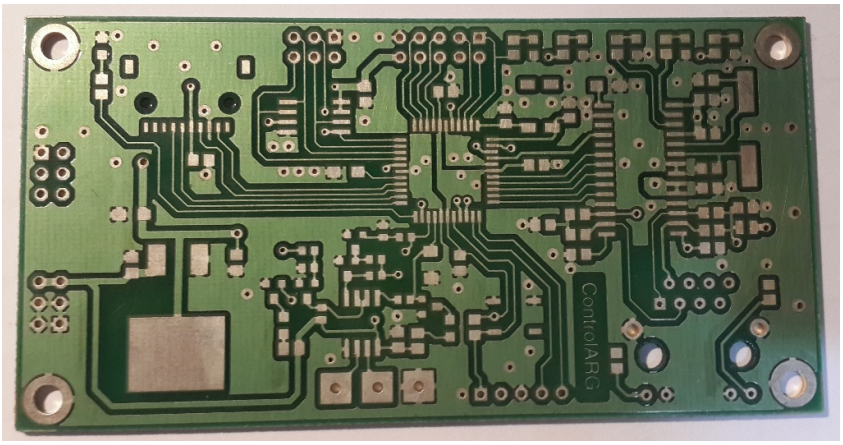


FIGURA 5.1 – Parte superior de la placa sin los componentes

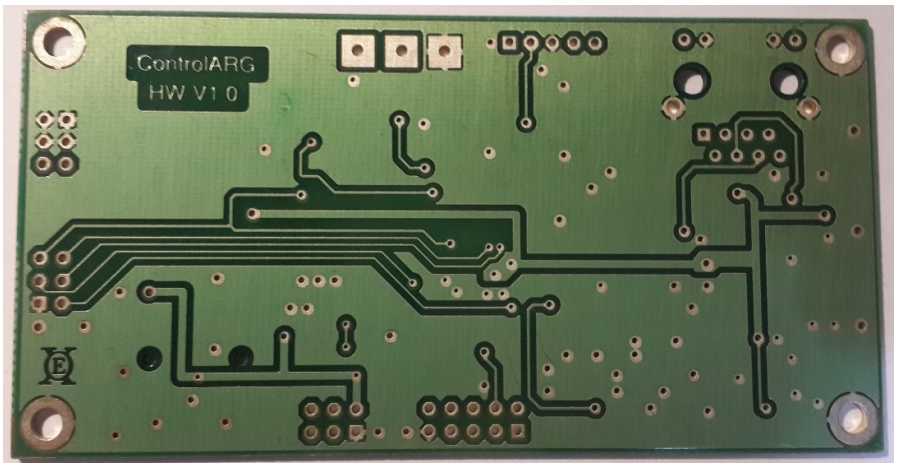


FIGURA 5.2 – Parte inferior de la placa

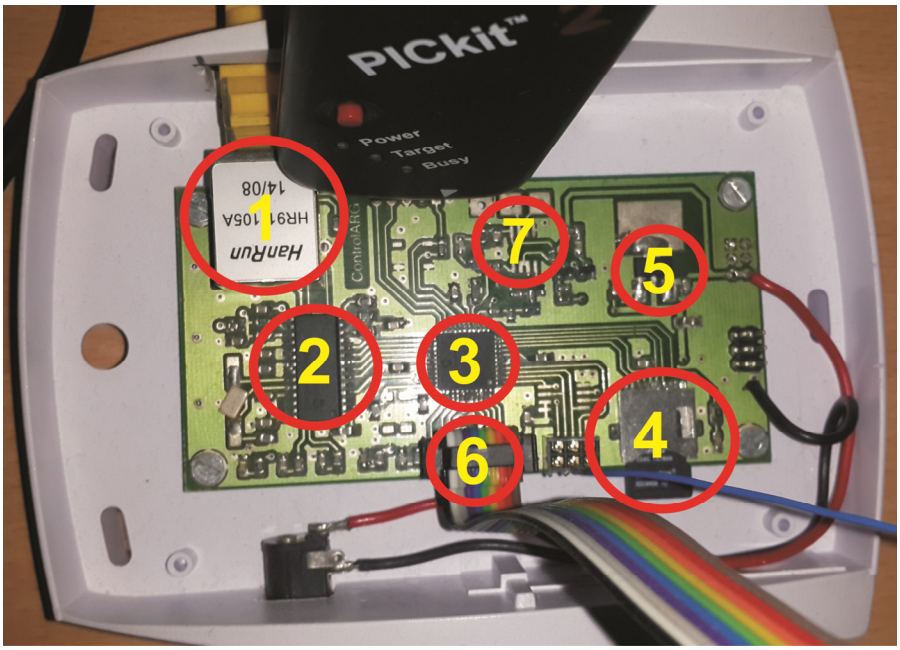


FIGURA 5.3 – Placa prototipo del CPU con los componentes montados

En la imagen anterior, podemos ver la placa con los componentes montados y una breve numeración de alguno de ellos, para entender un poco mejor su diseño y funcionamiento:

1. Conector Ethernet.
2. Controlador Ethernet.
3. Microcontrolador dsPIC33FJ128GP204.
4. Memoria micro SD.
5. Regulador de Tensión.
6. Conectores de comunicación I2C y puertos de entradas/salidas digitales.

7. Puerto de comunicación serie, para manejar el módulos GSM/GPRS o comunicarse con otro dispositivo.

6 MODEM GSM/GPRS

6.1 Introducción

El sistema utiliza un modem GSM/GPRS embebido para comunicarse inalámbricamente con el servidor al cuál se enlazan los datos funcionales y físicos que indican el estado de la subestación transformadora monitoreada.

Este módulo GSM/GPRS embebido permite establecer diversos canales de comunicación, a saber, llamadas de audio, mensajes de texto y comunicación TCP mediante GPRS. En el proyecto se hace uso de la conexión TCP mediante GPRS para enlazar los datos de telemetría al servidor de monitoreo. Y se utiliza el servicio de SMS para la indicación de alarmas, al/los responsable/s del mantenimiento de las instalaciones.

El módulo GSM/GPRS utilizado es de la firma Motorola, más precisamente el modelo G30L, ya que se contaba con una placa de desarrollo del mismo, lo cual permitió disminuir el tiempo de desarrollo del software de control dándonos la posibilidad de comenzar a plantear el programa de manejo e interpretación del módulo, sin detenernos o preocuparnos por el desarrollo del hardware funcional del mismo.

6.2 Modem GSM/GPRS embebido utilizado:

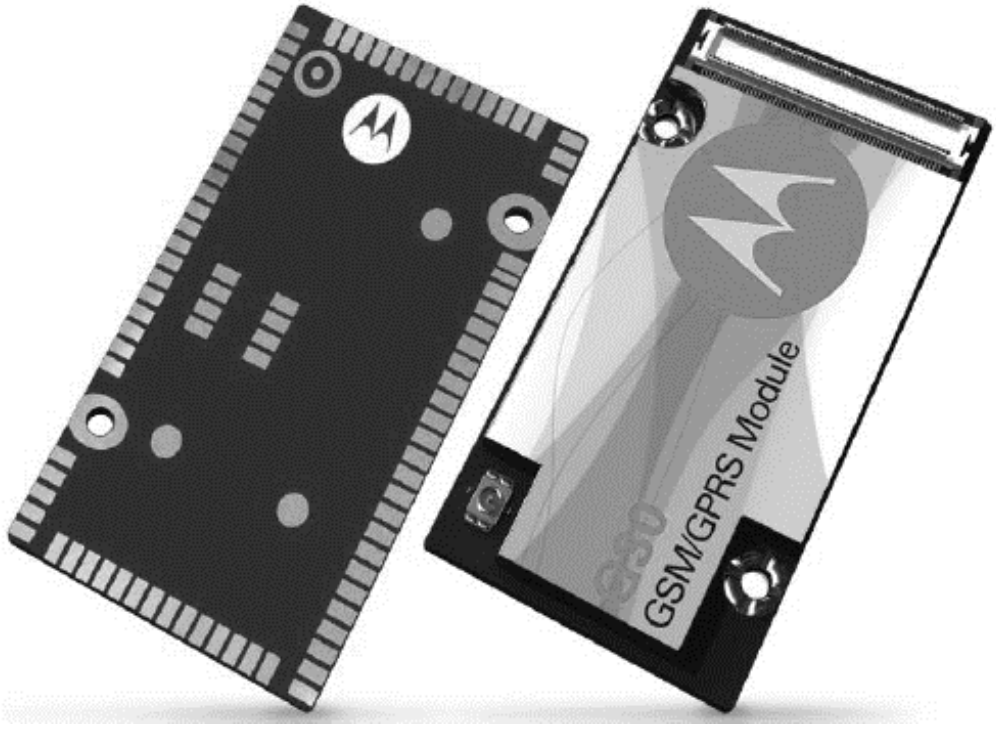


FIGURA 6.1 – Módulo GSM/GPRS utilizado

6.3 Por qué el Motorola G30?

[4] Además del obvio hecho de que Motorola es una empresa líder en el mercado de la telefonía, que nos brinda confianza y gran cantidad de soporte técnico de ayuda a la hora de desarrollar una aplicación en particular. Las ventajas que presenta este módulo embebido de comunicación GSM/GPRS son varias, a saber:

1. El módulo trabaja en las dos bandas GSM utilizadas en nuestro país (850 y 1900MHz) y presenta la capacidad de poder conmutar entre cualquiera de ellas de forma totalmente automática.
2. Por ser un módulo diseñado con concepto de uso automotriz, resulta robusto y compacto, lo cual lo hace ideal para aplicaciones, móviles.
3. El rango extendido de temperaturas de funcionamiento (-20 a +70 °C) del G30 permite que pueda ser utilizado en aplicaciones de gran exigencia térmica como es el caso de aplicaciones rurales.
4. Las distintas prestaciones incluidas en el módulo permiten aprovechar al máximo los servicios digitales de la red GSM con muy pocas exigencias de hardware y software externo.

6.4 Que alternativa de conexión debo utilizar

La decisión de que alternativa de comunicación utilizar en una aplicación, depende de diversos factores:

- En primer lugar dependerá de si se va a desarrollar una aplicación totalmente nueva o se va a integrar en una existente tratando de aprovechar lo que ya existe.
- Depende también de la cantidad de información que se debe cursar, a fin de evaluar la incidencia del costo de la comunicación frente al esfuerzo de desarrollo y ventajas que traiga aparejadas la implementación de la aplicación.
- Otro factor importante es si se trata de una aplicación puntual o deberá preverse un crecimiento a futuro de la misma.

Afortunadamente el Motorola G30 nos permite aprovechar la totalidad de las opciones que brinda la red GSM, por lo cual la decisión dependerá íntegramente de estos factores.

6.5 De qué servicios de comunicación disponemos?

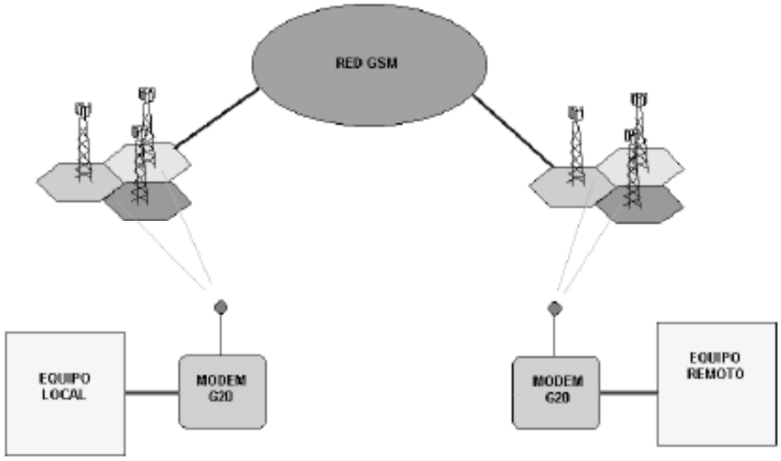
Existen tres servicios disponibles para establecer una comunicación digital en una red GSM, todos ellos soportados por el Módulo Motorola G30, estos son:

1. CSD (Circuit Switch Data)
2. SMS (Short Menssage Service)
3. GPRS (General Packet Radio Service)

A continuación se describen en detalle cada servicio y su alternativa de implementación.

6.5.1 CSD (Circuit Switch Data)

Este servicio nos permite establecer un enlace digital punto a punto entre dos Modems G30, dentro de una red GSM.



FUGURA 6.2 – Servicio de conexión CSD

Este servicio dedica un canal digital (similar al utilizado para una comunicación de voz) a la comunicación entre dos puntos. Por tal motivo, la prestadora va a facturar esta comunicación por minutos de aire que se utilicen como si fuera una llamada telefónica corriente.

Podríamos decir que esta solución es de muy fácil implementación, pero tiene como contrapartida el elevado costo de comunicación. Por tal motivo este método solo es recomendable en aplicaciones donde no se requiere una conexión On-Line, el costo de comunicación no es crítico y la cantidad de información mensual es baja.

La implementación de esta solución es simple, solo requiere de dos Modems G30, conectando uno al equipo (por un puerto RS232) y otro al equipo remoto (también por un puerto RS232).

En otras palabras, funciona como una comunicación de módem a módem por línea fija y se logra una velocidad o tasa de transferencia de datos de 9600 baudios (bit/seg) bidireccional.

6.5.2 SMS (Short Menssage Service)

Esta opción requiere o bien un módem G30 en ambos lados, con lo cual se puede enviar mensajes de texto, o bien uno en el lado remoto y una PC con protocolo SMPP en el lado local.

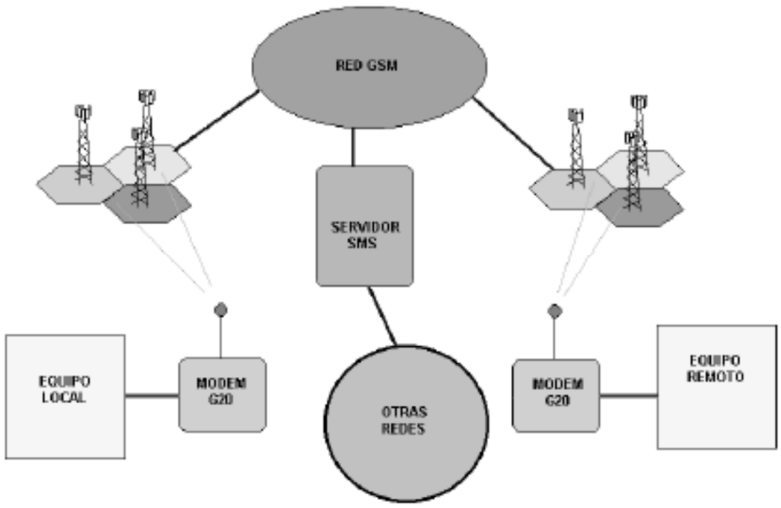


FIGURA 6.3 – Servicio de conexión SMS

Los mensajes de texto están limitados a 160 bytes por mensaje y se cobran por unidad, tenga 10 bytes o 160.

Esta solución tiene varios inconvenientes:

a) No hay garantía del tiempo que tarda en llegar un mensaje, puede durar segundos u horas, esto se debe a que en la comunicación interviene un servidor de SMS que encola los mensajes a medida que van llegando y los transmite a medida que tiene disponibilidad para hacerlo.
b) En el caso de transmitir paquetes mayores a 160 bytes, se deben cursar en diferentes mensajes y esto resulta antieconómico.
c) Por tal motivo, es recomendable optar por este servicio solo en el caso de requerir una comunicación donde la cantidad de información es baja y los tiempos de respuesta no son críticos.

6.5.3 GPRS (General Packet Radio Service)

Este es seguramente el servicio más importante de la red GSM para la transmisión de datos.

Con este servicio, la prestadora nos permite tener acceso a Internet utilizando el protocolo IP, ya sea con TCP o UDP.

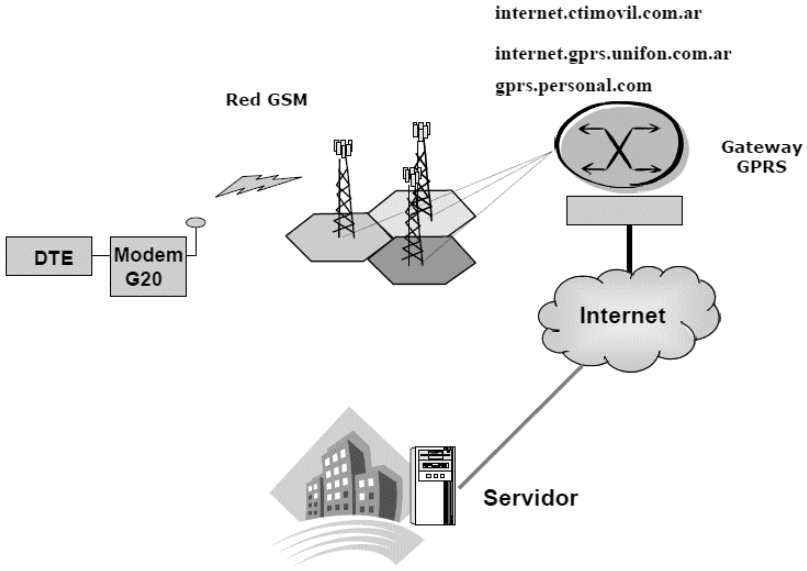


FIGURA 6.4 – Servicio de conexión GPRS

En este esquema se muestra el concepto de funcionamiento del servicio GPRS, la red dispone de un GATEWAY que nos da acceso a Internet, de esta forma el equipo remoto (DTE) puede abrir una conexión en forma directa con un servidor conectado a Internet con IP fija y transmitir la información.

La gran ventaja de este servicio es que está orientado a conexiones del tipo On-Line, por tal motivo *la prestadora facturará la cantidad de información cursada independientemente del tiempo de comunicación.*

El bajo costo de comunicación comparativo con los otros servicios descriptos, hacen del GPRS el servicio ideal para implementaciones que requieran o bien de transmisiones frecuentes o de gran cantidad de información a transmitir.

6.6 Uso del Stack Interno TCP/IP

A continuación se describe el uso del stack interno TCP/IP del módulo G30, a fin de denotar las principales características de una conexión IP inalámbrica.

6.6.1 Concepto de Funcionamiento

A modo de explicar conceptualmente el funcionamiento de esta conexión, nos remitimos a la imagen anterior, FIGURA 6.4, y procedemos por definir cada uno de los elementos que integran el sistema:

6.6.1.1 El equipo remoto

En este caso el DTE es el equipo que utilizará el servicio GPRS para transmitir datos hacia el servidor, utilizando para esto el modem G30. A continuación se explicara específicamente el procedimiento que debe llevar a cabo el software del mismo, para controlar el modem y ejecutar su cometido.

6.6.1.2 La red

Como se observa en la FIGURA 6.4, el servicio GPRS requiere que el conjunto DTE + Modem G30 se encuentren dentro del área de cobertura de la red GSM y registrado en la misma.

6.6.1.3 El Gateway

El prestador de servicio GSM, pone a disposición de los usuarios un Gateway GPRS de acceso público a los abonados de su red que cumple las siguientes funciones:

- Asignar a los usuarios que inicien sesión en el mismo una IP privada en forma dinámica.
- Permitir enviar y recibir paquetes IP hacia Internet.

El gateway dispone de un nombre propio llamado APN, se muestra en la FIGURA 6.4, por ejemplo, el APN de CTI, Personal y Unifón.

El inicio de sesión en el gateway es validado por un nombre de usuario y password definido por las prestadoras. Una vez iniciada la sesión, el gateway otorgará en forma dinámica una dirección IP privada dentro de la red GPRS y asociará una IP pública de Internet.

Este es un concepto importante, ya que la IP que obtendremos al iniciar la sesión no podrá ser vista desde Internet, cuando enviemos un paquete hacia la red lo haremos con una IP pública que se habrá asignado en esa sesión a nuestro equipo.

En otras palabras, el Servidor no sabrá nuestra IP hasta tanto no enviemos un paquete IP al mismo y nos identifiquemos.

6.6.1.4 El servidor

El Servidor debe poseer una IP fija de Internet, este tendrá un puerto TCP/IP en modo escucha (LISTEN) el cual recibirá los paquetes IP desde Internet. A fin de confirmar la recepción, es conveniente prever que el servidor confirme la llegada correcta de un paquete enviando un paquete hacia el DTE.

6.6.1.5 La Conexión

El procedimiento que utilizaremos para realizar la conexión es el siguiente:

a) Iniciaremos una sesión en el gateway para obtener una dirección de IP.

b) Abriremos el puerto del Servidor y enviaremos la información.
c) Esperaremos a que el Servidor confirme la recepción.
d) Cerramos el puerto
e) Queda a nuestro criterio si cerramos o no la sesión, *normalmente podemos mantenerla abierta ya que esta tarea no genera tráfico a ser facturado por la operadora.*

6.7 Programas de Control del Modem G30

6.7.1 Conceptos generales

El microcontrolador se comunica con el módulo G30 mediante un puerto serie, por este medio el PIC envía comandos AT al modem los cuales sirven para controlarlo. Los comandos AT son el lenguaje que el modem entiende, el prefijo AT, proviene de atención en inglés (AT = atention). Todos los comandos empiezan con AT + XXX, donde XXX es un prefijo determinado que corresponde a un comando que nos permite ejecutar tareas específicas sobre el modem, como ser: configuraciones, lectura de configuraciones o ejecutar acciones de comunicación, como efectuar una llamada, enviar un mensaje o establecer una sesión en el gateway. Para conocer en mayor detalle los comandos AT, dirigirse al Anexo II y/o consultar el manual completo de Motorola en el DVD anexo.

A cada comando AT enviado por el PIC le corresponde una respuesta específica por parte del modem, por ende la tarea del programa que se debe desarrollar para que el microcontrolador maneje el modem debe ser capaza de enviar los comandos AT específicos de cada tarea que se desea realizar, como ser, registrarse en la red, configurar los parámetros de la configuración serial, enviar y recibir mensajes, iniciar una sesión en el gateway para establecer una conexión TCP/IP inalámbrica y ser capaz de enviar y recibir paquetes de datos por este último canal de comunicación. Y una vez enviado el comando AT deberá ser capaz de interpretar la respuesta recibida desde el modem para saber si el comando se ejecutó con éxito o no.

A continuación se presentan diagramas de flujo que en los cuales se detallan los procedimientos a seguir y los comandos AT utilizados para lograr cada tarea.

En el Anexo I - Laboratorios en LabVIEW se presenta capturas de pantallas de las pruebas de laboratorio realizadas sobre cada rutina planteada, las cuales fueron efectuadas haciendo uso de un programa o VI creado en la plataforma de programación gráfica LabVIEW.

6.7.2 Inicialización del Modem

Para llevar a cabo la tarea de inicializar el modem G30, planteamos una función de inicialización que consiste básicamente en una máquina de estado en la cual se van enviando los comandos que permiten inicializar el modem y luego se analizan las respuestas recibidas desde el mismo, para ver cómo está ejecutándose el proceso.

En el diagrama de flujo presentado a continuación, Diagrama de flujo N°1, podemos ver la representación esquemática de la función de inicialización, se procede seguidamente a explicar en breve en que consiste cada uno de los pasos que se siguen:

NOTA 1: Una vez energizado el modem ya se encuentra en condiciones de comenzar a operar, para corroborar que el mismo se encuentra funcionando, en primera instancia y para dar inicio al proceso le enviamos el comando AT, al cual el modem debe responder con OK si es que está operativo. *En caso de no recibir respuesta o recibir una respuesta de error, se procede a reintentar el comando. Esta tarea iterativa se realiza en todos los pasos.*

NOTA 2: Se envía el comando ATE0, para desactivar el eco de la comunicación.

NOTA3: Podemos ver en este paso, que en un solo envío se concatenan varios comandos AT, esto es permitido en la sintaxis del protocolo AT, con la premisa de usar el carácter (,) para separar los comandos entre sí. En este conjunto de comando, encontramos: AT+CBAUD=19200, con este comando configuramos la velocidad de transferencia de datos en la comunicación serial del PIC con el modem, &KO, con este comando desactivamos el control de flujo en dicha comunicación, +CMGF=1, configuramos el tipo de SMS en texto, +CSMS=128, sirve para seleccionar el servicio de mensaje y por último enviamos el comando +CNMI=2,2 mediante el cual le decimos al modem que nos avise en caso de tener un SMS entrante, para de este modo leerlo, es decir configuramos una notificación por SMS entrante.

NOTA 4: Se envía el comando AT+CREG=1, para registrar el modem en la red. Luego se espera la respuesta y se analiza el contenido de la misma, este es un comando que presenta varias alternativas de respuesta en función al tipo de red al que se registre y/o en caso de que no se registre a ninguna. En el programa se contempla solo el ingreso o registro a la red, sin importar cuál sea esta.

NOTA 5: Por último, una vez registrado el modem en la red enviamos un simple comando AT para cerciorar que el modem se encuentre activo luego del procedimiento de inicialización.

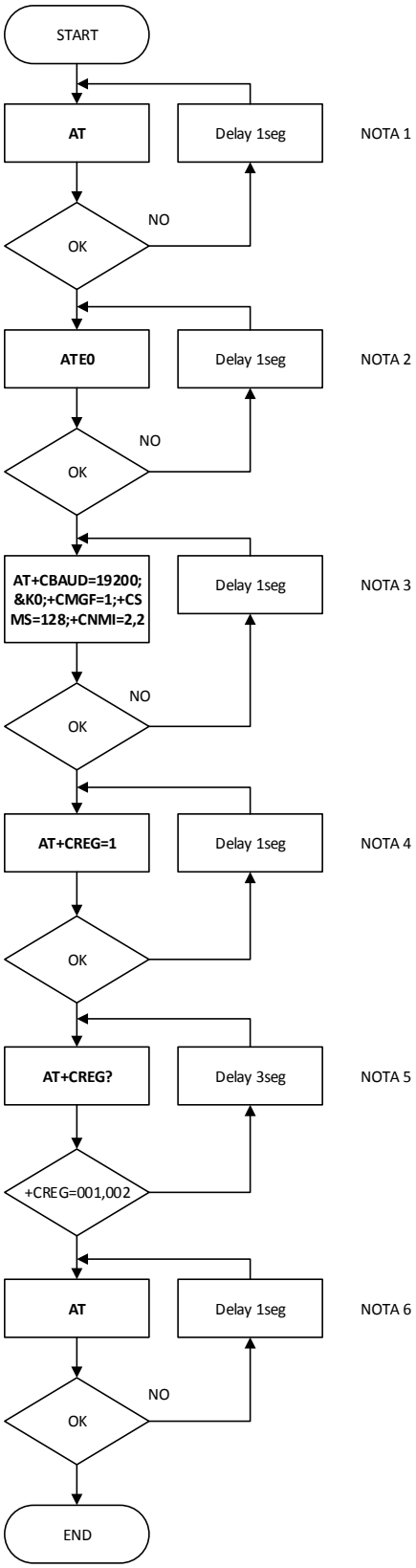


Diagrama de flujo N°1 – Inicialización del Modem

6.7.3 Envío de SMS

Para realizar el envío de un SMS se planteó una función, también siguiendo el esquema de una máquina de estado, que pasa por los estados descriptos en el diagrama de flujo N° 2, el cual se describe, para su mejor entendimiento en las siguientes notas.

Resulta evidente, pero no quiero dejar pasar por alto el hecho de que antes de proceder al intento de envío de un SMS el modem debe estar correctamente inicializado y debidamente registrado en la red, es decir que la función anteriormente descripta debió haberse ejecutado con éxito antes de intentar este procedimiento.

NOTA 1: Para iniciar el proceso de envío de un SMS, utilizamos el comando AT+CMGS, mediante el cual le digo al modem que quiero enviar un mensaje de texto y como argumento le paso el número del destinatario al cual deseo dirigirme, el otro parámetro que vemos (,145) es para indicar el formato en el que ingreso el numero aclarando si se hace uno no del código del país, por ejemplo en el caso de Argentina +54.

En caso de procesar correctamente el comando el modem responde con el carácter (>), indicándonos que está listo para efectuar el envío. En caso de no suceder esto, esperamos 1se y reintentamos la operación.

NOTA 2: Enviamos el texto del mensaje (hasta 160 caracteres) y la combinación de caracteres Ctrl+Z (Control + Z).

Luego de esto si recibimos una respuesta de OK por parte del modem, sabemos que el SMS se envió correctamente y damos por finalizado el proceso, en caso contrario reintentamos.

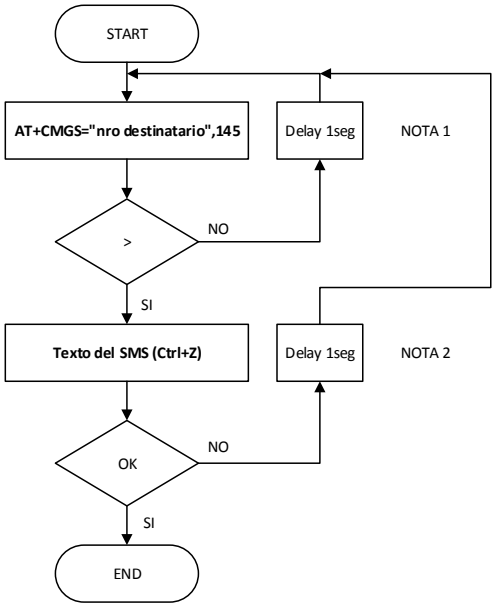


Diagrama de flujo N° 2 – Envío SMS

6.7.4 Recepción de SMS

El proceso de recepción de un SMS consiste en detectar el aviso de SMS entrante en el modem y luego separar el contenido del aviso del mensaje, entre el texto y el número del remitente. Una vez procesado el SMS entrante y extraída la información del mismo, enviamos al modem un aviso de recepción del mensaje, esto se hace para que el mismo repita el aviso en el próximo SMS entrante, de no hacerlo el modem dejará de informarnos cuando un mensaje llega. En el diagrama de flujo N°3, a continuación presentado, se ve gráficamente lo anteriormente explicado.

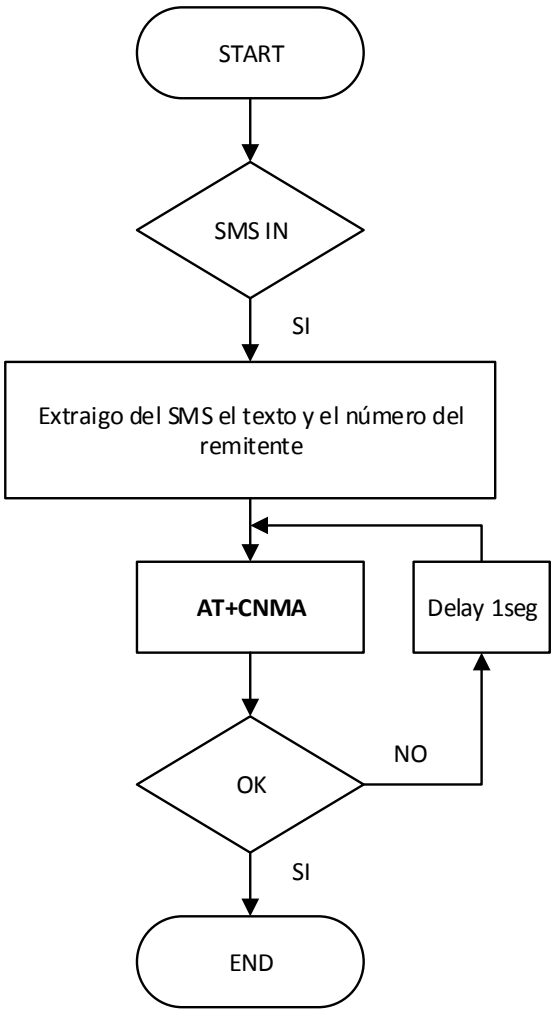


Diagrama de flujo N° 3 – Recepción SMS

6.7.5 Establecer una conexión TCP/IP

[5] Teniendo el modem inicializado y registrado en la red, procedemos por enviarle el comando AT+MIPCALL=1,"gprs.personal.com","gprs","gprs", para iniciar una sesión GPRS.

Lo cual vemos en el diagrama de flujo N° 4, Este comando crea una conexión remota de protocolo punto a punto, PPP (Point to Point Protocolo). Los parámetros que la misma requiere son: en primer lugar definir la operación a realizar 1 para establecer la sesión GPRS, y luego debemos definir el proveedor del servicio, un nombre de usuario y una contraseña (en el caso ejemplificado estos parámetros son,"gprs.personal.com","gprs","gprs" y están provistos por la empresa prestadora del servicio de telefonía).

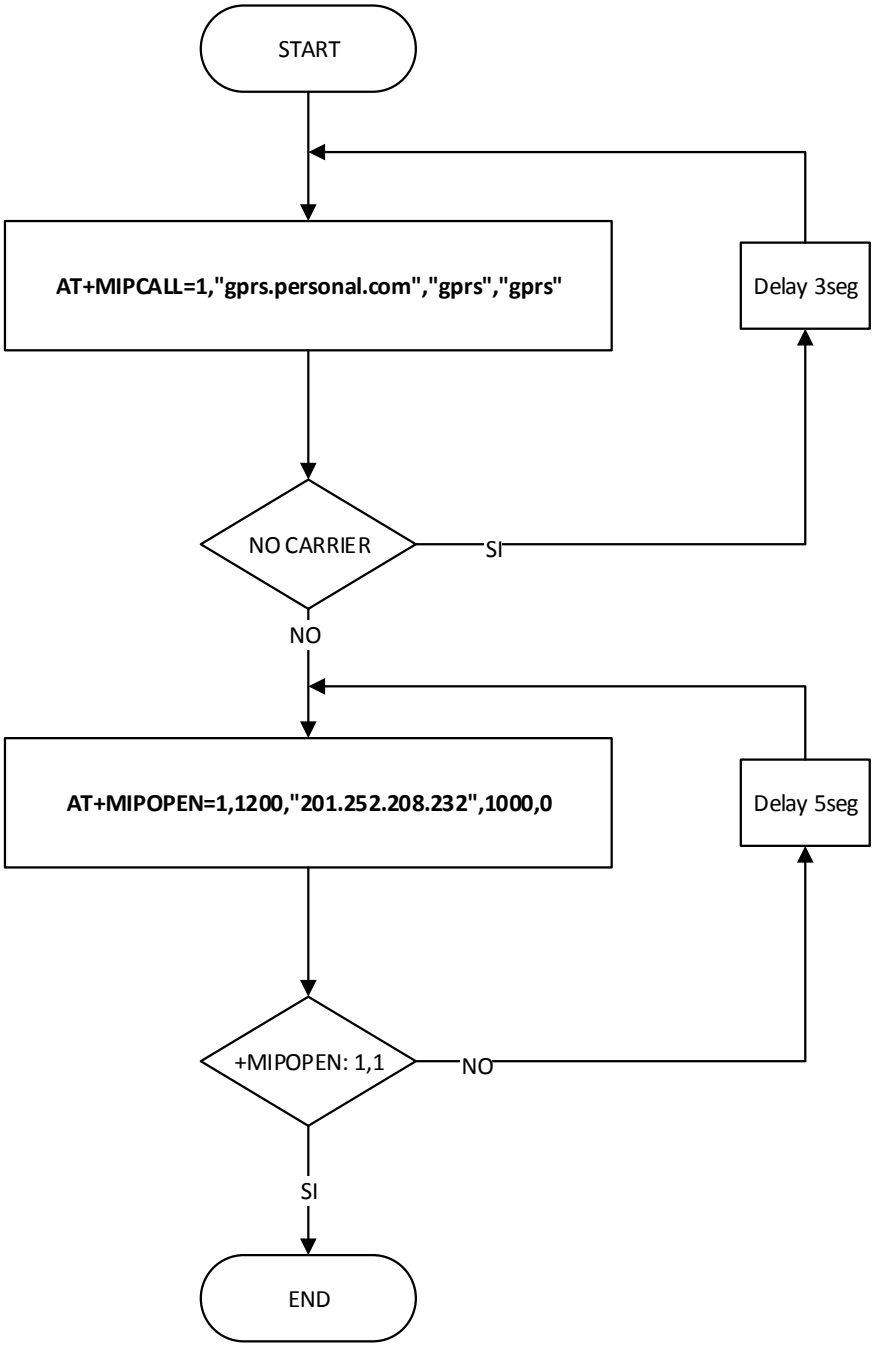


Diagrama de Flujo N° 4 – Establecimiento de una conexión TCP/IP

Debemos tener en cuenta que este comando no es cancelable, por ende una vez enviado, se debe necesariamente esperar una respuesta y no se podrá enviar otro comando hasta que esto suceda.

Como respuesta por parte del modem podremos tener un inicio de sesión exitoso, en cuyo caso ya se nos habrá asignado por parte de la red una dirección de IP o un intento fallido de inicio de sesión identificado con un NO CARRIER como respuesta por parte del modem. En caso de suceder esto último se esperará 3 segundos antes de reintentar.

Iniciada con éxito la sesión GPRS, se procede por abrir un puerto en el servidor para lo cual nos valemos del comando AT+MIPOPEN=1,1200,"201.252.208.232",1000,0 cuyos argumentos son: identificador del socket del modem que se está abriendo (1 en el ejemplo), puerto del modem que se utiliza para la conexión (1200), dirección IP del servidor remoto con el que quiero conectarme (201.252.208.232), puerto del servidor remoto que se quiere abrir (1000) y por último se define el protocolo a utilizar TCP o UDP (0=TCP).

Enviado este comando, si recibimos como respuesta por parte del modem +MIPOPEN: 1,1, quiere decir que el puerto se abrió con éxito y ya está establecida la conexión entre el modem y el servidor.

6.7.6 Enviar paquetes de datos por TCP/IP

Iniciada la sesión GPRS y abierto el puerto en el servidor remoto, estamos en condiciones de enviar paquetes de datos al mismo y para lograrlo procedemos como se detalla en el diagrama de flujo N° 5, en primer lugar enviando el comando AT+MIPSEND=1,Data Packet, el cual requiere como argumentos, en primer lugar identificar el socket a utilizar (socket 1 en el ejemplo) y luego el paquete de datos/información que se desea transmitir, el cual puede ser como máximo de hasta 504 caracteres. Si se quiere enviar un paquete de información más grande debe hacerse en sucesivos envíos parciales. Básicamente este comando carga el buffer de trasmisión del modem, dejando el paquete de información listo para su posterior envío.

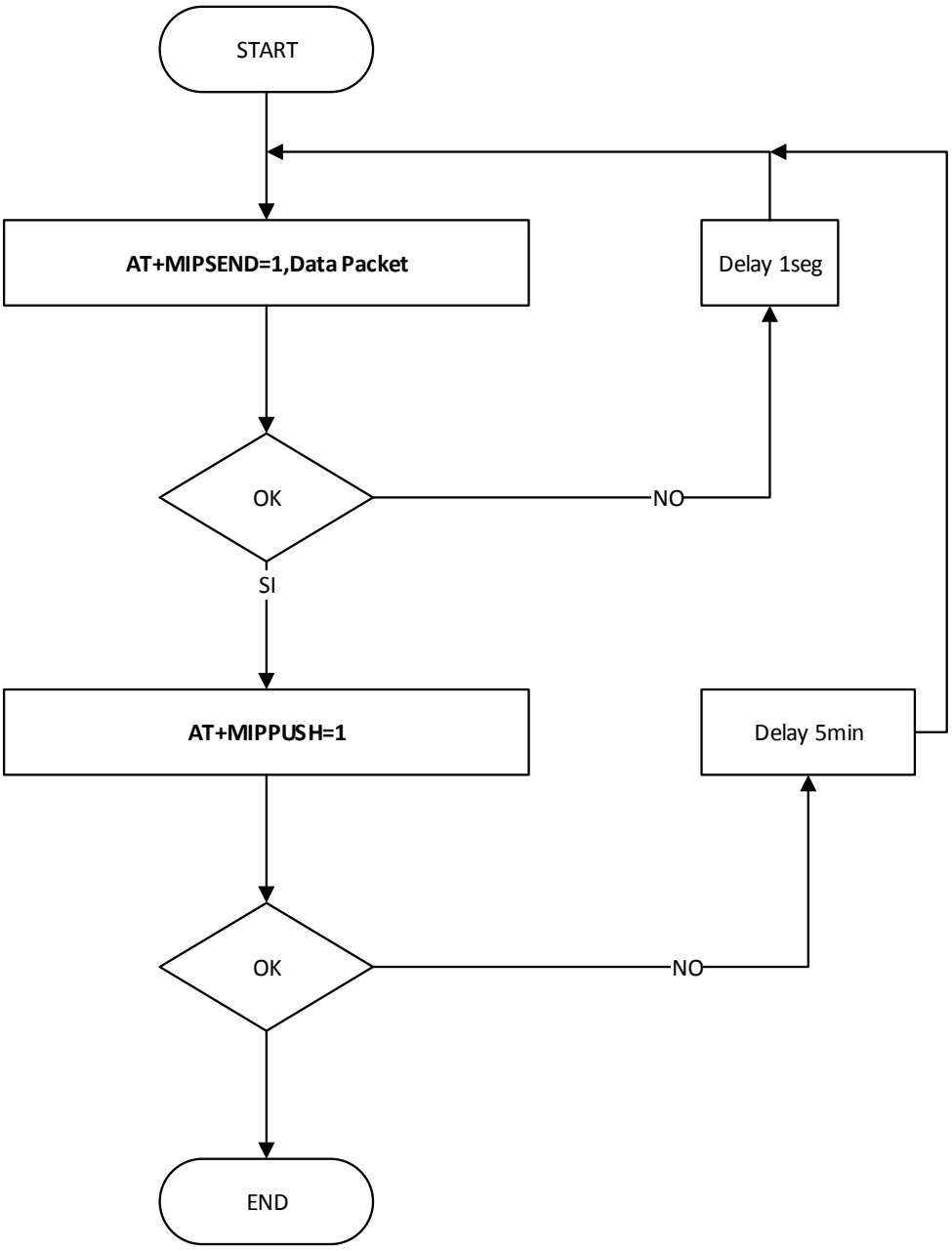


Diagrama de flujo N°5 – Envío de datos por TCP/IP

Si recibimos por respuesta OK, quiere decir que el modem proceso correctamente el comando y podemos proseguir, en caso contrario se reintentara luego de 1seg.

A continuación, para efectivizar el envío del paquete de información enviamos el comando AT+MIPPUSH=1, con el cual enviamos el buffer de transmisión, anteriormente cargado hacia la red.

Toda la información enviada al buffer de transmisión del modem G30 debe ser codificada en ASCII hexadecimal, es decir si queremos enviar una “A” (ASCII hexadecimal

41) debemos enviar dos caracteres “41”. El mismo criterio se utiliza para decodificar los datos recibidos del buffer.

Si recibimos un OK como respuesta por parte del modem sabemos que la información se envió con éxito y damos por finalizado el proceso, en caso de recibir un aviso de error reintentamos el proceso luego de 5 minutos.

6.7.7 Recibir paquetes de datos por TCP/IP

El proceso de recibir paquetes de datos consiste básicamente en identificar el aviso que el modem nos entrega cuando a arribado información por medio de la conexión GPRS y se encuentra el buffer cargado y listo para ser leído. El modem genera este aviso, mediante el envío del siguiente mensaje al PIC, +MIPRTCP: 1,0,Data Packet, en donde nos indica que el buffer de recepción recibió un paquete de datos, identificando el número de socket (1 en el ejemplo).

Lo que se debe hacer luego es guardar la información recibida, para su posterior procesamiento.

7 MÓDULO DE MEDICIÓN DE ENERGÍA

7.1 Introducción

El desarrollo del módulo de medición de energía, se basó en un utilizar una plataforma ya desarrollada por la empresa ControlARG, quien tenía desarrollado un multi-medidor de parámetros eléctricos, como ser, tensiones eficaces, corrientes eficaces y energías, basado en el IC (Circuito Integrado) ADE7758 de la marca Analog Devices (Ver Anexo II – Manuales y Catálogos).

La mejor opción para medidores electrónicos es usar DSPs (Procesadores Digitales de Señales) los cuales permiten digitalizar las ondas de tensión y corriente por medio de conversores analógico-digital para luego realizar cálculos. El procesamiento de señales digitalizadas permite el cálculo estable y exacto.

Hay dos clases de DSP, los programables y los de función fija. Para esta aplicación en particular se utilizó uno de función fija debido a su alta precisión, bajo costo, y la ventaja de requerir un menor trabajo de programación. Los de aplicación fija resultan, menos versátiles, pero de una mayor facilidad de aplicación para tareas específicas.

El IC (Circuito Integrado) ADE7758 es básicamente un DSP de función fija. Con capacidad de digitalizar una señal analógica a una tasa de muestreo de 26kSPS (kilo muestras por segundo), lo cual le permite alcanzar grandes niveles de precisión en las mediciones.

7.2 Descripción General del CI ADE7758

FIGURA 7.1 – IC ADE7758

El ADE7758 es un circuito integrado de alta precisión que permite realizar mediciones trifásicas de energía y cuenta además con interface serie SPI y dos salidas de pulsos. Es adecuado para para medir la energía activa, reactiva y aparente, para diferentes configuraciones trifásicas, tanto en delta como en estrella, ambas con tres o cuatro conductores.

Provee un sistema de calibración de potencia, offset del valor eficaz y calibración de fase. Posee también un registro de forma de onda, que permite acceder a las muestras de las

salidas de los ADCs e incorpora un circuito que detecta variaciones de alta y baja tensión, de corta duración, en donde los umbrales de tensión, así como la duración de la variación (número de medios ciclos de línea) son programables por el usuario.

El ADE7758 se comunica al exterior por medio de la interfaz SPI (interfaz serie de periféricos), la salida $\overline{IRQ}$ pasa a nivel bajo cuando uno o más eventos de la interrupción han ocurrido y el registro STATUS del IC indica la naturaleza de la interrupción.

7.3 Programa del PIC para leer el ADE7758

Como se explicó anteriormente el ADE7758 cuenta con una salida $\overline{IRQ}$ ("patita"), que se pone en nivel bajo de tensión para notificar el acontecimiento de un evento o interrupción, a partir de este concepto se plantea el programa que nos permitirá acceder a los registros del ADE7758, mediante el microcontrolador PIC y a través de la interfaz serie de comunicación de periféricos con la cual cuenta tanto el microcontrolador, como el IC ADE7758.

Conectando la salida $\overline{IRQ}$ a una entrada del micro capaz de generar una interrupción externa en el mismo por cambio de estado en la entrada. Se plantea un programa que quede a la espera de dicha interrupción y una vez ocurrida la misma, el procedimiento consiste en leer el registro STATUS del ADE7758 para saber que evento a acontecido y en función al evento que se haya dado se procede a la lectura de determinados registros.

Por ejemplo supongamos que configuramos el ADE7758 para que genere una interrupción cuando el registro de energía activa de la fase A esta lleno hasta la mita. Entonces al ocurrir este evento, se generará una interrupción en el ADE7758 que pondrá en bajo la salida $\overline{IRQ}$, generando de este modo una interrupción en el micro por lo cual este procederá a leer el registro STATUS y al enterarse que el motivo de la interrupción deberá proceder por leer el registro de acumulación de energía activa de la fase A.

7.4 Diseños de Circuitos Impresos del ADE7758

En lo que a diseño de circuitos impreso respecta, para la obtención de un prototipo funcional del módulo de medición de energía, se plantearon dos placas, una placa en la que va montada el ADE7758 con sus respectivas entradas, señales de tensión y corriente. Y por otra parte una placa en la que contiene al microcontrolador que maneja e interpreta al medidor de energía.

Ambas placas estaban ya diseñadas, en gran parte, por la empresa ControlARG, el trabajo que se realizó fue el agregado de los elementos necesarios para poder conectar el micro que controla e interpreta el medidor de energía con el Módulo CPU del sistema de telemetría

remoto de subestaciones transformadoras. Siendo dicha comunicación llevada a cabo a través del bus posterior e implementando el protocolo de comunicación I2C.

7.4.1 Circuito Impreso del ADE7758

El mismo se diseñó teniendo en cuenta las Notas de Aplicación del fabricante y el resultado obtenido se ve en los planos anexos N°4 – ADE 7758 Esquemático, N°5 – ADE 7758 Componentes, N°6 - ADE 7758 Board, y en las siguientes imágenes:

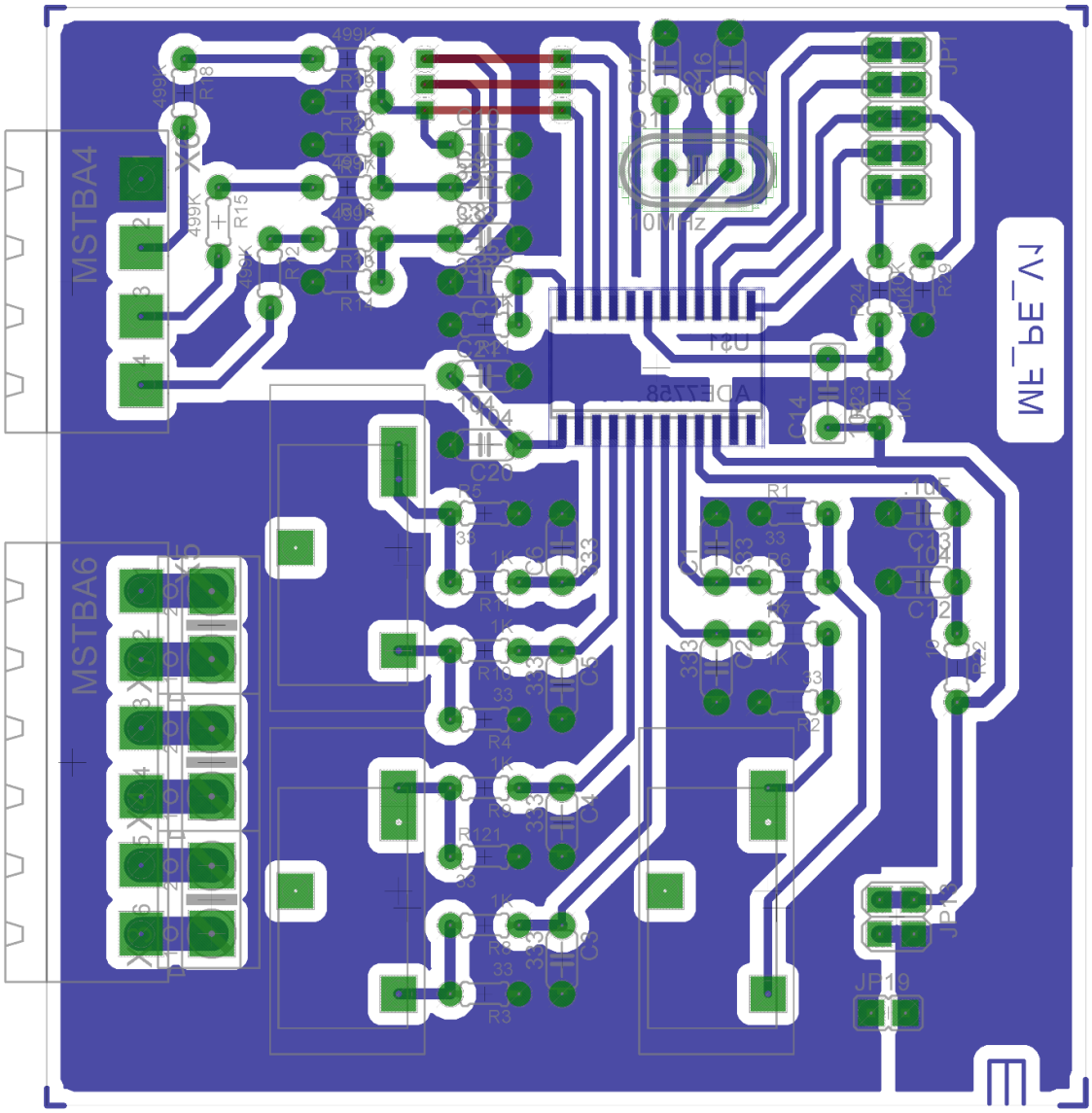


FIGURA 7.2 – Placa ADE7758 Imagen esquemática, generada en Eagle 6.2

Las imágenes a continuación presentadas, FIGURAS 7.3 y 7.4 son fotografías de las placas fabricadas, para obtener el prototipo funcional del módulo de medición de energía.

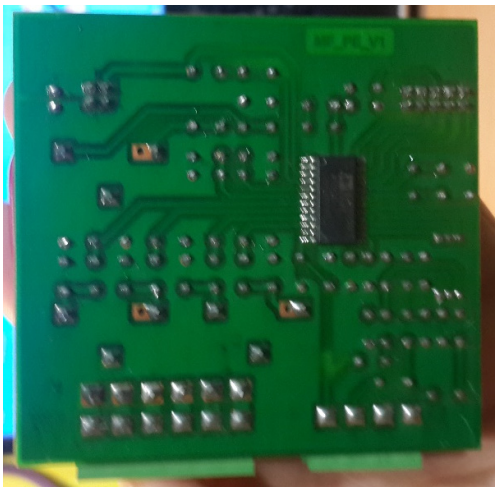


FIGURA 7.3 – Placa ADE 7758 fabricada a partir del diseño realizado, vista inferior.

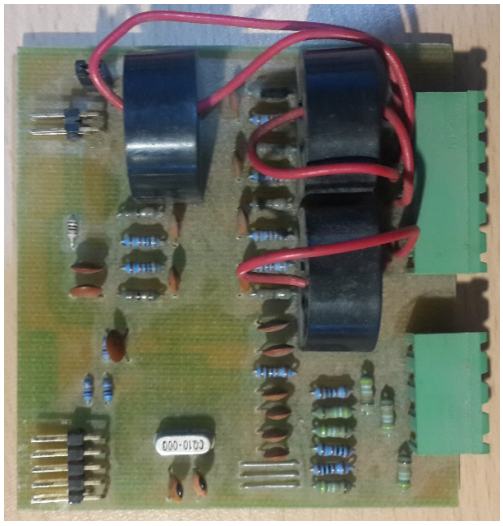


FIGURA 7.4 – Placa ADE 7758 fabricada a partir del diseño realizado, vista superior.

7.4.2 Circuito Impreso del PIC que interpreta el ADE7758

El diseño del mismo se desarrolló teniendo en cuenta las comunicaciones que se deben manejar, tanto con el IC de medición de energía, así como con el CPU del sistema de telemetría, el resultado obtenido se refleja en los planos anexos N°7 – Lector ADE 7758 Esquemático, N°8 – Lector ADE 7758 Componentes y N°9 – Lector ADE 7758 Board, y en la siguiente imagen ilustrativa, FIGURA 7.5, y en las fotografías tomadas de la placa real, fabricada para la obtención del prototipo funcional, FIGURAS 7.6 y 7.7.

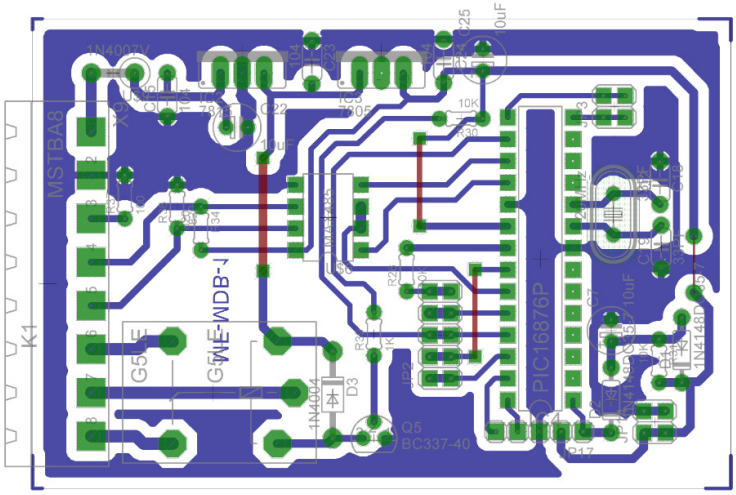


FIGURA 7.5 – Placa Lector ADE7758 captura de pantalla Eagle 6.2.

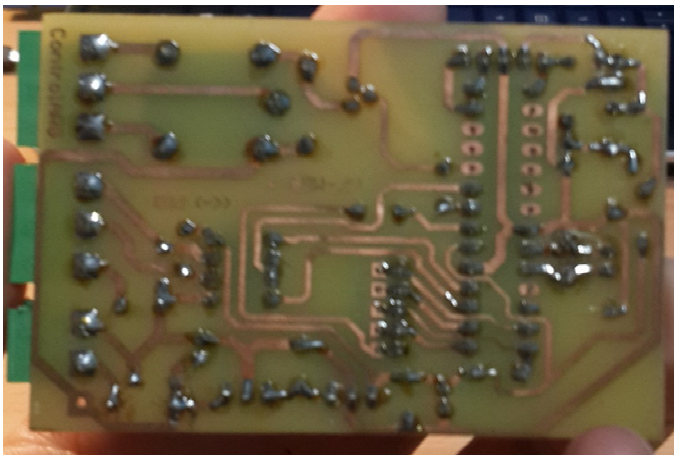


FIGURA 7.6 – Placa Lector ADE7758 fabricada a partir del diseño realizado, vista inferior.

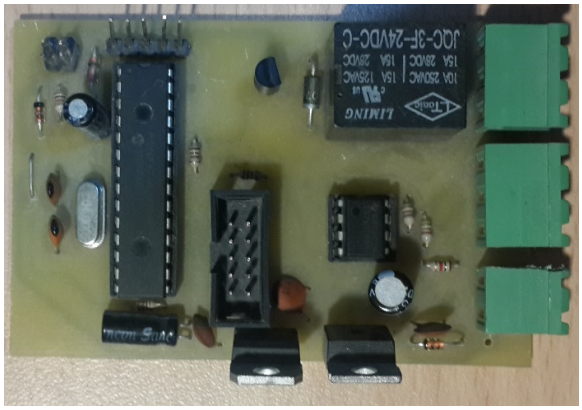


FIGURA 7.7 – Placa Lector ADE7758 fabricada a partir del diseño realizado, vista superior.

En la siguiente imagen, FIGURA 7.8, se muestra el modo en que se conectan ambas placas de prototipado, para poder comunicarlas entre sí, por medio de un cable plano (flat).

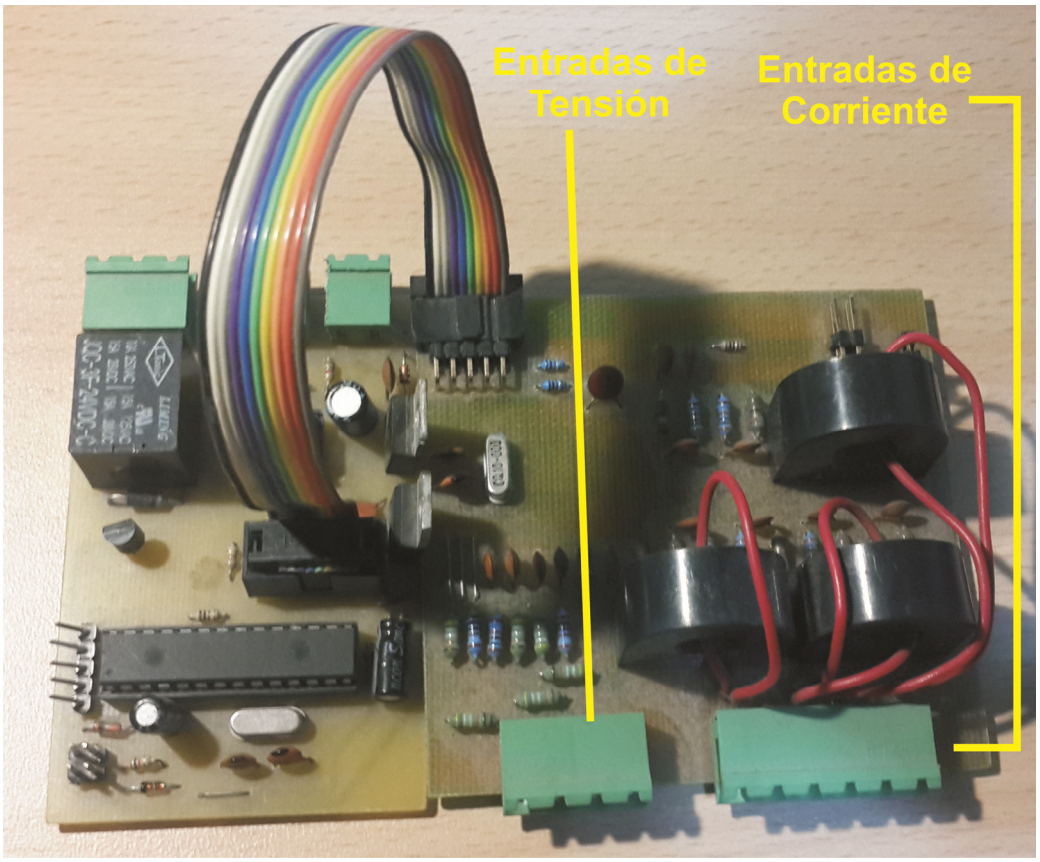


FIGURA 7.8 – Conexión entre ambas placas

7.5 Conexionado del módulo de medición a los TIs y TVs para efectuar las mediciones

Como se describió en primera instancia para efectuar las mediciones de los parámetros eléctricos nos valemos de las múltiples capacidades del circuito integrado medidor de energía ADE7758, a nivel físico o de conexionado este integrado se vale de 7 señales, para efectuar internamente los cálculos de energía. Estás señales provienen de los transformadores de corriente TIs (3 señales, una para cada fase) y de los transformadores de tensión TVs (4 señales, 3 tensiones de fase y la tensión del neutro). En el plano N°16 – Conexión del Módulo de Medición de Energía, se presenta un diagrama general de conexionado entre el módulo, los TIs y TVs.

8 MODULO GPS

8.1 Introducción

El proceso de desarrollo del módulo GPS consistió en la selección de un GPS, desarrollo de un programa para interpretar la trama entregada por el GPS mediante un PIC16F883 y por último diseño de una placa de circuito impreso para obtener un prototipo, pensando en la modularidad del sistema de telemetría en general y que cuente con comunicación UART para interpretar el GPS y comunicación I2C para comunicarse con el CPU.

8.2 GPS Utilizado

Para el desarrollo de este módulo hacemos uso de un GPS de la firma Quectel, más precisamente el modelo Quectel L20.

A continuación vemos una imagen del GPS utilizado:

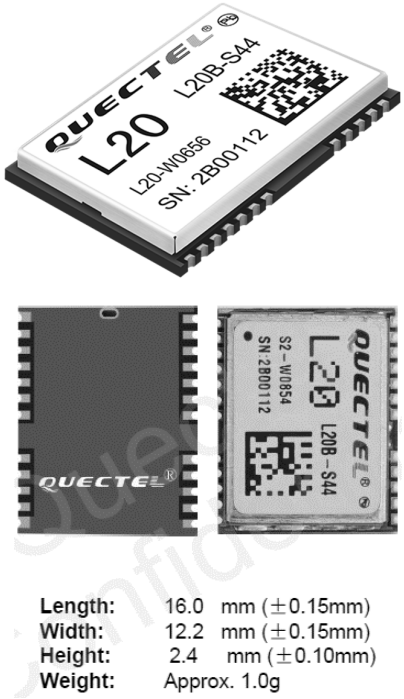


FIGURA 8.1 – Imagen del módulo GPS utilizado

Cuyas características principales son:

- Standard NMEA protocol: NMEA 0183 Standard V3.01.
- Bajo consumo de energía 39mA.
- Alta sensibilidad.
- Rápida sincronización en arranque en frío, de 3 a 9 seg.
- Tamaño compacto 16mm x 12.2mm x 2.4mm.

- Comunicación UART.
- Baud Rate por defecto 4800 bps

El motivo de elección de este GPS fueron las buenas prestaciones que el mismo presenta, anteriormente mencionadas, así como su bajo costo, la gran cantidad de documentación que el fabricante aporta como herramientas de desarrollo y el hecho de contar con una placa de desarrollo que nos permitió pasar al desarrollo del software de interpretación sin perder tiempo en diseñar y fabricar una placa de desarrollo.

A continuación se presenta una imagen de la placa de desarrollo utilizada:

FIGURA 8.2 – Fotografía de la placa de desarrollo empleada

8.3 Funcionamiento del GPS

8.3.1 Estructura del mensaje NMEA

El GPS una vez posicionado, lo cual se consigue entre 3 y 9seg después de su encendido, nos entregará, cada 1seg un paquete de información compuesto por un conjunto de tramas definidas en su estructura, sintaxis y contenido por el protocolo standard NMEA 0183. Este paquete de información recibe el nombre de mensaje NMEA.

En las especificaciones de dicho protocolo (Anexo II - Quectel L20 Quectel GPS Engine, GPS Protocol Specification) podremos ver que todas y cada una de las tramas contenidas en el paquete de información se caracterizan, por comenzar con los caracteres "$GP" y luego tenemos tres caracteres o letras que identifican el tipo de trama (cada tipo de trama contiene un conjunto distinto de información, y por ende se aplican en distintos campos).

En nuestro caso, la trama que satisface nuestras necesidades, en cuanto a datos requeridos en función a la aplicación es la RMS, por ende la trama que estamos interesados en leer comenzara con los caracteres, "$GPRMS". Luego del encabezado de la trama, anteriormente descripto, tenemos el campo de datos, en donde cada dato está separado (,) y el cual varía en la cantidad de datos, en función de la trama que se trate. Para indicar el fin del campo de datos se utiliza el carácter "*" y por ultimo cada mensaje termina con los caracteres ASCII <CR><NL> (CR=Carry Return, NL=New Line). En el cuadro que a continuación se expone se plantea de un modo más gráfico lo anteriormente escrito:

Campo	Longitud (bytes)	Descripción
$	1	Cada mensaje NMEA comienza con "$"
ID transmisor	1~2	"GP" el mensaje recibido proviene de un GPS "P" para mensajes de protocolos propietario
Identificador de mensaje NMEA	3	Tipo de trama de datos
Campo de Datos	Depende del tipo de mensaje NMEA	El campo de datos está delimitado por ","
*	1	Carácter que indica la finalización del campo de datos
Checksum	2	Número hexadecimal, calculado haciendo un XOR de todos los caracteres entre "$" y "*"
<CR><LF>		

8.3.2 Estructura de la Trama RMC

Campo	Ejemplo	Descripción
$	1	Cada mensaje NMEA comienza con "$"
GPRMC		Identificador del mensaje
UTC time	083557.942	Hora mundial. Formato hhmmss.sss
Fix Status		
Fix Status	A	"V" = Datos no válidos, "A" = Datos válidos
Latitud	3109.8883	Latitud, formato ggmm.mmmm (grados y minutos)

Indicador de N/S	N	"N" = Norte , "S" = Sur
Longitud	12123.4479	Longitud, formato gggmm.mmmm (grados y minutos)
Indicado de E/O	E	"E" = Este, "W" = Oeste
SOG	0.35	Velocidad sobre la tierra en knots
COG	133.35	Curso sobre la tierra en grados
Date	250915	Fecha en formato DDMMYY (día, mes y año)
Modo de Posicionamiento	A	"N" = Data enviada no valida "A" = Autónomo "D" = DGPS "E" = DR "R" = Posición de curso
*	1	Carácter que indica la finalización del campo de datos
Checksum	2	Número hexadecimal, calculado haciendo un XOR de todos los caracteres entre "$" y "*"
<CR><LF>		

8.4 Desarrollo del software de interpretación del GPS.

Para interpretar los datos entregados por el GPS, planteamos un programa que manejando, mediante interrupción, la recepción de datos en la UART del PIC16F883 vaya analizando cada carácter recibido y en una posterior comparación con respecto a lo esperado en la trama NMEA (National Marine Electronics Association) que deseamos leer vayamos guardando los datos en variables o espacios de memorias asignados para su posterior utilización.

8.5 Diseño del circuito impreso para el prototipo del módulo GPS:

En el diseño de la placa de circuito impreso de este módulo se tuvo en cuenta las funcionalidades con las que el mismo debe contar, como se explicó el microcontrolador que gobierna este módulo debe ser capaz de comunicarse con el GPS, mediante una comunicación serie, valiéndose de la UART del PIC16F883 y a su vez debe comunicarse por el bus posterior y mediante el protocolo I2C con el CPU del sistema. Teniendo en cuenta estas funcionalidades y valiéndonos de las hojas de aplicación dadas por el fabricante, ver anexo II – Quectel

Hardware Specification, el resultado obtenido puede verse en los planos N°10 – GPS Esquemático, N°11 – GPS Componentes y N°12 – GPS Board, y en la siguiente imagen:

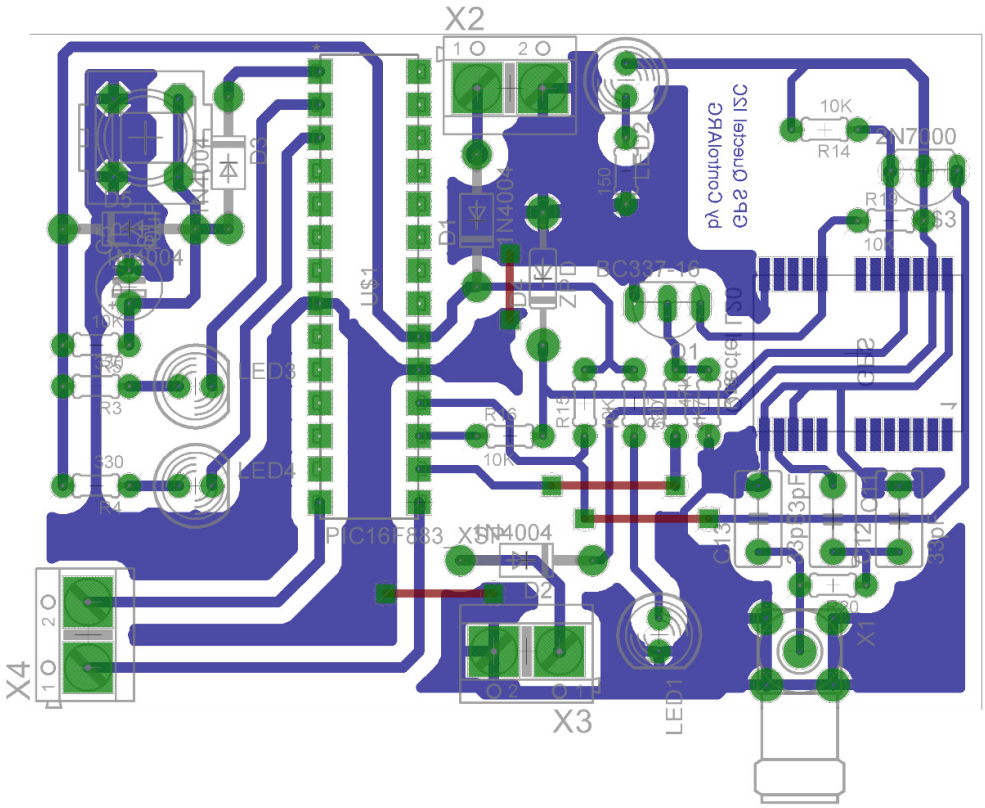


FIGURA 8.3 – Placa prototipo, módulo GPS, imagen esquemática generada con Eagle 6.2

9 PROTOCOLO I2C

9.1 Introducción

En el sistema desarrollado este es el protocolo mediante el cual se manejan las comunicaciones del CPU, con los diferentes periféricos que integran el sistema en su totalidad, como ser: modem GSM/GPRS, GPS y medidores de energía.

El motivo de utilización de este protocolo en particular fue el hecho de la gran adaptabilidad que el mismo presentaba para la tarea que se requería implementar, permitiéndonos este manejar hasta 127 esclavos (periféricos), con un maestro (CPU) y utilizando solo dos cables, ya que se caracteriza por ser un protocolo cuya capa física se implementa con solo dos cables (two wired interface).

Por estas particularidades y por ser que los microcontroladores PIC, que se utilizaron en el desarrollo del proyecto manejan este protocolo por hardware, es que optamos por implementar las comunicaciones del bus posterior sobre el mismo. [6]

9.2 Descripción Funcional del Protocolo I2C

[7] I²C es un bus de comunicaciones en serie. La versión 1.0 data del año 1992 y la versión 2.1 del año 2000, su diseñador es Philips. La velocidad es de 100 kbit/s en el modo estándar, aunque también permite velocidades de 3.4 Mbit/s. Es un bus muy usado en la industria, principalmente para comunicar microcontroladores y sus periféricos en sistemas integrados (Embedded Systems, o Sistemas Embebidos) y generalizando más para comunicar circuitos integrados entre sí que normalmente residen en un mismo circuito impreso.

La principal característica de I²C es que utiliza dos líneas para transmitir la información: una para los datos y otra para la señal de reloj. También es necesaria una tercera línea, pero esta sólo es la referencia (masa). Como suelen comunicarse circuitos en una misma placa que comparten una misma masa esta tercera línea no suele ser necesaria.

Las líneas se llaman:

- SDA: datos
- SCL: reloj
- GND: tierra

Las dos primeras líneas son de colector abierto, por lo que necesitan resistencias de pull-up (resistencia que intercalan entre la tensión de alimentación y la línea de comunicación).

Los dispositivos conectados al bus I²C tienen una dirección única para cada uno. También pueden ser maestros o esclavos. El dispositivo maestro inicia la transferencia de datos

y además genera la señal de reloj, pero no es necesario que el maestro sea siempre el mismo dispositivo, esta característica se la pueden ir pasando los dispositivos que tengan esa capacidad y es esta característica particular la que hace que al bus I²C se le denomine *bus multimaestro.*

Las transacciones en el bus I2C tienen este formato:

| start | A7 A6 A5 A4 A3 A2 A1 R/W | ACK | ... DATA ... | ACK | stop | idle |

- El bus está libre cuando SDA y SCL están en estado lógico alto.
- En estado bus libre, cualquier dispositivo puede ocupar el bus I²C como maestro.
- El maestro comienza la comunicación enviando un patrón llamado *"condición de start"*. Esto alerta a los dispositivos esclavos, poniéndolos a la espera de una transacción.
- El maestro se dirige al dispositivo con el que quiere hablar, enviando un byte que contiene los siete bits (A7-A1) que componen la dirección del dispositivo esclavo con el que se quiere comunicar, y el octavo bit (A0) de menor peso se corresponde con la operación deseada (L/E), lectura=1 (recibir del esclavo) y escritura=0 (enviar al esclavo).
- La dirección enviada es comparada por cada esclavo del bus con su propia dirección, si ambas coinciden, el esclavo se considera direccionado como esclavo-transmisor o esclavo-receptor dependiendo del bit R/W.
- El esclavo responde enviando un bit de ACK que le indica al dispositivo maestro que el esclavo reconoce la solicitud y está en condiciones de comunicarse.
- Seguidamente comienza el intercambio de información entre los dispositivos.
- El maestro envía la dirección del registro interno del dispositivo que se desea leer o escribir.
- El esclavo responde con otro bit de ACK
- Ahora el maestro puede empezar a leer o escribir bytes de datos. Todos los bytes de datos deben constar de 8 bits, el número máximo de bytes que pueden ser enviados en una transmisión no está restringido, siendo el esclavo quien fija esta cantidad de acuerdo a sus características.

- Cada byte leído/escrito por el maestro debe ser obligatoriamente reconocido por un bit de ACK por el dispositivo maestro/esclavo.
- Se repiten los 2 pasos anteriores hasta finalizar la comunicación entre maestro y esclavo.
- Aun cuando el maestro siempre controla el estado de la línea del reloj, un esclavo de baja velocidad o que deba detener la transferencia de datos mientras efectúa otra función, puede forzar la línea SCL a nivel bajo. Esto hace que el maestro entre en un estado de espera, durante el cual, no transmite información esperando a que el esclavo esté listo para continuar la transferencia en el punto donde había sido detenida.
- Cuando la comunicación finaliza, el maestro transmite una "condición de stop" para dejar libre el bus.
- Después de la "condición de stop", es obligatorio para el bus estar en idle durante unos microsegundos.

9.2.1 Desarrollo del software que controla el dispositivo maestro

La idea que se planteó desde un principio es que el dispositivo maestro consulte a los esclavos, ya sea para escribirlos, como para leerlos, cuando el mismo lo dese.

Todo lo que el software de control del esclavo debe hacer es cumplir la secuencia de operaciones necesarias, anteriormente descriptas para implementar la comunicación con el esclavo correspondiente, ya sea en una operación de lectura (recibir datos desde el esclavo), como de escritura (enviarle datos al esclavo).

9.2.2 Experimentación realizada para implementar el protocolo

Para implementar este protocolo de comunicación se llevó a cabo un proceso de desarrollo que comenzó con la investigación conceptual del protocolo, recopilando información y estudiando la misma, seguido por una etapa de implementación de un software tanto para manejar un dispositivo maestro (CPU), como los dispositivos esclavos (periféricos) y por ultimo una etapa de prueba de los programas planteados en los circuitos de prototipado.

La experimentación de campo realizada consistió en probar y depurar los softwares de control en implementación del protocolo tanto para el dispositivo maestro, como para los esclavos.

Haciendo de uso de un osciloscopio para observar y medir las formas de onda de las líneas de comunicación (SDA = Serial Data y SCL = Serial Clock) y compararlas con las formas de ondas proporcionadas por el fabricante de los micros en las hojas de aplicación y datos.

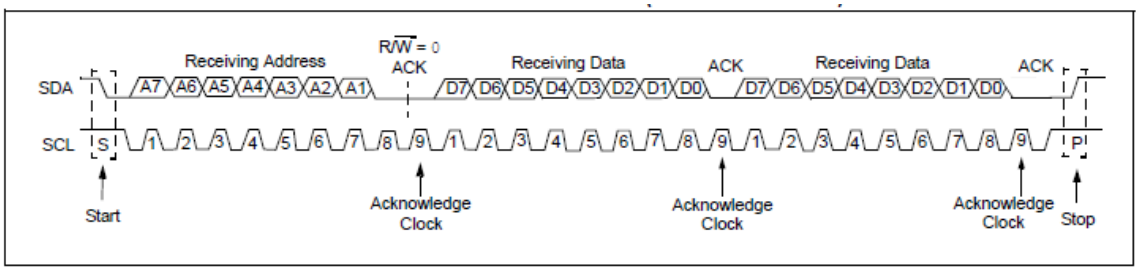
Además se efectuaron mediciones de tiempo, periodos y frecuencias, para corroborar que todo esté acorde a la tasa de transferencia de datos que se deseaba manejar, la cual se definió en el estándar de baja velocidad del bus 100Khz de frecuencia de clock, obteniéndose de este modo una tasa de transferencia de datos de 100 Kbits/seg.

Para corroborar los resultados obtenidos en la experimentación se hizo uso de las Notas de Aplicación: AN734 (Using the PIC Devices SSP and MSSP Modules for Slave I2C Comuncation), AN735 (Using the PIC Devices SSP and MSSP Modules for Master I2C Comuncation) y de las hojas de datos correspondientes a los PICs que se utilizaron, a recordar PIC16F883 y dsPIC33FJ128GP, todas obtenidas de las fuentes proporcionadas por Microchips.

Ver anexo II y/o DVD Anexo.

9.2.2.1 Escritura de un esclavo

La información proporcionada por el fabricante del micro en la hoja de datos nos indica que las formas de onda que debemos ver al momento de escribir un esclavo son:



Como vemos tenemos la condición de Start al iniciar la comunicación, en donde la línea de datos (SDA) pasa a bajo, mientras el clock está en alto. Lugo el maestro envía la dirección del esclavo con el cual desea comunicarse con el bit menos significativo (A0) en cero para indicar que desea iniciar una secuencia de escritura. Posteriormente en el noveno pulso del clock se genera las secuencia de Acknowledge (ACK), manteniendo la línea SDA en bajo y a continuación se envían los datos generándose siempre una secuencia de Acknowledge (ACK) al finalizar la transferencia del byte. Por último y para finalizar la transacción de información, cuando el maestro lo desea genera una secuencia de Stop, la cual se da cuando la línea SDA pasa a alto, estando el clock en alto.

A continuación podemos ver una captura de pantalla del osciloscopio, obtenida mediante las experimentaciones realizadas, en donde vemos claramente cómo se dan todas las

etapas de la comunicación, cumpliéndose con lo establecido con el protocolo. En todas las capturas de pantalla que se presentan, la línea de datos (SDA) está representada en color amarillo y la línea de reloj o clock (SCL) en color celeste.

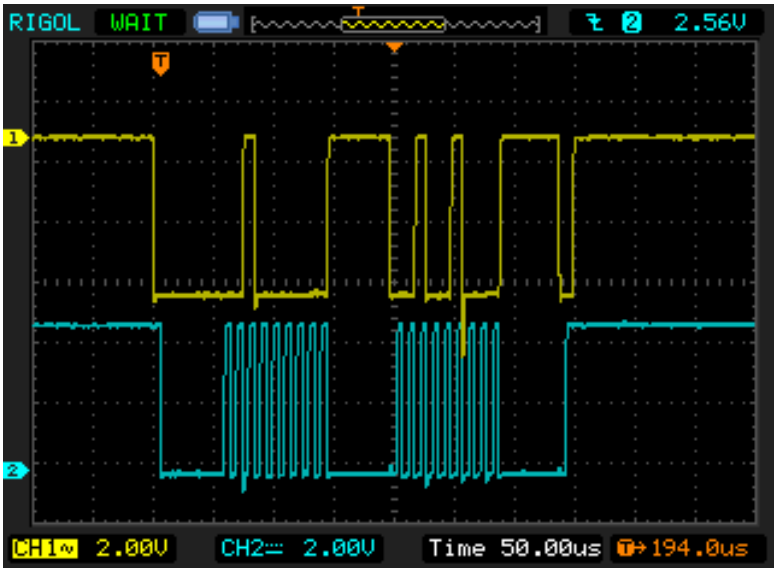


FIGURA 9.1 – Escritura de un esclavo

En la siguiente captura de pantalla del osciloscopio podemos ver en detalle la medición de frecuencia de la señal de clock y corroboramos de esta manera que el bus está trabajando a la velocidad deseada, cumpliéndose la tasa de transferencia de datos de 100 Kbits/seg.

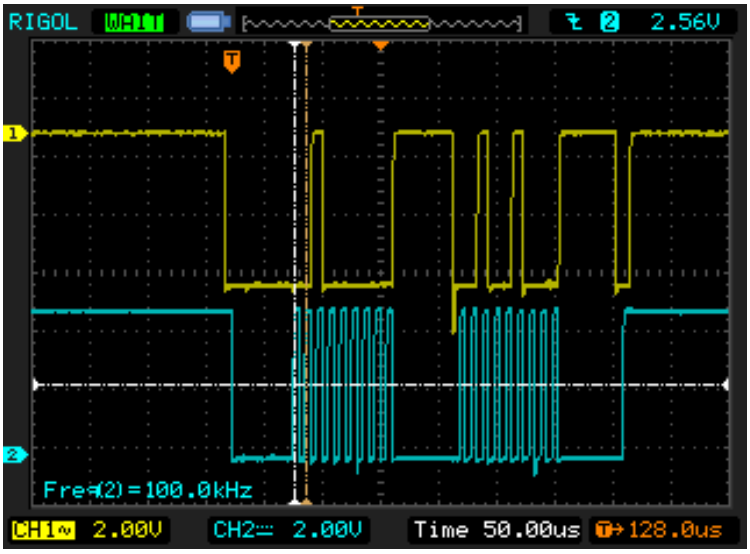


FIGURA 9.2 – Medición de la frecuencia del Clock

A continuación vemos la misma medición con más zoom en la captura de pantalla:

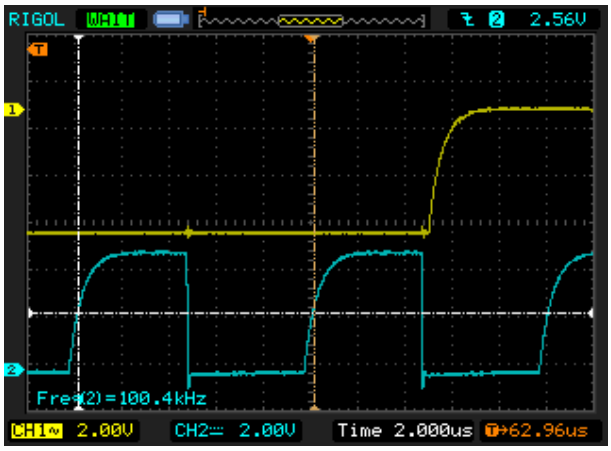


FIGURA 9.3 – Medición de frecuencia del Clock con más zoom

Como regla del protocolo, la aceptación de un dato (1 o 0), es decir la validación de un bit, se da siempre y cuando los cambios de estado en la línea de datos, se den mientras la línea de clock está en bajo. En la siguiente captura de pantalla podemos ver como se cumple correctamente este requisito del protocolo.

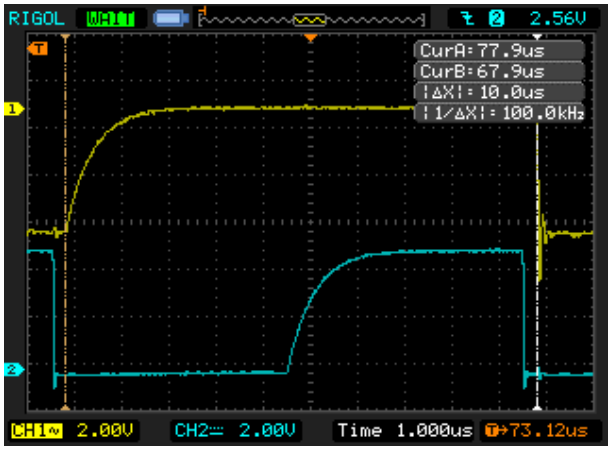
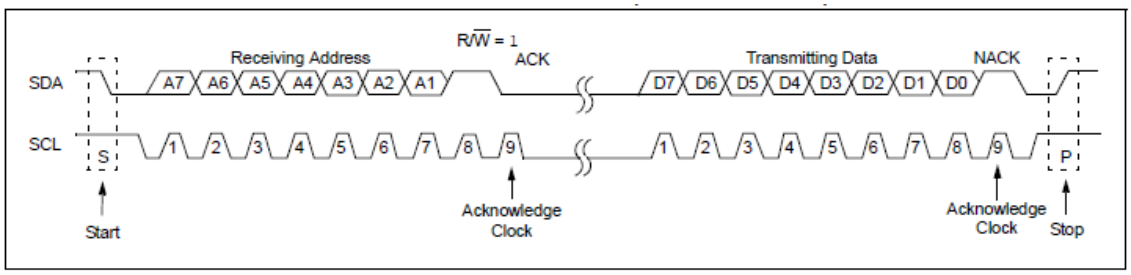


FIGURA 9.4 – Validación del dato en el protocolo I2C

9.2.2.2 Lectura de un esclavo

La información proporcionada por el fabricante del micro en la hoja de datos nos indica que las formas de onda que debemos ver al momento de leer un esclavo son:



Al momento de iniciar la comunicación el maestro genera la condición de start y seguidamente envía la dirección del esclavo con el cual desea comunicarse poniendo el bit menos significativo del byte de direccionamiento en 1, indicando de este modo que va a entablar una operación de lectura. Una vez recibido el byte de dirección el esclavo mantiene la línea de clock en bajo (generando lo que en ingles se denomina clock streching, o estiramiento del reloj), reteniendo el bus ocupado y tomándose el tiempo que le requiera generar la respuesta a la petición de lectura del maestro. Una vez elaborada la respuesta por parte del esclavo, el mismo libera la línea SCL permitiéndole al maestro volver a generar los pulsos de clock y comienza de esta manera el envío de datos desde el esclavo al maestro (tener presente que siempre es el maestro quien genera el pulso de reloj, ya sea que este escribiendo o leyendo un esclavo, es decir enviando o recibiendo datos).

A continuación vemos una captura de pantalla del osciloscopio en la cual se observa la transacción completa de datos maestro/esclavo en una operación de lectura de un esclavo.

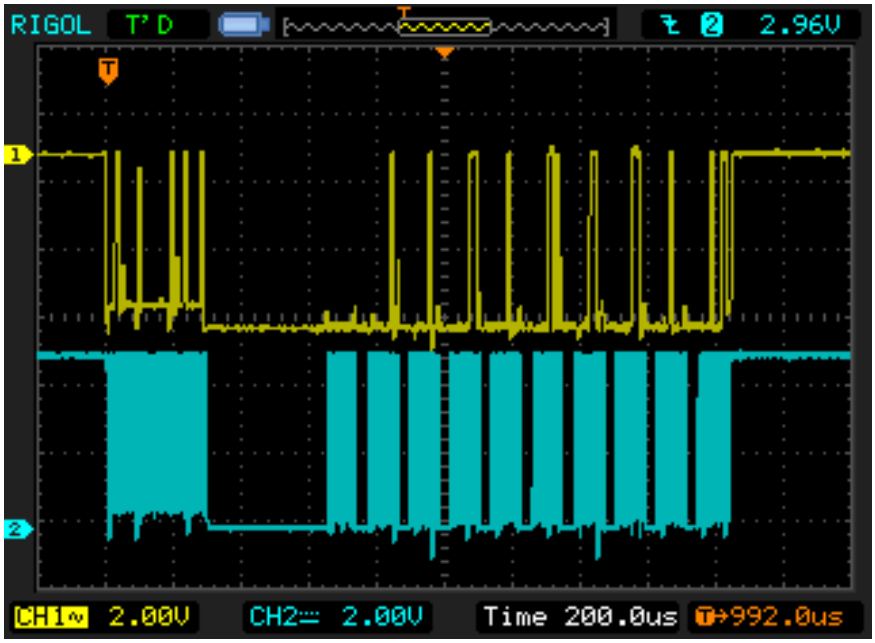


FIGURA 9.5 – Lectura de un esclavo

Como vemos en la siguiente imagen, la transacción comienza con una secuencia de start (SDA pasas de alto a bajo, mientras SCL está en alto), generada por el maestro. En la imagen también se aprecia la medición de frecuencia del clock, efectuada para corroborar que el bus este corriendo a la tasa de transferencia de datos deseada 100 Kbits/seg.

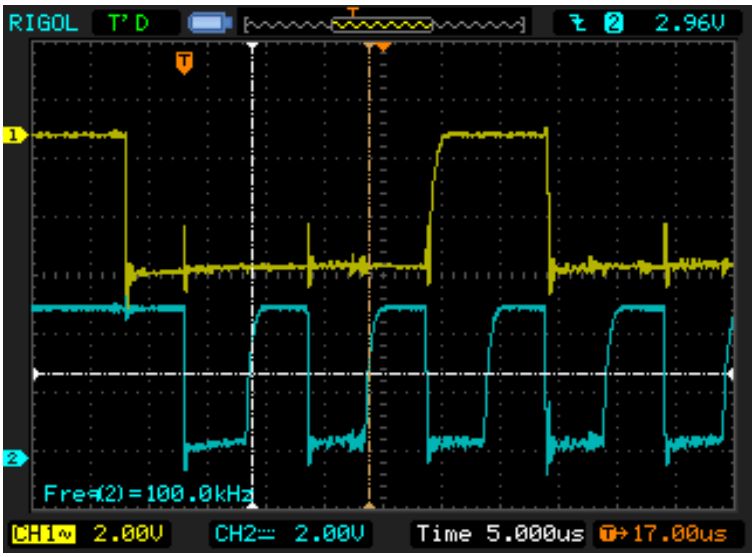


FIGURA 9.6 – Medición de frecuencia del Clock

A continuación, podemos ver como luego de la secuencia de start, el maestro envía al esclavo, tres bytes, uno correspondiente a la dirección del esclavo con el cual desea comunicarse en modo de escritura (bit menos significativo del byte de dirección en cero), otro para indicar la dirección del registro de memoria del esclavo desde el cual el maestro quiere comenzar a leer y por último el tercer byte enviado es la dirección del esclavo con el bit menos significativo en 1, para indicar que quiere iniciar una operación de lectura.

FIGURA 9.7 – Inicio secuencia de lectura

Es decir la operación de lectura, comienza como una operación de escritura en donde en primera instancia el maestro le envía al esclavo la dirección del registro de memoria a partir del cual quiere leerlo.

Luego de recibido el byte de direccionamiento con el bit de lectura en alto, el esclavo retiene la línea de clock SCL (amarilla) en bajo durante el tiempo que le lleve elaborar la respuesta y una vez procesada la respuesta, libera la línea de clock y el maestro comienza a recibir los bytes de datos que desea leer.

Por último y para finalizar la comunicación, una vez que el maestro leyó todos los datos (registros de memoria) que deseaba finaliza la transacción generando una secuencia de stop (SDA línea de datos, pasa de bajo a alto, mientras la línea de clock SCL permanece en alto).

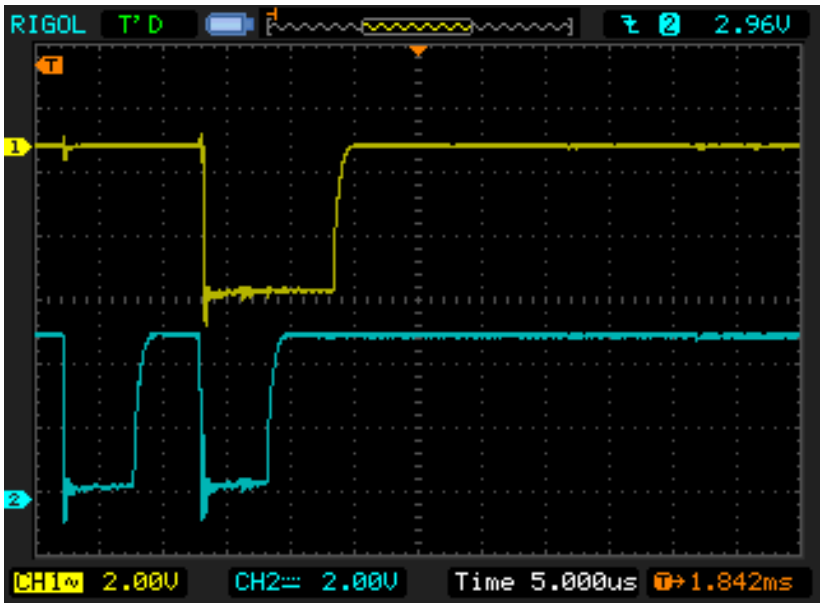


FIGURA 9.8 – Secuencia de Stop

9.3 Adaptación de 3,3V a 5V

En el proyecto abordado, se presentó un inconveniente técnico a la hora de implementar el protocolo de comunicación a nivel físico (en su capa física), ya que el microcontrolador utilizado para el CPU es un micro que funciona en 3,3V y los microcontroladores que manejan los periféricos son micros que funcionan en 5V, por ende se recurrió al diseño de un circuito que permita adaptar los niveles de tensión en la comunicación.

Para llevar a cabo dicho diseño hicimos uso de la nota de aplicación proporcionada por Philips al respecto (Philips es el inventor del protocolo I2C). La misma es la AN104441

(ver Anexo II) y en ella se explica detalladamente como diseñar un circuito bidireccional para adaptar los niveles de tensión haciendo uso de transistores FET y resistencias.

A continuación podemos ver el plano esquemático que se brinda en la nota de aplicación anteriormente mencionada y en base a la cual diseñamos la interface de comunicación.

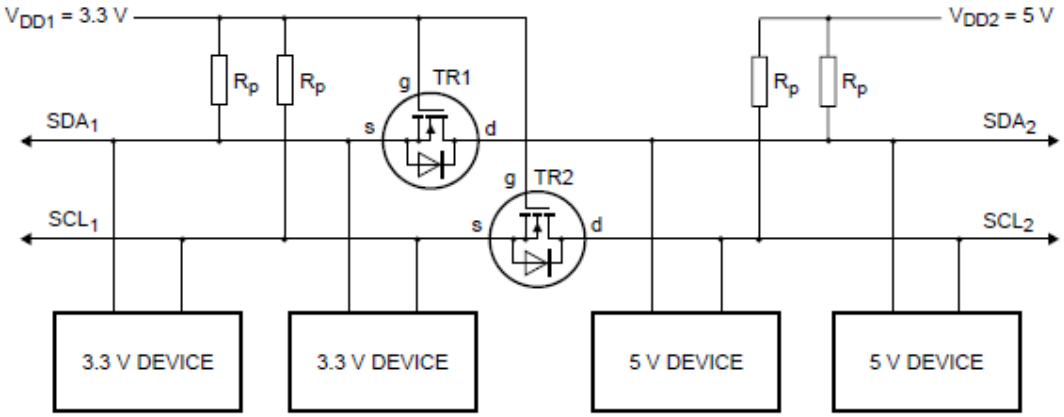


FIGURA 9.9 – Circuito adaptador de los niveles de tensión

En el diseño de la interface se utilizaron transistores MOSFET 2N7000, cuya hoja de datos se proporciona en el Anexo II, el motivo de selección del mismo fue principalmente su disponibilidad en el mercado local.

En el plano, N°13 – I2C Level Shifter Esquemático, podemos ver el esquemático del circuito diseñado, en los planos N°14 – I2C Level Shifter Componentes y N°14 – I2C Level Shifter Board podemos ver la placa de circuito impresa diseñada. La cual apreciamos también en la siguiente imagen:

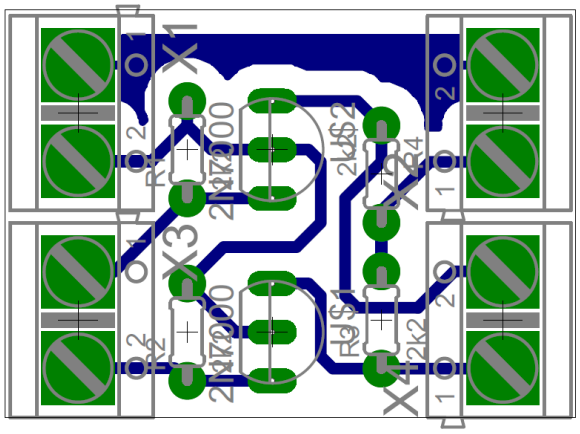


FIGURA 9.10 – Placa I2C Level Shifter, imagen ilustrativa generada con Eagle 6.2

9.3.1 Pruebas del circuito de adaptación de tensión:

Para corroborar el correcto funcionamiento del circuito diseñado, se procedió a montarlo en una placa de prototipado (también conocida como protoboard) y luego mediante el uso del osciloscopio se visualizaron las formas de onda para ver si efectivamente se generaba la conversión de niveles de tensión. Lo resultados obtenidos se presentan en las siguientes capturas de pantalla del osciloscopio:

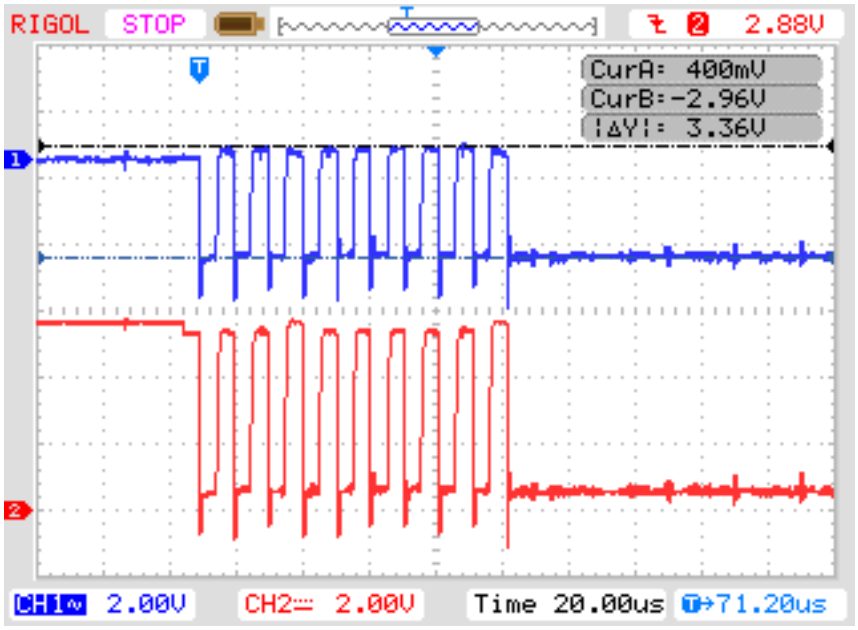


FIGURA 9.11 – Medición del voltaje señal de 3.3V

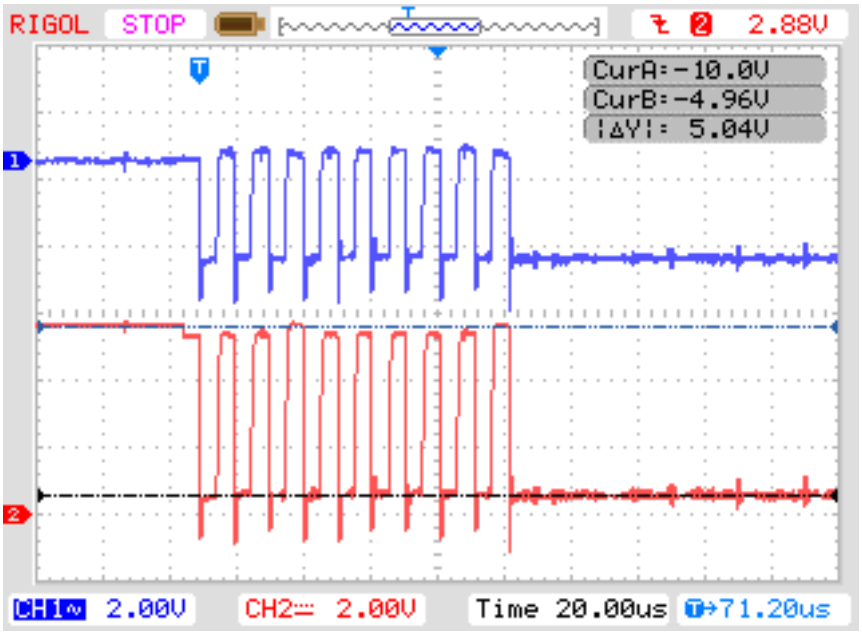


FIGURA 9.12 – Medición del voltaje señal de 3V

En la primer imagen, FIGURA 9.11, se ve mediante el uso de los cursores del osciloscopio que la señal presenta 3.36V de pico a pico, mientras que luego de pasar por el adaptador de niveles de tensión la señal, presenta la misma forma de onda, pero tiene un voltaje de pico a pico de 5.04V, FIGURA 9.12, quedando comprobado de esta manera el correcto funcionamiento del circuito diseñado.

La imagen, FIGURA 9.13, que vemos a continuación es el circuito de prueba que se utilizó, montado en un protoboard, o placa de prototipado.

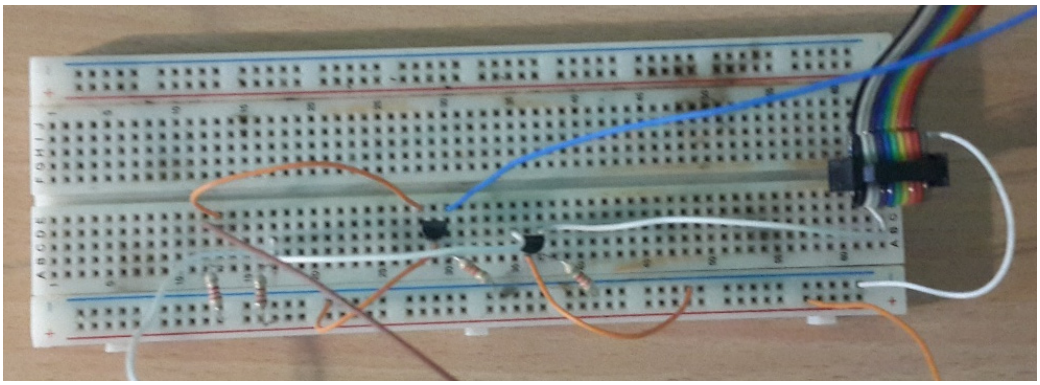


FIGURA 9.13 – Circuito montado en Protoboard para experimentación

En la misma podemos ver los dos transistores mosfet, uno para cada línea del bus y sus respectivas resistencias de pull-up.

10 MICROCHIP TCP/IP STACK

10.1 Introducción

Cuando hablamos del "Stack TCP/IP" (Stack en ingles significa pila) nos referimos al conjunto de programas, funciones, aplicaciones pre construidas (build-in), o librerías que el fabricante de los microcontroladores PIC, Microchips, nos proporciona para desarrollar aplicaciones embebidas que hagan uso de las posibilidades que una conexión TCP/IP brinda.

En el desarrollo del presente proyecto se hizo uso de estas librerías para implementar las funcionalidades de servidor web embebido. Esta es una muy útil característica que nos permite visualizar en un simple explorador web parámetros o variables del sistema que se esté censado/controlando entiempo real así como enviar datos al microcontrolador en forma remota.

El trabajo realizado con el stack TCP/IP consistió en investigar acerca de las funcionalidades del mismo y sus modos de aplicación y luego en una posterior etapa se desarrolló una pequeña aplicación para poner en prueba su funcionamiento. Básicamente se trabajó sobre dos conceptos:

10.2 Envió de datos desde el PIC a la Web:

Esta tarea se logra mediante lo que en el stack se conocen como variables dinámicas. En palabras básicas una variable dinámica es un valor que podemos visualizar en una página web, en tiempo real. Por ende podemos conocer el estado de entradas, salidas, sensores, etc. En tiempo real y de manera remota accediendo a la página web que el microcontrolador está cargando.

La utilidad de esta aplicación resulta evidente si pensamos que hoy en día encontramos acceso a exploradores web en un sinfín de dispositivos portables de uso cotidiano, como ser, Smartphones, tablets, etc. Por ende podríamos saber en tiempo real y desde la comodidad de nuestro teléfono de uso personal el estado de los parámetros que nos interese censar en un proceso o aplicación determinada.

Para implementar esta funcionalidad se desarrolló en lenguaje HTML una página web sencilla en la que se muestra un conjunto de valores cargados a partir de variables dinámicas desde el microcontrolador. A continuación vemos una imagen de la página web desarrollada, FIGRUA 10.1, con la aplicación corriendo. En la misma se observa una lista de las variables dinámicas implementadas, todos los parámetros de esa lista están siendo enviados desde el PIC a la web.

En el DVD adjunto se dejan video que muestran esta funcionalidad corriendo en tiempo real.

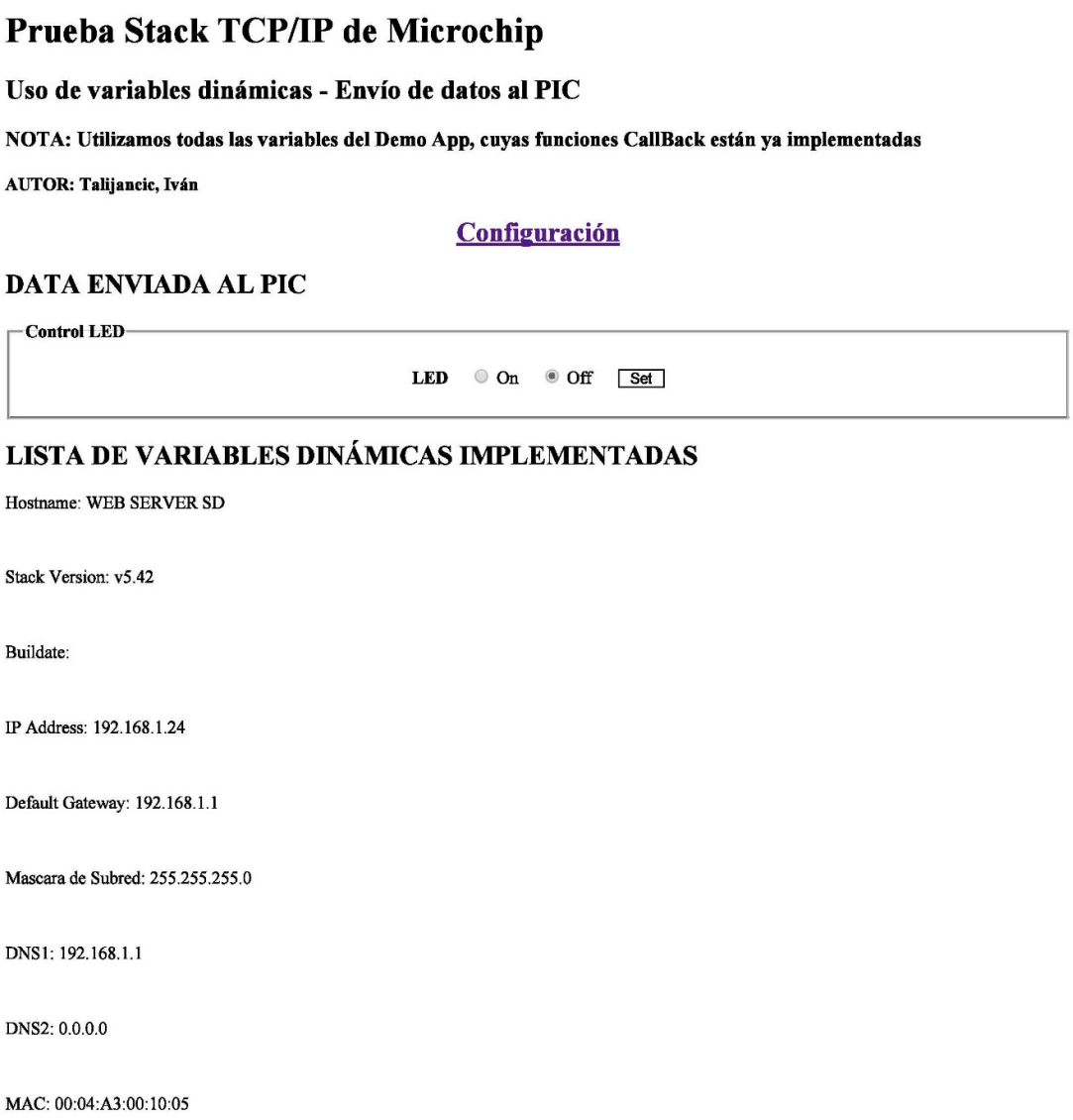


FIGURA 10.1 – Web desarrollada para probar el Stack

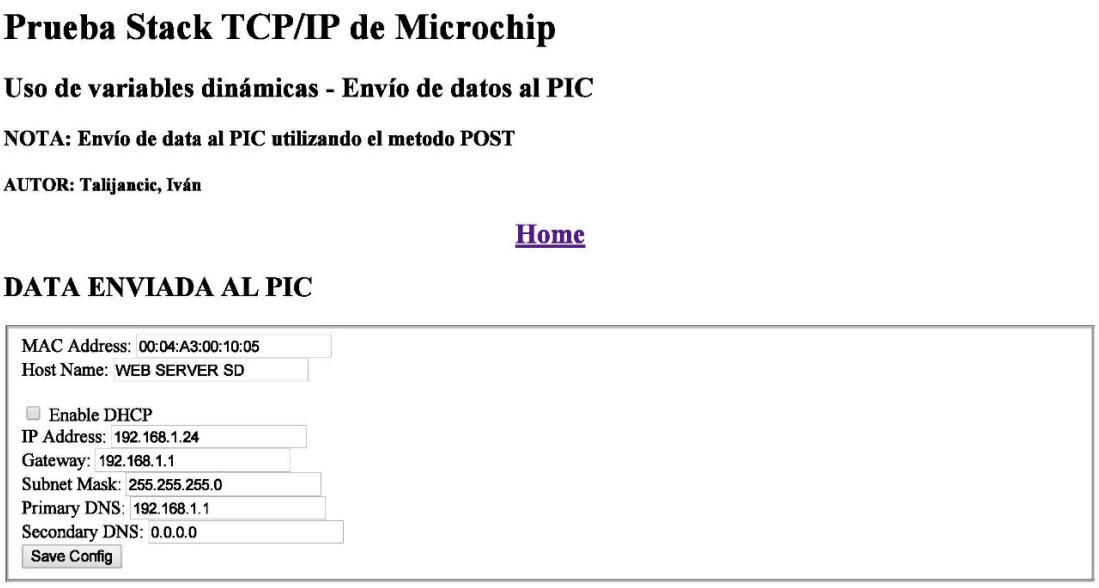
10.3 Envío de datos desde la web al PIC

La tarea de enviar datos desde la web al PIC es muy interesante porque brinda la posibilidad de manejar salidas, activarlas o desactivarlas, así como también enviar datos de configuración que modifiquen el comportamiento del programa que está corriendo en el microcontrolador.

Si observamos la imagen anterior podemos ver que hay un recuadro, debajo del título "DATA ENVIADA AL PIC", en el cual tenemos implementado un sistema de control para

establecer el estado de un led, prendido o apagado. A los fines de la simulación se manipulo un led, pero resulta evidente que la utilidad recae en que ese led es para nosotros una salida digital que podría estar accionando un relé y energizar la bobina de un contactor para encender el motor eléctrico de una bomba de agua, por dar un ejemplo de aplicación práctica real.

También se trabajó en el envío de grandes paquetes de información o datos desde la web al PIC lo cual nos permite, por ejemplo, definir parámetros de configuración. En la siguiente imagen, FIGURA 10.2, se ve la página web de prototipo desarrollada para poner a prueba esta funcionalidad.



FIGRUA 10.2 – Envío de los parámetros de configuración de la web al PIC

Como se ve en la imagen anterior en este prototipo de página de configuración se planteó una interface que nos permita definir los parámetros de la conexión de red del equipo, como ser el nombre del Host, la dirección de IP que quiero asignarle al mismo, el Getway mediante el cual tengo acceso a internet y demás parámetros funcionales que definen la configuración de red.

En el equipo final a desarrollar se pretende utilizar esta característica o funcionalidad, brindada por la conexión Ethernet en conjunto con el Stack TCP/IP para configurar el equipo en su totalidad. De modo tal que el usuario, pueda una vez adquirido el producto, definir los parámetros funcionales desde un simple navegador web.

11 PRESUPUESTO

11.1 Introducción

En el presente capitulo se realiza un resumido examen o análisis económico, para evaluar a grandes rasgos la viabilidad económica del proyecto, pensando en su posible comercialización. Desarrollando para ello en primera instancia un desglose del costo por módulo funcional que integra el dispositivo para obtener un costo total del dispositivo.

Es debido, tener en cuenta, que los cálculos de costo fueron realizados, a partir de los costos derivados de la fabricación de los prototipos, por ende son maximalistas, ya que pensando en una posible fabricación en cantidad, los costos de producción disminuirían notoriamente.

11.2 Costo por Módulo

- CPU: U$S 300
- MODULO DE MEDICIÓN DE ENERGÍA: U$S 200
- MÓDULO GSM/GPRS: U$S 200
- MODULO GPS: U$S 100
- ***TOTAL: U$S 800***

Como vemos en el desglose realizado, el costo del dispositivo, sin incluir los TIs, TVs y la Pt 100 (sensor de temperatura), es de ***U$S 800.***

11.3 Breve análisis de viabilidad económica:

Para efectuar un análisis de viabilidad económica del proyecto o del producto final al cual se pretende arribar, se encara el problema analizando el costo del equipo que se protegerá (transformador de distribución), en relación al costo del dispositivo diseñado.

Al costo del dispositivo, anteriormente detallado, se le agrega ahora el costo de los transformadores de medida TIs y TVs y el costo del sensor de temperatura, para luego sí compararlo con el costo del transformador y ver la relación de costos que existe.

La relación de costo se efectúa para transformadores de tres potencias, que son las más usadas por las empresas de distribución de energía, 630 kVa, 315 kVA y 125 kVA. En cada caso el costo del equipo es el mismo, lo que varían son los costos de los transformadores de medida.

11.3.1 Relación de costos trafo de 630 kVA

- Valor promedio de un transformador de distribución de 630 kVA : U$S 25.000

Costo total del equipo:

- Valor del equipo de protección diseñado: U$S 800
- Valor de la Pt 100 con transmisor: U$S 340
- Valor de los TV: U$S 190
- Valor de los TI: U$S 260
- TOTAL: U$S 1.590
- Relación de costos:

$$\%Val = \frac{\text{U\$S } 1.590}{\text{U\$S } 25.000}\ x\ 100$$

$$\%\boldsymbol{Val = 6,36}\%$$

Es decir, que el costo del sistema de protección propuesto es menor al 10% del costo de la máquina que estoy protegiendo, en el caso de un transformador de distribución de 630 kVA, por lo que es de esperarse que resulte en una inversión atractiva para aquel usuario que pretenda incrementar la seguridad en los equipos de su instalación.

11.3.2 Relación de costos trafo de 315 kVA

- Valor promedio de un transformador de distribución de 315 kVA : U$S 15.000

Costo total del equipo:

- Valor del equipo de protección diseñado: U$S 800
- Valor de la Pt 100 con transmisor: U$S 340
- Valor de los TV: U$S 190
- Valor de los TI: U$S 200
- TOTAL: U$S 1.530
- Relación de costos:

$$\%Val = \frac{\text{U\$S } 1.530}{\text{U\$S } 15.000}\ x\ 100$$

$$\%\boldsymbol{Val = 10,2}\%$$

Es decir, que el costo del sistema de protección propuesto es apenas superior al 10% del costo de la máquina que estoy protegiendo, por lo tanto, para el caso de un transformador

de 315kVA, podemos ver que no existe una mala relación costo beneficio, por ende es de esperarse que la inversión también resulte atractiva y racional.

11.3.3 Relación de costos trafo de 125 kVA

- Valor promedio de un transformador de distribución de 125 kVA : U$S 10.000

Costo total del equipo:

- Valor del equipo de protección diseñado: U$S 800
- Valor de la Pt 100: U$S 340
- Valor de los TV: U$S 190
- Valor de los TI: U$S 180
- TOTAL: U$S 1.510
- Relación de costos:

$$\%Val = \frac{\text{U\$S } 1.510}{\text{U\$S } 10.000}\ x\ 100$$

$$\%\boldsymbol{Val = 15,1}\%$$

Como vemos en el cálculo anteriormente realizado, para un trafo de 125 kVA la relación de costos, resulta menos favorable y ya que el costo del dispositivo de protección pasa a ser considerable en comparación al costo de la máquina que estoy protegiendo.

Debemos tener presente que en este análisis, se está trabajando sobre una sola de las ventajas que la implementación del equipo tendría para el usuario, que es impedir la avería de la máquina más costosa de la instalación.

Pero como bien se destacó en la memoria descriptiva, hay una serie de otras ventajas que la implementación del sistema traería aparejado, como ser un conocimiento en tiempo real del estado del sistema de distribución, mejoramiento del servicio brindado por parte de las empresas distribuidoras de energía, evitando periodos de desabasto y pudiendo hacer un uso más racional de sus equipos. Así como eliminar la riesgosa tarea de enviar operarios a realizar mediciones sobre las instalaciones, que debemos tener en cuenta, se realizan en altura en la mayoría de los casos, y obviamente operando con presencia de tensiones peligrosas.

Todo este conjunto de beneficios, no tenidos en cuenta en el breve análisis económico realizado, obviamente entraran en juego obrando a favor de la inversión, cuando el usuario se plantee si es o no es racional adquirir el equipo.

Es por ello que concluyo que resulta viable desde el punto de vista económico la realización del sistema propuesto en el presente proyecto final de carrera.

12 CONCLUSIONES Y PROPUESTAS DE MEJORAS

12.1 Conclusiones Generales

Hace aproximadamente dos años yo me encontraba desbordante de emoción y alegría al poder escribir, compilar y correr en el mundo real un programa que consistía en hacer que un microcontrolador prenda y apague un led. Esta simple tarea, sin gran aplicación práctica, significaba para mí la culminación exitosa de horas de investigación, recopilación de información y experimentación autodidacta para alcanzar un sueño que perseguía desde mi adolescencia, programar microcontroladores.

Finalizar este trabajo, desarrollando diversos programas para efectuar numerosas tareas, diseñando circuitos impresos, manejando un sin fin de herramientas computarizadas de asistencia al desarrollo y obteniendo prototipos funcionales que puedo ver y tocar, luego de imaginarlos, diseñarlos y fabricarlos, es haber alcanzado mi sueño con creces y mucho más.

En lo profesional puedo enumerar hasta el cansancio aspectos positivos acarreados en la realización de este proyecto, que van desde superar desafíos nuevos y desconocidos, hasta aprender a trabajar sobre un proyecto real llevado adelante por un grupo de trabajo.

En lo que al proyecto en sí respecta, considero alcanzado los objetivos en primera instancia planteados y creo que aunque queda un largo camino por recorrer, los primeros pasos y tal vez los más difíciles en un desarrollo de este tipo, fueron realizados con éxito. Dejando esto en evidencia que los conocimientos y habilidades adquiridos durante el desarrollo de mí formación profesional como futuro Ingeniero Electromecánico me permitieron y permitirán desempeñarme satisfactoriamente en el ámbito laboral.

12.2 Propuestas de Mejoras

Como posibles mejoras a aportar para la obtención de un producto final, sería interesante trabajar sobre los siguientes puntos:

- Poner a prueba otros módulos GSM/GPRS embebidos.
- Estudiar la utilización de módulos embebidos que en un solo encapsulado integren GSM/GPRS/GPS, así se podría implementar en un solo módulo todas estas funcionalidades.
- Estudiar la implementación de otro integrado para la realización de las mediciones de los parámetros eléctricos. Luego de hacer un pequeño estudio de alternativas, se presenta, al menos en primera instancia, como un buen sustituto del ADE 7758, el CI M90E32AS, de la firma ATMEL, ver Anexo II,

que es un circuito capaz de efectuar mediciones de energías, potencias, corrientes y tensiones, pero que presenta mayores facilidades en su aplicación ya que realiza internamente los cálculos de potencia y elimina las dificultades de calibración que se plantean en la utilización del ADE7758 de la firma Analog Devices.

- Profundizar el estudio del Stack TCP/IP de Microchip para implementar otras funcionalidades del mismo, como ser, envío y recepción de emails, trabajo en páginas web más depuradas desde el punto de vista gráfico , de modo tal que resulten más agradables e intuitivas para el usuario.

12.3 Curso de Acción Futura

Lo que proyecto a futuro para lograr el objetivo final de transformar mi proyecto final de carrera en un producto terminado y listo para su comercialización, puede resumirse en el siguiente conjunto de actividades:

- Poner a prueba un prototipo general, en el que se constate el correcto funcionamiento de todos los módulos en conjunto y operando sobre una subestación transformadora real.
- Encaminar un curso de acción para implementar las mejoras, anteriormente detalladas y las modificaciones que surjan de la experimentación o puesta a prueba del prototipo.

BIBLIOGRAFÍA

[1] http://www.mikroe.com/chapters/view/84/libro-de-la-programacion-de-los microcontroladores-pic-en-basic-capitulo-1-mundo-de-los-microcontroladores/

[2] EDUARDO GARCÍA BREIJO, Compilador C CCS y Simulador Proteus para Microcontroladores PIC, 1. Ed, Mexico, Alfaomega, 2008.

[3] https://es.wikipedia.org/wiki/LabVIEW

[4] EDUARDO ZUCCALA, Nota de Aplicación Módulo Motorola G20 – Análisis Básico para una comunicación digital óptima, Diciembre de 2004.

[5] EDUARDO ZUCCALA, Nota de Aplicación Módulo Motorola G20 – Ejemplo de Uso del Stack Interno TCP/IP, Diciembre de 2004.

[6] http://robots-argentina.com.ar/Comunicacion_busI2C.htm

[7] https://es.wikipedia.org/wiki/I%C2%B2C

ANEXO I: Laboratorios en LabVIEW

Explicación

En el presente anexo se muestran unas imágenes, capturas de pantalla, de las experimentaciones realizadas en el software LabVIEW para poner a prueba las rutinas de control (programas) del módulo GSM/GPRS embebido.

Básicamente el trabajo realizado consistió en hacer un programa que simule el comportamiento del microcontrolador y que tenga una interface de usuario que me permita ver lo que se está enviando al modem y las respuestas que el mismo va generando ante cada comando AT que se le envía.

A1.1 Rutina de Inicialización del Modem

En la siguiente imagen, FIGURA A1.1, se pone a prueba la rutina de inicialización del modem, planteada en el diagrama de flujo N°1.

En la interface del programa creado en LabVIEW, vemos un conjunto de botones que corresponden a los comando AT que quiero enviarle al modem, un ventana de visualización de lo enviado y una ventana de visualización de las respuestas recibidas.

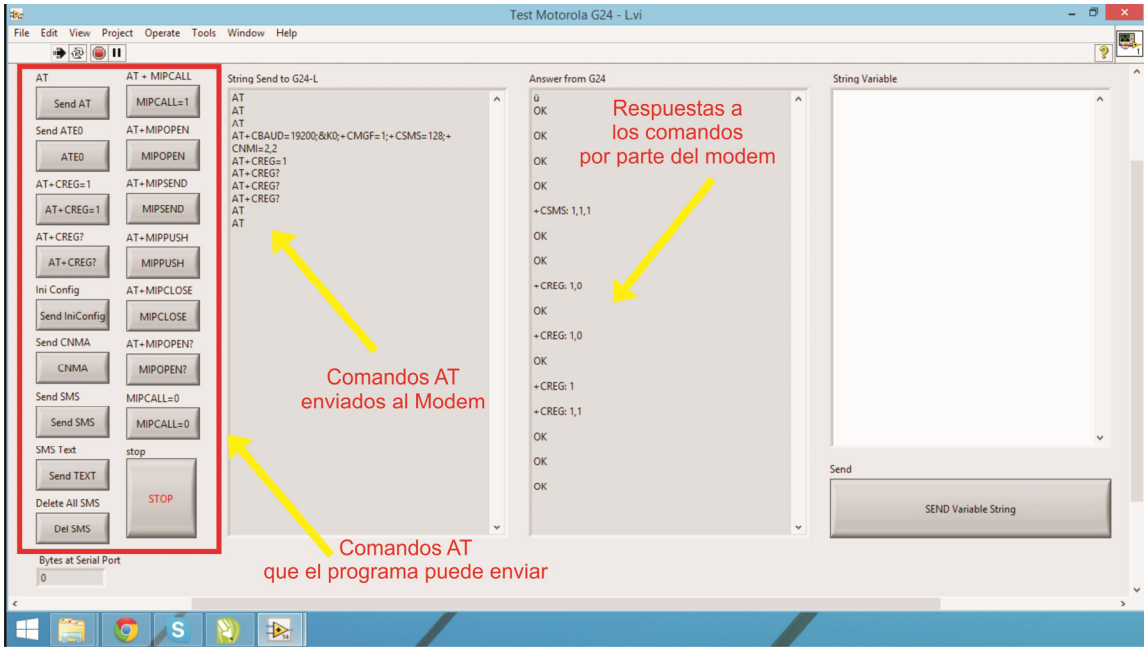


FIGURA A1.1

A1.2 Envío de un SMS

En la siguiente captura de pantalla, FIGURA A1.2, se ejecuta la rutina de envío de un SMS, planteada en el diagrama de flujo N°2.

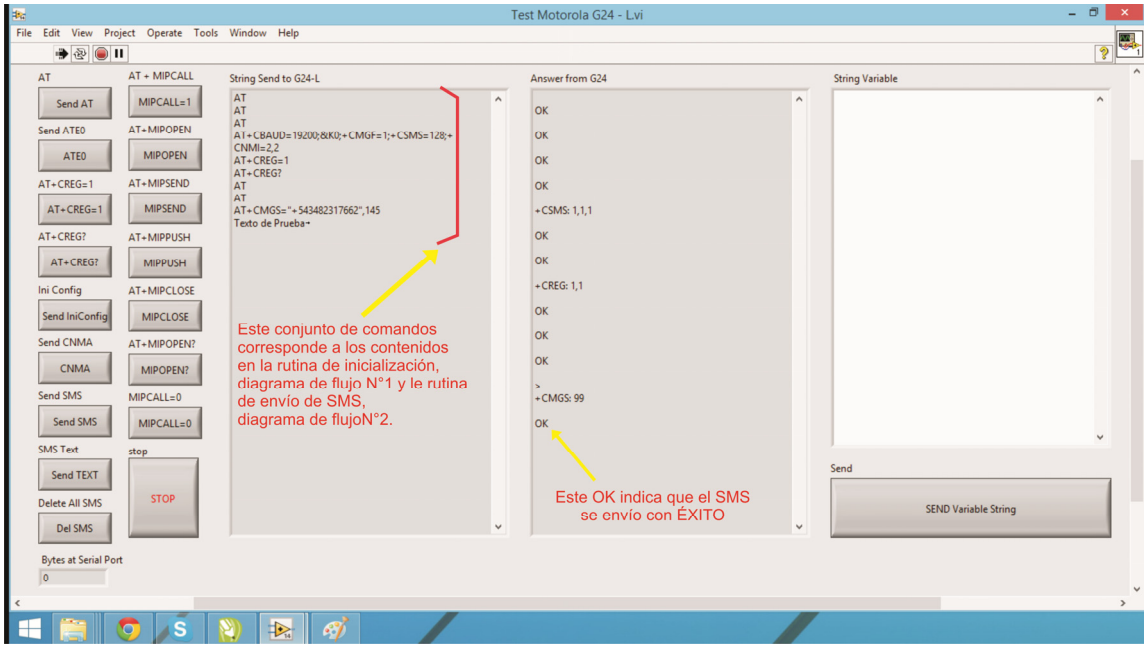


FIGURA A1.2

Como vemos en esta captura de pantalla, se sigue la rutina de envió de SMS y se ejecuta con éxito, lo cual sabemos por que recibimos el OK, por parte del modem al final del proceso.

En la siguiente captura de pantalla doble, FIGURA A1.3, lo que se muestra es la misma consecución de comandos AT enviados y respuestas recibidas por parte del modem, que en la FIGURA A1.2, pero con el agregado que también vemos el "código del programa", que como bien se explicó esta codificado en un lenguaje simbólico o gráfico, llamado lenguaje G, que nos permite escribir programas gráficamente a partir de bloques funcionales pre-construidos. Esta es la gran ventaja que presenta el LabVIEW y el motivo por el cual lo utilice como herramienta de asistencia al desarrollo del presente proyecto final de carrera.

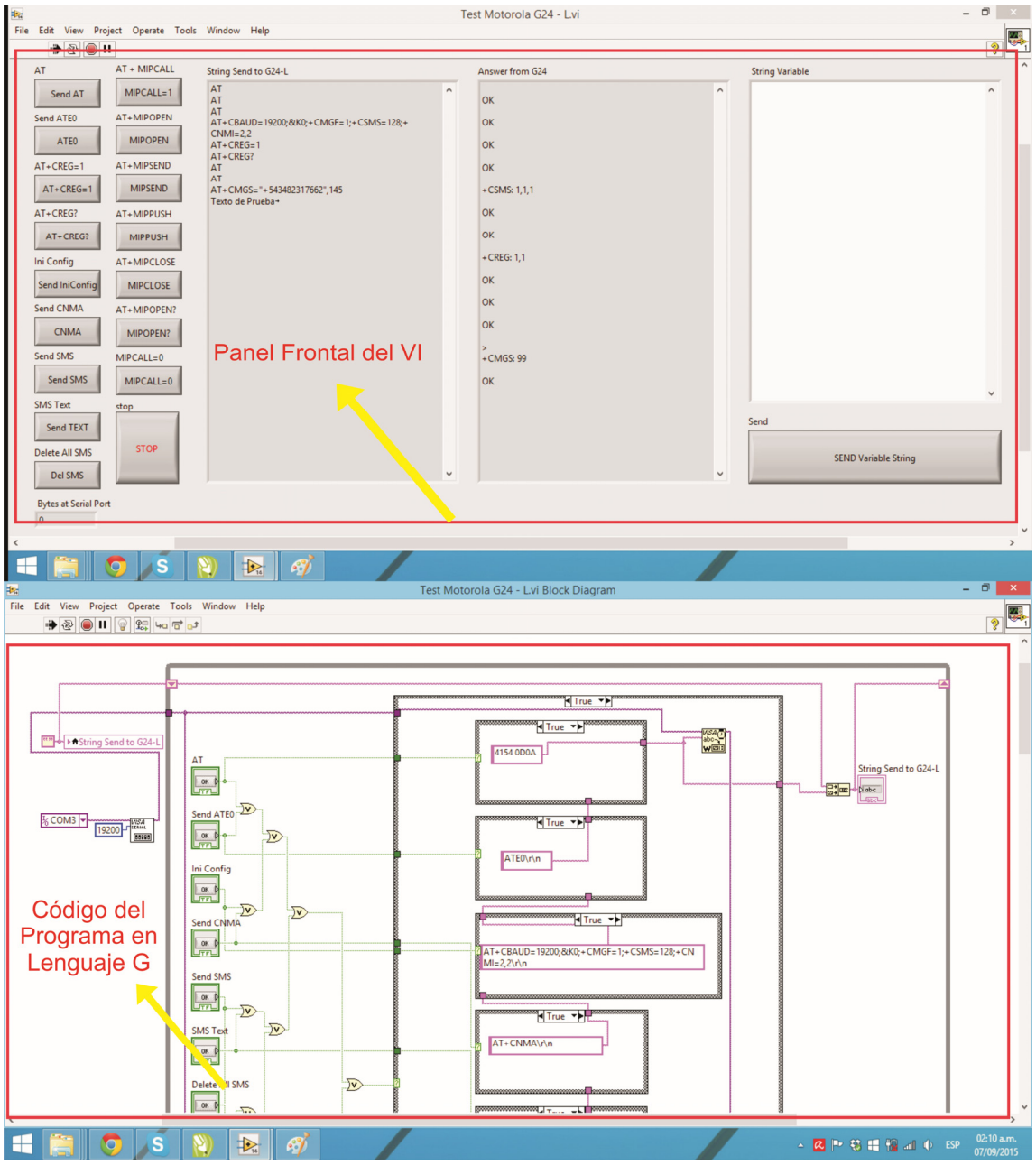


FIGURA A1.3

A1.3 Manejo del Stack TCP/IP

En la captura de pantalla, a continuación presentada, FIGURA A1.4, se visualiza la prueba de la rutina de establecimiento de una conexión TCP/IP, esquematizada en el diagrama de flujo N°4 y el posterior envío de datos, esquematizado en el diagrama de flujo N°5.

Tambien se visualiza un programa realizado en visual basic que hace las veces de servidor, manteniendo un puerto abierto y esperando que el modem GSM/GPRS se conecte a

él para enviarle o recibir datos. Este pequeño programa, simula a escala muy reducida la funcionalidad del servidor del sistema.

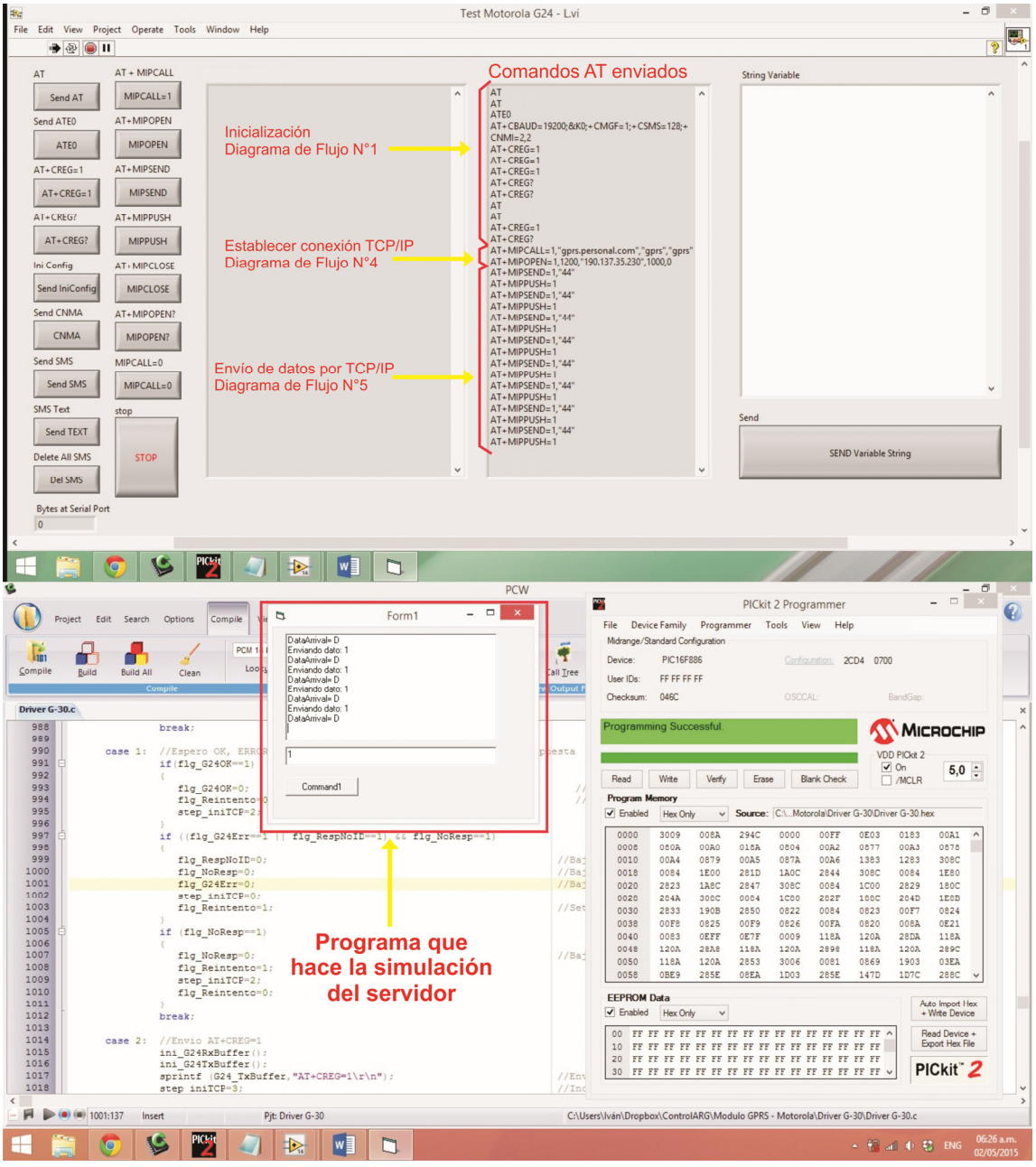


FIGURA A1.4

Observando la figura anterior vemos como se corroboró el correcto funcionamiento de las rutinas planteadas en lo referente a establecer una conexión TCP/IP y enviar datos a través de la misma.

Si vemos el dato enviado por el programa es un “44” y el recibido por el servidor es una “D” esto es así porque el 44 es la codificación ASCII hexadecimal de la letra D.

En la siguiente captura de múltiples pantallas, FIGURA A1.5, vemos como se ponen a prueba las mismas rutinas que en la FIGURA A1.4, pero con la diferencia que en este caso estamos observando las respuestas recibidas por parte del modem GSM/GPRS, ante cada comando AT enviado.

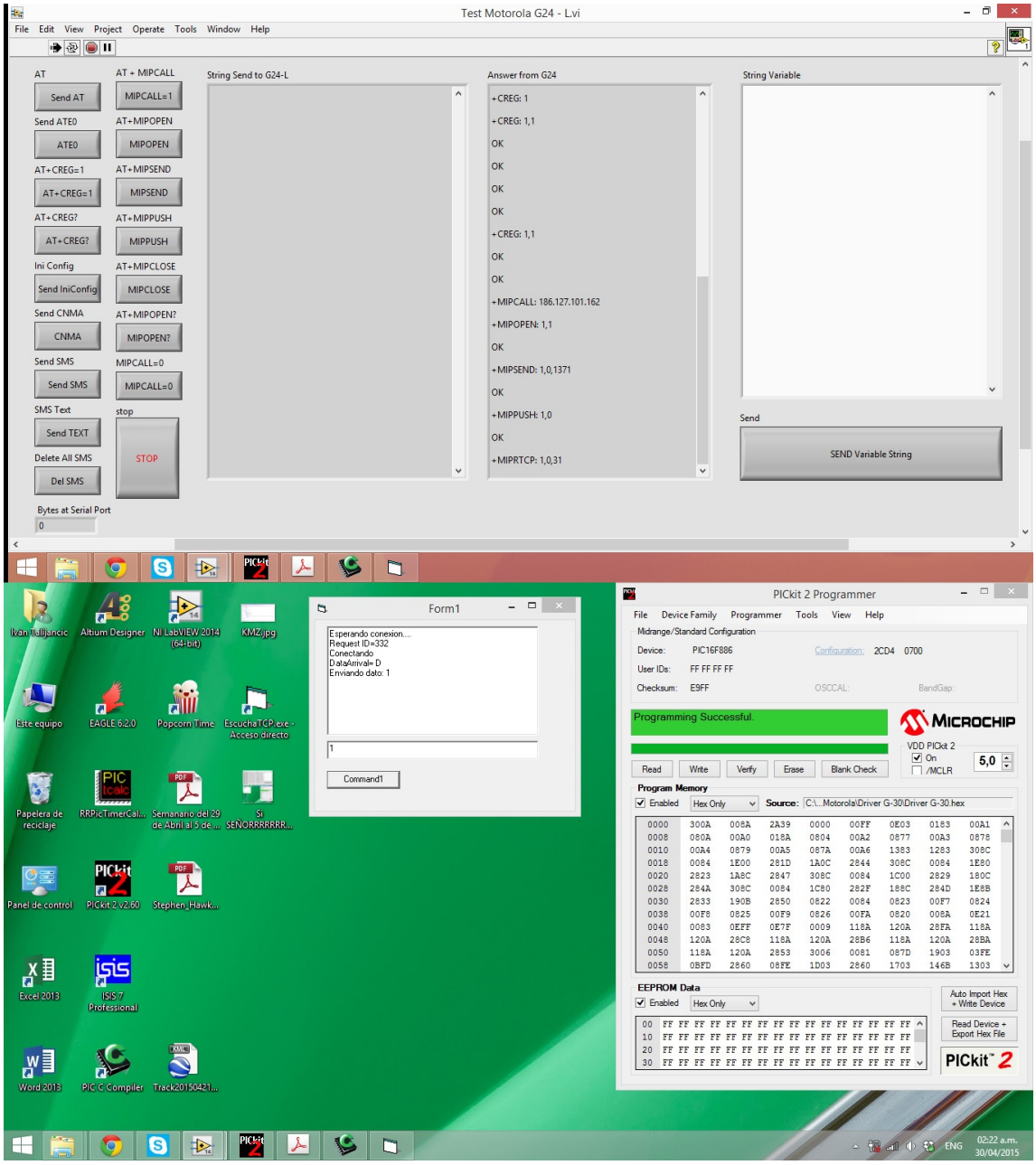


FIGURA A1.5

ANEXO II: Manuales y Catálogos

Lista de Manuales y Catálogos

- ADE 7758.
- Atmel M90E32AS.
- Motorola G30 Developer´s Guide, AT Comands Reference Manual.
- Quectel L20 Quectel GPS Engine, GPS Protocol Specification.
- Quectel L20 Quectel GPS Engine, Hardware Desing.
- Microchip AN735, Using the PICmicro Module for Master I2C Communicatios.
- Microchip AN735, Using the PICmicro Module for Master I2C Communicatios.
- Microchip AN734, Using the PICmicro Module for Slave I2C Communicatios.
- AN 10441, Level Shifting techniques in I2C bus desing.
- 2N7000 Hoja de datos.

ANEXO III: PLANOS

Lista de planos

- Plano N°1 – CPU Esquemático
- Plano N°2 – CPU Componentes
- Plano N°3 – CPU Board
- Plano N°4 – ADE7758 Esquemático
- Plano N°5 – ADE7758 Componentes
- Plano N°6 – ADE7758 Board
- Plano N°7 – Lector ADE7758 Esquemático
- Plano N°8 – Lector ADE7758 Componentes
- Plano N°9 – Lector ADE7758 Board
- Plano N°10 – GPS Esquemático
- Plano N°11 – GPS Componentes
- Plano N°12 – GPS Board
- Plano N°13 – I2C Level Shifter Esquemático
- Plano N°14 – I2C Level Shifter Componentes
- Plano N°15 – I2C Level Shifter Board
- Plano N°16 – Conexión del Módulo de Medición de Energía