# Microcontroladores
# Semana 6

Semestre 2023-2

Por Kalun José Lau Gan

1

## Preguntas previas

- ¿Cómo funcionan los sensores ultrasónicos?
  - Funcionando con señales ultrasónicas, se envia un pulso, éste rebota y regresa y se mide el tiempo recorrido por dicha señal
- ¿Los grupos de TP/TF pueden ser de diferente turno de laboratorio?
  - Si, la restricción es que no pueden formar con alumnos de diferentes secciones.
- ¿?

2

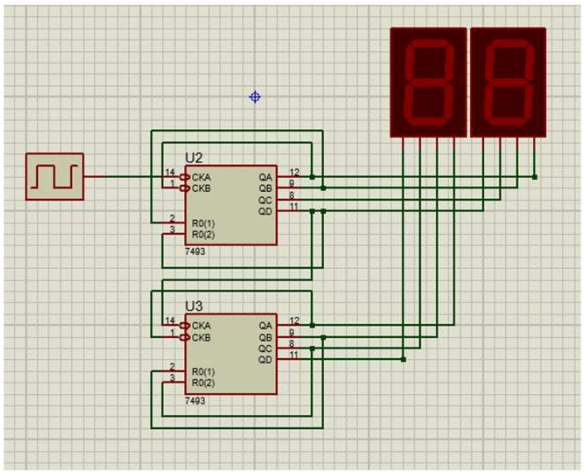
# Agenda:

- El módulo Timer0
  - Modo 8bit
  - Modo 16bit
  - Registros de configuración y operación
- Cálculo de temporización
- Generación de ondas cuadradas periódicas

3

# La previa: contadores digitales

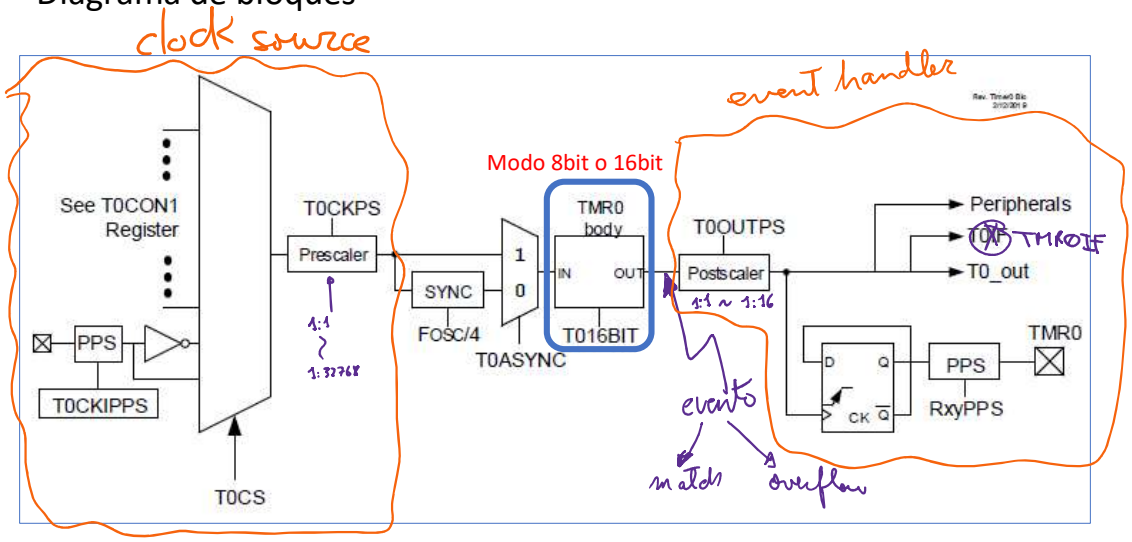
- Sistemas secuenciales, poseen reloj, un registro de cuenta, dependiendo del su función, puede incrementar o puede decrementar la cuenta según la entrada de reloj.




4

# El módulo Timer 0:

- Diagrama de bloques




5

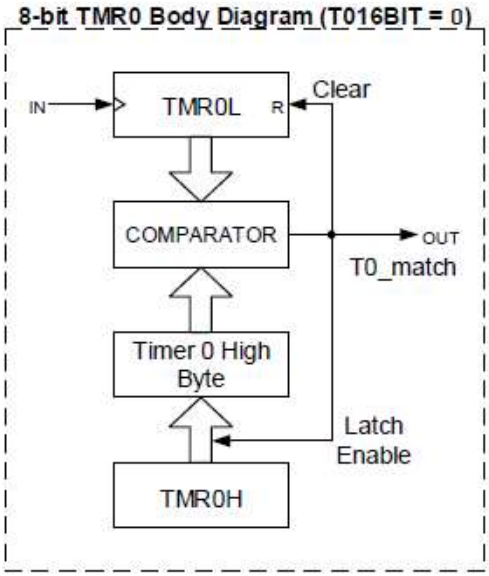
# El módulo Timer 0

- (Ref. Item 24 de la hoja técnica del microcontrolador PIC18F57Q43)
- Temporizador de cuenta ascendente
- Resolución 8 bits (0-254) ó 16 bits (0-65535)
- Las cuentas del Timer0 se alojan en:
  - TMR0H:TMR0L (16 bits)
  - TMR0L (8 bits)
- Diversas fuentes de reloj (revisar T0CON1 y T0CS)
- Divisor de frecuencia al reloj de entrada PRESCALER (1:1 – 1:32768)
- POSTSCALER de 1:1 a 1:16 (incrementos de uno en uno)
- Al activarse TMR0IF=1 ó T0IF=1 se debe de bajar manualmente la bandera para que se pueda detectar un nuevo desborde ó evento de match (simplemente haciendo “bcf PIR3, 7”; siendo el bit 7 el TMR0IF).
- En un evento de match o al desbordarse puede emitir interrupción al CPU si TMR0IE=1), revisar capítulo 11 de la hoja técnica.

6

# El módulo Timer 0:

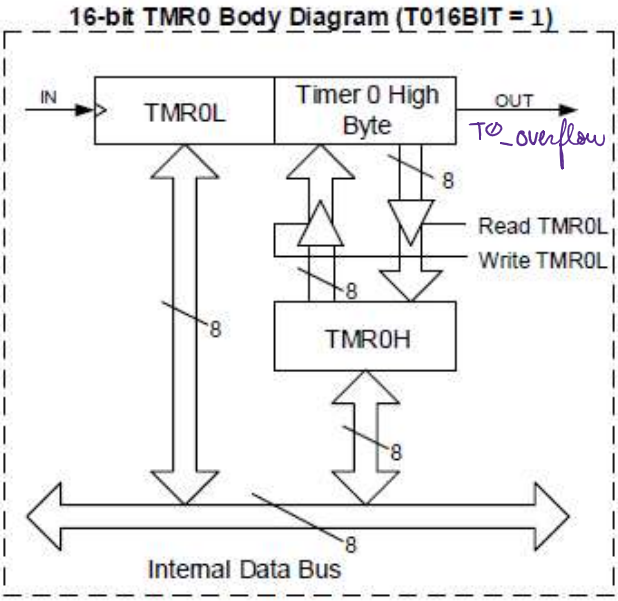
- Modo de trabajo en 8 bits:
  - Mejorado con respecto al Timer0 presente en el PIC18F4550 ó PIC18F45K50
  - El TMR0L se usa como el registro de cuenta
  - El TMR0H se utiliza como valor de comparación
  - Cuenta máxima es cuando TMR0H es 255, haciendo que se mande a cero la cuenta actual, quiere decir que tendrás un rango de 0-254
  - Cuando TMR0L es igual a TMR0H se produce un evento de "match" el cual limpia la cuenta y actualiza el valor de comparación.
  - Nunca se desborda




7

# El módulo Timer 0:

- Modo de trabajo en 16 bits:
- Tener en consideración el procedimiento estricto sobre el proceso de carga de un valor en la cuenta en modo 16 bits: Primero cargar en TMR0H y luego en TMR0L.
- El evento de desborde se produce cuando la cuenta esta en el valor mas alto (65535) y se recibe un pulso de reloj, ocasionando que la cuenta pase a 0 y levantándose la bandera de desborde (TMR0IF=1 ó T0IF=1) siempre y cuando POSTSCALER 1:1




8

# El módulo Timer 0:

- Temporización máxima si HFINTOSC = 4MHz
  - Empleando Opción FOSC/4 = 1MHz (1µs) como fuente de reloj al TMR0
  - Prescaler = 1:32768 (32768 µs por cuenta)
  - Modo = 16bits (65536 cuentas)
  - En el evento de desborde: 2147483648 µs (35 minutos)
  - Postscaler = 1:16
  - Nos sale al final 34,359,738,368 µs de temporización máxima! (aprox 9.5 horas)

9

# El módulo Timer 0

- Registros implicados en la operación del Timer0:
  - Registros de cuenta
    - TMR0H:TMR0L (16bits)
    - TMR0H es valor de comparación y TMR0L es el registro de cuenta (8bits)
  - Registros de configuración T0CON0 y T0CON1
  - Registros PIE3 (habilitadores), PIR3 (banderas) y IPR3 (prioridades) ubicando en la VIC (vectored interrupt controller cap 11)

bank 3

| Address | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0318 | TMR0L | 7:0 | TMR0L[7:0] | | | | | | | |
| 0x0319 | TMR0H | 7:0 | TMR0H[7:0] | | | | | | | |
| 0x031A | T0CON0 | 7:0 | EN | | OUT | MD16 | OUTPS[3:0] | | | |
| 0x031B | T0CON1 | 7:0 | CS[2:0] | | | ASYNC | CKPS[3:0] | | | |
| 0x04A1 | PIE3 | 7:0 | TMR0IE | CCP1IE | TMR1GIE | TMR1IE | TMR2IE | SPI1IE | SPI1TXIE | SPI1RXIE |
| 0x04B1 | PIR3 | 7:0 | TMR0IF | CCP1IF | TMR1GIF | TMR1IF | TMR2IF | SPI1IF | SPI1TXIF | SPI1RXIF |
| 0x0365 | IPR3 | 7:0 | TMR0IP | CCP1IP | TMR1GIP | TMR1IP | TMR2IP | SPI1IP | SPI1TXIP | SPI1RXIP |

10

# El módulo Timer 0

- Registro T0CON0:
  - Habilitador del módulo
  - Señal OUT
  - Modo (8 ó 16 bits)
  - Postscaler

**Name:** T0CON0
**Address:** 0x31A

Timer0 Control Register 0

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN | | OUT | MD16 | OUTPS[3:0] | | | |
| Access | R/W | | R | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – EN** TMR0 Enable

| Value | Description |
|---|---|
| 1 | The module is enabled and operating |
| 0 | The module is disabled |

**Bit 5 – OUT** TMR0 Output

**Bit 4 – MD16** 16-Bit Timer Operation Select

| Value | Description |
|---|---|
| 1 | TMR0 is a 16-bit timer |
| 0 | TMR0 is an 8-bit timer |

**Bits 3:0 – OUTPS[3:0]** TMR0 Output Postscaler (Divider) Select

| Value | Description |
|---|---|
| 1111 | 1:16 Postscaler |
| 1110 | 1:15 Postscaler |
| 1101 | 1:14 Postscaler |
| 1100 | 1:13 Postscaler |
| 1011 | 1:12 Postscaler |
| 1010 | 1:11 Postscaler |
| 1001 | 1:10 Postscaler |
| 1000 | 1:9 Postscaler |
| 0111 | 1:8 Postscaler |
| 0110 | 1:7 Postscaler |
| 0101 | 1:6 Postscaler |
| 0100 | 1:5 Postscaler |
| 0011 | 1:4 Postscaler |
| 0010 | 1:3 Postscaler |
| 0001 | 1:2 Postscaler |
| 0000 | 1:1 Postscaler |

11

# El módulo Timer 0

- Registro T0CON1:
  - Fuente de reloj
  - Sincronismo
  - Prescaler

**Name:** T0CON1
**Address:** 0x31B

Timer0 Control Register 1

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CS[2:0] | | | ASYNC | CKPS[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:5 – CS[2:0]** Timer0 Clock Source Select

| Value | Description |
|---|---|
| 111 | CLC1_OUT |
| 110 | SOSC |
| 101 | MFINTOSC (500 kHz) |
| 100 | LFINTOSC |
| 011 | HFINTOSC |
| 010 | $F_{OSC}/4$ |
| 001 | Pin selected by T0CKIPPS (Inverted) |
| 000 | Pin selected by T0CKIPPS (Non-inverted) |

**Bit 4 – ASYNC** TMR0 Input Asynchronization Enable

| Value | Description |
|---|---|
| 1 | The input to the TMR0 counter is not synchronized to system clocks |
| 0 | The input to the TMR0 counter is synchronized to Fosc/4 |

**Bits 3:0 – CKPS[3:0]** Prescaler Rate Select

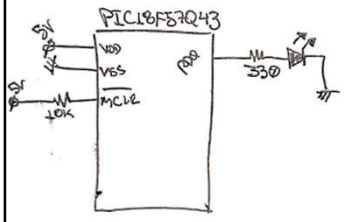
| Value | Description |
|---|---|
| 1111 | 1:32768 |
| 1110 | 1:16384 |
| 1101 | 1:8192 |
| 1100 | 1:4096 |
| 1011 | 1:2048 |
| 1010 | 1:1024 |
| 1001 | 1:512 |
| 1000 | 1:256 |
| 0111 | 1:128 |
| 0110 | 1:64 |
| 0101 | 1:32 |
| 0100 | 1:16 |
| 0011 | 1:8 |
| 0010 | 1:4 |
| 0001 | 1:2 |
| 0000 | 1:1 |

12

# Ejercicios:

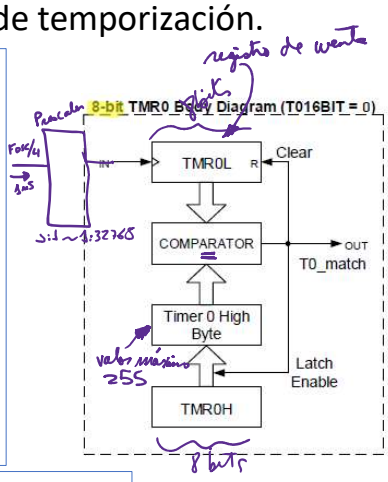
- Prender y apagar un LED en RD0 con un periodo de 1000ms aproximadamente (TON/TOF = 500ms) empleando el Timer0 como fuente de temporización.

Circuito de la aplicación:




$T = \frac{1}{F}$

Si uso modo 8 bits y fuente de reloj para el Timer0 FOSC/4 sabiendo que FOSC=4MHz:
- Si PSC es 1:1 cada cuenta es 1µs, voy a poder temporizar solo 255µs (evento match)
- Si PSC es 1:32768 entonces cada cuenta es 32768µs, la temporización máxima será 8.3 segundos aprox!
- Si PSC es 1:8192, cada cuenta es ahora 8192us, la temporización máxima será 2 segundos aprox.
- Si PSC es 1:2048, quiere decir que cada cuenta es de 2048µs, la temporización máxima será de aprox 500 ms (522.24 ms)!
- Si valor de comparación lo reduzco a 244, obtendré 499.71 ms, mucho mas cercano al valor solicitado)



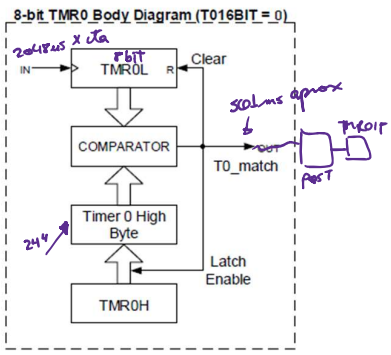

Lo que tenemos en configurar en el Timer0:
T0CON0: 80H (Timer0 enabled, POSTSCALER 1:1)
T0CON1: 4BH (FOSC/4, PRESCALER 1:2048)
TMR0H: 244

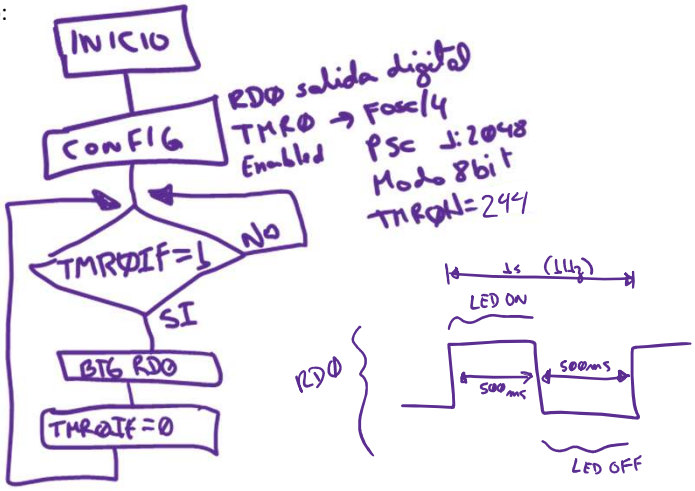
13

# Ejercicios:

- Prender y apagar un LED en RD0 con un periodo de 1000ms aproximadamente (TON/TOF = 500ms) empleando el Timer0 como fuente de temporización.

Diagrama de flujo:




14

# Ejercicios:

- Prender y apagar un LED en RD0 con un periodo de 500ms aproximadamente empleando el Timer0 como fuente de temporización.

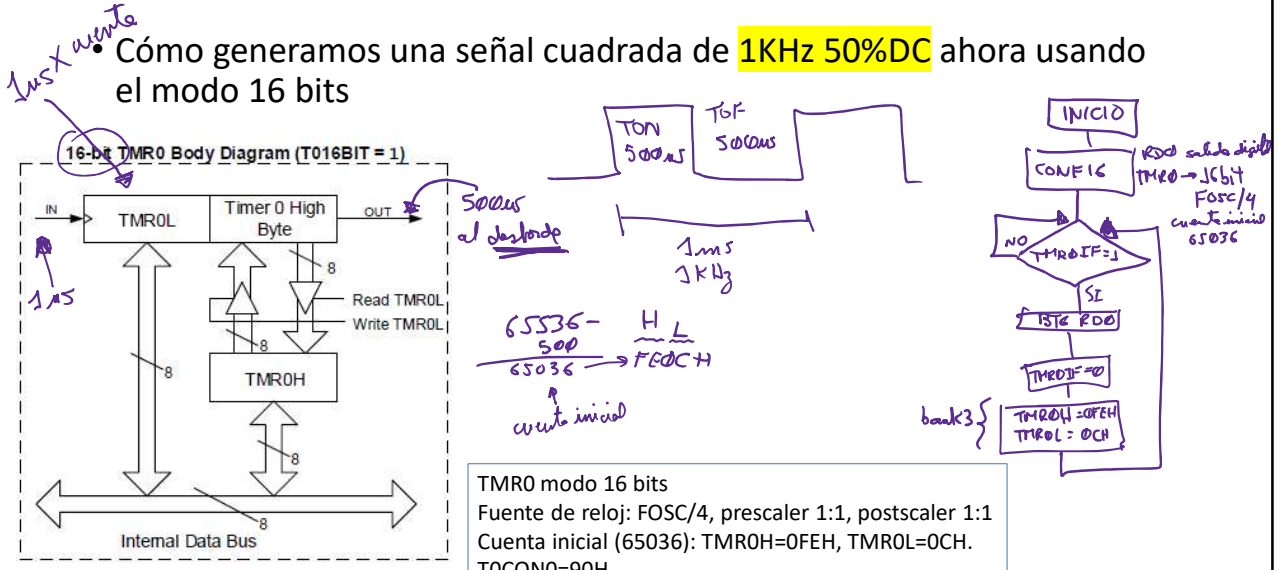
Código en XC8 PIC Assembler:

```
1         PROCESSOR 18F57Q43
2         #include "cabecera.inc"
3
4         PSECT upcino, class=CODE, reloc=2, abs
5     upcino:
6         ORG 000000H
7         bra configuro
8
9         ORG 000100H
10    configuro:
11        movlb 0H
12        movlw 60H
13        movwf OSCCON1, 1
14        movlw 02H
15        movwf OSCFRQ, 1
16        movlw 40H
17        movwf OSCEN, 1
18        movlb 3H
19        movlw 80H
20        movwf T0CON0, 1     ;TMR0 enabled, 8bit mode, postscaler 1:1
21        movlw 4BH
22        movwf T0CON1, 1     ;FOSC/4, async, prescaler 1:2048
23        movlw 245
24        movwf TMR0H         ;Valor de comparación 245
25        movlb 4H
26        bcf TRISD, 0, 1     ;RD0 como salida
27        bcf ANSELD, 0, 1    ;RD0 como digital
28
29    inicio:
30        btfss PIR3, 7, 1    ;Pregunto si hubo match en TMR0, TMR0IF=1
31        bra inicio
32        btg LATD, 0, 1      ;Complementamos RD0
33        bcf PIR3, 7, 1      ;Bajamos la bandera TMR0 IF
34        bra inicio
35
36        end upcino
```

15

# Ejercicios:

- Cómo generamos una señal cuadrada de 1KHz 50%DC ahora usando el modo 16 bits




TMR0 modo 16 bits
Fuente de reloj: FOSC/4, prescaler 1:1, postscaler 1:1
Cuenta inicial (65036): TMR0H=0FEH, TMR0L=0CH.
T0CON0=90H
T0CON1=40H

16

# Ejercicios:

- Cómo generamos una señal cuadrada de 1KHz 50%DC ahora usando el modo 16 bits

Código en XC8 PIC Assembler:

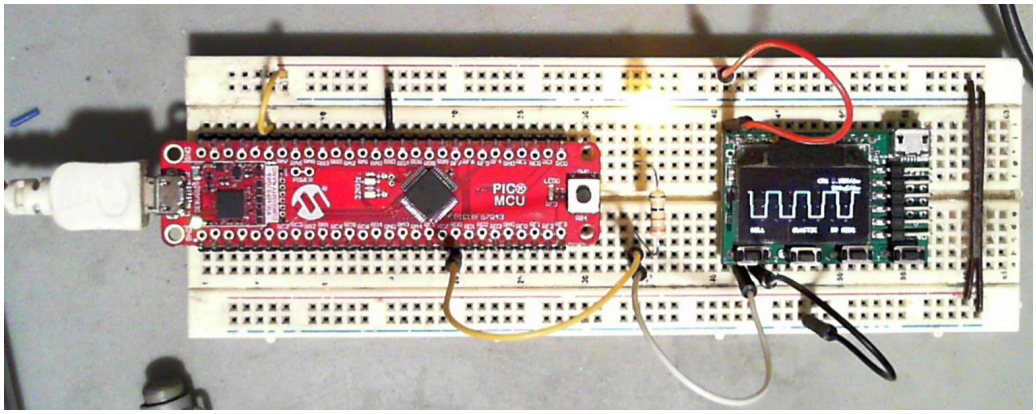
```
1         PROCESSOR 18F57Q43
2         #include "cabecera.inc"
3
4         PSECT upcino, class=CODE, reloc=2, abs
5     upcino:
6         ORG 000000H
7         bra configuro
8
9         ORG 000100H
10    configuro:
11        movlb 0H
12        movlw 60H
13        movwf OSCCON1, 1
14        movlw 02H
15        movwf OSCFRQ, 1
16        movlw 40H
17        movwf OSCEN, 1
18        movlb 3H
19        movlw 90H
20        movwf T0CON0, 1     ;TMR0 enabled, 16bit, postscaler 1:1
21        movlw 40H
22        movwf T0CON1, 1     ;FOSC/4, prescaler 1:1, async
23        movlb 4H
24        bcf TRISD, 0, 1     ;RD0 como salida
25        bcf ANSELD, 0, 1    ;RD0 como digital
```

```
27    inicio:
28        movlb 4H
29        btfss PIR3, 7, 1    ;Pregunto si TMR0IF=1
30        bra inicio          ;Aun no se levanta TMR0IF
31        btg LATD, 0, 1      ;Complemento a RD0
32        bcf PIR3, 7, 1      ;Bajamos bandera TMR0IF
33        movlb 3H
34        movlw 0FEH
35        movwf TMR0H, 1
36        movlw 0CH
37        movwf TMR0L, 1      ;Precarga de cuenta 65036 a TMR0
38        bra inicio
39
40        end upcino
```

17

# Ejercicios:

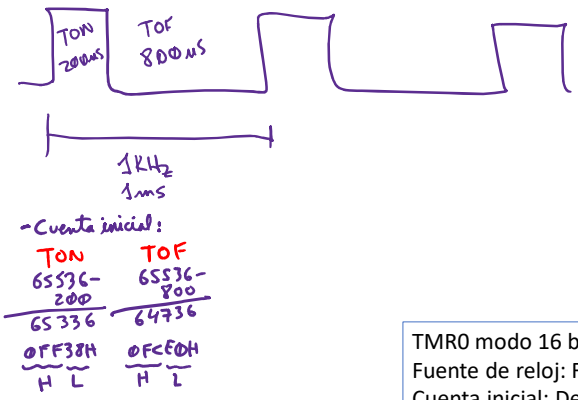
- Cómo generamos una señal cuadrada de 1KHz 50%DC ahora usando el modo 16 bits
  - Pruebas de la generación de onda cuadrada en el circuito implementado usando osciloscopio

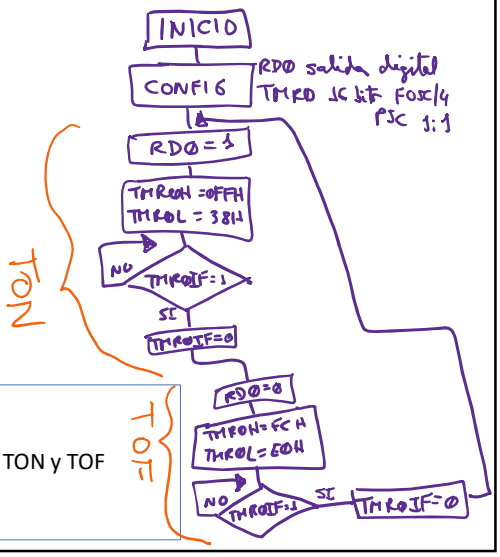



18

# Ejercicios:

- Cómo generamos una señal cuadrada de 1KHz 20%DC ahora usando el modo 16 bits




TMR0 modo 16 bits
Fuente de reloj: FOSC/4
Cuenta inicial: Depende de TON y TOF
T0CON0=90H
T0CON1=40H

19

# Ejercicios:

- Cómo generamos una señal cuadrada de 1KHz 20%DC ahora usando el modo 16 bits

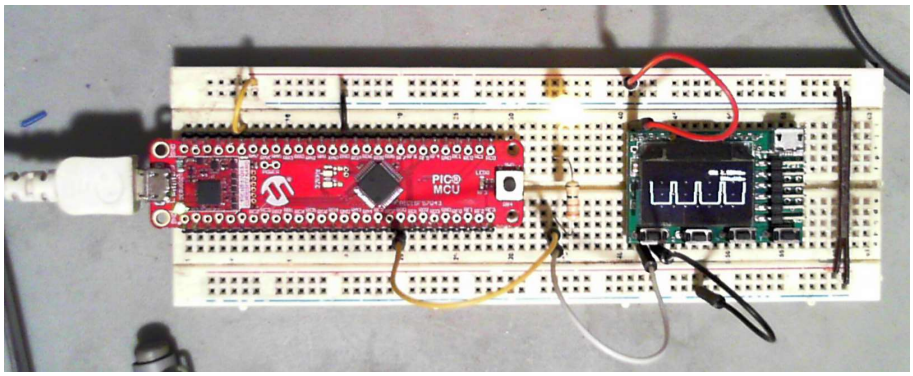
```
1       PROCESSOR 18F57Q43
2       #include "cabecera.inc"
3
4       PSECT upcino, class=CODE, reloc=2, abs
5   upcino:
6       ORG 000000H
7       bra configuro
8
9       ORG 000100H
10  configuro:
11      movlb 0H
12      movlw 60H
13      movwf OSCCON1, 1
14      movlw 02H
15      movwf OSCFRQ, 1
16      movlw 40H
17      movwf OSCEN, 1
18      movlb 3H
19      movlw 90H
20      movwf T0CON0, 1     ;TMR0 enabled, 16bit, postscaler 1:1
21      movlw 40H
22      movwf T0CON1, 1     ;FOSC/4, prescaler 1:1, async
23      movlb 4H
24      bcf TRISD, 0, 1     ;RD0 como salida
25      bcf ANSELD, 0, 1    ;RD0 como digital
```

```
27  inicio:
28      movlb 4H
29      bsf LATD, 0, 1      ;RD0 a uno
30      movlb 3H
31      movlw 0FFH
32      movwf TMR0H, 1
33      movlw 38H
34      movwf TMR0L, 1      ;Precarga de cuenta 65036 a TMR0
35      movlb 4H
36  otro:
37      btfss PIR3, 7, 1    ;Pregunto si TMR0IF=1
38      bra otro            ;Aun no se levanta TMR0IF
39      bcf PIR3, 7, 1      ;Bajamos bandera TMR0IF
40      bcf LATD, 0, 1      ;RD0 a cero
41      movlb 3H
42      movlw 0FCH
43      movwf TMR0H, 1
44      movlw 0E0H
45      movwf TMR0L, 1      ;Precarga de cuenta 64736 a TMR0
46      movlb 4H
47  otro2:
48      btfss PIR3, 7, 1    ;Pregunto si TMR0IF=1
49      bra otro2           ;Aun no se levanta TMR0IF
50      bcf PIR3, 7, 1      ;Bajamos bandera TMR0IF
51      bra inicio
52
53      end upcino
```

20

# Ejercicios:

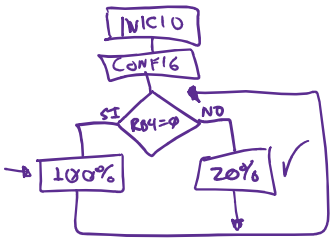
- Cómo generamos una señal cuadrada de 1KHz 20%DC ahora usando el modo 16 bits



21

# ¿Cómo intercambio el duty cycle de 20% a 100% mediante el pulsador en RB4?

- Preguntando al inicio por dicho botón




22

# ¿Cómo intercambio el duty cycle de 20% a 100% mediante el pulsador en RB4?

```
1       PROCESSOR 18F57Q43
2       #include "cabecera.inc"
3
4       PSECT upcino, class=CODE, reloc=2, abs
5   upcino:
6       ORG 000000H
7       bra configuro
8
9       ORG 000080H
10  configuro:
11      movlb 0H
12      movlw 60H
13      movwf OSCCON1, 1
14      movlw 02H
15      movwf OSCFRQ, 1
16      movlw 40H
17      movwf OSCEN, 1
18      movlb 3H
19      movlw 90H
20      movwf T0CON0, 1     ;TMR0 enabled, postsc 1:1 modo 16bit
21      movlw 40H
22      movwf T0CON1, 1     ;FOSC/4 Presc 1:1
23      movlw 0FEH
24      movwf TMR0H, 1
25      movlw 0CH
26      movwf TMR0L, 1      ;cuenta inicial de 65036 al arranque
27      movlb 4H
28      bcf TRISD, 0, 1     ;RD0 como salida
29      bcf ANSELD, 0, 1    ;RD0 como digital
30      bsf TRISB, 4, 1     ;RB4 como entrada
31      bcf ANSELB, 4, 1    ;RB4 como digital
32      bsf WPUB, 4, 1      ;RB4 con pullup activado
```

```
34  inicio:
35      btfsc PORTB, 4, 1   ;preguntamos si presionaste el boton en RB4
36      bra nopresione
37      bsf LATD, 0, 1      ;RD0 en uno
38      bra inicio
39
40  nopresione:
41      bsf LATD, 0, 1      ;RD0 en uno
42      movlb 3H
43      movlw 0FFH
44      movwf TMR0H, 1
45      movlw 38H
46      movwf TMR0L, 1      ;cuenta inicial de 65036 al arranque
47      movlb 4H
48      btfss PIR3, 7, 1    ;pregunto si TMR0IF=1
49      bra $-2             ;falso vuelvo a preguntar
50      bcf PIR3, 7, 1      ;bajo la bandera TMR0IF
51      bcf LATD, 0, 1      ;RD0 en cero
52      movlb 3H
53      movlw 0FCH
54      movwf TMR0H, 1
55      movlw 0E0H
56      movwf TMR0L, 1      ;cuenta inicial de 65036 al arranque
57      movlb 4H
58      btfss PIR3, 7, 1    ;pregunto si TMR0IF=1
59      bra $-2             ;falso vuelvo a preguntar
60      bcf PIR3, 7, 1      ;bajo la bandera TMR0IF
61      bra inicio          ;retorno a inicio
62
63      end upcino
```

23

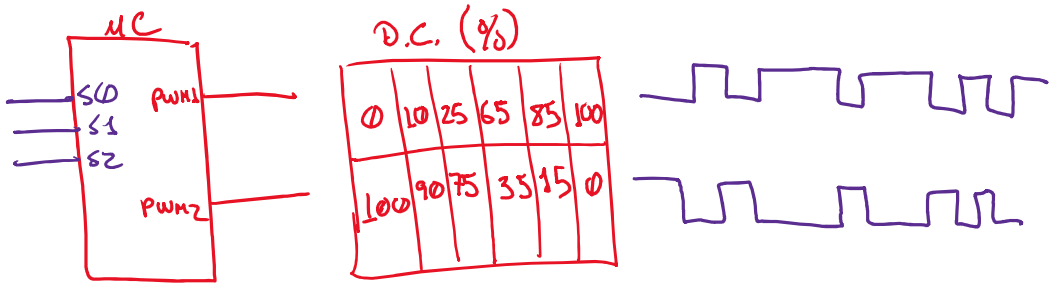
# Observaciones

- No sale exacto en las pruebas, esto debido a que no se ha contemplado el tiempo en que se demora en ejecutar las instrucciones
- Se tiene que hacer una compensación haciendo que el TMR0 cuente menos cuentas (periodo menor de temporizado) y se ajusta empleando nops
- Haciendo este tipo de compensaciones nos acercaremos a la frecuencia solicitada pero no será exacto.
- Esto nos hace pensar que no se va a poder hacer sistemas en tiempo real.

24

# Ejercicios propuestos

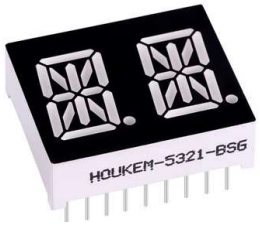
- En un PoR. ¿En qué estado se encuentra el Timer0, encendido o apagado?
- Si Fosc = 24MHz. ¿Cuál es la temporización máxima del Timer0 en modo 16 bits?
- Desarrollar un generador de PWM 2KHz con dos salidas complementarias y con opciones de dutycycle siguientes: 0%, 10%, 25%, 65%, 85% y 100%




25

# Ejercicios propuestos

- Conectar el microcontrolador PIC18F57Q43 con el siguiente dispositivo (display de dos dígitos de 14 segmentos multiplexados de ánodo común):




26

# Fin de la sesión

27