Nombre de la asignatura: Laboratorio de Programación de Avanzada

Maestro en Ciencias: Moisés Agustín Martínez Hernández

Nombre de la práctica: Voltaje de referencia y sensor de temperatura

Integrantes:

- Zuñiga Fragoso Diego Joel
- Manríquez Navarro Daniela del Carmen

Número de práctica: 5

Total de horas: 4 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 temperatura lm35.
- Implementar la lectura de 2 canales analógico para realizar una conversión de nivel de tensión con el propósito de medir, por medio del sensor, y mostrar la temperatura obtenida en un display así como hacer funcionar un motor de corriente directa regulando su velocidad conforme la temperatura.

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 lm35, en la otra entrada el voltaje dependerá de la temperatura que lea el sensor. La variación del voltaje se verá reflejada en la temperatura mostrada en un display controlado por un puerto de salida del microcontrolador, dicho display mostrará la temperatura que reciba el sensor dado decimales cuando la temperatura sea menor a 100 y sin decimales cuando sea 100 o más, todo dado en grados centígrados. También se incluirá un motor de corriente directa el cual funcionara todo el tiempo y su velocidad será regulada dependiendo de la temperatura que detecte el sensor.

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.

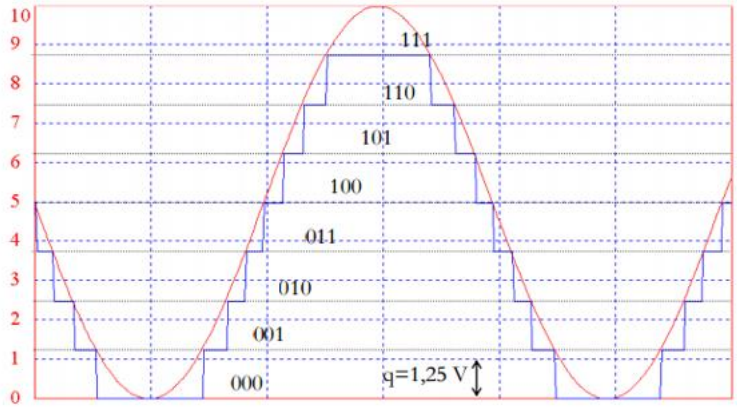


Figura. 1. Digitalización de una señal analógica por un convertidor A/D de 3 bits.

Sensor de temperatura LM35

El LM35 es un sensor de temperatura con una calibración de 1ºC de variación. A su salida genera una señal analógica de un voltaje diferente según la temperatura que esté captando en cada momento.

Por lo general, puede abarcar temperaturas de medición de entre -55ºC y 150ºC, también puede medir temperaturas muy frecuentes. El rango de temperaturas está limitado por la cantidad de voltajes variables que puede tener a su salida, que van desde los -550mV hasta los 1500mV. Es decir, cuando está midiendo una temperatura de 150ºC ya sabemos que va a dar 1500mV en su salida. Mientras que si tenemos -550mV quiere decir que está midiendo -55ºC.

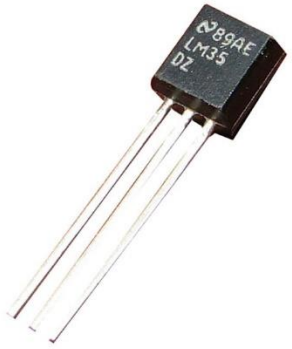


Figura 2. Sensor de temperatura LM35

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

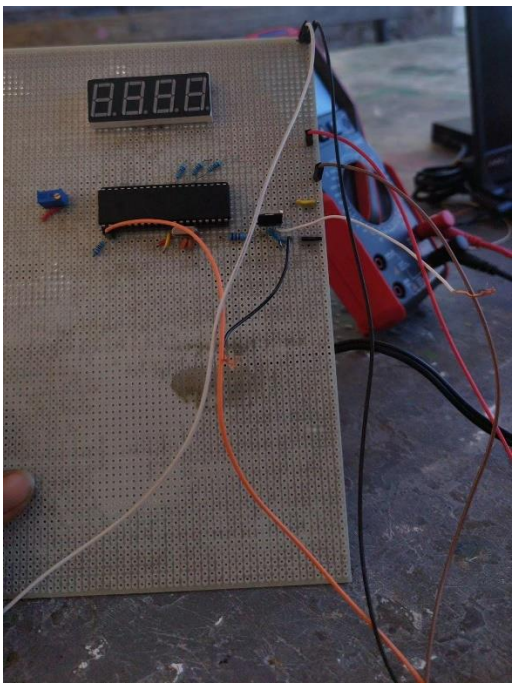
EQUIPO Y MATERIALES

- Software para programar microcontrolador.
- Software de simulación.
- Microcontrolador (sistema mínimo).
- Resistencia variable (trimpot, etc.).
- Transistor.
- 4 resistencias de 220 o 330 ohms.
- 1 motor de corriente directa.
- 1 display.

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.
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 temperatura en el display, así como mandando el voltaje necesario para que el motor esté funcionando en función a la temperatura.
   b) Cuando la temperatura cambie se verá reflejado en el display así como en la velocidad a la que funciona el 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.

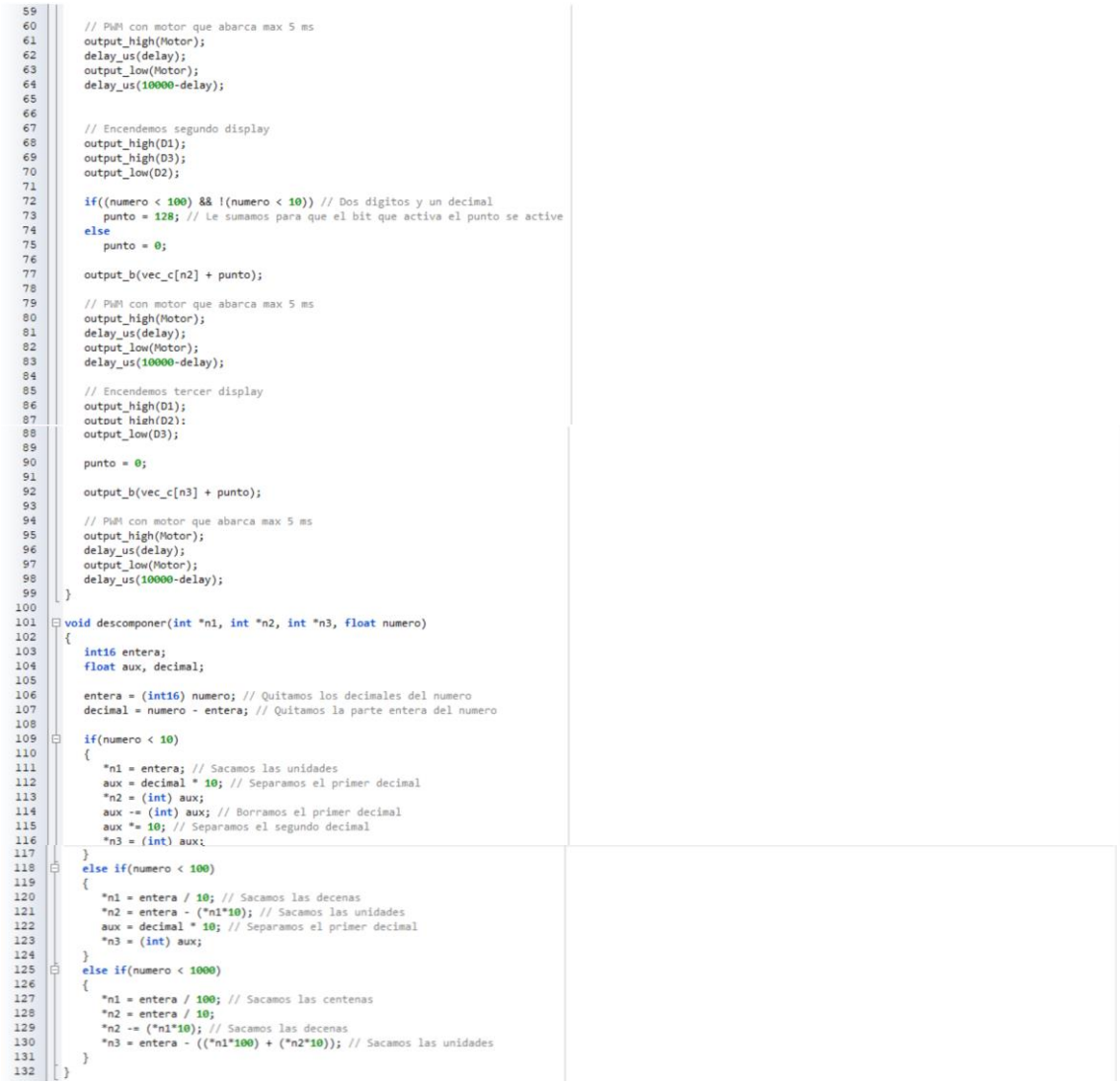
Resultados de la práctica



Circuito armando

Código explicado

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

#define D1 pin_d4
#define D2 pin_d5
#define D3 pin_d6
#define Motor pin_d0

void display(float);
void descomponer(int *, int *, int *, float);

int vec_c[10] = {0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D,0x7D,0x07,0x7F,0x67}; // Numeros para display Catodo
int16 delay;

void main()
{
   float data;

   setup_adc(adc_clock_div_2); // Sincronizamos las frecuencias
   setup_adc_ports(AN0_TO_AN1 | VSS_VREF); // Hay un sensor en AN0 y AN1. Establecemos el rango de los bits de resolucion, dependendiendo un voltaje de referencia (Vo
   set_tris_a(0b00000011); // 1 entrada 0 salida
   set_adc_channel(0);
   delay_us(10);

   while(true)
   {
      data = read adc(); // Obtiene la lectura del sensor
      data *= (150.0/1023.0); // Convierte los bits a grados centigrados

      for(int i = 0; i < 34; i++) // Hace que se actualize cada segundo
         display(data); // Recibe e imprime la temperatura. Tambien gira el motor en funcion de esta. (Tarda 30 ms)
   }
}

void display(float numero) // Tarda 15 ms
{
   int punto, n1, n2, n3;

   delay = (9000.0 / 150.0) * numero + 1000; // Convertimos la temperatura en un delay entre 2,000 y 10,000 us

   // Se descompone en 3 numeros
   descomponer( &n1, &n2, &n3, numero);

   // Los 3 numeros se distribuyen en el display

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

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

   output b(vec c[n1] + punto);
```



```
    // PWM con motor que abarca max 5 ms
    output_high(Motor);
    delay_us(delay);
    output_low(Motor);
    delay_us(10000-delay);


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

    if((numero < 100) && !(numero < 10)) // Dos 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[n2] + punto);

    // PWM con motor que abarca max 5 ms
    output_high(Motor);
    delay_us(delay);
    output_low(Motor);
    delay_us(10000-delay);

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

    punto = 0;

    output_b(vec_c[n3] + punto);

    // PWM con motor que abarca max 5 ms
    output_high(Motor);
    delay_us(delay);
    output_low(Motor);
    delay_us(10000-delay);
}

void descomponer(int *n1, int *n2, int *n3, float numero)
{
    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;
    }
    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;
    }
    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
    }
}
```

Conclusiones de la práctica

Joel Zúñiga:

En esta práctica utilicé por primera vez un sensor LM35 y también soldé por primera vez, fue la más difícil que he hecho, pero me ayudo a familiarizarme más con el uso de displays y ADC. Aunque hubo complicaciones se cumplió con el objetivo de la práctica satisfactoriamente.

Daniela Manríquez:

En esta práctica reforcé los conocimientos adquiridos en la práctica pasada de cómo utilizar el microcontrolador para recibir e interpretar las señales analógicas. Así como también aprendí como es que funciona el sensor de temperatura LM35 el cual da 10 mv por cada grado centígrado, con esto pude manipular a través del código la velocidad del motor.

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.