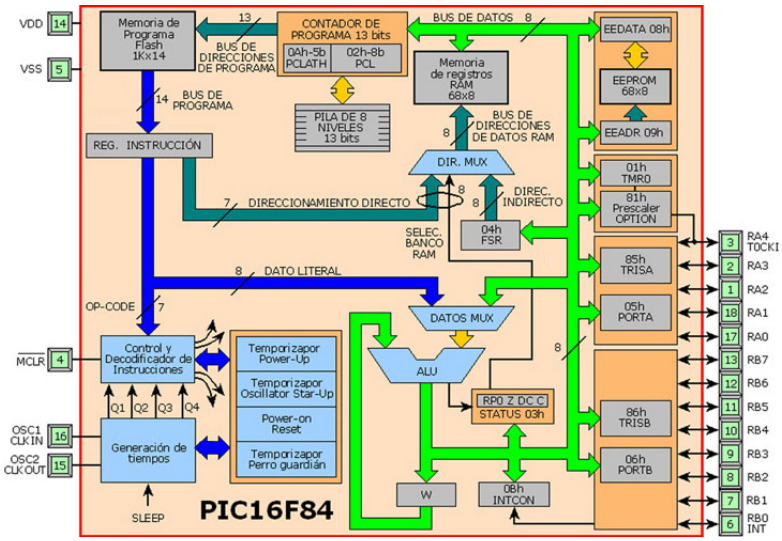
Proyecto GPS – SMS/GSM

Por: Rommel J. Contreras G.
rommeljose@gmail.com

Resumen

Se pretende un dispositivo que administre la comunicación con un GPS mediante el envío y resección de mensajes de texto (SMS, utilizando como elemento de control un microprocesador PIC 16F84A. Enviando un mensaje SMS, un usuario remoto puede consultar la ubicación geográfica del dispositivo, obteniendo la latitud, longitud, y otros parámetros, a la vuelta del mensaje enviado por el teléfono celular conectado al PIC. Para la programación se utilizó el ambiente de desarrollo integrado MPLAB de Microchip Technology Incorporate, mediante la herramienta MPASM (*Assembler*). El dispositivo fue simulado en PROTEUS VSM de Labcenter Electronics. Para la fase de prueba y diseño se utilizó un *protoboard*[i], y para el prototipo inicial se utilizó una baquelita pre perforada[ii]; soldando en ella los componente y conectores.

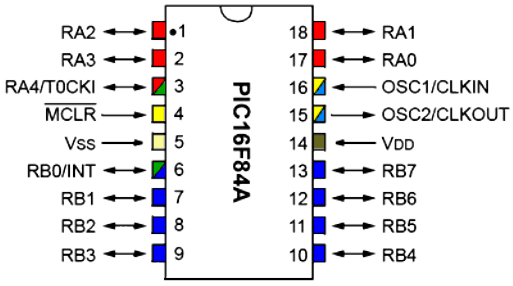
El PIC 16F84A



Este dispositivo desarrollado por la empresa Microchip en 1985 es un circuito integrado que contiene a escala reducida en su interior las partes principales de un micro computador; conocidos como PIC *(Peripheral Interface Controll)*. Se comunican con el exterior mediante buses o puertos configurables como entrada o salida. «Se emplea para controlar el funcionamiento de una tarea determinada y, debido a su reducido tamaño, suele ir incorporado en el propio dispositivo al que gobierna. Esta última característica es la que le confiere la denominación de «controlador incrustado» *(embedded controller).* El microcontrolador es un computador dedicado. En su memoria sólo reside un programa destinado a gobernar una aplicación determinada; sus líneas de entrada/salida soportan el conexionado de los sensores y actuadores del dispositivo a controlar, y todos los recursos complementarios disponibles tienen como única finalidad atender sus requerimientos. Una vez programado y configurado el microcontrolador solamente sirve para gobernar la tarea asignada»[iii].

El PIC 16F84 por muchos años ha sido uno de los microcontroladores más populares debido a su facilidad de uso y programación. Como todo los PIC de la gama media de Microchip, posee arquitectura Harvard y está basado en tecnología RISC, lo que nos permite una programación sencilla en *assembler* mediante un repertorio de 35 instrucciones. Entre sus características principales se pueden mencionar:

- Memoria Flash de programa (1K x 14).
- Memoria EEPROM de datos (64 x 8).
- Memoria RAM (68 registros x 8).
- Un temporizador/contador (timer de 8 bits).
- Un divisor de frecuencia.
- Dos puertos de entrada-salida (puerto A de 5 pines y el puerto B de 8 pines).



Otras características que lo destacan: Manejo de interrupciones (de 4 fuentes), perro guardián (*watchdog*), bajo consumo, frecuencia de reloj hasta 20MHz (5 MIPS; la frecuencia de reloj interna es un cuarto de la externa), no posee conversores analógicos-digital ni digital-analógicos, pipe-line de 2 etapas (1 para búsqueda de instrucción y otra para la ejecución de la instrucción; los saltos ocupan un ciclo más), 4 tipos distintos de instrucciones: orientadas a byte, orientadas a bit, operación entre registros, y operaciones de salto.

El sistema: GPS→ PIC ←→ TL/GSM

La interconexión entre el GPS y el PIC se realiza asincrónicamente mediante la utilización de un puerto serial bajo el estándar RS-232C, y la conexión entre el PIC y el teléfono celular (utilizado como modem GSM), mediante una comunicación serial asincrónica TTL (figura 1).

El GPS utilizado es un equipo GARMIN SRVY II, el cual posee un puerto de conexión serial (RS232) configurable a 4880 Bps. EL teléfono utilizado como modem GSM es un Nokia 5070 (inicialmente un NOKIA 3220)[iv], que se conecta al PIC interceptando las líneas seriales luego de la disección de un cable de conexión Nokia CA-42 que permite la comunicación de este tipo de teléfonos con una computadora tipo PC, mediante un puerto USB.

Para facilitar el desarrollo del diseño se utilizó una pantalla LCD (2x16 compatible Hitachi HD44780) como monitor de despliegue de información, lo que permite un mejor control sobre las actividades del sistema.

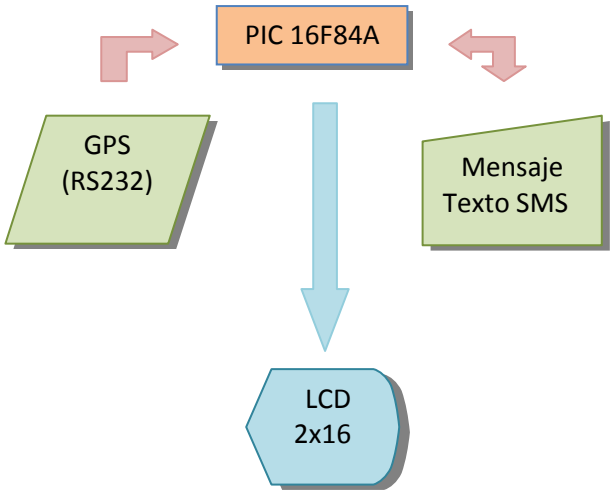


Figura 1.- Diagrama del Sistema

El sistema GPS

Fue desarrollado por el Departamento de Defensa de los EEUU, bajo el nombre *NAVTAR GPS (NAVigation Satellite Time And Ranging – Global Positioning System)*, a partir de 1973. Está compuesto por una constelación de 24 satélites situado sobre órbitas inclinadas de 55° respecto al ecuador terrestre, cada órbita contiene 4 satélites (3 operativos y 1 en reserva). El periodo de rotación de dichas plataformas espaciales es de 12 horas. La constelación satelital está pensada de manera que desde cualquier punto del planeta pueda recibirse la señal concurrente de al menos seis de ellos; sin embargo las especificaciones generales del sistema suponen una cobertura de 4 a 8 satélites con elevación superior a 15°; límite sobre el horizonte visible que se considera apto para una buena recepción de la señal. Cada satélite está equipado con un ordenador, un sistema de comunicación de muy alta frecuencia, y 4 relojes atómicos (dos de Cesio y dos de Rubidio), con un retardo de sincronización del orden de la milmillonésima de segundo. El control desde tierra se efectúa mediante un sistema llamado OCS *(Operational Control Station*), compuesto por una emisora principal MCS *(Master Control Station)* y una serie de estaciones de control dispuesta a lo largo del ecuador. La MCS coordina la sincronización de los relojes, del diagnostico y control de las órbitas; mediante el monitoreo constante de las efemérides (información sobre las órbitas exactas), almanaque (subconjunto de efemérides y parámetros de reloj), etc.

Los satélites trasmiten la información mediante modulación de espectro difuso *(Spread Spectrum Modulation)*, lo que dificultad al extremo la intersección e interferencia de la señal, ya que la información se confunde con el ruido electromagnético de fondo.

El sistema original consideraba dos sistemas de posicionamiento superpuesto, un *Servicio de Precisión Standard* (SPS) para uso civil, y un Servicio de Posición Precisa (PPS) para uso militar u oficial. Esta intrincada seguridad se implementó a partir de la generación en cada satélite de una frecuencia fundamental de 10,23 MHz, de la cual se derivan dos portadoras básicas L1 (1575,42 MHz; frecuencia base por 154) y L2 (1227,60 MHz; frecuencia base por 120). Para el cálculo de la ubicación cada satélite envía un par de códigos PRN *(Pseudo-Random Noise)* con una frecuencia de reloj de 1023 MHZ, que se repite cada milésima de segundo. El primero esta modulado sobre L1 y se denomina *C/A-Code (Coarse/Acquisition-Code)*, el segundo modulado simultáneamente sobre L1 y L2, denominado *P-Code (Precision-Code)*. El *C/A-Code* se aplica al SPS, y el *P-Code* al PPS.

Los receptores GPS de uso comercial están diseñados para trabajar con la frecuencia L1, portadora de la SPS diluida en precisión por la aplicación del *C/A-Code*. Actualmente dicho código no se aplicada sobre el SPS, lo que permite resolver posiciones dentro de un círculo de tres metros de diámetros. Para el cálculo de la posición sobre la superficie terrestre, básicamente se utiliza el tiempo que demora una señal desde el emisor al receptor, conocida la posición que ocupa el satélite en el espacio respecto al centro de masa de la tierra, unas trasformaciones trigonométricas en el receptor permiten calcular la ubicación. Repitiendo este procedimiento para varios satélites (mínimo tres) se obtiene la posición del receptor con unos pocos metros de incertezas. El intervalo de tiempo entre la emisión y la recepción se mide comparando el código C/A enviado por el satélite con una copia del mismo realizada por el receptor, sincronizado con el

satélite. Se obtiene la distancia de separación entre el emisor y receptor, luego de una serie de correcciones, entre ellas: la dispersión y difracción estratosférica, el efecto Dopler-Fizeau debido al desplazamiento relativo, y las sugeridas por Einstein respecto a los efectos relativistas y gravitatorios. Junto con el PRN el satélite envía un mensaje de 1500 bits trasmitido a 50 Bps, donde se incluyen las efemérides, almanaques, y demás parámetros de corrección (unos 30 segundos que justifican la demora operativa del GPS luego del encendido).

Protocolo NMEA

Cada fabricante de GPS posee un formato propietario para la entrega de la información al usuario mediante un puerto de comunicación tipo serial (actualmente Bluetooth y USB), pero tradicionalmente mediante el protocolo RS-232C a 4800 Bps. Se ha estandarizado un formato de intercambio dentro de los entandares NMEA, el protocolo NMEA-0183; el cual consiste de una serie de información formateada, separadas en sentencias llamadas tramas. Entre ellas $GPXTE, $GPBWC, $GPVTG, $GPRMC, $GPRMB, $GPRxx, $GPWPL, $GPGLL, y $GPRMZ. Entre estas, la trama $GPRMC *(Recommended minimum specific GPS/Transit data)*, es la más completa referente a la posición geográfica.

El protocolo NMEA es utilizado por una diversidad de sistemas de posicionamiento y navegación entre ellos el LORAM, OMEGA Navigation receiver, GPS, etc. Por ello todas las tramas identifican al inicio el sistema al cual pertenecen, $GP (en caso del GPS) y seguidamente identifican el tipo de trama y luego el resto de los parámetros separado por comas. En la tabla I se detallan estos parámetros y se muestran ejemplos de la trama $GPRMC.

Tabla I.

La trama $GPRMC *(Recommended minimum specific GPS/Transit data)*

$GPRMC,220516,A,5133.82,N,00042.24,W,173.8,231.8,130694,004.2,W*70

;Name	Example Data	Description
;UTC of position fix	220516	Time Stamp
;Data status	A	validity - A-ok, V-invalid
;Latitude of fix	5133.82	current Latitude
;N or S	N	North/South
;Longitude of fix	00042.24	current Longitude
;E or W	W	East/West
;Speed over ground in knots	173.8	Speed in knots
;Track made good in degrees	True 231.8	True course
;UT date	130694	Date Stamp
;Magnetic variation degrees	004.2	Variation
;E or W	W	East/West
;Checksum	*70	checksum

Format:
$GPRMC,hhmmss.ss,A,llll.ll,a,yyyyy.yy,a,x.x,x.x,ddmmyy,x.x,a*hh

El protocolo NEMEA-0183 está compuesto de varias tramas separadas por un carácter CR (*Carriage Return*) y un LF (*Line Feed*). Lo que obliga a leer todos los encabezados y discriminar del conjunto la trama de nuestro interés, mediante la Identificación de los caracteres iníciales; en nuestro caso $GPRMC. Los parámetros contenidos en las tramas NEMEA-0183, no tienen longitud fijas y están separados por comas; en algunas condiciones pueden faltar ciertos campos pero las comas marcan los límites de éstos aunque no contengan información. Antes del CR de finalización de trama, se coloca un asterisco seguido de dos dígitos hexadecimales que representan un "*checksum*"; la OR exclusiva de 8 bits de todos los caracteres que conforma la trama (sin incluir los símbolos $ y *). El análisis de las comas en la trama $GPRMC, permite aislar las coordenadas geográficas y los otros parámetros pertenecientes a dicha trama (tabla I).

Captura de la trama $GPRMC

Mediante un Ring Buffer de 56 byte, se leen las distintas tramas pertenecientes a protocolo NMEA-0183 suministrados por el GPS por el puerto serial RS232. Leyendo los encabezados de las distintas tramas se espera por el envío de la $GPRMC (o la que nos interese de acuerdo a la aplicación desarrollada), procediendo a capturar el resto de la trama, cargándola en el ring buffer para el análisis de las comas y la utilización de los parámetros geográficos (Figura 2). Considerando un uso futuro distinto del sistema, se desarrollaron (o adaptaron) las siguientes módulos o subrutinas:

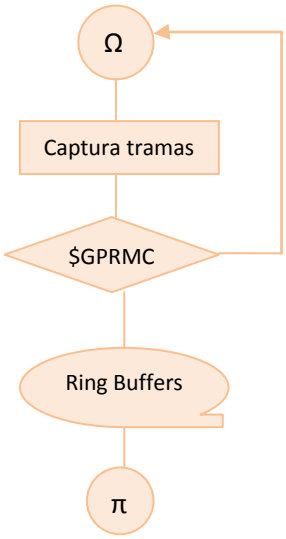


Figura 2

- **delay.inc** estados de espera
- **serial_232** comunicación RS232
- **captura_nmea_gprmc** captura la trama $GPRMC
- **parametro.inc|** constantes y registros

El Ring Buffer

A partir del inicio de los 68 registros de memoria RAM en el mapa de memoria del PIC 16F84A, posición 0Ch, se reservan 56 bytes (registros RAM de 8 bit c/u) para utilizarlos como buffers en la captura de datos por el puerto seria; tanto en la comunicación con el GPS como para el teléfono celular GSM. El espacio de memoria RAM se reserva mediante la instrucción **res 56**, que selecciona 56 bytes del mapa de memoria del PIC, a partir de la posición especificada por la directiva **ORG 0x0C**. Como apoyo didáctico esta instrucción se desnaturaliza para como **dato1 res 1**, para asignar nombres individuales a cada uno de los componentes de la trama (ver **parametros.inc**). El microprocesador considera los bytes consecutivos en la memoria física, como byte individuales, sobre estas individualidades se analiza el formato de la trama utilizando la subrutina **captura_nmea_gprmc.inc**, que ubica las comas y separa los componentes geográficos y temporales. Se lee la trama a partir de un espacio después de la primera coma, la función

coma_count detecta la coma (0x2C) e inicializa el contador que es luego incrementado con las subsiguientes comas por la función **coma_loop**[v]. En la subrutina **extrae_datos_de_nemea.inc**, la macro **go_EE** mediante el uso de una bandera permite almacenar en la EEPROM los datos de nuestro interés; se almacena en la EEPROM la posición geográfica al instante de la consulta SMS, para no enviar una posición temporalmente distinta al de la llegada del mensaje de texto considerando la velocidad o el desplazamiento del vehículo o plataforma donde se instale el ingenio; paralelamente esta información es desplegada en la pantalla LCD.

Temporización

Los retardos temporales necesarios para que el PIC opere con data serial, pueden lograrse mediante hardware (PIC 16F887) o por medio de ciclos repetitivos basados en software (PIC 16F84). Los retardos de tiempo basado en software se realizan mediante un bucle ó incrementado ó disminuyendo un contador hasta que este llega al retardo buscado. Un ciclo de máquina es el tiempo usado por el PIC para realizar sus operaciones internas y equivale a cuatro ciclos de reloj u oscilador. El ciclo de máquina es el tiempo que un micro controlador tarda en ejecutar una instrucción, en el caso del PIC 16F84 (y todos los de la gama media de Microchip) se requieren 4 ciclos de reloj para que se ejecute una instrucción sencilla que no implique salto. Si usamos un cristal de 4MHz cada instrucción sencilla se ejecuta en 1µs; las instrucciones que implican saltos y modifican el contador de programa (PC), requieren dos ciclos de máquina (se ejecutan en 2 µs).

El protocolo RS-232C

En el presente proyecto se emplea la comunicación asíncrona, la cual no requiere que el trasmisor este sincronizado con el receptor para enviar datos (maneja datos en tiempo real, con un intervalo de longitud arbitraria entre caracteres sucesivos). La comunicación de cada carácter está precedida de un bit de inicio; un cero lógico con duración igual al tiempo de un bit. Cuando la comunicación está inactiva, la línea se encuentra en el estado lógico 1, cuando la línea cambia de marca a espacio se interpreta cómo bit de inicio, después de este bit se reciben los bits de datos con tiempo de duración de bit cada uno. Cuando el último bit de datos ha sido enviado, el trasmisor pasa a nivel de marca (1-lógico) durante el tiempo de bit. Este bit es llamado bit de paro, e indica que todos los bits de data han sido enviados y la transmisión del carácter se ha completado. Si el detector no detecta el bit de marca final, esto implica un error de transmisión.

Reglas de transmisión asíncrona

La transmisión asíncrona que vamos a ver es la definida por la norma RS232:

a. Cuando no se envían datos por la línea, ésta se mantiene en estado alto.
b. Cuando se desea transmitir un carácter, se envía primero un bit de inicio que pone la línea a estado bajo (0) durante el tiempo de un bit.
c. Durante la transmisión, si la línea está a nivel bajo, se envía un 0 y si está a nivel alto se envía un 1.
d. A continuación se envían todos los bits del mensaje a transmitir con los intervalos que marca el reloj de transmisión. Por convenio se transmiten entre 5 y 8 bits.
e. Se envía primero el bit menos significativo, siendo el más significativo el último en enviarse.
f. A continuación del último bit del mensaje se envía el bit (o los bits) del final que hace que la línea se ponga a 1 por lo menos durante el tiempo mínimo de un bit. Estos bits pueden ser un bit de paridad para detectar errores y el bit o bits de stop, que indican el fin de la transmisión de un carácter.

Los datos codificados por esta regla, pueden ser recibidos siguiendo los pasos siguientes:

a. Esperar la transición 1 a 0 en la señal recibida.
b. Activar el reloj con una frecuencia igual a la del transmisor.
c. Muestrear la señal recibida al ritmo de ese reloj para formar el mensaje.
d. Leer un bit más de la línea y comprobar si es 1 para confirmar que no ha habido error en la sincronización.

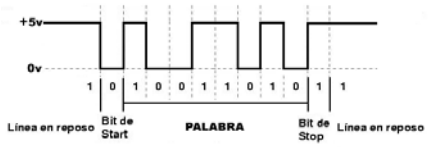
La transición de alto a bajo de la línea de trasmisor activa al receptor y este genera un conteo de tiempo de tal manera que realiza una lectura de la línea: medio bit después del evento (1/3 es aceptable). Si la lectura es realizada en un estado alto, se asume que la transmisión ocurrida fue generada por ruido en la línea. Si por el contrario, la lectura se hace durante un estado bajo, la transmisión se considera como válida y empieza a realizar lecturas secuenciales a intervalo de un bit (en la medianía del ancho de bit) hasta completar el carácter trasmitido.

El elemento clave es detectar el bit de arranque, bien sea a través de interrupciones, o bien a través de lecturas frecuentes de la línea que contiene los datos. En ambos casos, lo recomendable es que después de detectar el bit de arranque, la lectura de los bits restantes se realice en la mitad del bit, con un error permitido del 3% a 4% del tiempo de bit (este proyecto se consideró prudente hacer el muestreo a 1/3 del tiempo de bit).

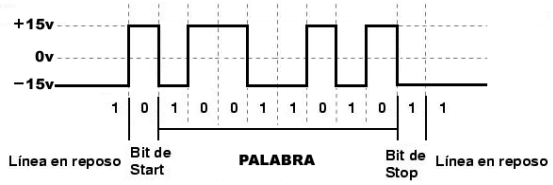
En los circuitos digitales se maneja la trasmisión serial "RS-232" en niveles lógicos TTL (0 a 5V), pero cuando la distancia aumenta se deben aumentar los niveles de voltaje.

La norma RS-232 (de 1962) especifica:

- Un uno lógico es un voltaje comprendido entre -5V y -15V en el trasmisor, y entre -3V y -25V en el receptor.
- Uno cero lógico es un voltaje comprendido entre 5V y 15V en el trasmisor, y entre 3V y 25 V en el receptor.



Transmisión en TTL del dato binario **01011001**. La línea en reposo está a nivel lógico alto (+5 voltios).

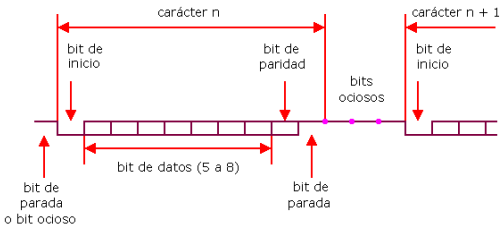


Transmisión en RS232 del dato binario **01011001**. La línea en reposo está a nivel lógico alto (-15 voltios).

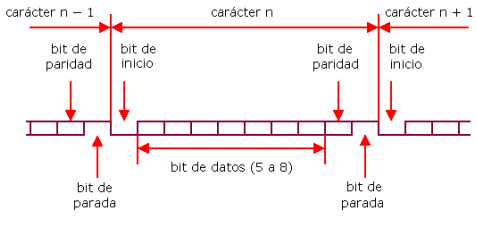
Velocidad de transmisión[vi]

En la transmisión asíncrona por cada carácter se envía al menos 1 bit de inicio y 1 bit de parada así como opcionalmente 1 bit de paridad. Esta es la razón de que los baudios no se correspondan con el número de bits de datos que son transmitidos.

Al final de cada carácter, la línea va a 1 en el bit de parada y permanece en ese estado durante un número arbitrario de bits ociosos. El inicio del nuevo carácter estará definido por la transición a cero del bit de inicio.



Transmisión asincrónica con velocidad menor que la máxima posible



Transmisión asincrónica con la velocidad máxima posible

Determinar cuántos bits de datos y caracteres son transmitidos de manera asíncrona en 1 segundo si se transmite a una velocidad de 4800 baudios con 1 bit de inicio, 2 bits de parada, 1 bit de paridad y 6 bits de datos por carácter:
Para transmitir un carácter se necesitará: 1 bit inicio + 6 bits datos + 1 bit paridad + 2 bits parada =10 bits. Como la velocidad de transmisión es 4800 baudios y cada carácter consume 10 bits, se transmitirán 480 caracteres por segundo (4800/10). Como cada carácter posee 6 bits de datos serán transmitidos 480 * 6 = 2880 bits de datos por segundo.

El GPS (Garmin SRVYII) configurado a 4800 Bps, emite una trama $GPRMC. Calculemos el tiempo que demora el PIC 16F84 con un cristal de 4 MHz en leer un Rin Buffers de 56 byte (sin considerar bit de parada):

A 4800 Bps el ancho de bit es igual período de la trasmisión; 1/4800 segundo = 208,33µs. Concluimos: En una trasmisión a 4800 Bps cada bit tiene un ancho de 208,33µs. Los 56 bits del Rin Buffers son leídos por el PIC en apenas 0,012s.

Rutinas del Puerto Serial

El protocolo RS-232C exige un control preciso del los retardos temporales (*delay*) y de las técnicas de captura y recepción de información. El PIC 16F84 no posee una solución hardware (USART) para el manejo de la data serial, por lo que exige un tratamiento por software a todo el proceso de comunicación serial. El primer paso es disponer de unas subrutinas temporales adaptadas a la velocidad de comunicación (4800 Bps), sabemos del ejemplo anterior que esto implica cálculo del orden de 208µs, el siguiente código[vii] permite alcanzar tal precisión:

```
;*******************************************************************************
;                          Rutina generadora de retardos
;*******************************************************************************
serialDelayThird
                          movlw   Veloc_Bps/3         ;
                          movwf   serialDelayCount    ; solamente cuenta
                          goto    serialDelayLoop     ; 1/3 del tiempo de bit
serialDelay                                           ; 4 ciclos de maquina + serialDelayLoop
                          movlw   Veloc_Bps           ;
                          movwf   serialDelayCount    ; contador decreciente para el DELAY_TIME
                          nop                          ; complemento de tiempo
serialDelayLoop                                       ;
                          decfsz  serialDelayCount,F  ; decrementa el contador
                          goto    serialDelayLoop     ; realiza el ciclo de espera
                          return                      ; regresa de la llamada
;*******************************************************************************
```

SerialDelayThird nos permite calcular el tiempo correspondiente a un tercio del tiempo de bit: para 4800 Bps la constante Veloc_Bps toma el valor de d'64' (decimal 64). Luego serialDelayCount toma el valor de la parte entera de 64/3. La rutina serialDelayLoop decrementará la constante serialDelayCount cíclicamente una unidad hasta llegar a cero, cuando de acuerdo a la instrucción decfsz (decremento y salto cuando sea cero) regresar a la rutina inicial desde donde se hizo la llamada. Todo esto en 69 ciclos de máquina (1 + 1 +2 +21 + 42 +2) o sea 69µs, que es el valor necesario para ubicarnos a 1/3 del ancho de bit, y proceder a la captura de datos; la rutina serialDelay realiza la operación para el tiempo completo de bit: 208µs.

Diagrama de flujo de la rutina de captura de data

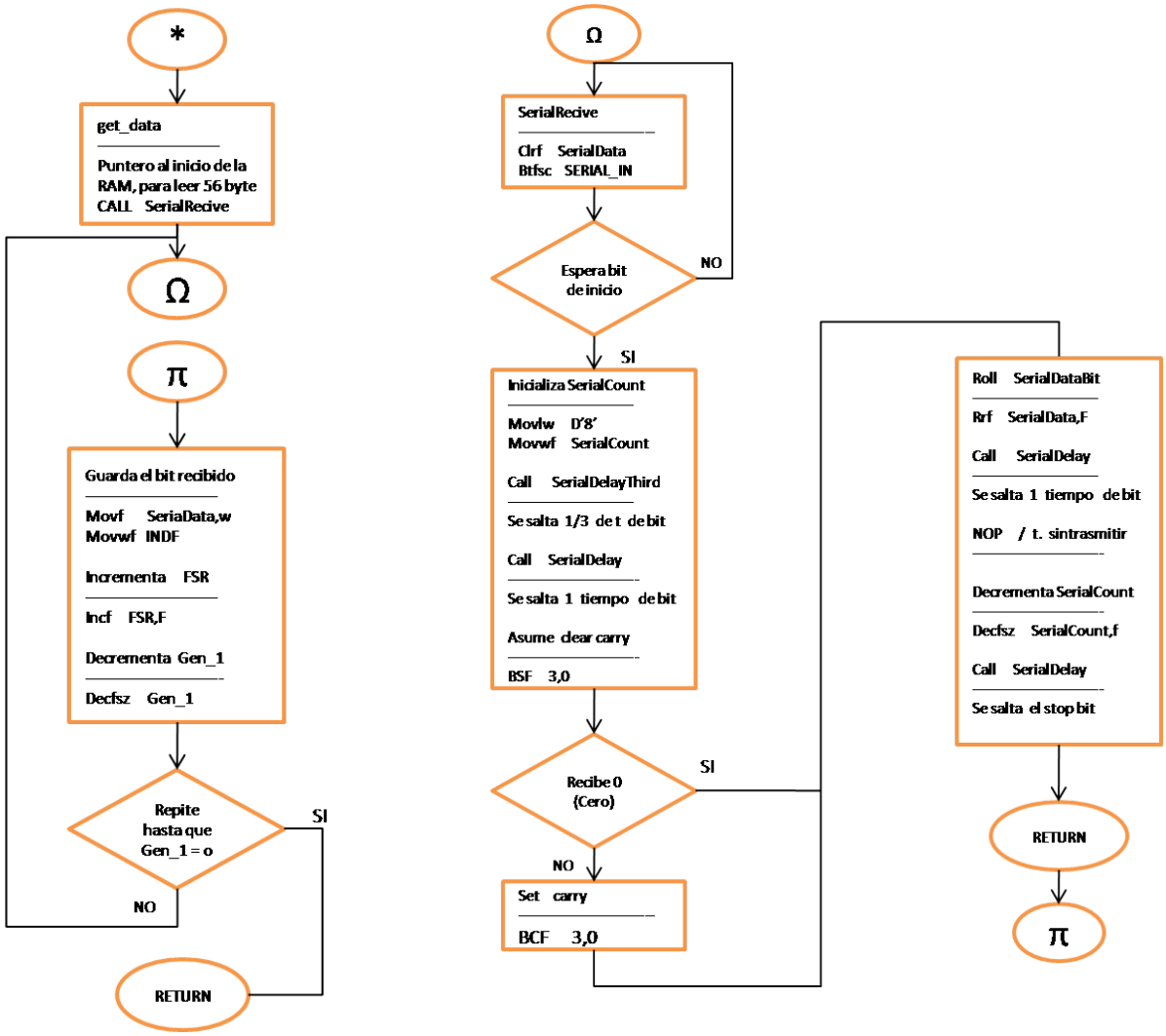
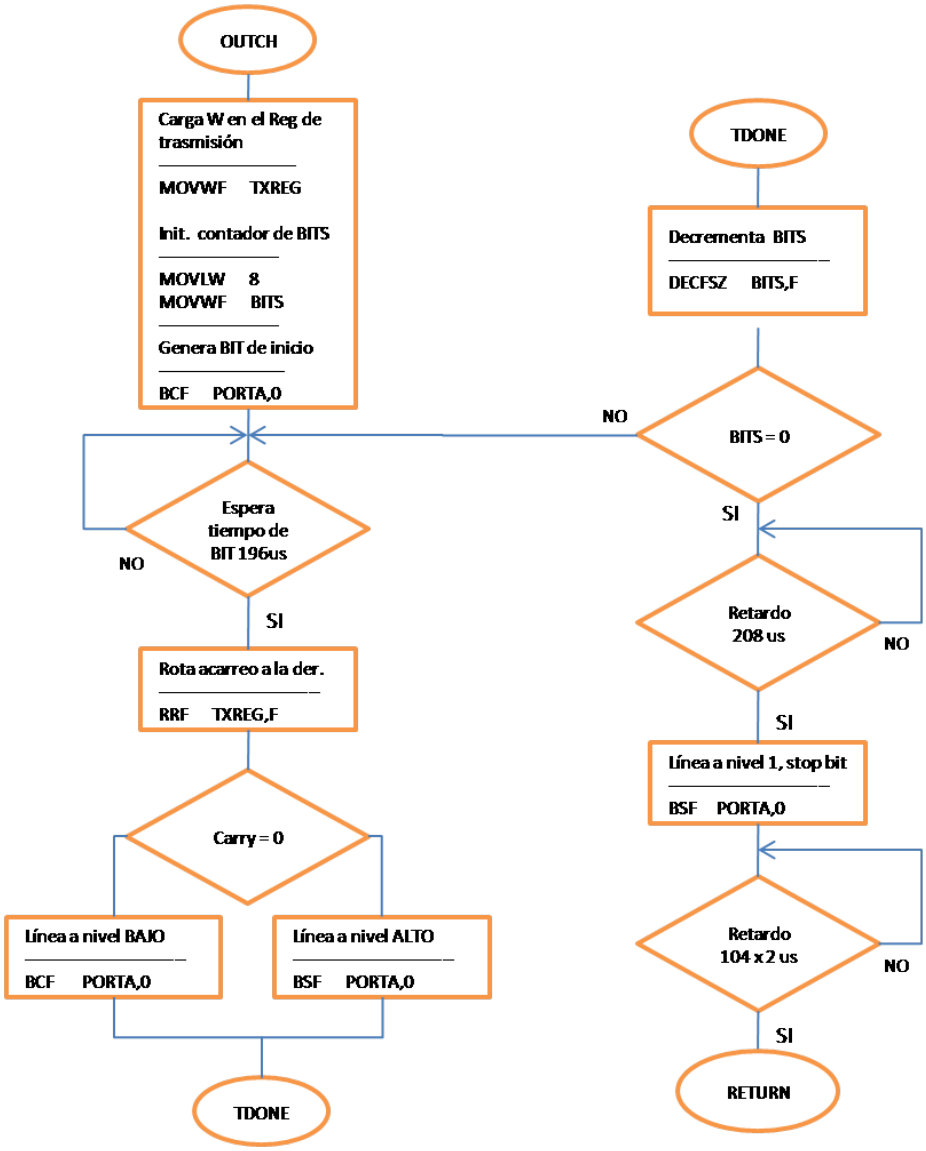


Diagrama de flujo de la rutina de envío de data

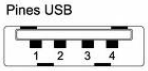


El Modem GSM

Ante la dificultad de adquirir un Modem GSM (tipo OEM), fue necesario utilizar un teléfono celular suscrito a proveedores locales (Digitel y Movistar). En la fase inicial de desarrollo se utilizó en Nokia 3220, que posee entre otras características: tecnología GSM, conector F-Bus, conector estándar Nokia de 14 pines *(pop-port)*, y responde efectivamente al conjunto de comandos AT-Nokia cuando se opera desde el Hyper termninal de Windows.XP. La configuración del teléfono para ser utilizado como módems externo GSM (accesorio de datos) es una de las dificultades cuando se utilizan teléfonos Nokia. Esta operación se activa por el envío de una extensa cadena de bits (unos 900), de cuyo formato no se dispuso. Sin embargo, mediante el uso del osciloscopio (un Textronik 1002) fue posible notar que el propio cable de conexión Nokia CA-42, enviaba la cadena de configuración al teléfono; desplegando en la pantalla del teléfono, el mensaje: "Accesorio de datos conectado".

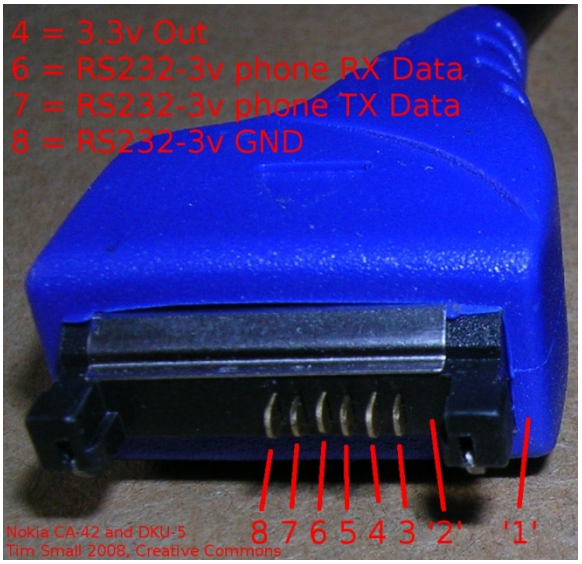
Cable CA-42

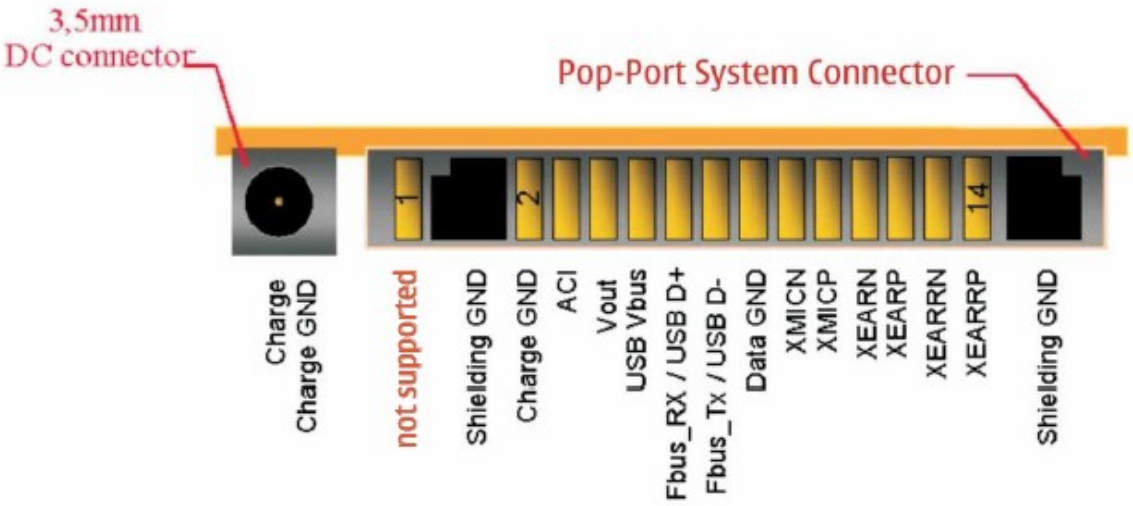
Es el cable utilizado para interconectar al PC los teléfonos Nokia con conectores estándares de 14 pines (también conocido como DKU-5), de acuerdo a la bibliografía consultada es un dispositivo de complejo esquema. Posee en uno de sus extremos un conector Nokia 14 pines, y en el otro extremo un conector USB; que esta internamente conectado a un circuito integrado ubicado debajo del encapsulado plástico (USB → RS232, Bridge Controller; OTI-6858). La conexión interna entre el circuito integrado y el conector de 14 pines se realiza mediante 5 líneas de colore negro, verde, blanco, rojo, y azul. **En resumen:** el cable azul se corresponde con la salida de datos del teléfono y la entrada para el PIC, y el cable blanco con la entrada de datos del teléfono y la salida del PIC (tabla II); y la señal ACI (cable rojo) debe desconectarse por un instante para poner el teléfono en modo de "Accesorio de datos conectado".

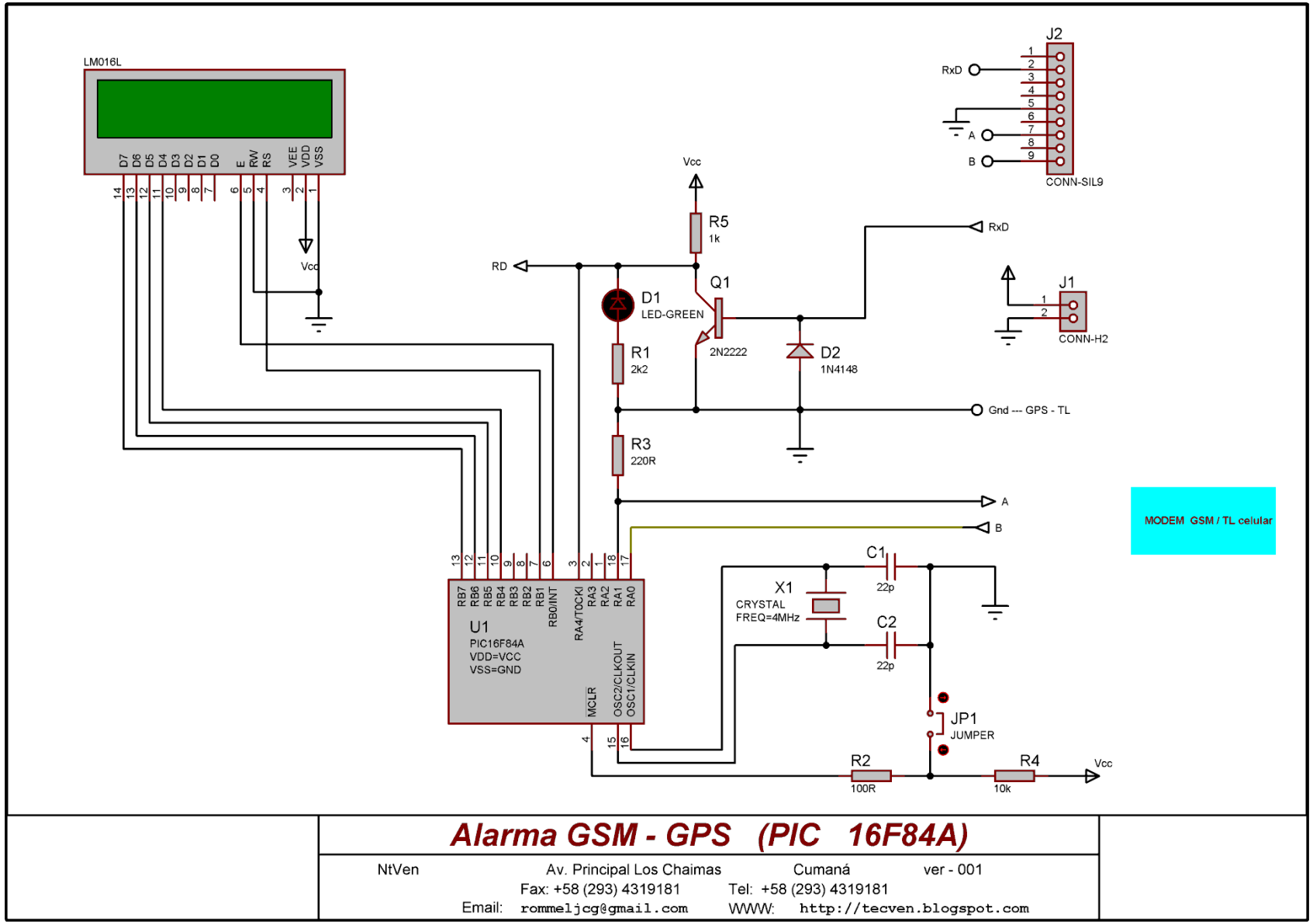
PIC 16F84A	Descripción	Pot–port		USB	Descripción
Vss/Pin 5	Tierra común	Pin **8** (USB GND)		Pin 4	GND
	Salida de datos teléfono	Pin **7** (FBus Tx D-)		Pin 2	FBus Tx/USB D-
Pto A1/pin 17 Salida datos	Entrada de datos teléfono	Pin **6** (FBus Rx D+)		Pin 3	FBus Rx/USB D+
	ACI - Insertion & removal detection	Pin **3** (ACI)		Pin 1	USB Vcc + 5v
Pto A0/pin 18 Entrada datos	Salida TTL de datos del teléfono / Entrada PIC	Pin **4** (USB Vbus)			
	Pot-port				USB

Nokia Pop Port 3100, 3200, 3300, 5100, 6100, 6610, 6800, 7210, 7250

1. V. In
2. GND
3. ACI (Auto Connect Ignition)
4. V. Out
5. USBPowDet (USB Power Detection)
6. FBUS Rx
7. FBUS Tx
8. GND
9. Microphone (-)
10. Microphone (+)
11. Earphone L (-)
12. Earphone L (+)
13. Earphone R (-)
14. Earphone R (+)







Alarma GSM - GPS (PIC 16F84A)

NtVen Av. Principal Los Chaimas Cumaná ver - 001
Fax: +58 (293) 4319181 Tel: +58 (293) 4319181
Email: romneljcg@gmail.com WWW: http://tecven.blogspot.com

[i] Projet Boards, KandH (model GL-36)
[ii] Stripboard Creativa, EZSBC1170
[iii] Angulo Usategui, José. 2003. **MICROCONTROLADORES «PIC»**. Diseño práctico de aplicaciones. Primera parte: El PIC16F84. Lenguajes PBASIC y Ensamblador. Tercera edición.
[iv] El NOKIA 5070 presentó una estabilidad mayor en el conjunto del sistema con una línea Movistar. El Nokia 3220 operaba con una inestabilidad aleatoria, el servicio fue suministrado por Digitel, que posee para la fecha un servicio de mensaje de textos más eficiente.
[v] Adaptadas de las usada por Pierbassano Turrini (2003): http://www.qsl.net/iw2bc/disp_gps.html
[vi] Tomado de Sistmas Microprogramables de emiliotoboso@wanadoo.es
[vii] Adaptadas de las usada por Pierbassano Turrini (2003): http://www.qsl.net/iw2bc/disp_gps.html