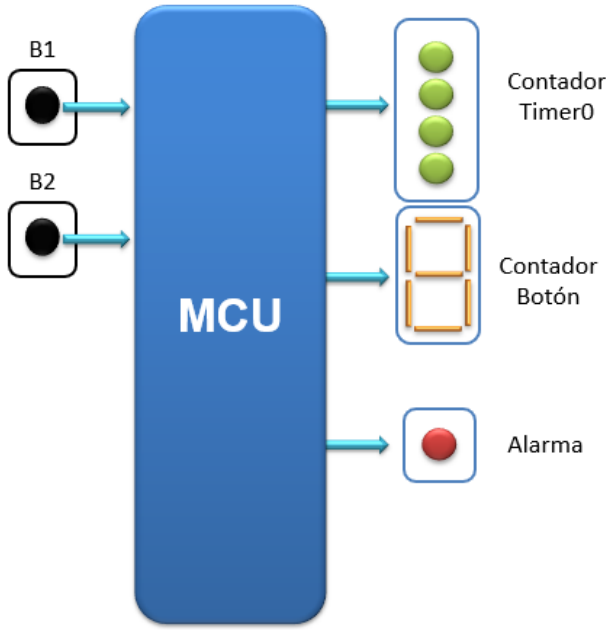
Laboratorio 2. Botones y Timer 0

Circuito Propuesto



Pre Lab (30%)

*Se debe entregar antes del inicio del laboratorio. Se sube en canvas en formato *.zip con el nombre prelab.* ***El circuito debe estar armado antes del laboratorio.***

Diseñe e implemente un contador binario de 4 bits en el que cada incremento se realizará cada 100ms, utilizando el Timer0. Puede escoger el oscilador de su elección. **No utilice interrupciones.**

```
//*****************************************************************************
*******************************************************************
// Universidad del Valle de Guatemala
// IE2023: Programación de Microcontroladores
// Autor: Juan Fer Maldonado
// Contador de 4 bits con display de 7 segmentos.asm
// Descripción: lab2, existe un contador que funciona a 100 ms, además de que hay
un contador de 4 bits y un display de 7 segmentos.
// Hardware: ATMega328P
// Created: 4/02/2024 17:06:34
//*****************************************************************************
*******************************************************************
// Encabezado
//*****************************************************************************
*******************************************************************
.include "M328PDEF.inc"
.cseg
.org 0x00
//*****************************************************************************
*******************************************************************
// Configuración de la Pila
//*****************************************************************************
*******************************************************************
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R17, HIGH(RAMEND)
OUT SPH, R17
//*****************************************************************************
*******************************************************************
// Configuración de MCU
//*****************************************************************************
*******************************************************************
D7Segmentos1: .DB 0b1000000, 0b1111001, 0b0100100, 0b0110000, 0b0011001,
0b0010010, 0b0000010, 0b1111000, 0b0000000, 0b0010000 ; Aqui debe de agregarse los
valores de A-F
D7Segmentos2: .DB 0b0100000, 0b0100000, 0b0000000, 0b0000000, 0b0000000,
0b0000000, 0b0000000, 0b0100000, 0b0000000, 0b0000000 ; Aqui debe de agregarse
cuando se apaga el segmento G EN c.

SETUP:
	// Configuracion de reloj.
	LDI R31,  (1 << CLKPCE)
	STS CLKPR, R31
	LDI R31, 0b0000_0100
	STS CLKPR, R31
	CALL Init_T0		; Configuro el timer 0 a 1024 prescaler.
	//Configuracion de los I/O Ports
	//DDRB SOLO LO USAMOS PARA DECIRLE QUE ES SALIDA O ENTRADA.
	//		76543210
	LDI	R16, 0b00111111	; Estoy asignando que los puertos PB del 5-0 son
Salidas
	OUT	DDRB, R16
```

```
       LDI    R16, 0b00111111   ; Estoy asignando que los puertos PC de 5-0 son
salidas.
       OUT    DDRC, R16
       LDI    R16, 0b00000000   ; Estoy asignando que los puertos PD de 7-0 son
entradas.
       OUT    DDRD, R16
       LDI R17, 0b00001100  //0b00011111  ; Activo que los botones serán pullup,
debido a que los pushbutton están conectados de PC0 - PC4
       OUT PORTD, R17
//**********************************************************************************
**********************************************************************
// Loop infinito
//**********************************************************************************
**********************************************************************
Contador:
       LDI R26, 0           ; Inicializo el contador de la posición R26, posee los
valores para los leds verdes.
       LDI R28, 0                   ; Inicializo el contador de la posición R28,
posee los valores para los leds amarillos.
       LDI R25, 1           ; Inicializo R25 para sumar al contador 1.
       LDI R29, 0           ; Es una bandera que nos servirá para evitar el rebote
en PB1.
       LDI R30, 0                   ; Es una bandera que nos servirá para evitar el
rebote en PB2.
LOOP:
              IN R16, TIFR0      ; Chequear los registros
              CPI R16, (1<<TOV0) ; Comparo si el bit de overflow se enciende.
              BRNE LOOP          ; Si no, regresa a revisar el registro.
              LDI R16, 100       ; Si esta encendido, incializo R16 en 100.
              OUT TCNT0, R16     ; Escribe
              SBI TIFR0, TOV0    ; Apaga el bit de overflow.
              INC R20            ; Incrementa el contador que se usa para el
retardo en milisegundos.
              CPI R20, 1 //2     ; Comparo si este es uno, si es uno, son 100 ms,
de lo contrario, regresa.
              BRNE LOOP
              ADD R28, R25       ; Sumo 1 al contador de leds.
              CPI R28, 16        ; Si el contador es igual a 16, se reinicia el
valor mostrado de los leds.
              BRNE Enciendeleds  ; Muestra el estado actual.
              LDI R28, 0         ; Inicializa el contador de leds.

Enciendeleds:
              OUT PORTB, R28     ; Muestro los leds.
           CLR R20            ; Reinicio el contador de retardo.
              //IN R21, PINC         ; Leo el PINC completo para evaluar si esta
oprimido el boton o no PIN es para leer lo que hay en un puerto.
              //SBRC R21, 0 //1           ; Revisa si el bit 0 tiene 0 y salta
omitiendo la siguiente instruccion si esta asi.
              CALL PresionaPB1   ; Llamo al delay del primer botón solo cuando el
bit 0 esta en 1 (oprimido)
              //IN R21, PINC      ; Leo el PINC cuando el bit 0 esta en 0 y
tambien si esta en 1
```

```
            //SBRS R21, 0               ; Revisa si el bit 0 tiene 1 y salta
omitiendo la siguiente instruccion si esta así.
            CALL IncrementaPB1  ; Llamo a la subfunción de encendido PB1 por que
el bit 0 esta en 0 (Se haya oprimido o no).
            RJMP LOOP
//**************************************************************************
************************************************************
// Subrutina
//**************************************************************************
************************************************************
Init_T0:
      LDI R16, 0        ; Obtenido del video.
      OUT TCCR0A, R16   ; Esto según la Datasheet en 15.9.1, trabaja en modo
normal.
      LDI R16, (1 << CS02) | (1 << CS00)     ; Configuramos el prescaler según
15.9.2,
      OUT TCCR0B, R16
      LDI R16, 100
      OUT TCNT0, R16
      RET
PresionaPB1:
//    LDI R16, 0                  ; Inicializo la posición de memoria R16.
      LDI R29, 1           ; Es una bandera para saber que se oprimió el botón y
debo incrementar el contador.
//RetrasoPB1:
//    INC R16                           ; Inicializo la posición de memoria R16.
//    CPI R16, 100          ; Retardo de 100 ms
//    BRNE RetrasoPB1      ; Mientras no sea 100, regresa a RetrasoPB1
      RET
IncrementaPB1:
    CPI R29, 1                    ; Verificar la bandera que indica que se
detectó que se oprimió el botón.
      BRNE ActivaLedInc     ; Si la bandera indica que no se oprimió, solo muestra
los led.
      LDI R29, 0                  ; Apaga la bandera que indica que se oprimió el
botón.
      CPI R26, 15          ; Si el contador es quince, no sumará.
      //LDI R17, 0b0000000   ; Inicializo PB con 0, debido a que no quiero que se
encienda ningún led azul.
      BREQ ActivaLedInc    ; Se va a mostrar el valor de quince.
      //MOV R26, R28
      LDI R26, 7          ; Aquí irá el contador en sustitución de este comando.
      //INC R26             ; SI la bandera indica que se oprimió y el contador
no es quince, incrementa el contador y muestra los led.
      MOV R16, R26          ; Muevo el valor del display.
      LDI ZH, HIGH(D7Segmentos1 << 1)  ; Defino donde termina el segmento de
datos 1.
      LDI ZL, LOW(D7Segmentos1 << 1)   ; Defino donde inicia el segmento de datos
1.
      ADD ZL, R16                      ; Me desplazo R16 cantidades en el
segmento de datos 1.
      LPM R27, Z                       ; Leo el dato en la posición Z del
segmento de datos 1.
```

```
	MOV R16, R26			; Muevo el valor para el display.
	LDI ZH, HIGH(D7Segmentos2 << 1)	; Defino donde termina el segmento de
datos 2.
	LDI ZL, LOW(D7Segmentos2 << 1)	; Defino donde inicia el segmento de datos
2.
	ADD ZL, R16			; Me desplazo R16 cantidades en el
segmento de datos 2.
	LPM R17, Z			; Leo el dato en la posición Z del
segmento de datos 2.
ActivaLedInc:
	OUT PORTC, R27		; Muestro los segmentos en el display.
	ADD R17, R28		; Mantengo el segmento G en su estado y enciendo
los led dependiendo del contador.
	OUT PORTB, R17		; Muestro el valor seleccionado en el segmento
de datos 2.
	RET
```