TECNOLÓGICO DE MONTERREY®

Curso:

Laboratorio de Microcontroladores

Práctica #4: LCD

Manuel Madrigal Valenzuela ID: A01114070

Efraín Duarte López ID: A01113813

Mauricio Capistrán Garza

Noviembre de 2016

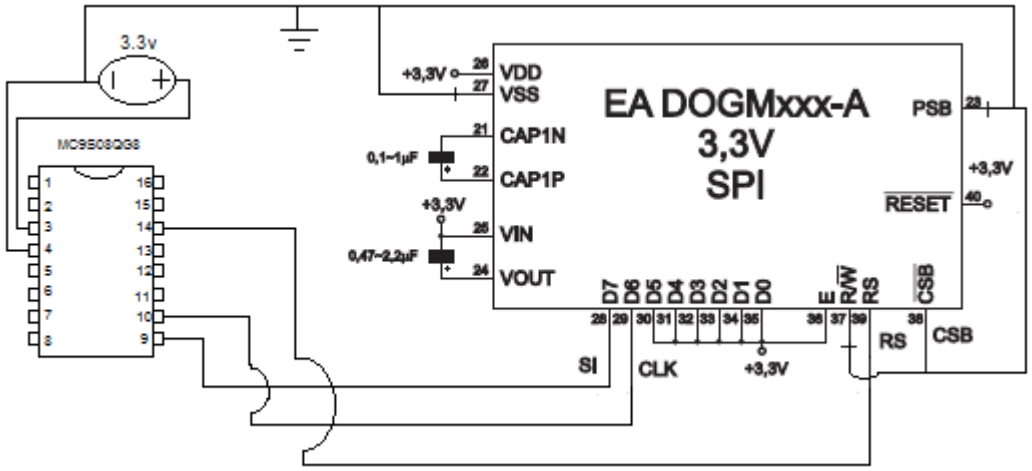
Objetivo:

Controlar por SPI una pantalla LCD a través del microcontrolador MC9S08QG8 para mostrar: temperatura del motor, velocidad (RPS) y dirección (CW o CCW).

Componentes a usar:

- 1 MC9S08QG8 (microcontrolador)
- 2 ULN2803 (arreglo de transistores Darlington)**[2]**
- Motor a pasos unipolar (PM55L-048)**[3]**
- 1 LM35 (sensor de temperatura) **[4]**
- 1 adaptador de USB a serial (DB9)
- 1 MAX232
- Pantalla LCD (EA DOGM162W)**[5]**

Esquemático del circuito LCD:

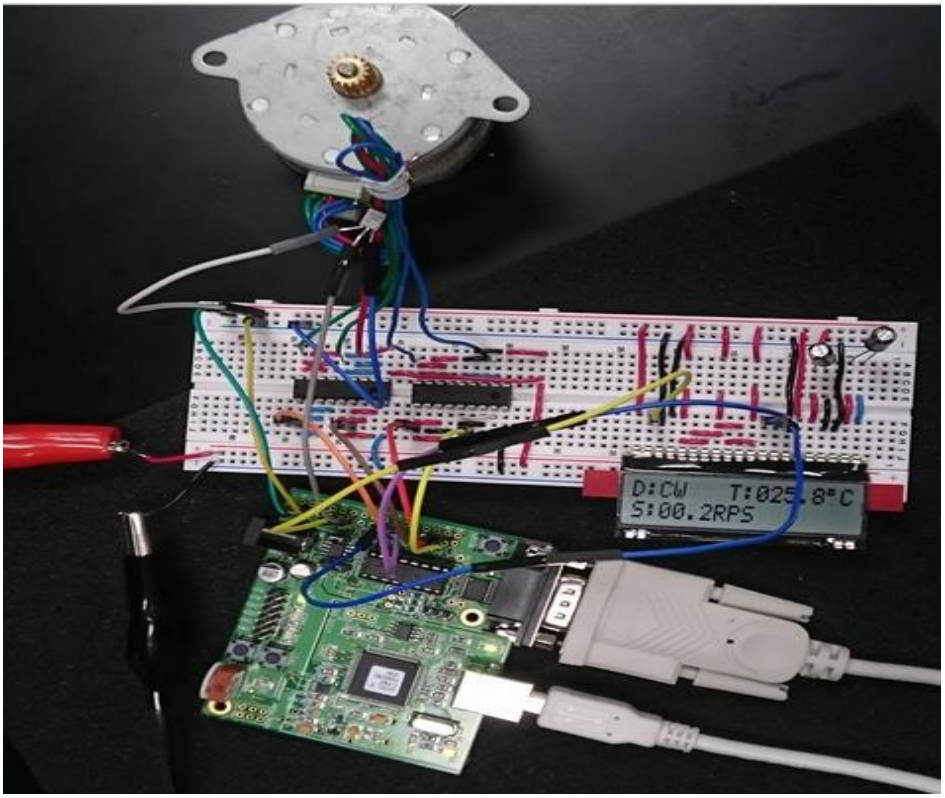


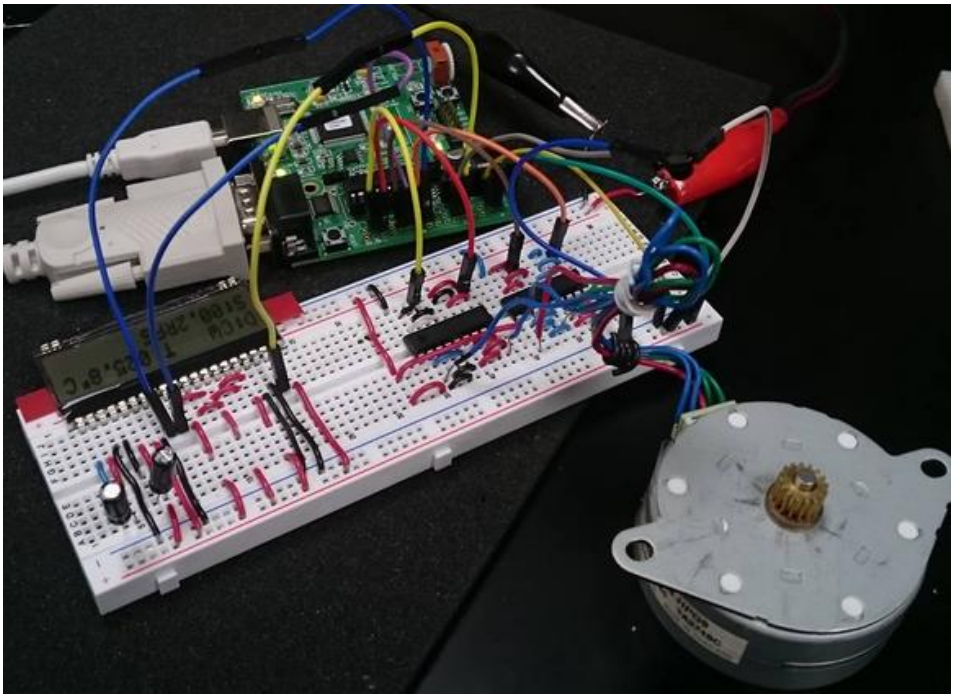
Esquemático del circuito LCD

Diseño eléctrico:

Se eligió alimentar con una fuente de 3.3 V a la pantalla LCD para poder usar esta misma en el microcontrolador. Además, se seleccionaron dos capacitores de 1μF porque están dentro del rango de valores de los que sugiere el proveedor en la hoja de datos si se decide alimentar la pantalla con 3.3V.

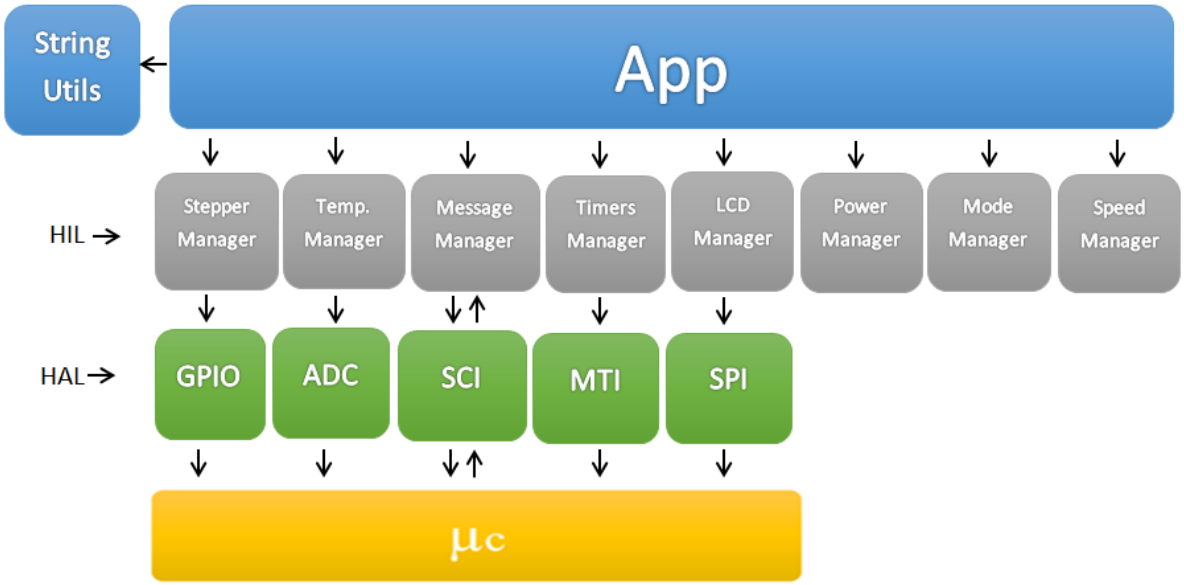
Fotos del circuito armado:





Las imágenes anteriores muestran el circuito montado en un protoboard y el microcontrolador en la tarjeta programable. Además se muestra la conexión de la pantalla LCD y su funcionamiento.

Arquitectura de software:



El driver del LCD se encarga de que el proceso de comunicación con la pantalla LCD se lleve a cabo correctamente. Este driver se implementó de forma no bloqueante por medio de una máquina de estados, conteniendo los siguientes estados: IDLE, SET_POSITION y WRITE_CHAR.

Antes de inicializar la máquina de estados, el driver se encarga de inicializar la pantalla LCD con una serie de comandos que la dejan lista para empezar a recibir datos.

Tabla de comandos de inicialización usados									
Comandos	**DB7**	**DB6**	**DB5**	**DB4**	**DB3**	**DB2**	**DB1**	**DB0**	**HEX**
Function set	0	0	1	1	1	0	0	1	39
Contrast set	0	1	1	1	1	0	0	0	78
Power control	0	1	0	1	0	1	0	1	55
Follower control	0	1	1	0	1	1	0	1	6D
Display ON/OFF	0	0	0	0	1	1	1	1	0F

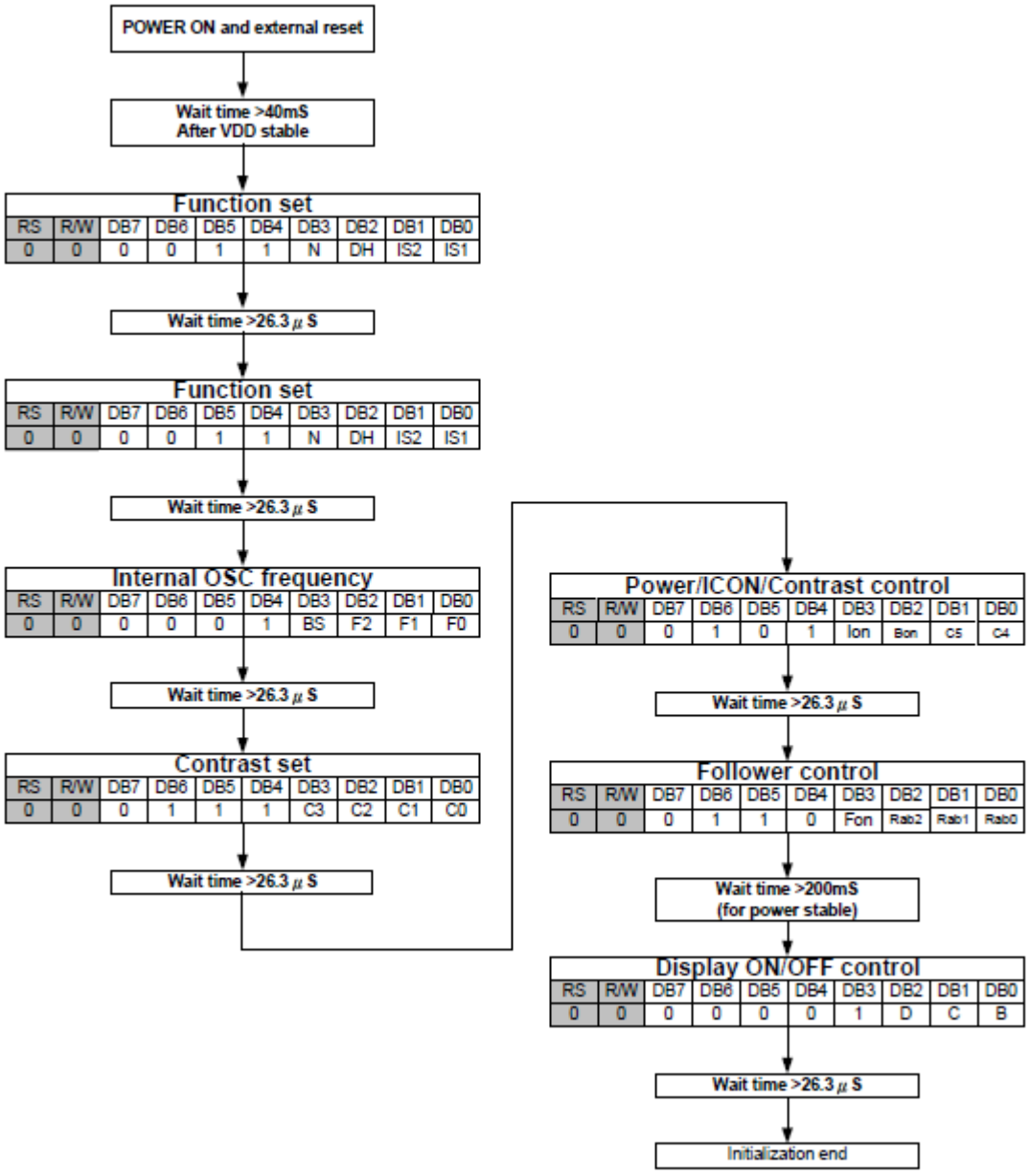


Diagrama de flujo de inicialización

Una vez hecho esto, la máquina de estados es inicializada comenzando en IDLE. La máquina de estados se mantiene en este estado mientras no se mande llamar la función de “LCD_Manager_WriteMessage”, la cual lleva a cabo la tarea de guardar el mensaje a enviar en un buffer, establece la variable bIsBussy a TRUE y cambia la máquina de estados al siguiente estado: SET_POSITION. En este estado se configura la posición de la pantalla en la cual se desea comenzar a escribir, dejándola lista para recibir los datos que se desean desplegar. Por último, el estado WRITE_CHAR, al cual se accesa directamente después de configurar la posición en el estado anterior, se encarga de mandar un solo carácter del mensaje cada vez que la función periódica es mandada a

llamar, esto se hace con el objetivo de que el microcontrolador tenga tiempo de hacer otras tareas de mayor prioridad, por ejemplo: controlar el motor. Una vez que se ha enviado el mensaje completo a la pantalla LCD, la máquina de estados regresa al estado IDLE, comenzando el ciclo nuevamente.

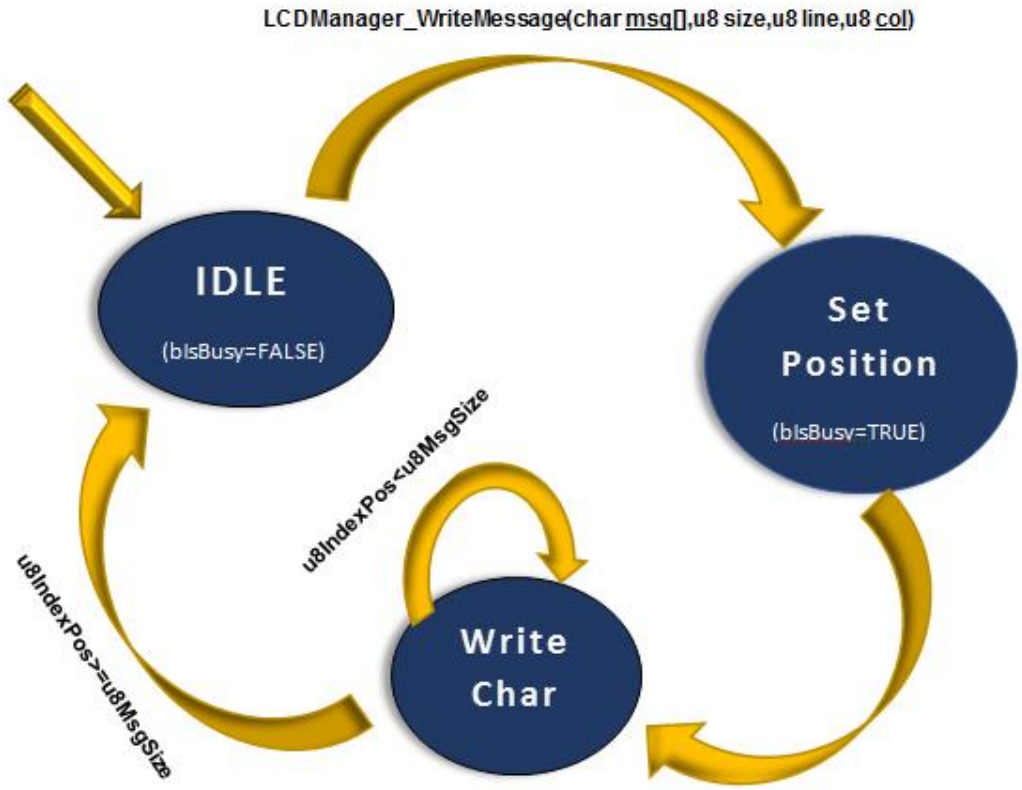
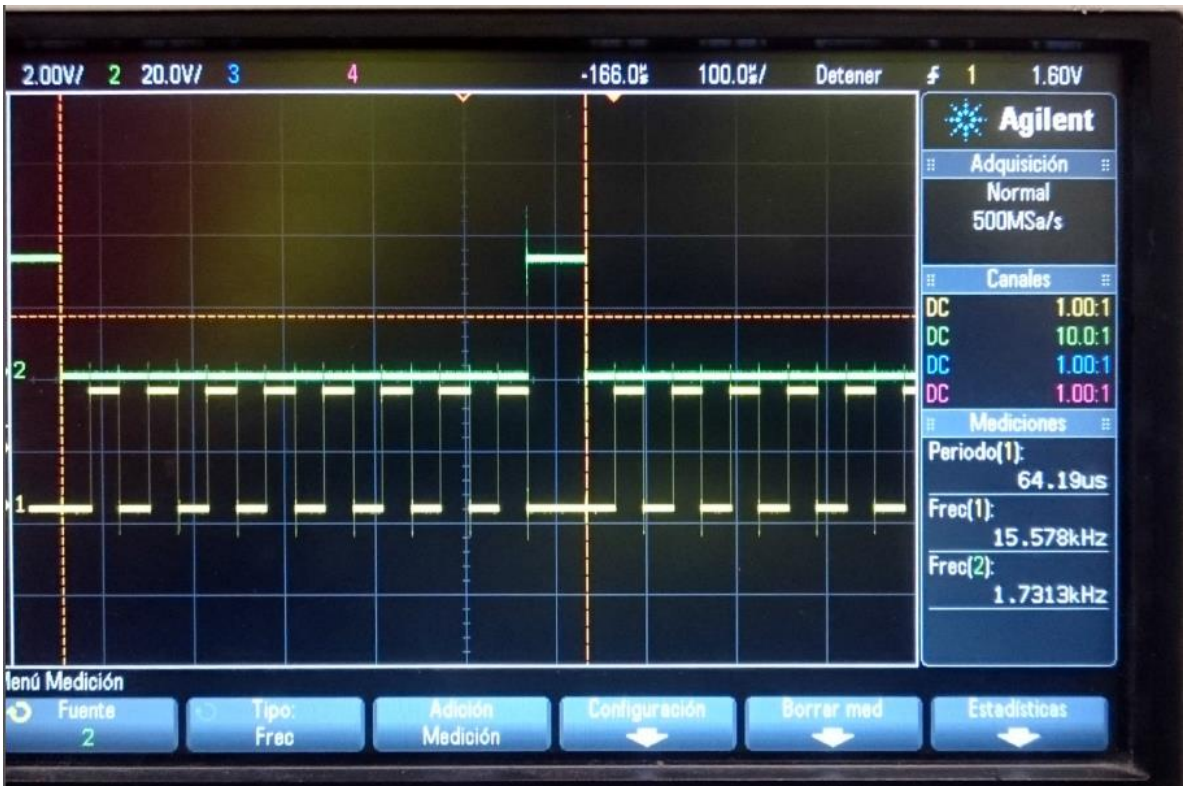


Diagrama de estados

Capturas de osciloscopio:



Captura en osciloscopio del Clock y el MOSI[1]

Captura en osciloscopio del Clock y el MOSI[2]

En las capturas anteriores se muestran dos señales de color amarillo y verde, representando al clock y MOSI respectivamente.

En la ilustración 1 se manda el dato 0 en un byte con un clock de 15.57 kHz. En esta se puede ver un flanco de subida que corresponde a IDLE.

En la segunda imagen se manda el dato 252 (01111110 en binario) donde se puede apreciar el flanco de subida del IDLE seguido de un flanco de bajada el cual es el primer bit del dato y al final del ciclo se aprecia otro flanco de bajada que es el último bit del dato. Por lo tanto, se necesitan 9 ciclos de reloj para mandar un dato completo, 8 para cada bit del dato y uno más para el IDLE.

Conclusión:

El funcionamiento del trabajo fue óptimo cumpliendo ampliamente con los requerimientos. Considerar las siguientes observaciones durante el desarrollo de esta práctica:

- Surgieron problemas con la inicialización de la pantalla, provocando que esta no funcionase correctamente o simplemente no funcionara en algunas ocasiones. Este problema se presentó debido a que la pantalla estaba siendo alimentada y conectada al microcontrolador incluso antes de que este arrancara. Cuando el microcontrolador arranca, no podemos saber qué es lo que hay configurado por el puerto SPI, y como la pantalla estaba conectada, pudiese ser que recibiera información no intencionada por el usuario, arruinando toda la transmisión consecutiva. Un solo bit en el inicio puede hacer que toda la información que se mande posteriormente ya no sea entendida por la pantalla LCD.
  Lo correcto es alimentar la pantalla hasta que estemos seguros de la configuración para el módulo SPI.

Una mejora posible para solucionar el problema de la inicialización de la pantalla sería alimentarla a través de un pin del microcontrolador, de este modo podríamos activarla en el momento indicado, justo después de la inicialización del módulo SPI.

Bibliografía:

1. .. (2013). Secuencias para manejar motores paso a paso (unipolar). Septiembre 19, 2016, de. Sitio web: http://server-die.alc.upv.es/asignaturas/lsed/2002-03/MotoresPasoaPaso/ftomotpap.htm
2. .. (2015). ULN2803 Darlington Transistor Arrays. septiembre 19, 2016, de Texas Instruments Sitio web: http://www.ti.com/lit/ds/symlink/uln2803a.pdf.
3. .. (2004). PM55L-048. Septiembre 22, 2016, de Minebea Motor Manufacturing Corporation Sitio web: http://www.nmbtc.com/pdf/motors/PM55L048.pdf
4. .. (agosto, 2016). LM35 Precision Centigrade Temperature Sensors. Noviembre 2, 2016, de Texas Instruments Sitio web: http://www.ti.com/general/docs/lit/getliterature.tsp?genericPartNumber=lm35&fileType=pdf
5. .. (2012). Dog series 3.3V. noviembre 21, 2016, de Electronic Assembly Sitio web: www.lcd-module.com/eng/pdf/doma/dog-me.pdf