República Bolivariana de Venezuela
Ministerio del Poder Popular para la Defensa
Universidad Nacional Experimental Politécnica
De la Fuerza Armada Nacional Bolivariana
Núcleo Aragua – Sede Maracay

Microprocesadores
Práctica N°9

PROCOLO DE COMUNICACIÓN I2C

Carlos Gutiérrez CI: 26.666.347
Manuel Pantoja CI: 26.248.682

Junio, 2019

HISTORIA DEL PROTOCOLO I2C

I2C significa Circuito Interintegrado (Por sus siglas en Inglés Inter-Integrated Circuit) es un protocolo de comunicación serial desarrollado por Phillips Semiconductors allá por la década de los 80s. El protocolo I2C toma e integra lo mejor de los protocolos SPI y UART. Con el protocolo I2C podemos tener a varios maestros controlando uno o múltiples esclavos. Esto puede ser de gran ayuda cuando se van a utilizar varios microcontroladores para almacenar un registro de datos hacia una sola memoria o cuando se va a mostrar información en una sola pantalla.

INFORMACIÓN DEL PROTOCOLO

El protocolo I2C utiliza sólo dos vías o cables de comunicación:
SDA – Serial Data. Es la vía de comunicación entre el maestro y el esclavo para enviarse información.
SCL – Serial Clock. Es la vía por donde viaja la señal de reloj.
I2C es un protocolo de comunicación serial.
Como podemos observar, el protocolo I2C envía información a través de una sola vía de comunicación. La información es enviada bit por bit de forma coordinada.
Al igual el protocolo SPI, el protocolo I2C trabaja de forma síncrona. Esto quiere decir que el envío de bits por la vía de comunicación SDA está sincronizado por una señal de reloj que comparten tanto el maestro como el esclavo a través de la vía SCL.

¿Cómo funciona el protocolo I2C?

Con el protocolo I2C la información viaja en mensajes. Los mensajes van divididos en tramas de datos. Cada mensaje lleva una trama con una dirección la cuál transporta la dirección binaria del esclavo al que va dirigido el mensaje, y una o más tramas que llevan la información del mensaje. También el mensaje contiene condiciones de inicio y paro, lectura y escritura de bits, y los bits ACK y NACK. Todo esto va entre cada sección de datos.

Imagen n°1 Estructura del dato

Condición de Inicio – Start: La vía SDA cambia de un nivel de voltaje Alto a un nivel de voltaje Bajo, antes de que el canal SCL cambie de Alto a nivel Bajo.
Condición de Paro – Stop: La vía SDA ahora cambia de un nivel de voltaje Bajo a Alto, después de que la vía SCL cambia de Bajo a Alto.
Trama de Dirección – Addres Frame: Es una secuencia única que va de los 7 a los 10 bits. Esta sección (Frame) se envía a cada Esclavo, y va a identificar al Esclavo con el que el Maestro se quiere comunicar.
Bit para Lectura/Escritura A – Read/Write Bit A: Es un bit de información enviado a los Esclavos. Por medio de este bit el Maestro indica si le va enviar información al Esclavo (Nivel Bajo de voltaje = Escritura), o si el Maestro quiere solicitarle información al Esclavo (Nivel Alto de Voltaje = Lectura).
Bit ACK/NACK : Después de cada sección (Frame) de información enviada en un mensaje, podemos notar que lleva un bit acknowledge/no-acknowledge (reconocido/no-reconocido). Esto ayuda a identificar si la información fue enviada correctamente. En seguida de que se envía un Frame, si este fue recibido con éxito, se retorna un bit ACK al remitente. Si la información no fue recibida con éxito, se retorna un bit NACK.

EJEMPLO DE COMUNICACIÓN I2C

Para este ejemplo de comunicación se emplea un PIC16F877, 3 memorias EEPROM 24AA00, un teclado matricial y el debugger de comunicación I2C del proteus solo para observar la trama de datos. El montaje es el siguiente:

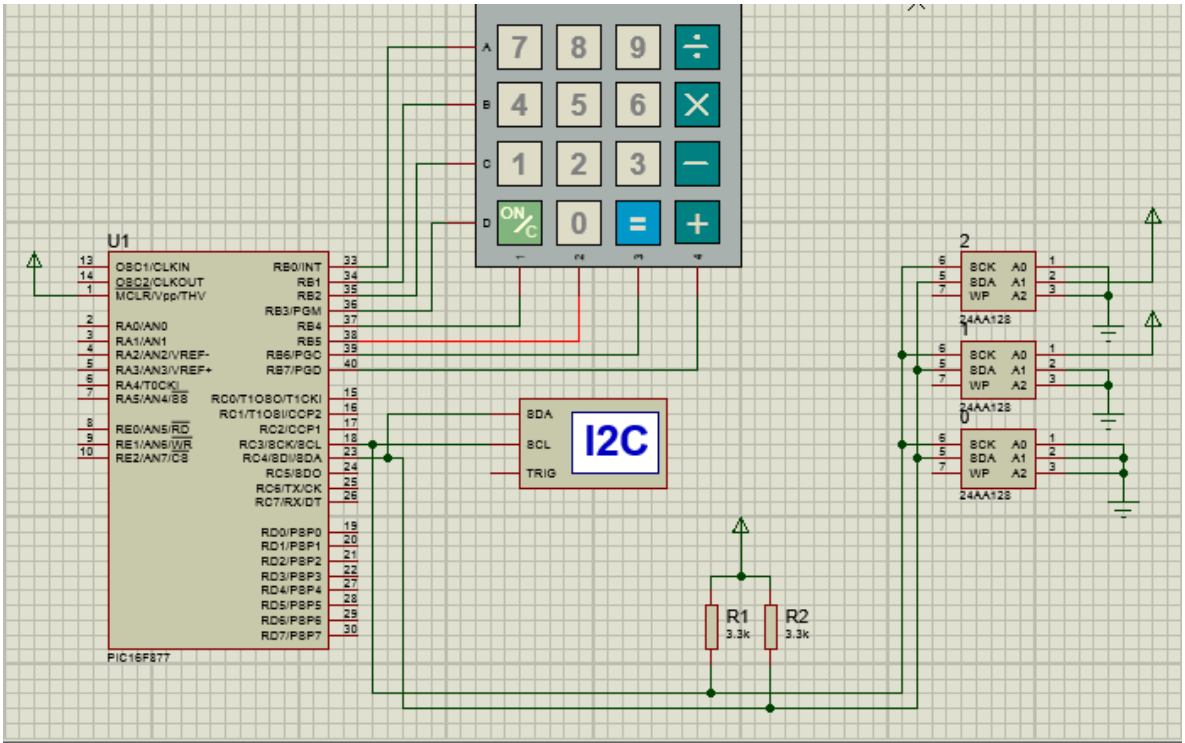

Imagen n°2 Montaje proteus

Como se puede observar en las líneas de dirección de las memorias (A0, A1, A2) cada uno tiene una configuración distinta, esto para asignarle a cada una dirección diferente para poder seleccionarlos a la hora de establecer la comunicación serial, como se observa en la imagen de la trama de datos. Para el código del micro se designó la dirección de cada memoria a una tecla distinta, la de división, multiplicación y resta, luego se espera por la entrada de datos en este caso un número del 1 al 9 e inicia la comunicación, mandando el Bit de start, la dirección antes seleccionada, la dirección de registro de la memoria que en este caso son 2 bytes, y luego el dato que es el número selecionado con el teclado para luego mandar el Bit de parada.

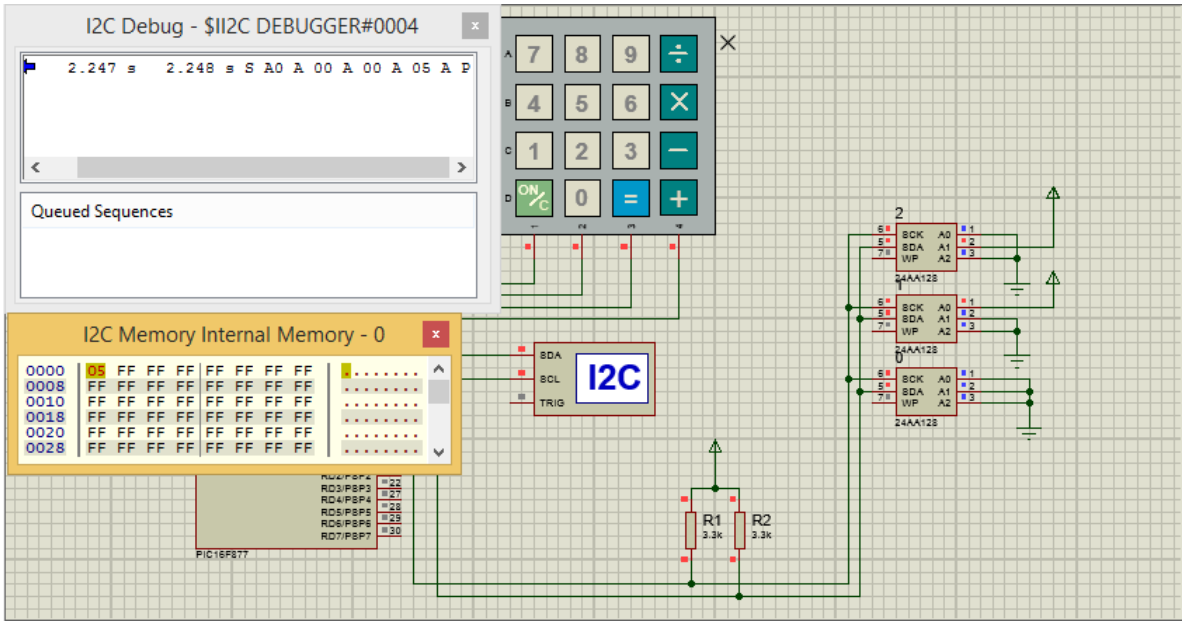

Imagen n°3 Montaje proteus en funcionamiento

En el código para lenguaje c se usa #USE I2C para establecer que se utilizará dicho protocolo, posteriormente en una línea se colocan los parámetros como la condición del dispositivo entre maestro y esclavo, los pines SDA Y SCL, la velocidad de la conexión etc, para el proceso de la comunicación en sí se utilizan las siguientes funciones:

i2c_start();
i2c_write();
i2c_stop();

Donde “i2c_start()” inicia la comunicación, es decir, manda el bit de start, la dirección y espera el Bit de reconocimiento, luego la función “i2c_write()” manda los datos, en el caso de necesitar más de un dato para la conexión se puede mandar más de un “i2c_write()” en este caso se utilizarán 3, luego con el “i2c_stop” detenemos la comunicación.

El código se presenta a continuación:

```
#include <16F877.h>
#FUSES HS,NOWDT
#USE DELAY (CLOCK = 4M)
#DEFINE USE_PORTB_KBD TRUE
#INCLUDE <kbd4X4_1.C>
#use standard_io(b)
#use standard_io(c)
#include <stdlib.h>
#USE I2C (MASTER, SDA= PIN_C4, SLOW, SCL= PIN_C3, NOFORCE_SW)

void main () {
int dispositivo;
char k='0';
int i=0;
kbd_init();
port_b_pullups(true);

while(1){
k=kbd_getc();
i=(k-48);
if(k=='A'){Dispositivo=0;}
if(k=='B'){Dispositivo=1;}
if(k=='C'){Dispositivo=2;}
if((i>-1)&&(i<10)){

if(Dispositivo==0){
i2c_start();
i2c_write(0xA0);
i2c_write(0x00);
i2c_write(0x00);
i2c_write(i);
i2c_stop();
delay_ms(100);
i=0;}

if(Dispositivo==1){
i2c_start();
i2c_write(0xA2);
i2c_write(0x00);
i2c_write(0x00);
i2c_write(i);
i2c_stop();
delay_ms(100);
i=0;
}


if(Dispositivo==2){
i2c_start();
i2c_write(0xA4);
i2c_write(0x00);
i2c_write(0x00);
i2c_write(i);
i2c_stop();
delay_ms(100);
i=0;
```

Imagen n°4 código