

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

LABORATORIO DE MICROCOMPUTADORAS

SEMESTRE 2023-2

PREVIO PRACTICA 6

CONVERTIDOR ANALÓGICO/DIGITAL

GRUPO 11

INTEGRANTES:

- CRUZ CEDILLO DANIEL ALEJANDRO
- TÉLLEZ GALLARDO CAROLINA

PROFESOR:

ING. ROMAN V. OSORIO COMPARAN

FECHA DE ENTREGA: 21 DE ABRIL 2023

CALIFICACIÓN

1.- Empleando el canal de su elección del convertido A/D, realizar un programa en el cuál, de acuerdo a una entrada analógica que se ingrese por este canal, se represente el resultado de la conversión en un puerto paralelo utilizar el arreglo de leds para ver la salida, como se muestra en la figura 6.1.

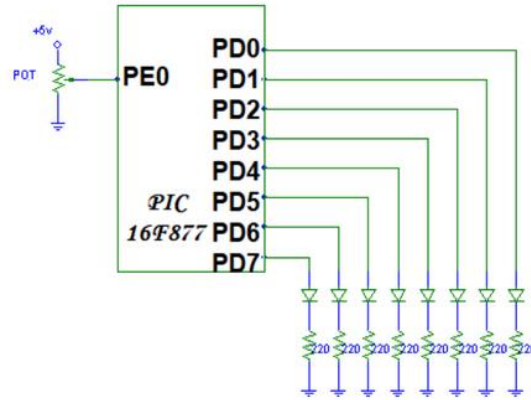


Figura 6.1 Circuito con lectura de una señal analógica

```

C:\Users\52556\Downloads\P6E1.asm
processor 16f877
include <pl6f877.inc>

J equ h'20'
K equ h'21'

org 0
goto inicio
org 5

inicio:
bsf STATUS, RP0
bcf STATUS, RP1
movlw 00H
movwf ADCON1
movlw 00H
movwf TRISE
bcf STATUS, RP0
movlw b'11111001'

movwf ADCON0
clrf PORTD

CONVERTIDOR:
bsf ADCON0, 2
call RETARDO
bcf ADCON0, 2
movfw ANSWER

```

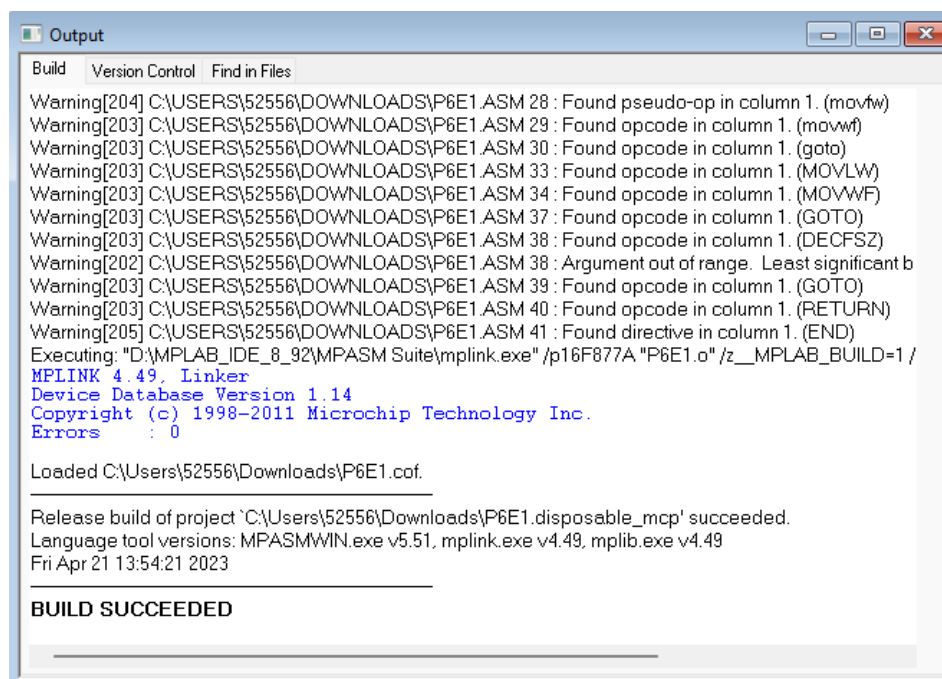
```
goto CONVERTIDOR

RETARDO:
MOVLW D'25'
MOVWF J
JLOOP: MOVWF K
KLOOP: DECFSZ K, F
GOTO KLOOP
DECFSZ J, P
GOTO JLOOP
RETURN
END
```

En las imágenes mostradas en la parte superior podemos apreciar el código que se utilizó para solucionar el ejercicio 1 de la practica el cual nos solicitaba una respuesta en función del comportamiento de los leds, para ello en primer lugar definimos las entradas del código las cuales tienen como nombre J y Q una vez realizada esta acción procedemos a definir dos funciones: Función inicio y función convertidor la primera la definimos de esa forma ya que definimos el estatus de las dos variables declaradas y hacemos los cambios y registros necesarios para que el código funcione de forma que no haya 'basura' en los puertos.

Para la función convertidor la cual es la función más importante de todo el código junto con la función retardo, en la primera realizamos el cambio de analógico a digital y a su vez llama a las funciones jloop y kloop las cuales transfieren el flujo del trabajo o del proceso hasta que la función las detenga.

COMPILACIÓN



```
Output
Build Version Control Find in Files
Warning[204] C:\USERS\52556\DOWNLOADS\P6E1.ASM 28 : Found pseudo-op in column 1. (movfw)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E1.ASM 29 : Found opcode in column 1. (movwf)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E1.ASM 30 : Found opcode in column 1. (goto)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E1.ASM 33 : Found opcode in column 1. (MOVLW)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E1.ASM 34 : Found opcode in column 1. (MOVWF)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E1.ASM 37 : Found opcode in column 1. (GOTO)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E1.ASM 38 : Found opcode in column 1. (DECFSZ)
Warning[202] C:\USERS\52556\DOWNLOADS\P6E1.ASM 38 : Argument out of range. Least significant b
Warning[203] C:\USERS\52556\DOWNLOADS\P6E1.ASM 39 : Found opcode in column 1. (GOTO)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E1.ASM 40 : Found opcode in column 1. (RETURN)
Warning[205] C:\USERS\52556\DOWNLOADS\P6E1.ASM 41 : Found directive in column 1. (END)
Executing: "D:\MPLAB_IDE_8_92\MPASM Suite\mplink.exe" /p16F877A "P6E1.o" /z__MPLAB_BUILD=1 /
MPLINK 4.49. Linker
Device Database Version 1.14
Copyright (c) 1998-2011 Microchip Technology Inc.
Errors : 0

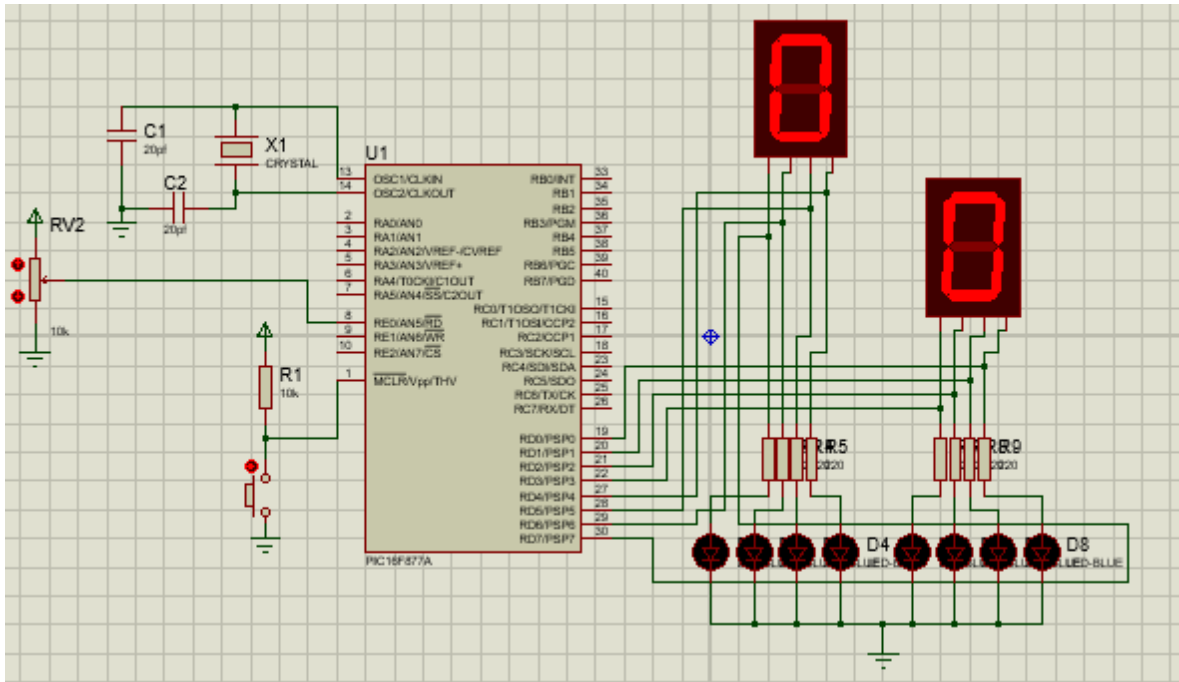
Loaded C:\Users\52556\Downloads\P6E1.cof.

Release build of project 'C:\Users\52556\Downloads\P6E1.disposable_mcp' succeeded.
Language tool versions: MPASMWIN.exe v5.51, mplink.exe v4.49, mplib.exe v4.49
Fri Apr 21 13:54:21 2023

BUILD SUCCEEDED
```

Como podemos apreciar en la imagen superior el código mostrado en mlab es correcto ya que la compilación se llevó a cabo sin complicaciones y resulto exitosa. A continuación mostraremos el circuito construido en proteus el cual se simuló, obteniendo a través de esta el comportamiento del sistema.

SIMULACIÓN EN PROTEUS



Para la simulación en PROTEUS, en primer lugar colocamos el microcontrolador, para este ejercicio vamos a utilizar dos displays de siete segmentos los cuales nos van a ayudar a realizar la conversión de las señales, es por eso que colocamos de igual manera las resistencias, botones, tierras y capacitores necesarios para el funcionamiento del circuito, el cual funcionara de la siguiente manera:

- Los leds ubicados en la parte inferior del circuito se prenderán (todos) y posteriormente se apagaran llevando una secuencia especificada por el codigo.

2.- Utilizando el circuito anterior, realizar un programa que indique el rango en el cuál se encuentra el voltaje a la entrada del convertidor canal seleccionado. Mostrar el valor en un display de 7 segmentos.

```
C:\Users\52556\Downloads\P6E2.asm

processor 16f877
include <pl6f877.inc>

J equ h'20'
K equ h'21'

org 0
goto inicio
org 5

inicio:
bsf STATUS, RP0
bcf STATUS, RP1
movlw 00H
movwf ADCON1
movlw 00H
movwf TRISD
bcf STATUS, RP0
movlw b'11111001'

movwf ADCON0
clrf PORTD

CONVERTIDOR:
bsf ADCON0, 2
call RETARDO
bcf ADCON0, 2
movfw ADRESH
```

En esta primer parte del código se definieron las variables J y K y la dirección de memoria que tendrán en el microcontrolador, se define la función INICIO en la cual se asignan los estatus de las mismas y también se hace la limpia de los puertos para evitar la basura que pueda tener.

```
C:\Users\52556\Downloads\P6E2.asm

clrf PORTD

CONVERTIDOR:
bsf ADCON0, 2
call RETARDO
bcf ADCON0, 2
movfw ADRESH
sublw d'85'
btfsc STATUS, C
goto rango1
movfw ADRESH
sublw d'170'
btfsc STATUS, C
goto rango2
movlw h'07'
movwf PORTD
goto CONVERTIDOR

rango1:
movlw h'01'
movwf PORTD
goto CONVERTIDOR

rango2:
movlw h'03'
movwf PORTD
goto CONVERTIDOR

rango3:
movlw h'07'
```

En esta parte del código se define la codificación de la función CONVERTIDOR en la cual se realiza la clasificación de los rangos que se definirán, para ello utilizamos un nuevo comando sublw para que se puedan manipular los registros del microcontrolador y el estatus que se tendrán.

```

C:\Users\52556\Downloads\P6E2.asm

rango1:
    movlw h'01'
    movwf PORTD
    goto CONVERTIDOR
rango2:
    movlw h'03'
    movwf PORTD
    goto CONVERTIDOR
rango3:
    movlw h'07'
    movwf PORTD
    goto CONVERTIDOR

RETARDO:
    MOVLW D'25'
    MOVWF J
JLOOP: MOVWF K
KLOOP: DECFSZ K, F
    GOTO KLOOP
    DECFSZ J, P
    GOTO JLOOP
    RETURN
    END

```

En este código resolvimos la problemática planteada en el ejercicio dos ya que se nos solicitaba que se indicara el rango del voltaje a la entrada del convertidor es por eso que definimos tres rangos en el código y a su vez conservamos las funciones Retardo y convertidor del ejercicio 1. En este código la respuesta estará definida por los rangos que se le asignen y así al momento de simularlo con la herramienta PROTEUS se verá el funcionamiento que se obtiene.

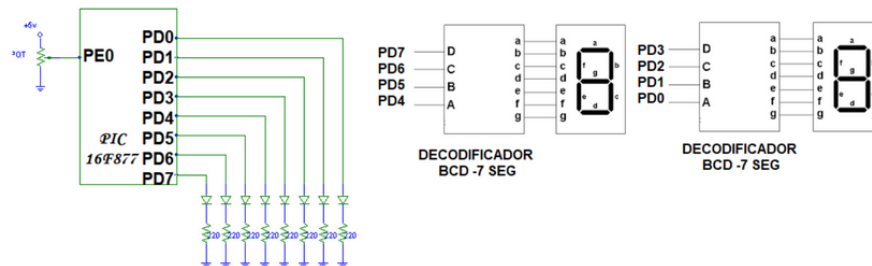
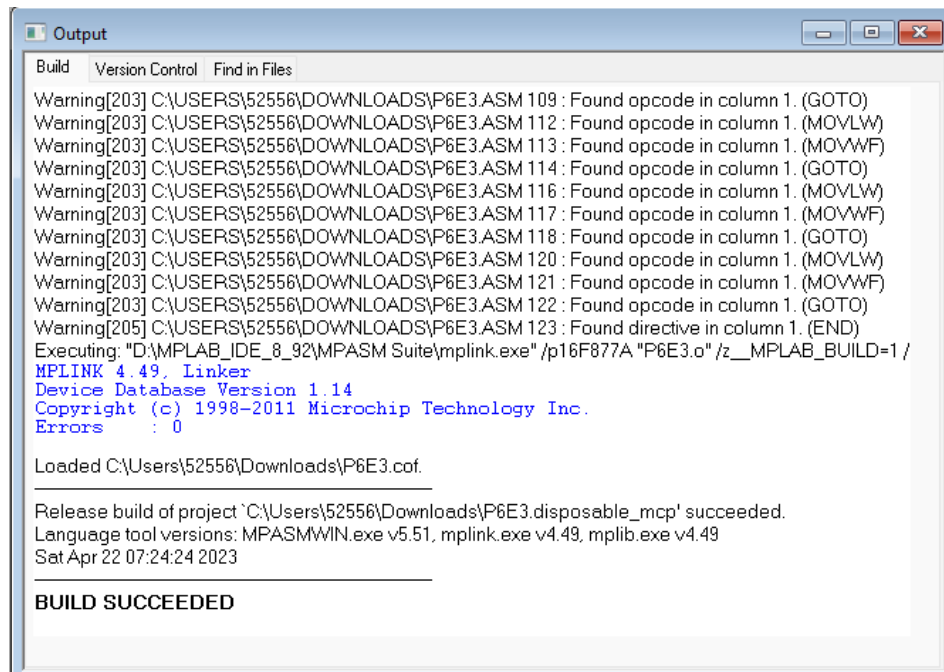


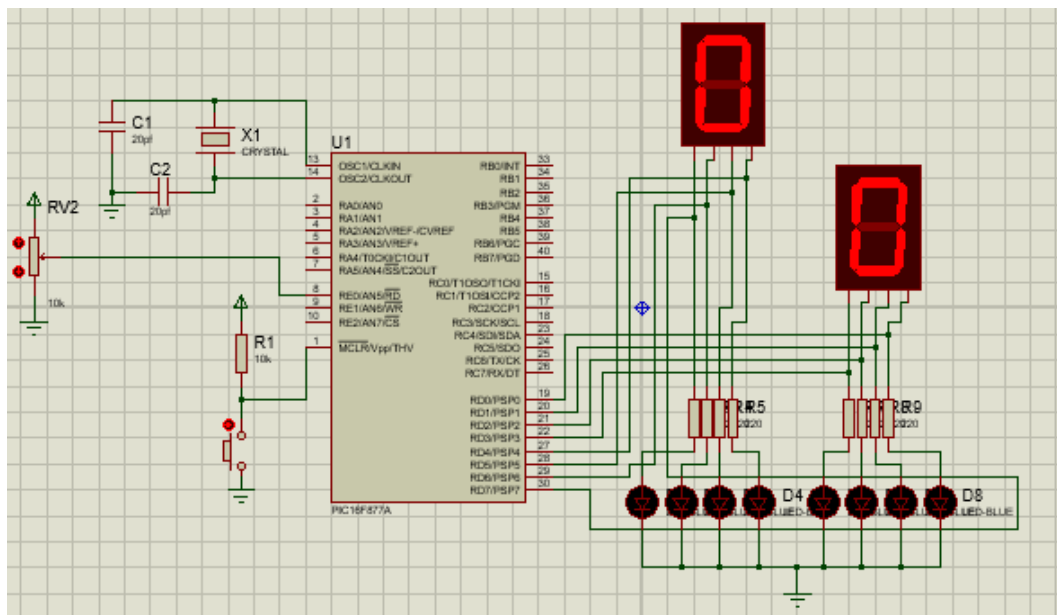
Figura 6.2 Circuito actividad 2

Entrada Analógica V _e	Salida
0 – 0.99 V	0
1.0 – 1.99 V	1
2.0 – 2.99 V	2
3.0 – 3.99 V	3
4.00 – 4.80 V	4
4.80 – 5.00 V	5



En la imagen anterior se puede apreciar la compilación del código propuesto para la solución esta compilación resulto exitosa y en las siguientes imágenes podremos visualizar la simulación con el respectivo circuito en PROTEUS.

SIMULACIÓN EN PROTEUS



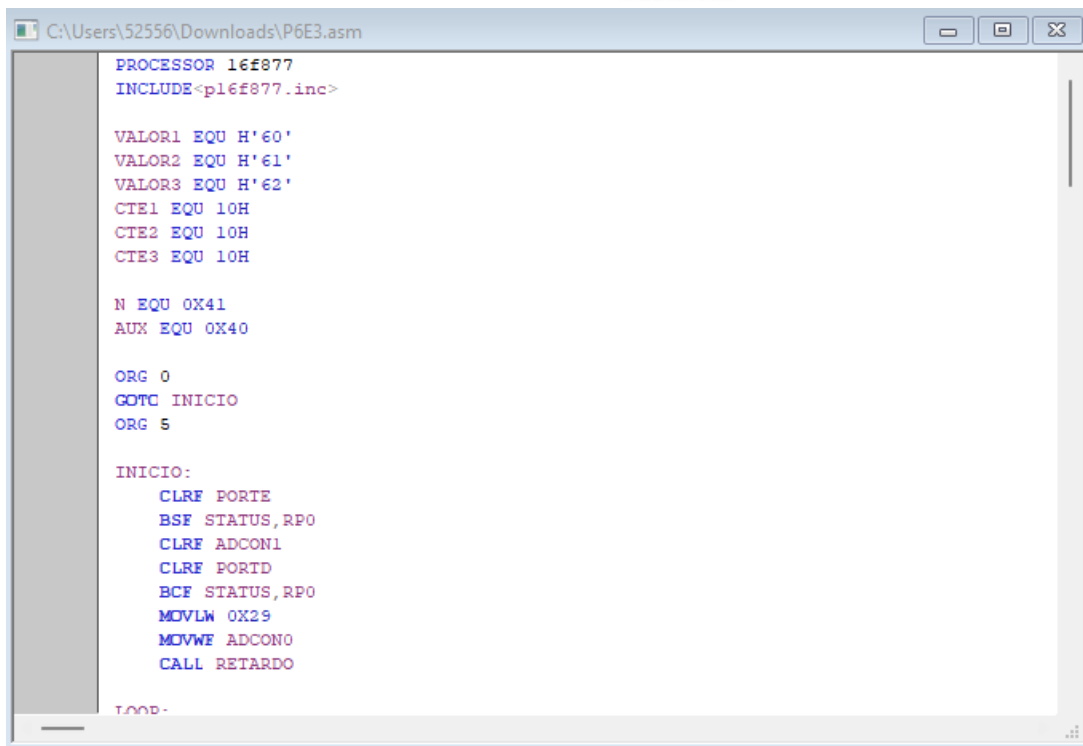
Para la simulación del ejercicio 2 utilizamos el mismo circuito del ejercicio 1 ya que cuenta con las herramientas necesarias para la simulación y no se necesitan hacer modificaciones algunas.

En este ejercicio el comportamiento del circuito se caracterizara del anterior ya que aquí si se utilizaran los displays de siete segmentos para marcar el voltaje que estamos midiendo, es por eso que en este circuito se agregan las resistencias para cada uno y se conectan al costado del microcontrolador.

3.- Realizar un programa, de manera que identifique cuál de tres señales analógicas que ingresan al convertidor A/D es mayor que las otras dos; representar el resultado de acuerdo al contenido de la tabla 6.2.

Señal	PD2	PD1	PD0
Ve1>Ve2 y Ve3	0	0	1
Ve2>Ve1 y Ve3	0	1	1
Ve3>Ve1 y Ve2	1	1	1

Tabla 6.2



```

C:\Users\52556\Downloads\P6E3.asm
PROCESSOR 16f877
INCLUDE<p16f877.inc>

VALOR1 EQU H'60'
VALOR2 EQU H'61'
VALOR3 EQU H'62'
CTE1 EQU 10H
CTE2 EQU 10H
CTE3 EQU 10H

N EQU 0X41
AUX EQU 0X40

ORG 0
GOTO INICIO
ORG 5

INICIO:
    CLRF PORTE
    BSF STATUS,RP0
    CLRF ADCON1
    CLRF PORTD
    BCF STATUS,RP0
    MOVLW 0X29
    MOVWF ADCON0
    CALL RETARDO

LOOP:

```

Para la primera parte del código para el ejercicio tres de esta práctica definimos más variables ya que ahora debemos identificar tres señales analógicas y con estas hacer una comparación para definir cuál es la que tiene mayor magnitud y conforma la tabla mostrada debemos analizar el resultado.

Es por eso que en esta propuesta de solución se definieron tres valores VALOR1, VALOR2 y VALOR3 además también se definen tres constantes CTE1, CTE2, CTE3 y las variables extra N y AUX las cuales nos ayudaran a realizar la comparación.

Como en los anteriores códigos se define una función INICIO que se encarga de la 'preparación' de los puertos para que el código tenga las condiciones iniciales necesarias, en este caso con la función CLRf podemos ver que se limpiaron los tres puertos correspondientes a las señales que se analizarán mas adelante.

```

C:\Users\52556\Downloads\P6E3.asm

LOOP:
    BSF ADCON0,GO
ESPERA:
    BSF ADCON0,GO
    GOTC ESPERA
    MOVF ADRESH,W
    MOVWF 20H
    MOVWF 30H

    MOVLW 0X31
    MOVWF RETARDO
    BSF ADCON0,GO

ESPERA2:
    BTFSC ADCON0,GO
    GOTC ESPERA2
    MOVF ADRESH,W
    MOVWF 21H
    MOVWF 31H

    MOVLW 0X39
    MOVWF ADCON0
    CALL RETARDO
    BSF ADCON0,GO
ESPERA3:
    BTFSC ADCON0,GO
    GOTC ESPERA3
    MOVF ADRESH,W

```

```

C:\Users\52556\Downloads\P6E3.asm

    MOVF ADRESH,W
    MOVWF 22H
    MOVWF 32H
;ORDENAR LAS ENTRADAS+
;LOCALIZACIÓN DEL NUMERO MAYOR ->0X2F
INICIO_A:
    CLRf N
ALGORITMO:
    MOVLW 0X20
    MOVWF FSR
LOOPA:
    MOVF INDF,W
    MOVWF AUX
    INCF FSR,1
    BTFSC FSR,4
    GOTC ITERACION

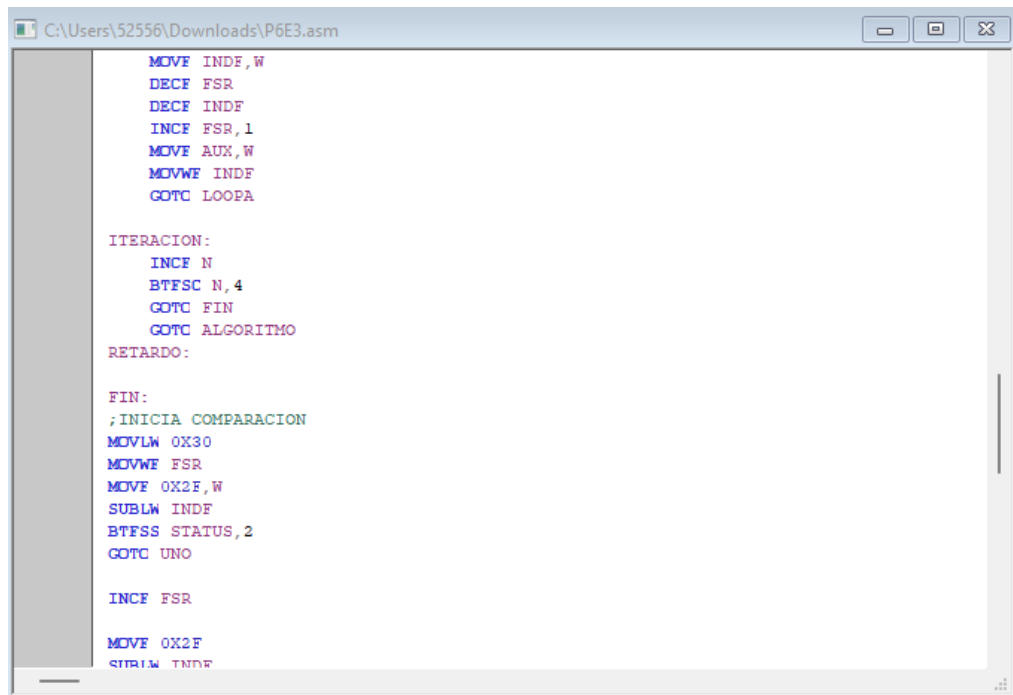
    MOVF INDF,W
    SUBWF AUX,W
    BTFSC STATUS,C
    GOTC SWAP
    GOTC LOOPA

SWAP:
    MOVF INDF,W
    DECF FSR
    DECF INDF
    INCF FSR,1

```

En estos bloques de código definiremos lo que son las funciones LOOP, ESPERA1, ESPERA2, ESPERA3 y también la función más importante del código ALGORITMO, en la

cual se lleva a cabo la comparación entre señales, es por eso que dentro de esta función definimos un SWAP y una ITERACIÓN funciones importantes para llevar a cabo el acomodo de las señales.



```
C:\Users\52556\Downloads\P6E3.asm

    MOVF INDF,W
    DECF FSR
    DECF INDF
    INCF FSR,1
    MOVF AUX,W
    MOVWF INDF
    GOTO LOOPA

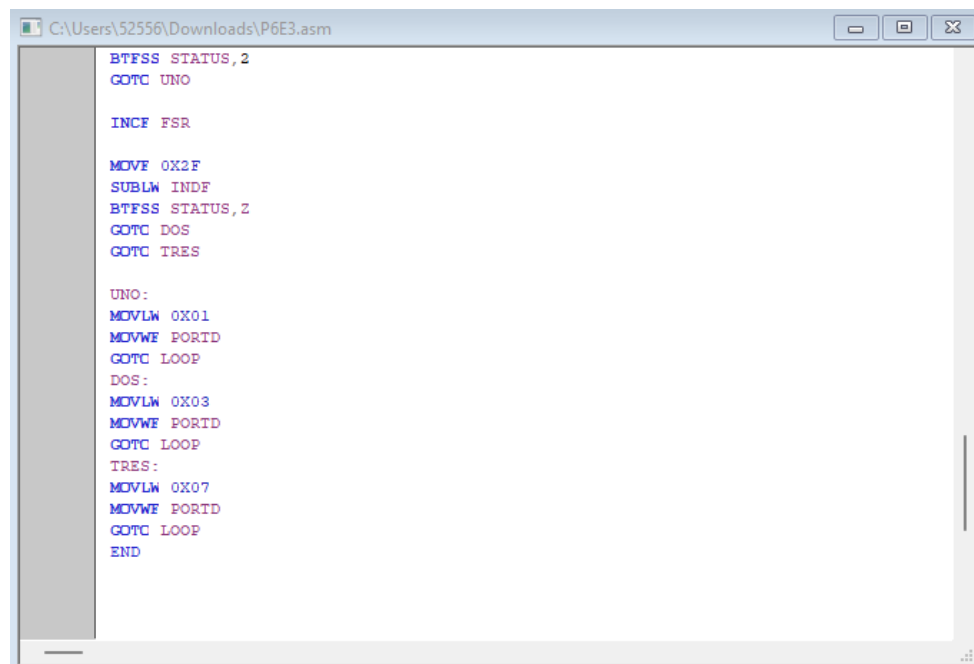
ITERACION:
    INCF N
    BTFSN N,4
    GOTO FIN
    GOTO ALGORITMO

RETARDO:

FIN:
; INICIA COMPARACION
MOVLW 0X30
MOVWF FSR
MOVF 0X2F,W
SUBLW INDF
BTFSN STATUS,2
GOTO UNO

    INCF FSR

    MOVF 0X2F
    SUBLW INDF
```



```
C:\Users\52556\Downloads\P6E3.asm

    BTFSN STATUS,2
    GOTO UNO

    INCF FSR

    MOVF 0X2F
    SUBLW INDF
    BTFSN STATUS,2
    GOTO DOS
    GOTO TRES

UNO:
    MOVLW 0X01
    MOVWF PORTD
    GOTO LOOP
DOS:
    MOVLW 0X03
    MOVWF PORTD
    GOTO LOOP
TRES:
    MOVLW 0X07
    MOVWF PORTD
    GOTO LOOP
END
```

COMPILACIÓN

Como podemos observar la compilación de este código ha sido exitosa por lo cual procederemos a llevarlo a simulación con los respectivos circuitos para ver si el comportamiento es correcto.

```

Output
Build Version Control Find in Files

Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 109 : Found opcode in column 1. (GOTO)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 112 : Found opcode in column 1. (MOVLW)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 113 : Found opcode in column 1. (MOVWF)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 114 : Found opcode in column 1. (GOTO)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 116 : Found opcode in column 1. (MOVLW)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 117 : Found opcode in column 1. (MOVWF)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 118 : Found opcode in column 1. (GOTO)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 120 : Found opcode in column 1. (MOVLW)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 121 : Found opcode in column 1. (MOVWF)
Warning[203] C:\USERS\52556\DOWNLOADS\P6E3.ASM 122 : Found opcode in column 1. (GOTO)
Warning[205] C:\USERS\52556\DOWNLOADS\P6E3.ASM 123 : Found directive in column 1. (END)
Executing: "D:\MPLAB_IDE_8_92\MPASM Suite\mplink.exe" /p16F877A "P6E3.o" /z__MPLAB_BUILD=1 /
MPLINK 4.49, Linker
Device Database Version 1.14
Copyright (c) 1998-2011 Microchip Technology Inc.
Errors : 0

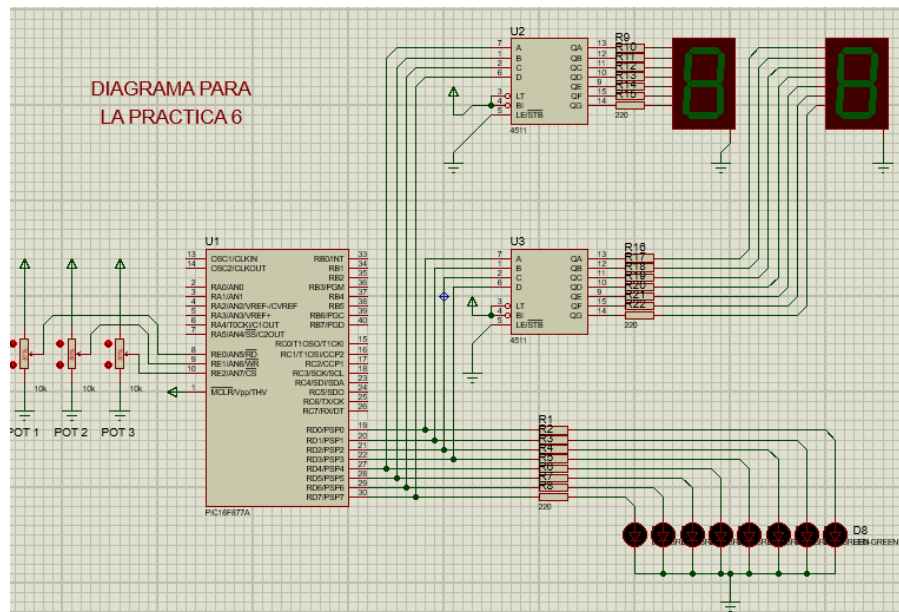
Loaded C:\Users\52556\Downloads\P6E3.cof.

Release build of project 'C:\Users\52556\Downloads\P6E3.disposable_mcp' succeeded.
Language tool versions: MPASMWIN.exe v5.51, mplink.exe v4.49, mplib.exe v4.49
Sat Apr 22 07:24:24 2023

BUILD SUCCEEDED

```

SIMULACIÓN EN PROTEUS



Para el ejercicio tres tuvimos que modificar el circuito propuesto anteriormente ya que como equipo comprobamos que en los circuitos anteriores no funcionaban de forma totalmente

correcta, por lo cual decidimos agregar dos decodificadores para los dos displays de siete segmentos entonces realizamos las modificaciones correspondientes pero al momento de cargar el código pudimos notar que el comportamiento del circuito no era el correcto entonces deducimos que el problema de todo estaba en la codificación pero ya no logramos obtener el resultado esperado aunque realizamos varias modificaciones.