Universidad de Buenos Aires
Facultad De Ingeniería
Año 2021 – 1.er Cuatrimestre

(82.07) Laboratorio de microprocesadores

Entrega N.º 1: Manejo de los puertos
Curso 01
11 de Mayo de 2020

Docentes a cargo:
Stola, Gerardo Luis
Salaya, Juan Guido
Cofman, Fernando

Integrante	Padrón	Correo electrónico
Lützelschwab, Nahila	100686	nlützelschwab@fi.uba.ar

Índice

1. Objetivos del proyecto

Los objetivos del presente trabajo práctico son conocer el correcto manejo de registros de puertos del microcontrolador ATmega 328p y la utilidad de la resistencia pull-up interna de dicho microcontrolador.

Este microcontrolador presenta tres puertos: B (pines digitales del 8 al 13), C (entradas analógicas) y D (pines digitales del 0 al 7). Cada puerto es controlado por tres registros: DDRx, PORTx y PINx. El registro DDRx determina si el pin es una entrada asignando 0, o una salida asignando 1. El registro PORTx es de dirección de datos, es decir, controla si el pin está en nivel alto o bajo. Por último, el registro PINx permite leer el estado de un pin, este es sólo de lectura. Cada bit de estos registros corresponden con un sólo pin.

Para el desarrollo del presente proyecto se hizo uso de una placa Arduino UNO basada en el microcontrolador ATmega 328p.

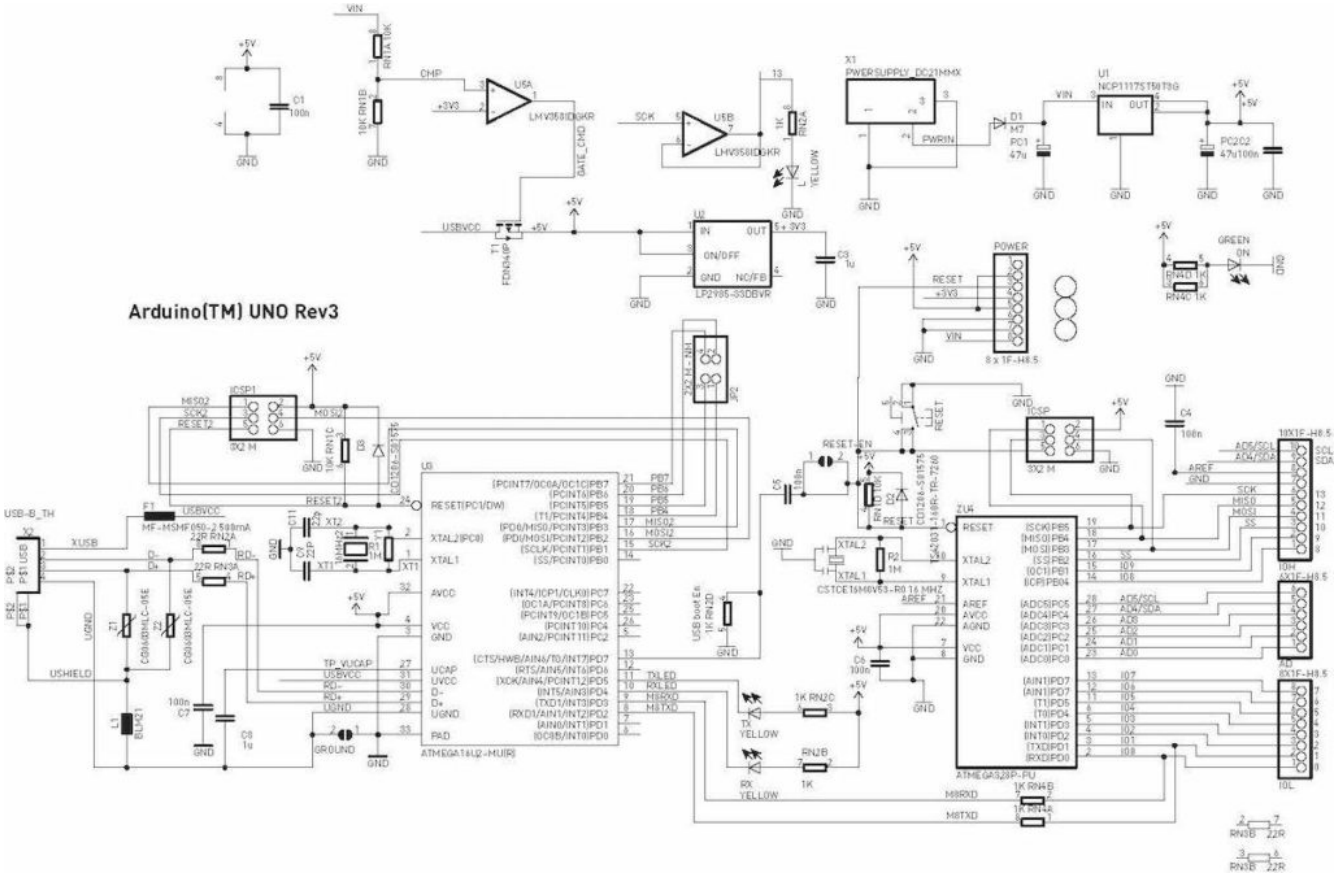

Figura 1: Esquematico completo de Arduino UNO

2. Descripción del proyecto

Para familiarizarse con el manejo de registros de puertos del microcontrolador, se realizó en primera medida un programa que permite parpadear un diodo LED; luego se implementó otro programa que prenda un LED cuando se presiona un pulsador y quede parpadeando hasta que se apague presionando un segundo pulsador. Por último, para conocer la utilidad de la resistencia pull-up interna se modificó el circuito y programa haciendo uso de esta misma.

3. Lista de componentes

A continuación se listan los componentes utilizados para la implementación del proyecto.

Componentes	Precio por unidad($USD)
Placa Arduino UNO	10,33
Protoboard de 830 puntos	3,10
Cables macho-macho para protoboard	0,26
Diodo LED rojo	0,11
2 Pulsadores tact switch de 5mm	0,26
Dos resistencias de valor $10k\Omega$	0,056
Una resistencia de valor 330Ω	0,080
Gasto total aproximado	18,60

Tabla 1: Gasto total en componentes

4. Parpadeo de un diodo LED

Como se mencionó antes, se utilizó una placa Arduino UNO a la cual se le conectó al pin 5 del puerto B un diodo LED rojo. Por lo tanto, en el código implementado se seteó el registro DDR de manera tal que el puerto B se comporte como salida. Para hacer parpadear el diodo, se realizó un ciclo que se repite indefinidamente, en el cual se configura el registro PORTB en nivel alto de manera que el diodo se mantenga encendido durante un intervalo de tiempo de $1s$ y luego se configura en nivel bajo para apagarlo con el mismo tiempo que de encendido.

Circuitalmente, cuando el registro PORTB se encuentra en nivel alto se tiene una tensión de alimentación $V_{CC} = 5V$ y una caida de tensión sobre el diodo LED rojo de $V_D = 1{,}6V$. Para limitar la corriente del diodo se utilizó una resistencia de $R = 330\Omega$, dando como resultado una corriente $I_D = 10{,}3mA$, con una intensidad media bajo las especificaciones del fabricante. En cambio, cuando el registro PORTB se encuentra en nivel bajo, no habrá corriente que circule por el LED, quedando de esta manera apagado.

4.1. Esquemático

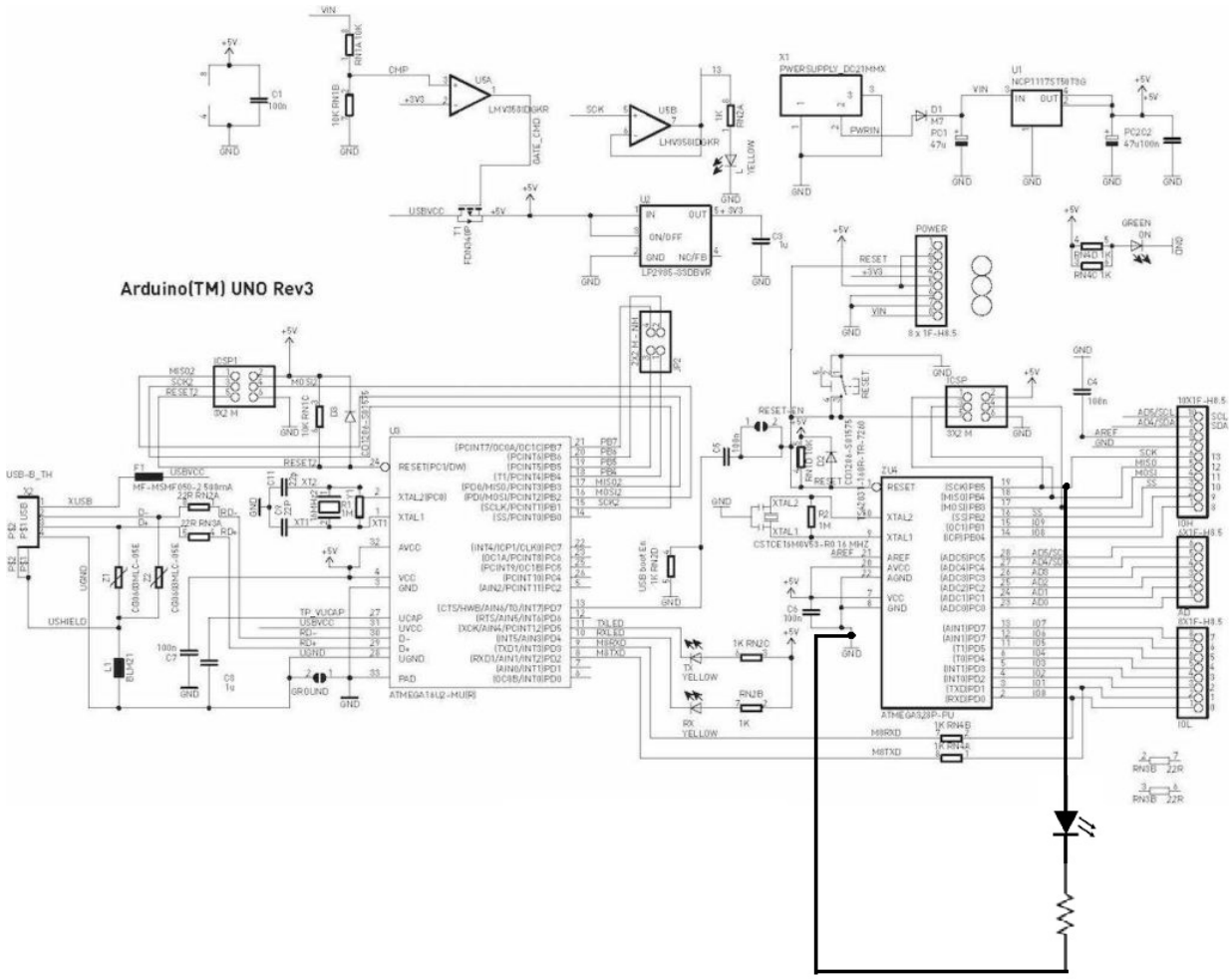

Figura 2: Esquemático parpadeo LED conectado al pin digital 13 del Arduino UNO

4.2. Diagrama de flujo

En la figura **??** se puede observar el diagrama de flujo del algoritmo implementado.

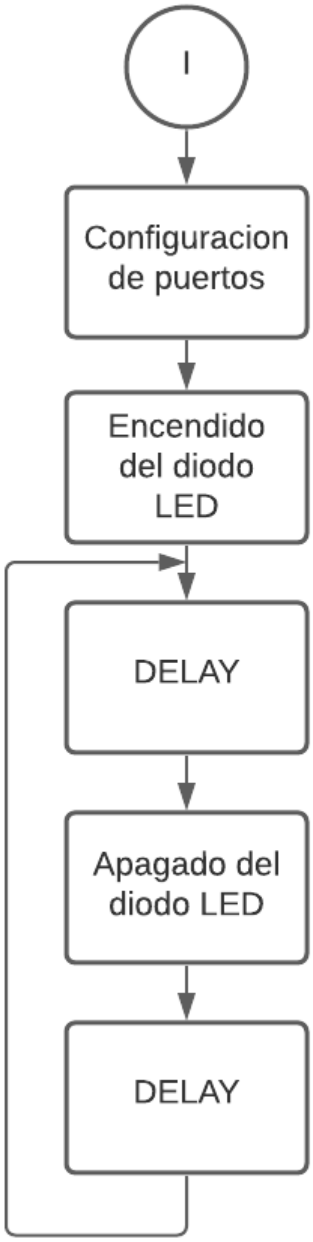

Figura 3: Diagrama de flujo

4.3. Diagrama en bloques

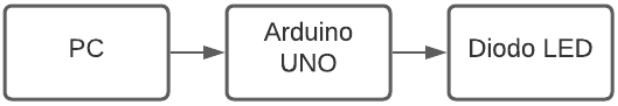

Figura 4: Diagrama en bloques

4.4. Código fuente

Para lograr el retardo entre conmutaciones de niveles se tuvo en cuenta que el Arduino UNO presenta un oscilador de cristal de $16MHz$, siendo de esta manera cada ciclo de 62,5ns.

El código implementado es el siguiente:

```
;
; Laboratorio de microprocesadores (86.07)
; Nahila Lutzelschwab
; Padron : 100686
; 1er cuatrimestre 2021
; Turno Martes 19hs
;


; Defino los registros con nombres significativos
 .DEF C1 = R20
 .DEF C2 = R21
 .DEF C3 = R22


; Codigo para la primera practica

 LDI R16, 0xFF     ; Carga 11111111 en el registro R16
 OUT DDRB, R16     ; Configura el puerto B como salida

 L1:
   SBI PORTB,5   ; Setea el bit 5 del puerto B en 1 (prende LED)
   CALL DELAY    ; Mantiene el led encendido 500ms
   CBI PORTD, 5 ; Setea en el bit 5 del puerto B un 0 (apaga LED)
   CALL DELAY    ; Mantiene el led apagado 500ms
   RJMP L1     ; Repite rutina indefinidamente

;Rutina de retardo de 500ms para 16MHz
 DELAY:
    LDI C1, 41 ; Carga 101001 en C1 (registro R20)
  L2:
    LDI C2, 150 ; Carga 10010110 en C2 (registro R21)
   L3:
    LDI C3, 128 ;Carga 10000000 en C3 (registro R22)
   L4:
     DEC C3
     BRNE L4
     DEC C2
     BRNE L3
     DEC C1
     BRNE L2
   RET
```

Cálculo del tiempo de retardo:

Para obtener un tiempo deseado de aproximadamente $500ms$ para $16MHz$ se realizó una rutina de ciclos anidados. Para ello se tuvo en cuenta que las instrucciones LDI y DEC demoran un ciclo de reloj, mientras que la instrucción BRNE demora dos ciclo excepto en la última realización de cada loop que demora sólo un ciclo. De esta manera, el primer loop (L4) realiza 3 ciclos durante 127 veces y cuando C3 es 0x00 realiza 2 ciclos. Luego, se realiza el segundo loop (L3) de la siguiente manera: se vuelve a realizar el primer loop donde C3 toma el valor de 0XFF y se decrementa hasta 0x00, por lo tanto lo realiza 255 veces con 3 ciclos y cuando C2 llega a 0x00 realiza 2 ciclos. Para el último loop (L4), se realiza de manera semejante al segundo, donde C3 y C2 toman el valor 0xFF y se realizan hasta que C1 toma el valor 0x00.

1+1+1 = 3

(1+2)*127 + (1+1) = 383

(1+2)*255 + (1+1) = 767 *149 = 114.283

(1+2)*149 +(1+1) = 449

((1+2)*255 + (1+1))* 255 * 40 +(1+1) = 7.823.402

(1+2)*40 +(1+1) = 122

Quedando de esta manera una cantidad total de ciclos de 7.938.20, resultando en un retardo de 490ms

5. Parpadeo de un diodo LED con pulsadores

Se realizó un programa que prenda un diodo LED cuando se presiona un pulsador, se mantenga parpadeando y se apague cuando se presiona un segundo pulsador. Se utilizó el diagrama esquemático de la figura **??**. Como se puede observar, el primer pulsador se encuentra conectado con el pin digital 13 del Arduino, es decir, el pin 7 del puerto D; y el segundo pulsador está conectado al pin 0 del puerto B. Cada pulsador se encuentra conectdado desde el otro terminal a una resistencia de $10K\Omega$. El diodo LED rojo se encuentra conectado al pin 4 digital, es decir, el pin 2 del puerto D. Por lo tanto, en el código implementado se seteó el registro DDR de manera tal que el puerto D tenga como salida sólo el pin donde se encuentra conectado el diodo, y el registro DDR del puerto B seteado de manera que se comporte como entrada.

Se implementó una rutina de encendido y apagado de LED y una rutina de retardo. Para la primera se tuvo en cuenta que el pulsador 1 enciende el LED, y mientras que no se apriete el pulsador 2 continuará parpadeando. La rutina de retardo implementada es la misma que para el parpadeo del led sin pulsadores, de $500ms$.

5.1. Esquemático

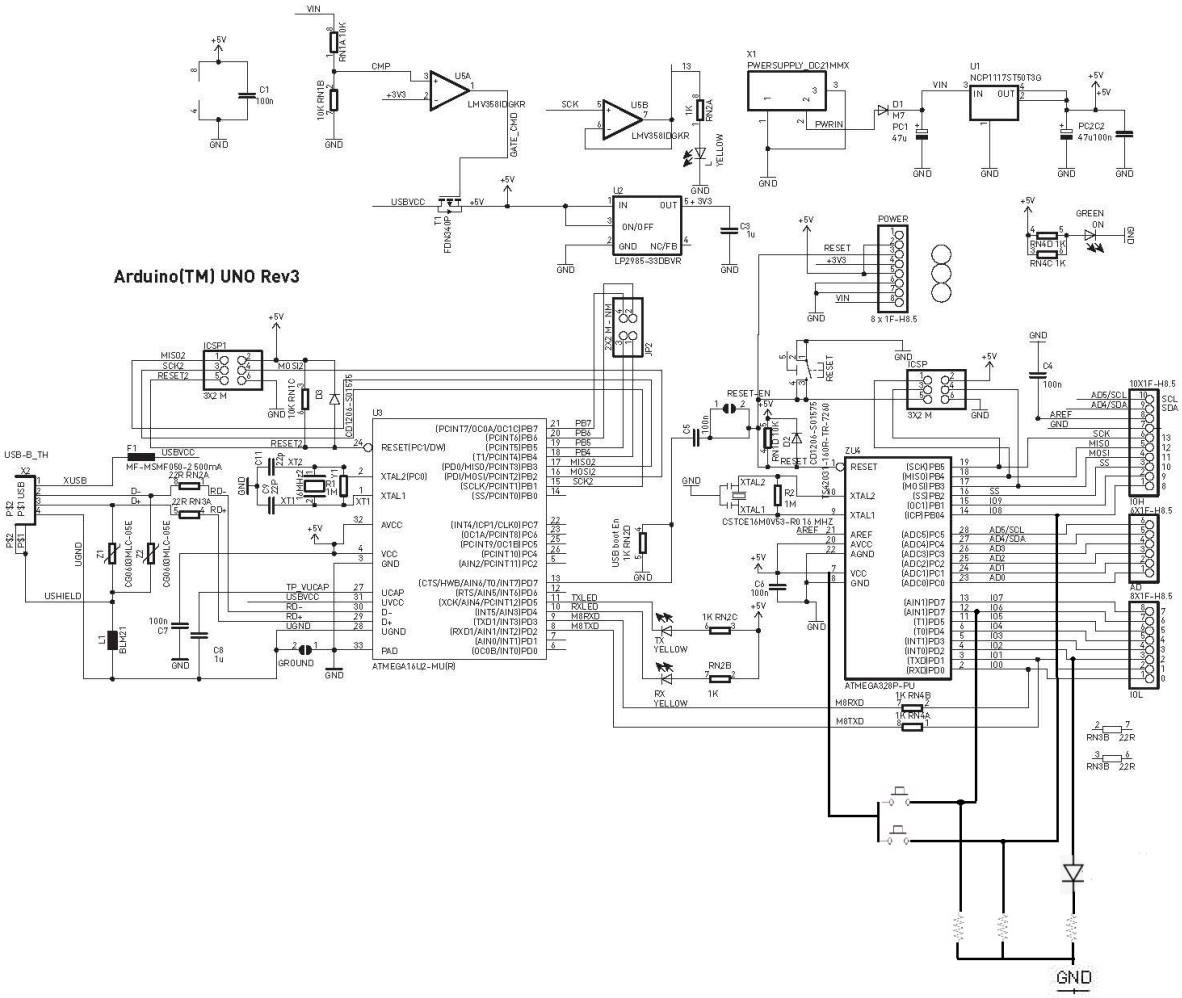

Figura 5: Esquemático parpadeo LED conectado al pin digital 13 del Arduino UNO

5.2. Diagrama de flujo

En la figura **??** se puede observar el diagrama de flujo del algoritmo implementado.

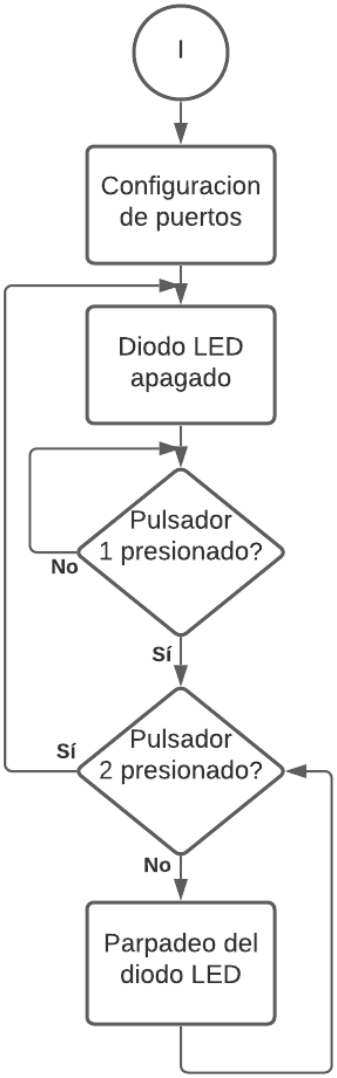

Figura 6: Diagrama de flujo

5.3. Diagrama en bloques

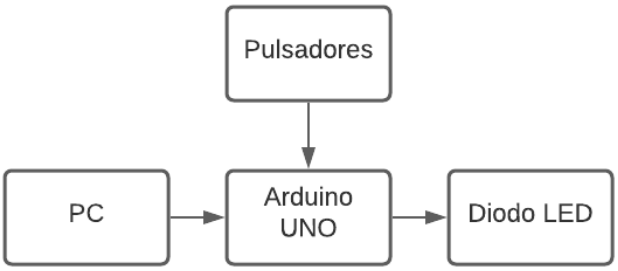

Figura 7: Diagrama en bloques

5.4. Código fuente

El código implementado es el siguiente:

```
1 ;
2 ; Laboratorio de microprocesadores (86.07)
3 ; Nahila Lutzelschwab
```

```
; Padron : 100686
; 1er cuatrimestre 2021
; Turno Martes 19hs
;

    ; Defino los registros con nombres significativos
     .DEF regB = R20
     .DEF regD = R21
     .DEF C1 = R22
     .DEF C2 = R23
     .DEF C3 = R24

   LDI R25, 0x4 ; Carga al registro R25 (el unico bit de salida es el led, bit 2 del puerto D ;y l pulsador 1 es de entrada)
   OUT DDRD, R25  ; Guarda informacion del registro R25 en el registro DDR del puerto D
   LDI R26, 0x0 ; Carga al registro R26 (el puerto B es de entrada, solo esta conectado el pulsador 2)
   OUT DDRB, R26  ; Guarda informacion del registro R26 en el registro DDR del puerto B

; El LED se encuentra apagado hasta que se presione el pulsador 1
 L1:
   IN regB, PINB  ; Lee y carga informacion del registro PIN del puerto B en regB (registro R20)
   SBRS regB, 0 ; Se fija si el bit 0 del registro R20 est  en 1, si lo esta enciende el led y PC ->PC+2 (o 3), sino PC -> PC+1
   RJMP L1    ; Repite rutina indefinidamente

 BLINK:
   IN regD, PIND    ; Lee y carga informaci n del registro PIN del puerto D en regD (registro R21)
   SBRC regD, 7   ; Se fija si el bit 7 del registro R21 est  en 0, si lo est  PC ->PC+2 (o 3), sino significa que el pulsador 2 esta presionado PC -> PC+1 y apaga led
   RJMP LED_OFF   ; Rutina que apaga el led
   SBI PORTD, 2   ; Setea un 1 en el bit 2 del puerto D y enciende el led
   CALL DELAY     ; Mantiene encendido el led
   SBRC regD, 7   ; Se fija si el bit 7 del registro R21 est  en 0, si lo esta PC ->PC+2 (o 3), sino significa que el pulsador 2 esta presionado PC -> PC+1 y apaga led
   RJMP LED_OFF   ; Rutina que apaga el led
   CBI PORTD, 2   ; Setea un 0 en el bit 2 del puerto D y apaga el led
   CALL DELAY     ; Mantiene apagado el led
   RJMP BLINK     ; Repite rutina indefinidamente

 ; Apagado del led (pulsador 2)
 LED_OFF:
  CBI PORTD, 2    ; Setea un 0 en el bit 2 del puerto D y apaga el led

 ; Se mantiene apagado
 LED_STILL_OFF:
 IN regB, PINB    ; Lee y carga informacion del registro PIN del puerto B en regB (registro R20)
 SBRC regB, 0   ; Se fija si el bit 0 del registro R20 est  en 0, si lo esta PC ->PC+2 (o 3) y sigue leyendo el registro PIN del puerto B, sino significa que el pulsador 1 esta presionado PC -> PC+1 y parpadea led
 RJMP BLINK
 RJMP LED_STILL_OFF

;Rutina de retardo de 500ms para 16MHz (misma rutina que parpadeo de led sin pulsadores)
 DELAY:
    LDI C1, 41 ; Carga 101001 en C1 (registro R22)
  L2:
    LDI C2, 150 ; Carga 10010110 en C2 (registro R23)
   L3:
    LDI C3, 128 ;Carga 10000000 en C3 (registro R24)
   L4:
     DEC C3
     BRNE L4
     DEC C2
     BRNE L3
     DEC C1
```

```
63      BRNE L2
64  RET
```

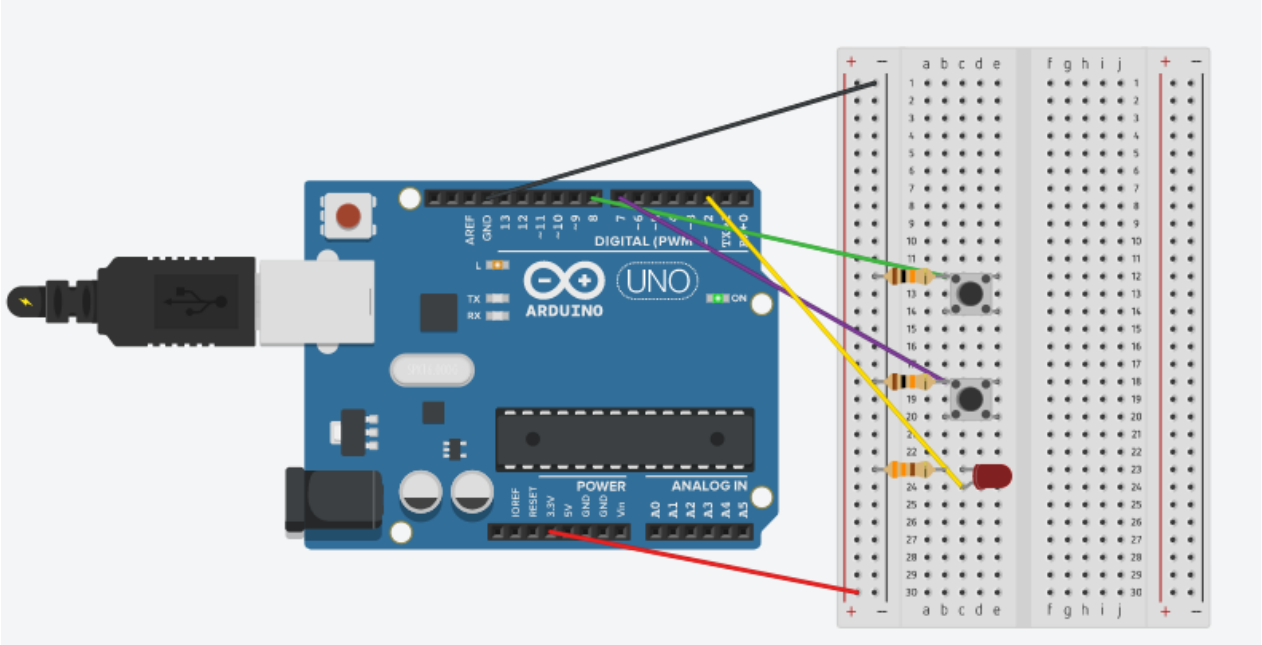

Figura 8: Conexiones parpadeo LED con pulsadores

6. Resistencia pull-up interna

Para conocer la utilidad de la de la resistencia pull-up interna que contiene cada puerto, se implementó un circuito similar al del parpadeo del diodo LED con pulsadores, pero en este caso se reemplazaron las resistencias de $10K\Omega$ conectadas a los pulsadores por las resistencias pull-up. Se utilizó el diagrama esquemático de la figura 12.

Las resistencias pull-up establecen un estado lógico alto en un pin cuando no se presiona el pulsador (su caida de tensión es V_{CC}) y cuando se lo presiona establece un estado lógico bajo. Sabiendo que el microcontrolador presenta dichas resistencias, para habilitarlas se escribió un 1 en el bit correspondiente del registro PORTx.

6.1. Esquemático

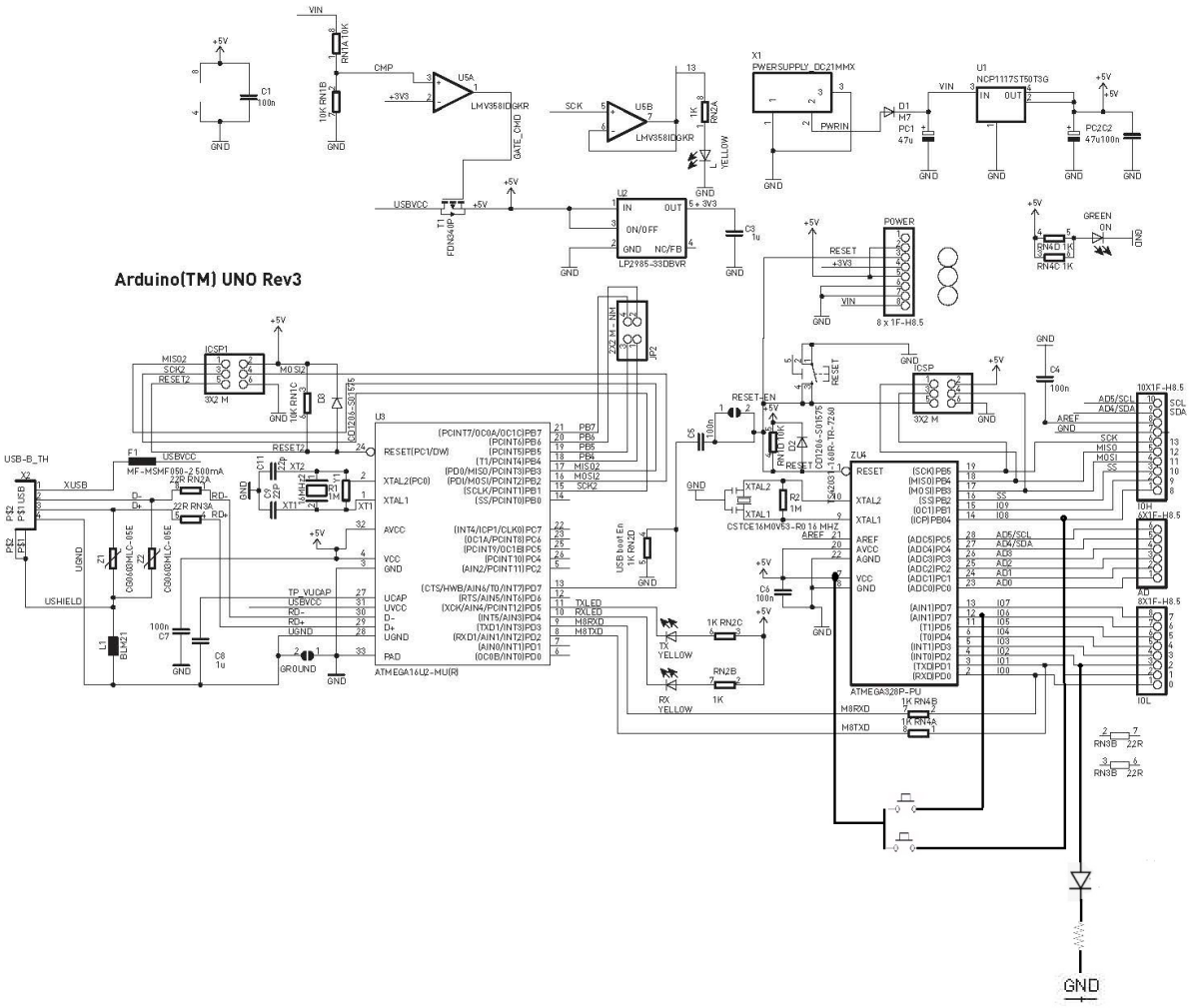

Figura 9: Esquemático del parpadeo LED con pulsadores haciendo uso de las resistencias pull-up internas de los puertos

6.2. Diagrama de flujo

En la figura **??** se puede observar el diagrama de flujo del algoritmo implementado.

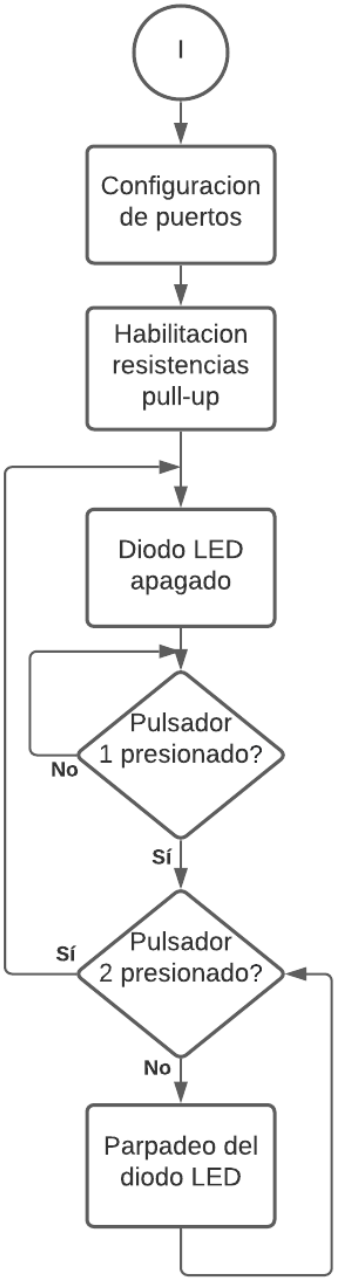

Figura 10: Diagrama de flujo

6.3. Diagrama en bloques

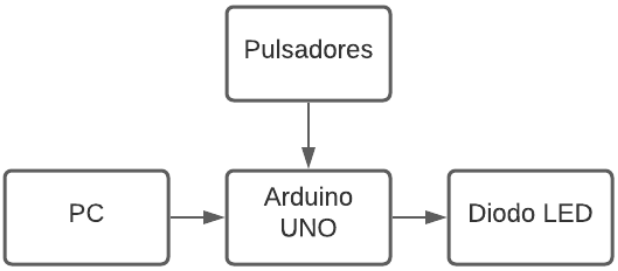

Figura 11: Diagrama en bloques

6.4. Código fuente

El código de los dos pulsadores fue modificado de forma tal que se utilicen las resistencias internas pull-up de los puertos y teniendo en cuenta que cuando se apreta el pulsador da un 0 lógico en vez de un 1 lógico como sucedia en el caso anterior.

```
;
; Laboratorio de microprocesadores (86.07)
; Nahila Lutzelschwab
; Padron : 100686
; 1er cuatrimestre 2021
; Turno Martes 19hs
;


 ; Defino los registros con nombres significativos
     .DEF regB = R20
     .DEF regD = R21
     .DEF C1 = R22
     .DEF C2 = R23
     .DEF C3 = R24


   LDI R25, 0x4 ; Carga al registro R25 (el  nico  bit de salida es el led, bit 2 del puerto D
     ;y l pulsador 1 es de entrada)
   OUT DDRD, R25  ; Guarda informaci n del registro R25 en el registro DDR del puerto D
   LDI R26, 0x0 ; Carga al registro R26 (el puerto B es de entrada, s lo est  conectado el
    pulsador 2)
   OUT DDRB, R26  ; Guarda informaci n del registro R26 en el registro DDR del puerto B

   LDI R25, 0x1 ; Habilita el resistor pull-up del bit 0 del puerto B
   OUT PORTB, R25
   LDI R26, 0x80
   OUT PORTD, R26 ; Habilita el resistor pull-up del bit 7 del puerto D


 L1:
   IN regB, PINB  ; Lee y carga informaci n del registro PIN del puerto B en regB (registro
    R20)
   SBRC regB, 0 ; Se fija si el bit 0 del registro R20 est  en 0, si lo est  PC ->PC+2 (o 3)
     y enciende el led, sino PC -> PC+1
   RJMP L1    ; Repite rutina indefinidamente



 BLINK:
   IN regD, PIND    ; Lee y carga informaci n del registro PIN del puerto B en regB (registro
     R20)
   SBRS regD, 7   ; Se fija si el bit 7 del registro R21 est  en 1, si lo est  PC ->PC+2 (o
    3), sino significa que el pulsador 2 esta presionado PC -> PC+1 y apaga led
   RJMP LED_OFF   ; Rutina que apaga el led
   SBI PORTD, 2   ; Setea un 1 en el bit 2 del puerto D y enciende el led
   CALL DELAY     ; Mantiene encendido el led
   SBRS regD, 7   ; Se fija si el bit 7 del registro R21 est  en 0, si lo est  PC ->PC+2 (o
    3), sino significa que el pulsador 2 esta presionado PC -> PC+1 y apaga led
   RJMP LED_OFF   ; Rutina que apaga el led
   CBI PORTD, 2   ; Setea un 0 en el bit 2 del puerto D y apaga el led
   CALL DELAY     ; Mantiene apagado el led
   RJMP BLINK     ; Repite rutina indefinidamente


   ; Apagado del led (pulsador 2)
 LED_OFF:
  CBI PORTD, 2    ; Setea un 0 en el bit 2 del puerto D y apaga el led


  ; Se mantiene apagado
 LED_STILL_OFF:
```

```
56    IN regB, PINB     ; Lee y carga informaci n del registro PIN del puerto B en regB (registro
        R20)
57    SBRS regB, 0    ; Se fija si el bit 0 del registro R20 est  en 1, si lo est  PC ->PC+2 (o
       3) y sigue leyendo el registro PIN del puerto B, sino significa que el pulsador 1 esta
       presionado PC -> PC+1 y parpadea led
58    RJMP BLINK
59    RJMP LED_STILL_OFF
60
61
62   ;Retardo de 500ms para 16MHz (misma rutina que parpadeo de led sin pulsadores)
63  DELAY:
64     LDI C1, 41 ; Carga 101001 en C1 (registro R22)
65   L2:
66     LDI C2, 150 ; Carga 10010110 en C2 (registro R23)
67    L3:
68     LDI C3, 128 ;Carga 10000000 en C3 (registro R24)
69    L4:
70       DEC C3
71       BRNE L4
72       DEC C2
73       BRNE L3
74       DEC C1
75       BRNE L2
76    RET
```

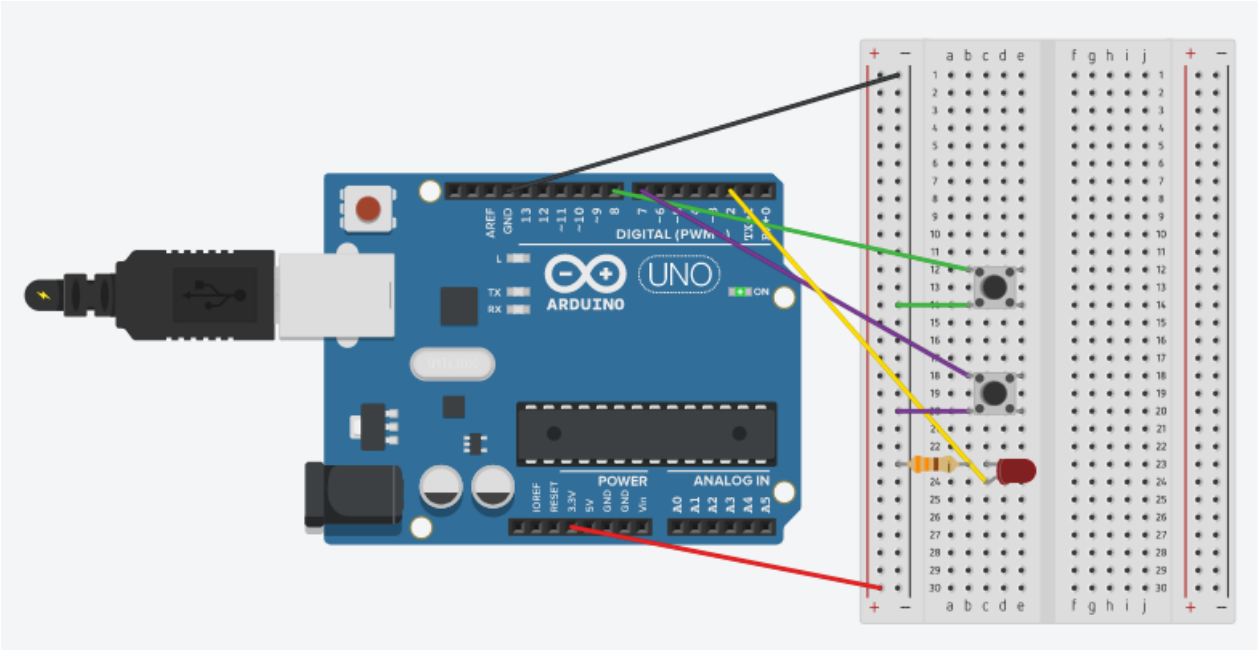

Figura 12: Conexiones del parpadeo LED con pulsadores haciendo uso de las resistencias pull-up internas de los puertos

7. Resultados

`https://youtu.be/biTEQFkXcmg`

8. Conclusiones

Se puede concluir que el trabajo práctico permitió conocer el manejo de registros de puertos de microcontrolador ATmega 328p mediante el uso de la placa Arduino UNO y la utilidad de la resistencia pull-up interna de dicho microcontrolador. Se puede decir que los resultados conseguidos fueron los esperados.

9. Bibliografía

- Mazidi, M. A., Naimi, S., Naimi, S. (2010). AVR Microcontroller and Embedded Systems: Using Assembly and C (Pearson Custom Electronics Technology). New Jersey, United States of America: Pearson
- ATMEGA328P Datasheet (PDF) - ATMEL.http://ww1.microchip.com/downloads/ Corporation. en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P$_{D}atasheet.pdf$