



Universidad Nacional de Lanús

Departamento de Desarrollo Productivo y Tecnológico

**Carrera: Licenciatura en Sistemas**

**Asignatura: Arquitectura de Computadoras**

**Alumno: Ramil Elías**

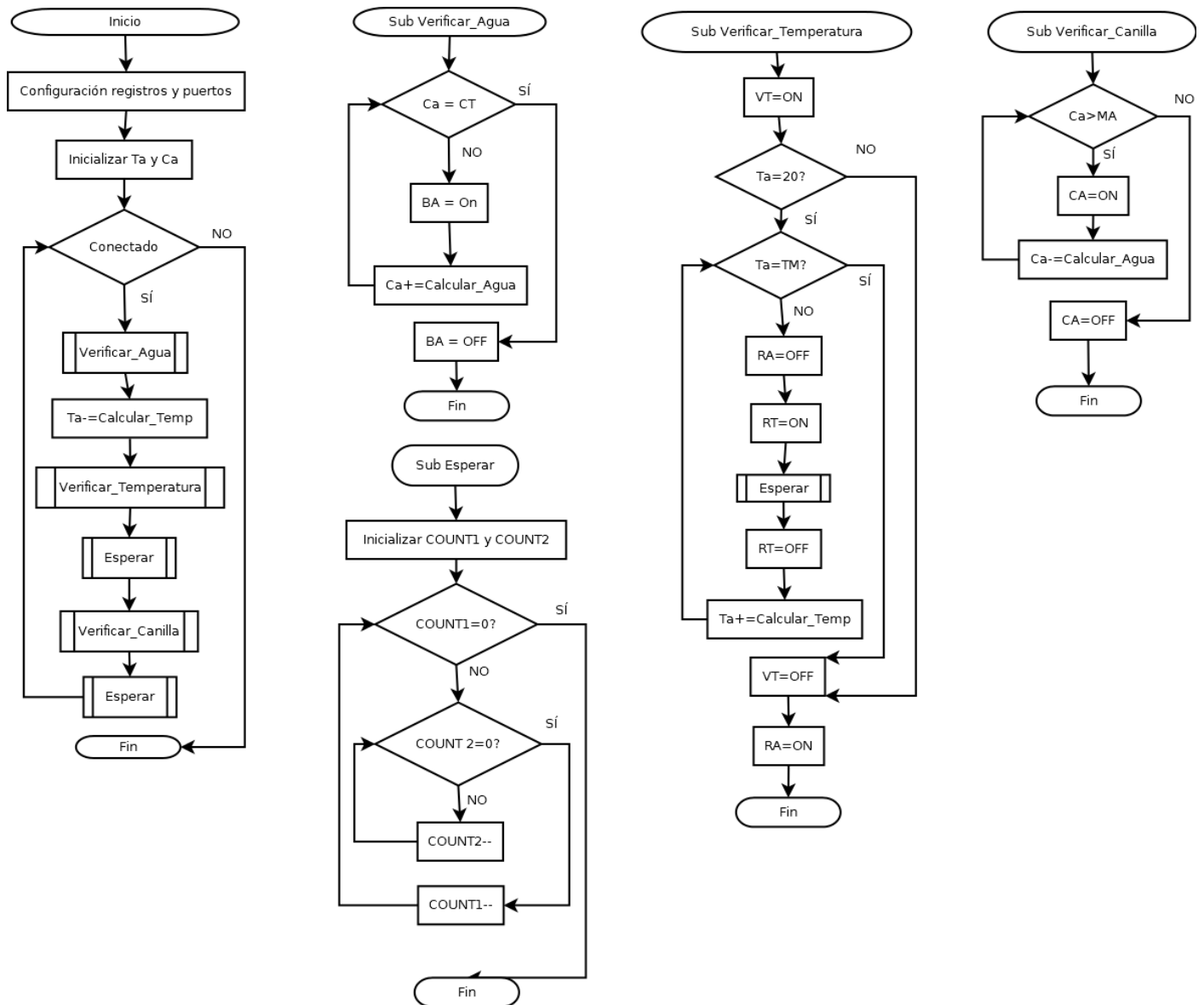
**Docentes: Roberto García**

***Miguel Lanzeni***

**Año: 2023**

**Cuatrimestre: 1º Año - 2º Cuatrimestre**

## Ejercicio 1:



; Configuración del PIC16F628A

list p=16F628A ; Lista de instrucciones del PIC16F628A

#include <P16F628A.INC> ; Archivo de inclusión específico del PIC16F628A

\_\_CONFIG 3F10

; Definición bloque de control

CBLOCK 0x20

COUNT1

COUNT2

ENDC

; Registros:

Ta equ 0x20

; Pos Temperatura actual del agua en °C

Ca equ 0x21

; Pos Cantidad actual del agua en litros

Aux equ 0x22

; Pos Auxiliar

; Constantes

#DEFINE CT d'110'

; Capacidad del Termotanque en litros

#DEFINE Tm d'20'

; Temperatura mínima de trabajo en °C

#DEFINE TM d'45'

; Temperatura máxima donde debe dejar de calentar en °C

#DEFINE MA d'50'

; Min de Agua en el termotanque para cerrar la canilla en litros

```
#DEFINE Calculo_Agua d'10'      ; Para hacer los calculos del agua
#DEFINE Calculo_Temp d'5'       ; Para hacer los calculos de la Temperatura
```

```
; Luces de los leds
```

```
#DEFINE BA 0   ; Azul ---> Bomba de Agua
#DEFINE VT 1   ; Rojo ---> Verificando Temperatura del agua
#DEFINE RT 2   ; Amarillo ---> Resistencia Trabajando
#DEFINE RA 3   ; Verde ---> Resistencia Apagada
#DEFINE CA 4   ; Blanco ---> Canilla Abierta
```

```
ORG 0x00
GOTO Inicio
```

```
; Rutina de interrupción
ORG 0x04
RETFIE
```

```
; Subrutina para esperar 1 microsegundo
Esperar1ms:
```

```
    movlw d'250'
    movwf COUNT1
```

```
loop
    nop
    decfsz COUNT1, 1
    goto loop

    return
```

```
; Subrutina para esperar 250 microsegundos
Esperar250ms:
```

```
    movlw d'250'
    movwf COUNT2
```

```
loop2
    call Esperar1ms
    decfsz COUNT2, 1
    goto loop2

    return
```

```
; Subrutina para verificar si hay agua suficiente en el termotanque
```

```
Verificar_Agua:
```

```
    ; Verificar si el termotanque tiene suficiente agua (si es menor de 110 litros)
```

```
    ; Si es menor, se enciende la bomba (BA) y se espera a que la cantidad de agua alcance el nivel necesario.
```

```
loop_agua
```

```
    movlw CT                      ; Comparo cantidad total de agua con la cantidad actual
    movwf Aux
    movfw Ca
    subwf Aux, w
```

```
    btfsc STATUS, Z              ; Si CT = Ca apago la bomba de agua
    goto tanque_full
```

```
    bsf PORTB, BA                ; Prender bomba de agua
```

```
    movlw Calculo_Agua           ; Aumento la cantidad de agua
    addwf Ca, w
    movwf Ca
```

```
    goto loop_agua
```

```
tanque_full
```

```
    ; Apagar la bomba (BA)
```

bcf PORTB, BA

return ; Fin Verificar\_Agua

;Subrutina para verificar la temperatura del agua

Verificar\_Temperatura:

bsf PORTB, VT ; Led que indica que se esta verificando la temp

movfw Ta ; Comparo temp actual con la temp min del agua

movwf Aux

movlw Tm

subwf Aux, w

btfss STATUS, Z ; Ta es mayor a Tm no hago más nada

goto agua\_caliente

loop\_temp

movlw TM ; Comparo la temp actual con la temp max del agua

movwf Aux

movfw Ta

subwf Aux, w

btfsc STATUS, Z ; Si Ta = TM apago la resistencia

goto agua\_caliente

bcf PORTB, RA ; Prender resistencia

bsf PORTB, RT ; Parpadeo de resistencia trabajando

call Esperar250ms

bcf PORTB, RT

movlw Calculo\_Temp ; Calculo de la nueva Ta

addwf Ta, w

movwf Ta

goto loop\_temp

agua\_caliente

bcf PORTB, VT ; Indicar que finalizo la verificacion de temp

bsf PORTB, RA ; Resistencia apagada

return ; Fin Verificar\_Temperatura

; Subrutina para verificar si la canilla debe abrirse o cerrarse

Verificar\_Canilla:

loop\_canilla

movfw Ca ; Comparar Cant actual con Min de agua

movwf Aux

movlw MA

subwf Aux, w

btfsc STATUS, Z ; Si Ca - MA = 0 cierro la canilla

goto cerrar\_canilla

bsf PORTB, CA ; Abrir canilla

movlw Calculo\_Agua ; A la cantidad actual de agua le resto

subwf Ca, w

movwf Ca

goto loop\_canilla

```

cerrar_canilla
    bcf PORTB, CA          ; Cerrar canilla

    return    ; Fin Verificar_Canilla

Inicio:
    ; Configuración de puertos
    bsf STATUS, RP0       ; Seleccionar el banco 1 de registros
    clrf TRISB            ; Config TRISB
    bcf STATUS, RP0       ; Deseleccionar el banco de registros 1 (volver al banco 0)

    bsf PORTB, RA         ; Resistencia apagada y termotanque prendido

    ; Inicializar valores:
    movlw d'25'           ; Temperatura actual del agua
    movwf Ta
    movlw d'90'           ; Cantidad actual de agua
    movwf Ca

Bucle_Principal
    call Verificar_Agua

    movlw Calculo_Temp     ; Bajar temperatura
    subwf Ta, w
    movwf Ta

    call Verificar_Temperatura

    call Esperar250ms      ; Espera de 1 segundo
    call Esperar250ms
    call Esperar250ms
    call Esperar250ms

    call Verificar_Canilla

    call Esperar250ms      ; Espera de 1 segundo
    call Esperar250ms
    call Esperar250ms
    call Esperar250ms

    goto Bucle_Principal

end                        ; Fin del programa

```

## **Ejercicio 2:**

```

; Configuración del PIC16F628A
    list    p=16F628A     ; Lista de instrucciones del PIC16F628A
    #include <P16F628A.INC> ; Archivo de inclusión específico del PIC16F628A
    __CONFIG 3F10

; Definición de constantes, bloque de control
    CBLOCK 0x20
    COUNT1
    COUNT2
    ENDC
    ORG 0x00
    GOTO Inicio

; Rutina de interrupción
    ORG 0x04
    RETFIE

; Rutina para esperar 1 microsegundo
Esperar1ms:
    movlw d'250'

```

```

        movwf COUNT1
loop
    nop
    decfsz COUNT1, 1
    goto loop
    return
; Rutina para esperar 250 microsegundos
Esperar250ms:
    movlw d'250'
    movwf COUNT2
loop2
    call Esperar1ms
    decfsz COUNT2, 1
    goto loop2
    return
; Programa principal
Inicio:
    ; Configurar puertos
    BSF STATUS, RP0                ; Seleccionar el banco de registros 1
    MOVLW 0x00
    MOVWF TRISB
    BCF STATUS, RP0                ; Deseleccionar el banco de registros 1 (volver al banco 0)
LOOP
    ; Punto 1: Encender todos los leds (RB0, RB1, RB2, RB3)
    MOVLW 0x0F
    MOVWF PORTB
    ; Esperar un segundo
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    ; Punto 2: Encender y apagar todos los leds cada un segundo
    MOVLW 0x00
    MOVWF PORTB
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    MOVLW 0x0F
    MOVWF PORTB
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    MOVLW 0x00
    MOVWF PORTB
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    ; Punto 3: Encender los leds durante un segundo y apagarlos leds durante medio segundo, repetir 4 veces
    MOVLW 0x0F
    MOVWF PORTB
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    CALL Esperar250ms
    MOVLW 0x00
    MOVWF PORTB
    CALL Esperar250ms
    CALL Esperar250ms
    MOVLW 0x0F
    MOVWF PORTB
    CALL Esperar250ms

```

```

CALL Esperar250ms
CALL Esperar250ms
CALL Esperar250ms
MOVLW 0x00
MOVWF PORTB
CALL Esperar250ms
CALL Esperar250ms
MOVLW 0x0F
MOVWF PORTB
CALL Esperar250ms
CALL Esperar250ms
CALL Esperar250ms
CALL Esperar250ms
MOVLW 0x00
MOVWF PORTB
CALL Esperar250ms
CALL Esperar250ms
MOVLW 0x0F
MOVWF PORTB
CALL Esperar250ms
CALL Esperar250ms
CALL Esperar250ms
CALL Esperar250ms
MOVLW 0x00
MOVWF PORTB
CALL Esperar250ms
CALL Esperar250ms
    ; Punto 4: Encender los LEDs de RB0 a RB3 con una demora de 500ms entre ellos
    MOVLW 0x01
    MOVWF PORTB
    CALL Esperar250ms
CALL Esperar250ms
    MOVLW 0x03
    MOVWF PORTB
    CALL Esperar250ms
CALL Esperar250ms
    MOVLW 0x07
    MOVWF PORTB
    CALL Esperar250ms
CALL Esperar250ms
    MOVLW 0x0F
    MOVWF PORTB
    CALL Esperar250ms
CALL Esperar250ms
    ; Punto 5: Apagar los LEDs de RB3 a RB0 con una demora de 500ms entre ellos
    MOVLW 0x07
    MOVWF PORTB
    CALL Esperar250ms
CALL Esperar250ms
    MOVLW 0x03
    MOVWF PORTB
    CALL Esperar250ms
CALL Esperar250ms
    MOVLW 0x01
    MOVWF PORTB
    CALL Esperar250ms
CALL Esperar250ms
    MOVLW 0x00
    MOVWF PORTB
    CALL Esperar250ms
CALL Esperar250ms
GOTO LOOP
END

```

