INSTITUTO POLITECNICO NACIONAL
SECRETARIA ACADEMICA
DIRECCION DE EDUCACION MEDIA SUPERIOR

CENTRO DE ESTUDIOS CIENTIFICOS Y TECNOLOGICOS num. 9
"JUAN DE DIOS BATIZ"

Unidad de Aprendizaje: **MICROELECTRONICA PROGRAMABLE.**	Turno: MATUTINO
Nivel: 6°	Periodo: Tercera evaluación.
Especialidad o área: Sistemas Digitales.	Ciclo escolar: 2020-2021 "B".
Fecha del examen: 14 de Junio de 2021.	Contenido a evaluar: Unidad III y IV.
Horario del examen: 11:00 a 12:50 hrs.	Duración del examen: Dos horas

Tipo de examen: **Único.**

Calificación:

Alumno__**Morales Martínez José Antonio** _____

Boleta__2019090265_____Firma___**Morales Martínez José Antonio** _____

Profesor de esta asignatura en este grupo_ Jesús Alberto Olivares Vargas _____Grupo____6IM2_____

I.- Sección Primera. **PARTE A.**

INSTRUCCIONES
CONTESTE EN EL PARENTESIS LA LETRA DE LA RESPUESTA CORRECTA.

PARTE A:

1.- EL CIRCUITO 74LS595 ES UN:……..……..……..…….(C)
A) CONTADOR DE RIZADO. B) MILTIPLEXOR DE DATOS.
C) REGISTRO DE CARGA EN SERIE Y SALIDA PARALELO D) REGISTRO DE CORRIMIENTO ALEATORIO.

2.- EL CIRCUITO MAX232 ES UN CIRCUITO QUE CONTIENE:……..……..….(B)
A) 14 TERMINALES. B) 16 TERMINALES.
C) 28 TERMINALES. D) 40 TERMINALES.

3.- LOS ELEMENTOS BASICOS EN UNA COMUNICACIÓN SERIAL SON:……...……..…….(A)
A) UN Tx. MEDIO FISICO Y UN Rx. B) UN REGISTRO DE CORRIMIENTO Y UN Rx.
C) UN AMPLIFICADOR OPERACIONAL Y UN OSC. D) UN TRANSISTOR Y UN DCE.

4.- EL CIRCUITO MAX232 ES UN CIRCUITO QUE CONTIENE:……..…….....…..……..……..…(C)
A) TRES TX Y DOS RX. B) CUATRO TX Y DOS RX.
C) DOS TX Y DOS RX. D) CINCO TX Y DOS RX.

5.- QUE EFECTO CREA LA CAPACITANCIA EN UNA LINEA DE TRANSMISION DE PAR DE COBRE:….....(A)
A) LA ATENUACION DEL PULSO B) LA AMPLIFICACION DEL PULSO.
C) LA INVERSION DE POLARIDAD DEL PULSO. D) LA DEFORMACION DEL PULSO.

6.-EL CONECTOR UTILIZADO EN LA COMUNICACIÓN SERIAL RS232 ES:…….....……..…(B)
A) UN RJ 45.. B) UN DB 9.
C) CENTRONIX. D) UN USB TIPO A.

7.- EL FORMATO PARA LA TRANSMISION EN SERIE RS232 ES:……..……(B)

A) 2 BIT DE INICIO, 9 BITS DE DATOS, Y 2 DE PARADA.

B) 1 BIT DE INICIO, 8 BITS DE DATOS, Y 1 DE PARADA.

C) 3 BIT DE INICIO, 7 BITS DE DATOS, Y 3 DE PARADA.

D) 5 BIT DE INICIO, 6 BITS DE DATOS, Y 1 DE PARIDAD.

8.- EL BAUDRATE (BITS/SEG) CORRECTO EN LA TX DE DATOS ES:……..…...……(C)

A) 1300 BAUDS.

B) 9100 BAUDS.

C) 2400 BAUDS.

D) 4300 BAUDS.

9.- EL BIT DE PARIDAD EN EL ESTÁNDAR RS232-C ES UTILIZADO PARA:.....……...……(C)

A) SINCRONIZAR LOS DATOS.

B) LA INVITACION A DESCONEXION.

C) LA DETECCION DE ERRORES EN LA TX.

D) CONTROLAR UNA TRASMISION DE DATOS.

10.- UN TIPO DE MODUACION EN LAS LINEAS DE TX ES:…….……..…….…….....……(D)

A) UHF.

B) TSK.

C) VHF.

D) ASK.

Valor por reactivo: **0.1 de punto.**
Valor de la sección PARTE A: **1 punto.**

II.- Sección A Segunda

INSTRUCCIONES

EL PROBLEMA DEBERA CONTENER SU DIAGRAMA DE FLUJO Y SU PROGRAMA EN ELENGUAJE ENSAMBLADOR DEL PIC 16F877, NO SE PODRA CONTESTAR EL INCISO SIGUIENTE SI NO SE CONTESTO EL ANTERIOR.

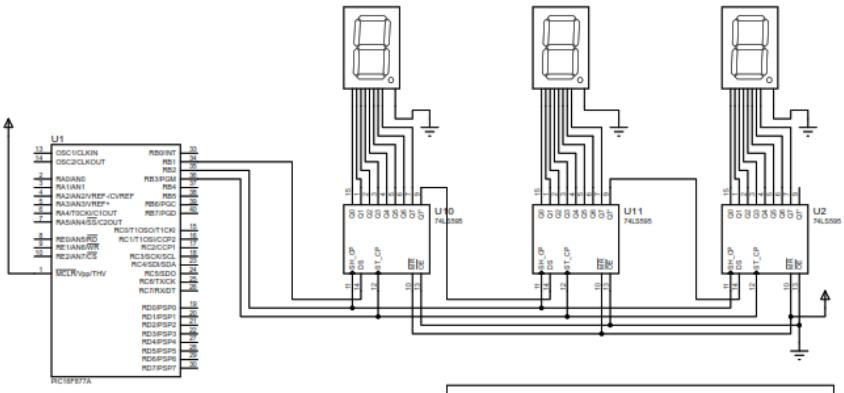
PARTE B:

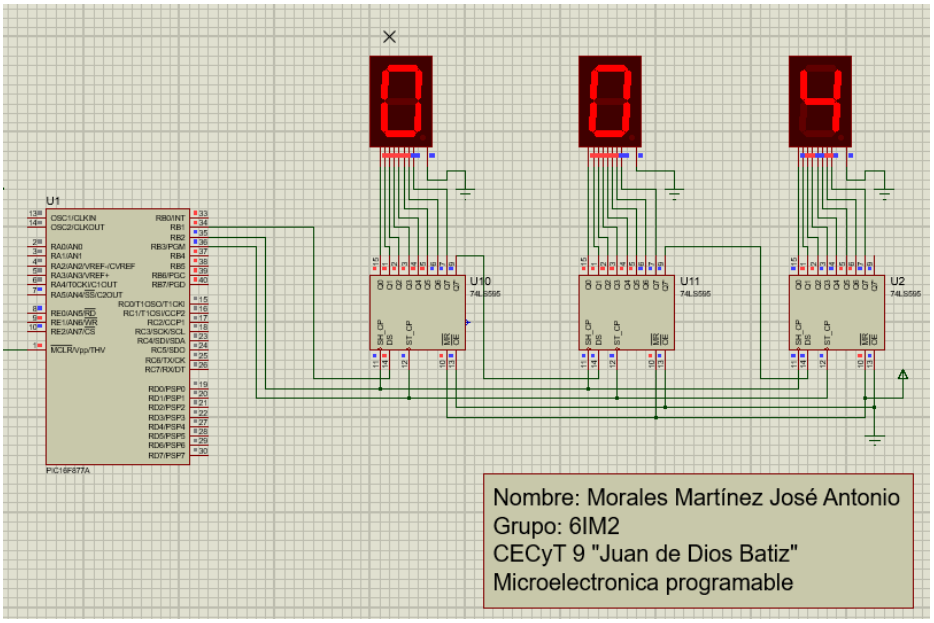
PROBLEMA: DESARROLLE UN SISTEMA DIGITAL QUE CUMPLA CON LAS SIGUIENTES CARACTERISTICAS:

A) UTILIZANDO EL PIC16F877, SE DESEAN CONECTAR TRES DISPLAYS DE 7 SEGMENTOS DE CATODO COMUN, PERO EL PIC SOLO CUENTA CON 3 PINES DISPONIBLES RB1, RB2, RB3. ¿MEDIANTE UN DIAGRAMA ESQUEMATICO CREADO EN PROTEUS PROPONGA EL HARDWARE ADICIONAL Y DIBUJE A DETALLE, LA CONEXIÓN DE ESTOS DISPLAYS AL MICROCONTROLADOR?

NOTA: RECUERDE EL EJERCICIO DEL CIRCUITO 74LS595, DE COMUNICACIÓN SERIAL Y LOS REGISTROS DE CORRIMIENTO.

Valor: 1 PUNTO.





B) REALICE EL PROGRAMA EN MPLAB PARA QUE CUANDO SE ARRANQUE EL SISTEMA, EN EL DISPLAY SE MUESTRE LA CUENTA 000 Y POR CADA SEGUNDO TRANSCURRIDO SE INCREMENTE EN UNO, HASTA LLEGAR A 999, SI TRANSCURRE UN SEGUNDO MAS EL SISTEMA VOLVERA A EMPEZAR LA CUENTA, Y ASI ESTE PERMANECE POR TIEMPO INDEFINIDO, COMENTE TODAS LAS INSTRUCCIONES Y AGREGUELO AL EXAMEN.

Valor del programa: **1 punto.**

Valor de los comentarios: **2 puntos.**

```
;INSTITUTO POLITECNICO NACIONAL
;CECYT 9 JUAN DE DIOS BATIZ
;
;EXAMEN TERCER PARCIAL.
;.
;GRUPO:6IM2
;
;INTEGRANTE
;Morales Martínez José Antonio
;
;Este programa ejecuta un contador de 000 a 999 mostrando los datos en displays de 7 segmentos
;haciendo uso de una comunicación serial con el PIC16F877A

;-----------------------------------------------------------------------------------------------

list p=16F877A; // Directiva utilizada para definir el microcontrolador a utilizar


#include "c:\program files (x86)\microchip\mpasm suite\p16f877a.inc";
```

```
;Bits de configuracion
__config _XT_OSC & _WDT_OFF & _PWRTE_ON & _BODEN_OFF & _LVP_OFF & _CP_OFF; ALL

;-----------------------------------------------------------------------------------------------------
;fosc = 4 Mhz.
;Ciclo de trabajo del PIC = (1/fosc)*4 = 1us.
;t int =(256-R)*(P)*((1/4000000)*4) = 1 ms ;// Tiempo de interrupción.
;R=131, p=8.
;frec int = 1/ t int = 1Khz.
;-----------------------------------------------------------------------------------------------------
;
;Registros de proposito general Banco 0 de Memoria RAM.
;
;Registros propios de estructura del programa

res_w       equ     0x20;   //Registro de resplado de la variable W en la subruitna de interrupción
res_status  equ             0x21;   //Registro de resplado de la variable status en la subruitna de interrupción
res_pclath  equ             0x22;   //Registro de resplado de la variable pclath en la subruitna de interrupción
res_fsr     equ             0x23;   //Registro de resplado de la variable fsr en la subruitna de interrupción
presc_1     equ             0x24;   //T int= T interrupcion(0.001s)*presc_1 multiplica por un escalar al tiempo de interrupcion base
presc_2     equ             0x25;   //T int= T interrupcion(0.001s)*presc_1*presc_2 multiplica por un escalar al tiempo de interrupcion base
banderas    equ             0x26;   //Registro en donde se definen bits banderas (bandera_c, bandera_D, bandera_clear)
cont_milis  equ             0x27;   //Registro que lleva la cuenta de las unidades de milisegundos (0-255)
cont_seg    equ     0x28;   //Registro auxiliar
Reg_TXH     equ             0x29;   //Byte alto del conjunto de bits de transmisión de datos serial 16-2
Reg_TXM     equ             0x2A;   //Byte medio del conjunto de bits de transmision de datos serial 8-16
Reg_TXL     equ             0x2D;   //Byte bajo del conjunto de bits de transmisión de datos serial 1-8
contador    equ             0X2B;   //Registro auxiliar para las iteraciones encargadas de rotar los bytes y enviarlos
temporal    equ     0x2E;   //Registro auxiliar en la conversión de binario a 7 segmentos
cta_uniseg  equ             0x2F;   //Dirección de la memoria RAM para el registro de las unidades de segundo
cta_decseg  equ     0x30;   //Dirección de la memoria RAM para el registro de las decenas de segundo
cta_censeg  equ             0x31;   //Dirección de la memoria RAM para el registro de las unidades de minuto

;-----------------------------------------------------------------------------------------------------
;Constantes
;Constantes de caracteres en siete segmentos.
Car_A           EQU b'01110111'; Caracter A en siete segmentos.
Car_b           EQU b'01111100'; Caracter b en siete segmentos.
Car_C           EQU b'00111001'; Caracter C en siete segmentos.
Car_cc          EQU b'01011000'; Caracter c en siete segmentos.
Car_d           EQU b'01011110'; Caracter D en siete segmentos.
Car_E           EQU b'01111001'; Caracter E en siete segmentos.
Car_F           EQU b'01110001'; Caracter F en siete segmentos.
Car_G           EQU b'00111101'; Caracter G en siete segmentos.
Car_gg          EQU b'01101111'; Caracter g en siete segmentos.
Car_H           EQU b'01110110'; Caracter H en siete segmentos.
Car_hh          EQU b'01110100'; Caracter h en siete segmentos.
Car_I           EQU b'00000110'; Caracter I en siete segmentos.
Car_ii          EQU b'00000100'; Caracter i en siete segmentos.
Car_L           EQU b'00111000'; Caracter L en siete segmentos.
Car_J           EQU b'00011111'; Caracter J en siete segmentos.
Car_N           EQU b'00110111'; Caracter N en siete segmentos.
Car_M           EQU b'00101011'; Caracter M en siete segmentos.
Car_O           EQU b'00111111'; Caracter O en siete segmentos.
Car_oo          EQU b'01011100'; Caracter o en siete segmentos.
```

```
Car_P                   EQU b'01110011'; Caracter P en siete segmentos.
Car_q                   EQU b'01100111'; Caracter q en siete segmentos.
Car_R                   EQU b'01010000'; Caracter R en siete segmentos.
Car_S                   EQU b'01101101'; Caracter S en siete segmentos.
Car_t                   EQU b'01111000'; Caracter t en siete segmentos.
Car_U                   EQU b'00111110'; Caracter U en siete segmentos.
Car_uu                  EQU b'00000110'; Caracter u en siete segmentos.
Car_y                   EQU b'01101110'; Caracter y en siete segmentos.
Car_Z                   EQU b'01011011'; Caracter Z en siete segmentos.
Car_0                   EQU b'00111111'; Caracter 0 en siete segmentos.
Car_1                   EQU b'00000110'; Caracter 1 en siete segmentos.
Car_2                   EQU b'01011011'; Caracter 2 en siete segmentos.
Car_3                   EQU b'01001111'; Caracter 3 en siete segmentos.
Car_4                   EQU b'01100110'; Caracter 4 en siete segmentos.
Car_5                   EQU b'01101101'; Caracter 5 en siete segmentos.
Car_6                   EQU b'01111101'; Caracter 6 en siete segmentos.
Car_7                   EQU b'00000111'; Caracter 7 en siete segmentos.
Car_8                   EQU b'01111111'; Caracter 8 en siete segmentos.
Car_9                   EQU b'01100111'; Caracter 0 en siete segmentos.
Car_                    EQU     b'00001000'; Caracter _ en siete segmentos.
Car_null        EQU b'00000000'; Caracter nulo en siete segementos.


;-----------------------------------------------------------------------------------------------

;banderas del registro banderas.
ban_int         equ     .0;     //Bit bandera de retardo 1s
sin_bd1         equ     .1;     //Sin Uso bd1.
sin_bd2         equ     .2;     //Sin Uso bd2.
sin_bd3         equ     .3;     //Sin Uso bd3.
sin_bd4         equ     .4;     //Sin Uso bd4.
sin_bd5         equ     .5;     //Sin Uso bd5.
sin_bd6         equ     .6;     //Sin Uso bd6.
sin_bd7         equ     .7;     //Sin Uso bd7.
;-----------------------------------------------------------------------------------------------
;Asignacion de los bits de los puertos de I/O.
;Puerto A.
Sin_usoRA0              equ                     .0; // Sin uso RA0.
Sin_usoRA1              equ                     .1; // Sin uso RA1.
Sin_usoRA2              equ                     .2; // Sin uso RA2.
Sin_usoRA3              equ                     .3; // Sin uso RA3.
Sin_usoRA4              equ                     .4; // Sin uso RA4.
Sin_usoRA5              equ                     .5; // Sin uso RA5.

proga                   equ     b'11111'; // Programacion Inicial del Puerto A.

;Puerto B.
Sin_usoRB0              equ             .0; // Sin uso RB0.
Data_in                 equ             .1; // Salida para mandar los datos de manera serial al 74LS595
CLK_Serie               equ             .2; // Salida para generar el CLK serie.
CLK_Paralelo    equ             .3; // Salida para generar el CLK paralelo.
Sin_usoRB4              equ             .4; // Sin uso RB4.
Sin_usoRB5              equ             .5; // Sin uso RB5.
Sin_usoRB6              equ             .6; // Sin uso RB6.
Sin_usoRB7              equ             .7; // Sin uso RB7.

progb                   equ     b'11110001'; // Programacion Inicial del Puerto B.

;Puerto C.
```

```
Sin_usoRC0              equ                     .0; // Sin uso RC0.
Sin_usoRC1              equ                     .1; // Sin uso RC1.
Sin_usoRC2              equ                     .2; // Sin uso RC2.
Sin_usoRC3              equ                     .3; // Sin uso RC3.
Sin_usoRC4              equ                     .4; // Sin uso RC4.
Sin_usoRC5              equ                     .5; // Sin uso RC5.
Sin_usoRC6              equ                     .6; // Sin uso RC6.
Sin_usoRC7              equ                     .7; // Sin uso RC7.

progc                   equ     b'11111111'; // Programacion Inicial del Puerto C.

;Puerto D.
Sin_usoRD0                      equ                     .0; // Sin uso RD0.
Sin_usoRD1                      equ                     .1; // Sin uso RD1.
Sin_usoRD2                      equ                     .2; // Sin uso RD2.
Sin_usoRD3                      equ                     .3; // Sin uso RD3.
Sin_usoRD4                      equ                     .4; // Sin uso RD4.
Sin_usoRD5                      equ                     .5; // Sin uso RD5.
Sin_usoRD6                      equ                     .6; // Sin uso RD6.
Sin_usoRD7                      equ                     .7; // Sin uso RD7.

progd                   equ     b'11111111'; // Programacion Inicial del Puerto D como salidas.

;Puerto E.
Sin_usoRE0                      equ                     .0; // Sin uso RE0.
Sin_usoRE1                      equ                     .1; // Sin uso RE1.
Sin_usoRE2                      equ                     .2; // Sin uso RE2.

proge                   equ     b'111'; // Programacion inicial del Puerto E como salidas.
;-----------------------------------------------------------------------------------------------------
                                        ;================
                                        ;==Vector Reset==
                                        ;================
                        org 0x0000;                             // dirección de inicio de la memoria donde el IDE
comenzara a ensamblar
vec_reset               clrf PCLATH;                    // Limpia el registro PCLATH
                        goto prog_prin;                 // ve para la etiqueta prog_ini
;-----------------------------------------------------------------------------------------------------  ;-----------------------------------------------
-----------------------------------------------------
                ;============================
                ;== Subrutina de Interrupciones  ==
                ;============================
            org 0004h;          //Direccion de memoria donde se encuentra la subrtuina de servicio de interrupcion
vec_int
           movwf res_w;         //Respaldar el estado del registro w
            movf status,w;      //Respaldar las banderas de la alu en la variable auxiliar res_status
           movwf res_status;  //Respaldar las banderas de la alu en la variable auxiliar res_status
            clrf status;        //Limpia el registro STATUS
            movf pclath,w;      //Respaldar el estado del registro pclath en la variable auxiliar res_pclath
            movwf res_pclath;   //Respaldar el estado del registro pclath en la variable auxiliar res_pclath
            clrf pclath;        //Limpia el registro pclath
            movf fsr,w;         //Respaldar el estado del registro fsr en la variable auxiliar res_sfr
            movwf res_fsr;      //Respaldar el estado del registro fsr en la variable auxiliar res_sfr

            btfsc intcon,t0if;  //Si el bit t0if del registro intcon es igual a 0 salta
            call rutina_int;  //LLamada a la subrutina de interrupciones

sal_int     movlw .131;         //Carga al TMR0 la constante de inicio de conteo 131
            movwf tmr0;         //Carga al TMR0 la constante de inicio de conteo 131
```

```
            movf res_fsr,w;          //Regresa el contenido reslpaldado en res_fsr al registro fsr
            movwf fsr;               //Regresa el contenido reslpaldado en res_fsr al registro fsr
            movf res_pclath,w;       //Regresa el contenido reslpaldado en res_pclath al registro pclath
            movwf pclath;            //Regresa el contenido reslpaldado en res_pclath al registro pclath
            movf res_status,w;       //Regresa el contenido reslpaldado en res_status al registro status
                  movwf status;              //Regresa el contenido reslpaldado en res_status al registro status
            movf res_w,w;            //Regresa el contenido reslpaldado en res_r al registro w

            retfie;                  //Regresar al programa principal
;-------------------------------------------------------------------------------------------------

                ;=============================
                ;== Subrutina de Interrupciones  ==
                ;=============================
rutina_int  incf cont_milis,f;       //Incrementa la variable cont milis en una unidad y guarda en el mismo registro
            incf presc_1,f;          //Incrementa la variable presc 1 en una unidad y guarda en el mismo registro

            movlw .100;              //Carga a w la varible con la cual se compara el contenido de prescalador
            xorwf presc_1,w;         //XOR entre registro presc 1 y el registro de trabajo
           btfsc status,z;           //Si el bit z del registro status es igual a 0 salta
            goto sig_int;            //Ve a la siguiente condicion del prescalador
            goto sal_rutint;         //Ve a la salida de la subrutina

          sig_int    clrf presc_1;   //Limpia el registro presc 1
            incf presc_2,f;          //Incrementa la variable presc 2 y guarda en el mismo registro
            movlw .10;               //Carga a w 10 v variablecual se compara el contenido de prescalador
           xorwf presc_2,w;          //XOR entre registro presc 2 y el registro de trabajo
            btfss status,z;          //Si el bit z del registro status es igual a 1 salta
           goto sal_rutint;          //Ve a la salida de la subrutina
            clrf presc_1;            //Limpia el registro presc 1
            clrf presc_2;            //Limpia el registro presc 2

sal_rutext   bsf banderas,ban_int;//Pon a 1 el bit ban int del registro banderas(retardo 1s)

sal_rutint  bcf intcon,t0if;  //Pon a 0 el bit bandera t0if puesto a 1 por la interrupcion
            return;                          //Regresar al programa principal

;-------------------------------------------------------------------------------------------------------------

                                ;======================
                                ;==Subrutina de inicio==
                                ;======================
prog_ini            bsf STATUS,RP0;              //Coloca al programa  en el bco. 1 de ram
                                movlw 0x02;                  // Mueve la constante 0X02 al registro w
                                movwf OPTION_REG ^0x80;      // Configura el preescalador y Desactiva los pull-up
                                movlw proga;                 // Configura al porta
                                movwf TRISA ^0x80;           // como entradas sin uso
                                movlw progb;                 // Configura al portb como salidas
                                movwf TRISB ^0x80;           // salidas  de tranmisión de datos
                                movlw progc;                 // Configura al portc
                                movwf TRISC ^0x80;           // como entradas  sin uso
                                movlw progd;                 // Configura al portd
                                movwf TRISD ^0x80;           // como entradas  sin uso
                                movlw proge;                 // Configura al porte
                                movwf TRISE ^0x80;           // como entradas  sin uso
                                movlw 0x06;                  // Configuración del registro
                                movwf ADCON1 ^0x80;          // ADCON1
```

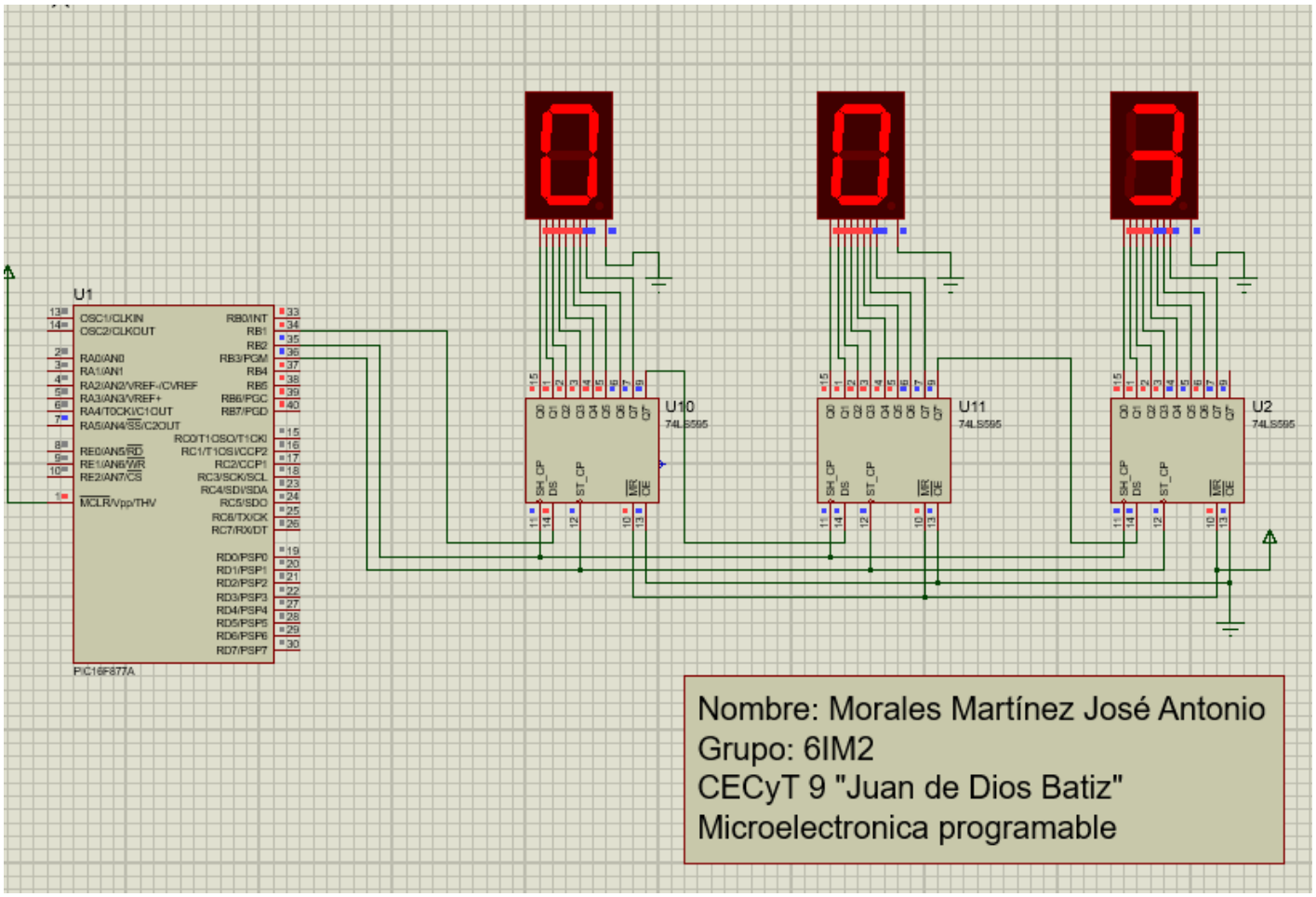
```
                        bcf     STATUS,RP0;             //regresa al bco. 0 de ram

        movlw 0xa0;                     // Habilita la interrupcion del TMR0, Las globales
        movwf intcon;                   // y borra las banderas de interrupción
        movlw .131;                     //Se fija el numero en binario desde
        movwf tmr0;                     //comenzará la cuenta en binario

                        clrf portd;                     //Inicializa la salida de datos serial
                        clrf Reg_TXL;                   //Inicializa los registros de transmisión de datos
                        clrf Reg_TXM;                   //Inicializa los registros de transmisión de datos
                        clrf Reg_TXH;                   //Inicializa los registros de transmisión de datos

                        return;                         //Regresa de la subrutina de inicializacion
;-------------------------------------------------------------------------------------------------------------

                        ;======================
                        ;==Programa Principal==
                        ;======================
prog_prin       call prog_ini;                  //Llamada a la subrutina de inicio


cuenta_time
                        movf cta_uniseg,w;              //Respalda el contenido de cta_uniseg
                        movwf temporal;                 //en la variable temporal
                        call convbin_7seg;              //Llamada a la subrutina de conversion de binario a 7
segmentos
                        movf temporal,w;                //Envia el dato en 7 segmentos al registro que se encargará
                        movwf Reg_TXH;                  //de la transmisión serie


                        movf cta_decseg,w;              //Respalda el contenido de cta_decseg
                        movwf temporal;                 //en la variable temporal
                        call convbin_7seg;              //Llamada a la subrutina de conversion de binario a 7
segmentos
                        movf temporal,w;                //Envia el dato en 7 segmentos al registro que se encargará
                        movwf Reg_TXM;                  //de la transmisión serie


                        movf cta_censeg,w;              //Respalda el contenido de cta_censeg
                        movwf temporal;                 //en la variable temporal
                        call convbin_7seg;              //Llamada a la subrutina de conversion de binario a 7
segmentos
                        movf temporal,w;                //Envia el dato en 7 segmentos al registro que se encargará
                        movwf Reg_TXL;                  //de la transmisión serie

                        call TX_Datos;                  //Llamada a ala subrutina que envia los datos de forma serial
                        call retardo_1s;        //Retardo en la aparición de caracteres en los displays
;-------------------------------------------------------------------------------------------------------------

                        incf cta_uniseg,f;      //Incrementa la variable cta unidades de segundo
                        movlw .10;              // Resta entre el contenido de cta_uniseg
                        subwf cta_uniseg,w;     // y 10 decimal
                        btfss status,Z;         //Compara el resultado en caso de que no sea cero
                        goto cuenta_time;       //Si es 1 muestra el caracter en los displays
                        clrf cta_uniseg;        //De lo contrario reinicia el contenido de las unidades de segundo

                        incf cta_decseg,f;      //Incrementa la variable cta decenas de segundo
                        movlw .10;              // Resta entre el contenido de cta_decseg
                        subwf cta_decseg,w;     // y 10 decimal
```

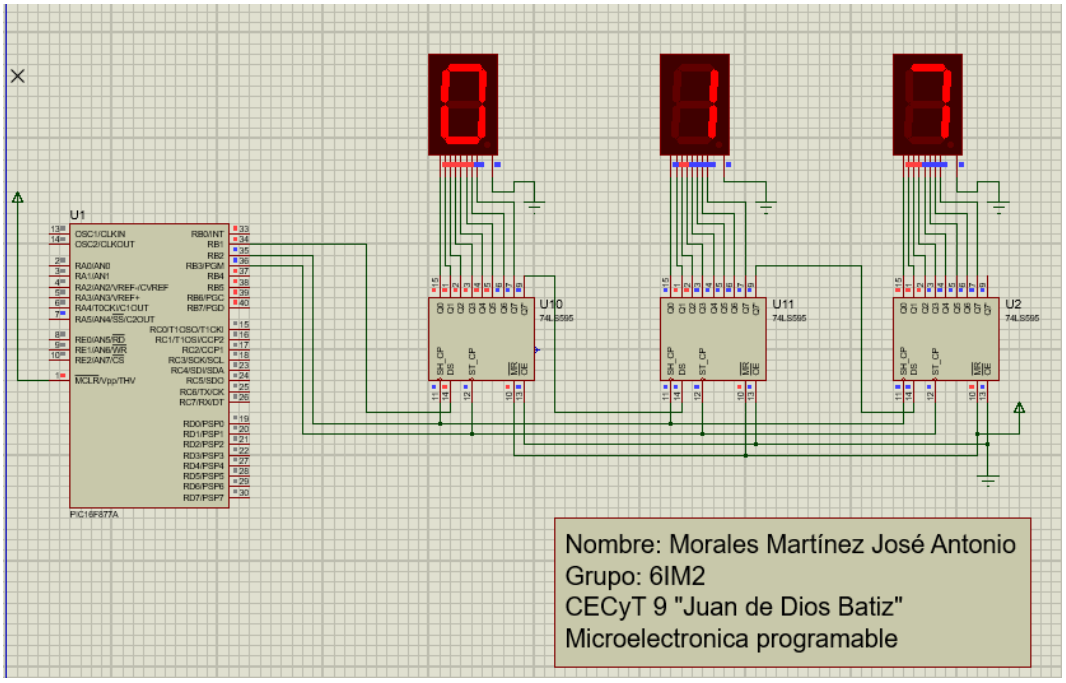
```
                        btfss status,Z;         //Compara el resultado en caso de que no sea cero
                        goto cuenta_time;       //Si es 1 muestra el caracter en los displays
                        clrf cta_uniseg;        //De lo contrario Reinicia el contenido de las unidades de segundo
                        clrf cta_decseg;        //y las decenas de segundo

                        incf cta_censeg,f;      //Incrementa la variable cta centenas de segundo
                        movlw .10;              //Resta entre el contenido de cta_censeg
                        subwf cta_censeg,w;     //y 10 decimal
                        btfss status,Z;         //Compara el resultado en caso de que no sea cero
                        goto cuenta_time;       //Si es 1 muestra el caracter en los displays
                        clrf cta_uniseg;        //Si es 1 muestra el caracter en los displays
                        clrf cta_decseg;        // decenas de segundo
                        clrf cta_censeg;        //y las centenas de segundos
                        goto cuenta_time;       //Reincia la cuenta en 000



;-------------------------------------------------------------------------------------------------

                        ;=====================================================
                        ;          ===Subrutina de Tx. de datos en formato serial =====
                        ;=====================================================

TX_Datos        movlw .24;                      //Se realizaran 24 iteraciones para envir los
                        movwf contador;         //24 bits que conforman el word de datos
sig_TX
                        rlf Reg_TXL,f;          //Rota un bit el contenido del registro de trasmición de datos alto
                        rlf Reg_TXM,f;          //Rota un bit el contenido del registro de trasmición de datos medio
                        rlf Reg_TXH,f;          //Rota un bit el contenido del registro de trasmición de datos bajo

                        btfss status,c;         //Escanea el estado del bit C el cual es el destino del bit más
significativo al rotar
                        goto bit_en_cero;       //Cuando el bit es 0 se pondra a 0 el bit de data In
                        bsf  portb,Data_in;     //Cuando el bit es 1 se pondra a 1 el bit de data In
                        goto gen_pulser;        //Pulso que permite la entrada de datos al integrado 74LS595
bit_en_cero;
                        bcf  portb,Data_in;     //Cuando el bit es 0 se pondra a 0 el bit de data In
                        nop;                    //No operacion,para activar/desactivar dos bits consecutivamnete se
requiere un nop entre ellos
gen_pulser
                        bsf  portb,CLK_Serie;   //Flanco de subida del pulso CLK
                        nop;                    //Espera en estado alto
                        bcf  portb,CLK_Serie;   //Flanco de bajada del pulso CLK
                        decfsz contador,f;      //Disminuye el contador, finaliza una vez que se enviaron todos los
bits serie
                        goto sig_TX;            //Rota y envia el siguiente bit del conjunto de bits que se enviaran de
forma serie

                        bsf portb,CLK_Paralelo; //Una vez que se enviaron los datos a cada uno de 16 FF D en daisy
chain
                        nop;                    //Esta información pasará a los FFs pararelelos
                        bcf portb,CLK_Paralelo; //Mediante el pulso CLK_Paralelo

                        return;


;-------------------------------------------------------------------------------------------------
                        ;=================================================
```

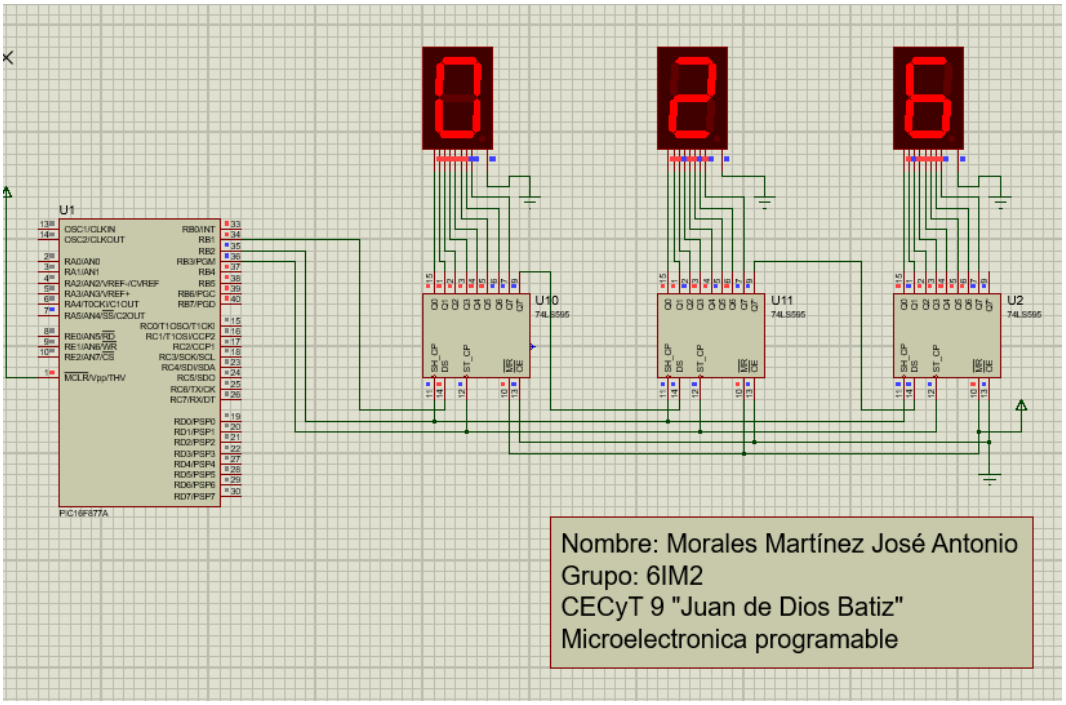
```
                        ;==Subrutina de convertidor de binario a 7 segmentos=
                        ;================================================
;-------------------------------------------------------------------------------------------
convbin_7seg    movlw .0;                       //Cargar 0 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_cero;          // ve para fue_cero

                        movlw .1;               //Cargar 1 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_uno;           // ve para fue_uno

                        movlw .2;               //Cargar 2 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_dos;           // ve para fue_dos

                        movlw .3;               //Cargar 3 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_tres;          // ve para fue_tres

                        movlw .4;               /Cargar 4 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_cuatro;        // ve para fue_cuatro

                        movlw .5;               //Cargar 5 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_cinco;         // ve para fue_cinco

                        movlw .6;               //Cargar 6 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_seis;          // ve para fue_seis

                        movlw .7;               //Cargar 6 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_siete;         // ve para fue_siete

                        movlw .8;               //Cargar 7 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_ocho;          // ve para fue_siete

                        movlw .9;               //Cargar 8 binario al registro w
                        subwf temporal,w;       //Resta entre el registro temporal y w y guardalo en el registro w
                        btfsc status,Z;         //Si el bit Z del registro STATUS es igual a 0 salta
                        goto fue_nueve;         // ve para fue_ocho
                        goto sal_subcov7seg;



fue_cero                movlw car_0;            //Regresa el 0 binario en
                        movwf temporal;         //0 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina
```

```
fue_uno                 movlw car_1;            //Regresa el 1 binario en
                        movwf temporal;         //1 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina
fue_dos                 movlw car_2;            //Regresa el 2 binario en
                        movwf temporal;         //2 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina
fue_tres                movlw car_3;            //Regresa el 3 binario en
                        movwf temporal;         //3 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina
fue_cuatro              movlw car_4;            //Regresa el 4 binario en
                        movwf temporal;         //4 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina
fue_cinco               movlw car_5;            //Regresa el 5 binario en
                        movwf temporal;         //5 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina
fue_seis                movlw car_6;            //Regresa el 6 binario en
                        movwf temporal;         //6 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina
fue_siete               movlw car_7;            //Regresa el 7 binario en
                        movwf temporal;         //7 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina
fue_ocho                movlw car_8;            //Regresa el 8 binario en
                        movwf temporal;         //8 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina
fue_nueve               movlw car_9;            //Regresa el 9 binario en
                        movwf temporal;         //9 7 segmentos a  traves del registro temporal
                        goto sal_subcov7seg; // Ve a la salida de la subrutina


sal_subcov7seg  return;                         //Retorno de la subrutina


;-------------------------------------------------------------------------------------------------


                        ;==============================
                        ;==Subrutina de retardo de 1ms=
                        ;==============================

retardo_1ms             clrf cont_milis;        //Limpia el registro cont milis
loop_1ms                movlw .1;               //Mueve la constante 1 al registro de trabajo
                        subwf cont_milis,w;     //Resta entre el registro cont milis menos el registro de trabajo
                        btfss status,Z;         //Si el bit Z del registro STATUS es igual a 1 salta
                        goto loop_1ms;          //Ve para la etiqueta loop_1ms

                        return;                 //regresa de la subrutina

;-------------------------------------------------------------------------------------------------


                        ;==============================
                        ;==Subrutina de retardo de 250ms=
                        ;==============================

retardo_250ms           clrf cont_milis;        //Limpia el registro cont milis
loop_250                movlw .250;                     //Mueve la constante 250 al registro de trabajo
                        subwf cont_milis,w;             //Resta entre el registro cont milis menos el registro de trabaj
                        btfss status,Z;                 //Si el bit Z del registro STATUS si es igual a 1 salta de lo contrario
ejecuta normalmente
```

```
                    goto loop_250;              //Ve para la etiqueta loop_41ms

                    return;                     //regresa de la subrutina

;-------------------------------------------------------------------------------------------------

                    ;==============================
                    ;==Subrutina de retardo de 1s=
                    ;==============================
retardo_1s

                    call retardo_250ms;         //4 peridiidas de 250ms
                    call retardo_250ms;         //para formar un retardo
                    call retardo_250ms;         //total de 1s
                    call retardo_250ms;
                    return;                             //regresa de la subrutina

;-------------------------------------------------------------------------------------------------


end                                                     //Fin del programa
```







Valor de la sección PARTE B: **4 PUNTOS.**

VALOR TOTAL DEL EXAMEN: 5 PUNTOS.