Práctica 2
Programación de Microcontroladores PIC II

- Características del PIC 16F877A
  - Organización de la memoria
  - Puertos de E/S
- Transferencia del programa al PIC
- Ejemplos
- Tareas a realizar

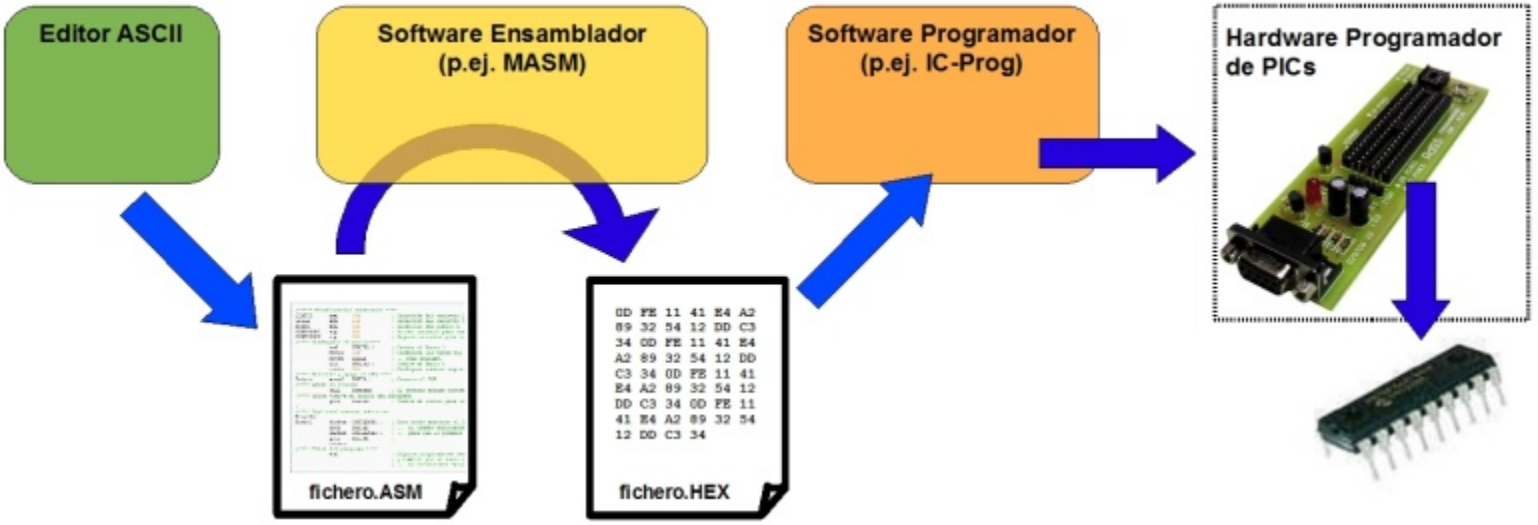
Introducción

Objetivos

- Conocer las principales características del PIC 16F877A.
- Ser capaz de programar código para el PIC16F877A haciendo uso de sus recursos: registros, puertos, etc.
- Simular los programas desarrollados en el entorno MPLAB
- Transferir los programas una vez simulados para su ejecución directa en el PIC 16F877A.

Entorno de trabajo

- La **metodología** que propusimos en la práctica anterior incluía:



- Ahora nos falta transferir el programa desarrollado en MPLAB para que se ejecute en nuestro PIC 16F877A.

Organización de la memoria

- **MEMORIA DE DATOS** dividida en dos áreas:
  - **Registros de funciones especiales** (**SFR**, Special Function Registers)
    - Controlan la operación de la CPU y los periféricos.
    - Se implementan como RAM estática
    - Se inicializan a un valor por defecto después de la alimentación del microcontrolador.

  - **Registros de propósito general** (**GPR**, General Purpose Registers)
    - Almacenamiento de datos.
    - No se inicializan a un valor por defecto después de la alimentación.
  - La transferencia entre registros ha de hacerse a través del registro **W**.
  - Estructura en bancos de 128 bits accesibles mediante STATUS<7:5>
  - Acceso directo (bits RP1:RP0) o indirecto (bit IRP y registro FSR) a la información.

Organización de la memoria

- Distribución de las áreas de registros SFR Y GPR en cada banco de memoria del PIC16F876A/877A

Bank 0	File Address	Bank 1	File Address	Bank 2	File Address	Bank 3	File Address
Indirect addr.(*)	00h	Indirect addr.(*)	80h	Indirect addr.(*)	100h	Indirect addr.(*)	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD(1)	08h	TRISD(1)	88h		108h		188h
PORTE(1)	09h	TRISE(1)	89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved(2)	18Eh
TMR1H	0Fh		8Fh	EEADRH	10Fh	Reserved(2)	18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h	General Purpose Register 16 Bytes	117h	General Purpose Register 16 Bytes	197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch	CMCON	9Ch		11Ch		19Ch
CCP2CON	1Dh	CVRCON	9Dh		11Dh		19Dh
ADRESH	1Eh	ADRESL	9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
General Purpose Register 96 Bytes	20h	General Purpose Register 80 Bytes	A0h	General Purpose Register 80 Bytes	120h	General Purpose Register 80 Bytes	1A0h
			EFh		16Fh		1EFh
accesses 70h-7Fh	7Fh	accesses 70h-7Fh	F0h FFh	accesses 70h-7Fh	170h 17Fh	accesses 70h - 7Fh	1F0h 1FFh

Posiciones de memoria sin implementar. Se leerá un 0.

(*) No es un registro real.

1. Estos registros no están implementados en el PIC16F876A.
2. Estos registros están reservados. Deben mantenerse sin usar.

Organización de la memoria

- Detalle de los registros SPR del PIC16F87XA

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 0											
00h(3)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	31, 150
01h	TMR0	Timer0 Module Register								xxxx xxxx	55, 150
02h(3)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	30, 150
03h(3)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	22, 150
04h(3)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	31, 150
05h	PORTA	—	—	PORTA Data Latch when written: PORTA pins when read						--0x 0000	43, 150
06h	PORTB	PORTB Data Latch when written: PORTB pins when read								xxxx xxxx	45, 150
07h	PORTC	PORTC Data Latch when written: PORTC pins when read								xxxx xxxx	47, 150
08h(4)	PORTD	PORTD Data Latch when written: PORTD pins when read								xxxx xxxx	48, 150
09h(4)	PORTE	—	—	—	—	—	RE2	RE1	RE0	---- -xxx	49, 150
0Ah(1,3)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	30, 150
0Bh(3)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	24, 150
0Ch	PIR1	PSPIF(3)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	26, 150
0Dh	PIR2	—	CMIF	—	EEIF	BCLIF	—	—	CCP2IF	-0-0 0--0	28, 150
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	60, 150
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	60, 150
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	--00 0000	57, 150
11h	TMR2	Timer2 Module Register								0000 0000	62, 150
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	61, 150

Organización de la memoria

- Detalle de los registros SPR del PIC16F87XA (continuación)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 0											
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	79, 150
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	82, 82, 150
15h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	63, 150
16h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	63, 150
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	64, 150
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	112, 150
19h	TXREG	USART Transmit Data Register								0000 0000	118, 150
1Ah	RCREG	USART Receive Data Register								0000 0000	118, 150
1Bh	CCPR2L	Capture/Compare/PWM Register 2 (LSB)								xxxx xxxx	63, 150
1Ch	CCPR2H	Capture/Compare/PWM Register 2 (MSB)								xxxx xxxx	63, 150
1Dh	CCP2CON	—	—	CCP2X	CCP2Y	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	64, 150
1Eh	ADRESH	A/D Result Register High Byte								xxxx xxxx	133, 150
1Fh	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON	0000 00-0	127, 150

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8>, whose contents are transferred to the upper byte of the program counter.

2: Bits PSPIE and PSPIF are reserved on PIC16F873A/876A devices; always maintain these bits clear.

3: These registers can be addressed from any bank.

4: PORTD, PORTE, TRISD and TRISE are not implemented on PIC16F873A/876A devices, read as '0'.

5: Bit 4 of EEADRH implemented only on the PIC16F876A/877A devices.

Organización de la memoria

- Detalle de los registros SPR del PIC16F87XA (continuación)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 1											
80h(3)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	31, 150
81h	OPTION_REG	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	23, 150
82h(3)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	30, 150
83h(3)	STATUS	IRP	RP1	RP0	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	0001 1xxx	22, 150
84h(3)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	31, 150
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	43, 150
86h	TRISB	PORTB Data Direction Register								1111 1111	45, 150
87h	TRISC	PORTC Data Direction Register								1111 1111	47, 150
88h(4)	TRISD	PORTD Data Direction Register								1111 1111	48, 151
89h(4)	TRISE	IBF	OBF	IBOV	PSPMODE	—	PORTE Data Direction bits			0000 -111	50, 151
8Ah(1,3)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	30, 150
8Bh(3)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	24, 150
8Ch	PIE1	PSPIE(2)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	25, 151
8Dh	PIE2	—	CMIE	—	EEIE	BCLIE	—	—	CCP2IE	-0-0 0--0	27, 151
8Eh	PCON	—	—	—	—	—	—	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	---- --qq	29, 151
8Fh	—	Unimplemented								—	—
90h	—	Unimplemented								—	—

Organización de la memoria

- Detalle de los registros SPR del PIC16F87XA (continuación)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 1											
91h	SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	83, 151
92h	PR2	Timer2 Period Register								1111 1111	62, 151
93h	SSPADD	Synchronous Serial Port (I^2C mode) Address Register								0000 0000	79, 151
94h	SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	79, 151
95h	—	Unimplemented								—	—
96h	—	Unimplemented								—	—
97h	—	Unimplemented								—	—
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	111, 151
99h	SPBRG	Baud Rate Generator Register								0000 0000	113, 151
9Ah	—	Unimplemented								—	—
9Bh	—	Unimplemented								—	—
9Ch	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	135, 151
9Dh	CVRCON	CVREN	CVROE	CVRR	—	CVR3	CVR2	CVR1	CVR0	000- 0000	141, 151
9Eh	ADRESL	A/D Result Register Low Byte								xxxx xxxx	133, 151
9Fh	ADCON1	ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0	00-- 0000	128, 151

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8>, whose contents are transferred to the upper byte of the program counter.

2: Bits PSPIE and PSPIF are reserved on PIC16F873A/876A devices; always maintain these bits clear.

3: These registers can be addressed from any bank.

4: PORTD, PORTE, TRISD and TRISE are not implemented on PIC16F873A/876A devices, read as '0'.

5: Bit 4 of EEADRH implemented only on the PIC16F876A/877A devices.

Puertos de E/S

- Se pueden considerar los periféricos más sencillos
- Implementan las entrada-salida de la MCU.
- Se utilizan mediante dos registros: PORTx (datos) y TRISx (control)
  - Pueden implementarse hasta siete puertos de características distintas (x puede sustituirse con identificadores de puerto desde la A hasta la G)
  - Cada uno de los bits de TRISx establece la dirección de la información de su correspondiente bit (pin) en PORTx.
  - Un 1 configura el pin como entrada.
  - Una operación de lectura obtiene el nivel presente en el terminal implicado.
  - Un 0 configura ese pin como salida
    - Manteniendo el bit de salida mediante un latch.
  - Después de un reset todos los bits de TRISx son 1.
- Los pines de entrada/salida pueden estar multiplexados con varios periféricos.
- Para conocer con exactitud las características de cada puerto en concreto es imprescindible consultar las hojas de características de cada dispositivo

Puertos de E/S

Puerto A

- Puerto bidireccional de 6 bits.
- RA4 Tiene entrada Trigger Schmitt y salida drenador abierto, el resto admiten niveles de entrada TTL y salida CMOS.
- Debe configurarse si se quiere que funcione de forma **analógica** o **digital**.
- Su correspondiente registro de dirección es TRISA
  - Poniendo a uno un bit del registro TRISA se establece como entrada el correspondiente pin de PORTA..
  - Poniendo a cero un bit de TRISA se establece como salida el correspondiente pin de PORTA.

Ejemplo de inicialización del puerto A:

```
BCF     STATUS, RP0   ;
BCF     STATUS, RP1   ; Bank0
CLRF    PORTA         ; Initialize PORTA by clearing output data latches
BSF     STATUS, RP0   ; Select Bank 1
MOVLW   0x06          ; Configure all pins
MOVWF   ADCON1        ; as digital inputs
MOVLW   0xCF          ; Value used to initialize data direction
MOVWF   TRISA         ; Set RA<3:0> as inputs RA<5:4> as outputs
                      ; TRISA<7:6>are always read as '0'.
```

Puertos de E/S

- **Puerto A:** analógico o digital

ADCON1 REGISTER (ADDRESS 9Fh)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit

1 = Right justified. Six (6) Most Significant bits of ADRESH are read as '0'.
0 = Left justified. Six (6) Least Significant bits of ADRESL are read as '0'.

bit 6 **ADCS2:** A/D Conversion Clock Select bit (ADCON1 bits in shaded area and in **bold**)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	FOSC/2
0	01	FOSC/8
0	10	FOSC/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	FOSC/4
1	01	FOSC/16
1	10	FOSC/64
1	11	FRC (clock derived from the internal A/D RC oscillator)

bit 5-4 **Unimplemented:** Read as '0'

Puertos de E/S

❑ **Puerto A:** analógico o digital

Configuración del puerto A como líneas digitales

```
MOVLW   0x06
MOVWF   ADCON1
```

ADCON1 REGISTER (ADDRESS 9Fh)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits

PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C/R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	AN3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	AN3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	AN3	VSS	2/1
011x	D	D	D	D	D	D	D	D	—	—	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	AN3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1/2

A = Analog input D = Digital I/O
C/R = # of analog input channels/# of A/D voltage references

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Note: On any device Reset, the port pins that are multiplexed with analog functions (ANx) are forced to be an analog input.

Puertos de E/S

Puerto A

- Funciones

Name	Bit#	Buffer	Function
RA0/AN0	bit 0	TTL	Input/output or analog input.
RA1/AN1	bit 1	TTL	Input/output or analog input.
RA2/AN2/VREF-/CVREF	bit 2	TTL	Input/output or analog input or VREF- or CVREF.
RA3/AN3/VREF+	bit 3	TTL	Input/output or analog input or VREF+.
RA4/T0CKI/C1OUT	bit 4	ST	Input/output or external clock input for Timer0 or comparator output. Output is open-drain type.
RA5/AN4/$\overline{SS}$/C2OUT	bit 5	TTL	Input/output or analog input or slave select input for synchronous serial port or comparator output.

Legend: TTL = TTL input, ST = Schmitt Trigger input

- Resumen de los registros asociados con el puerto A

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
05h	PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	--0x 0000	--0u 0000
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	--11 1111
9Ch	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	0000 0111
9Dh	CVRCON	CVREN	CVROE	CVRR	—	CVR3	CVR2	CVR1	CVR0	000- 0000	000- 0000
9Fh	ADCON1	ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0	00-- 0000	00-- 0000

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTA.

Puertos de E/S

- **Puerto B**
  - Puerto de 8 bits bidireccionales
  - Su correspondiente registro de dirección es TRISB
    - Poniendo a uno un bit del registro TRISB se establece como entrada el correspondiente pin de PORTB..
    - Poniendo a cero un bit de TRISB se establece como salida el correspondiente pin de PORTB
- Tres pines del PORTB están multiplexados con *In-Circuit Debugger* y la función *Low-Voltage Programming*: RB3/PGM, RB6/PGC y RB7/PGD
- Cuatro de los pines de PORTB, RB7:RB4, tienen la capacidad de provocar una interrupción cuando están configurados como entradas.

Puertos de E/S

- La escritura en un puerto implica una lectura-modificación-escritura.
- Puede acarrear problemas en operaciones de escritura sobre puertos en los que unos pines están configurados como entradas y otros como salidas:

```
; Configuración del puerto B: PORTB<7:4> entradas
; PORTB<3:0> Salidas
; PORTB<7:6> tienen pull-ups externos y no están conectados a otro circuito
;
                    ; PORT latch PORT pins
                    ; ---------- ---------
BCF PORTB, 7        ; 01pp pppp  11pp pppp
BCF PORTB, 6        ; 10pp pppp  11pp pppp
BSF STATUS, RP0 ;
BCF TRISB, 7        ; 10pp pppp  11pp pppp
BCF TRISB, 6        ; 10pp pppp  10pp pppp
```

Puertos de E/S

Puerto B

- Funciones

Name	Bit#	Buffer	Function
RB0/INT	bit 0	TTL/ST(1)	Input/output pin or external interrupt input. Internal software programmable weak pull-up.
RB1	bit 1	TTL	Input/output pin. Internal software programmable weak pull-up.
RB2	bit 2	TTL	Input/output pin. Internal software programmable weak pull-up.
RB3/PGM(3)	bit 3	TTL	Input/output pin or programming pin in LVP mode. Internal software programmable weak pull-up.
RB4	bit 4	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB5	bit 5	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB6/PGC	bit 6	TTL/ST(2)	Input/output pin (with interrupt-on-change) or in-circuit debugger pin. Internal software programmable weak pull-up. Serial programming clock.
RB7/PGD	bit 7	TTL/ST(2)	Input/output pin (with interrupt-on-change) or in-circuit debugger pin. Internal software programmable weak pull-up. Serial programming data.

Legend: TTL = TTL input, ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.

2: This buffer is a Schmitt Trigger input when used in Serial Programming mode or in-circuit debugger.

3: Low-Voltage ICSP Programming (LVP) is enabled by default which disables the RB3 I/O function. LVP must be disabled to enable RB3 as an I/O pin and allow maximum compatibility to the other 28-pin and 40-pin mid-range devices.

Puertos de E/S

Puerto B

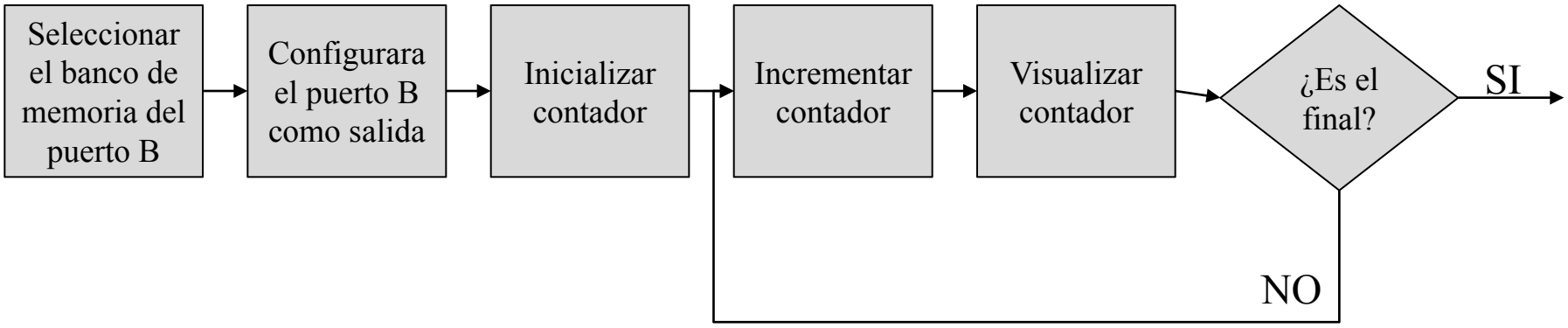
- Resumen de los registros asociados con el puerto B

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
06h, 106h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu
86h, 186h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
81h, 181h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTB.

Ejemplo

- Nos planteamos el siguiente problema:
  - Crearemos un programa para un PIC16F77A funcionando a 8MHZ encargado de contar hasta 0x5f. Cuando lo alcance se detendrá en un bucle no operativo. El valor del contador se visualizará en 8 diodos LED conectados al puerto B.

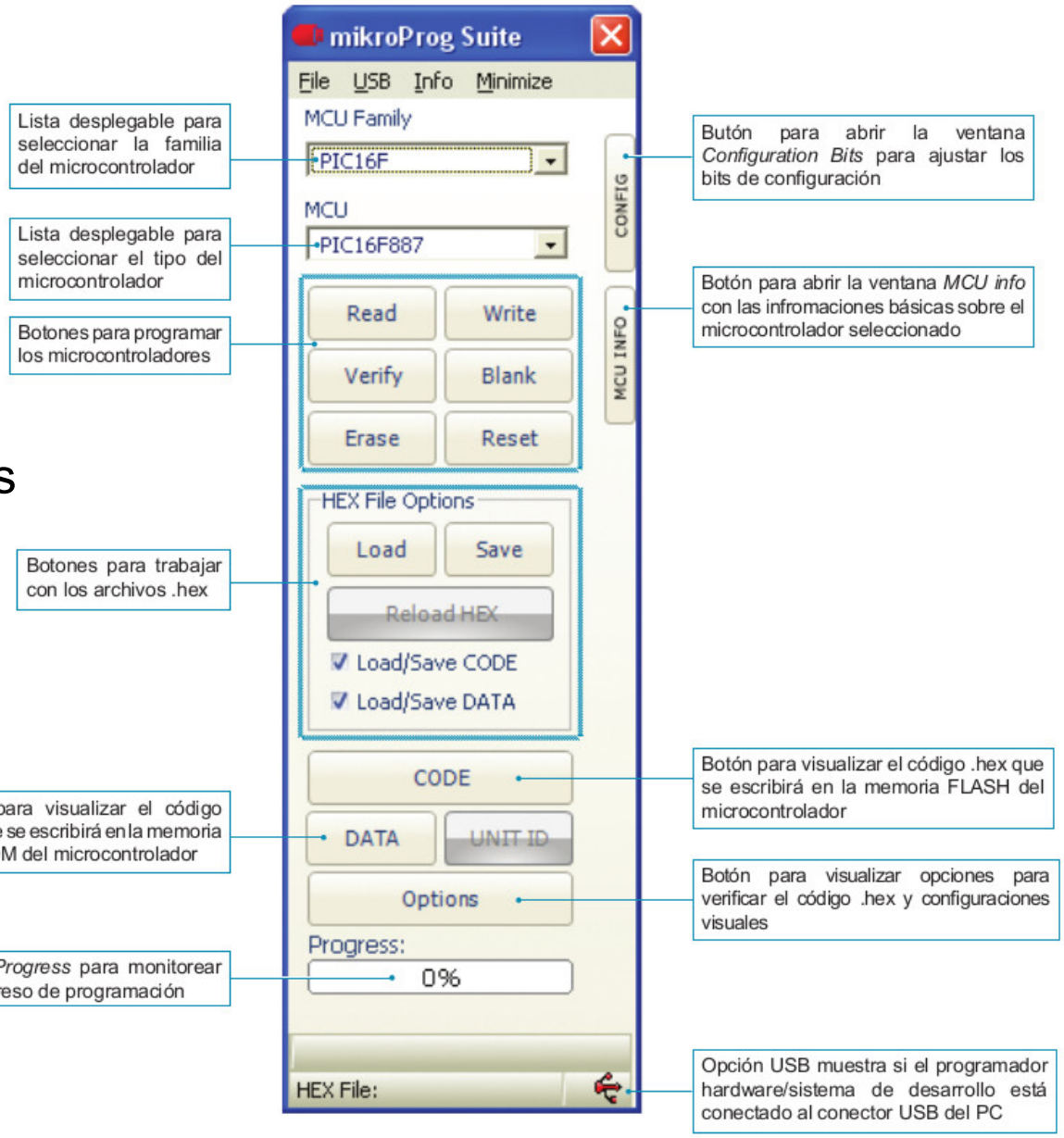


Ejemplo

```
LIST    P=16F877A       ; Seleccionamos el micro

; Asignación de etiquetas a registros.
f       EQU   0x01      ; registro f
portb   EQU   0x06      ; Dirección del registro del puerto B
estado  EQU   0x03      ; Dirección del registro de estado
conta   EQU   0x0C      ; Lo usamos como variable contadora

        ORG   0         ; El programa comienza en la dirección 0
        GOTO  Inicio    ; salta a la dirección 5 para sobrepasar el vector INT.
        ORG   5

Inicio  BSF   estado,5  ; Selecciona banco 1 para llegar a TRISB
        MOVLW 0x00
        MOVWF portb     ; Y se especifica que es de salida
        BCF   estado,5  ; Selección del banco 0 para trabajar con el puerto
        CLRF  conta     ; Ponemos nuestro contador a 0
bucle1  INCF  conta,f   ; conta + 1 --> conta (incrementa el contador)
        MOVF  conta,W   ; conta se carga en W
        MOVWF portb     ; W se carga en el registro de datos del puerto B
        MOVLW 0x5f      ; W <-- 0x5f        (Final de cuenta deseado)
        SUBWF conta,W   ; conta - W --> W. Si es cero, la cuenta está acabada
        BTFSS estado,2  ; Explora Z y si vale 1 es que W vale 0
                        ; se produce salto en ese caso por fin de cuenta
        GOTO bucle1     ; Si Z = 0 se vuelve a bucle1
bucle2  GOTO bucle2     ; Si Z = 1 se produce un bucle infinito
END
```

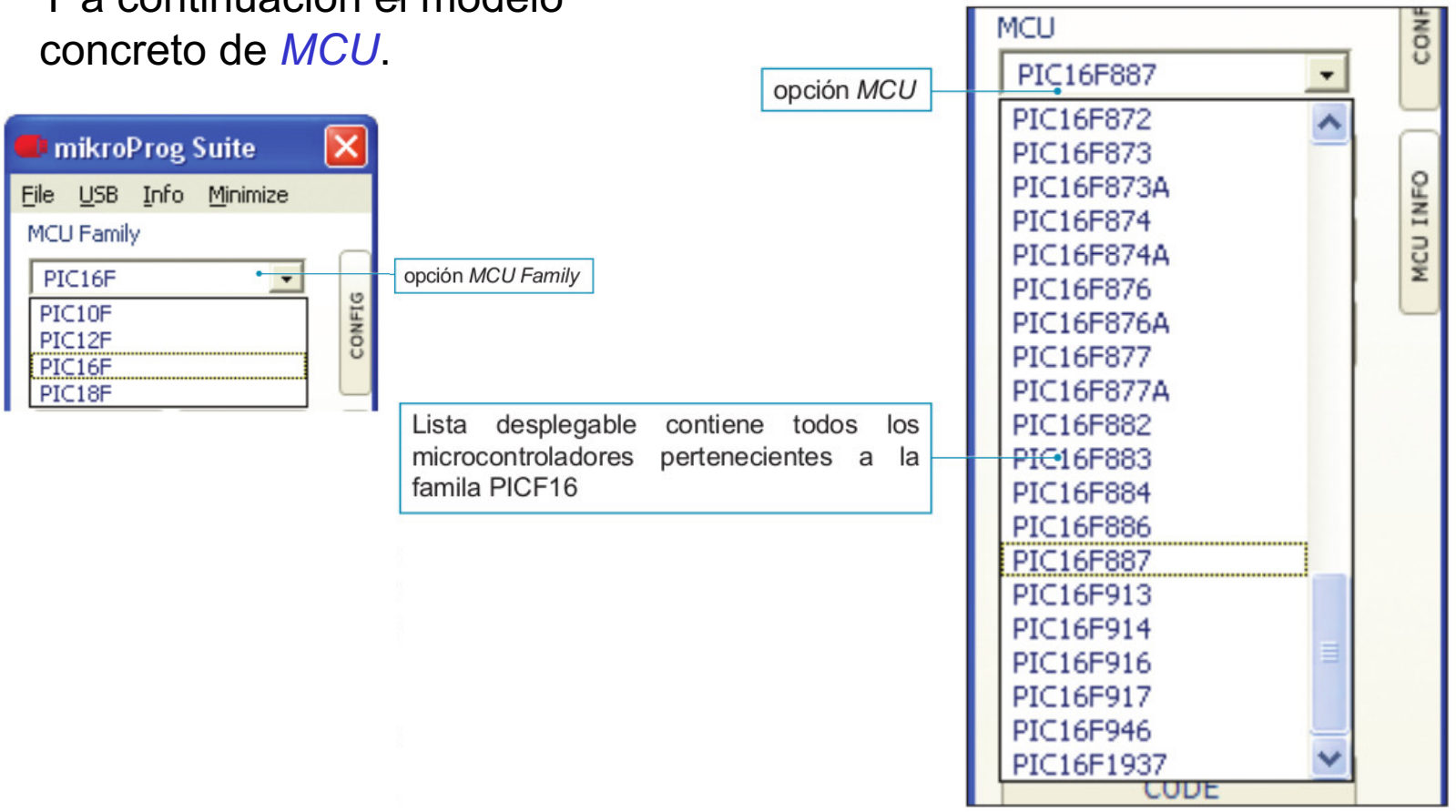
Transferencia del programa al PIC

- El **programa** mikroProg Suite nos permite programar el microcontrolador.
- La ventana principal incluye las opciones básicas para programar los microcontroladores, junto con dos opciones de programación avanzadas que permiten ajustar los bits de configuración.



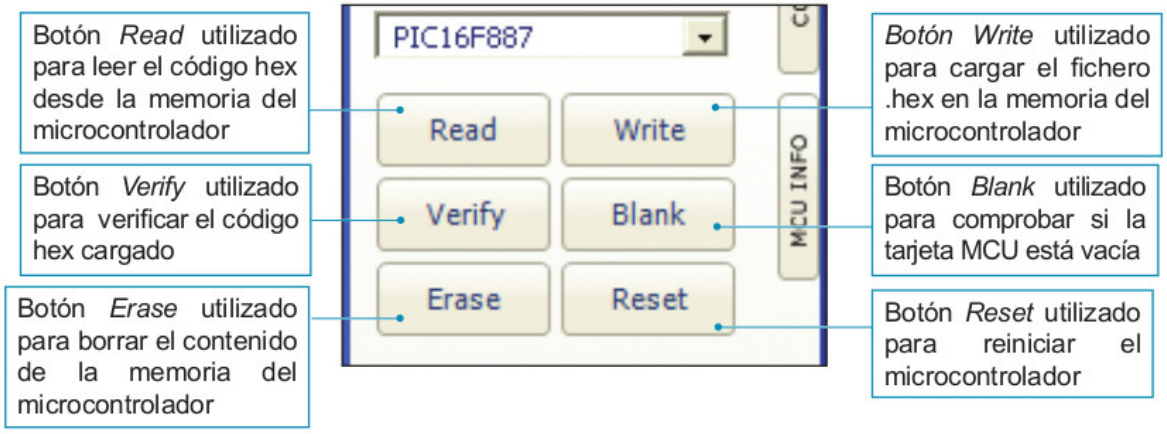
Transferencia del programa al PIC

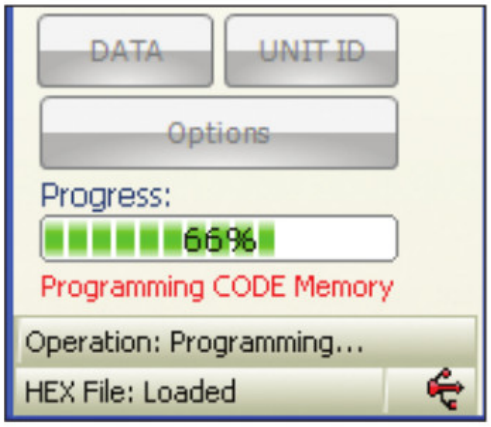
- Lo primero que hay que hacer es seleccionar la familia y el tipo del programador que será programado.
  - Para ello se elige la opción *MCU Family*
  - Y a continuación el modelo concreto de *MCU*.



Transferencia del programa al PIC

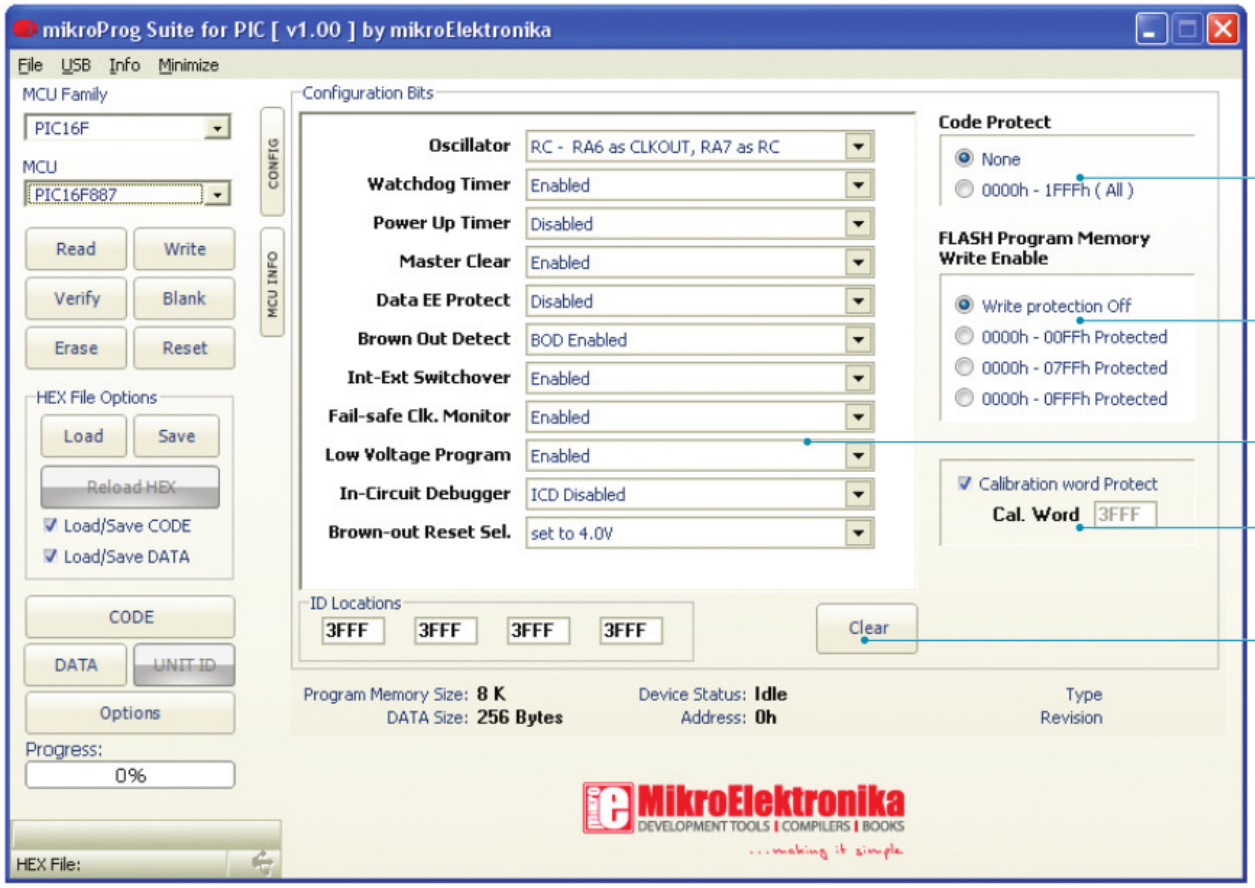
- A continuación ya podemos cargar el archivo .hex generado previamente.
- El proceso de la programación empieza cuando se pulsa el botón *Write*.
- La barra *Progress* permite conocer el progreso de su programación.





Transferencia del programa al PIC

- La opción *CONFIG*



Opción para proteger el código hex

Opción para deshabilitar cargar los datos en algunas localidades de memoria FLASH del microcontrolador

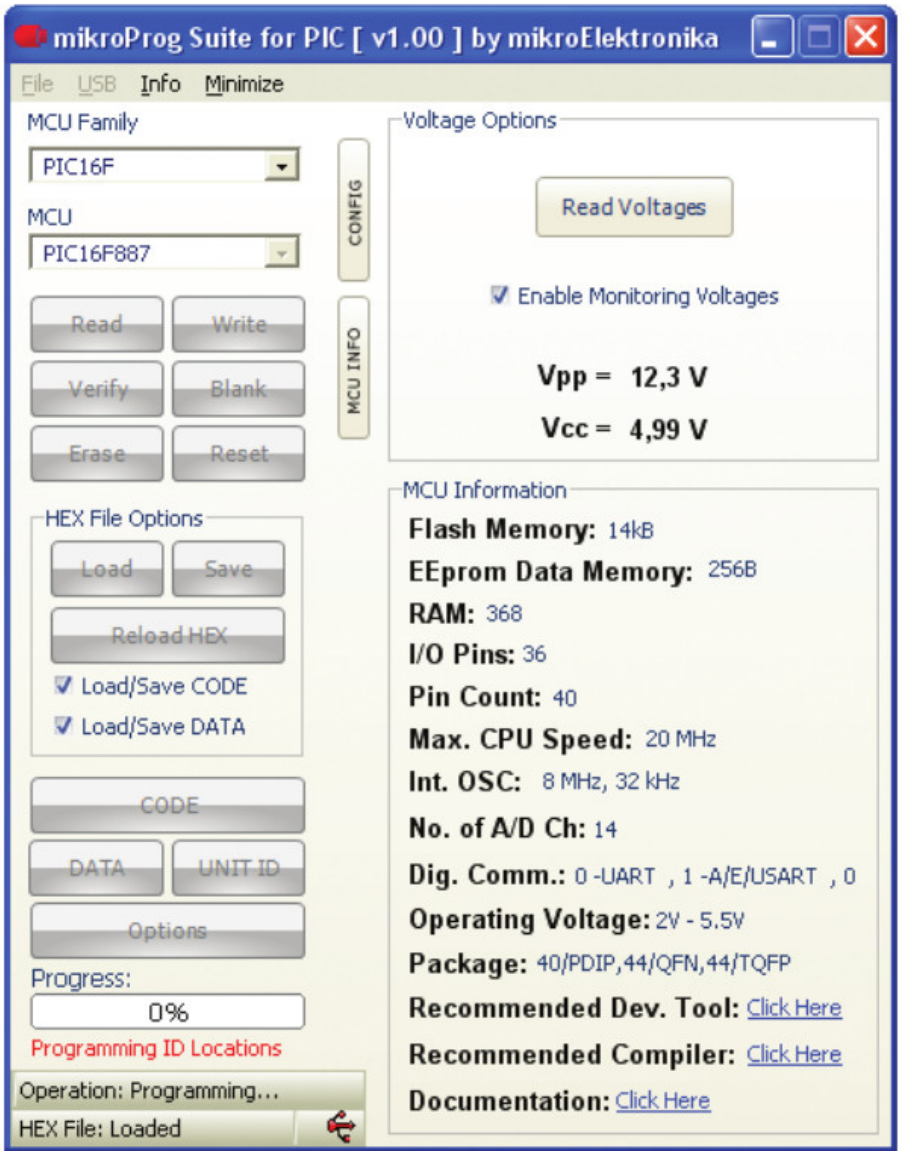
Opciones para ajustar los parámetros del microcontrolador

Protección de la palabra de calibración

Botón para ajustar los bits de configuración a los valores por defecto

Transferencia del programa al PIC

- La opción *MCU INFO*



Ejemplo

- Vamos a probar el programa realizado anteriormente en la placa UNI DS3

Crearemos un programa para un PIC16F877A funcionando a 8MHZ encargado de contar hasta 0x5f. Cuando lo alcance se detendrá en un bucle no operativo. El valor del contador se visualizará en 8 diodos LED conectados al puerto B.

```
#include <p16f877a.inc>

            LIST P=16f877a          ; Seleccionamos el micro
__config _WDT_OFF & _HS_OSC         ; Deshabilitamos el watch dog y fijamos el oscilador

; Asignación de etiquetas a registros.
f           EQU     0x01            ; registro f
portb       EQU     0x06            ; Dirección del registro del puerto B
estado      EQU     0x03            ; Dirección del registro de estado
conta       EQU     0x20            ; Lo usamos como variable contadora

            ORG     0               ; El programa comienza en la dirección 0
            GOTO    Inicio          ; salta a la dirección 5 para sobrepasar el vector INT.
            ORG     5

Inicio      BSF     estado,5        ; Selecciona banco 1 para llegar a TRISB
            MOVLW   0x00
            MOVWF   portb           ; Y se especifica que es de salida
            BCF     estado,5        ; Selección del banco 0 para trabajar con el puerto
            CLRF    conta           ; Ponemos nuestro contador a 0
bucle1      INCF    conta,f         ; conta + 1 --> conta (incrementa el contador)
            MOVF    conta,W         ; conta se carga en W
            MOVWF   portb           ; W se carga en el registro de datos del puerto B
            MOVLW   0x5f            ; W <-- 0x5f        (Final de cuenta deseado)
            SUBWF   conta,W         ; conta - W --> W. Si es cero, la cuenta estÃ¡ acabada
            BTFSS   estado,2        ; Explora Z y si vale 1 es que W vale 0
                                    ; se produce salto en ese caso por fin de cuenta
            GOTO bucle1                ; Si Z = 0 se vuelve a bucle1
bucle2      GOTO bucle2                ; Si Z = 1 se produce un bucle infinito
            END
```

Tareas a realizar

Desarrollar las siguientes tareas primero con simulación en MPLAB y luego transfiriéndolo a la placa UNI DS3:

1. Realizar un programa para el PIC16F77A que lea el estado de los 6 interruptores E0-E5 conectados al puerto A y refleje el nivel lógico de los mismos sobre los leds S0-S5 conectados al puerto B. ¿Qué consideraciones de configuración han de tenerse en cuenta en el puerto A? ¿Se encienden todos los leds? En caso negativo indicar la razón de este comportamiento.
2. Se pide realizar un programa para el PIC16F77A que active secuencialmente, de una en una, las ocho salidas de la puerta B (RB0-RB7), provocando un efecto de desplazamiento de derecha a izquierda.
3. Desarrollar una aplicación que encienda cada 0,5 segundos un led del puerto B (de menor a mayor peso), durante 0,5 segundos y posteriormente los leds correspondientes al puerto D (de mayor a menor peso). El proceso debe repetirse hasta que se active el pulsador correspondiente a RB0, quedando a partir de ese momento encendidos los leds del puerto B y D.
4. Desarrollar una aplicación que cuente el número de veces que se cierra el pulsador RB0 y las muestre en binario natural por el puerto D. Aunque se cierre el pulsador una vez, es posible que sobre el puerto D se produzca más de un incremento de cuenta. ¿Cuál puede ser la causa? ¿Cómo puedes solucionarlo?

Nota: para realizar estas tareas deberá realizarse una rutina de retardo que permita apreciar visualmente el resultado.