

Microcontroladores

Semestre: 2021-1

Profesor: Kalun José Lau Gan

Semana 4: Módulo Timer0

1

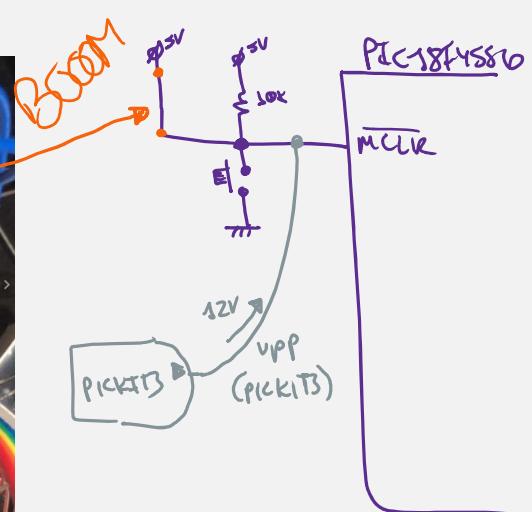
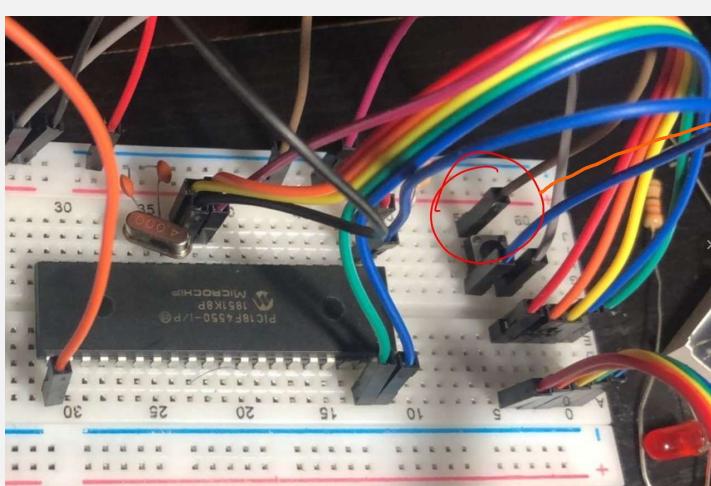
¿Preguntas previas?

- ¿Subirá el ppt de la semana 3?
 - En el transcurso de la semana, tienen la grabación disponible de la clase que posee mayor información.
- El PC solo se mencionó un ejemplo (decodificador de display de 7 segmentos). ¿Hay alguna otra utilidad?
 - El PC es un registro contador incremental +x el cuál almacena la dirección de memoria de la siguiente instrucción a ejecutar, si modificas dicho registro al término de la ejecución de la instrucción actual se irá a la dirección dada ahí.
 - Utilidad práctica es hacer saltos dentro de la memoria de programa pero no es muy conveniente ya que tenemos instrucción dedicadas a ello (GOTO, BRA, CALL).

2

- ~~clrf TRISD, 2~~
 - No es correcto, la instrucción clrf solo va el registro a la derecha para que se limpie
- ~~bsf TRISD, 2~~
 - Para hacer que el puerto RD2 sea salida se debe de usar bcf TRISD, 2

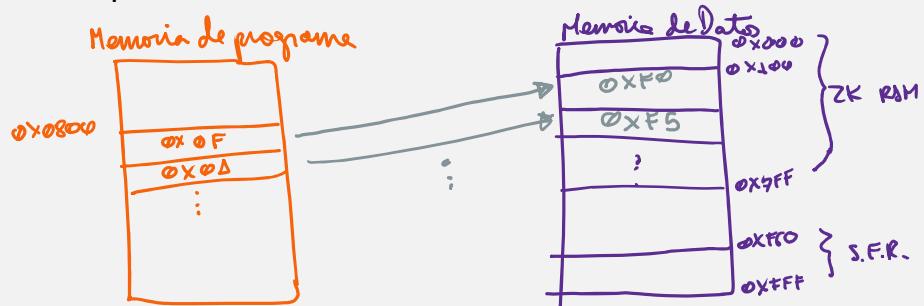
3



4

Ejemplo sobre manipulación de datos entre memoria de programa y memoria de datos

- Memoria de programa dirección 0x0800
 - 0x0F, 0x0A, 0x08, 0x10, 0xFF, 0x3D, 0x4B, 0x15
- En la memoria de datos en la dirección 0x100 escribir los datos que están en la memoria de programa en dirección 0x0800 pero en complemento



```

inicio:    movlw 0x08
           movwf TBLPTRH
           movlw 0x00
           movwf TBLPTRL
           lfsr 0, 0x100
           movlw .8
           cpfseq TBLPTRL
           goto aunno
           goto yatermine
           TBLRD*
           comf TABLAT, w
           movwf INDF0
           incf TBLPTRL, f
           incf FSR0L, f
           goto loop
yatermine:nop
end

```

5

Preguntas previas:

- Acerca del LB1:
 - Los grupos de asignan de manera aleatoria al inicio de la sesión de laboratorio y se habilitará el acceso del instructivo en PDF en el AV.

6

Sobre algoritmos:



7

Agenda:

- El modulo Timer 0
- Aplicaciones con temporizadores
- Multiplexación de displays de siete segmentos

8

El módulo Timer 0

- (Ref. Item 11 de la hoja técnica del microcontrolador PIC18F4550)
- Temporizador de cuenta ascendente
- Resolución 8 bits (0-255) ó **16 bits (0-65535)**
- Las cuentas del Timer0 se alojan en:
 - TMROH:TMR0L (16 bits)
 - TMR0L (8 bits)
- **Tener en consideración el procedimiento estricto sobre el tratamiento de la cuenta en modo 16 bits.**
- Diversas fuentes de reloj: interno (FOSC/4) o externo (pin6 TOCKI)
- Divisor de frecuencia al reloj de entrada (1:2 – 1:256)
- El desborde se produce cuando la cuenta esta en el valor mas alto y se recibe un pulso de reloj, ocasionando que la cuenta pase a 0 y levantándose la bandera de desborde (TMROIF=1)
- Al activarse TMROIF=1 se debe de bajar manualmente la bandera para que se pueda detectar un nuevo desborde.
- Al desbordarse puede emitir interrupción (TMROIF = 1), revisar interrupciones y sus 10 registros implicados
- Se usa el registro TOCON (SFR 0xFD5) para configurar el Timer0 (por defecto TOCON=0xFF)

9

El Timer 0

- Diagrama de bloques:

FIGURE 11-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)

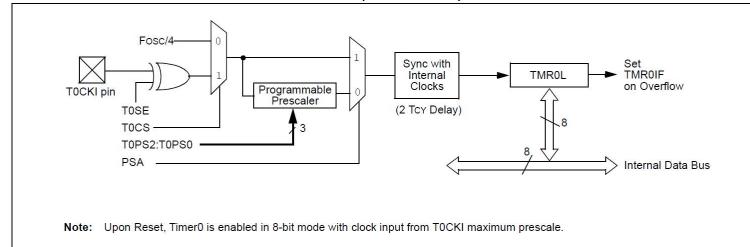
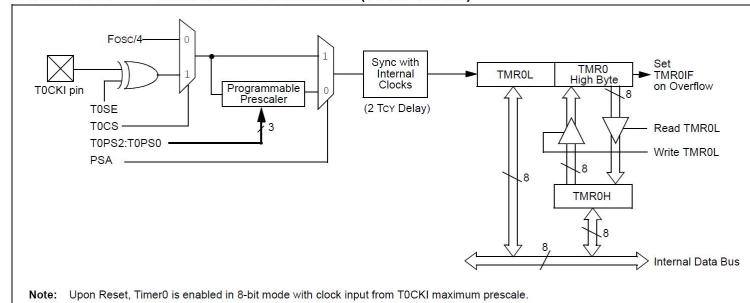


FIGURE 11-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)



10

Timer 0 – Modos de trabajo

- Modo temporizador (reloj interno)
 - Ej. Generador de ondas cuadradas, base de tiempo para la multiplexación de los displays de siete segmentos
 - **No se usa para aplicaciones en tiempo real (relojes, cronómetros)**
- Modo contador (empleando pin TOCKI)
 - Ej. Velocímetro para bicicleta

11

El Timer0 – Registro T0CON

REGISTER 11-1: T0CON: TIMER0 CONTROL REGISTER													
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1						
TMR0ON	T08BIT	TOCS	TOSE	PSA	T0PS2	T0PS1	T0PS0						
bit 7													
bit 0													
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											
bit 7	TMR0ON: Timer0 On/Off Control bit												
	1 = Enables Timer0												
	0 = Stops Timer0												
bit 6	T08BIT: Timer0 8-Bit/16-Bit Control bit												
	1 = Timer0 is configured as an 8-bit timer/counter												
	0 = Timer0 is configured as a 16-bit timer/counter												
bit 5	TOCS: Timer0 Clock Source Select bit												
	1 = Transition on TOCKI pin												
	0 = Internal instruction cycle clock (CLKO)												
bit 4	TOSE: Timer0 Source Edge Select bit												
	1 = Increment on high-to-low transition on TOCKI pin												
	0 = Increment on low-to-high transition on TOCKI pin												
bit 3	PSA: Timer0 Prescaler Assignment bit												
	1 = Timer0 prescaler is NOT assigned. Timer0 clock input bypasses prescaler.												
	0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.												
bit 2:0	T0PS2:T0PS0: Timer0 Prescaler Select bits												
	111 = 1:256 Prescale value												
	110 = 1:128 Prescale value												
	101 = 1:64 Prescale value												
	100 = 1:32 Prescale value												
	011 = 1:16 Prescale value												
	010 = 1:8 Prescale value												
	001 = 1:4 Prescale value												
	000 = 1:2 Prescale value												

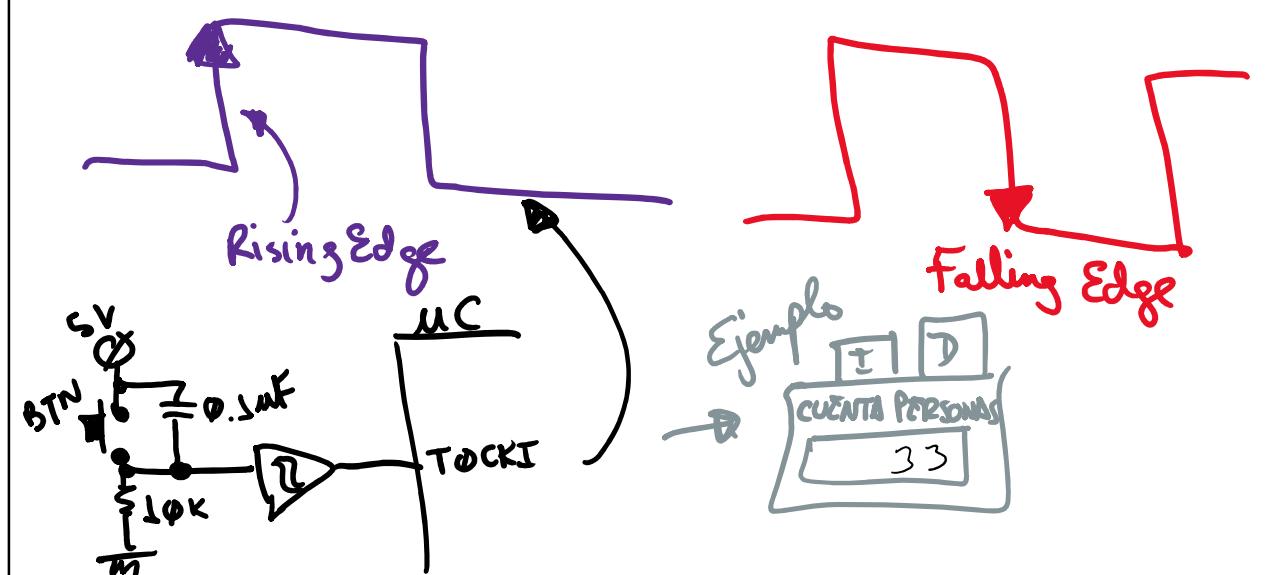
12

Procedimiento para ingresar una cuenta inicial al Timer 0 en modo 16 bits

1. Si por ejemplo se quiere ingresar el número 5536 como cuenta inicial, convertirlo a hexadecimal (DEC 5536 = HEX 0x15A0)
2. Se ingresa el dato de 8 bit mas significativo a TMR0H, en el ejemplo 0x15 hacia TMR0H.
3. Se ingresa el dato de 8 bit menos significativo a TMR0L, haciendo esto se sube en simultáneo el TMR0H al registro de cuentas del Timer0, en el ejemplo 0xA0 hacia TMR0L.
4. Recordar que luego del desborde se deberá ingresar nuevamente la cuenta inicial para preservar el temporizado de manera continua.

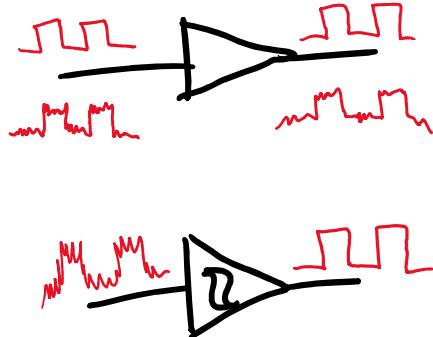
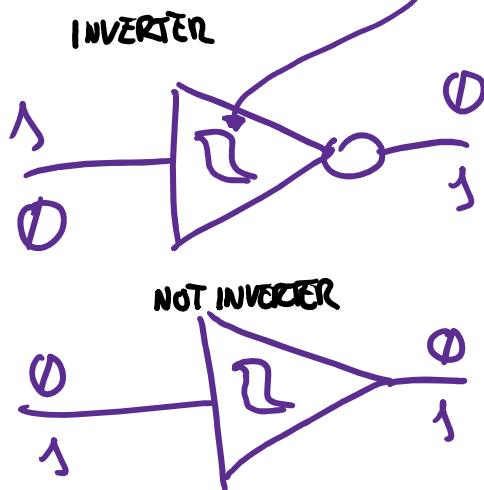
13

Recordando Rising Edge vs Falling Edge



14

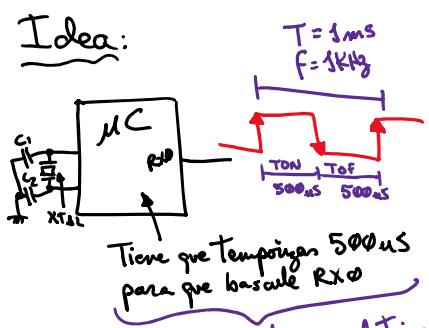
Recordando el SCHMITT TRIGGER



15

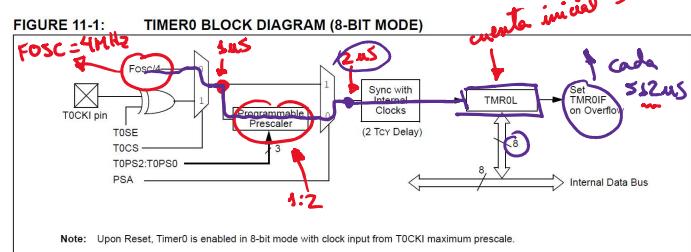
Ejemplo: Generar una onda cuadrada de 1KHz DC 50 % empleando el Timer0 modo 8 bits:

Idea:



Tiene que temporizar 500μs para que bascule Rx0
Lo va a hacer el Timer0

¿Cómo hago para que el Timer0 temporice 500μs?



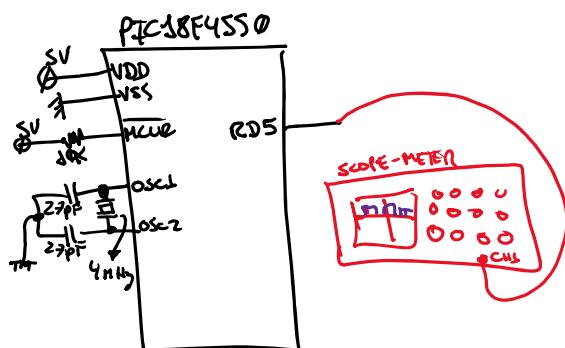
Resumen:

- 1: Modo 8 bit - temporizador ($F_{OSC}/4$)
 - 2: Prescaler 1:2
 - 3: Cuenta inicial de 6
- $\Rightarrow T0CON = \#C0$

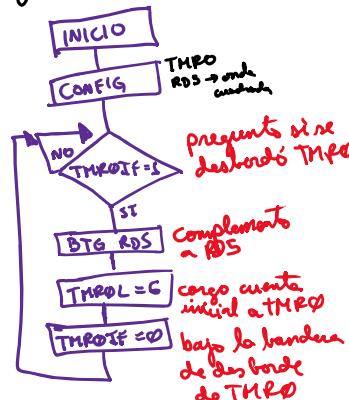
16

Ejemplo: Generar una onda cuadrada de 1KHz empleando el Timer0 modo 8 bits:

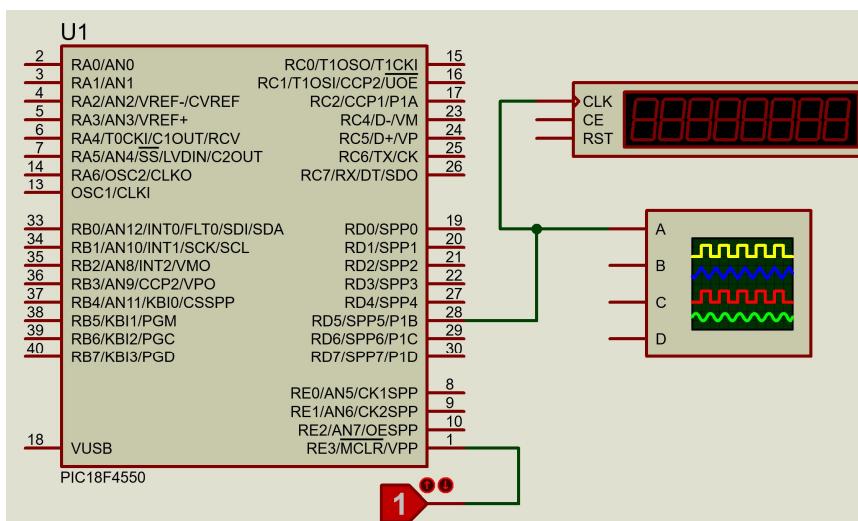
Circuito:



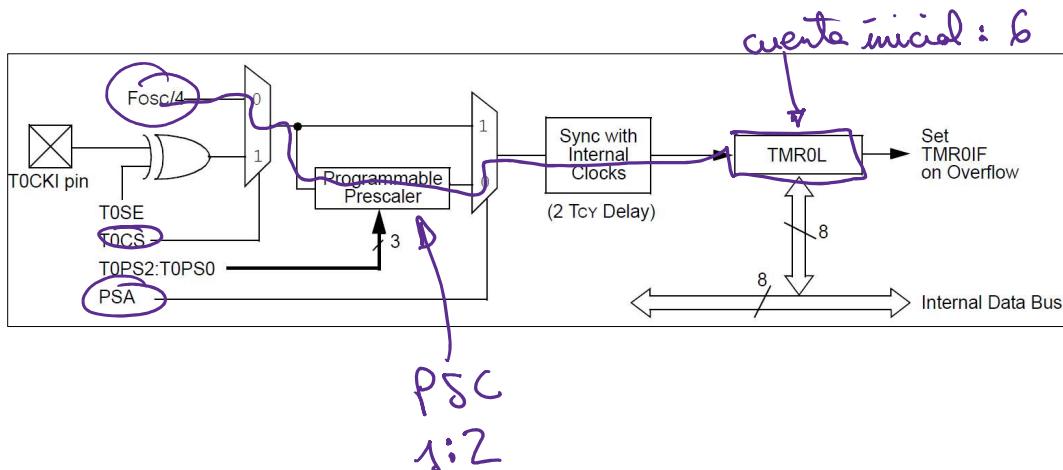
Algoritmo:



Hardware



Continuación...



19

Configurar el TOCON:

REGISTER 11-1: TOCON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMROON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7	X	X	X	X	X	X	bit 0

Legend:

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

bit 7 **TMROON: Timer0 On/Off Control bit**
 1 = Enables Timer0
 0 = Stops Timer0

bit 6 **T08BIT: Timer0 8-Bit/16-Bit Control bit**
 1 = Timer0 is configured as an 8-bit timer/counter
 0 = Timer0 is configured as a 16-bit timer/counter

bit 5 **T0CS: Timer0 Clock Source Select bit**
 1 = Transition on T0CKI pin (**Fosc/4**)
 0 = Internal instruction cycle clock (CLKO)

bit 4 **T0SE: Timer0 Source Edge Select bit**
 1 = Increment on high-to-low transition on T0CKI pin
 0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA: Timer0 Prescaler Assignment bit**
 1 = Timer0 prescaler is NOT assigned. Timer0 clock input bypasses prescaler.
 0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

bit 2-0 **T0PS2:T0PS0: Timer0 Prescaler Select bits**

111 = 1:256 Prescale value
110 = 1:128 Prescale value
101 = 1:64 Prescale value
100 = 1:32 Prescale value
011 = 1:16 Prescale value
010 = 1:8 Prescale value
001 = 1:4 Prescale value
000 = 1:2 Prescale value

Modo contador
(Contareros)

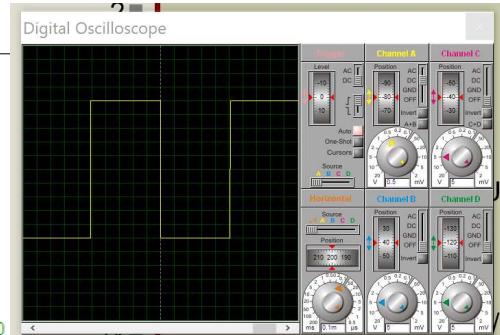
20

Propuesta de código en MPASM de lo anterior

```

17      org 0x0000
18      goto init_conf
19
20      org 0x0020
21  init_conf:
22      movlw 0xC0
23      movwf T0CON      ;Timer0 ON 8bit, FOSC/4, 1:2PSC
24      bcf TRISD, 5    ;RD5 como salida
25  loop:
26      btfss INTCON, TMR0IF  ;Pregunto si se desbordó TMR0
27      goto loop          ;No se desbordó
28      btg LATD, 5        ;Basculo RD5
29      movlw .6
30      movwf TMROL      ;Cargo valor inicial de 6 a TMRO
31      bcf INTCON, TMR0IF ;Bajamos la bandera de desborde de TMRO
32      goto loop
33  end

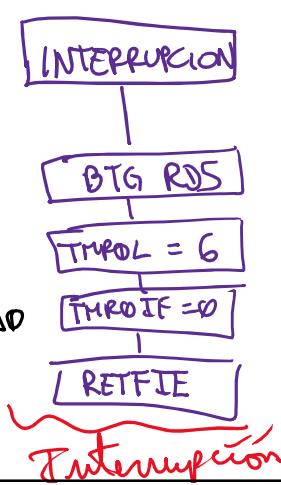
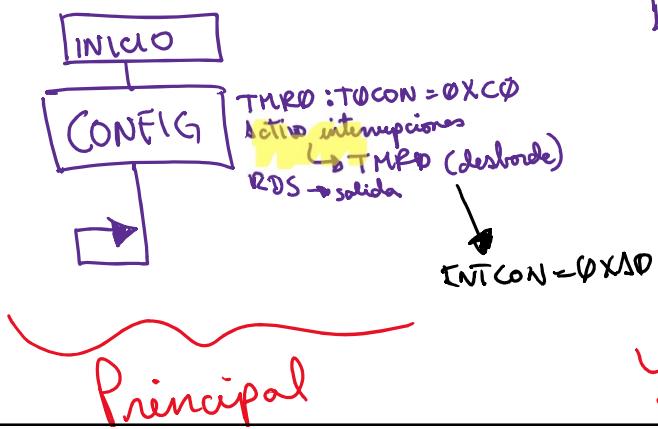
```



21

Mejora del generador de onda cuadrada de 1KHz empleando interrupciones

Diagramas de flujo:



22

Código en MPASM del ejemplo pero ahora con TMRO_ISR

```

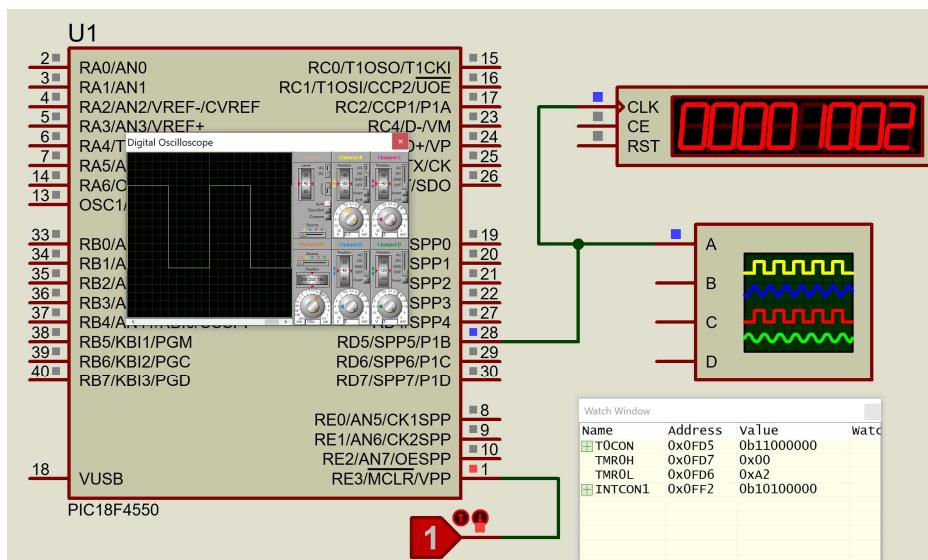
17      org 0x0000          ;Vector de RESET
18      goto init_conf
19
20      org 0x0008          ;Vector de interrupcion
21      goto TMRO_ISR
22
23      org 0x0020
24      init_conf:
25      movlw 0xC0
26      movwf TOCON           ;Timer0 ON 8bit, FOSC/4, 1:2PSC
27      movlw 0xA0
28      movwf INTCON          ;Activamos la interrupción de desborde de TMRO
29      bcf TRISD, 5          ;RD5 como salida
30      loop:
31      goto loop
32
33      TMRO_ISR:
34      btg LATD, 5          ;Basculo RD5
35      movlw .12
36      movwf TMROL           ;Cargo valor inicial de 6 a TMRO
37      bcf INTCON, TMROIF   ;Bajamos la bandera de desborde de TMRO
38      retfie
39      end

```

23

Simulación

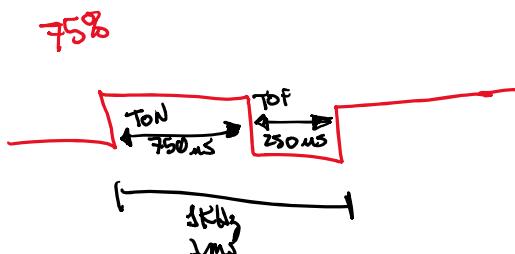
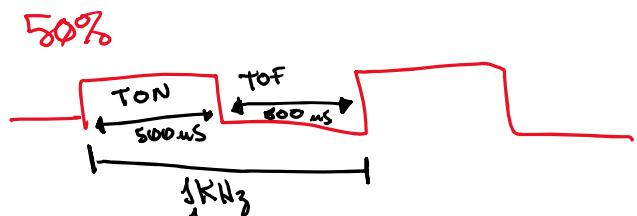
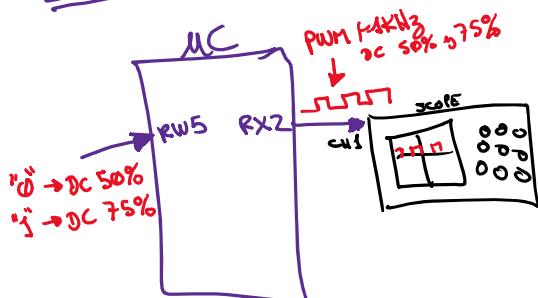
- Se aprecia una mejor precisión:



24

Ejemplo: Desarrollar un generador de PWM con frecuencia 1KHz y con dos opciones de Duty Cycle 50% y 75%.

Idea:

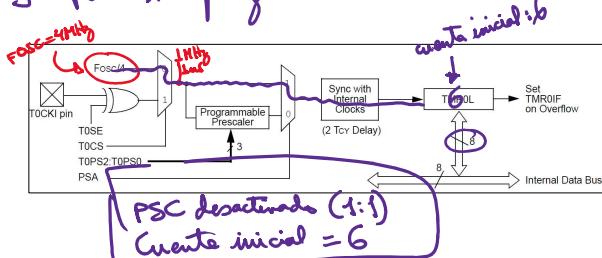


Nota: Debemos de configurar tres temporizaciones en el Timer0:
250μs, 500μs y 750μs.

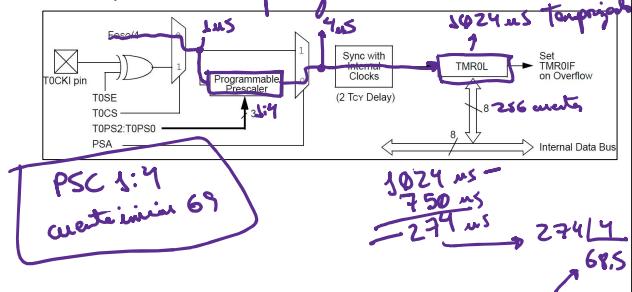
25

Cont. del ejemplo: Configuración del Timer0 según tiempos de temporización requeridos

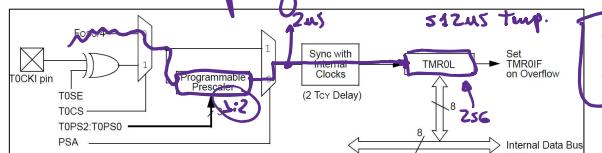
1º Para temporizar 250μs:



3º Para temporizar 750μs

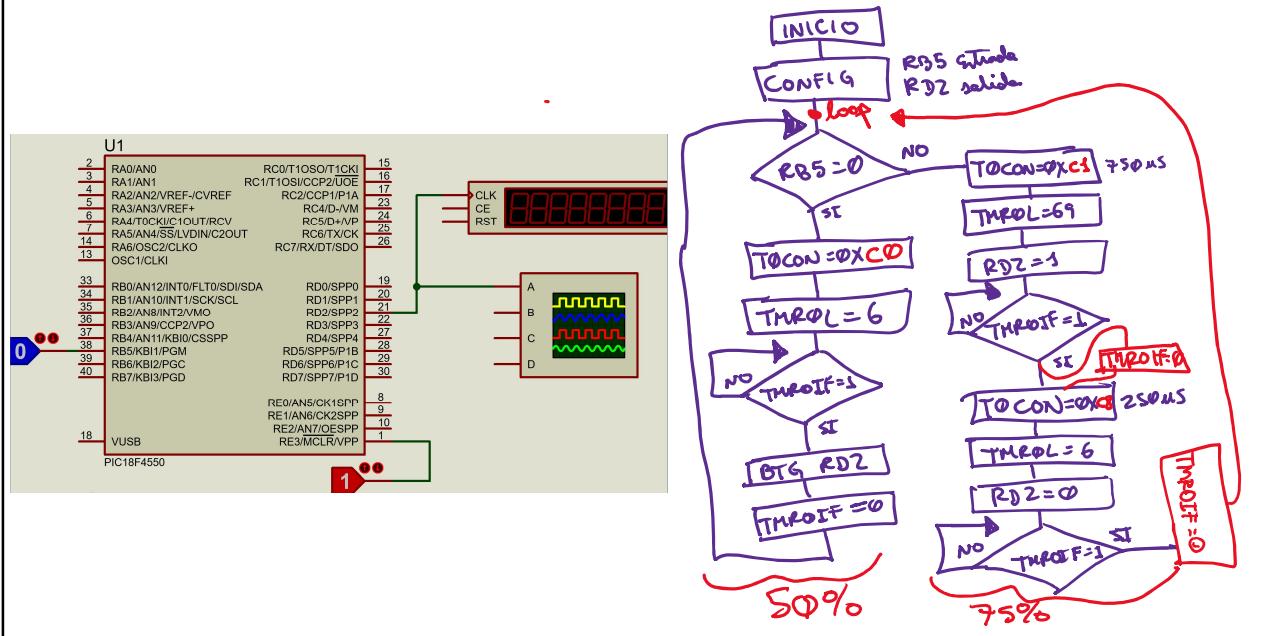


2º Para temporizar 500μs



26

Cont. del ejemplo: Hardware y desarrollo del diagrama de flujo



27

Cont. del ejemplo: código en MPASM

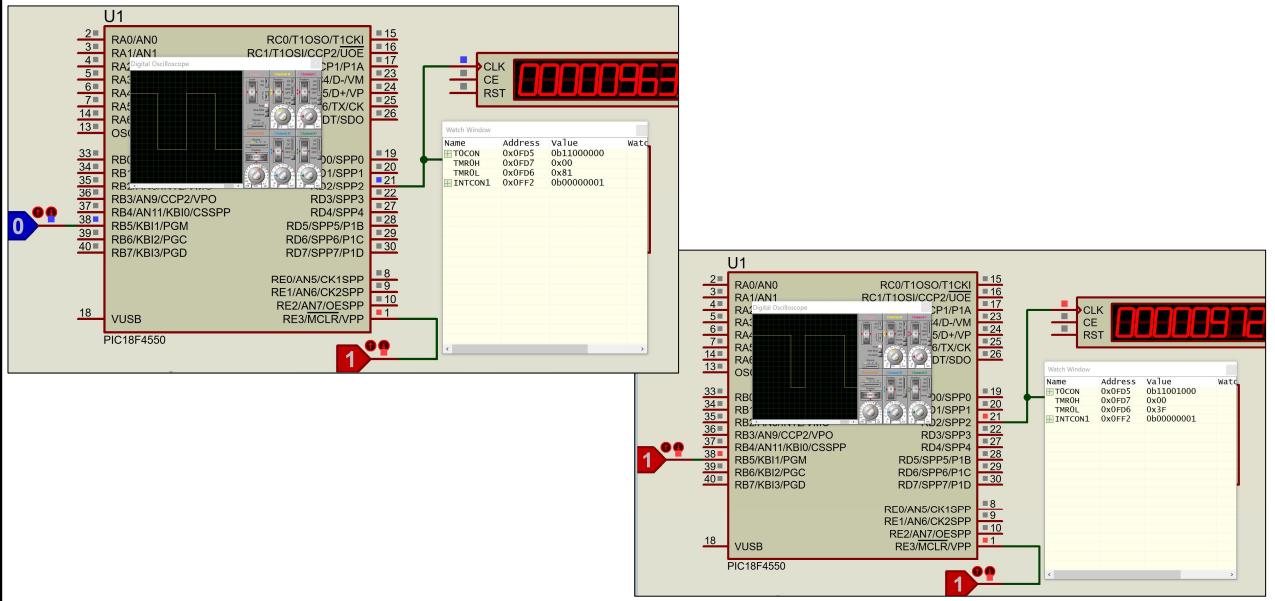
```

17     org 0x0000
18     goto init_conf
19
20     org 0x0020
21     init_conf:
22         bcf TRISD, 2
23     loop:
24         btfsc PORTB, 5
25         goto setentaycinco
26         movlw 0xC0
27         movwf T0CON
28         movlw .6
29         movwf TMR0L
30     otrol:
31         btfss INTCON, TMR0IF
32         goto otrol
33         btf LATD, 2
34         bcf INTCON, TMR0IF
35         goto loop
36     setentaycinco:
37         movlw 0xC1
38         movwf T0CON
39         movlw .69
40         movwf TMR0L
41         bsf LATD, 2
42
43     otro2:
44         btfss INTCON, TMR0IF
45         goto otro2
46         bcf INTCON, TMR0IF
47         movlw 0xC8
48         movwf T0CON
49         movlw .6
50         movwf TMR0L
51         bcf LATD, 2
52
53     otro3:
54         btfss INTCON, TMR0IF
55         goto otro3
56         bcf INTCON, TMR0IF
      goto loop
      end

```

28

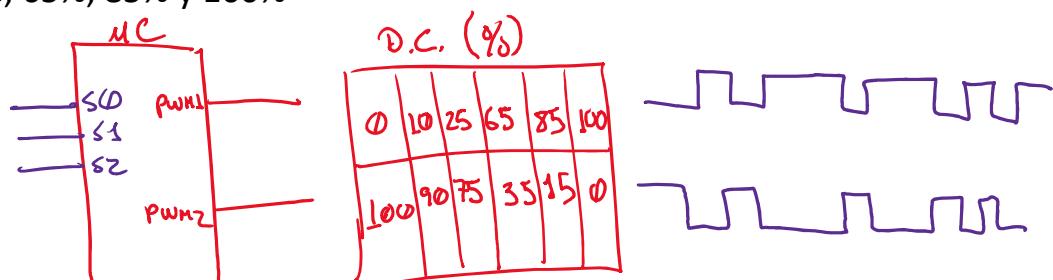
Cont. del ejemplo: Simulación



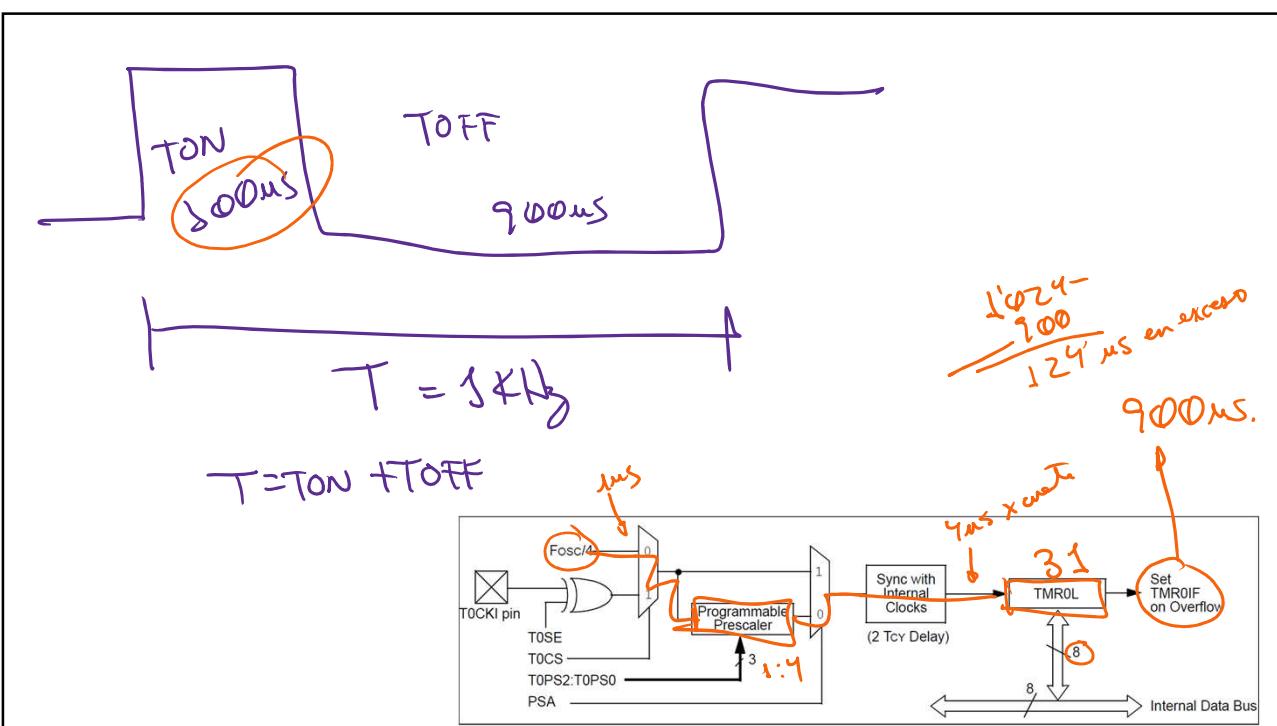
29

Fin de la sesión

- Links adicionales:
 - Microchip Timer0 Tutorial part1: <http://ww1.microchip.com/downloads/en/devicedoc/51682a.pdf>
 - Microchip Timer0 Tutorial part2: <http://ww1.microchip.com/downloads/en/DeviceDoc/51702a.pdf>
- Ejercicio: Desarrollar un generador de PWM 2KHz con dos salidas complementarias y con opciones de dutycycle siguientes: 0%, 10%, 25%, 65%, 85% y 100%



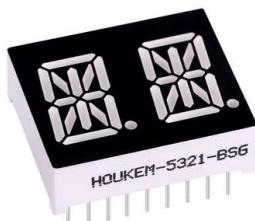
30



31

Cuestionario:

- En un PoR. ¿En qué estado se encuentra el Timer0, encendido o apagado?
- Si Fosc = 24MHz. ¿Cuál es la temporización máxima del Timer0 en modo 16 bits?
- ¿Qué es lo que hace la instrucción BTG?
- Si Fosc = 12MHz. ¿Cuánto se demorará en ejecutar la instrucción CPFSGT?
- Hacer un circuito de conexión entre el microcontrolador PIC18F4550 y el siguiente dispositivo:



32