



Instituto Politécnico Nacional

Escuela Superior de Cómputo

Introducción a los Microcontroladores

Montes Meza Adbel Anahí
“Ejercicios Básicos con un PIC”

Práctica 1

3CM13

Equipo 10

Montaño Reyes Jennifer

Pardo Alanis Arturo Isaac

Ortega Alcocer Humberto Alejandro

21 de Noviembre del 2022

Práctica 1 (Índice)

Práctica 1 (Índice)

1. Objetivo.

1.1. Objetivo del ejercicio uno.

1.2. Objetivo del ejercicio dos.

1.2.1. Objetivo del ejercicio dos (versión A)

1.2.2. Objetivo del ejercicio dos (versión B)

1.3. Objetivo del ejercicio tres

1.4. Objetivo del ejercicio cuatro

2. Lista de Materiales

2.1. Materiales Físicos (Hardware)

2.2. Materiales Virtuales (Software)

3. Desarrollo

3.1. Primer ejercicio

3.1.1. Diagrama de Bloques (Esquemático)

3.1.2. Diagrama de Flujo (UML)

3.1.3. Código

3.1.4. Simulación

3.2. Segundo ejercicio (versión A)

3.2.1. Diagrama de Bloques (Esquemático)

3.2.2. Diagrama de Flujo (UML)

3.2.3. Código

3.2.4. Simulación

3.2.4.1. Suma

3.2.4.2. Resta

3.2.4.3. División

3.2.4.4. Multiplicación

3.3. Segundo ejercicio (versión B)

3.3.1. Diagrama de Bloques (Esquemático)

3.3.2. Diagrama de Flujo (UML)

3.3.3. Código

3.3.4. Simulación

3.3.4.1. Suma

3.3.4.2. Resta

3.3.4.3. Multiplicación

3.3.4.4. División

3.4. Tercer ejercicio

3.4.1. Diagrama de Bloques (Esquemático)

3.4.2. Diagrama de Flujo (UML)

3.4.3. Código

3.4.4. Simulación

[3.5. Cuarto ejercicio](#)

[3.5.1. Diagrama de Bloques \(Esquemático\)](#)

[3.5.2. Diagrama de Flujo \(UML\)](#)

[3.5.3. Código](#)

[3.5.4. Simulación](#)

[4. Hoja de Firmas](#)

[5. Conclusiones](#)

[5.1. Montaña Reyes Jennifer](#)

[5.2. Pardo Alanis Arturo Isaac](#)

[5.3. Ortega Alcocer Humberto Alejandro](#)

1. Objetivo.

En esta práctica se busca realizar 3 ejercicios básicos con nuestro microcontrolador, el *dsPIC30F3013*. Estos ejercicios tendrán como objetivo general mostrar las capacidades, cualidades y funcionamiento del dispositivo, así como el flujo de trabajo que se deberá seguir con el microcontrolador para llevar a cabo el desarrollo de aplicaciones y módulos a futuro.

Cada uno de los ejercicios planteados por la profesora cuentan con un objetivo particular, sin embargo, en lo general podemos decir que se buscará establecer flujos de ejecución con su circuitería asociada para llevar a cabo la operación de un programa simple de E/S, una calculadora y el uso de un *display* de siete segmentos de cátodo común.

1.1. Objetivo del ejercicio uno.

En este ejercicio se buscará leer una entrada a partir de un *dipswitch* y mostrar la misma información en un arreglo de leds correspondiente. El microcontrolador actuará como un “replicador”, en tanto el dispositivo leerá la entrada, realizará un corrimiento de bits para ajustar el dato a los puertos disponibles y, finalmente, la mostrará en los LEDs correspondientes.

1.2. Objetivo del ejercicio dos.

Este ejercicio consistirá en realizar una calculadora básica. Nuestro microcontrolador deberá recibir un *código de operación* (previamente definido por la profesora) con el cuál se procederá a realizar la operación en cuestión. Los códigos de operación definidos son:

| Código de Operación (Binario) | Operación |
|-------------------------------|----------------|
| 00 | Suma |
| 01 | Resta |
| 10 | Multiplicación |
| 11 | División |

Así, nuestro microcontrolador leerá en dos puertos la combinación del código de operación y realizará la operación en cuestión con un valor fijo en el código. Este ejercicio cuenta con dos variaciones (que serán llamadas A y B), las cuales difieren en el modo de realizar las operaciones particulares de Multiplicación y División.

1.2.1. Objetivo del ejercicio dos (versión A)

En esta versión usaremos las funciones integradas en el lenguaje ensamblador para realizar la Multiplicación y División, es decir: **MUL** y **DIV**.

1.2.2. Objetivo del ejercicio dos (versión B)

En esta versión usaremos corrimientos de bits para realizar las operaciones ya que, como sabemos, un corrimiento *a la izquierda* es equivalente a realizar una multiplicación, y un corrimiento *a la derecha* es equivalente a realizar una división.

1.3. Objetivo del ejercicio tres

En este ejercicio se desarrollará un programa en ensamblador que muestre el Número de Boleta del IPN de algún integrante del equipo basándose en un código de operación de entrada. Para esto, hemos seleccionado el número de boleta de **Humberto Alcocer** cuyo número de boleta es **2016630495**.

Usaremos un dipswitch para introducir el código del número a mostrar y se mostrará, en un display de siete segmentos de cátodo común, el dígito correspondiente del número de boleta en cuestión.

La tabla que define el código asociado a cada dígito en el número de la boleta será la siguiente:

| Código (Binario) | Dígito de la Boleta |
|------------------|---------------------|
| 0000 | 2 |
| 0001 | 0 |
| 0010 | 1 |
| 0011 | 6 |
| 0100 | 6 |
| 0101 | 3 |
| 0110 | 0 |
| 0111 | 4 |
| 1000 | 9 |
| 1001 | 5 |

1.4. Objetivo del ejercicio cuatro

En este ejercicio se tomará como base el ejercicio #3 (mostrar el número de boleta de un integrante del equipo) pero añadiendo retardos entre cada “paso” del proceso para así no depender del *dip-switch* y que se muestre todo seguido.

2. Lista de Materiales

Para la realización de la práctica se requieren de distintos materiales tanto físicos (hardware) como virtuales o lógicos (software). El motivo de esto es que realizaremos el flujo de trabajo desde la programación (usando un IDE proporcionado por la profesora) hasta el armado del circuito que mostrará la ejecución del código de manera práctica.

2.1. Materiales Físicos (Hardware)

Los materiales requeridos para esta práctica son:

- 1 DSPIC30f3013.
- 1 Pickit 3.
- 1 Zócalo de programación.
- 14 Resistencias de 1kOhm.
- 1 Display de siete segmentos de cátodo común.
- 1 Dip switch de 4 bits.
- 2 Push button.
- 7 LEDs.
- Cable tipo jumper (a discreción) para pruebas y *debug*.
- Cable electrónico (a discreción).
- Fuente de alimentación 5V.
- Protoboard.
- Computadora Personal con puerto USB.

2.2. Materiales Virtuales (Software)

Los elementos virtuales o lógicos empleados para el desarrollo de la práctica son:

- Sistema Operativo Microsoft Windows 10, 11 o superior.
- MPLAB X IDE v6.05.

3.1.3. Código

```
/**@brief ESTE PROGRAMA MUESTRA LOS BLOQUES QUE FORMAN UN PROGRAMA
; * EN ENSAMBLADOR, LOS BLOQUES SON:
; * BLOQUE 1. OPCIONES DE CONFIGURACIÓN DEL DSC: OSCILADOR, WATCHDOG,
; *     BROWN OUT RESET, POWER ON RESET Y CODIGO DE PROTECCION
; * BLOQUE 2. EQUIVALENCIAS Y DECLARACIONES GLOBALES
; * BLOQUE 3. ESPACIOS DE MEMORIA: PROGRAMA, DATOS X, DATOS Y, DATOS NEAR
; * BLOQUE 4. CODIGO DE APLICACION
; * @device: DSPIC30F4013
; * @oscillator: FRC, 7.3728MHz
; */

.equ __30F3013, 1
.include "p30F3013.inc"

;*****
; BITS DE CONFIGURACIÓN
;*****

;.....
;SE DESACTIVA EL CLOCK SWITCHING Y EL FAIL-SAFE CLOCK MONITOR (FSCM) Y SE
;ACTIVA EL OSCILADOR INTERNO DE 7.3728MHZ (FAST RC) PARA TRABAJAR
;FSCM: PERMITE AL DISPOSITIVO CONTINUAR OPERANDO AUN CUANDO OCURRA UNA FALLA
;EN EL OSCILADOR. CUANDO OCURRE UNA FALLA EN EL OSCILADOR SE GENERA UNA TRAMPA
;Y SE CAMBIA EL RELOJ AL OSCILADOR FRC
;.....

config __FOSC, CSW_FSCM_OFF & FRC

;.....
;SE DESACTIVA EL WATCHDOG
;.....

config __FWDI, WDT_OFF

;.....
;SE ACTIVA EL POWER ON RESET (POR), BROWN OUT RESET (BOR), POWER UP TIMER (PWRT)
;Y EL MASTER CLEAR (MCLR)
;POR: AL MOMENTO DE ALIMENTAR EL DSPIC OCURRE UN RESET CUANDO EL VOLTAJE DE
;ALIMENTACION ALCANZA UN VOLTAJE DE UMBRAL (VPOR), EL CUAL ES 1.85V
;BOR: ESTE MODULO GENERA UN RESET CUANDO EL VOLTAJE DE ALIMENTACION DECAE
;POR DEBAJO DE UN CIERTO UMBRAL ESTABLECIDO (2.7V)
;PWRT: MANTIENE AL DSPIC EN RESET POR UN CIERTO TIEMPO ESTABLECIDO, ESTO AYUDA
;A ASEGURAR QUE EL VOLTAJE DE ALIMENTACION SE HA ESTABILIZADO (16ms)
;.....

config __FBORPOR, PBOR_ON & BORV27 & PWRT_16 & MCLR_EN

;.....
;SE DESACTIVA EL CODIGO DE PROTECCION
;.....

config __FGS, CODE_PROT_OFF & GWRP_OFF

;*****
; SECCION DE DECLARACION DE CONSTANTES CON LA DIRECTIVA .EQU (= DEFINE EN C)
;*****

.equ MUESTRAS, 64      ;NUMERO DE MUESTRAS

;*****
; DECLARACIONES GLOBALES
;*****

;.....
;PROPORCIONA ALCANCE GLOBAL A LA FUNCION _wreg_init, ESTO PERMITE LLAMAR A LA
;FUNCION DESDE UN OTRO PROGRAMA EN ENSAMBLADOR O EN C COLOCANDO LA DECLARACION
;"EXTERN"
;.....

.global _wreg_init

;.....
;ETIQUETA DE LA PRIMER LINEA DE CODIGO
;.....

.global __reset

;.....
;DECLARACION DE LA ISR DEL TIMER 1 COMO GLOBAL
;.....

.global __T1Interrupt
```



```
;*****  
;CONSTANTES ALMACENADAS EN EL ESPACIO DE LA MEMORIA DE PROGRAMA  
;*****  
    .section .myconstbuffer, code  
;.....  
;ALINEA LA SIGUIENTE PALABRA ALMACENADA EN LA MEMORIA  
;DE PROGRAMA A UNA DIRECCIÓN MÚLTIPLO DE 2  
;.....  
    .palign 2  
  
ps_coeff:  
    .hword    0x0002, 0x0003, 0x0005, 0x000A  
;short int ps_coeff[] = {0x0002, 0x0003, 0x0005, 0x000A };  
  
;*****  
;VARIABLES NO INICIALIZADAS EN EL ESPACIO X DE LA MEMORIA DE DATOS  
;*****  
    .section .xbss, bss, xmemory  
  
x_input: .space 2*MUESTRAS      ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE  
;char x_input[128];  
;short int x_input[64];  
;int x_input[32];  
;*****  
;VARIABLES NO INICIALIZADAS EN EL ESPACIO Y DE LA MEMORIA DE DATOS  
;*****  
  
    .section .ybss, bss, ymemory  
  
y_input: .space 2*MUESTRAS      ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE  
;*****  
;VARIABLES NO INICIALIZADAS LA MEMORIA DE DATOS CERCANA (NEAR), LOCALIZADA  
;EN LOS PRIMEROS 8KB DE RAM  
;*****  
    .section .nbss, bss, near  
  
var1:    .space 2                ;LA VARIABLE VARI RESERVA 1 WORD DE ESPACIO  
  
;*****  
;SECCION DE CODIGO EN LA MEMORIA DE PROGRAMA  
;*****  
    .text                        ;INICIO DE LA SECCION DE CODIGO  
  
__reset:  
    MOV     #__SP_init,    W15    ;INICIALIZA EL STACK POINTER  
  
    MOV     #__SPLIM_init, W0     ;INICIALIZA EL REGISTRO STACK POINTER LIMIT  
    MOV     W0,            SPLIM  
  
    NOP  
    ;UN NOP DESPUES DE LA INICIALIZACION DE SPLIM  
  
    CALL    _WREG_INIT         ;SE LLAMA A LA RUTINA DE INICIALIZACION DE REGISTROS  
    ;OPCIONALMENTE USAR RCALL EN LUGAR DE CALL  
  
    CALL    INI_PERIFERICOS  
  
CICLO:  
    MOV     PORTF,          W0  
    NOP  
    LSR     W0,              #2,    W0    ;W0 = W0 >> 2  
    AND     #0X0F,          W0        ;W0 = W0 & 0X000F  
    ;AND W0, #0XF,          W0  
    ;MOV     #0X000F,        W1  
    ;AND     W0,             W1,      W0  
    MOV     W0,              PORTB  
    NOP  
    GOTO    CICLO  
  
;/*@brief ESTA RUTINA INICIALIZA LOS PERIFERICOS DEL DSC  
; */
```

```
INI_PERIFERICOS:
    CLR    PORTF      ;PORTF = 0
    NOP
    CLR    LATF       ;LATF = 0
    NOP
    SETM   TRISF      ;TRISF = 0xFFFF
    NOP

    CLR    PORTB      ;PORTB = 0
    NOP
    CLR    LATB       ;LATB = 0
    NOP
    CLR    TRISB      ;TRISB = 0
    NOP
    SETM   ADPCFG     ;ADPCFG = 0xFFFF
                    ;SE DESHABILITA EL ADC

    RETURN

;/**@brief ESTA RUTINA INICIALIZA LOS REGISTROS Wn A 0x0000
; */
WREG_INIT:
    CLR    W0
    MOV    W0,        W14
    REPEAT #12
    MOV    W0,        [++W14]
    CLR    W14
    RETURN

;/**@brief ISR (INTERRUPT SERVICE ROUTINE) DEL TIMER 1
; * SE USA PUSH.S PARA GUARDAR LOS REGISTROS W0, W1, W2, W3,
; * C, Z, N Y DC EN LOS REGISTROS SOMBRA
; */
__T1Interrupt:
    PUSH.S

    BCLR   IFS0, #T1IF      ;SE LIMPIA LA BANDERA DE INTERRUPCION DEL TIMER 1

    POP.S

    RETFIE                  ;REGRESO DE LA ISR

.END                        ;TERMINACION DEL CÓDIGO DE PROGRAMA EN ESTE ARCHIVO
```

3.1.4. Simulación

The screenshot displays a microcontroller simulation environment. The top section shows assembly code for a loop labeled 'CICLO:' and an initialization routine 'INI_PERIFERICOS:'. The code includes instructions for moving data between registers and ports, performing logical operations, and shifting bits.

```

122  CICLO:
123      MOV     PORTF,    W0
124      NOP
125      LSR     W0,        #2,        W0 ;W0 = W0 >> 2
126      AND     #0X0F,     W0          ;W0 = W0 & 0X000F
127      ;AND W0, #0X0F,     W0
128      ;MOV     #0X000F,   W1
129      ;AND     W0,        W1,        W0
130      MOV     W0,        PORTB
131      NOP
132      GOTO    CICLO
133
134      ;**@brief ESTA RUTINA INICIALIZA LOS PERIFERICOS DEL DSC
135      ; */
136      INI_PERIFERICOS:
  
```

Below the code, the 'Watches' window is visible, showing the current state of several Special Function Registers (SFRs):

| Name | Type | Address | Value |
|-------|------|---------|--------|
| WREG0 | SFR | 0x0 | 0x0002 |
| PORTB | SFR | 0x2C8 | 0x0000 |
| PORTF | SFR | 0x2E0 | 0x0002 |

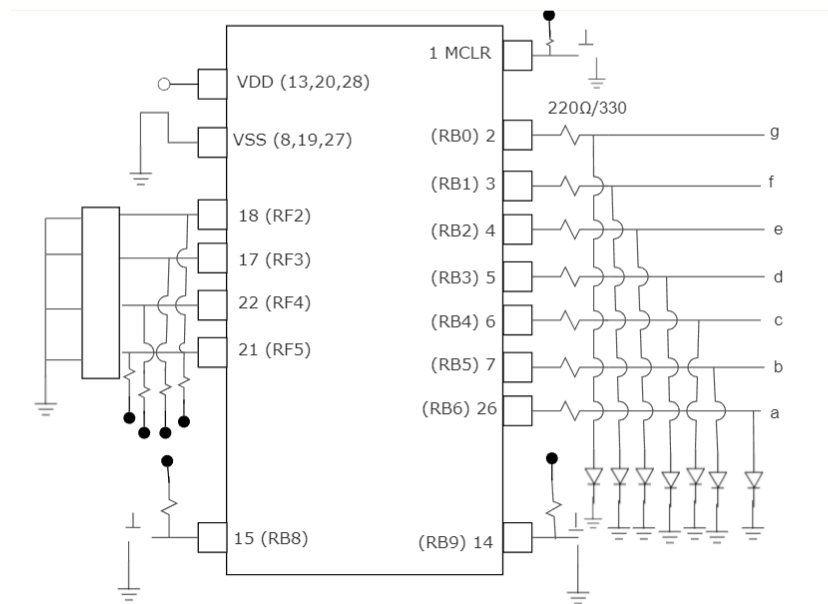
The bottom section of the image shows the same simulation environment after a few more instructions have been executed. The 'Watches' window now shows updated values:

| Name | Type | Address | Value |
|-------|------|---------|--------|
| WREG0 | SFR | 0x0 | 0x0002 |
| PORTB | SFR | 0x2C8 | 0x0002 |
| PORTF | SFR | 0x2E0 | 0x0002 |

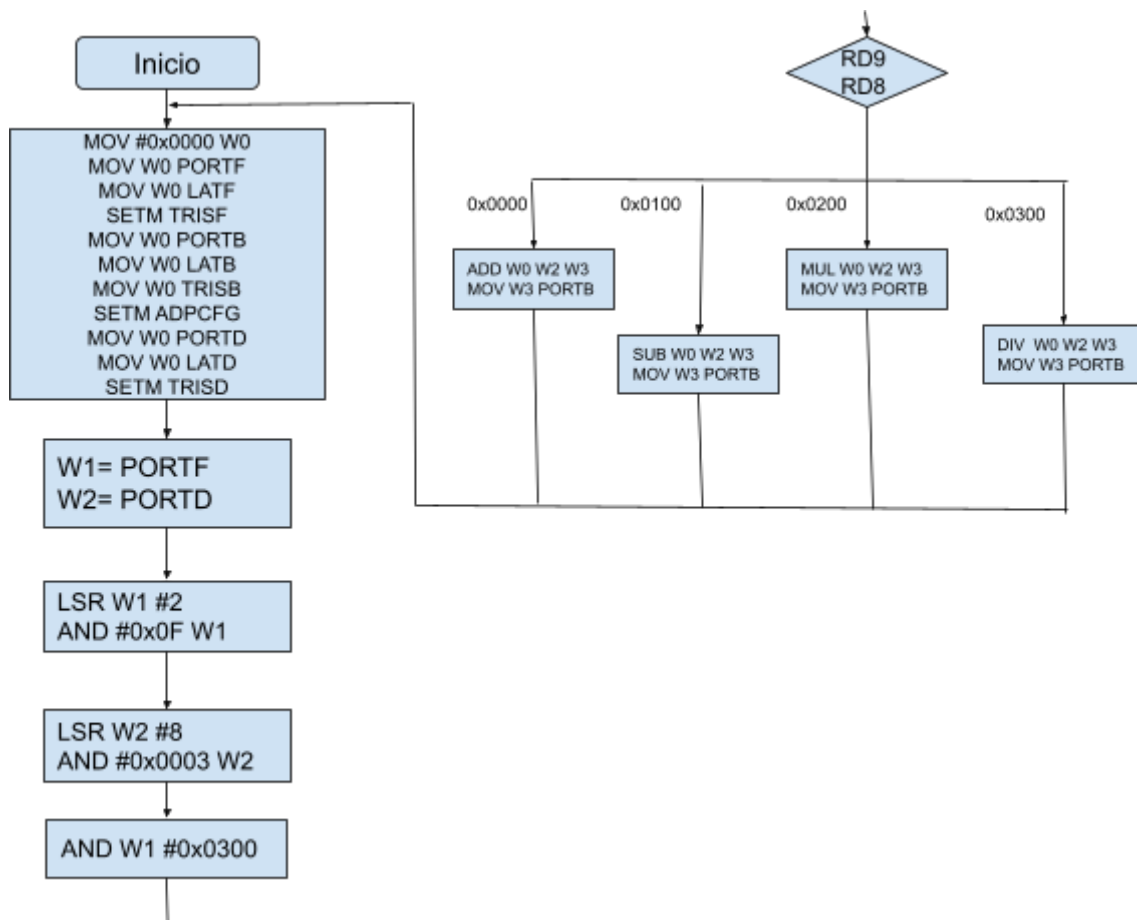
3.2. Segundo ejercicio (versión A)

El segundo ejercicio consiste en programar una calculadora básica, para esta versión (A), usaremos en el código las instrucciones directamente para realizar las operaciones, es decir: MULT, DIV, SUM y DIF.

3.2.1. Diagrama de Bloques (Esquemático)



3.2.2. Diagrama de Flujo (UML)



3.2.3. Código

```
/**@brief ESTE PROGRAMA MUESTRA LOS BLOQUES QUE FORMAN UN PROGRAMA
; * EN ENSAMBLADOR, LOS BLOQUES SON:
; * BLOQUE 1. OPCIONES DE CONFIGURACION DEL DSC: OSCILADOR, WATCHDOG,
; *      BROWN OUT RESET, POWER ON RESET Y CODIGO DE PROTECCION
; * BLOQUE 2. EQUIVALENCIAS Y DECLARACIONES GLOBALES
; * BLOQUE 3. ESPACIOS DE MEMORIA: PROGRAMA, DATOS X, DATOS Y, DATOS NEAR
; * BLOQUE 4. CODIGO DE APLICACION
; * @device: DSPIC30F4013
; * @oscillator: FRC, 7.3728MHz
; */

.equ __30F3013, 1
.include "p30F3013.inc"

;*****
; BITS DE CONFIGURACION
;*****

;.....
;SE DESACTIVA EL CLOCK SWITCHING Y EL FAIL-SAFE CLOCK MONITOR (FSCM) Y SE
;ACTIVA EL OSCILADOR INTERNO DE 7.3728MHZ (FAST RC) PARA TRABAJAR
;FSCM: PERMITE AL DISPOSITIVO CONTINUAR OPERANDO AUN CUANDO OCURRA UNA FALLA
;EN EL OSCILADOR. CUANDO OCURRE UNA FALLA EN EL OSCILADOR SE GENERA UNA TRAMPA
;Y SE CAMBIA EL RELOJ AL OSCILADOR FRC
;.....

config __FOSC, CSW_FSCM_OFF & FRC

;.....
;SE DESACTIVA EL WATCHDOG
;.....

config __FWDI, WDT_OFF

;.....
;SE ACTIVA EL POWER ON RESET (POR), BROWN OUT RESET (BOR), POWER UP TIMER (PWRT)
;Y EL MASTER CLEAR (MCLR)
;POR: AL MOMENTO DE ALIMENTAR EL DSPIC OCURRE UN RESET CUANDO EL VOLTAJE DE
;ALIMENTACION ALCANZA UN VOLTAJE DE UMBRAL (VPOR), EL CUAL ES 1.85V
;BOR: ESTE MODULO GENERA UN RESET CUANDO EL VOLTAJE DE ALIMENTACION DECAE
;POR DEBAJO DE UN CIERTO UMBRAL ESTABLECIDO (2.7V)
;PWRT: MANTIENE AL DSPIC EN RESET POR UN CIERTO TIEMPO ESTABLECIDO, ESTO AYUDA
;A ASEGURAR QUE EL VOLTAJE DE ALIMENTACION SE HA ESTABILIZADO (16ms)
;.....

config __FBORPOR, PBOR_ON & BORV27 & PWRT_16 & MCLR_EN

;.....
;SE DESACTIVA EL CODIGO DE PROTECCION
;.....

config __FGS, CODE_PROT_OFF & GWRP_OFF

;*****
; SECCION DE DECLARACION DE CONSTANTES CON LA DIRECTIVA .EQU (= DEFINE EN C)
;*****

.equ MUESTRAS, 64      ;NUMERO DE MUESTRAS

;*****
; DECLARACIONES GLOBALES
;*****

;.....
;PROPORCIONA ALCANCE GLOBAL A LA FUNCION _wreg_init, ESTO PERMITE LLAMAR A LA
;FUNCION DESDE UN OTRO PROGRAMA EN ENSAMBLADOR O EN C COLOCANDO LA DECLARACION
;"EXTERN"
;.....

.global _wreg_init

;.....
;ETIQUETA DE LA PRIMER LINEA DE CODIGO
;.....

.global __reset

;.....
;DECLARACION DE LA ISR DEL TIMER 1 COMO GLOBAL
;.....

.global __T1Interrupt
```

```

;*****
;CONSTANTES ALMACENADAS EN EL ESPACIO DE LA MEMORIA DE PROGRAMA
;*****
.section .myconstbuffer, code
;.....
;ALINEA LA SIGUIENTE PALABRA ALMACENADA EN LA MEMORIA
;DE PROGRAMA A UNA DIRECCION MULTIPLO DE 2
;.....
.palign 2

ps_coeff:
.hword 0x0002, 0x0003, 0x0005, 0x000A
;short int ps_coeff[] = {0x0002, 0x0003, 0x0005, 0x000A };

;*****
;VARIABLES NO INICIALIZADAS EN EL ESPACIO X DE LA MEMORIA DE DATOS
;*****
.section .xbss, bss, xmemory

x_input: .space 2*MUESTRAS ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE
;char x_input[128];
;short int x_input[64];
;int x_input[32];
;*****
;VARIABLES NO INICIALIZADAS EN EL ESPACIO Y DE LA MEMORIA DE DATOS
;*****

.section .ybss, bss, ymemory

y_input: .space 2*MUESTRAS ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE
;*****
;VARIABLES NO INICIALIZADAS LA MEMORIA DE DATOS CERCANA (NEAR), LOCALIZADA
;EN LOS PRIMEROS 8KB DE RAM
;*****

.section .nbss, bss, near

var1: .space 2 ;LA VARIABLE VARI RESERVA 1 WORD DE ESPACIO

;*****
;SECCION DE CODIGO EN LA MEMORIA DE PROGRAMA
;*****
.text ;INICIO DE LA SECCION DE CODIGO

__reset:
MOV #__SP_init, W15 ;INICIALIZA EL STACK POINTER

MOV #__SPLIM_init, W0 ;INICIALIZA EL REGISTRO STACK POINTER LIMIT
MOV W0, SPLIM

NOP ;UN NOP DESPUES DE LA INICIALIZACION DE SPLIM

CALL _WREG_INIT ;SE LLAMA A LA RUTINA DE INICIALIZACION DE REGISTROS
;OPCIONALMENTE USAR RCALL EN LUGAR DE CALL

CALL INI_PERIFERICOS

CICLO:

MOV PORTF, W1
NOP
LSR W1, #2, W1 ;W1 = W1 >> 2
AND #0X0F, W1 ;W1 = W1 & 0X000F

MOV PORTD, W2
NOP
LSR W2, #8, W2 ;W2 = W2 >> 8
AND #0X0003, W2 ;W0 = W0 & 0X0003

MOV #3, W3

```

```

        BTSS    W2,#0 ;SI ES 1 SE SALTA
        GOTO CEROS
        GOTO UNOS

        GOTO    CICLO
CEROS:
        BTSS    W2,#1 ;SI ES 1 SE SALTA
        GOTO SUMA
        GOTO MULT
UNOS:
        BTSS    W2,#1 ;SI ES 1 SE SALTA
        GOTO RESTA
        GOTO DIVI
SUMA:
        ADD     W1,    W3,    W4
        MOV     W4,    PORTB
        NOP
        GOTO CICLO
RESTA:
        SUB     W1,    W3,    W4
        MOV     W4,    PORTB
        NOP
        GOTO CICLO
MULT:
        MUL.UU  W1,    W3,    W4
        MOV     W4,    PORTB
        NOP
        GOTO CICLO
DIVI:
        REPEAT #17
        DIV.U   W1,W3
        MOV     W0,    PORTB
        NOP
        GOTO CICLO

;/**@brief ESTA RUTINA INICIALIZA LOS PERIFERICOS DEL DSC
; */
INI_PERIFERICOS:
        MOV     #0X0000,    W0
        NOP
        MOV     W0,        PORTF;PORTF = 0X0000
        NOP
        MOV     W0,        LATF;LATF = 0X0000
        NOP
        SETM    TRISF        ;TRISF = 0XFFFF
        NOP
        MOV     W0,        PORTB;PORTB = 0X0000
        NOP
        MOV     W0,        LATB;LATB = 0X0000
        NOP
        MOV     W0,        TRISB;TRISB = 0X0000
        NOP
        SETM    ADPCFG
        NOP
        MOV     W0,        PORTD;PORTD = 0X0000
        NOP
        MOV     W0,        LATD;LATD = 0X0000
        NOP
        SETM    TRISD        ;TRISD = 0XFFFF

        RETURN

;/**@brief ESTA RUTINA INICIALIZA LOS REGISTROS Wn A 0X0000
; */
_WREG_INIT:
        CLR     W0
        MOV     W0,        W14

```

```

REPEAT #12
MOV W0, [++W14]
CLR W14
RETURN

;/**@brief ISR (INTERRUPT SERVICE ROUTINE) DEL TIMER 1
; * SE USA PUSH.S PARA GUARDAR LOS REGISTROS W0, W1, W2, W3,
; * C, Z, N Y DC EN LOS REGISTROS SOMBRA
; */
__T1Interrupt:
PUSH.S

BCLR IFS0, #T1IF ;SE LIMPIA LA BANDERA DE INTERRUPCION DEL TIMER 1

POP.S

RETFIE ;REGRESO DE LA ISR

END ;TERMINACION DEL CODIGO DE PROGRAMA EN ESTE ARCHIVO

```

3.2.4. Simulación

3.2.4.1. Suma

The image displays two screenshots of the Keil uVision IDE, showing the assembly code and the variable window for a simulation.

Top Screenshot:

- Assembly Code:**

```

114 MOV W0, SPLIN
115
116 NOP ;ON NOP DESPUES DE LA INICIALIZACION DE SPLIN
117
118 CALL _WREG_INIT ;SE LLAMA A LA RUTINA DE INICIALIZACION DE REGISTROS
119 ;OPCIONALMENTE USAR RCALL EN LUGAR DE CALL
120 CALL INI_PERIFERIOS
121
122 CICLO:
123
124 MOV PORTF, W1
125
126 NOP
127 LSR W1, R2, W1 ;W1 = W1 >> 2
128 AND R000F, W1 ;W1 = W1 & 0x000F
129

```
- Variable Window:**

| Name | Type | Address | Value |
|-------|------|---------|--------|
| PORTF | SFR | 0x20 | 0x0000 |
| WREG0 | SFR | 0x0 | 0x0000 |
| WREG1 | SFR | 0x2 | 0x0000 |
| PORTB | SFR | 0x2C | 0x0000 |
| PORTD | SFR | 0x24 | 0x0000 |
| WREG2 | SFR | 0x4 | 0x0000 |

Bottom Screenshot:

- Assembly Code:** (Same as the top screenshot, but the execution has progressed to line 128).
- Variable Window:**

| Name | Type | Address | Value |
|-------|------|---------|--------|
| PORTF | SFR | 0x20 | 0x0000 |
| WREG0 | SFR | 0x0 | 0x0000 |
| WREG1 | SFR | 0x2 | 0x0000 |
| PORTB | SFR | 0x2C | 0x0000 |
| PORTD | SFR | 0x24 | 0x0000 |
| WREG2 | SFR | 0x4 | 0x0000 |

The image displays three sequential screenshots of the Keil uVision IDE, showing the assembly source code and the variable watch window for a program named 'p1_2'.

First Screenshot (Lines 125-128):

```

125      NOP                                ;/EN NOP DESPUES DE LA INICIALIZACION DE SPIN
126      CALL _WREG_INIT                    ;/E LLAMA A LA RUTINA DE INICIALIZACION DE REGISTROS
127      CALL INT_PERIFERICOS              ;/OPCIONALMENTE USAR BCALL EN LOSAS DE CALL
128      MOV PORTF, W1
129      LSR W1, #2                          ;W1 = W1 >> 2
130      AND #0X00F, W1                      ;W1 = W1 & 0X000F
131      MOV PORTD, W2
132      LSR W2, #8                          ;W2 = W2 >> 8
133      AND #0X0009, W2                    ;W2 = W2 & 0X0009
134      MOV #0X0000, W5
135      MOV #0X0001, W6
136      MOV #0X0002, W7
137      CPSNE W2, W5
138      GOTO BUGA
139      NOP
140      CPSNE W2, W6
141      GOTO RESTA
142      NOP
143      CPSNE W2, W7
144      GOTO BUGA
145      NOP
146      CPSNE W2, W7
147      GOTO BUGA

```

Variable Watch (First Screenshot):

| Name | Type | Address | Value |
|-------|------|---------|--------|
| PORTF | SFR | 0x2E0 | 0x003C |
| WREG0 | SFR | 0x0 | 0x0000 |
| WREG1 | SFR | 0x2 | 0x000F |
| PORTB | SFR | 0x2C8 | 0x0000 |
| PORTD | SFR | 0x2D4 | 0x0000 |
| WREG2 | SFR | 0x4 | 0x0000 |

Second Screenshot (Lines 129-136):

```

129      MOV PORTF, W1
130      LSR W1, #2                          ;W1 = W1 >> 2
131      AND #0X00F, W1                      ;W1 = W1 & 0X000F
132      MOV PORTD, W2
133      LSR W2, #8                          ;W2 = W2 >> 8
134      AND #0X0009, W2                    ;W2 = W2 & 0X0009
135      MOV #0X0000, W5
136      MOV #0X0001, W6
137      MOV #0X0002, W7
138      CPSNE W2, W5
139      GOTO BUGA
140      NOP
141      CPSNE W2, W6
142      GOTO RESTA
143      NOP
144      CPSNE W2, W7
145      GOTO BUGA
146      NOP
147      CPSNE W2, W7
148      GOTO BUGA

```

Variable Watch (Second Screenshot):

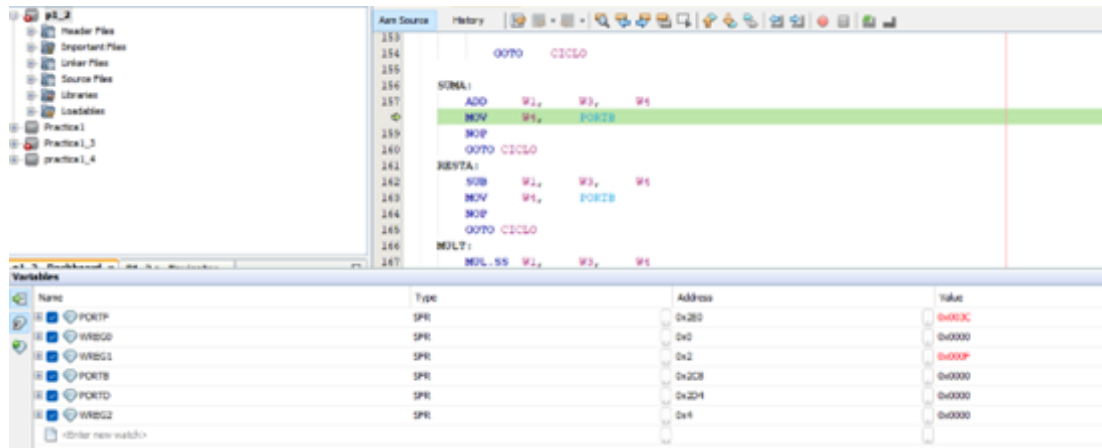
| Name | Type | Address | Value |
|-------|------|---------|--------|
| PORTF | SFR | 0x2E0 | 0x003C |
| WREG0 | SFR | 0x0 | 0x0000 |
| WREG1 | SFR | 0x2 | 0x000F |
| PORTB | SFR | 0x2C8 | 0x0000 |
| PORTD | SFR | 0x2D4 | 0x0000 |
| WREG2 | SFR | 0x4 | 0x0000 |

Third Screenshot (Lines 139-147):

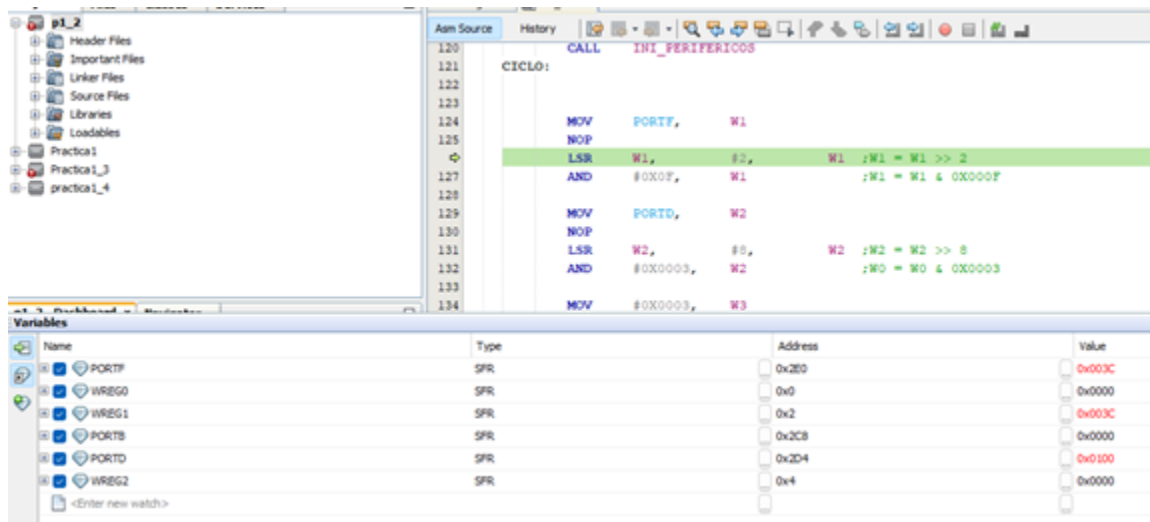
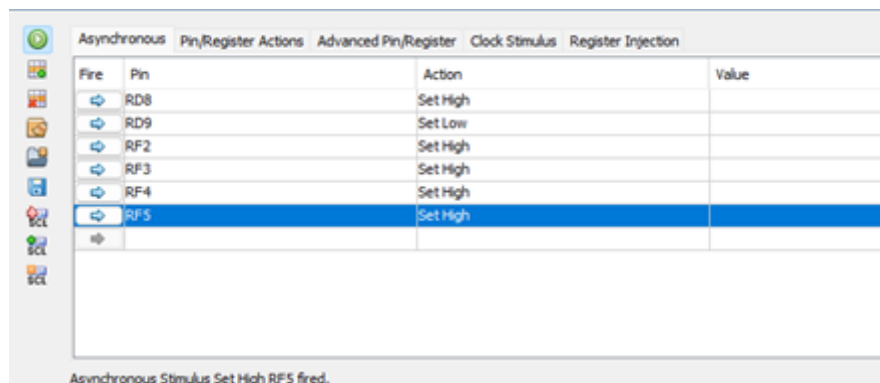
```

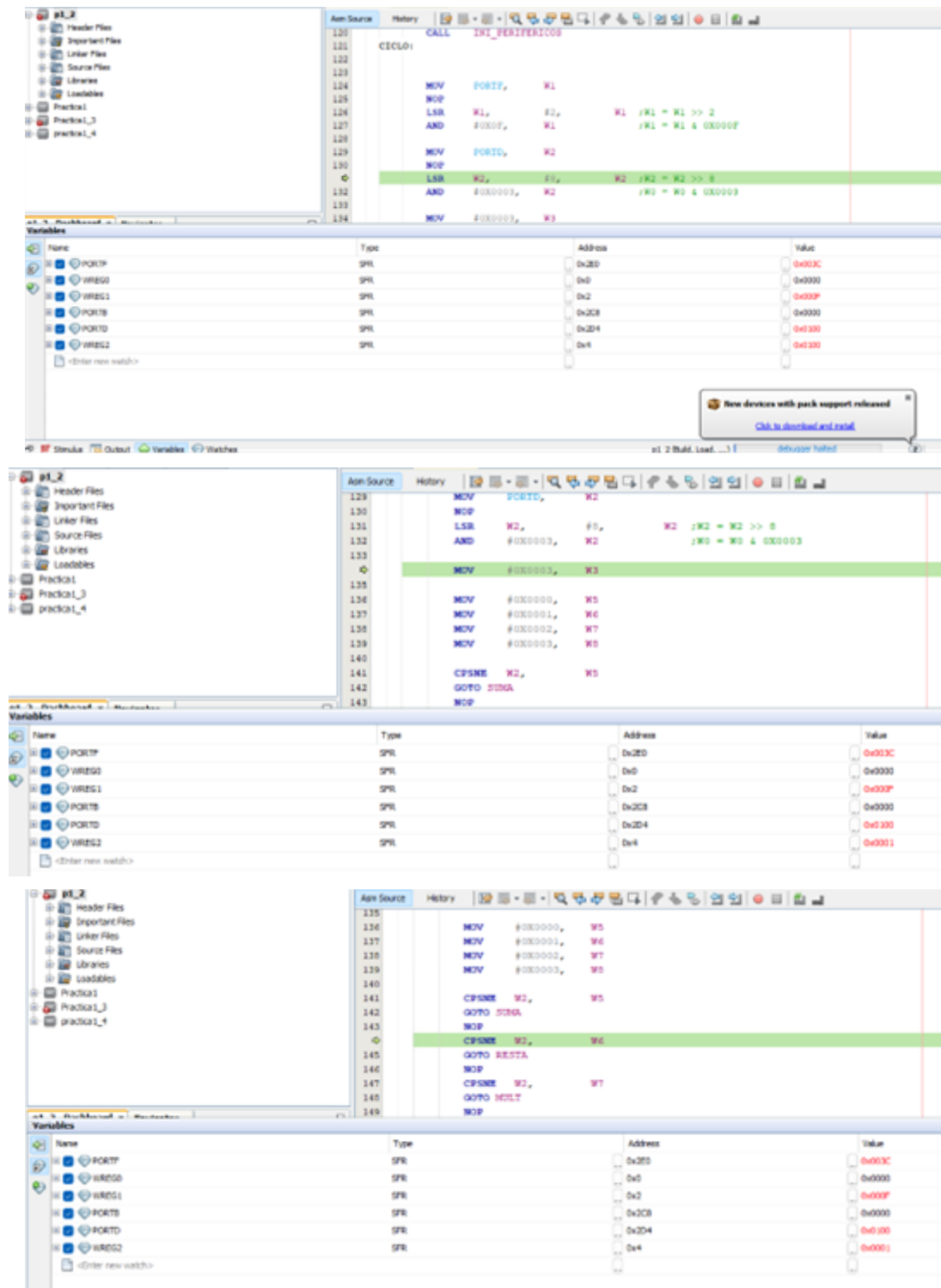
139      MOV #0X0000, W5
140      MOV #0X0001, W6
141      MOV #0X0002, W7
142      MOV #0X0003, W8
143      CPSNE W2, W5
144      GOTO BUGA
145      NOP
146      CPSNE W2, W6
147      GOTO RESTA
148      NOP
149      CPSNE W2, W7
150      GOTO BUGA
151      NOP
152      CPSNE W2, W7
153      GOTO BUGA
154      NOP
155      CPSNE W2, W7
156      GOTO BUGA
157      NOP
158      CPSNE W2, W7
159      GOTO BUGA
160      NOP
161      CPSNE W2, W7
162      GOTO BUGA
163      NOP
164      CPSNE W2, W7
165      GOTO BUGA
166      NOP
167      CPSNE W2, W7
168      GOTO BUGA
169      NOP
170      CPSNE W2, W7
171      GOTO BUGA
172      NOP
173      CPSNE W2, W7
174      GOTO BUGA
175      NOP
176      CPSNE W2, W7
177      GOTO BUGA
178      NOP
179      CPSNE W2, W7
180      GOTO BUGA
181      NOP
182      CPSNE W2, W7
183      GOTO BUGA
184      NOP
185      CPSNE W2, W7
186      GOTO BUGA
187      NOP
188      CPSNE W2, W7
189      GOTO BUGA
190      NOP
191      CPSNE W2, W7
192      GOTO BUGA
193      NOP
194      CPSNE W2, W7
195      GOTO BUGA
196      NOP
197      CPSNE W2, W7
198      GOTO BUGA
199      NOP
200      CPSNE W2, W7
201      GOTO BUGA
202      NOP
203      CPSNE W2, W7
204      GOTO BUGA
205      NOP
206      CPSNE W2, W7
207      GOTO BUGA
208      NOP
209      CPSNE W2, W7
210      GOTO BUGA
211      NOP
212      CPSNE W2, W7
213      GOTO BUGA
214      NOP
215      CPSNE W2, W7
216      GOTO BUGA
217      NOP
218      CPSNE W2, W7
219      GOTO BUGA
220      NOP
221      CPSNE W2, W7
222      GOTO BUGA
223      NOP
224      CPSNE W2, W7
225      GOTO BUGA
226      NOP
227      CPSNE W2, W7
228      GOTO BUGA
229      NOP
230      CPSNE W2, W7
231      GOTO BUGA
232      NOP
233      CPSNE W2, W7
234      GOTO BUGA
235      NOP
236      CPSNE W2, W7
237      GOTO BUGA
238      NOP
239      CPSNE W2, W7
240      GOTO BUGA
241      NOP
242      CPSNE W2, W7
243      GOTO BUGA
244      NOP
245      CPSNE W2, W7
246      GOTO BUGA
247      NOP
248      CPSNE W2, W7
249      GOTO BUGA
250      NOP
251      CPSNE W2, W7
252      GOTO BUGA
253      NOP
254      CPSNE W2, W7
255      GOTO BUGA
256      NOP
257      CPSNE W2, W7
258      GOTO BUGA
259      NOP
260      CPSNE W2, W7
261      GOTO BUGA
262      NOP
263      CPSNE W2, W7
264      GOTO BUGA
265      NOP
266      CPSNE W2, W7
267      GOTO BUGA
268      NOP
269      CPSNE W2, W7
270      GOTO BUGA
271      NOP
272      CPSNE W2, W7
273      GOTO BUGA
274      NOP
275      CPSNE W2, W7
276      GOTO BUGA
277      NOP
278      CPSNE W2, W7
279      GOTO BUGA
280      NOP
281      CPSNE W2, W7
282      GOTO BUGA
283      NOP
284      CPSNE W2, W7
285      GOTO BUGA
286      NOP
287      CPSNE W2, W7
288      GOTO BUGA
289      NOP
290      CPSNE W2, W7
291      GOTO BUGA
292      NOP
293      CPSNE W2, W7
294      GOTO BUGA
295      NOP
296      CPSNE W2, W7
297      GOTO BUGA
298      NOP
299      CPSNE W2, W7
300      GOTO BUGA
301      NOP
302      CPSNE W2, W7
303      GOTO BUGA
304      NOP
305      CPSNE W2, W7
306      GOTO BUGA
307      NOP
308      CPSNE W2, W7
309      GOTO BUGA
310      NOP
311      CPSNE W2, W7
312      GOTO BUGA
313      NOP
314      CPSNE W2, W7
315      GOTO BUGA
316      NOP
317      CPSNE W2, W7
318      GOTO BUGA
319      NOP
320      CPSNE W2, W7
321      GOTO BUGA
322      NOP
323      CPSNE W2, W7
324      GOTO BUGA
325      NOP
326      CPSNE W2, W7
327      GOTO BUGA
328      NOP
329      CPSNE W2, W7
330      GOTO BUGA
331      NOP
332      CPSNE W2, W7
333      GOTO BUGA
334      NOP
335      CPSNE W2, W7
336      GOTO BUGA
337      NOP
338      CPSNE W2, W7
339      GOTO BUGA
340      NOP
341      CPSNE W2, W7
342      GOTO BUGA
343      NOP
344      CPSNE W2, W7
345      GOTO BUGA
346      NOP
347      CPSNE W2, W7
348      GOTO BUGA
349      NOP
350      CPSNE W2, W7
351      GOTO BUGA
352      NOP
353      CPSNE W2, W7
354      GOTO BUGA
355      NOP
356      CPSNE W2, W7
357      GOTO BUGA
358      NOP
359      CPSNE W2, W7
360      GOTO BUGA
361      NOP
362      CPSNE W2, W7
363      GOTO BUGA
364      NOP
365      CPSNE W2, W7
366      GOTO BUGA
367      NOP
368      CPSNE W2, W7
369      GOTO BUGA
370      NOP
371      CPSNE W2, W7
372      GOTO BUGA
373      NOP
374      CPSNE W2, W7
375      GOTO BUGA
376      NOP
377      CPSNE W2, W7
378      GOTO BUGA
379      NOP
380      CPSNE W2, W7
381      GOTO BUGA
382      NOP
383      CPSNE W2, W7
384      GOTO BUGA
385      NOP
386      CPSNE W2, W7
387      GOTO BUGA
388      NOP
389      CPSNE W2, W7
390      GOTO BUGA
391      NOP
392      CPSNE W2, W7
393      GOTO BUGA
394      NOP
395      CPSNE W2, W7
396      GOTO BUGA
397      NOP
398      CPSNE W2, W7
399      GOTO BUGA
400      NOP
401      CPSNE W2, W7
402      GOTO BUGA
403      NOP
404      CPSNE W2, W7
405      GOTO BUGA
406      NOP
407      CPSNE W2, W7
408      GOTO BUGA
409      NOP
410      CPSNE W2, W7
411      GOTO BUGA
412      NOP
413      CPSNE W2, W7
414      GOTO BUGA
415      NOP
416      CPSNE W2, W7
417      GOTO BUGA
418      NOP
419      CPSNE W2, W7
420      GOTO BUGA
421      NOP
422      CPSNE W2, W7
423      GOTO BUGA
424      NOP
425      CPSNE W2, W7
426      GOTO BUGA
427      NOP
428      CPSNE W2, W7
429      GOTO BUGA
430      NOP
431      CPSNE W2, W7
432      GOTO BUGA
433      NOP
434      CPSNE W2, W7
435      GOTO BUGA
436      NOP
437      CPSNE W2, W7
438      GOTO BUGA
439      NOP
440      CPSNE W2, W7
441      GOTO BUGA
442      NOP
443      CPSNE W2, W7
444      GOTO BUGA
445      NOP
446      CPSNE W2, W7
447      GOTO BUGA
448      NOP
449      CPSNE W2, W7
450      GOTO BUGA
451      NOP
452      CPSNE W2, W7
453      GOTO BUGA
454      NOP
455      CPSNE W2, W7
456      GOTO BUGA
457      NOP
458      CPSNE W2, W7
459      GOTO BUGA
460      NOP
461      CPSNE W2, W7
462      GOTO BUGA
463      NOP
464      CPSNE W2, W7
465      GOTO BUGA
466      NOP
467      CPSNE W2, W7
468      GOTO BUGA
469      NOP
470      CPSNE W2, W7
471      GOTO BUGA
472      NOP
473      CPSNE W2, W7
474      GOTO BUGA
475      NOP
476      CPSNE W2, W7
477      GOTO BUGA
478      NOP
479      CPSNE W2, W7
480      GOTO BUGA
481      NOP
482      CPSNE W2, W7
483      GOTO BUGA
484      NOP
485      CPSNE W2, W7
486      GOTO BUGA
487      NOP
488      CPSNE W2, W7
489      GOTO BUGA
490      NOP
491      CPSNE W2, W7
492      GOTO BUGA
493      NOP
494      CPSNE W2, W7
495      GOTO BUGA
496      NOP
497      CPSNE W2, W7
498      GOTO BUGA
499      NOP
500      CPSNE W2, W7
501      GOTO BUGA
502      NOP
503      CPSNE W2, W7
504      GOTO BUGA
505      NOP
506      CPSNE W2, W7
507      GOTO BUGA
508      NOP
509      CPSNE W2, W7
509      GOTO BUGA
510      NOP
511      CPSNE W2, W7
511      GOTO BUGA
512      NOP
513      CPSNE W2, W7
513      GOTO BUGA
514      NOP
515      CPSNE W2, W7
515      GOTO BUGA
516      NOP
517      CPSNE W2, W7
517      GOTO BUGA
518      NOP
519      CPSNE W2, W7
519      GOTO BUGA
520      NOP
521      CPSNE W2, W7
521      GOTO BUGA
522      NOP
523      CPSNE W2, W7
523      GOTO BUGA
524      NOP
525      CPSNE W2, W7
525      GOTO BUGA
526      NOP
527      CPSNE W2, W7
527      GOTO BUGA
528      NOP
529      CPSNE W2, W7
529      GOTO BUGA
530      NOP
531      CPSNE W2, W7
531      GOTO BUGA
532      NOP
533      CPSNE W2, W7
533      GOTO BUGA
534      NOP
535      CPSNE W2, W7
535      GOTO BUGA
536      NOP
537      CPSNE W2, W7
537      GOTO BUGA
538      NOP
539      CPSNE W2, W7
539      GOTO BUGA
540      NOP
541      CPSNE W2, W7
541      GOTO BUGA
542      NOP
543      CPSNE W2, W7
543      GOTO BUGA
544      NOP
545      CPSNE W2, W7
545      GOTO BUGA
546      NOP
547      CPSNE W2, W7
547      GOTO BUGA
548      NOP
549      CPSNE W2, W7
549      GOTO BUGA
550      NOP
551      CPSNE W2, W7
551      GOTO BUGA
552      NOP
553      CPSNE W2, W7
553      GOTO BUGA
554      NOP
555      CPSNE W2, W7
555      GOTO BUGA
556      NOP
557      CPSNE W2, W7
557      GOTO BUGA
558      NOP
559      CPSNE W2, W7
559      GOTO BUGA
560      NOP
561      CPSNE W2, W7
561      GOTO BUGA
562      NOP
563      CPSNE W2, W7
563      GOTO BUGA
564      NOP
565      CPSNE W2, W7
565      GOTO BUGA
566      NOP
567      CPSNE W2, W7
567      GOTO BUGA
568      NOP
569      CPSNE W2, W7
569      GOTO BUGA
570      NOP
571      CPSNE W2, W7
571      GOTO BUGA
572      NOP
573      CPSNE W2, W7
573      GOTO BUGA
574      NOP
575      CPSNE W2, W7
575      GOTO BUGA
576      NOP
577      CPSNE W2, W7
577      GOTO BUGA
578      NOP
579      CPSNE W2, W7
579      GOTO BUGA
580      NOP
581      CPSNE W2, W7
581      GOTO BUGA
582      NOP
583      CPSNE W2, W7
583      GOTO BUGA
584      NOP
585      CPSNE W2, W7
585      GOTO BUGA
586      NOP
587      CPSNE W2, W7
587      GOTO BUGA
588      NOP
589      CPSNE W2, W7
589      GOTO BUGA
590      NOP
591      CPSNE W2, W7
591      GOTO BUGA
592      NOP
593      CPSNE W2, W7
593      GOTO BUGA
594      NOP
595      CPSNE W2, W7
595      GOTO BUGA
596      NOP
597      CPSNE W2, W7
597      GOTO BUGA
598      NOP
599      CPSNE W2, W7
599      GOTO BUGA
600      NOP
601      CPSNE W2, W7
601      GOTO BUGA
602      NOP
603      CPSNE W2, W7
603      GOTO BUGA
604      NOP
605      CPSNE W2, W7
605      GOTO BUGA
606      NOP
607      CPSNE W2, W7
607      GOTO BUGA
608      NOP
609      CPSNE W2, W7
609      GOTO BUGA
610      NOP
611      CPSNE W2, W7
611      GOTO BUGA
612      NOP
613      CPSNE W2, W7
613      GOTO BUGA
614      NOP
615      CPSNE W2, W7
615      GOTO BUGA
616      NOP
617      CPSNE W2, W7
617      GOTO BUGA
618      NOP
619      CPSNE W2, W7
619      GOTO BUGA
620      NOP
621      CPSNE W2, W7
621      GOTO BUGA
622      NOP
623      CPSNE W2, W7
623      GOTO BUGA
624      NOP
625      CPSNE W2, W7
625      GOTO BUGA
626      NOP
627      CPSNE W2, W7
627      GOTO BUGA
628      NOP
629      CPSNE W2, W7
629      GOTO BUGA
630      NOP
631      CPSNE W2, W7
631      GOTO BUGA
632      NOP
633      CPSNE W2, W7
633      GOTO BUGA
634      NOP
635      CPSNE W2, W7
635      GOTO BUGA
636      NOP
637      CPSNE W2, W7
637      GOTO BUGA
638      NOP
639      CPSNE W2, W7
639      GOTO BUGA
640      NOP
641      CPSNE W2, W7
641      GOTO BUGA
642      NOP
643      CPSNE W2, W7
643      GOTO BUGA
644      NOP
645      CPSNE W2, W7
645      GOTO BUGA
646      NOP
647      CPSNE W2, W7
647      GOTO BUGA
648      NOP
649      CPSNE W2, W7
649      GOTO BUGA
650      NOP
651      CPSNE W2, W7
651      GOTO BUGA
652      NOP
653      CPSNE W2, W7
653      GOTO BUGA
654      NOP
655      CPSNE W2, W7
655      GOTO BUGA
656      NOP
657      CPSNE W2, W7
657      GOTO BUGA
658      NOP
659      CPSNE W2, W7
659      GOTO BUGA
660      NOP
661      CPSNE W2, W7
661      GOTO BUGA
662      NOP
663      CPSNE W2, W7
663      GOTO BUGA
664      NOP
665      CPSNE W2, W7
665      GOTO BUGA
666      NOP
667      CPSNE W2, W7
667      GOTO BUGA
668      NOP
669      CPSNE W2, W7
669      GOTO BUGA
670      NOP
671      CPSNE W2, W7
671      GOTO BUGA
672      NOP
673      CPSNE W2, W7
673      GOTO BUGA
674      NOP
675      CPSNE W2, W7
675      GOTO BUGA
676      NOP
677      CPSNE W2, W7
677      GOTO BUGA
678      NOP
679      CPSNE W2, W7
679      GOTO BUGA
680      NOP
681      CPSNE W2, W7
681      GOTO BUGA
682      NOP
683      CPSNE W2, W7
683      GOTO BUGA
684      NOP
685      CPSNE W2, W7
685      GOTO BUGA
686      NOP
687      CPSNE W2, W7
687      GOTO BUGA
688      NOP
689      CPSNE W2, W7
689      GOTO BUGA
690      NOP
691      CPSNE W2, W7
691      GOTO BUGA
692      NOP
693      CPSNE W2, W7
693      GOTO BUGA
694      NOP
695      CPSNE W2, W7
695      GOTO BUGA
696      NOP
697      CPSNE W2, W7
697      GOTO BUGA
698      NOP
699      CPSNE W2, W7
699      GOTO BUGA
700      NOP
701      CPSNE W2, W7
701      GOTO BUGA
702      NOP
703      CPSNE W2, W7
703      GOTO BUGA
704      NOP
705      CPSNE W2, W7
705      GOTO BUGA
706      NOP
707      CPSNE W2, W7
707      GOTO BUGA
708      NOP
709      CPSNE W2, W7
709      GOTO BUGA
710      NOP
711      CPSNE W2, W7
711      GOTO BUGA
712      NOP
713      CPSNE W2, W7
713      GOTO BUGA
714      NOP
715      CPSNE W2, W7
715      GOTO BUGA
716      NOP
717      CPSNE W2, W7
717      GOTO BUGA
718      NOP
719      CPSNE W2, W7
719      GOTO BUGA
720      NOP
721      CPSNE W2, W7
721      GOTO BUGA
722      NOP
723      CPSNE W2, W7
723      GOTO BUGA
724      NOP
725      CPSNE W2, W7
725      GOTO BUGA
726      NOP
727      CPSNE W2, W7
727      GOTO BUGA
728      NOP
729      CPSNE W2, W7
729      GOTO BUGA
730      NOP
731      CPSNE W2, W7
731      GOTO BUGA
732      NOP
733      CPSNE W2, W7
733      GOTO BUGA
734      NOP
735      CPSNE W2, W7
735      GOTO BUGA
736      NOP
737      CPSNE W2, W7
737      GOTO BUGA
738      NOP
739      CPSNE W2, W7
739      GOTO BUGA
740      NOP
741      CPSNE W2, W7
741      GOTO BUGA
742      NOP
743      CPSNE W2, W7
743      GOTO BUGA
744      NOP
745      CPSNE W2, W7
745      GOTO BUGA
746      NOP
747      CPSNE W2, W7
747      GOTO BUGA
748      NOP
749      CPSNE W2, W7
749      GOTO BUGA
750      NOP
751      CPSNE W2, W7
751      GOTO BUGA
752      NOP
753      CPSNE W2, W7
753      GOTO BUGA
754      NOP
755      CPSNE W2, W7
755      GOTO BUGA
756      NOP
757      CPSNE W2, W7
757      GOTO BUGA
758      NOP
759      CPSNE W2, W7
759      GOTO BUGA
760      NOP
761      CPSNE W2, W7
761      GOTO BUGA
762      NOP
763      CPSNE W2, W7
763      GOTO BUGA
764      NOP
765      CPSNE W2, W7
765      GOTO BUGA
766      NOP
767      CPSNE W2, W7
767      GOTO BUGA
768      NOP
769      CPSNE W2, W7
769      GOTO BUGA
770      NOP
771      CPSNE W2, W7
771      GOTO BUGA
772      NOP
773      CPSNE W2, W7
773      GOTO BUGA
774      NOP
775      CPSNE W2, W7
775      GOTO BUGA
776      NOP
777      CPSNE W2, W7
777      GOTO BUGA
778      NOP
779      CPSNE W2, W7
779      GOTO BUGA
780      NOP
781      CPSNE W2, W7
781      GOTO BUGA
782      NOP
783      CPSNE W2, W7
783      GOTO BUGA
784      NOP
785      CPSNE W2, W7
785      GOTO BUGA
786      NOP
787      CPSNE W2, W7
787      GOTO BUGA
788      NOP
789      CPSNE W2, W7
789      GOTO BUGA
790      NOP
791      CPSNE W2, W7
791      GOTO BUGA
792      NOP
793      CPSNE W2, W7
793      GOTO BUGA
794      NOP
795      CPSNE W2, W7
795      GOTO BUGA
796      NOP
797      CPSNE W2, W7
797      GOTO BUGA
798      NOP
799      CPSNE W2, W7
799      GOTO BUGA
800      NOP
801      CPSNE W2, W7
801      GOTO BUGA
802      NOP
803      CPSNE W2, W7
803      GOTO BUGA
804      NOP
805      CPSNE W2, W7
805      GOTO BUGA
806      NOP
807      CPSNE W2, W7
807      GOTO BUGA
808      NOP
809      CPSNE W2, W7
809      GOTO BUGA
810      NOP
811      CPSNE W2, W7
811      GOTO BUGA
812      NOP
813      CPSNE W2, W7
813      GOTO BUGA
814      NOP
815      CPSNE W2, W7
815      GOTO BUGA
816      NOP
817      CPSNE W2, W7
817      GOTO BUGA
818      NOP
819      CPSNE W2, W7
819      GOTO BUGA
820      NOP
821      CPSNE W2, W7
821      GOTO BUGA
822      NOP
823      CPSNE W2, W7
823      GOTO BUGA
824      NOP
825      CPSNE W2, W7
825      GOTO BUGA
826      NOP
827      CPSNE W2, W7
827      GOTO BUGA
828      NOP
829      CPSNE W2, W7
829      GOTO BUGA
830      NOP
831      CPSNE W2, W7
831      GOTO BUGA
832      NOP
833      CPSNE W2, W7
833      GOTO BUGA
834      NOP
835      CPSNE W2, W7
835      GOTO BUGA
836      NOP
837      CPSNE W2, W7
837      GOTO BUGA
838      NOP
839      CPSNE W2, W7
839      GOTO BUGA
840      NOP
841      CPSNE W2, W7
841      GOTO BUGA
842      NOP
843      CPSNE W2, W7
843      GOTO BUGA
844      NOP
845      CPSNE W2, W7
845      GOTO BUGA
846      NOP
847      CPSNE W2, W7
847      GOTO BUGA
848      NOP
849      CPSNE W2, W7
849      GOTO BUGA
850      NOP
851      CPSNE W2, W7
851      GOTO BUGA
852      NOP
853      CPSNE W2, W7
853      GOTO BUGA
854      NOP
855      CPSNE W2, W7
855      GOTO BUGA
856      NOP
857      CPSNE W2, W7
857      GOTO BUGA
858      NOP
859      CPSNE W2, W7
859      GOTO BUGA
860      NOP
861      CPSNE W2, W7
861      GOTO BUGA
862      NOP
863      CPSNE W2, W7
863      GOTO BUGA
864      NOP
865      CPSNE W2, W7
865      GOTO BUGA
866      NOP
867      CPSNE W2, W7
867      GOTO BUGA
868      NOP
869      CPSNE W2, W7
869      GOTO BUGA
870      NOP
871      CPSNE W2, W7
871      GOTO BUGA
872      NOP
873      CPSNE W2, W7
873      GOTO BUGA
874      NOP
875      CPSNE W2, W7
875      GOTO BUGA
876      NOP
877      CPSNE W2, W7
877      GOTO BUGA
878      NOP
879      CPSNE W2, W7
879      GOTO BUGA
880      NOP
881      CPSNE W2, W7
881      GOTO BUGA
882      NOP
883      CPSNE W2, W7
883      GOTO BUGA
884      NOP
885      CPSNE W2, W7
885      GOTO BUGA
886      NOP
887      CPSNE W2, W7
887      GOTO BUGA
888      NOP
889      CPSNE W2, W7
889      GOTO BUGA
890      NOP
891      CPSNE W2, W7
891      GOTO BUGA
892      NOP
893      CPSNE W2, W7
893      GOTO BUGA
894      NOP
895      CPSNE W2, W7
895      GOTO BUGA
896      NOP
897      CPSNE W2, W7
897      GOTO BUGA
898      NOP
899      CPSNE W2, W7
899      GOTO BUGA
900      NOP
901      CPSNE W2, W7
901      GOTO BUGA
902      NOP
903      CPSNE W2, W7
903      GOTO BUGA
904      NOP
905      CPSNE W2, W7
905      GOTO BUGA
906      NOP
907      CPSNE W2, W7
907      GOTO BUGA
908      NOP
909      CPSNE W2, W7
909      GOTO BUGA
910      NOP
911      CPSNE W2, W7
911      GOTO BUGA
912      NOP
913      CPSNE W2, W7
913      GOTO BUGA
914      NOP
915      CPSNE W2, W7
915      GOTO BUGA
916      NOP
917      CPSNE W2, W7
917      GOTO BUGA
918      NOP
919      CPSNE W2, W7
919      GOTO BUGA
920      NOP
921      CPSNE W2, W7
921      GOTO BUGA
922      NOP
923      CPSNE W2, W7
923      GOTO BUGA
924      NOP
925      CPSNE W2, W7
925      GOTO BUGA
926      NOP
927      CPSNE W2, W7
927      GOTO BUGA
928      NOP
929      CPSNE W2, W7
929      GOTO BUGA
930      NOP
931      CPSNE W2, W7
931      GOTO BUGA
932      NOP
933      CPSNE W2, W7
933      GOTO BUGA
934      NOP
935      CPSNE W2, W7
935      GOTO BUGA
936      NOP
937      CPSNE W2, W7
937      GOTO BUGA
938      NOP
939      CPSNE W2, W7
939      GOTO BUGA
940      NOP
941      CPSNE W2, W7
941      GOTO BUGA
942      NOP
943      CPSNE W2, W7
943      GOTO BUGA
944      NOP
945      CPSNE W2, W7
945      GOTO BUGA
946      NOP
947      CPSNE W2, W7
947      GOTO BUGA
948      NOP
949      CPSNE W2, W7
949      GOTO BUGA
950      NOP
951      CPSNE W2, W7
951      GOTO BUGA
952      NOP
953      CPSNE W2, W7
953      GOTO BUGA
954      NOP
955      CPSNE W2, W7
955      GOTO BUGA
956      NOP
957      CPSNE W2, W7
957      GOTO BUGA
958      NOP
959      CPSNE W2, W7
959      GOTO BUGA
960      NOP
961      CPSNE W2, W7
961      GOTO BUGA
962      NOP
963      CPSNE W2, W7
963      GOTO BUGA
964      NOP
965      CPSNE W2, W7
965      GOTO BUGA
966      NOP
967      CPSNE W2, W7
967      GOTO BUGA
968      NOP
969      CPSNE W2, W7
969      GOTO BUGA
970      NOP
971      CPSNE W2, W7
971      GOTO BUGA
972      NOP
973      CPSNE W2, W7
973      GOTO BUGA
974      NOP
975      CPSNE W2, W7
975      GOTO BUGA
976      NOP
977      CPSNE W2, W7
977      GOTO BUGA
978      NOP
979      CPSNE W2, W7
979      GOTO BUGA
980      NOP
981      CPSNE W2, W7
981      GOTO BUGA
982      NOP
983      CPSNE W2, W7
983      GOTO BUGA
984      NOP
985      CPSNE W2, W7
985      GOTO BUGA
986      NOP
987      CPSNE W2, W7
987      GOTO BUGA
988      NOP
989      CPSNE W2, W7
989      GOTO BUGA
990      NOP
991      CPSNE W2, W7
991      GOTO BUGA
992      NOP
993      CPSNE W2, W7
993      GOTO BUGA
994      NOP
995      CPSNE W2, W7
995      GOTO BUGA
996      NOP
997      CPSNE W2, W7
997      GOTO BUGA
998      NOP
999      CPSNE W2, W7
999      GOTO BUGA
1000      NOP
1001      CPSNE W2, W7
1001      GOTO BUGA
1002      NOP
1003      CPSNE W2, W7
1003      GOTO BUGA
1004      NOP
1005      CPSNE W2, W7
1005      GOTO BUGA
1006      NOP
1007      CPSNE W2, W7
1007      GOTO BUGA
1008      NOP
1009      CPSNE W2, W7
1009      GOTO BUGA
1010      NOP
1011      CPSNE W2, W7
1011      GOTO BUGA
1012      NOP
1013      CPSNE W2, W7
1013      GOTO BUGA
1014      NOP
1015      CPSNE W2, W7
1015      GOTO BUGA
1016      NOP
1017      CPSNE W2, W7
1017      GOTO BUGA
1018      NOP
1019      CPSNE W2, W7
1019      GOTO BUGA
1020      NOP
1021      CPSNE W2, W7
1021      GOTO BUGA
1022      NOP
1023      CPSNE W2, W7
1023      GOTO BUGA
1024      NOP

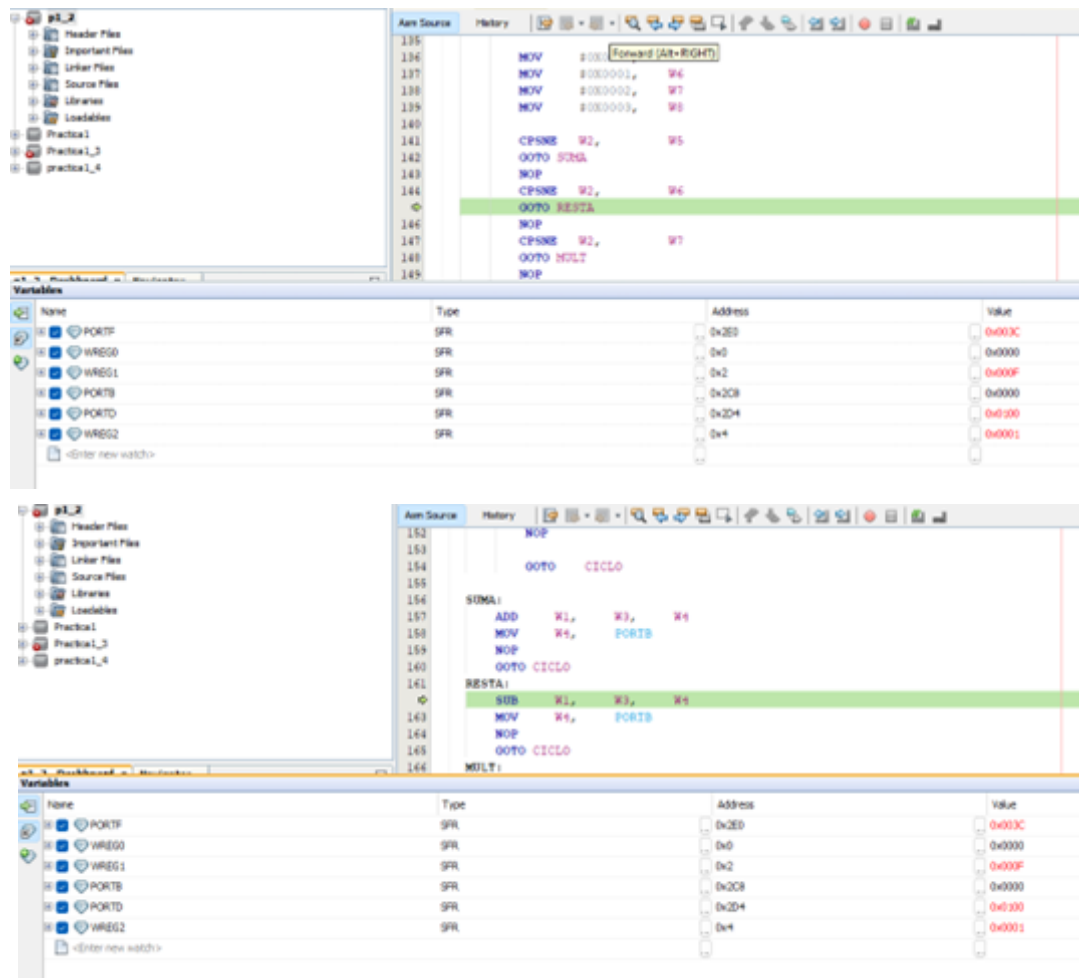
```



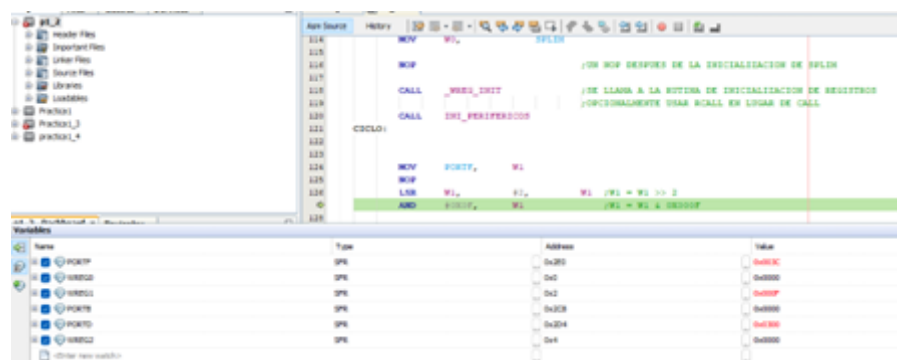
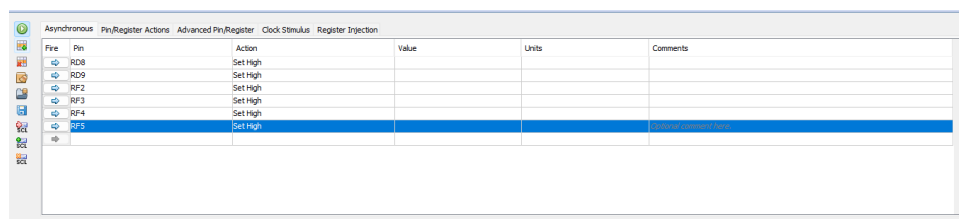
3.2.4.2. Resta



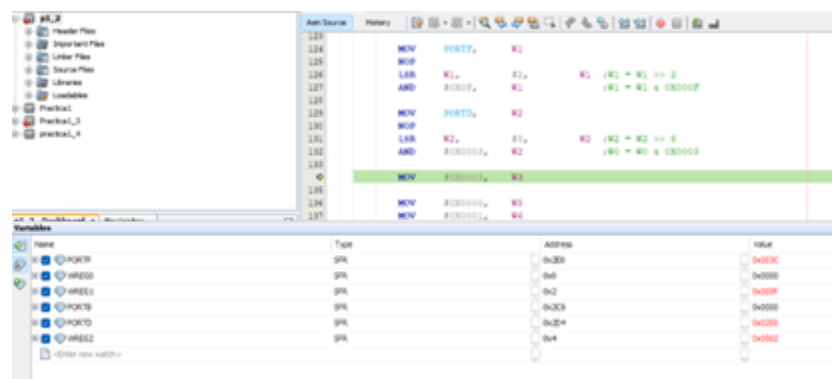
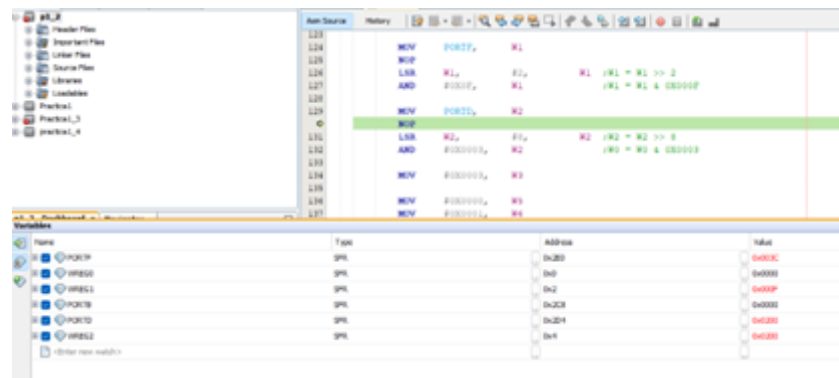
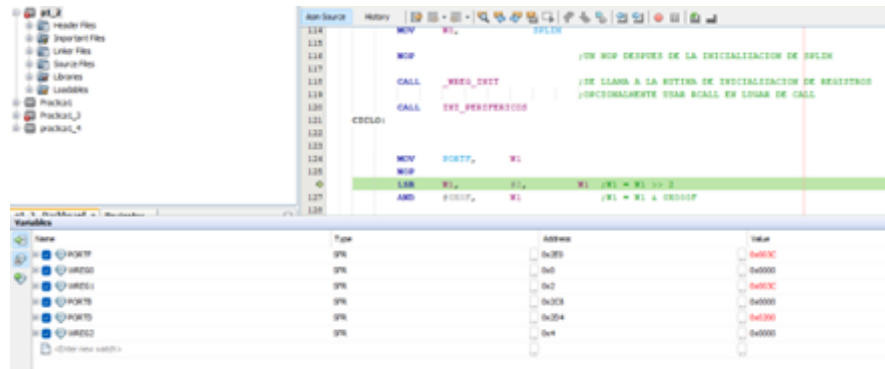
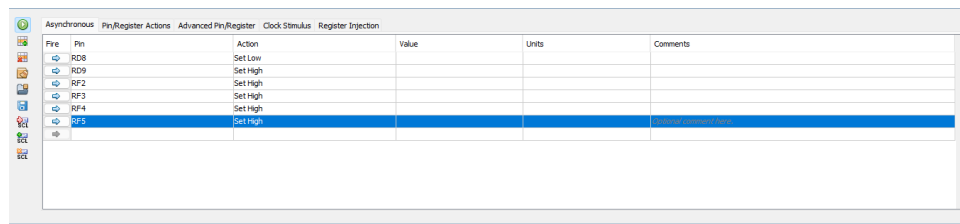


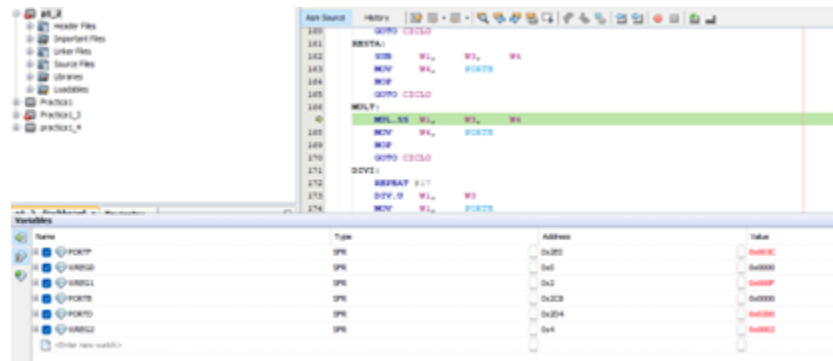


3.2.4.3. División



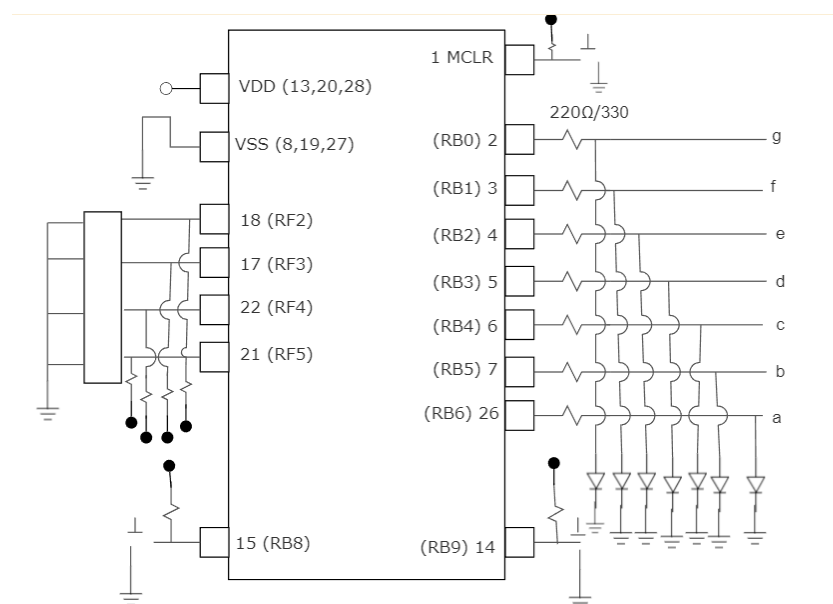
[illegible]



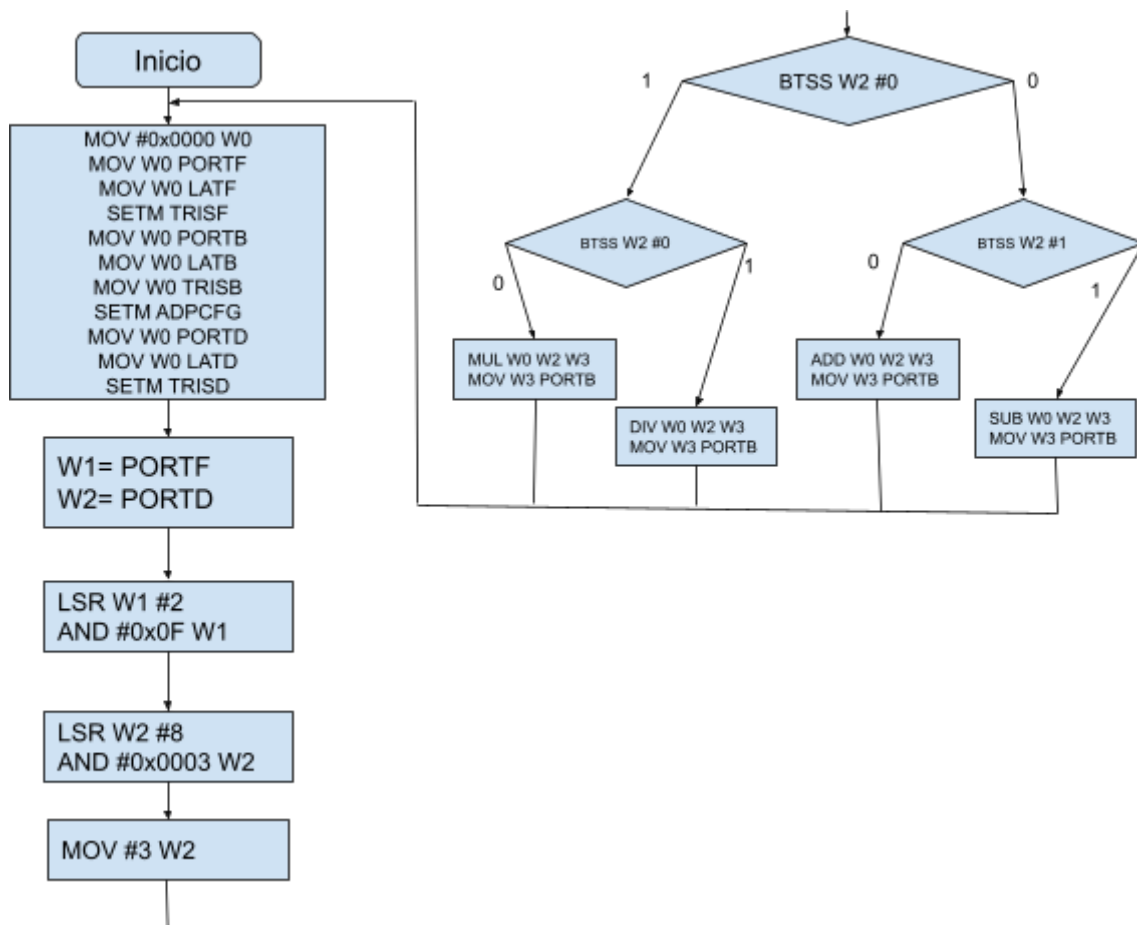


Siguiendo con la versión A, este segundo ejercicio realiza una calculadora básica *pero*, a diferencia de la primer versión, aquí utilizamos corrimientos de bits para realizar las operaciones directamente.

3.3.1. Diagrama de Bloques (Esquemático)



3.3.2. Diagrama de Flujo (UML)



3.3.3. Código

```

/**@brief ESTE PROGRAMA MUESTRA LOS BLOQUES QUE FORMAN UN PROGRAMA
 * EN ENSAMBLADOR, LOS BLOQUES SON:
 * BLOQUE 1. OPCIONES DE CONFIGURACION DEL DSC: OSCILADOR, WATCHDOG,
 * BROWN OUT RESET, POWER ON RESET Y CODIGO DE PROTECCION
 * BLOQUE 2. EQUIVALENCIAS Y DECLARACIONES GLOBALES
 * BLOQUE 3. ESPACIOS DE MEMORIA: PROGRAMA, DATOS X, DATOS Y, DATOS NEAR
 * BLOQUE 4. CODIGO DE APLICACION
 * @device: DSPIC30F4013
 * @oscillator: FRC, 7.3728MHz
 */

.equ __30F3013, 1
.include "p30F3013.inc"

;*****
; BITS DE CONFIGURACION
;*****
;.....
;SE DESACTIVA EL CLOCK SWITCHING Y EL FAIL-SAFE CLOCK MONITOR (FSCM) Y SE
;ACTIVA EL OSCILADOR INTERNO DE 7.3728MHZ(FAST RC) PARA TRABAJAR
;FSCM: PERMITE AL DISPOSITIVO CONTINUAR OPERANDO AUN CUANDO OCURRA UNA FALLA
;EN EL OSCILADOR. CUANDO OCURRE UNA FALLA EN EL OSCILADOR SE GENERA UNA TRAMPA
;Y SE CAMBIA EL RELOJ AL OSCILADOR FRC
;.....
config __FOSC, CSW_FSCM_OFF & FRC
;.....
;SE DESACTIVA EL WATCHDOG
;.....
config __FWDTP, WDT_OFF
;.....

```



```
;SE ACTIVA EL POWER ON RESET (POR), BROWN OUT RESET (BOR), POWER UP TIMER (PWRT)
;Y EL MASTER CLEAR (MCLR)
;POR: AL MOMENTO DE ALIMENTAR EL DSPIC OCURRE UN RESET CUANDO EL VOLTAJE DE
;ALIMENTACION ALCANZA UN VOLTAJE DE UMBRAL (VPOR), EL CUAL ES 1.85V
;BOR: ESTE MODULO GENERA UN RESET CUANDO EL VOLTAJE DE ALIMENTACION DECAE
;POR DEBAJO DE UN CIERTO UMBRAL ESTABLECIDO (2.7V)
;PWRT: MANTIENE AL DSPIC EN RESET POR UN CIERTO TIEMPO ESTABLECIDO, ESTO AYUDA
;A ASEGURAR QUE EL VOLTAJE DE ALIMENTACION SE HA ESTABILIZADO (16ms)
;.....
        config __FBORPOR, PBOR_ON & BORV27 & PWRT_16 & MCLR_EN
;.....
;SE DESACTIVA EL CODIGO DE PROTECCION
;.....
        config __FGS, CODE_PROT_OFF & GWRP_OFF
;.....
;*****
; SECCION DE DECLARACION DE CONSTANTES CON LA DIRECTIVA .EQU (= DEFINE EN C)
;*****
        .equ MUESTRAS, 64           ;NUMERO DE MUESTRAS
;*****
; DECLARACIONES GLOBALES
;*****
;.....
;PROPORCIONA ALCANCE GLOBAL A LA FUNCION _wreg_init, ESTO PERMITE LLAMAR A LA
;FUNCION DESDE UN OTRO PROGRAMA EN ENSAMBLADOR O EN C COLOCANDO LA DECLARACION
;"EXTERN"
;.....
        .global _wreg_init
;.....
;ETIQUETA DE LA PRIMER LINEA DE CODIGO
;.....
        .global __reset
;.....
;DECLARACION DE LA ISR DEL TIMER 1 COMO GLOBAL
;.....
        .global __T1Interrupt
;.....
;*****
;CONSTANTES ALMACENADAS EN EL ESPACIO DE LA MEMORIA DE PROGRAMA
;*****
        .section .myconstbuffer, code
;.....
;ALINEA LA SIGUIENTE PALABRA ALMACENADA EN LA MEMORIA
;DE PROGRAMA A UNA DIRECCION MULTIPLO DE 2
;.....
        .palign 2

ps_coeff:
        .hword    0x0002, 0x0003, 0x0005, 0x000A
;short int ps_coeff[] = {0x0002, 0x0003, 0x0005, 0x000A };
;*****
;VARIABLES NO INICIALIZADAS EN EL ESPACIO X DE LA MEMORIA DE DATOS
;*****
        .section .xbss, bss, xmemory

x_input: .space 2*MUESTRAS           ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE
;char x_input[128];
;short int x_input[64];
;int x_input[32];
;*****
;VARIABLES NO INICIALIZADAS EN EL ESPACIO Y DE LA MEMORIA DE DATOS
;*****
        .section .ybss, bss, ymemory

y_input: .space 2*MUESTRAS           ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE
```

```
;*****  
;VARIABLES NO INICIALIZADAS LA MEMORIA DE DATOS CERCANA (NEAR), LOCALIZADA  
;EN LOS PRIMEROS 8KB DE RAM  
;*****  
        .section .nbss, bss, near  
  
var1:    .space 2                ;LA VARIABLE VAR1 RESERVA 1 WORD DE ESPACIO  
  
;*****  
;SECCION DE CODIGO EN LA MEMORIA DE PROGRAMA  
;*****  
.text                ;INICIO DE LA SECCION DE CODIGO  
  
__reset:  
    MOV    #__SP_init,    W15    ;INICIALIZA EL STACK POINTER  
  
    MOV    #__SPLIM_init, W0     ;INICIALIZA EL REGISTRO STACK POINTER LIMIT  
    MOV    W0,            SPLIM  
  
    NOP                                ;UN NOP DESPUES DE LA INICIALIZACION DE SPLIM  
  
    CALL    _WREG_INIT              ;SE LLAMA A LA RUTINA DE INICIALIZACION DE REGISTROS  
                                ;OPCIONALMENTE USAR RCALL EN LUGAR DE CALL  
    CALL    INI_PERIFERICOS  
  
CICLO:  
  
    MOV    PORTF,          W1  
    NOP  
    LSR    W1,             #2,      W1    ;W1 = W1 >> 2  
    AND    #0X0F,          W1        ;W1 = W1 & 0X000F  
  
    MOV    PORTD,          W2  
    NOP  
    LSR    W2,             #8,      W2    ;W2 = W2 >> 8  
    AND    #0X0003,        W2        ;W0 = W0 & 0X0003  
  
    MOV    #3,             W3  
  
    BTSS   W2,#0 ;SI ES 1 SE SALTA  
    GOTO   CEROS  
    GOTO   UNOS  
  
    GOTO   CICLO  
  
CEROS:  
    BTSS   W2,#1 ;SI ES 1 SE SALTA  
    GOTO   SUMA  
    GOTO   MULT  
  
UNOS:  
    BTSS   W2,#1 ;SI ES 1 SE SALTA  
    GOTO   RESTA  
    GOTO   DIVI  
  
SUMA:  
    ADD    W1,             W3,      W4  
    MOV    W4,             PORTB  
    NOP  
    GOTO   CICLO  
  
RESTA:  
    SUB    W1,             W3,      W4  
    MOV    W4,             PORTB  
    NOP  
    GOTO   CICLO  
  
MULT:  
    MUL.UU W1,             W3,      W4  
    MOV    W4,             PORTB  
    NOP  
    GOTO   CICLO  
  
DIVI:
```

```
REPEAT #17
DIV.U    W1,W3
MOV      W0,    PORTB
NOP
GOTO CICLO

;/**@brief ESTA RUTINA INICIALIZA LOS PERIFERICOS DEL DSC
; */
INI_PERIFERICOS:
MOV      #0X0000,    W0
NOP
MOV      W0,    PORTF;PORTF = 0X0000
NOP
MOV      W0,    LATF;LATF = 0X0000
NOP
SETM     TRISF      ;TRISF = 0XFFFF
NOP
MOV      W0,    PORTB;PORTB = 0X0000
NOP
MOV      W0,    LATB;LATB = 0X0000
NOP
MOV      W0,    TRISB;TRISB = 0X0000
NOP
SETM     ADPCFG
NOP
MOV      W0,    PORTD;PORTD = 0X0000
NOP
MOV      W0,    LATD;LATD = 0X0000
NOP
SETM     TRISD      ;TRISD = 0XFFFF

RETURN

;/**@brief ESTA RUTINA INICIALIZA LOS REGISTROS Wn A 0X0000
; */
_WREG_INIT:
CLR      W0
MOV      W0,    W14
REPEAT   #12
MOV      W0,    [++W14]
CLR      W14
RETURN

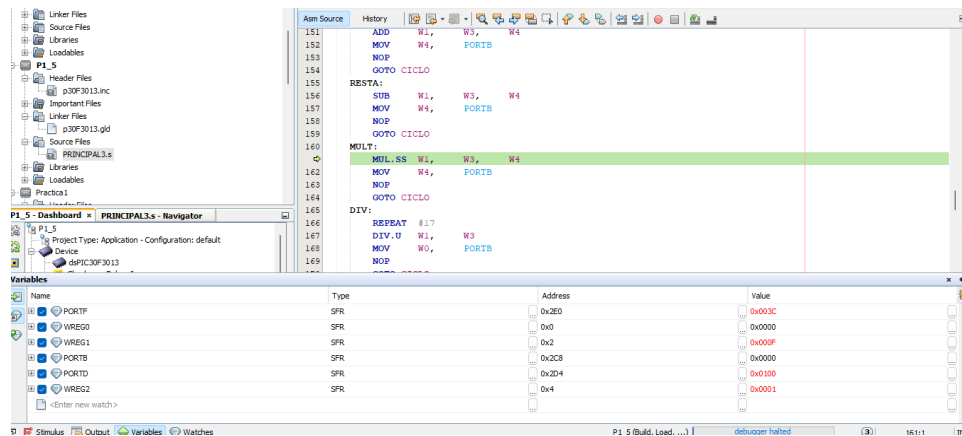
;/**@brief ISR (INTERRUPT SERVICE ROUTINE) DEL TIMER 1
; * SE USA PUSH.S PARA GUARDAR LOS REGISTROS W0, W1, W2, W3,
; * C, Z, N Y DC EN LOS REGISTROS SOMBRA
; */
_T1Interrupt:
PUSH.S

BCLR     IFS0, #T1IF      ;SE LIMPIA LA BANDERA DE INTERRUPCION DEL TIMER 1

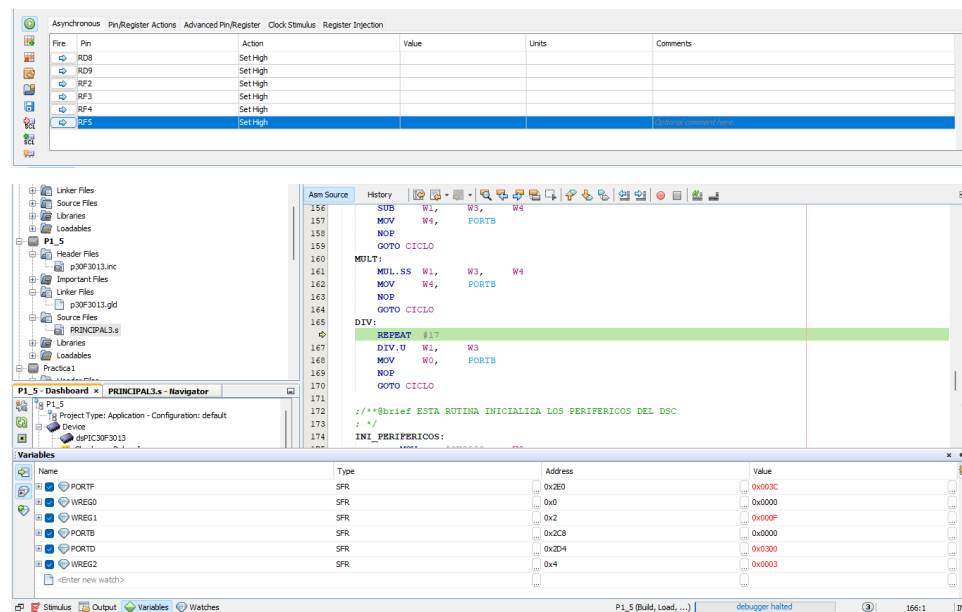
POP.S

RETFIE      ;REGRESO DE LA ISR

END          ;TERMINACION DEL CODIGO DE PROGRAMA EN ESTE ARCHIVO
```

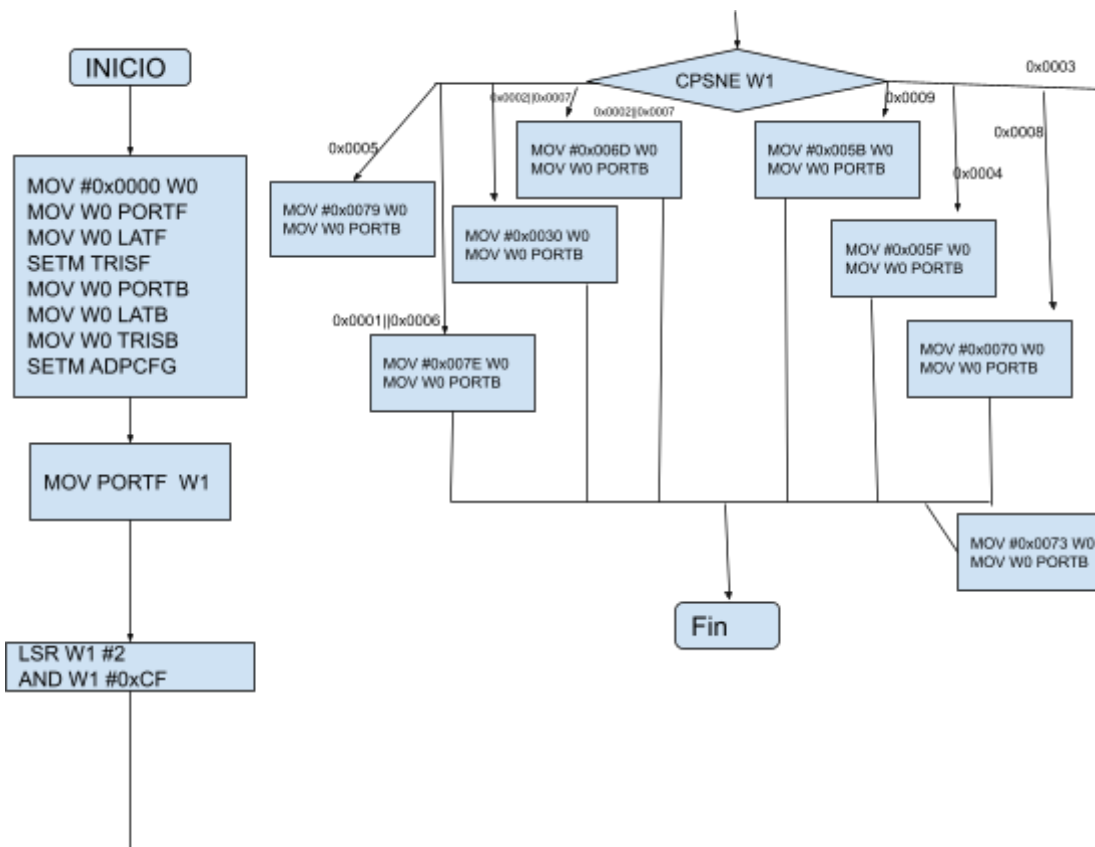
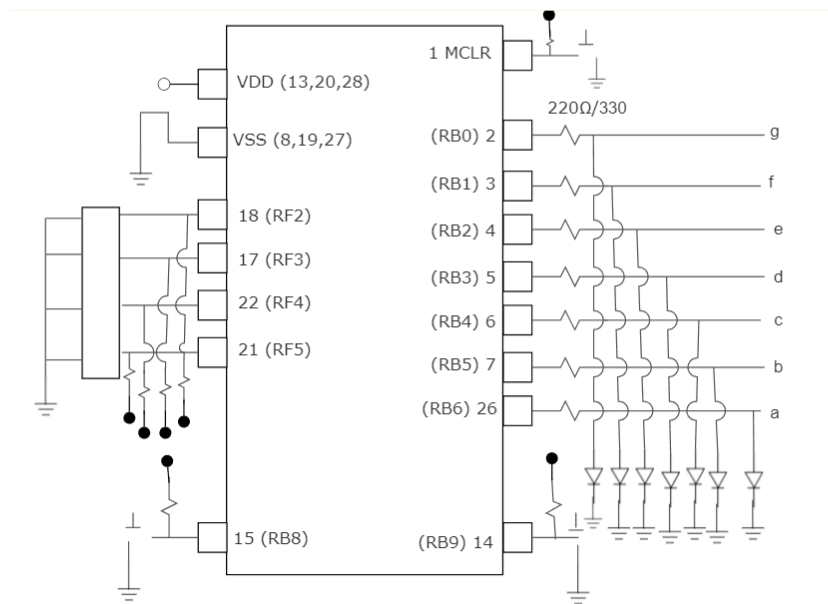



3.3.4.4. División



3.4. Tercer ejercicio

El tercer ejercicio consiste en mostrar un número de boleta en un display de siete segmentos de cátodo común, para esto el usuario colocará el ID del número a mostrar (10 dígitos), en un *dip-switch* y se deberá mostrar en el display el dígito del número de boleta correspondiente.



```

;/**@brief ESTE PROGRAMA MUESTRA LOS BLOQUES QUE FORMAN UN PROGRAMA
; * EN ENSAMBLADOR, LOS BLOQUES SON:
; * BLOQUE 1. OPCIONES DE CONFIGURACION DEL DSC: OSCILADOR, WATCHDOG,
; * BROWN OUT RESET, POWER ON RESET Y CODIGO DE PROTECCION

```

```
; * BLOQUE 2. EQUIVALENCIAS Y DECLARACIONES GLOBALES
; * BLOQUE 3. ESPACIOS DE MEMORIA: PROGRAMA, DATOS X, DATOS Y, DATOS NEAR
; * BLOQUE 4. CÓDIGO DE APLICACIÓN
; * @device: DSPIC30F4013
; * @oscillator: FRC, 7.3728MHz
; */

.equ __30F3013, 1
.include "p30F3013.inc"
;*****
; BITS DE CONFIGURACIÓN
;*****
;.....
;SE DESACTIVA EL CLOCK SWITCHING Y EL FAIL-SAFE CLOCK MONITOR (FSCM) Y SE
;ACTIVA EL OSCILADOR INTERNO DE 7.3728MHZ(FAST RC) PARA TRABAJAR
;FSCM: PERMITE AL DISPOSITIVO CONTINUAR OPERANDO AUN CUANDO OCURRA UNA FALLA
;EN EL OSCILADOR. CUANDO OCURRE UNA FALLA EN EL OSCILADOR SE GENERA UNA TRAMPA
;Y SE CAMBIA EL RELOJ AL OSCILADOR FRC
;.....
config __FOSC, CSW_FSCM_OFF & FRC
;.....
;SE DESACTIVA EL WATCHDOG
;.....
config __FWDTP, WDT_OFF
;.....
;SE ACTIVA EL POWER ON RESET (POR), BROWN OUT RESET (BOR), POWER UP TIMER (PWRT)
;Y EL MASTER CLEAR (MCLR)
;POR: AL MOMENTO DE ALIMENTAR EL DSPIC OCURRE UN RESET CUANDO EL VOLTAJE DE
;ALIMENTACIÓN ALCANZA UN VOLTAJE DE UMBRAL (VPOR), EL CUAL ES 1.85V
;BOR: ESTE MODULO GENERA UN RESET CUANDO EL VOLTAJE DE ALIMENTACIÓN DECAE
;POR DEBAJO DE UN CIERTO UMBRAL ESTABLECIDO (2.7V)
;PWRT: MANTIENE AL DSPIC EN RESET POR UN CIERTO TIEMPO ESTABLECIDO, ESTO AYUDA
;A ASEGURAR QUE EL VOLTAJE DE ALIMENTACIÓN SE HA ESTABILIZADO (16ms)
;.....
config __FBORPOR, PBOR_ON & BORV27 & PWRT_16 & MCLR_EN
;.....
;SE DESACTIVA EL CÓDIGO DE PROTECCIÓN
;.....
config __FGS, CODE_PROT_OFF & GWRP_OFF
;*****
; SECCIÓN DE DECLARACIÓN DE CONSTANTES CON LA DIRECTIVA .EQU (= DEFINE EN C)
;*****
.equ MUESTRAS, 64 ;NÚMERO DE MUESTRAS
;*****
; DECLARACIONES GLOBALES
;*****
;.....
;PROPORCIONA ALCANCE GLOBAL A LA FUNCIÓN _wreg_init, ESTO PERMITE LLAMAR A LA
;FUNCIÓN DESDE UN OTRO PROGRAMA EN ENSAMBLADOR O EN C COLOCANDO LA DECLARACIÓN
;"EXTERN"
;.....
.global _wreg_init
;.....
;ETIQUETA DE LA PRIMER LINEA DE CÓDIGO
;.....
.global __reset
;.....
;DECLARACIÓN DE LA ISR DEL TIMER 1 COMO GLOBAL
;.....
.global __T1Interrupt
;*****
;CONSTANTES ALMACENADAS EN EL ESPACIO DE LA MEMORIA DE PROGRAMA
;*****
.section .myconstbuffer, code
;.....
;ALINEA LA SIGUIENTE PALABRA ALMACENADA EN LA MEMORIA
```

```
;DE PROGRAMA A UNA DIRECCION MULTIPLO DE 2
/.....
    .palign 2

ps_coeff:
    .hword    0x0002, 0x0003, 0x0005, 0x000A
;short int ps_coeff[] = {0x0002, 0x0003, 0x0005, 0x000A };

;*****
;VARIABLES NO INICIALIZADAS EN EL ESPACIO X DE LA MEMORIA DE DATOS
;*****

    .section .xbss, bss, xmemory

x_input: .space 2*MUESTRAS      ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE
;char x_input[128];
;short int x_input[64];
;int x_input[32];
;*****
;VARIABLES NO INICIALIZADAS EN EL ESPACIO Y DE LA MEMORIA DE DATOS
;*****

    .section .ybss, bss, ymemory

y_input: .space 2*MUESTRAS      ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE
;*****
;VARIABLES NO INICIALIZADAS LA MEMORIA DE DATOS CERCANA (NEAR), LOCALIZADA
;EN LOS PRIMEROS 8KB DE RAM
;*****

    .section .nbss, bss, near

var1:    .space 2              ;LA VARIABLE VARI RESERVA 1 WORD DE ESPACIO
var2:    .space 2
var3:    .space 2
var4:    .space 2
var5:    .space 2
var6:    .space 2
var7:    .space 2
var8:    .space 2
var9:    .space 2
var0:    .space 2

;*****
;SECCION DE CODIGO EN LA MEMORIA DE PROGRAMA
;*****

.text      ;INICIO DE LA SECCION DE CODIGO

__reset:

    MOV __SP_init,    W15 ;INICIALIZA EL STACK POINTER

    MOV  __SPLIM_init, W0      ;INICIALIZA EL REGISTRO STACK POINTER LIMIT
    MOV  W0,          SPLIM

    NOP                      ;UN NOP DESPUES DE LA INICIALIZACION DE SPLIM

    CALL _WREG_INIT          ;SE LLAMA A LA RUTINA DE INICIALIZACION DE REGISTROS
                              ;OPCIONALMENTE USAR RCALL EN LUGAR DE CALL

    CALL  INI_PERIFERICOS

CICLO:
    MOV PORTF,    W1
    NOP
    LSR W1,    #2,    W1 ;W1 = W1 >> 2
    AND #0X0F,    W1      ;W1 = W1 & 0X000F
    MOV #0,W2
    mov W2, var0
    MOV #1,W2
    mov W2, var1
    MOV #2,W2
    mov W2, var2
```



```
MOV #3,W2
mov W2, var3
MOV #4,W2
mov W2, var4
MOV #5,W2
mov W2, var5
MOV #6,W2
mov W2, var6
MOV #7,W2
mov W2, var7
MOV #8,W2
mov W2, var8
MOV #9,W2
mov W2, var9
    MOV var0,W3
CPSNE W1, W3
GOTO DOS
NOP
    MOV var2,W3
CPSNE W1, W3
GOTO UNO
NOP
    MOV var7,W3
CPSNE W1, W3
GOTO CERO
NOP

MOV var1,W3
CPSNE W1, W3
GOTO CERO
NOP
    MOV var3,W3
CPSNE W1, W3
GOTO NUEVE
NOP
    MOV var6,W3
CPSNE W1, W3
GOTO CERO
NOP

MOV var4,W3
CPSNE W1, W3
GOTO TRES
NOP
    MOV var8,W3
CPSNE W1, W3
GOTO OCHO
NOP

MOV var5,W3
CPSNE W1, W3
GOTO CUATRO
NOP

MOV var9,W3
CPSNE W1, W3
GOTO TRES
NOP

MOV var9,W3
CPSTGT W1, W3
NOP
GOTO ERROR
NOP

;MOV #0, W2
;MOV #1, W3
;MOV #2, W4
```

```
;MOV #3, W5
;MOV #4, W6
;MOV #8, W7
;MOV #9, W8
;MOV #0x000A, W9

;CPSNE W1, W2
;GOTO CERO
;NOP
;CPSNE W1, W3
;GOTO UNO
;NOP
;CPSNE W1, W4
;GOTO DOS
;NOP
;CPSNE W1, W5
;GOTO TRES
;NOP
;CPSNE W1, W6
;GOTO CUATRO
;NOP
;CPSNE W1, W7
;GOTO OCHO
;NOP
;CPSNE W1, W8
;GOTO NUEVE
;NOP
;CPSGT W1, W9
;GOTO ERROR
        GOTO CICLO

CERO:
    MOV #0X007E,W0
    MOV W0, PORTB
    NOP
    GOTO CICLO

UNO:
    MOV #0X0030,W0
    MOV W0, PORTB
    NOP
    GOTO CICLO

DOS:
    MOV #0X006D,W0
    MOV W0, PORTB
    NOP
    GOTO CICLO

TRES:
    MOV #0X0079,W0
    MOV W0, PORTB
    NOP
    GOTO CICLO

CUATRO:
    MOV #0X0033,W0
    MOV W0, PORTB
    NOP
    GOTO CICLO

OCHO:
    MOV #0X007F,W0
    MOV W0, PORTB
    NOP
    GOTO CICLO

NUEVE:
    MOV #0X0073,W0
    MOV W0, PORTB
    NOP
    GOTO CICLO

ERROR:
    MOV #0X004F,W0
```

```
MOV    W0,    PORTB
NOP
GOTO CICLO

;/**@brief ESTA RUTINA INICIALIZA LOS PERIFERICOS DEL DSC
; */
INI_PERIFERICOS:
MOV    #0X0000,    W0
NOP
MOV    W0,    PORTF;PORTF = 0X0000
NOP
MOV    W0,    LATF;LATF = 0X0000
NOP
SETM   TRISF    ;TRISF = 0XFFFF
NOP
MOV    W0,    PORTB;PORTB = 0X0000
NOP
MOV    W0,    LATB;LATB = 0X0000
NOP
MOV    W0,    TRISB;TRISB = 0X0000
NOP
SETM   ADPCFG
RETURN

;/**@brief ESTA RUTINA INICIALIZA LOS REGISTROS Wn A 0X0000
; */
WREG_INIT:
CLR    W0
MOV    W0,    W14
REPEAT #12
MOV    W0,    [++W14]
CLR    W14
RETURN

;/**@brief ISR (INTERRUPT SERVICE ROUTINE) DEL TIMER 1
; * SE USA PUSH.S PARA GUARDAR LOS REGISTROS W0, W1, W2, W3,
; * C, Z, N Y DC EN LOS REGISTROS SOMBRA
; */
__T1Interrupt:
PUSH.S

BCLR   IFS0, #T1IF    ;SE LIMPIA LA BANDERA DE INTERRUPCION DEL TIMER 1

POP.S

RETFIE    ;REGRESO DE LA ISR

END    ;TERMINACION DEL CODIGO DE PROGRAMA EN ESTE ARCHIVO
```

3.4.4. Simulación

The simulation environment displays the following assembly code in the 'Asm Source' window:

```

129 CALL INI_PERIFERIOS
130
131
132 CICLO:
133     MOV PORTF, W1
134     NOP
135     LSR W1, #2, W1 ;W1 = W1 >> 2
136     AND #0X0F, W1 ;W1 = W1 & 0X000F
137
138     MOV #0,W2
139     MOV W2, var0
140     MOV #1,W2
141     MOV W2, var1
142     MOV #2,W2
143     MOV W2, var2

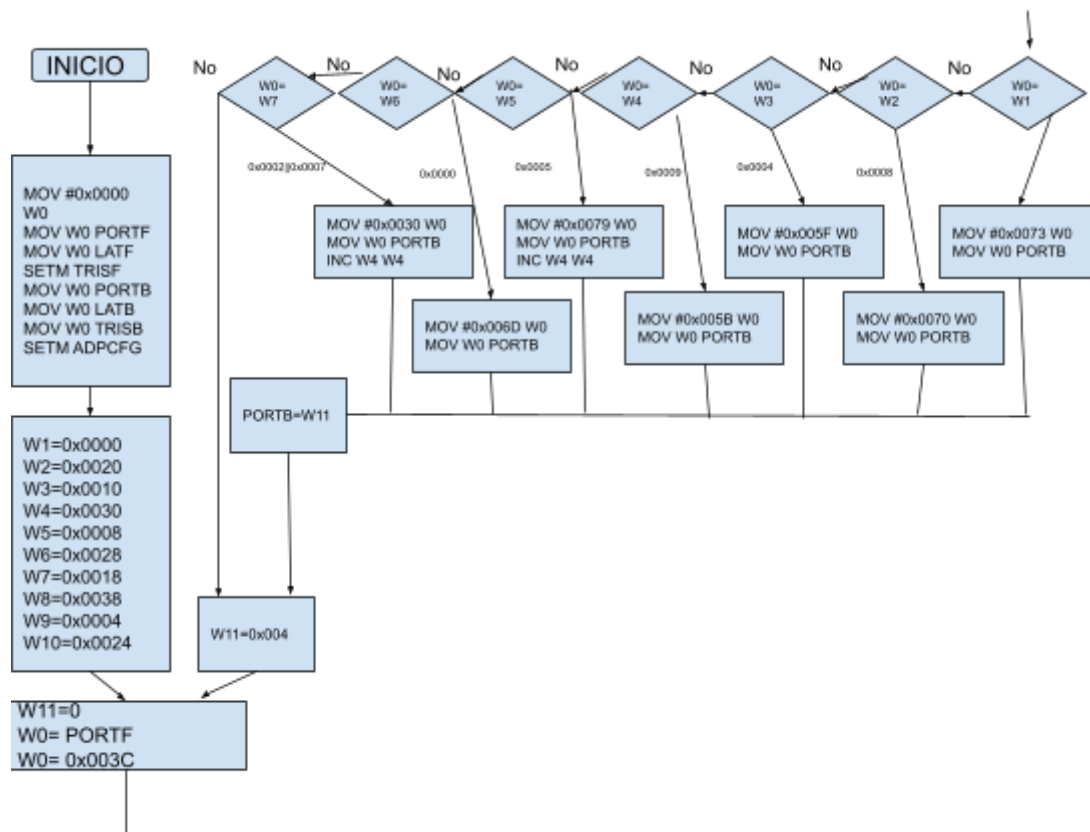
```

The 'Variables' window shows the state of the following variables:

| Name | Type | Address | Value |
|-------|------|---------|--------|
| PORTF | SFR | 0x2E0 | 0x003C |
| WREG0 | SFR | 0x0 | 0x0000 |
| WREG1 | SFR | 0x2 | 0x0000 |
| PORTB | SFR | 0x2C8 | 0x0000 |
| PORTD | SFR | 0x2D4 | 0x0000 |
| WREG2 | SFR | 0x4 | 0x0000 |

The four screenshots show the progression of the simulation, with the 'Variables' window updating the values of the registers and memory locations as the code executes.

3.5.2. Diagrama de Flujo (UML)



3.5.3. Código

```

/**@brief ESTE PROGRAMA MUESTRA LOS BLOQUES QUE FORMAN UN PROGRAMA
 * EN ENSAMBLADOR, LOS BLOQUES SON:
 * BLOQUE 1. OPCIONES DE CONFIGURACION DEL DSC: OSCILADOR, WATCHDOG,
 * BROWN OUT RESET, POWER ON RESET Y CODIGO DE PROTECCION
 * BLOQUE 2. EQUIVALENCIAS Y DECLARACIONES GLOBALES
 * BLOQUE 3. ESPACIOS DE MEMORIA: PROGRAMA, DATOS X, DATOS Y, DATOS NEAR
 * BLOQUE 4. CODIGO DE APLICACION
 * @device: DSPIC30F4013
 * @oscilLator: FRC, 7.3728MHz
 */

.equ __30F3013, 1
.include "p30F3013.inc"

;*****
; BITS DE CONFIGURACION
;*****
;.....
;SE DESACTIVA EL CLOCK SWITCHING Y EL FAIL-SAFE CLOCK MONITOR (FSCM) Y SE
;ACTIVA EL OSCILADOR INTERNO DE 7.3728MHZ (FAST RC) PARA TRABAJAR
;FSCM: PERMITE AL DISPOSITIVO CONTINUAR OPERANDO AUN CUANDO OCURRA UNA FALLA
;EN EL OSCILADOR. CUANDO OCURRE UNA FALLA EN EL OSCILADOR SE GENERA UNA TRAMPA
;Y SE CAMBIA EL RELOJ AL OSCILADOR FRC
;.....
config __FOSC, CSW_FSCM_OFF & FRC
;.....
;SE DESACTIVA EL WATCHDOG
;.....
config __FWDIT, WDT_OFF
;.....
;SE ACTIVA EL POWER ON RESET (POR), BROWN OUT RESET (BOR), POWER UP TIMER (PWRT)
;Y EL MASTER CLEAR (MCLR)
;POR: AL MOMENTO DE ALIMENTAR EL DSPIC OCURRE UN RESET CUANDO EL VOLTAJE DE
;ALIMENTACION ALCANZA UN VOLTAJE DE UMBRAL (VPOR), EL CUAL ES 1.85V

```

```
;BOR: ESTE MODULO GENERA UN RESET CUANDO EL VOLTAJE DE ALIMENTACION DECAE
;POR DEBAJO DE UN CIERTO UMBRAL ESTABLECIDO (2.7V)
;FWRT: MANTIENE AL DSPIC EN RESET POR UN CIERTO TIEMPO ESTABLECIDO, ESTO AYUDA
;A ASEGURAR QUE EL VOLTAJE DE ALIMENTACION SE HA ESTABILIZADO (16ms)
;.....
config __PBORPOR, PBOR_ON & BORV27 & FWRT_16 & MCLR_EN
;.....
;SE DESACTIVA EL CODIGO DE PROTECCION
;.....
config __FGS, CODE_PROT_OFF & GWRP_OFF

;*****
; SECCION DE DECLARACION DE CONSTANTES CON LA DIRECTIVA .EQU (= DEFINE EN C)
;*****
.equ MUESTRAS, 64           ;NUMERO DE MUESTRAS

;*****
; DECLARACIONES GLOBALES
;*****
;.....
;PROPORCIONA ALCANCE GLOBAL A LA FUNCION _wreg_init, ESTO PERMITE LLAMAR A LA
;FUNCION DESDE UN OTRO PROGRAMA EN ENSAMBLADOR O EN C COLOCANDO LA DECLARACION
;"EXTERN"
;.....
.global _wreg_init
;.....
;ETIQUETA DE LA PRIMER LINEA DE CODIGO
;.....
.global __reset
;.....
;DECLARACION DE LA ISR DEL TIMER 1 COMO GLOBAL
;.....
.global __T1Interrupt

;*****
;CONSTANTES ALMACENADAS EN EL ESPACIO DE LA MEMORIA DE PROGRAMA
;*****
.section .myconstbuffer, code
;.....
;ALINEA LA SIGUIENTE PALABRA ALMACENADA EN LA MEMORIA
;DE PROGRAMA A UNA DIRECCION MULTIPLO DE 2
;.....
.palign 2

ps_coeff:
.hword 0x0002, 0x0003, 0x0005, 0x000A
;short int ps_coeff[] = {0x0002, 0x0003, 0x0005, 0x000A };

;*****
;VARIABLES NO INICIALIZADAS EN EL ESPACIO X DE LA MEMORIA DE DATOS
;*****
.section .xbss, bss, xmemory

x_input: .space 2*MUESTRAS      ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE
;char x_input[128];
;short int x_input[64];
;int x_input[32];
;*****
;VARIABLES NO INICIALIZADAS EN EL ESPACIO Y DE LA MEMORIA DE DATOS
;*****
.section .ybss, bss, ymemory

y_input: .space 2*MUESTRAS      ;RESERVANDO ESPACIO (EN BYTES) A LA VARIABLE
;*****
;VARIABLES NO INICIALIZADAS LA MEMORIA DE DATOS CERCANA (NEAR), LOCALIZADA
;EN LOS PRIMEROS 8KB DE RAM
;*****
```

```
.section .nbss, bss, near

var1:    .space 2           ;LA VARIABLE VAR1 RESERVA 1 WORD DE ESPACIO
var2:    .space 2
var3:    .space 2
var4:    .space 2
var5:    .space 2
var6:    .space 2
var7:    .space 2
var8:    .space 2
var9:    .space 2
var0:    .space 2

;*****
;SECCION DE CODIGO EN LA MEMORIA DE PROGRAMA
;*****
.text    ;INICIO DE LA SECCION DE CODIGO

__reset:

    MOV #__SP_init,    W15 ;INICIALIZA EL STACK POINTER

    MOV  __SPLIM_init, W0   ;INICIALIZA EL REGISTRO STACK POINTER LIMIT
    MOV  W0,           SPLIM

    NOP                    ;UN NOP DESPUES DE LA INICIALIZACION DE SPLIM

    CALL __WREG_INIT       ;SE LLAMA A LA RUTINA DE INICIALIZACION DE REGISTROS
                                ;OPCIONALMENTE USAR RCALL EN LUGAR DE CALL

    CALL  INI_PERIFERICOS

CICLO:
    PUSH  W0
    MOV   #1000, W0
    CALL  RETRASOBOLETA
    POP   W0
    MOV   #0,W2
    mov W2, var0
    MOV   #1,W2
    mov W2, var1
    MOV   #2,W2
    mov W2, var2
    MOV   #3,W2
    mov W2, var3
    MOV   #4,W2
    mov W2, var4
    MOV   #5,W2
    mov W2, var5
    MOV   #6,W2
    mov W2, var6
    MOV   #7,W2
    mov W2, var7
    MOV   #8,W2
    mov W2, var8
    MOV   #9,W2
    mov W2, var9
    MOV   var0,W3
    CPSNE W1, W3
    GOTO  DOS
    NOP
    MOV   var2,W3
    CPSNE W1, W3
    GOTO  UNO
    NOP
    MOV   var7,W3
    CPSNE W1, W3
    GOTO  CERO
    NOP
    MOV   var1,W3
```



```
CPSNE W1, W3
GOTO CERO
NOP
MOV var3,W3
CPSNE W1, W3
GOTO NUEVE
NOP
MOV var6,W3
CPSNE W1, W3
GOTO CERO
NOP

MOV var4,W3
CPSNE W1, W3
GOTO TRES
NOP
MOV var8,W3
CPSNE W1, W3
GOTO OCHO
NOP

MOV var5,W3
CPSNE W1, W3
GOTO CUATRO
NOP

MOV var9,W3
CPSNE W1, W3
GOTO TRES
NOP

MOV var9,W3
CPSGT W1, W3
NOP
MOV #0, W4
NOP

GOTO CICLO

CERO:
MOV #0X007E,W0
MOV W0, PORTB
INC W4,W4
NOP
GOTO CICLO

UNO:
MOV #0X0030,W0
MOV W0, PORTB
INC W4,W4
NOP
GOTO CICLO

DOS:
MOV #0X006D,W0
MOV W0, PORTB
INC W4,W4
NOP
GOTO CICLO

TRES:
MOV #0X0079,W0
MOV W0, PORTB
INC W4,W4
NOP
GOTO CICLO

CUATRO:
MOV #0X0033,W0
MOV W0, PORTB
INC W4,W4
```

```
    NOP
    GOTO CICLO
OCHO:
    MOV    #0X007F,W0
    MOV    W0,    PORTB
    INC    W4,W4
    NOP
    GOTO CICLO
NUEVE:
    MOV    #0X0073,W0
    MOV    W0,    PORTB
    INC    W4,W4
    NOP
    GOTO CICLO

    MOV    #614,   W5
    MOV    #10,    W6
RETRASOBOLETA:
    PUSH    W1
CICLO2_1MS:
    MOV    #614,   W1
CICLO1_1MS:
    DEC    W1, W1
    BRA    NZ, CICLO1_1MS
    DEC    W0, W0
    BRA    NZ, CICLO2_1MS
    POP    W1
    RETURN

;/**@brief ESTA RUTINA INICIALIZA LOS PERIFERICOS DEL DSC
; */
INI_PERIFERICOS:
    MOV    #0X0000,    W0
    MOV    #0X0000,    W4
    NOP
    MOV    W0,    PORTF;PORTF = 0X0000
    NOP
    MOV    W0,    LATF;LATF = 0X0000
    NOP
    SETM    TRISF    ;TRISF = 0XFFFF
    NOP
    MOV    W0,    PORTB;PORTE = 0X0000
    NOP
    MOV    W0,    LATB;LATB = 0X0000
    NOP
    MOV    W0,    TRISB;TRISB = 0X0000
    NOP
    SETM    ADPCFG
    RETURN

;/**@brief ESTA RUTINA INICIALIZA LOS REGISTROS Wn A 0X0000
; */
_WREG_INIT:
    CLR    W0
    MOV    W0,    W14
    REPEAT    #12
    MOV    W0,    [++W14]
    CLR    W14
    RETURN

;/**@brief ISR (INTERRUPT SERVICE ROUTINE) DEL TIMER 1
; * SE USA PUSH.S PARA GUARDAR LOS REGISTROS W0, W1, W2, W3,
; * C, Z, N Y DC EN LOS REGISTROS SOMBRA
; */
_T1Interrupt:
    PUSH.S
```

```

BCLR IFS0, #T1IF          ;SE LIMPIA LA BANDERA DE INTERRUPCION DEL TIMER 1

POP.S

RETFIE                    ;REGRESO DE LA ISR

END                        ;TERMINACION DEL CODIGO DE PROGRAMA EN ESTE ARCHIVO

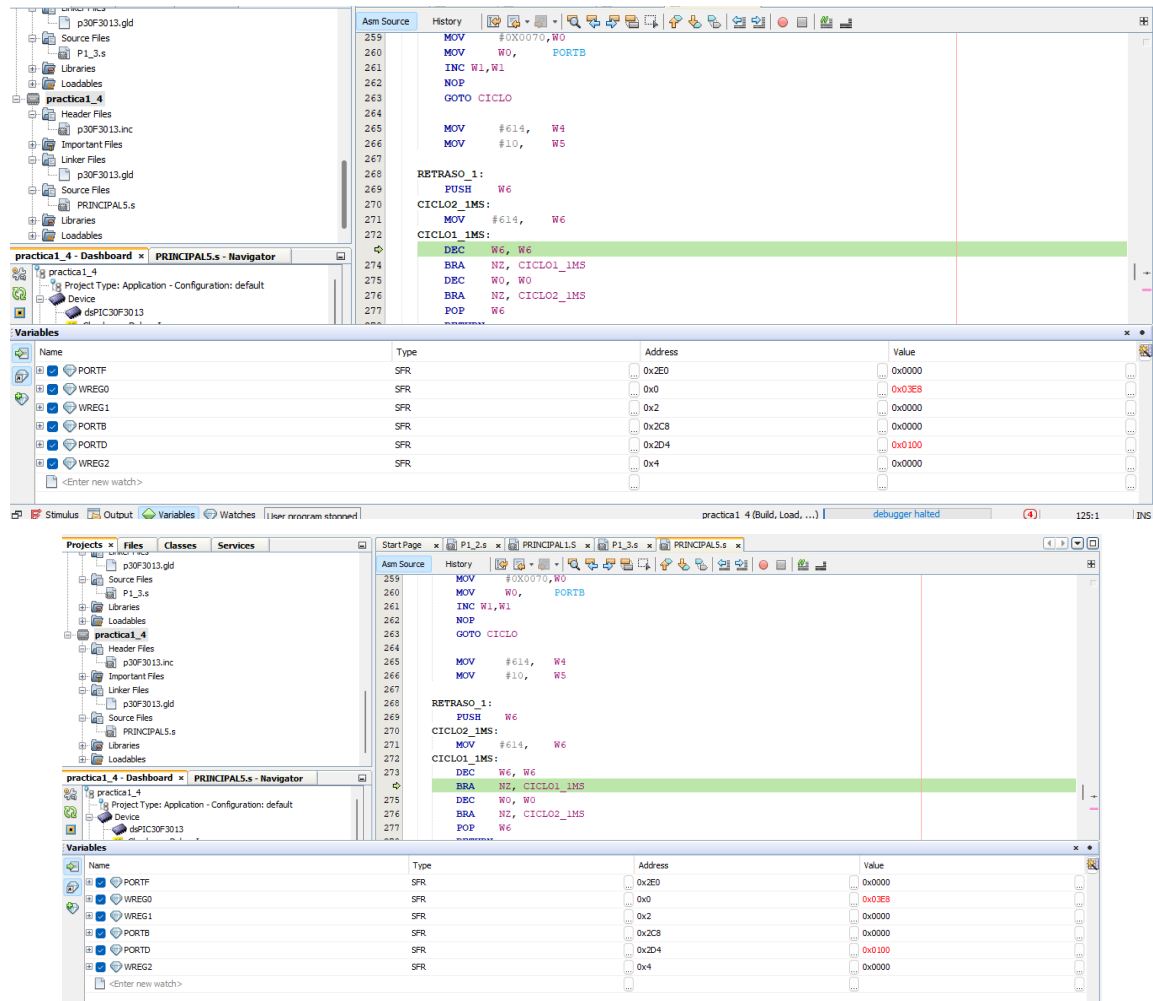
```

3.5.4. Simulación

The simulation environment shows the following details:

- Pin Register Actions:** A table showing actions for various pins. Pin 5 is highlighted with the action 'SetLow'.
- Assembly Source:** The main program code is displayed, including:
 - Initialization of PORTF, LATF, and TRISF.
 - A loop structure with a delay (RETRASO_1) and a counter (CICLO1_1MS).
 - Assembly instructions like MOV, NOP, SETM, and RETM.
- Variables:** A table showing the current values of registers:

| Name | Type | Address | Value |
|-------|------|---------|--------|
| PORTF | SFR | 0x2E0 | 0x0000 |
| WREG0 | SFR | 0x0 | 0x03E8 |
| WREG1 | SFR | 0x2 | 0x0000 |
| PORTB | SFR | 0x2C8 | 0x0000 |
| PORTD | SFR | 0x2D4 | 0x0100 |
| WREG2 | SFR | 0x4 | 0x0000 |



4. Hoja de Firmas



Instituto Politécnico Nacional
Escuela Superior de Cómputo



| | |
|--------------|--|
| Profesor: | Adbel Anahí Montes Meza |
| Grupo: | 3CM13 |
| Equipo: | 10 |
| Integrantes: | Montaño Reyes Jennifer Pardo Alanis Arturo Isaac Ortega Alcocer Humberto Alejandro |

| Ejercicio | Fecha de entrega | Funcionó | Firma |
|-----------------------------------|------------------|----------|--------|
| 1. Lectura/escritura puerto | 14/Nov/22 | ✓ | Montes |
| 2. Calculadora | 18/Nov/22 | ✓ | Montes |
| 3. Boleta en display con switch | 16/Nov/22 | ✓ | Montes |
| 4. Boleta en display con contador | 23/Nov/22 | ✓ | Montes |

5. Conclusiones

5.1. Montaña Reyes Jennifer

Las prácticas ayudaron a comprender las diferentes capacidades que presenta un microcontrolador, averiguamos el uso para su programación y su construcción de manera física. Comprendimos temas con los que no habíamos tenido acercamiento, de modo que fue de gran importancia para las siguientes prácticas que se avecinan.

5.2. Pardo Alanis Arturo Isaac

Con la realización de estos ejercicios de esta primera práctica, sirvió como un buen ejemplo para dar como primer acercamiento al uso de los circuitos de microcontroladores y de MPLAB, ya que estos ejercicios fueron diversos por lo que hacían cada uno de estos ejercicios. También gracias a esta práctica se pudo hacer uso de los conocimientos que hemos visto desde el comienzo de la materia y ha sido de gran utilidad para reafirmar y comprender de mejor manera lo visto en clase.

5.3. Ortega Alcocer Humberto Alejandro

En esta práctica fue donde realmente comencé a ver el uso de un microcontrolador físico. Hasta ahora, y gracias en gran medida a la pandemia, toda mi interacción durante la carrera con circuitos y electrónica en general fue mediante simuladores. El poder trabajar físicamente con componentes, circuitos y sus debidas consideraciones (y correcciones) me ayudó a asentar de mejor forma los conocimientos vistos en clase y a tener muchos más cuidado en el diseño de mis circuitos pues, como pude ver en el desarrollo de esta práctica, requiere de mucho cuidado para no dañar los componentes.