Nombre de la asignatura: Laboratorio de programación de avanzada

Maestro en Ciencias: Moisés Agustín Martínez Hernández

Nombre de la práctica: GPS

Integrantes:
- Zúñiga Fragoso Diego Joel
- Manríquez Navarro Daniela del Carmen

Número de práctica: 9

Total de horas: 2 Hrs.

Objetivos

- Aprender a configurar pines de salida del microcontrolador como puerto serial para envío y recepción de datos.
- Configurar una PC con el propósito tener una comunicación bidireccional con el microcontrolador.
- Comunicar el microcontrolador con la PC por medio del protocolo RS-232 con la finalidad de controlar las funciones del microcontrolador.
- Configurar el microcontrolador para recibir los datos enviados por el GPS.

Descripción de la práctica

Recibir información del microcontrolador a la PC a través de comunicación serial, usando el protocolo RS-232. El microcontrolador recibirá los datos proporcionados por el GPS y ,mediante el código, se realizará la conversión para obtener las coordenadas en el formato aceptado por Google maps, para que al buscarlas en la plataforma de la ubicación correcta.

Marco teórico

Los modos de transmisión de datos se dividen en cuatro tipos:

- *Simplex*. Se dice a la transmisión que puede ocurrir en un sólo sentido, sea sólo para recibir o sólo para transmitir.

- *Half-duplex.* Se refiere a la transmisión que puede ocurrir en ambos sentidos pero no al mismo tiempo, en donde una ubicación puede ser un transmisor y un receptor, pero no los dos al mismo tiempo.
- *Full-duplex*. Se dice a la transmisión que puede ocurrir en ambos sentidos y al mismo tiempo.
- *Full/full-duplex.* Con este modo de transmisión es posible transmitir y recibir simultáneamente, pero no necesariamente entre las dos ubicaciones, es decir una estación puede transmitir a una segunda estación y recibir de una tercera estación al mismo tiempo.

La norma RS-232 es la más habitual en la comunicación serie. Básicamente comunica un equipo terminal de datos (DTE o Data Terminal Equipment) y el equipo de comunicación de datos (DCE o Data Communications Equipment). Las características eléctricas de la señal en esta norma establecen que la longitud máxima entre el DTE y el DCE no puede ser superior a 15 metros y la velocidad máxima de transmisión es de 20,000 bps. Los noveles lógicos no son compatibles TTL (Transistor-Transistor Logic), deben situarse dentro de los siguientes rangos: 1 lógico entre -3V y -15V y 0 lógico entre +3V y +15V. Se utilizan conectores de 25 patillas (DB25) o de 9 patillas (DB9) siendo asignado el conector macho al DTE y el conector hembra al DCE.
Para la comunicación full dúplex, se debe de conectar un mínimo número de señales, *TXD y RXD* así como *GND.* Los microcontroladores utilizan señal TTL por lo que se debe utilizar un conversor de nivel a *RS232*, como el *MAX232*.
En la actualidad los puertos serie en las PC están prácticamente desaparecidos. Como solución de pueden utilizar cables de conversión *SERIE-USB*. La estructura de un dato en comunicación serial se muestra a continuación, Figura 1.

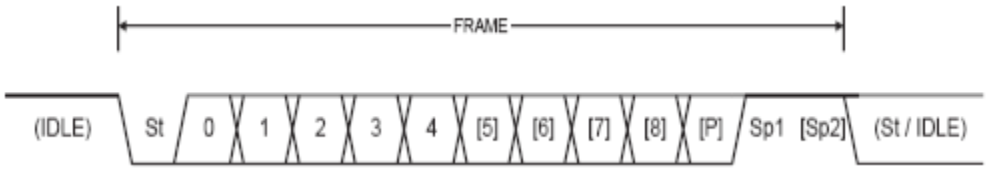


Figura 1. Estructura de un dato en comunicación serial.

Donde:
- St: Bit de Arranque siempre es activo en bajo.
- N: Número de datos de 0 a 8 bits.
- P: Bit de paridad.
- Sp: Bits de parada puede ser 1 o 2. Siempre son activos en altos.

La señal permanece en un nivel lógico alto mientras no realiza ninguna trasferencia de datos. Para empezar a transmitir datos el transmisor coloca la línea en nivel bajo durante el tiempo de in bit, este se llama bit de arranque, a continuación empieza a transmitir con el mismo intervalo de tiempo los bits de datos, que pueden ser de 7 u 8 bits, comenzando por los bits menos

significativos y terminando con los más significativos. Al final de la transmisión se envía un bit de paridad, si estuviera activa esta opción, y por último los bits de parada, que pueden ser 1 o 2, después de esto la línea vuelve a un estado lógico alto, y el transmisor está listo para enviar el siguiente dato.

GPS

Módulo GPS basado en el chip NEO-6M de Ublox . Este módulo utiliza la tecnología de Ublox para brindar una buena información de posicionamiento. También incluye una antena GPS de 25 x 25 mm, con conector UFL. Posee una pequeña batería y una memoria eeprom que permite guardar los últimos datos de posicionamiento y así lograr una detección más rápida.
La antena se conecta al módulo a través de un cable UFL, lo que brinda flexibilidad al momento de posicionarlo. Es importante que la antena siempre apunte hacia el cielo para obtener el mejor rendimiento. Útil para aplicaciones de seguimiento con automóviles y otras aplicaciones móviles.

El módulo dispone de una salida TTL serial. Posee cuatro pines: TX, RX, VCC y GND. Se puede descargar y emplear el software u-center para configurar el GPS y cambiar su modo de funcionamiento.

El voltaje de operación del módulo es de 3.3V - 5V, pero la lógica del puerto serial (Tx, Rx) es de 3.3V. Si se va a utilizar el pin Rx con lógica de 5V, se debe emplear un divisor de tensión para reducirla a 3.3V. Por lo general solo se usa el pin Tx ya que para conocer el posicionamiento solo se requiere leer las sentencias NMEA que genera el chip.

Cuando el módulo se alimenta por primera vez o lleva tiempo sin ser encendido, es recomendable llevarlo a un lugar despejado para que pueda detectar los canales GPS y actualizar su fecha/hora y posición en el menor tiempo posible.

El módulo dispone de un LED de indicación. Mientras no se haya obtenido el posicionamiento y la fecha/hora, permanecerá apagado. Al momento de obtenerse la información, comenzará a parpadear.

EQUIPO Y MATERIALES

- Software para programar microcontrolador.
- Software de simulación.
- Microcontrolador.
- Circuito integrado MAX232.
- Cable de conversión SERIE-USB.
- Resistencia variable.
- Módulo GPS.

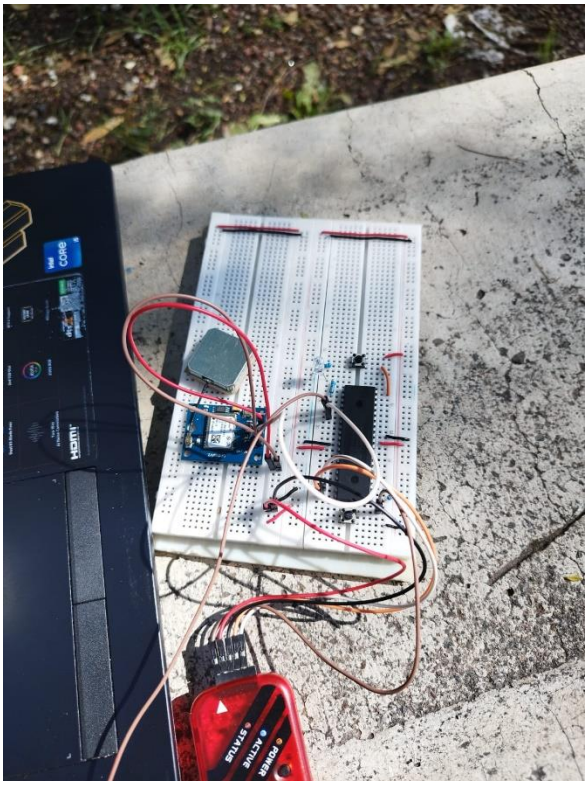
DESARROLLO DE LA PRÁCTICA

1. Configurar el microcontrolador considerando:
   a) Habilitar el ADC con la mayor resolución posible del microcontrolador.

b) Habilitar los puertos de comunicación serial del microcontrolador y configurar sus parámetros para recibir los datos proporcionados por el GPS.
2. Diseñar un circuito para comunicar el microcontrolador con la PC:
3. Desarrollar un programa para el microcontrolador el cuál recibirá los datos proporcionados por el módulo GPS, los convertirá y mostrará en la PC en formato de coordenadas para ingresar en Google maps.
4. Una vez que la simulación sea la correcta, desarrollar el armado físico de circuito.

Resultados de la práctica

Circuito armando



Código explicado

#include <18f4550.h> // Libreria del Microcontrolador

//#device adc = 10 // Resolucion del ADC en bits

#fuses INTRC, NOWDT, NOPROTECT, NOLVP, CPUDIV1, PLL1 // Fusibles (Configuraciones del microcontrolador)

#use delay(clock = 8M)

```
// rs232(rcp = (Pin receptor), xmit = (pin transmisor), baud = (Velocidad de transferencia), bits = 8, parity = n)

#use rs232(rcv = pin_c7, xmit = pin_c6, baud = 9600, bits = 8, parity = n)

#include <stdlib.h>


#define BOTON PIN_D1

#define LED PIN_D3 // Informa si hay una ubicacion nueva para imprimir

#define BufferMAX 50


char Buffer[BufferMAX], BufferLat[10], BufferLon[11], DLat, DLon;

float Latitud = 0.0, Longitud = 0.0;


int flag = 0;


// Solo hay conflicto si se utiliza en transmisor en la interrupcion del receptor (Se desincroniza).

// Se puede transmitir datos pero fuera de la interrupcion de recepcion


// Interrupcion para recibir datos de GPS

#int_rda

void ReceptGPS()

{

  int i;
```

```
  char signal = getch();


  if(signal == '$')
  {
    for(i = 0; i < BufferMAX; i++)
    {
      Buffer[i] = getch();
    }


    if(Buffer[2] == 'R' && Buffer[3] == 'M' && Buffer[4] == 'C' && i >= 43)      // Buffer tiene una lectura de ubicacion
    {
      if(Buffer[16] == 'A' && !flag)          // La lectura es valida
      {
        output_high(LED);


        // Obtenemos string de ubicacion


        DLat = Buffer[29];   DLon = Buffer[43];


        for(i = 0; i < 11; i++)
        {
```

```
            BufferLon[i] = Buffer[31 + i];


            if(i < 10)
               BufferLat[i] = Buffer[18 + i];
          }


          flag = 1; // La ubicacion fue actualizada
        }
        else {}                    // La lectura no es valida
      }
    }


}


void main()
{
  float auxf;    int auxi;   char auxc[3];


  enable_interrupts(GLOBAL);      enable_interrupts(int_rda);


  printf("Bienvenido, presione el boton para obtener la ubicacion\r\n\r\n");
```

```
while(true)
{
  // Bandera para actualizar ubicacion

  if(input(BOTON)) // Quiero saber ubicacion
  {
    while(input(BOTON));

    if(flag) // Si hay una ubicacion para imprimir
    {
      // Convertimos las coordenadas a numeros y la imprimimos

      auxc[1] = BufferLat[0];    auxc[2] = BufferLat[1];
      auxi = atoi(&auxc[1]);        auxf = atoe(&BufferLat[2]);

      Latitud = auxi + (auxf / 60.0);

      auxc[0] = BufferLon[0];    auxc[1] = BufferLon[1];    auxc[2] = BufferLon[2];
      auxi = atoi(auxc);        auxf = atoe(&BufferLon[3]);

      Longitud = auxi + (auxf / 60.0);
```

```
        // Imprime la ubicacion

        printf("Coordenadas para Gmaps:\r\n%.5f %c  %.5f %c\r\n\r\n", Latitud, DLat, Longitud,
DLon);



        output_low(LED);

        flag = 0; // Se espera una actualizacion de ubicacion

      }

      else

        printf("No hay ubicacion disponible\r\n\r\n");

    }

  }

}
```

Conclusiones de la práctica

Joel Zúñiga:

Antes de hacer esta práctica no conocía el funcionamiento del módulo GPS, y me impresiono mucho la gran precisión de este, que junto a su versátil funcionamiento con el protocolo RS232, lo convierten en un módulo muy útil y fácil de utilizar. Otra cosa que aprendí mejor a utilizar en esta practica fueron las interrupciones, las cuales establecen prioridades dentro del código, ejecutando los procesos mas prioritarios como el recibir los datos del GPS.

Daniela Manríquez:

En esta práctica aprendí como es que funciona el módulo GPS, como es que envía los datos y cuáles son los que dan la ubicación. También aprendí y reforze conocimiento de programación, al momento de separar los datos, convertirlos y mostrarlos para obtener las coordenadas.

Bibliografía

García E. (2008). Compilador C CCS y simulador PROTEUS para microcontroladores PIC: Alfa Omega.

Valez F. (2007). Microcontroladores: Fundamentos y aplicaciones con PIC. España: ALFAOMEGA.