Congreso Internacional en Ingeniería Electrónica. Mem. Electro 2015, Vol. 37, pp 370-375, Chihuahua, Chih. México
http://depi.itchihuahua.edu.mx/display/memorias_electro/MemoriaElectro2015
ISSN 1405-2172

DISEÑO Y CONSTRUCCIÓN DE UN MÓDULO MULTI-SENSOR CON CAPACIDADES DE COMUNICACIÓN SPI

Arellano Barragán Hever, Rojas-Estrada Juan-Antonio*, Quintero-Alvarez Patricia, Rincón Martínez Ernesto J., Gutiérrez Delgado J. A., López Saldivar F. R.
Instituto Tecnológico de Nuevo León
División de Estudios de Posgrado e Investigación
Posgrado en Ingeniería Mecatrónica
Av. Eloy Cavazos 2001
Tel. 81 8157 0500
e-mail: hever.arellano@itnl.edu.mx, juan.antonio.rojas@itnl.edu.mx, patricia.quintero@itnl.edu.mx, ernesto.jesus.rincon@itnl.edu.mx, juan.gutdel@gmail.com, rsl+itnl@oblivio.org
*Autor correspondiente

RESUMEN

En este artículo se describe el diseño y la construcción de un Módulo sensor ultrasónico de 6 canales con interfaz de comunicación tipo Serial Peripheral Interface (SPI). El objetivo general es contar con un sistema de aplicación múltiple, entre otras, para construir redes de sensores y/o sensores inteligentes utilizando esquemas de fusión de datos o redes neuronales. La aplicación principal se enfoca en la robótica móvil. El principio fundamental de este módulo se basa en la implementación en Firmware del protocolo de comunicación SPI en modo esclavo, sobre un microcontrolador de la familia AVR de ATMEL. Para este caso particular se utilizaron 6 sensores ultrasónicos genéricos del tipo HY-SRF04, logrando contar con los procedimientos de la activación independiente de cada uno así como la lectura también en forma independiente a través de comandos SPI. Paralelamente se generó una rutina para su aplicación en un remolque de robot, teniendo resultados aceptables.

Palabras Clave: Módulo Multi-Sensor, Red de Sensores, Sensores inteligentes, SPI.

ABSTRACT

This paper describes the design and construction of a 6-channel ultrasonic sensor module, using a Serial Peripheral Interface (SPI) communication capability; the main goal of this development is to use this device, among others, in building Sensor Networks and/or Smart Sensors through the use of Data Fusion or Neural Networks schemes. The main application of this system is on mobile robots operations. The main principle of this Sensor Module is based on the firmware implementation of the SPI protocol over an AVR family Microcontroller of ATMEL. On this particular case, 6 HY-SRF04 ultrasonic sensors where used, each one can be triggered and accessed in a self-contained way through SPI commands. At the same time a rutine was prepared to test the Multi sensor module in a trailer of a robot mobile with acceptable results.

Keywords: Multi Sensor Module, Sensor Network, Smart Sensor, SPI.

1. INTRODUCCIÓN

Un módulo multi-sensor [1], es un subsistema que proporciona información de varios sensores individuales a través de una sola interfaz de comunicación, el objetivo principal de este módulo es simplificar el acceso a las lecturas de los sensores al utilizar una única interfaz de comunicación para colectar todos los datos.

Existen diversas alternativas para comunicar un sistema maestro con los subsistemas de sensado, algunas de ellas son I2C, SPI, UART, etc. como los descritos en [2].

El protocolo de comunicación SPI es ampliamente utilizado en sistemas embebidos para comunicar dispositivos y/o subsistemas que integran un hardware más grande, como por ejemplo los dispositivos dentro de una tarjeta principal de computadora. Uno de los objetivos principales del sistema multi-sensor, es utilizarlo en la integración de sistemas de instrumentación y/o percepción en robots móviles y no sólo como dispositivo aislado o independiente, se propone la interfaz de comunicación SPI como una herramienta que facilita la integración de sistemas de mayor complejidad añadiendo flexibilidad a las etapas de diseño. Este dispositivo puede ser utilizado en diversas plataformas de hardware, (Microcontroladores, *Raspberry pi*, *Beagle bone*, entre muchas otras), que incluyen en su hardware la interfaz SPI en modo maestro.

La Figura 1, muestra la arquitectura de un Módulo Multi-Sensor, se observa que más de un sensor está conectado al módulo; internamente, el controlador cuenta con un multiplexor para seleccionar el canal deseado, un dispositivo con capacidad para almacenar los datos adquiridos y la interfaz de comunicación para transmitir los datos al sistema maestro.

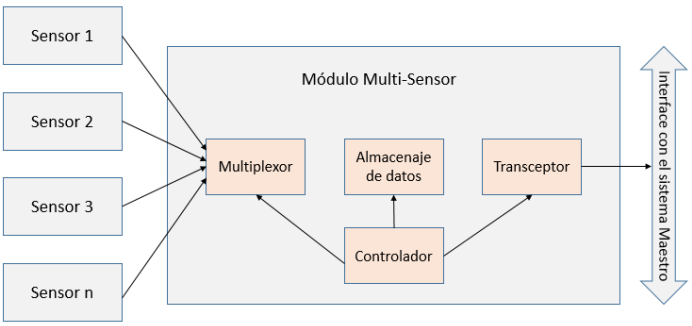

Figura 1. Esquema de Módulo Multi-sensor

ISSN 1405-2172

2. EL SENSOR ULTRASONICO HY-SRF04

El sensor HY-SRF-04 es un sensor de proximidad ultrasónico, de propósito general, este dispositivo, necesita 2 pines de entrada-salida para ser operado, el sensor debe recibir un pulso en la terminal de disparo (trigger), para iniciar la operación de sensado, la respuesta del sensor se obtiene a través del pin eco, en esta terminal se puede leer un pulso con duración proporcional a la distancia a la que fue encontrado un objeto.

Figura 2. Sensor Ultrasónico HY-SRF04.

2.1. Diagrama de tiempo del sensor ultrasónico.

La Figura 3 muestra el diagrama de tiempo de la operación del sensor ultrasónico [3], nótese que es necesario enviar un pulso a través de la terminal de disparo de al menos 10 μs para que el sensor reconozca una señal de inicio válida e inicie su operación.

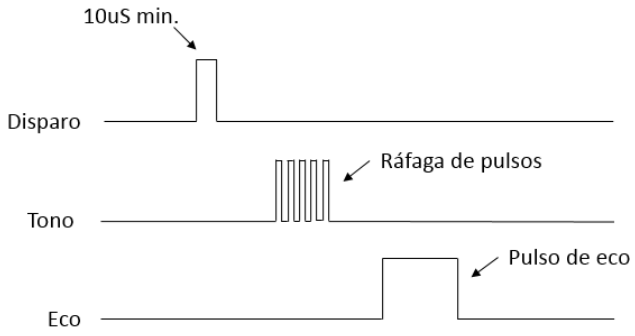

Figura 3, Diagrama de tiempo del sensor HY-SRF05.

Una vez que el sensor reconoce un pulso de inicio válido, este envía una ráfaga de pulsos con una frecuencia de 40 KHz. Unos instantes después, el valor de la línea de eco cambia a un estado alto y permanece así en espera el reflejo de los pulsos previamente transmitidos, tan pronto el micrófono detecta el reflejo del tren de pulsos enviado, la señal de eco cambia a un estado bajo, el tiempo de duración del pulso eco, es proporcional al doble de la distancia a la que se encuentra el objeto detectado, es decir lo que tarda el pulso en ir y regresar hasta el obstáculo.

3. EL PROTOCOLO DE COMUNICACIÓN SPI

Para la implementación del módulo Multi-sensor, se utilizó el protocolo de comunicación SPI [4], una de las razones principales para su uso, es que muchos sensores en el mercado cuentan con esta interfaz, esto permite que múltiples sensores, de diversos tipos, puedan ser interconectados a la misma red SPI y manejados por un solo sistema SPI maestro.

La comunicación serial de periféricos es un protocolo de comunicación que utiliza un bus de 3 hilos, (CLK, Din, Dout). Existen diversas configuraciones en las que un sistema SPI puede trabajar, en este caso utilizaremos una de las más simples, llamada comúnmente configuración de "esclavos independientes", en esta configuración se utiliza una línea de selección de dispositivo o Chip-Select (CS), para cada dispositivo esclavo conectado a la red.

En la Figura 4 se presenta un esquema de interconexión entre un dispositivo SPI maestro y 3 esclavos.

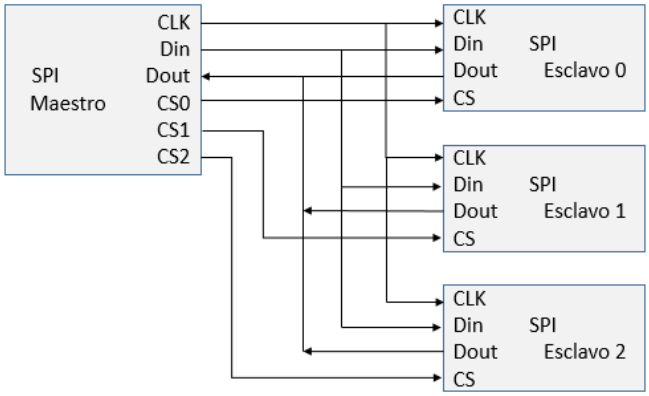

Figura 4. Esquema de conexión de dispositivos SPI.

3.1. Tiempo de muestreo

Existe una relación directa entre la distancia máxima de detección de objetos y el tiempo que tarda el dispositivo ultrasónico en producir un resultado, esto se debe a que a las ondas sonoras les toma más tiempo en ir y regresar si la distancia a la que se espera que lleguen es más grande, este tiempo de espera es similar al tiempo de conversión de un convertidor de análogo a digital (ADC).

El tiempo que se debe esperar un dato en relación con el rango máximo de detección se puede calcular con la siguiente formula:

$$\boldsymbol{t = 2R_m/V_s} \quad (1)$$

donde:

t es el tiempo de respuesta en segundos
R_m es el rango máximo en metros

ISSN 1405-2172

V_s es la velocidad del sonido en m/s

En consecuencia, es necesario tomar en cuenta esta circunstancia en el momento de solicitar una lectura de sensor, en la comunicación SPI se debe insertar un periodo de reloj más largo inmediatamente después de que se ha enviado el último bit del comando SPI, el tiempo de duración de este último ciclo de reloj se determina con la ecuación (1), este retardo, dará tiempo al módulo para tomar la lectura del sensor y almacenar la respuesta en la variable *datoout* para su posterior transmisión al dispositivo maestro, no es necesario continuar enviando ciclos de reloj largos, una vez que se ha realizado la lectura del sensor, la frecuencia del reloj se debe reestablecer, para ahorrar tiempo, sobre todo en procesos que requieran muchas lecturas en periodos relativamente cortos de tiempo.

La Velocidad del Sonido

Otro punto importante a considerar para el desarrollo de este sistema, es que la velocidad del sonido depende de la temperatura, dado que no es posible, en general, realizar experimentos bajo condiciones de temperatura controlados, ha sido necesario incluir dentro del módulo multisensor un sistema de compensación térmica [5].
Para calcular de una forma más exacta el valor de la velocidad del sonido y con ello compensar las lecturas de nuestros sensores, se utilizó un sencillo sensor de temperatura analógico LM35, una vez obtenido el valor de la temperatura ambiente, la velocidad se calcula con la siguiente ecuación:

$$V_s = 331.3 + (0.606 * T_C) \quad (2)$$

donde:
V_s =Velocidad del sonido
T_C = Temperatura en Centígrados.

Una vez calculada la velocidad del sonido de acuerdo a las condiciones de temperatura de ambiente, se utiliza este dato para calcular la distancia hacia los objetos.

La comunicación Serial

Este dispositivo utiliza un esquema de comunicación similar a la utilizada por el ADC MCP3008 de Microchip [6], es decir, recibe códigos de control de 4 bits, y responde según sea el caso con la lectura del sensor solicitado como se muestra en la Tabla 1.

Tabla 1. Comandos de lectura

Bits de Control				Dec	Canal seleccionado
Single	D2	D1	D0		
1	0	0	0	0	Canal 0
1	0	0	1	1	Canal 1
1	0	1	0	2	Canal 2
1	0	1	1	3	Canal 3
1	1	0	0	4	Canal 4
1	1	0	1	5	Canal 5
~~1~~	~~1~~	~~1~~	~~0~~	~~6~~	~~Canal 6~~
~~1~~	~~1~~	~~1~~	~~1~~	~~7~~	~~Canal 7~~

Note que los canales 6 y 7 no están disponibles ya que solo existen físicamente 6 sensores conectados al microcontrolador, por supuesto, el número de canales puede ser mayor si se utiliza un microcontrolador con más pines de entrada, el número de comandos de control también puede ser extendido tanto como sea necesario bajo el mismo principio de funcionamiento, una de las principales ventajas de este sistema es su capacidad de expansión.

3.2. Diagrama de tiempos de SPI.

La Figura 5 muestra el diagrama de tiempos [7] de la comunicación serial entre el dispositivo maestro y el esclavo, el sistema maestro controla el reloj CLK, (en flanco de subida) la comunicación se lleva a cabo bit a bit, cada vez que el CLK cambia de estado bajo a estado alto. Para iniciar la comunicación, primero se activa el dispositivo enviando un estado bajo al pin CS, se envía un bit de inicio en alto al pin *Din*, el cual es ignorado por el dispositivo esclavo y en seguida el código de control de 4 bits.
Una vez que se ha completado el envío del comando de control, es necesario hacer una pausa, la cual se describió en la sección 3.2, este retardo está representado en el gráfico como tSAMPLE, y depende del rango máximo de alcance del sensor previamente definido, es decir que habrá un periodo de reloj más largo de lo normal, nótese que no es indispensable que los ciclos de reloj sean periódicos. Si se continúa enviando ciclos de reloj, el dispositivo esclavo enviará el valor obtenido de la lectura del sensor bit a bit a través del pin *Dout.* En este caso la respuesta, es un dato de 10 bits pero esta longitud puede ser configurable.

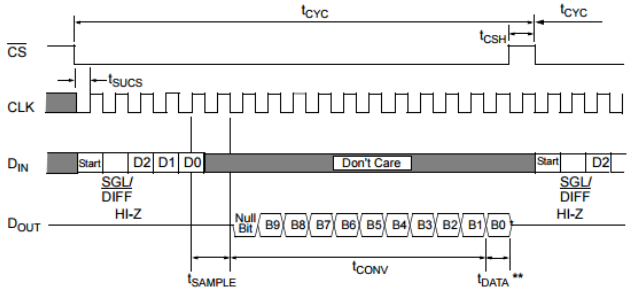

Figura 5. Diagrama de tiempos de la comunicación serial

4. ESQUEMÁTICO

La Figura 6, muestra el esquemático del módulo Multi-Sensor, donde se observa que cada uno de los sensores ultrasónicos HY-SRF04, necesita un mínimo de 2 pines de ES del microcontrolador para obtener las lecturas. Es posible incrementar la cantidad de sensores conectados al módulo utilizando un microcontrolador con más pines de ES, o mediante el uso del sensor HY-SRF05, que puede usarse en configuración de un solo pin, a un precio más elevado.

ISSN 1405-2172

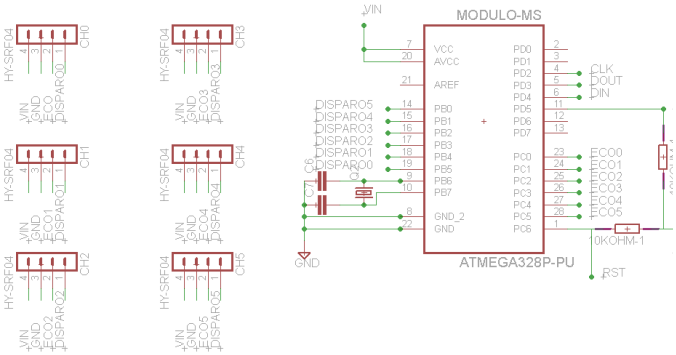

Figura 6. Esquemático del Módulo Multi-Sensor.

5. IMPLEMENTACIÓN DE SPI POR FIRMWARE.

La Figura 7, muestra el diagrama de flujo para la implementación del protocolo de comunicación SPI en modo esclavo, el flujo de programa inicia, una vez que el Chip Select del dispositivo se encuentre en un nivel bajo (cero lógico), el disparador principal de las acciones en el programa es un evento en la señal de reloj CLK, existen 3 condiciones principales dentro del flujo del programa.

I. El contador de ciclos de reloj *Cont*, es menor o igual a 4, en esta parte de la secuencia el microcontrolador lee el comando de control bit a bit a través del pin *Din*, el dato es almacenado en la variable *Code*, utilizando un OR lógico y corrimientos.

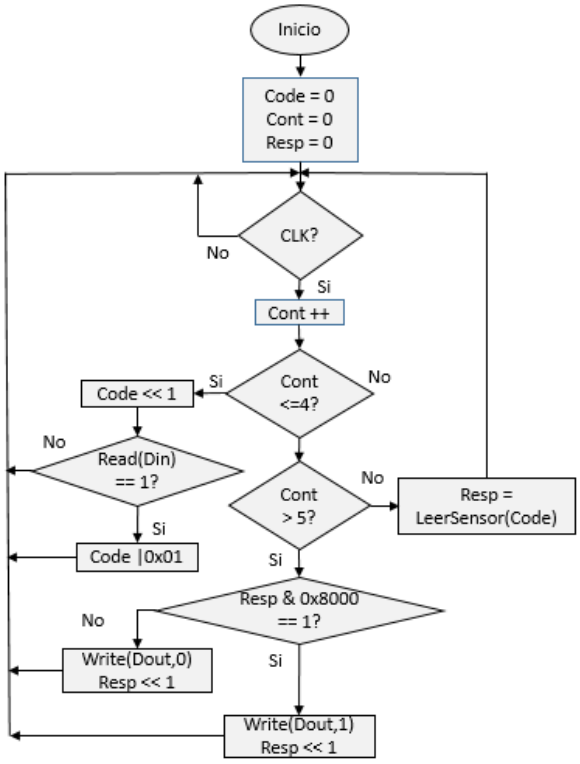

Figura 7. Diagrama de flujo del programa SPI esclavo.

II. El contador de ciclos de reloj *Cont* es exactamente igual a 5, en esta parte del programa el microcontrolador ha recibido el comando de control y procede a hacer la lectura de ese sensor en particular.
El ciclo de reloj de 5 a 6 debe durar lo suficiente para dar tiempo al sensor de tomar una lectura valida de acuerdo con lo descrito en la sección 2.1

III. El contador de ciclos de reloj *Cont* es mayor de 5, en este momento el controlador ya tiene la lectura del sensor deseado almacenado en la variable *Resp*, y puede empezar a escribir el valor bit a bit a través del pin *Dout*, utilizando un AND lógico y corrimientos.

5.1. Codificación para Arduino.

Sin duda uno de los ambientes de desarrollo para microcontroladores más populares es el de la plataforma Arduino, por ello se presenta en esta sección la codificación básica para la implementación del protocolo SPI en modo esclavo, de modo que pueda ser probada fácilmente.

```
const int Dout = 3;
const int Din = 4;
const int CS   = 5;

unsigned int datos = 0;
boolean last_CS =1;
byte datoin = 0x00;
word datoout = 0xAAAA;
```

La primera sección del código muestra las definiciones básicas, las primeras 3 líneas definen los pines donde se ubicaran las señales *Din*, *Dout* y *CS*.
La variable *datos*, guarda la cantidad de bits recibidos, *last_CS* el ultimo estado de la señal del Chip Select, *datoin* el valor del código de control recibido y finalmente *datoout* guardara el valor que se quiera devolver dada una medición u cualquier otra acción para la que el controlador este programado, este caso la variable contiene un dato de prueba 0xAAAA.

```
void setup()
{
 pinMode (CS,INPUT);
 pinMode (Dout,INPUT);
 pinMode (Din,INPUT);
 }
```

La segunda sección del código inicializa los pines de Entrada Salida, es importante recalcar que el pin *Dout* debe estar configurado como entrada (baja impedancia) cunando el dispositivo no está activo, es decir cuando la señal *CS* está en alto, solamente debe ser configurado como salida, cuando el

dispositivo este activo, de otro modo el dato no se propagara en la red SPI.

```
void loop(){
  if((!digitalRead(CS)) && (last_CS)){
    attachInterrupt(0,CLK,RISING);
    last_CS = 0;
    pinMode (Dout,OUTPUT);
    datos = 0;
    datoin = 0;
    }
  if((digitalRead(CS)) && (!last_CS)){
    detachInterrupt(0);
    Serial.println(datos);
    last_CS = 1;
    pinMode (Dout,INPUT);
  }
}
```

En la siguiente sección del código, constantemente se verifica si el dispositivo ha recibido la señal de activo, para ello, el pin de CS debe estar en bajo y su anterior estado debe alto, eso garantiza que hubo una transición de estado en la señal y nos asegura una señal de inicio válida.
Cuando el dispositivo está activo (la primera condición if), se inicializan las variables a 0, se configura el *Dout* como salida y se habilita la interrupción externa 0 en flanco de subida, la interrupción fungirá como entrada de reloj, cada vez que esta se active, se ejecutará el código dentro de la función CLK().
Para el caso en el que el dispositivo este desactivado, (es decir la segunda condición if), se deshabilita la interrupción, de esta forma el dispositivo hace caso omiso a cualquier mensaje que provenga de la red y el pin Dout vuelve a ser una entrada para permitir la propagación de los mensajes en el bus de comunicación SPI.

```
void CLK(){
  if(datos <= 4)
    datoin = datoin << 1;
    if(digitalRead(Din))
        datoin = datoin | 0x01;

  if(datos == 5)
    switch(datoin){
      case 0x18: "HACER ALGO" break;

      case 0x19: "HACER ALGO" break;
       .
       .
       .
    }
  if(datos > 5){
    digitalWrite(Dout,LOW);
    if(datoout & 0x8000)
      digitalWrite(Dout,HIGH);
   datoout = datoout << 1;
  }
  datos ++;
}
```

La cuarta y última sección del código, define lo que sucede cuando el dispositivo está activo y se recibe una señal de reloj, este segmento consta de 3 partes principales.
Primero se recibe el código de control es decir el dato que determinará la acción a tomar por el dispositivo esclavo, el código colecta bit a bit este dato, en este caso son solo 4 bits, pero puede ser extendido a muchos más.
Posteriormente y una vez obtenido el dato de control, se ejecuta la acción definida por el código de control que ha sido recibido, la forma más simple es haciendo un arreglo de casos, comparando el dato de entrada con alguno de los códigos de control predefinidos y asignados a las diferentes acciones posibles, puede haber tantos casos como códigos de control sean definidos, en este caso particular, la lectura de alguno de los sensores ultrasónicos [8].
Por último, el dato guardado en la variable *datoout*, es enviado como respuesta, el dato es liberado bit a bit, a la red SPI, nótese que no existe un límite en el número de bits que pueden ser enviados, mientras la señal de reloj siga llegando el dispositivo seguirá enviando bits, por lo que es posible enviar respuestas de tantos bits como se quiera, en este caso *dotoout* es una variable de 16 bits, pero este valor puede ser extendido.
Finalmente para terminar con una transmisión de información, el dispositivo maestro debe desactivar al esclavo poniendo la señal de CS en alto.

6. RESULTADOS

Las gráficas de las Figuras 8 y 9 muestran una serie de lecturas de cada uno de los canales del módulo multisensor, para el primer caso una serie de objetos fueron colocados a una distancia lineal de 20, 30, 40, 50, 60 y 70 cm respectivamente. Justo en frente de cada sensor ultrasónico, las lecturas fueron solicitadas a través de una interfaz SPI utilizando una tarjeta *Raspberry pi* y un script de Python [9], aunque se tomaron una gran cantidad de lecturas, el grafico muestra solo 100 de ellas para cada sensor, con el objetivo de dar claridad a las muestras. Las leyendas en las gráficas se explican cómo sigue: **S5r** es el valor REAL, a la que se encontraba el objeto justo enfrente del sensor 5, mientras que **S5m** es el valor de la lectura MEDIDA por el módulo multisensor en el canal 5, la misma regla aplica para cada leyenda sucesivamente.

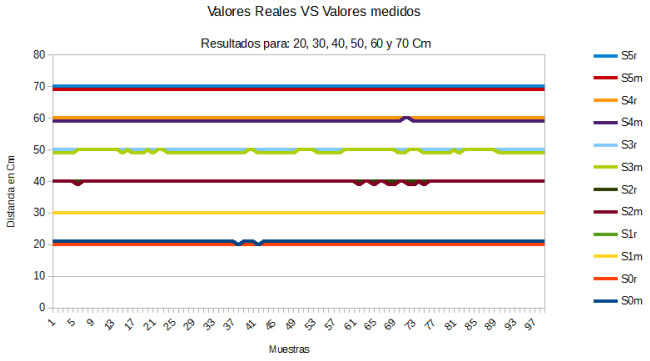

Figura 8. Gráfica de Lecturas vs. Distancia real

La Figura 9 muestra la gráfica de otra serie de datos tomados esta vez a distancias de 17, 25, 34, 47, 53 y 65 cm.

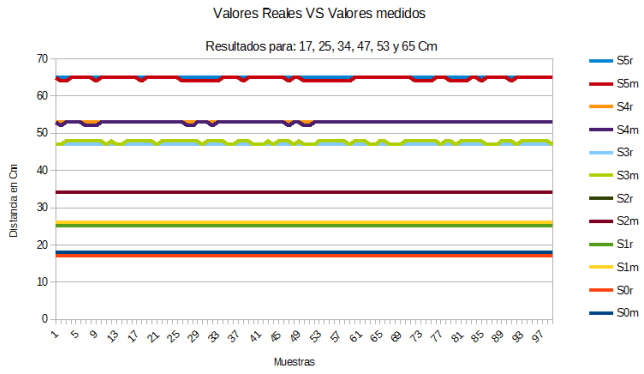

Figura 9. Gráfica de Lecturas vs. Distancia real

Se obtuvieron muy buenos resultados dentro del rango de los 70 cm, las variaciones entre la distancia real y el valor medido obtenido no fueron mayores a 1 cm para cada una de las lecturas que se tomaron, para fines del uso de este dispositivo en robots móviles los resultados son aceptables.

7. CONCLUSIONES

Los resultados obtenidos en este trabajo concuerdan con el objetivo inicial y con las características que se obtienen para el Módulo Multi-Sensor y se pueden enumerar de la siguiente manera: es capaz de detectar objetos a una distancia máxima lineal de 70 cm. en cada uno de los 6 sensores de que consta este dispositivo; es compatible con un bus SPI estándar y puede ser usado por cualquier plataforma de hardware que cuente con hardware SPI en modo maestro; es sumamente configurable ya que tanto los comandos de control como el tamaño del valor de la medición, pueden ser extendidos a palabras de más o menos bits.

Además el rango máximo de detección puede ser configurable y va desde un par de centímetros a (en teoría) más de 4 metros en su escala máxima.

Puede ser utilizado dentro de redes SPI con más de un dispositivo de su misma especie u otros diferentes.

Puede ser fácilmente modificado para el manejo de otro tipo de sensores y/u otras aplicaciones. Se propone como trabajo futuro; la utilización de filtros digitales que mejoren la respuesta del módulo multisensor y eviten lecturas en falso o errores aleatorios, para esto, se estaría contemplando un filtro de Kalman, optimizar el tiempo de respuesta del sistema, es otro tema, aunque en los experimentos realizados se pudieron obtener un total de 10 lecturas por segundo, no se determinó exactamente cuál es el límite y como tercer punto, mejorar el tiempo de cálculo ya que se invierte un tiempo considerable realizando cálculo dentro del microcontrolador, en principio es posible hacer un análisis que nos ayude a simplificar las operaciones y el tiempo de ejecución de cálculos relativamente complejos para una arquitectura de 8 bits.

8. REFERENCIAS

[1] Md. Sajjad Rahaman, Masud H Chowdhury, Irfan Nasir, Lih-Tyng Hwang, “VSIB: A Sensor Bus Architecture for Smart-Sensor Network”, World Congress on Computer Science and Information Engineering, 2009.

[2] Eugene Y. Song, Kang B. Lee, Sensor Network based on IEEE 1451.0 and IEEE p1451.2-RS232, IEEE International Instrumentation and Measurement Technology Conference,2008.

[3] HY-SRF04 specifcation (non official), Online article, http://www.robot-electronics.co.uk/htm/srf04tech.htm, Mayo de 2015.

[4] Serial Peripheral Interface Bus, Wikipedia Database, Online Article, http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus, Mayo de 2015

[5] Clausio Canali, Giorgio de Cicco, Brundo Morten, Mria Prudenziati, and Andrea Taroni, “A Temperature Compensated Ultrasonic Sensor Operating in Air for Distance and Proximity Measurements”, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, VOL. IE-29, NO. 4, NOVEMBER 1982

[6] Microchip MCP3008, hoja de datos, http://www.microchip.com/wwwproducts/Devices.aspx?product=MCP3008http://ww1.microchip.com/downloads/en/DeviceDoc/21295d.pdf, Mayo de 2015.

[7] M.K. Md Arshad, U. Hashim, Chew Ming Choo, “Characteristics of Serial Peripheral Interfaces (SPI) Timing Parameters for Optical Mouse Sensor”, ICSE2006 Kuala Lumpur Malaysia, 2006.

[8] Arduino Forum, Online article, HY-SRF05 Ultrasonic Sensor, http://forum.arduino.cc/index.php?topic=89524.0, Mayo de 2015.

[9] Adafruit Learn Data Base, Online article, Analog Inputs for Raspberry Pi Using the MCP3008, https://learn.adafruit.com/reading-a-analog-in-and-controlling-audio-volume-with-the-raspberry-pi/, Mayo de 2015.