M16

Universidad Nacional Autónoma de México

Facultad de Ingeniería

Semestre 2023-2

Laboratorio de Microcomputadoras

Proyecto 3:

Convertidor Analógico Digital

Profesor:

M.I. Ruben Anaya García

Alumnos:	Número de cuenta:
Barreiro Valdez Alejandro	317520888
Gil Márquez Arath Emiliano	317083875
Herrera Carrillo Cristhian	317094662
Zepeda Baeza Jessica	317520747

Grupo Teoría: 1

Fecha de realización: 7 de mayo de 2023

Fecha de entrega: 12 de mayo de 2023

Introducción

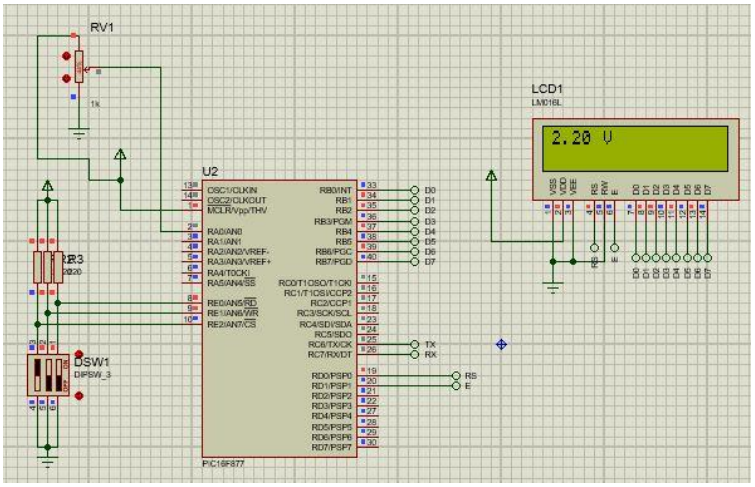
En este proyecto se hizo uso del convertidor analógico digital del microcontrolador PIC16F877. Se utilizó el convertidor junto con el display del proyecto pasado para mostrar la entrada de voltaje que suministra un potenciómetro. En este informe se mostrarán los diferentes modos de salida que existen y qué significa cada uno.

Un convertidor analógico-digital (ADC, por sus siglas en inglés) es un dispositivo que convierte una señal analógica continua en una señal digital discreta. En otras palabras, el ADC toma una señal analógica, como la tensión de una corriente eléctrica, y la convierte en una secuencia de valores digitales discretos que pueden ser procesados por un ordenador o cualquier otro dispositivo digital.
El proceso de conversión analógica-digital se realiza en dos pasos principales: muestreo y cuantificación. El muestreo implica tomar mediciones de la señal analógica a intervalos regulares de tiempo, y la cuantificación implica asignar un valor digital discreto a cada una de estas mediciones. La precisión del ADC se mide en bits, y se refiere al número de valores discretos diferentes que el ADC puede representar.

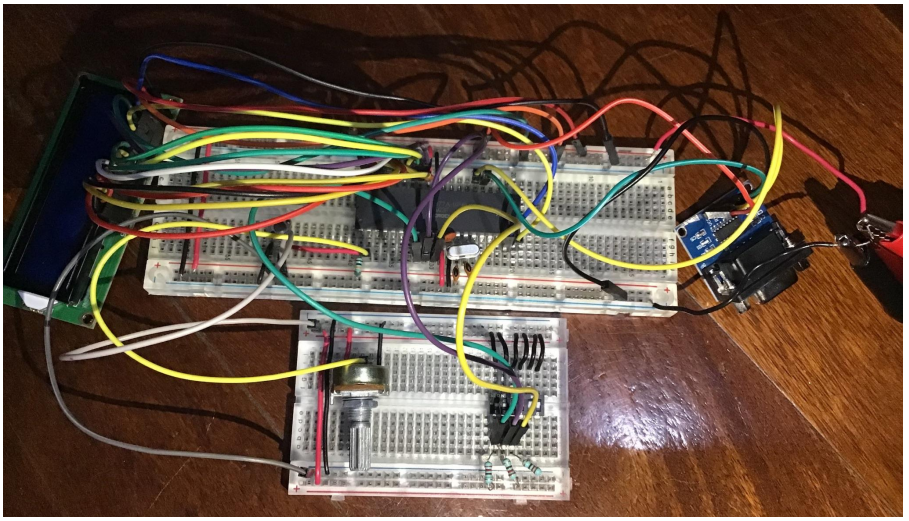
El convertidor analógico digital permite recibir señales analógicas y las convierte a información de 10 bits. Existen 8 canales que se pueden utilizar para procesar las señales analógicas. Para cambiar el puerto a analógico se utiliza el registro ADCON1. Posteriormente, se debe configurar la fuente de reloj, el canal de entrada y prender el convertidor utilizando el registro ADCON0. Después, se inicia y se espera a que se realice la conversión. Por último, se tendrá el resultado en el registro ADRESH. Este proceso logra la conversión.

Desarrollo

Lo primero que se desarrolló fue el circuito necesario para este proyecto. Este proyecto utiliza los componentes de los proyectos pasados y añade un potenciómetro como entrada. Se utilizó un *dipswitch* de tres entradas y el display de cristal líquido. Lo único nuevo que se aprendió a alambrar fue el potenciómetro el cual tiene tres pines. Uno se utiliza para el voltaje de entrada, otro para el voltaje de salida y el último se debe conectar a tierra.
El código que se realizó en este proyecto debía realizar diversas tareas a partir del modo en el *dipswitch* que se selecciona. Se debe mostrar el valor del convertidor analógico digital en el display de cristal líquido de cinco maneras: en binario, en decimal, en hexadecimal, convertido a voltaje y de manera gráfica con una pila. Además, se realizó la simulación en Proteus para entender cómo tenía que ser el diagrama del proyecto funcionando.



Después de la simulación que se realizó en Proteus se alambraron los elementos necesarios para el proyecto y se probó el código. Los resultados fueron los esperados y se logró realizar cada uno de los modos pedidos. El circuito que se alambró para este proyecto fue el siguiente.



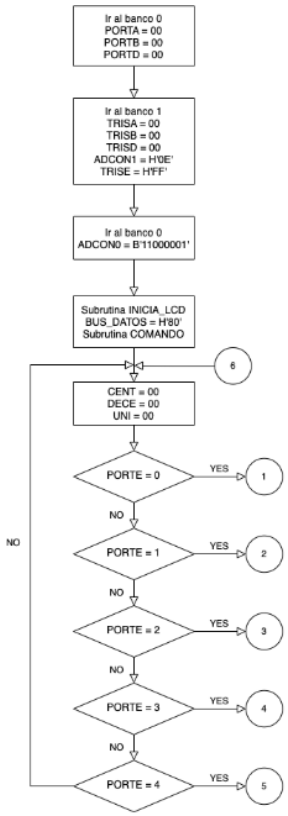
Algoritmos

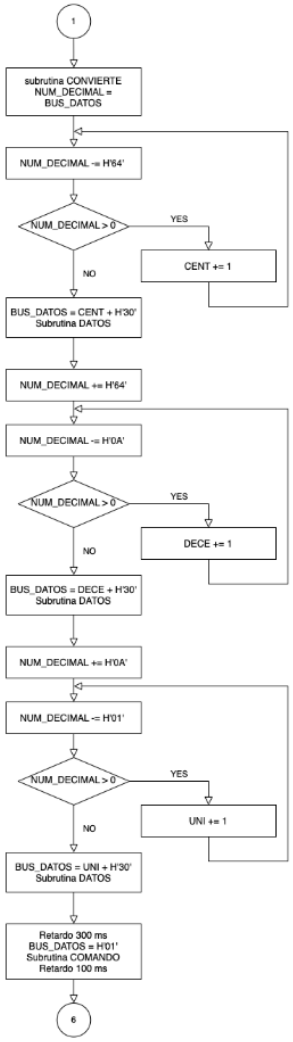
Para este proyecto se realizó una carta ASM que se divide en 7:

1. El programa principal
2. La opción 0 del Switch de 3 bits: Número decimal
3. La opción 1 del Switch de 3 bits: Número hexadecimal
4. La opción 2 del Switch de 3 bits: Número binario
5. La opción 3 del Switch de 3 bits: Voltaje suministrado
6. La opción 4 del Switch de 3 bits: Carácter de Pila

7. Tres subrutinas creadas: INC_CENT, INC_DECE y CONVIERTE

En la carta del programa principal se hacen las configuraciones de los puertos paralelos, definiendo el A, B y D como salidas y E como entrada. Se configura ADCON1 con el valor 0E para definir la terminal A0 como analógico (irá conectada al potenciómetro) y las demás terminales de A y el puerto E (irá conectado al switch) como digitales y se realiza la subrutina de inicialización del LCD. Por último, se verifica el valor del Switch de 3 bits, comparándolo con 0, 1, 2, 3, y 4. Cuando se encuentra su valor se hace un salto a otra parte del código que implementa la acción a realizar. Una vez que se realiza o cuando el valor no corresponde a las opciones indicadas, regresa a verificar el valor del Switch.

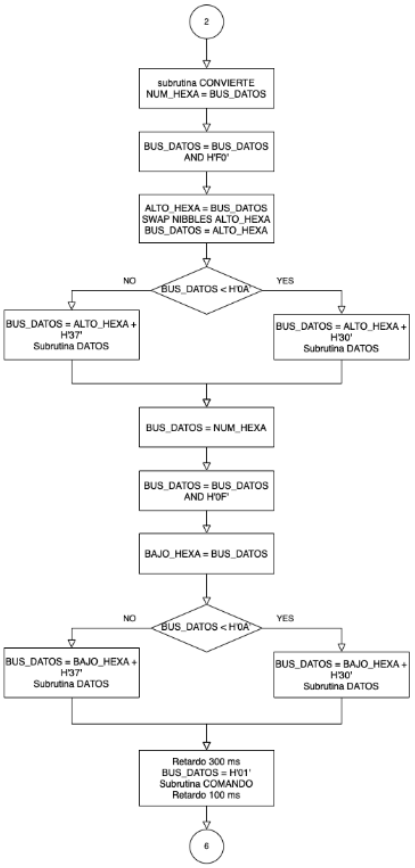




La opción 0 muestra el valor analógico del voltaje suministrado como un número decimal. Primero llama a la subrutina CONVIERTE para obtener el número analógico en el bus de datos y lo guarda en una variable. Para encontrar las centenas le resta 100 (64 hexadecimal) a la variable hasta que dicha variable sea negativa, incrementando en 1 la variable CENT cada vez que hace la resta y no se ha llegado al valor negativo. Una vez que se obtiene el valor negativo, se le suma H’30’ a la variable CENT para obtener el valor

ASCII del número en la variable y se manda al bus de datos para llamar la subrutina DATOS y que se muestre en el Display.

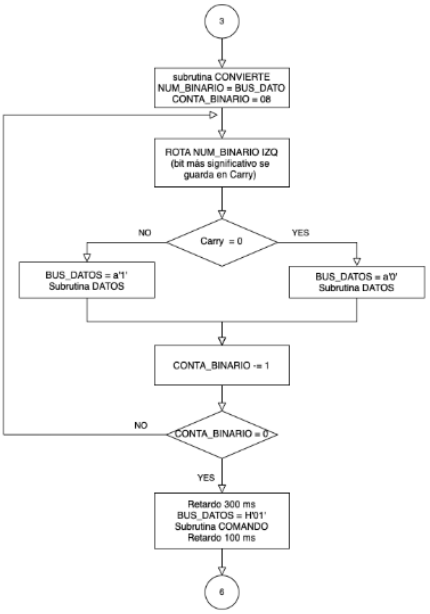
Después se le suma 100 a la variable para regresar al valor positivo y ahora se repite el proceso pero para encontrar las decenas. Para ello, se le resta de 10 en 10 (0A hexadecimal) y el contador de las restas se guarda en DECE. De igual forma, al llegar al valor negativo, se obtiene el ASCII del número en DECE y se manda al Display. Por último se repite el proceso para las unidades, restando de 1 en 1 y guardando el contador en UNI. Cuando se llega al valor negativo se encuentra su ASCII y se manda al Display. Por último se llama un retardo de 300ms, luego se manda al bus de datos 01 para llamar la subrutina COMANDO y por último se hace un retardo de 100ms. Finalmente regresa al programa principal a verificar el valor del Switch.



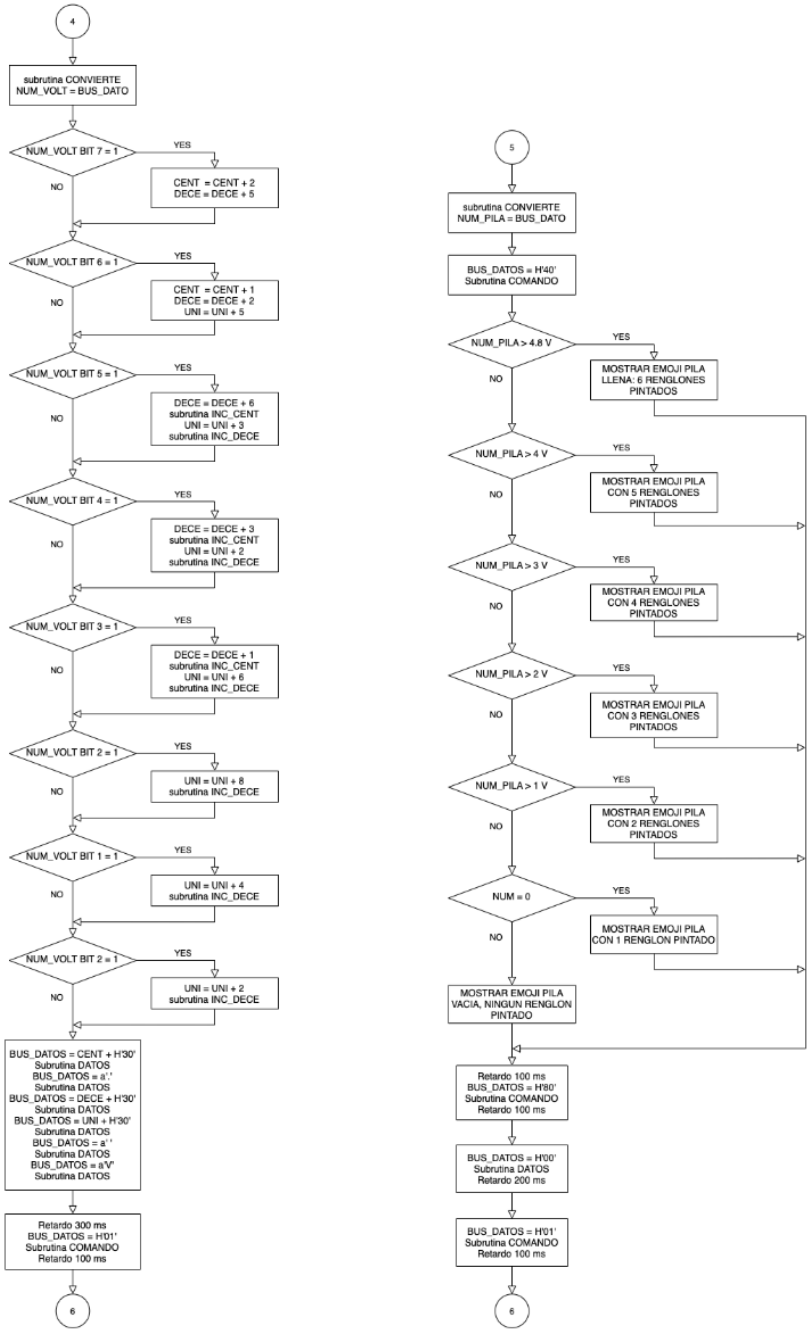
La opción 1 muestra el valor analógico del voltaje suministrado como un número hexadecimal. Primero llama a la subrutina CONVIERTE para obtener el número analógico en el bus de datos y lo guarda en una variable. Se hace un AND entre el bus de datos y F0 para obtener solo los primeros 4 bits del número ingresado. Y se pasan a otra variable ALTO_HEXA y para hacer un swap de nibbles y pasar el valor a su parte baja. Se manda al bus de datos y se verifica si este valor es menor a A hexadecimal para obtener su valor en ASCII correcto. Si es mayor o igual se le suma H’37’ al bus de datos (para una letra ASCII)

y si es menor se le suma H’30’ (para un número ASCII). Después se llama la subrutina DATOS.

Después se vuelve a pasar el valor del número ingresado completo al bus de datos y se hace un AND con 0F para ahora obtener la mitad baja del número. Se verifica si es menor a A y de igual forma se suma H’30’ si sí lo es y H’37’ si es mayor o igual. Después se llama la subrutina DATOS para mandarlo al Display. Por último, se llama un retardo de 300ms, luego se manda al bus de datos 01 para llamar la subrutina COMANDO y por último se hace un retardo de 100ms. Finalmente regresa al programa principal a verificar el valor del Switch.

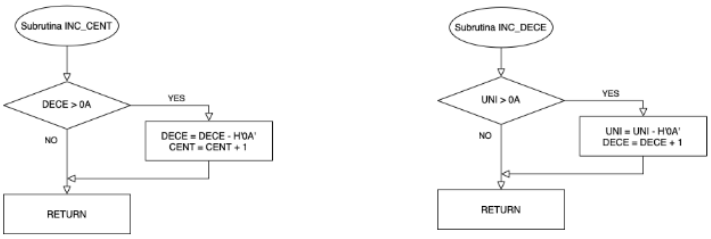


La opción 2 muestra el valor analógico del voltaje suministrado como un número binario. Primero llama a la subrutina CONVIERTE para obtener el número analógico en el bus de datos, lo guarda en una variable e inicializa un contador en 08. Se hace un ciclo donde se hace un corrimiento a la izquierda de la variable, lo que va a ocasionar que el bit más significativo se almacene en Carry. Después se analiza si Carry es 0 porque en ese caso se manda al bus de datos el ASCII de 0 y en caso contrario se manda el ASCII de 1. Después se llama la subrutina DATOS y se decrementa en 1 el contador. En caso de que el contador no sea cero aún se repite el ciclo. Cuando llega a 0 se llama un retardo de 300ms, luego se manda al bus de datos 01 para llamar la subrutina COMANDO y por último se hace un retardo de 100ms. Finalmente regresa al programa principal a verificar el valor del Switch.

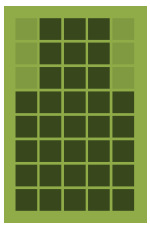
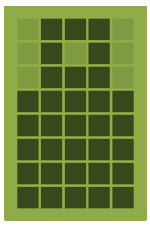
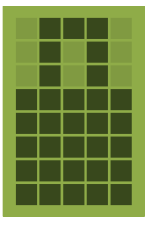
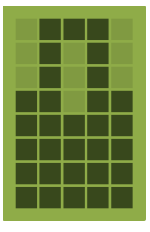
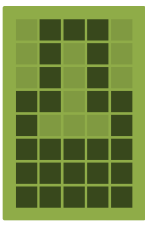
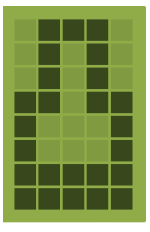
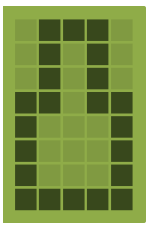


La opción 3 muestra el valor analógico del voltaje suministrado en el formato #.## V. Dado que cada bit del valor convertido está asignado a un voltaje, se verifica qué bits están prendidos empezando por el más significativo y en caso de estarlos se suma el respectivo valor asignado a ese bit. Se divide el número a sumar en centenas, decenas y unidades. Entonces primero llama a la subrutina CONVIERTE para obtener el número analógico en el bus de datos y lo guarda en una variable. Después se verifica si el bit 7 es 1, en caso de serlo se deben sumar 2.5 V entonces se suma 2 a CENT y 5 a DECE. Para el bit 6, se suma 1 a CENT, 2 a DECE y 5 a UNI (1.25 V). Este proceso se repite por cada bit con la mitad del voltaje anterior. A partir del bit 5, al sumar a DECE y UNI se mandan a llamar las

subrutinas INC_CENT e INC_DECE respectivamente. Estas subrutinas verifican que la suma haya obtenido un resultado mayor o igual 0A y en ese caso, incrementan en 1 a las centenas o decenas y a la variable actual le restan 0A. Esto con el fin de obtener un número decimal al que se le pueda sumar H'30' para obtener su ASCII al final de todas las comparaciones. Una vez que se tiene el valor final de DECE, CENT y UNI, se manda el ASCII de CENT, de '.', de DECE, de UNI, de ' ' y finalmente de 'V' al LCD llamando la subrutina DATOS después de cada caracter. Se llama un retardo de 300ms, luego se manda al bus de datos 01 para llamar la subrutina COMANDO y por último se hace un retardo de 100ms. Finalmente regresa al programa principal a verificar el valor del Switch.

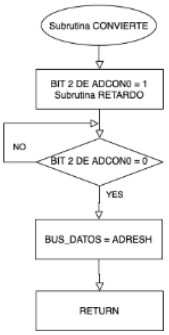


La opción 4 muestra un caracter en forma de pila que se ve más o menos llena dependiendo el valor del voltaje suministrado. Para ello, al encontrar el rango de valores entre los que se encuentra el voltaje se manda cada fila del caracter de pila como un valor binario al bus de datos y se llama la subrutina DATOS. Lo primero al entrar en esta opción es llamar a la subrutina CONVIERTE para obtener el número analógico en el bus de datos y lo guarda en una variable. Se manda al bus de datos el valor H'40' para activar la CGRAM. Se verifica después entre qué valores está el voltaje empezando la comparación con el mayor voltaje y en caso de que sea menor se hace la comparación con el siguiente voltaje pero en caso de que se mayor se mandan los respectivos datos al LCD, como muestra la siguiente tabla.

> 4.8 V	4 - 4.8 V	3 - 4 V	2 - 3 V	1 - 2 V	0 - 1 V	0 V
						

Al terminar de mandar el caracter al LCD, se llama un retardo de 100ms, se manda al bus de datos H'80' para llamar la subrutina COMANDO junto con otro retardo de 100ms. Después se manda al bus de datos H'00' para llamar a la subrutina DATOS y a un retardo de 200ms para ver el caracter en el Display. Lo último que se manda al bus de datos es

H’01’ para llamar la subrutina COMANDO y a un retardo de 100ms para limpiar la pantalla. Finalmente regresa al programa principal a verificar el valor del Switch.



Por último, la subrutina CONVIERTE obtiene el valor analógico del voltaje suministrado. Básicamente le indica al convertidor que puede iniciar el proceso de obtener los 10 bits que indican el voltaje al encender el bit 2 de ADCON0. Llama una subrutina de retardo y verifica si el bit 2 de ADCON0 está apagado, lo que indica que acabó de convertir. Si aún no está apagado lo espera volviendo a verificar y si ya está apagado manda el valor convertido en ADRESH al bus de datos.

Programa comentado

Este programa a base de un sistema mínimo con un microcontrolador PIC16F877A, realiza varias acciones dependiendo de los valores de entrada que reciba de un Dip Switch conectado a los puertos del microcontrolador y mediante la configuración de los registros ADCON0 y ADCON1 es posible realizar la conversión de una señal analogica a digital recibida del voltaje generado mediante la manipulación de un potenciómetro. Con la información obtenida se puede realizar conversiones de números decimales, binarios y hexadecimales, además de generar el valor del voltaje con punto decimal y su representación con el carácter de una pila, que podemos apreciar en la pantalla de un LCD configurado por medio de comandos dentro del programa.

```
PROCESSOR 16f877            ;Procesador a utilizar
INCLUDE <p16f877.INC>

VALOR equ h'20'             ;1er valor para el retardo
VALOR1 equ h'21'    ;2do valor para el retardo
VALOR2 equ h'22'    ;3er valor para el retardo

CENT EQU h'23'              ;almacena el dígito de las centenas
DECE EQU h'24'              ;almacena el dígito de las decenas
UNI EQU h'25'               ;almacena el dígito de las unidades
NUM_DECIMAL EQU h'26'   ;almacena el número a convertir a decimal

NUM_HEXA EQU h'27'          ;almacena el número a convertir a hexadecimal
ALTO_HEXA EQU h'28'         ;almacena el dígito alto hexadecimal
BAJO_HEXA EQU h'29'         ;almacena el dígito bajo hexadecimal
```

```
NUM_BINARIO EQU h'2A'              ;almacena el número a convertir a binario
CONTA_BINARIO EQU h'2B'            ;contador para la conversión a binario

NUM_VOLT EQU h'2C'
V48 equ B'11111010'          ;Valor calculado de 4.8V
V4 equ B'11010000'           ;Valor calculado de 4V
V3 equ B'10011100'           ;Valor calculado de 3V
V2 equ B'01101000'           ;Valor calculado de 2V
V1 equ B'00110100'           ;Valor calculado de 1V

NUM_PILA EQU h'2D'

ORG 0                              ;vector de reset
GOTO INICIO                        ;salto al inicio del programa

ORG 5                              ;Origen del programa



INICIO:
   CLRF PORTA      ;limpia Puerto A
   CLRF PORTB      ;limpia Puerto B
   CLRF PORTD     ;limpia Puerto D

   BSF STATUS, 5              ;BIT 5 DE STATUS = 1
   BCF STATUS, 6              ;BIT 6 DE STATUS = 0, BANCO 01
   CLRF TRISB                 ;Puerto B como salida
   CLRF TRISA                 ;Bits 1-5 Puerto A como salida
   CLRF TRISD                 ;Bits 0-7 Puerto D como salida
   BSF TRISA,0                ;Bit 0 Puerto A como entrada

   MOVLW 0X0E                 ;W = 0E
   MOVWF ADCON1               ;ADCON1 = 0E -> Puerto A (bits 1 a 5) y E digitales, Puerto A (bit 0) analogico

   BSF TRISE, 0X00
   BSF TRISE, 0X01
   BSF TRISE, 0X02            ;Puerto E como entrada

   BCF STATUS, 5              ;BIT 5 DE STATUS = 0, BANCO 00
   MOVLW B'11000001'          ;W = 11000001
   MOVWF ADCON0               ;Configura ADCON0 con W: usará reloj interno, canal 0, DONE, conv.A/D encendido

   CALL INICIA_LCD            ;inicialización del LCD
   MOVLW 0X80                 ;W = 80
   CALL COMANDO               ;manda un control a LCD

SWITCH_3                      ;Verifica el valor del Switch de 3 bits
   MOVLW 0X00                 ;W = 00
   MOVWF CENT                 ;CENT = 0
   MOVWF DECE                 ;DECE = 0
   MOVWF UNI                  ;UNI = 0

   MOVF PORTE, W              ;W = PORTE
   ;SUBLW 0X00                ;W = W - 0
   BTFSC STATUS, Z            ;Si Z = 0 salta
   GOTO DECIMAL               ;Z = 1: salta a opción 0 (decimal)
   GOTO OPCION1               ;Z = 0: PORTE no es 0

OPCION1
   MOVF PORTE, W              ;W = PORTE
   XORLW 0X01                 ;W = W XOR 1
   BTFSC STATUS, Z     ;Si Z = 0 salta
   GOTO HEXA                  ;Z = 1: opción 1 (hexadecimal)
   GOTO OPCION2               ;Z = 0: PORTE no es 1
```

```
OPCION2
   MOVF PORTE, W          ;W = PORTE
   XORLW 0X02             ;W = W XOR 2
   BTFSC STATUS, Z        ;Si Z = 0 salta
   GOTO BINARIO           ;Z = 1: opción 2 (binario)
   GOTO OPCION3           ;Z = 0: PORTE no es 2

OPCION3
   MOVF PORTE, W          ;W = PORTE
   XORLW 0X03             ;W = W XOR 3
   BTFSC STATUS, Z        ;Si Z = 0 salta
   GOTO VOLTAJE           ;Z = 1: opción 3 (voltaje)
   GOTO OPCION4           ;Z = 0: PORTE no es 3

OPCION4
   MOVF PORTE, W          ;W = PORTE
   XORLW 0X04             ;W = W XOR 4
   BTFSC STATUS, Z        ;Si Z = 0 salta
   GOTO PILA              ;Z = 1: opcion 4 (pila)
   GOTO SWITCH_3          ;Vuelve a SWITCH_3 para volver a verificar su valor



DECIMAL
   CALL CONVIERTE         ;Configuración para el LCD
   MOVWF NUM_DECIMAL      ;NUM_DECIMAL = W
   GOTO CENTENAS

CENTENAS
   MOVLW 0X64             ;W = 64 = 100
   SUBWF NUM_DECIMAL      ;NUM_DECIMAL - 100
   BTFSC STATUS, C        ;Si C = 0 salta
   INCF CENT              ;CENT + 1 (C = 1)
   BTFSS STATUS, C        ;Si C = 1 salta
   GOTO CARGARCENTENA     ;C = 0
   GOTO CENTENAS          ;C = 1

CARGARCENTENA
   MOVF CENT, W           ;W = CENT
   ADDLW 0X30

   CALL DATOS             ;Comando para mostrar el carácter del numero en el LCD
   MOVLW 0X64
   ADDWF NUM_DECIMAL      ;Se suma un 64 = 100 a NUM_DECIMAL
   GOTO DECENAS

DECENAS
   MOVLW 0X0A             ;W = 0A = 10
   SUBWF NUM_DECIMAL      ;NUM_DECIMAL - 10
   BTFSC STATUS, C        ;Si C = 0 salta
   INCF DECE              ;CENT + 1 (C = 1)
   BTFSS STATUS, C        ;Si C = 1 salta
   GOTO CARGARDECENA      ;C = 0
   GOTO DECENAS           ;C = 1

CARGARDECENA
   MOVF DECE, W           ;W = DECE
   ADDLW 0X30

   CALL DATOS             ;Comando para mostrar el carácter del numero en el LCD
   MOVLW 0X0A
   ADDWF NUM_DECIMAL      ;Se suma un A0 = 10 a NUM_DECIMAL
   GOTO UNIDADES

UNIDADES
```

```
   MOVLW 0X01              ;W = 01 = 1
   SUBWF NUM_DECIMAL   ;NUM_DECIMAL - 10
   BTFSC STATUS, C         ;Si C = 0 salta
   INCF UNI                ;CENT + 1 (C = 1)
   BTFSS STATUS, C         ;Si C = 1 salta
   GOTO CARGARUNIDAD   ;C = 0
   GOTO UNIDADES           ;C = 1

CARGARUNIDAD
   MOVF UNI, 0X00          ;W = UNI
   ADDLW 0X30

   CALL DATOS              ;Comando para mostrar el carácter del numero en el LCD
   CALL RET100MS
   CALL RET100MS
   CALL RET100MS           ;Llamado a subrutinas para retardos
   MOVLW H'01'
   CALL COMANDO            ;Comando para limpiar pantalla del LCD
   CALL RET100MS

   GOTO SWITCH_3           ;Vuelve a SWITCH_3 para volver a verificar su valor



HEXA
   CALL CONVIERTE          ;Configuración para el LCD
   MOVWF NUM_HEXA          ;NUM_HEXA = W
   ANDLW 0XF0              ;Operación AND entre W y F0
   MOVWF ALTO_HEXA         ;ALTO_HEXA = W
   SWAPF ALTO_HEXA         ;Intercambia los nibbles
   MOVLW 0X0A
   SUBWF ALTO_HEXA, W  ;ALTO_HEXA - W
   BTFSS STATUS, C         ;Si C = 1 salta
   GOTO NUMERO_ALTO    ;C = 0
   GOTO LETRA_ALTO         ;C = 1

;Calcula la letra de los primeros 4 bits para mostrarlo en el LCD
LETRA_ALTO
   MOVF ALTO_HEXA, W
   ADDLW 0X37
   CALL DATOS
   GOTO BAJO

;Calcula el número de los primeros 4 bits para mostrarlo en el LCD
NUMERO_ALTO
   MOVF ALTO_HEXA, W
   ADDLW 0X30
   CALL DATOS
   GOTO BAJO

;Últimos 4 bits
BAJO
   MOVF NUM_HEXA, W        ;NUM_HEXA = W
   ANDLW 0X0F              ;Operacion AND entre W y 0F
   MOVWF BAJO_HEXA         ;BAJO_HEXA = W
   MOVLW 0X0A
   SUBWF BAJO_HEXA, W  ;BAJO_HEXA - W
   BTFSS STATUS, C         ;Si C = 1 salta
   GOTO NUMERO_BAJO    ;C = 0
   GOTO LETRA_BAJO         ;C = 1

;Calcula la letra de los últimos 4 bits para mostrarlo en el LCD
LETRA_BAJO
   MOVF BAJO_HEXA, W
   ADDLW 0X37
```

```
   CALL DATOS              ;Comando para mostrar el carácter en el LCD
   CALL RET100MS
   CALL RET100MS
   CALL RET100MS           ;Llamado a subrutinas para retardos
   MOVLW H'01'
   CALL COMANDO            ;Comando para limpiar pantalla del LCD
   CALL RET100MS
   GOTO SWITCH_3           ;Vuelve a SWITCH_3 para volver a verificar su valor

;Calcula el numero de los últimos 4 bits para mostrarlo en el LCD
NUMERO_BAJO
   MOVF BAJO_HEXA, W
   ADDLW 0X30
   CALL DATOS              ;Comando para mostrar el carácter en el LCD
   CALL RET100MS
   CALL RET100MS
   CALL RET100MS           ;Llamado a subrutinas para retardos
   MOVLW H'01'
   CALL COMANDO            ;Comando para limpiar pantalla del LCD
   CALL RET100MS
   GOTO SWITCH_3           ;Vuelve a SWITCH_3 para volver a verificar su valor


BINARIO
   CALL CONVIERTE          ;Configuración para el LCD
   MOVWF NUM_BINARIO       ;NUM_BINARIO = W

   MOVLW 0X08
   MOVWF CONTA_BINARIO     ;COMTA_BINARIO = 8

LOOPBIN
   RLF NUM_BINARIO         ;Rota los bits de NUM_BINARIO hacia la izquierda
   BTFSS STATUS, C         ;Si C = 1 salta
   GOTO ZERO               ;C = 0
   GOTO ONE                ;C = 1

;Mustra un 0 en pantalla del LCD
ZERO
   MOVLW a'0'
   CALL DATOS              ;Comando para mostrar el carácter en el LCD
   DECFSZ CONTA_BINARIO    ;CONTA_BINARIO - 1
   GOTO LOOPBIN
   CALL RET100MS
   CALL RET100MS
   CALL RET100MS           ;Llamado a subrutinas para retardos
   MOVLW H'01'
   CALL COMANDO            ;Comando para limpiar pantalla del LCD
   CALL RET100MS
   GOTO SWITCH_3           ;Vuelve a SWITCH_3 para volver a verificar su valor

;Mustra un 1 en pantalla del LCD
ONE
   MOVLW a'1'
   CALL DATOS              ;Comando para mostrar el carácter en el LCD
   DECFSZ CONTA_BINARIO    ;CONTA_BINARIO - 1
   GOTO LOOPBIN
   CALL RET100MS
   CALL RET100MS
   CALL RET100MS           ;Llamado a subrutinas para retardos
   MOVLW H'01'
   CALL COMANDO            ;Comando para limpiar pantalla del LCD
   CALL RET100MS
   GOTO SWITCH_3           ;Vuelve a SWITCH_3 para volver a verificar su valor
```

```
;Voltaje en pantalla LCD
VOLTAJE
   CALL CONVIERTE          ;Configuracion para el LCD
   MOVWF NUM_VOLT          ;NUM_VOLT = W

   BTFSS NUM_VOLT,7        ;Bit 7 = 1 salta
   GOTO V125               ;B7 = 0
   MOVLW 0X02              ;W = 2 (B7 = 1)
   ADDWF CENT              ;CENT + W
   MOVLW 0X05              ;W = 5
   ADDWF DECE              ;DECE + W

V125
   BTFSS NUM_VOLT,6        ;Bit 6 = 1 salta
   GOTO V063               ;B6 = 0
   MOVLW 0X01              ;W = 1 (B6 = 1)
   ADDWF CENT              ;CENT + W
   MOVLW 0X02              ;W = 2
   ADDWF DECE              ;DECE + W
   MOVLW 0X05              ;W = 5
   ADDWF UNI               ;UNI + W

V063
   BTFSS NUM_VOLT,5        ;Bit 5 = 1 salta
   GOTO V032               ;B5 = 0
   MOVLW 0X06              ;W = 6 (B5 = 1)
   ADDWF DECE              ;DECE + W
   CALL INC_CENT           ;Llamado a INC_CENT
   MOVLW 0X03              ;W = 3
   ADDWF UNI               ;UNI + W
   CALL INC_DECE           ;Llamado a INC_DECE

V032
   BTFSS NUM_VOLT,4        ;Bit 4 = 1 salta
   GOTO V016               ;B4 = 0
   MOVLW 0X03              ;W = 3 (B4 = 1)
   ADDWF DECE              ;DECE + W
   CALL INC_CENT           ;Llamado a INC_CENT
   MOVLW 0X02              ;W = 2
   ADDWF UNI               ;UNI + W
   CALL INC_DECE           ;Llamado a INC_DECE

V016
   BTFSS NUM_VOLT,3        ;Bit 3 = 1 salta
   GOTO V008               ;B3 = 0
   MOVLW 0X01              ;W = 1 (B3 = 1)
   ADDWF DECE              ;DECE + W
   CALL INC_CENT           ;Llamado a INC_CENT
   MOVLW 0X06              ;W = 6
   ADDWF UNI               ;UNI + W
   CALL INC_DECE           ;Llamado a INC_DECE

V008
   BTFSS NUM_VOLT,2        ;Bit 2 = 1 salta
   GOTO V004               ;B2 = 0
   MOVLW 0X08              ;W = 8 (B2 = 1)
   ADDWF UNI               ;UNI + W
   CALL INC_DECE           ;Llamado a INC_DECE

V004
   BTFSS NUM_VOLT,1        ;Bit 1 = 1 salta
   GOTO V002               ;B1 = 0
   MOVLW 0X04              ;W = 4 (B1 = 1)
   ADDWF UNI               ;UNI + W
   CALL INC_DECE           ;Llamado a INC_DECE
```

```
V002
   BTFSS NUM_VOLT,0        ;Bit 0 = 1 salta
   GOTO FINAL_VOLT         ;B0 = 0
   MOVLW 0X02              ;W = 2 (B0 = 1)
   ADDWF UNI               ;UNI + W
   CALL INC_DECE           ;Llamado a INC_DECE

;Cargando los numeros del voltaje al LCD
FINAL_VOLT
   MOVF CENT, W
   ADDLW 0X30
   CALL DATOS      ;Comando para mostrar el valor de CENT en el LCD
   MOVLW a'.'
   CALL DATOS      ;Comando para mostrar el . en el LCD
   MOVF DECE, W
   ADDLW 0X30
   CALL DATOS      ;Comando para mostrar el valor de DECE en el LCD
   MOVF UNI, W
   ADDLW 0X30
   CALL DATOS      ;Comando para mostrar el valor de UNI en el LCD
   MOVLW a' '
   CALL DATOS      ;Comando para mostrar el " " en el LCD
   MOVLW a'V'
   CALL DATOS      ;Comando para mostrar el V en el LCD
   CALL RET100MS
   CALL RET100MS
   CALL RET100MS           ;Llamado a subrutinas para retardos
   MOVLW H'01'
   CALL COMANDO            ;Comando para limpiar pantalla del LCD
   CALL RET100MS
   GOTO SWITCH_3           ;Vuelve a SWITCH_3 para volver a verificar su valor



PILA
   CALL CONVIERTE          ;Configuración para el LCD
   MOVWF NUM_PILA          ;NUM_PILA = W
   MOVLW 0X40              ;W = 40
   CALL COMANDO            ;Activa la CGRAM

   MOVLW V48               ;W = V48
   SUBWF NUM_PILA, W   ;NUM_PILA - W
   BTFSS STATUS,C          ;Si C = 1 salta
   GOTO COMP_PILA4         ;C = 0
   GOTO PILA_LLENA         ;C = 1

COMP_PILA4
   MOVLW V4                ;W = V4
   SUBWF NUM_PILA,W        ;NUM_PILA - W
   BTFSS STATUS,C          ;Si C = 1 salta
   GOTO COMP_PILA3         ;C = 0
   GOTO PILA_V45           ;C = 1

COMP_PILA3
   MOVLW V3                ;W = V3
   SUBWF NUM_PILA,W    ;NUM_PILA - W
   BTFSS STATUS,C          ;Si C = 1 salta
   GOTO COMP_PILA2         ;C = 0
   GOTO PILA_V34           ;C = 1

COMP_PILA2
   MOVLW V2                ;W = V2
   SUBWF NUM_PILA,W    ;NUM_PILA - W
   BTFSS STATUS,C          ;Si C = 1 salta
   GOTO COMP_PILA1         ;C = 0
```

```
    GOTO PILA_V23               ;C = 1

COMP_PILA1
    MOVLW V1                    ;W = V1
    SUBWF NUM_PILA,W    ;NUM_PILA - W
    BTFSS STATUS,C              ;Si C = 1 salta
    GOTO COMP_PILA0             ;C = 0
    GOTO PILA_V12               ;C = 1

COMP_PILA0
    MOVLW 0x00                  ;W = 0
    SUBWF NUM_PILA,W    ;NUM_PILA - W
    BTFSS STATUS,Z              ;Si Z = 1 salta
    GOTO PILA_V01               ;C = 0
    GOTO PILA_VACIA             ;C = 1

;Caso pila vacia, carga caracter de pila vacía
PILA_VACIA
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00010001'
    CALL DATOS
    MOVLW B'00010001'
    CALL DATOS
    MOVLW B'00010001'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    GOTO FINAL_PILA

;Caso pila  0-1V, carga carácter de pila
PILA_V01
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00010001'
    CALL DATOS
    MOVLW B'00010001'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    GOTO FINAL_PILA

;Caso pila 1-2V, carga carácter de pila
PILA_V12
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00010001'
```

```
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    GOTO FINAL_PILA

;Caso pila 2-3V, carga carácter de pila
PILA_V23
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    GOTO FINAL_PILA

;Caso pila 3-4V, carga carácter de pila
PILA_V34
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    GOTO FINAL_PILA

;Caso 4-4.8V, carga carácter de pila
PILA_V45
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001010'
    CALL DATOS
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
```

```
    CALL DATOS
    GOTO FINAL_PILA

;Caso pila llena, carga carácter de pila
PILA_LLENA
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00001110'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS
    MOVLW B'00011111'
    CALL DATOS


FINAL_PILA
    CALL RET100MS
    MOVLW 0X80
    CALL COMANDO            ;Se posiciona en la primera posición del LCD
    CALL RET100MS
    MOVLW 0X00
    CALL DATOS              ;Comando para mostrar el carter en el LCD
    CALL RET100MS
    CALL RET100MS           ;Llamado a subrutinas para retardos
    MOVLW H'01'
    CALL COMANDO            ;Comando para limpiar pantalla del LCD
    CALL RET100MS
    GOTO SWITCH_3           ;Vuelve a SWITCH_3 para volver a verificar su valor


;Comfiguracion del LCD
INICIA_LCD
    MOVLW 0X30
    CALL COMANDO
    CALL RET100MS
    MOVLW 0X30
    CALL COMANDO
    CALL RET100MS
    MOVLW 0X38
    CALL COMANDO
    MOVLW 0X0C
    CALL COMANDO
    MOVLW 0X01
    CALL COMANDO
    MOVLW 0X06
    CALL COMANDO
    MOVLW 0X02
    CALL COMANDO
    RETURN

COMANDO
    MOVWF PORTB             ;PORTB = W
    CALL RET200             ;LLAMADO A UN RETARDO
    BCF PORTD, 0    ;BIT 0 DEL PORTA = 0  --> RS
    BSF PORTD, 1    ;BIT 1 DEL PORTA = 1  --> D
    CALL RET200             ;LLAMADO A UN RETARDO
    BCF PORTD, 1    ;BIT 1 DEL PORTA = 0  --> D
    RETURN
```

```
DATOS
   MOVWF PORTB                    ;PORTB = W
   CALL RET200                ;LLAMADO A RETARDO
   BSF PORTD, 0    ;BIT 0 DEL PORTA = 1  --> RS
   BSF PORTD, 1    ;BIT 1 DEL PORTA = 1  --> D
   CALL RET200                ;LLAMADO A RETARDO
   BCF PORTD, 1    ;BIT 1 DEL PORTA = 0  --> D
   CALL RET200                ;LLAMADO A RETARDO
   CALL RET200                ;LLAMADO A RETARDO
   RETURN

CONVIERTE
   BSF ADCON0,2                   ;prende bit 2 de ADCON0 -> GO: inicia la comparacion
   CALL RETARDO                   ;llama subrutina de retardo

ESPERA
   BTFSC ADCON0,2          ;checa bit 2 de ADCON0
   GOTO ESPERA                    ;Si es 1, regresa a la etiqueta ESPERA
   MOVF ADRESH,W           ;Si es 0, W = ADRESH
   RETURN

;Actualiza el valor de centenas en caso de que decenas llegue a 0A o mayor
INC_CENT
   MOVLW 0X0A              ;W=0A
   SUBWF DECE,W;W = DECE-0A
   BTFSS STATUS, C         ;salta si C = 1
   RETURN                  ;return si C=0
   INCF CENT               ;CENT = CENT + 1 (C=1)
   MOVWF DECE              ;DECE = W
   RETURN

;Actualiza el valor de centenas en caso de que decenas llegue a 0A o mayor
INC_DECE
   MOVLW 0X0A              ;W=0A
   SUBWF UNI,W             ;W = UNI-0A
   BTFSS STATUS, C         ;salta si C = 1
   RETURN                  ;return si C=0
   INCF DECE               ;DECE = DECE + 1 (C=1)
   MOVWF UNI               ;UNI = W
   CALL INC_CENT           ;llama INC_CENT
   RETURN


RETARDO
   MOVLW 0X30
   MOVWF VALOR
LOOP
   DECFSZ VALOR
   GOTO LOOP
   RETURN

RET200
   MOVLW 0X02
   MOVWF VALOR1
LOOP1
   MOVLW d'164'
   MOVWF VALOR
LOOP2
   DECFSZ VALOR, 1
   GOTO LOOP2
   DECFSZ VALOR1, 1
   GOTO LOOP1
   RETURN

RET100MS
```

```
   MOVLW 0X03
   MOVWF VALOR
TRES
   MOVLW 0XFF
   MOVWF VALOR1
DOS
   MOVLW 0XFF
   MOVWF VALOR2
UNO
   DECFSZ VALOR2
   GOTO UNO
   DECFSZ VALOR1
   GOTO DOS
   DECFSZ VALOR
   GOTO TRES
   RETURN


END        ;Fin del programa
```

Conclusiones y/o comentarios

Barreiro Valdez Alejandro:

Un Convertidor Analógico Digital es un dispositivo que convierte señales analógicas en señales digitales y es esencial en una amplia variedad de aplicaciones, incluyendo el procesamiento de señales del día a día para su procesamiento en una computadora. Para este proyecto se pudo observar cómo el microcontrolador PIC cuenta con uno de estos dispositivos y lo útil que puede ser para procesar datos de entrada y operar computacionalmente sobre ellos. Se entendieron los diferentes registros que se utilizan para poder habilitar el convertidor y hacer uso de este. Además, se logró realizar el alambrado y la simulación del sistema deseado para cumplir cada uno de los problemas planteados utilizando nuevos conceptos y conceptos utilizados en proyectos anteriores.

Gil Márquez Arath Emiliano:

Para la creación del convertidor A/D como un voltímetro Digital fue necesario emplear nuestros conocimientos de electrónica, programación en lenguaje ensamblador y del funcionamiento del microcontrolador, cosa que hemos estado aprendiendo a lo largo del curso y que nos permitió realizar el proyecto de forma rápida y sencilla. Nuestro programa fue capaz de leer una señal analógica y convertirlo a una representación digital en este caso números, que con el uso de los puertos del microcontrolador y un switch la información generada podía mostrarse de diferentes maneras en la pantalla de un LCD.

Herrera Carrillo Cristhian:

Los convertidores analógico-digitales (ADCs) son dispositivos esenciales en el procesamiento de señales analógicas en sistemas digitales. Estos dispositivos permiten la conversión de señales analógicas a una forma digital que puede ser procesada por un ordenador o cualquier otro dispositivo digital. La precisión del ADC es crucial en muchas aplicaciones, y la elección del tipo y la calidad del ADC dependerá de la aplicación específica en la que se utilice.

Hoy en día, los ADCs se utilizan en una amplia variedad de aplicaciones, desde la medición de señales analógicas en la industria hasta la captura de señales de audio y vídeo en dispositivos electrónicos de consumo. También son esenciales en la comunicación digital, donde se utilizan para convertir señales de voz y datos analógicos en señales digitales que pueden ser transmitidas a través de redes de comunicación digitales.

Zepeda Baeza Jessica:

Los ADCs son vitales para la adquisición de datos y el procesamiento en tiempo real en campos como la automatización industrial, la medicina, las comunicaciones, la investigación científica y la electrónica de consumo.

La selección del ADC adecuado dependerá de los requisitos específicos de la aplicación, como la resolución, la precisión, la velocidad de conversión y la relación señal/ruido. Además, la calidad de la señal de entrada y la interferencia electromagnética también pueden afectar la precisión del ADC.