Nombre de la asignatura: Laboratorio de Programación de Avanzada

Maestro en Ciencias: Moisés Agustín Martínez Hernández

Nombre de la práctica: Banda de posiciones con sensor de proximidad infrarrojo y puente H.

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

Número de práctica: 7

Total de horas: 2 Hrs.

Objetivos
- Comprender el concepto de la conversión analógica a digital, las variables que influyen en el proceso como la frecuencia de muestreo, la resolución del dispositivo, los niveles de tensión y acondicionamiento de la señal.
- Aprender a configurar pines de salida para leer señales de tipo Analógicas en el microcontrolador.
- Comprender el concepto de voltaje de referencia para configurar y convertir la lectura del sensor de proximidad infrarrojo.
- Comprender como funciona un puente H y usarlo para controlar la dirección de giro de un motor DC.
- Implementar la lectura de canales analógico para realizar una conversión de nivel de tensión con el propósito de medir la distancia, por medio del sensor, y mover un objeto a cierta distancia.

Descripción de la práctica

Se utilizará una entrada analógica para leer un voltaje aplicado al microcontrolador, el nivel de voltaje será el necesario para configurar la referencia del sensor de proximidad infrarrojo, en la otra entrada el voltaje dependerá de la distancia que lea el sensor. La variación del voltaje se verá reflejada el sentido de giro del motor DC controlado por un puerto de salida del microcontrolador, dicho motor cambiará su sentido de giro en función de la distancia a la que se encuentre el objeto para colocarlo a la distancia que se desea que esté, dicha distancia será definida por el usuario mediante botones con distancia predefinidas y por medio de un potenciómetro. También se agregará un display para mostrar la distancia a la que se encuentra el objeto.

Marco teórico

Conversión Analógica a Digital

La salida de los sensores, que permiten al equipo electrónico interaccionar con el entorno, es normalmente una señal analógica, continua en el tiempo. En consecuencia, esta información debe convertirse a binaria (cada dato analógico decimal codificado a una palabra formada por unos y ceros) con el fin de adaptarla a los circuitos procesadores y de presentación. Un convertidor analógico-digital (CAD) es un circuito electrónico integrado cuya salida es la palabra digital resultado de convertir la señal analógica de entrada. La conversión a digital se realiza en dos fases: cuantificación y codificación. Durante la primera se muestrea la entrada y a cada valor analógico obtenido se asigna un valor o estado, que depende del número de bits del CAD. El valor cuantificado se codifica en binario en una palabra digital, cuyo número de bits depende de las líneas de salida del CAD. Estos dos procesos determinan el diseño del circuito integrado.
En un CAD de N bits hay 2N estados de salida y su resolución (porción más pequeña de señal que produce un cambio apreciable en la salida) se expresa como 1/2N (una parte en el número de estados). Con frecuencia la resolución se expresa a partir del margen de entrada del convertidor para definir el intervalo de cuantización o espacio de 1 LSB (Least Significant Bit; bit menos significativo).
La figura 1 representa la respuesta de un convertidor A/D de 3 bits a una entrada analógica sinodal de 1 kHz de frecuencia, valor medio 5 V y valor cresta a cresta de 10 V, coincidentes con el margen de entrada. En ella se observan los 23=8 estados de la salida, correspondientes a los códigos binarios desde el 000 al 111. Cada intervalo de cuantización tiene una anchura de 10 (V)/8 (estados)=1,25 V.

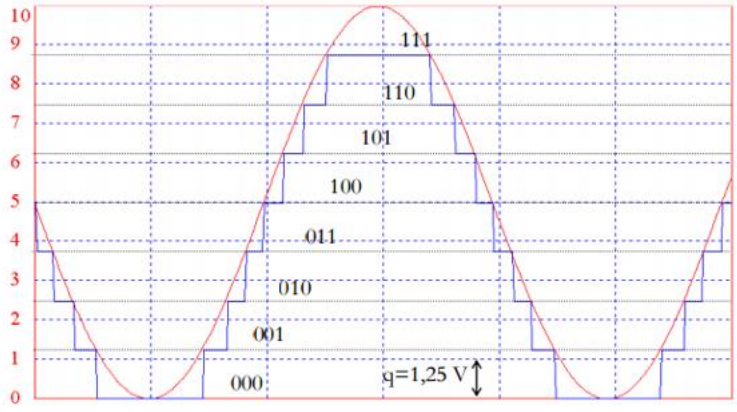


Figura. 1. Digitalización de una señal analógica por un convertidor A/D de 3 bits.

Sensor de proximidad infrarrojo

El sensor de proximidad infrarrojo sharp es un dispositivo opto electrónico que permite realizar mediciones de distancia mediante la reflexión de un rayo infrarrojo en el objeto a detectar.

El sensor tiene una salida analógica que puede leerse mediante un convertidor analógico a digital en un microcontrolador. El voltaje de salida varía en un rango de 2.8 volts a 15 cms y 0.4 volts a 150 cms, dependiendo del modelo, con una fuente de alimentación comprendida entre 4.5 y 5.5 VDC.

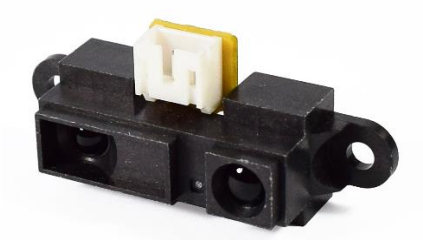


Figura 2. Sensor de proximidad Sharp.

Voltaje de referencia

El voltaje de referencia es el estándar primario por el cual un conversor analógico-digital (ADC) o un conversor digital-analógico (DAC) juzga o produce el voltaje de la señal analógica. Es un nivel de voltaje constante y específico que no se ve afectado por las variables externas. Una referencia de voltaje estable y confiable permite una conversión precisa entre las señales analógicas y digitales.

Motor de corriente directa

Un motor de corriente directa es un dispositivo que convierte la energía eléctrica de corriente continua en energía mecánica, provocando un movimiento rotatorio. El motor funciona gracias a la acción de un campo magnético que genera fuerzas electromagnéticas.

Figura 3. Motor de corriente directa

Puente H

El puente H es un circuito electrónico que permite a un motor eléctrico DC girar en ambos sentidos, avanzar y retroceder.

Los puentes H ya vienen hechos en algunos circuitos integrados, pero también se pueden construir a partir de componentes eléctricos y/o electrónicos.

Un puente H se construye con 4 interruptores (mecánicos o mediante transistores). Cuando los interruptores S1 y S4 están cerrados (S2 y S3 abiertos) se aplica una tensión

haciendo girar el motor en un sentido. Abriendo los interruptores S1 y S4 (cerrando S2 y S3), el voltaje se invierte, permitiendo el giro en sentido inverso del motor.

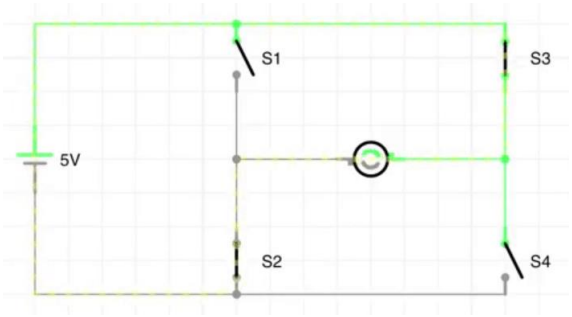


Figura 4. Diagrama de puente H.

Un puente H no solo se usa para invertir el giro de un motor, también se puede usar para frenarlo de manera brusca, al hacer un corto entre los bornes del motor, o incluso puede usarse para permitir que el motor frene bajo su propia inercia, cuando desconectamos el motor de la fuente que lo alimenta.

EQUIPO Y MATERIALES

- Software para programar microcontrolador.
- Software de simulación.
- Microcontrolador (sistema mínimo).
- Resistencia variable (trimpot, etc.).
- 4 botones.
- 1 motor de corriente directa.
- 1 display.
- 1 sensor de proximidad.
- 1 potenciómetro.
- 1 puente H.

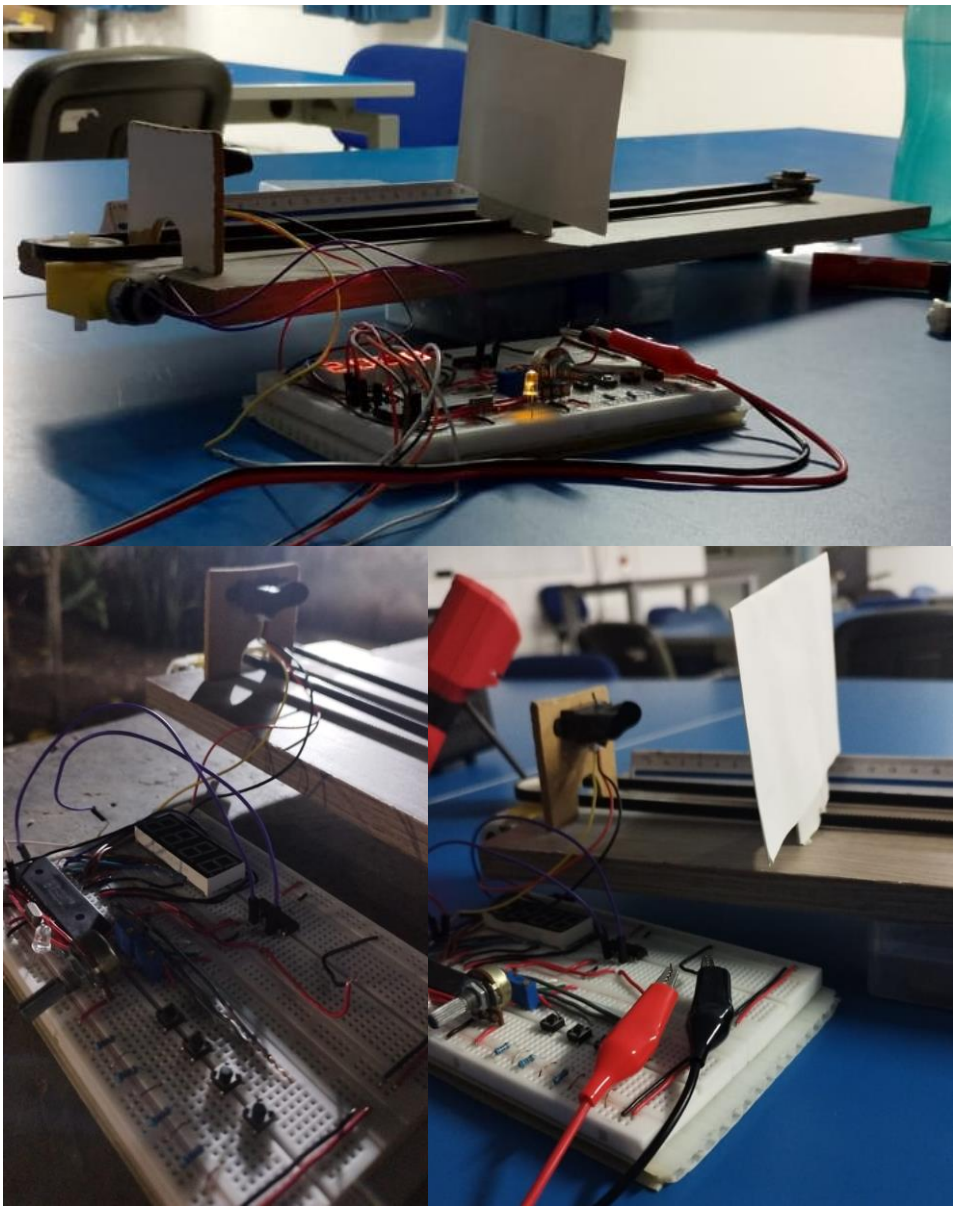
DESARROLLO DE LA PRÁCTICA

1. Configurar el microcontrolador. Considerando:
   a) Frecuencia de reloj 4 MHz.
   b) Habilitar el ADC con la mayor resolución posible del microcontrolador (10 bits).
   c) Configurar un canal analógico del microcontrolador para conectar la señal del sensor.
   d) Configurar un canal analógico del microcontrolador para conectar la señal del voltaje de referencia.
   e) Configurar las salidas digitales necesarias para controlar la activación de los pines del display.
   f) Configurar las salidas digitales necesarias para controlar la activación del motor mediante un puente H.
2. Escribir un código para que efectué las siguientes instrucciones:
   a) Desde el inicio del programa el microcontrolador estará recibiendo la señal del sensor y mostrando la distancia en el display, así como mandando el voltaje necesario para que el puente H este controlando el giro del motor dependiendo de la distancia del objeto.

b) Cuando la distancia cambie se verá reflejado en el display así como en la dirección de giro del motor.

3. Realizar la simulación adecuada para validar el código.
4. Programar el microcontrolador e implementar el circuito necesario para cumplir los requisitos de los puntos 1 y 2.
5. Implementar el circuito en físico.

Resultados de la práctica



Circuito armando

Código explicado

```
#include <18f4550.h>   // Libreria del Microcontrolador
#device adc = 10 // Resolucion del ADC en bits
#fuses XT, NOWDT, NOPROTECT, NOLVP, CPUDIV1, PLL1 // Fusibles
(Configuraciones del microcontrolador)
#use delay(clock = 4M)

#include <math.h>

void Acoplamiento();
void Display(float);
void Descomponer(int *, int *, int *, int *, float);

#define LED pin_d0
// Displays
#define D1 pin_e0
#define D2 pin_d4
#define D3 pin_d6
#define D4 pin_d7
// Botones
#define BOT1 pin_d2
#define BOT2 pin_d3
#define BOT3 pin_c6
#define BOT4 pin_c7
// Distancia Constante
#define MIN 10
#define MAX 31

int vec_c[10] = {0x3F, 0x06, 0x5B, 0x4F, 0x66,
0x6D,0x7D,0x07,0x7F,0x67}; // Numeros para display Catodo
int16 duty[] = {200, 500, 850}; // Obtenidos con el excel
int MODO = 1; // Bandera para establecer el MODO

float dist_act;
int16 dist_prox, cont = 0;

void main()
{
   // CONFIFGURACION DE ADC
   setup_adc(adc_clock_div_2); // Sincronizamos las frecuencias

   set_tris_a(0b00001111); // 1 entrada 0 salida

   // CONFIGURACION DE PWM
   setup_timer_2(T2_DIV_BY_4, 249, 1); // Primer parametro
modificamos el preescalador
   setup_ccp1(CCP_PWM | CCP_PWM_HALF_BRIDGE, 10); // Activa el PMW
en P1A y P1B. (Activar PWM, Modo de puente H, Desfase (En funcion
de la cuenta del timer ))

   set_pwm1_duty(duty[1]);

   while(true)
   {
      cont++;

      if(input(BOT1)) // Si se presiona el boton 1 (PREDETERMINADO)
```

```
         MODO = 1;
      if(input(BOT2))
         MODO = 2;

      if((((int16) dist_act) != dist_prox) || (cont == 30))
      {
         cont = 0;

         // DISTANCIA ACTUAL
         setup_adc_ports(AN0_TO_AN1, VREF_VREF);
         set_adc_channel(1);
         delay_us(10);
         dist_act = read_adc();
         dist_act = 0.000000000052405283702418 71*pow(dist_act, 4) -
0.000000145087371979661 6*pow(dist_act, 3) +
0.000155220116880837 7*pow(dist_act, 2) - 0.088303023838816*dist_act
+ 35.902364490061590; // A partir del sensor obtenemos la distancia
actual
      }

      Display(dist_act);

      if(MODO == 1) // MODO 1 POTENCIOMETRO (PREDETERMINADO)
      {
         // DISTANCIA PROXIMA
         setup_adc_ports(AN0_TO_AN1);
         set_adc_channel(0); // Lectura del Potenciometro (1023 ->
0)
         delay_us(10);
         dist_prox = read_adc() / 1023.0 * (MAX - MIN) + MIN;
      }
      else if(MODO == 2) // MODO 2 BOTONES 3 Y 4
      {
         if(input(BOT3))
         {
            // DISTANCIA PROXIMA
            dist_prox = 15;
         }
         if(input(BOT4))
         {
            // DISTANCIA PROXIMA
            dist_prox = 25;
         }
      }
      else {}

      // Cada ciclo despues de establecer la distancia proxima,
hacemos acoplamiento

      if(((int16) dist_act) != dist_prox)
      {
         output_high(LED); // LED de acoplamiento
         Acoplamiento();
      }
      else
      {
         output_low(LED); // LED de acoplamiento
         set_pwm1_duty(duty[1]); // Detenemos el giro
```

```
      }

      //Display(dist_prox);
   }
}

void Acoplamiento()
{
   int direccion;

   // Establecemos la direccion de giro

   if(dist_act < dist_prox)
      direccion = 2; // Izquierda
   else if(dist_act > dist_prox)
      direccion = 0; // Derecha

   // Configuramos el giro
   set_pwm1_duty(duty[direccion]);
}

void Display(float numero) // Tarda 20 ms
{
   int punto, n1, n2, n3, n4;

   // Se descompone en 4 numeros
   Descomponer( &n1, &n2, &n3, &n4, numero);

   // Los 4 numeros se distribuyen en el display

   // Encendemos primer display
   output_high(D2);
   output_high(D3);
   output_high(D4);
   output_low(D1);

   if(numero < 10) // Un digito y tres decimales
      punto = 128; // Le sumamos para que el bit que activa el
punto se active
   else
      punto = 0;

   output_b(vec_c[n1] + punto);

   delay_ms(5);

   // Encendemos segundo display
   output_high(D1);
   output_high(D3);
   output_high(D4);
   output_low(D2);

   if(numero < 100 & !(numero < 10)) // Dos digitos y dos decimales
      punto = 128; // Le sumamos para que el bit que activa el
punto se active
   else
      punto = 0;
```

```
   output_b(vec_c[n2] + punto);

   delay_ms(5);

   // Encendemos tercer display
   output_high(D1);
   output_high(D2);
   output_high(D4);
   output_low(D3);

   if((numero < 1000) && !(numero < 100) && !(numero < 10)) // Tres
digitos y un decimal
      punto = 128; // Le sumamos para que el bit que activa el
punto se active
   else
      punto = 0;

   output_b(vec_c[n3] + punto);

   delay_ms(5);

   // Encendemos cuarto display
   output_high(D1);
   output_high(D2);
   output_high(D3);
   output_low(D4);

   output_b(vec_c[n4]);

   delay_ms(5);
}

void Descomponer(int *n1, int *n2, int *n3, int *n4, float numero)
// Buscar forma de acortar
{
   int16 entera;
   float aux, decimal;

   entera = (int16) numero; // Quitamos los decimales del numero
   decimal = numero - entera; // Quitamos la parte entera del
numero

   if(numero < 10)
   {
      *n1 = entera; // Sacamos las unidades
      aux = decimal * 10; // Separamos el primer decimal
      *n2 = (int) aux;
      aux -= (int) aux; // Borramos el primer decimal
      aux *= 10; // Separamos el segundo decimal
      *n3 = (int) aux;
      aux -= (int) aux; // Borramos el segundo decimal
      aux *= 10; // Separamos el tercer decimal
      *n4 = (int) aux;
   }
   else if(numero < 100)
   {
      *n1 = entera / 10; // Sacamos las decenas
      *n2 = entera - (*n1*10); // Sacamos las unidades
```

```
      aux = decimal * 10; // Separamos el primer decimal
      *n3 = (int) aux;
      aux -= (int) aux; // Borramos el primer decimal
      aux *= 10; // Separamos el segundo decimal
      *n4 = (int) aux;
   }
   else if(numero < 1000)
   {
      *n1 = entera / 100; // Sacamos las centenas
      *n2 = entera / 10;
      *n2 -= (*n1*10); // Sacamos las decenas
      *n3 = entera - ((*n1*100) + (*n2*10)); // Sacamos las
unidades
      aux = decimal * 10; // Separamos el primer decimal
      *n4 = (int) aux;
   }
   else if(numero < 10000)
   {
      *n1 = entera / 1000; // Sacamos los millares
      *n2 = (entera - (*n1*1000)) / 100; // Sacamos las centenas
      *n3 = entera / 10;
      *n3 -= ((*n1*100) + (*n2*10)); // Sacamos las decenas
      *n4 = entera - ((*n1*1000) + (*n2*100) + (*n3*10)); //
Sacamos las unidades
   }
}
```

Conclusiones de la práctica

Joel Zúñiga:

Cumplimos el objetivo de la práctica implementando casi todo lo visto en el curso, aprendí mucho de esta práctica y también la forma tan peculiar de trabajar con un sensor de distancia infrarrojo.

Daniela Manríquez:

En esta práctica reforcé los conocimientos adquiridos en la práctica pasada de cómo utilizar el PWM, apliqué los conocimientos vistos en métodos numéricos para obtener la ecuación con la que trabaja el sensor de proximidad. Así como también aprendí como es que funciona el sensor de proximidad Sharp y el puente H, el cual permite controlar el giro del motor DC.

Bibliografía

González J. (2001). Instrumentación Electrónica. Universidad de Cadiz: Boixareu Editores.

Grindling G. & Weiss B. (2007). Introduction to Microcontrollers. Vienna University of Technology.

(2014). Optoacoplador 4N28 Salida Transistor. 19/07/2016, de Carrod Electronica Sitio web: http://www.carrod.mx/products/optoacoplador-4n28-salida-transistor.