Departamento de Electrónica
Laboratorio de Microprocesadores - 86.07

Trabajo Práctico Obligatorio Nº4: Interrupción Externa

Profesor:	Ing. Guillermo Campiglio
Cuatrimestre/Año:	1º/2020
Turno de las clases prácticas	Miércoles
Jefe de trabajos prácticos:	Ing. Pedro Ignacio Martos
Docente guía:	Ing. Fabricio Baglivo, Ing. Fernando Pucci

Autor			Seguimiento del proyecto								
Maximiliano	Porta	98800									

Observaciones:

.....
.....
.....
.....
.....
.....
.....
.....

Fecha de aprobación				Firma J.T.P

10 de julio de 2020

Índice

1. Objetivo

Consiste en aprender mas código del lenguaje ensamblador,aprender a usar las interrupciones del microcontrolador y ver como funcionan, activando salidas o entradas del mismo, también en ver como cambia el circuito con resistencias de PullUp y sin ella.

2. Descripción

2.1. Introducción

Para el desarrollo de este trabajo de laboratorio se utilizo el diagrama de la Figura 1 y el diagrama de la Figura 2.Se observo en la hoja de datos del Arduino la corriente máxima de entrada/salida ($I = 20mA$), usando las leyes de ohm se obtiene la ecuación 1,con una Resistencia de 220Ω y la tensión para un diodo led verde, se obtuvo que la corriente de salida total es $I_{Out} = 10{,}9mA$, por lo cual es menor a la máxima permitida por pin del arduino.
Mediante la programación se busca poder controlar mediante una interrupción externa (INT0-PD2) el encendido de un led (PB1) y en caso de que no se active se mantiene encendido el led (PB0).

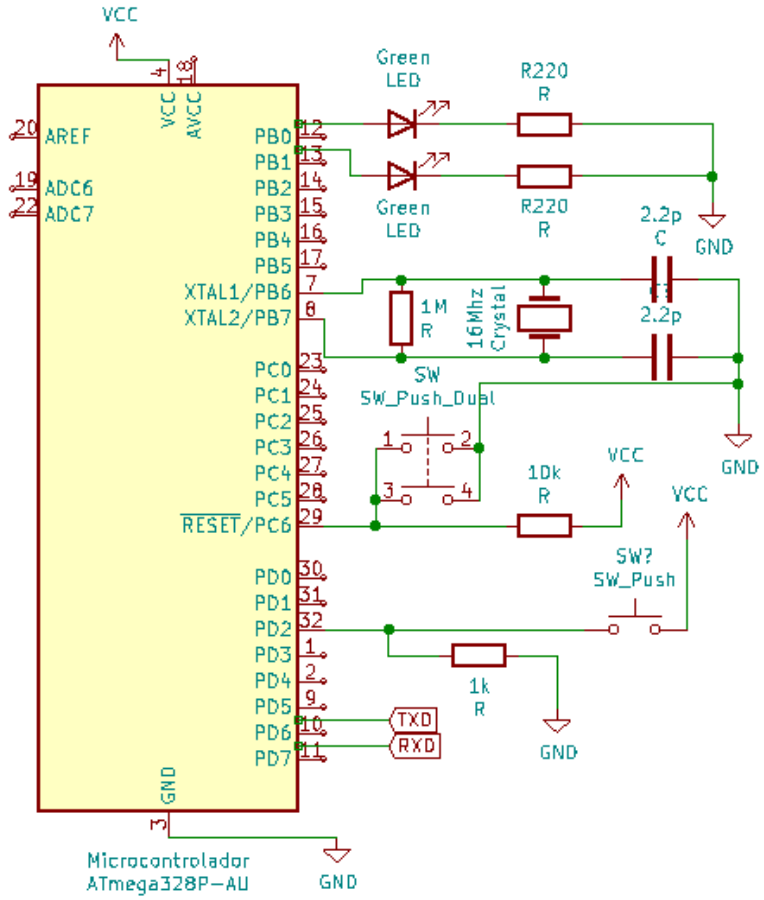

Figura 1: Diagrama de conexión sin resistencia de Pull-Up

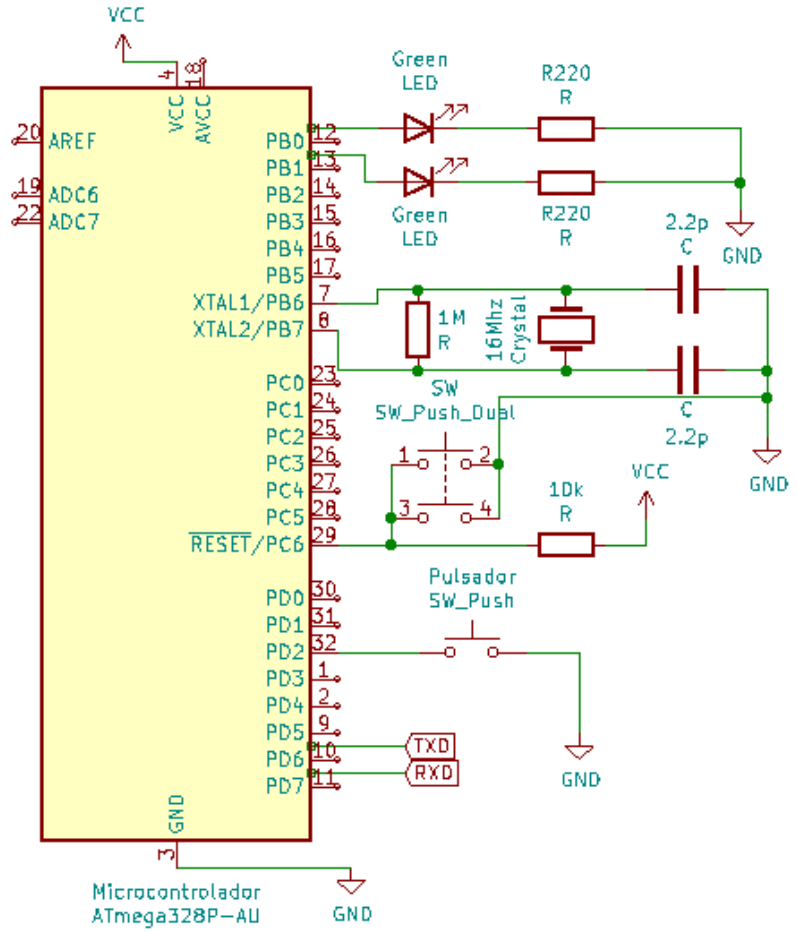

Figura 2: Diagrama de conexión con resistencia de Pull-Up

$$I_{Led} = \frac{V_{CC} - V_{Led}}{R_{Led}} \tag{1}$$

2.2. Desarrollo

Como se observa en la figura 1, en este caso no se usa la resistencia interna de pull-up, por lo tanto se conecta el pulsador a Vcc y se tiene un resistor de $1K\Omega$ para no exceder la corriente máxima de entrada del dispositivo,en este caso entrara un uno lógico . Para la configuración con la resistencia interna de pull-up como se ve en la figura 2 y como se observa en la figura 3, acá la lógica cambia un poco por que la resistencia de pull-up ya esta alimentada con una tensión positiva por por eso para cerrar el circuito tengo que conectar el pulsador a Gnd, su habilitación depende de valor del flip-flop PORTxn, del flip-flop DDRxn y del bit PUD (PUD: Pull-Up Disable). El bit PUD se encuentra en el Registro I/O denominado SFIOR (SFIOR: Special Function I/O Register, Registro I/O de función especial). Con estos 3 bits configurado DDRxn en 0, PORTxn en 1 y PUD en 0. Esto garantiza un 1 lógico cuando el dispositivo está abierto. El otro extremo del botón o interruptor se conecta a tierra, de manera que cuando se cierra el circuito introduce un 0 lógico, el resistor de Pull-Up evita un corto entre el voltaje de alimentación y tierra.

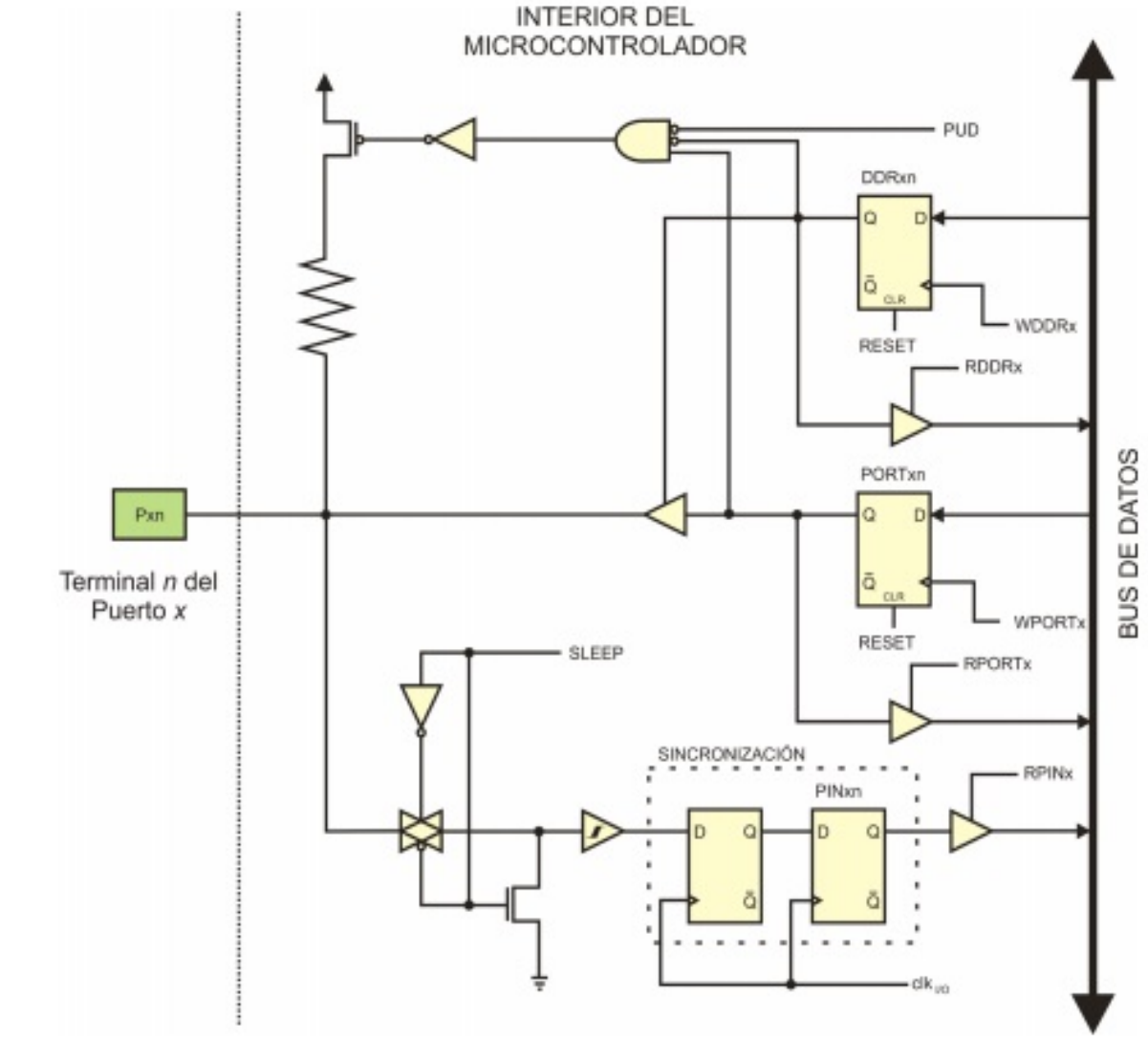

Figura 3: Diagrama de interno del microcontrolador

2.3. Diagrama de Conexión del Circuito Físico

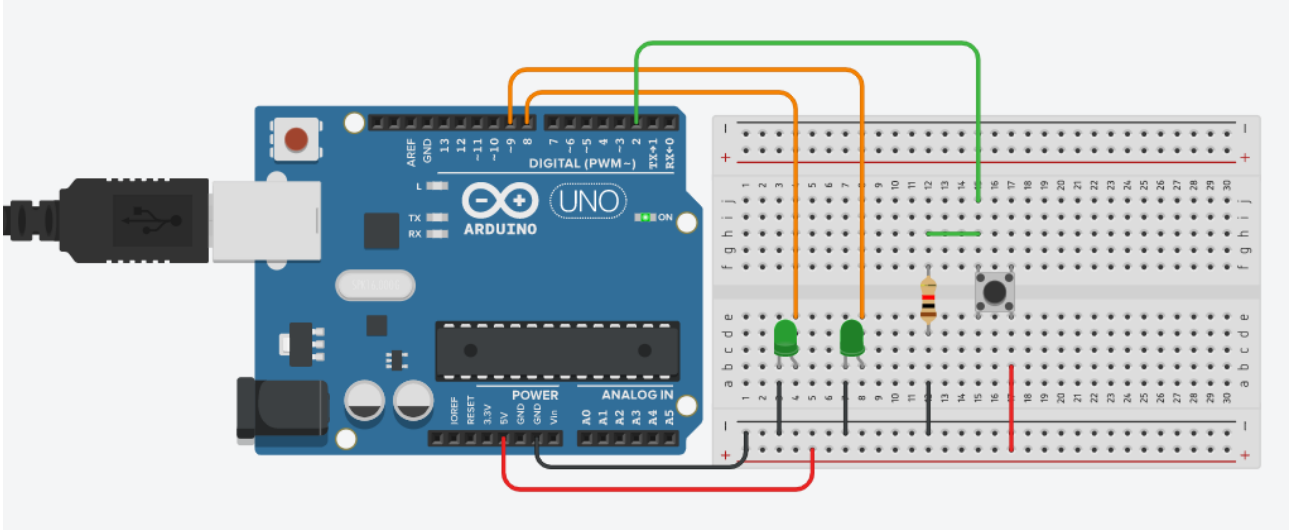

Figura 4: Diagrama del Circuito Sin Resistencia de Pull-Up

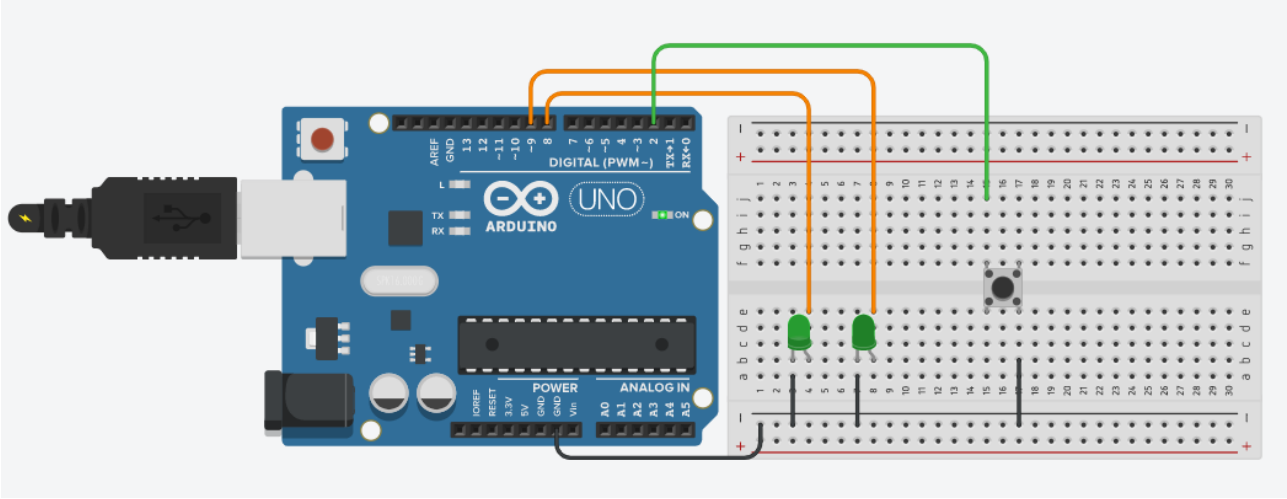

Figura 5: Diagrama del Circuito Con Resistencia de Pull-Up

2.4. Esquema en Bloques

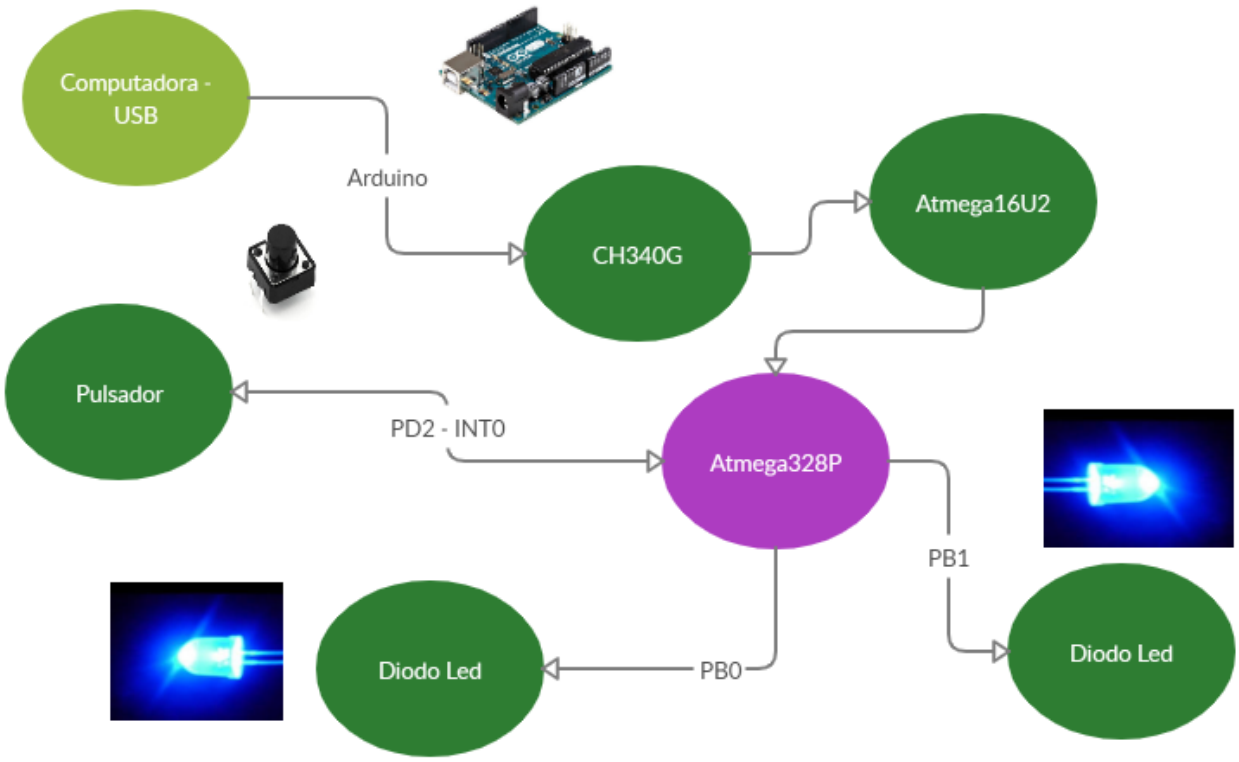

Figura 6: Diagrama en bloques

2.5. Materiales Utilizados

- x1(2.5 Ars)Pulsador de 2 o 4 pines
- x2(14 Ars)Diodo Led verde de 3mm (Alta Luminosidad).
- x1(2.1 Ars)Resistor de 220Ω de carbón 1/4 Watt.
- x1(1.05 Ars)Resistor de $1K\Omega$ de carbón 1/4 Watt.
- 1x(725 Ars)Placa Arduino Uno con USB.
- 1x(96.5 Ars)Placa Experimental.
- 1x(10 Ars)Cables de conexión.
- Costo del Proyecto : 852 Ars

2.6. Diagrama de Flujo

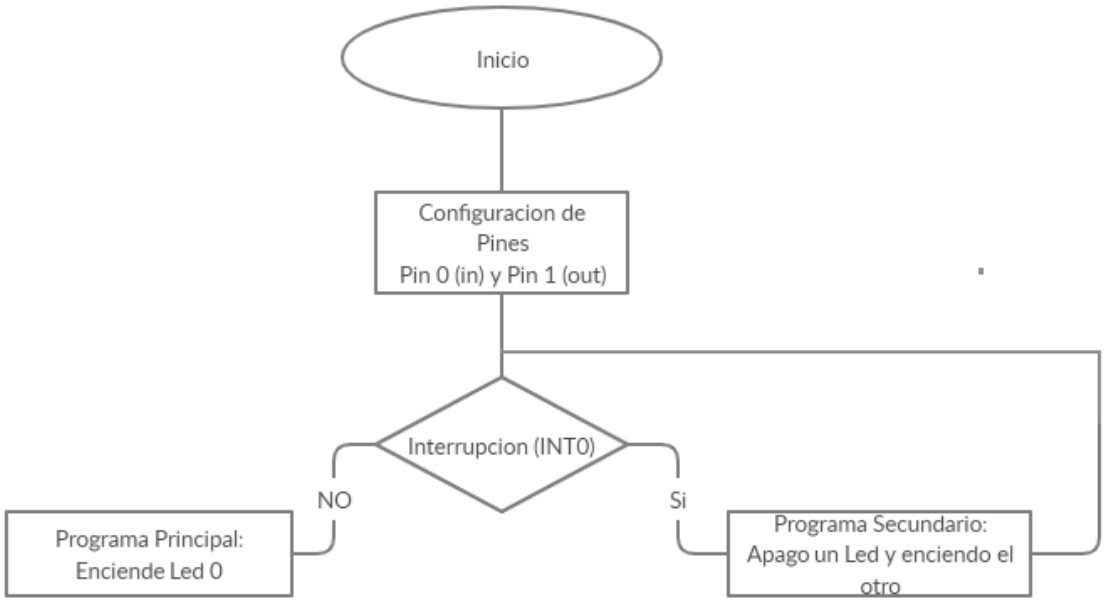

Figura 7: Diagrama de flujo

2.7. Lógica del Programa

Primero habilito las interrupciones externas (INT0) ,luego configuro en el registro de control EICRA, interrupciones por flanco descendente, esto lo logro colocando (1-ISC01 y 0-ISC00), luego tengo dos programas el principal y el de interrupción, el principal se encarga de encender el led en el PB0, luego en el programa de interrupción, activo un contador para que parpadee el led en PB1, y ajusto el delay que desarrolle en el Trabajo Practico 1 para que parpadee a un frecuencia de 1Hz que es lo mismo que 1 segundo, cuando termina,vuelve el programa principal esperando una nueva interrupción.

2.8. Links de funcionamiento del circuito

Sin Resistencia de Pull-Up:`https://youtu.be/AOSQOrvcIIc`
Con Resistencia de Pull-Up:`https://youtu.be/TYdiQNyVOFA`

3. Código

3.1. Código Sin Resistencia de Pull-Up

```
.INCLUDE "m328pdef.inc"

.CSEG
.ORG 0X0000
        JMP CONFIGURACIONES

.ORG INT0addr                   ;ES LA POSICION 0X02
        JMP INTERRUPCION

.ORG INT_VECTORS_SIZE           ;52 WORDS EN 328P

CONFIGURACIONES:
        LDI R16,0XFF            ;CONFIGURO EL PUERTO B
        OUT DDRB,R16            ;CONFIGURO EL PUERTO B COMO SALIDA
        LDI R16,0X00
        OUT PORTB,R16           ;PONGO TODO EN CERO

        LDI R16,(1<<ISC01|0<<ISC00)
        ;CONF INT0 POR FLANCO DESCENDENTE
        STS EICRA,R16           ;SETEO INT0 EN 0X69
        CLR R16
        LDI R16,(1<<INT0)       ;ACTIVO INT0
        OUT EIMSK,R16           ;PASO EL REG A 0X1D
        CLR R16

        SEI                     ;HABILITO INTERRUPCIONES

MAIN:
        SBI PORTB,0             ;ENCIENDE EL LED 0
        JMP MAIN

INTERRUPCION:
        CLI
        CBI PORTB,0                 ;APAGO EL LED 0
        LDI R17,0x05

PARPADEO:
        DEC R17
        SBI PORTB,1
        CALL DELAY
```

```
        CBI PORTB,1
        CALL DELAY
        CPI R17,0
        BRNE PARPADEO

        SBI PORTB,0
        SEI
        RETI

DELAY:                          ;CONFIGURO EL DELAY

        LDI R20, 64
    t3: LDI R19, 250
    t2: LDI R18, 250
    t1: NOP
; hasta aca son 4 ciclos de maquina por iteracion -> 250ns

        DEC R18                 ; 250 x 250ns = 62.5us
        BRNE t1

        DEC R19
        BRNE t2                 ; 250 x 50us = 15.625ms

        DEC R20
        BRNE t3                 ; 64 x 15.625ms = 1s = 1Hz
        ; Con esto logre un retardo de Medio Segundo
        RET
        ; Vuelve a la linea siguiente de la que fue llamado
```

3.2. Código con Resistencia de Pull-Up

```
.INCLUDE "m328pdef.inc"

.CSEG                       ;COMIENZO ESCRITURA EN ROM
.ORG 0X0000                     ;INICIO DE LA ROM
        JMP CONFIGURACIONES

.ORG INT0addr                   ;ES LA POSICION 0X02
        JMP INTERRUPCION

.ORG INT_VECTORS_SIZE           ;52 WORDS EN 328P

CONFIGURACIONES:
        LDI R16,0X00
        OUT DDRD,R16
        LDI R16,0XFF
        OUT PORTD,R16
        ;ACTIVO EL PUERTO D CON RESISTENCIAS DE PULL-UP

        LDI R16,0XFF            ;CONFIGURO EL PUERTO B
        OUT DDRB,R16            ;CONFIGURO EL PUERTO B COMO SALIDA
```

```
        LDI R16,0X00
        OUT PORTB,R16           ;PONGO TODO EN CERO

        LDI R16,(1<<ISC01|0<<ISC00)
        ;CONF INT0 POR FLANCO DESCENDENTE
        STS EICRA,R16           ;SETEO INT0 EN 0X69
        CLR R16
        LDI R16,(1<<INT0)       ;ACTIVO INT0
        OUT EIMSK,R16           ;PASO EL REG A 0X1D
        CLR R16

        SEI                     ;HABILITO INTERRUPCIONES

MAIN:
        SBI PORTB,0             ;ENCIENDE EL LED 0
        JMP MAIN

INTERRUPCION:
        CLI
        CBI PORTB,0             ;APAGO EL LED 0
        LDI R17,0x05

PARPADEO:
        DEC R17
        SBI PORTB,1
        CALL DELAY
        CBI PORTB,1
        CALL DELAY
        CPI R17,0
        BRNE PARPADEO

        SBI PORTB,0
        SEI
        RETI

DELAY:                          ;CONFIGURO EL DELAY

        LDI R20, 64
    t3: LDI R19, 250
    t2: LDI R18, 250
    t1: NOP
; hasta aca son 4 ciclos de maquina por iteracion -> 250ns

        DEC R18                 ; 250 x 250ns = 62.5us
        BRNE t1

        DEC R19
        BRNE t2                 ; 250 x 50us = 15.625ms

        DEC R20
```

```
BRNE t3                    ; 64 x 15.625ms = 1s = 1Hz
; Con esto logre un retardo de Medio Segundo
RET
; Vuelve a la linea siguiente de la que fue llamado
```

4. Resultados y Conclusiones

4.1. Resultados

Se logro controlar mediante interrupciones externas que el microcontrolador cambie de su programa principal a otro según la interrupción que fue activada, una vez que termina de ejecutar esa interrupción, vuelve al programa original.

4.2. Conclusiones

Aprendí a utilizar las interrupciones del microcontrolador y a ver la diferencia entre usar las resistencias de Pull Up o no usarlas, al usarla se evita que haya un posible cortocircuito entre Vcc y Tierra, también tuve el problema que el pulsador tenia un rebote y me ejecutaba dos veces la interrupción, esto lo solucione cambiando de pulsador.