Universidad de Buenos Aires
Facultad de Ingeniería

86.07 - Laboratorio de microprocesadores

TP2: Entradas/Salidas - Interrupciones externas

Berard, Lucia Magdalena
101213 - lberard@fi.uba.ar

1er cuatrimestre de 2021

En el siguiente trabajo práctico se controlará un display de 7 segmentos a partir de pulsadores. Para las entradas se trabajará tanto con lectura programada como con interrupciones. Se buscará comprender las características DC del microcontrolador, analizando los consumos de corriente requeridos y disponibles, de acuerdo a las hojas de datos.

Índice

1. Introducción

1.1. Interrupciones externas

Las interrupciones son un recurso esencial de los sistemas embebidos. Basicamente la interrupción es un mecanismo mediante el cual el programa puede, ante cierto evento, suspender lo que esta haciendo y pasar a atender una rutina de alta prioridad. Una vez finalizada retoma su actividad anterior.

Las interrupciones externas sirven para detectar un estado lógico o un cambio de estado en algunas terminales de entrada del microcontrolador. El ATMega328P tiene dos interrupciones externas, INT0 e INT1:

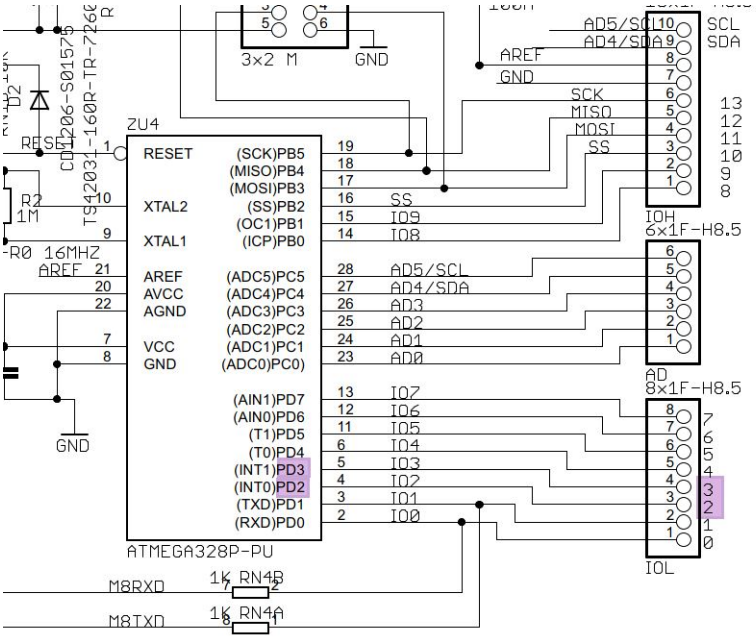

Figura 1: Pines para INT0 e INT1 del Arduino UNO

Se pueden activar por un nivel bajo de voltaje o por flancos de subida o bajada. Para este ejercicio se activara mediante flanco descendente.

Los registros de las interrupciones son:

- **EICRA**: External Interrupts Control Register A. Configura la señal extenra que va a generar la interrupción.
- **EIMSK**: External Interrupts Mask Register. Contiene los bits enable.
- **EIFR**: External Interrupts Flags Register. Contiene los bits de flag

Se utiliza el registro de estado SREG para guardar algún resultado de la interrupción pero no se almacena automáticamente, esto se realiza utilizando las instrucciones PUSH y POP.

1.2. Resistencia de pull down

Cuando se conecta un pin del microcontrolador a un switch, este al presionarlo nos presenta un nivel alto en el pin. Cuando el switch está abierto, la resistencia de pull-down nos da un cero o nivel bajo en el pin del microcontrolador. De esta manera se evita que cuando el pulsador está abierto, el pin , ya sea PD2 como PD3, queda circuitalmente “flotando”, es decir, no es ni un “1” ni un “0” lógico.

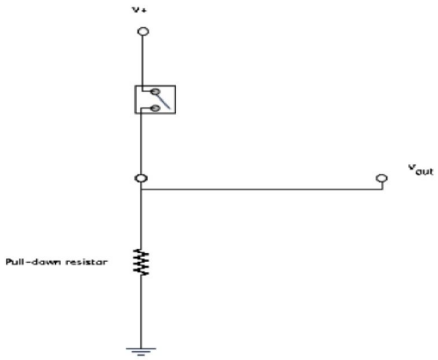

Figura 2: Resistencia de pull-down

1.3. Display de 7 segmentos

Para la visualización de los distintos dígitos se utilizó un display de 7 segmentos cátodo común.

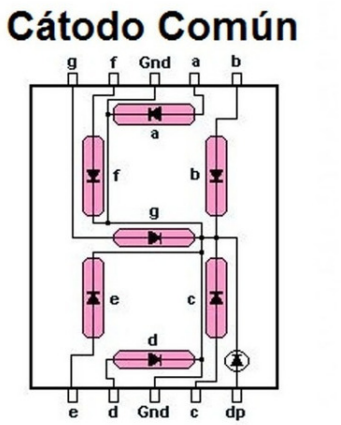

Figura 3: Display 7 segmentos cátodo común

Cuyos valores fueron determinados de la siguiente manera:

Número	g f e d c b a	Hexadecimal
0	0 1 1 1 1 1 1	0x3F
1	0 0 0 0 1 1 0	0x06
2	1 0 1 1 0 1 1	0x5B
3	1 0 0 1 1 1 1	0x4F
4	1 1 0 0 1 1 0	0x66
5	1 1 0 1 1 0 1	0x6D
6	1 1 1 1 1 0 1	0x7D
7	0 0 0 0 1 1 1	0x07
8	1 1 1 1 1 1 1	0x7F
9	1 1 0 1 1 1 1	0x6F

Cuadro 1: Caption

Estos valores se cargaron en una tabla en la memoria del programa de la siguiente manera:

```
DISPLAY_ROM:    .db 0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6f
```

Dado que el puerto D ya esta siendo utilizado por las interrupciones. Se colocaron los pines en el puerto B. Sin embargo el puerto B solo tiene disponibles 6 salidas digitales por lo que el ultimo segmento, 'g', se conecto a PD7 (pin 7 del Arduino).

Se utilizaron los siguientes puertos:

(g,f,e,d,c,b,a) = (PD7,PB5,PB4,PB3,PB2,PB1,PB0)

Esto implica configurar el puerto B y el pin del puerto D como salida. No se puede configurar todo el puerto D como salida ya que se conectaron en PD2 y PD3 las interrupciones que se deben configurar como entradas.

Luego para poder mostrar los dígitos en el display se utilizaron máscaras para leer los valores de la tabla en la memoria del programa y poder separarlos y colocar el valor correspondiente en sus pines.

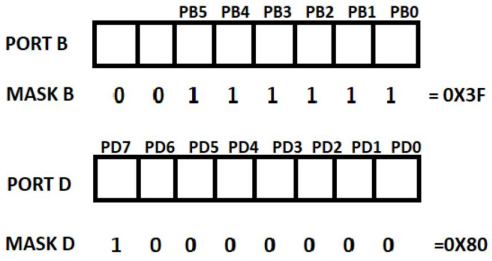

Figura 4: Macros y configuración de puertos

2. Desarrollo

2.1. Conexión y configuración de los puertos

En el siguiente esquema se trata de un display de 7 segmentos de cátodo común por lo que el nodo común está conectado a GND y por lo tanto los LEDs se encenderán sólo al colocar un "1" lógico. La conexión de los pines se realizó utilizando los puertos 4-10 del Arduino (parte alta del puerto D y parte baja del puerto B). Respecto a los pulsadores, estan conectados a los pines PD2 y PD3 (parte baja del puerto D), que en el microcontrolador Atmega328 se vinculan a las interrupciones externas INT0 e INT1. Dado que esta vinculación es fija, la posición de estos pulsadores no puede modificarse.

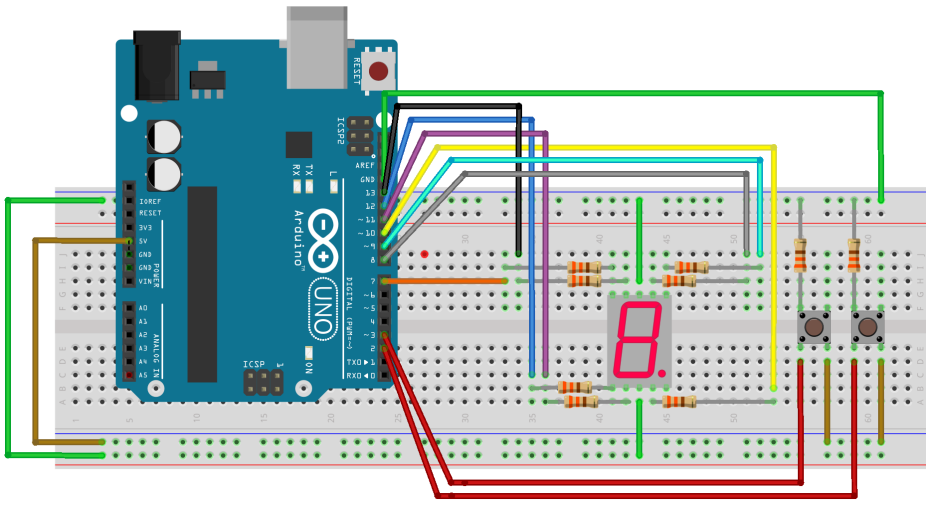

Figura 5: Diagrama esquemático

2.2. Programa

El display inicialmente muestra el dígito "5". Al presionar el pulsador conectado a PD2 se decrementa el contador y al presionar el conectado a PD3, se incrementa.

Se debe evitar que al presionar los pulsadores el display avance más de un dígito. Esta circunstancia indeseada se llama "rebote" y está vinculada al ruido originado en la mecánica del pulsador.

Los valores de los siete segmentos de cada uno de los diez símbolos decimales deberán estar declarados en una tabla a ser almacenada en la memoria del programa.

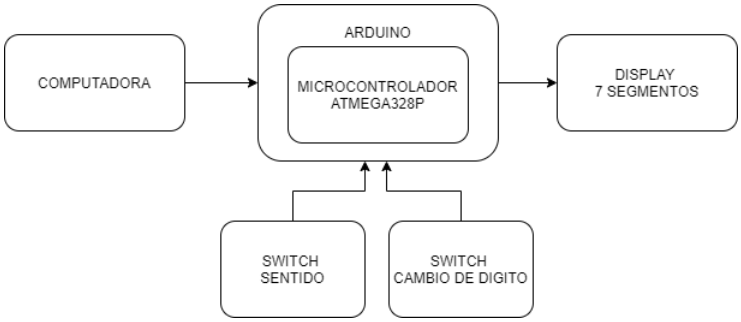

Figura 6: Diagrama en bloques

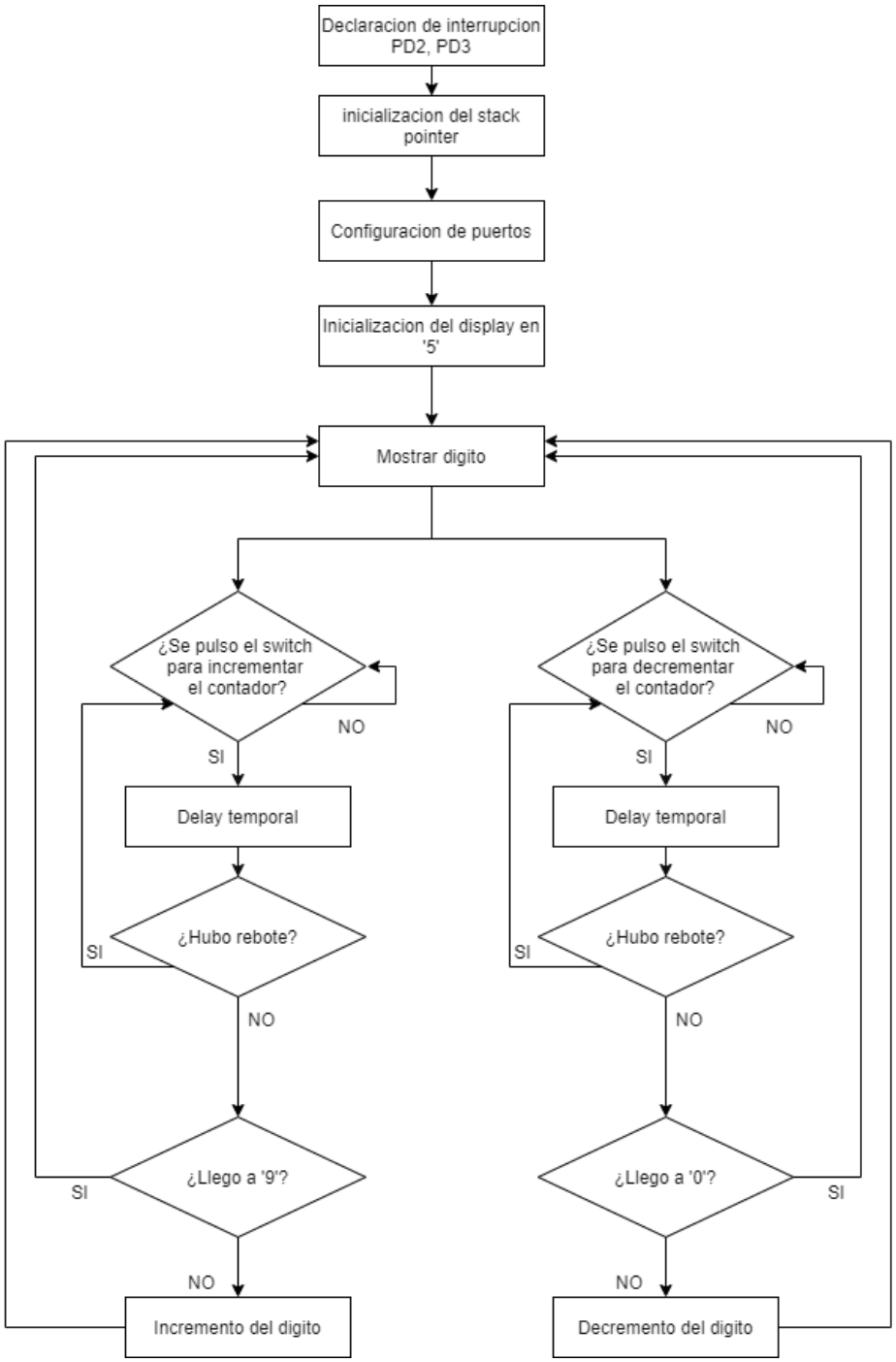

Figura 7: Diagrama de flujo

Para el desarrollo del programa se utilizó un archivo `Macros.inc` donde se declararon las metainstrucciones de assembler para mejorar la modularización y lectura del código principal ubicado en `main.asm`.

Para ello se usaron los siguientes registros auxiliares y valores constantes:

```
;Registro para el contador
.def        counter = R18

;Registros auxiliares
.def        dummyreg1 = R19
.def        dummyreg2 = R20
.def        dummyreg3 = R21

;Registros y mascaras para leer los valores de la tabla
.def        PART_B = R22
.def        PART_D = R23
.equ        MASK_D = 0x80
.equ        MASK_B = 0x3F

;Constantes de configuracion del programa
.equ        MAX_VALUE = 9
.equ        MIN_VALUE = 0
.equ        START_VALUE = 5
.equ        DELAY_TIME = 96
```

Listing 1: Registros y constantes utilizadas

2.2.1. Macros utilizadas

```
;Configuracion resistencias de pull up-------------------------------------------------------
.macro configpullup
                clr                     dummyreg
                ldi                     dummyreg, @0        ;(PORTD pull-ups activados)
                out                     @1,dummyreg
.endmacro

;Configuracion de los puertos ----------------------------------------------------------------
.macro configports
        ;Pongo el puerto D como salida y los PIN INT0 e INT1 como entrada
        ldi             dummyreg1, (1<<DDD7)
        out             DDRD, dummyreg1

        ; Se habilitan las resistencias de pull up de PD2 Y PD3
        ldi             dummyreg1,(1<<PD3)|(1<<PD2)
        out             PORTD,dummyreg1

        ldi             dummyreg1, 0XFF
        out             DDRB, dummyreg1
        ldi             dummyreg1,0
        out             PORTB,dummyreg1
.endmacro

;Configuracion de las interrupciones ---------------------------------------------------------
.macro configINT
                clr             dummyreg1
                ldi             dummyreg1,(1<<ISC01)|(1<<ISC11)
                sts             EICRA,dummyreg1

                ldi             dummyreg1,  (1<<INT0)|(1<<INT1)
                out             EIMSK,dummyreg1

                sei
.endmacro
```

Listing 2: Macros de configuración

```
;Inicializacion del Stack Pointer-----------------------------------------------------------
.macro initSP
        ldi             dummyreg1, HIGH(RAMEND)
        out             SPH, dummyreg1
        ldi             dummyreg1, LOW(RAMEND)
        out             SPL, dummyreg1
.endmacro


; esta macro inicializa el puntero Z a una posicion en ROM-----------------------------------
.macro        initZ
              ldi                       Zh,HIGH(@0<<1)
              ldi                       Zl,LOW(@0<<1)
.endmacro

;Inicializacion del contador en un valor especifico-----------------------------------------
.macro initCont
              clr                       contador
              ldi                       contador,ValorInic
              leerNum         contador
.endmacro
```

Listing 3: Macros para inicializaciones

```
; Inicio del display -------------------------------------------------------------------------
.macro initDisplay
              clr                       counter
              ldi                       counter,START_VALUE
              getNumber       counter
.endmacro

; Borra el display ---------------------------------------------------------------------------
.macro clearDisplay
              ldi                       dummyreg,0x00
              out                       PORTB, dummyreg

              ldi                       dummyreg, 0x00
              out                       PORTD, dummyreg
.endmacro

; Muestra el display -------------------------------------------------------------------------
.macro showDigit
              getNumber       counter
              out                       PORTB,PART_B
              out                       PORTD,PART_D
.endmacro

; Lee valores de la tabla --------------------------------------------------------------------
.macro getNumber
              ;Inicializo puntero y luego lo pongo en la posicion que quiero leer
              initZ             TABLE_DISPLAY
              add                       ZL, @0

              ;Cargo los valores
              lpm                       PART_B,Z
              lpm                       PART_D,Z

              ;Aplico las mascaras
              andi              PART_D, MASK_D
              andi              PART_B, MASK_B
.endmacro
```

Listing 4: Macros para el funcionamiento del display

2.2.2. Código principal

```
.include "m328pdef.inc"
.include "macros.inc"

;--------------------------------------------------------------------
```

```
5   ;                                                   Interrupciones
6   ;-----------------------------------------------------------------------
7
8   .org INT0addr
9           rjmp        INT0action
10  .org INT1addr
11          rjmp        INT1action
12  .cseg
13  .org 0x00
14          jmp                 main
15
16  ;-----------------------------------------------------------------------
17  ;                                                           Main
18  ;-----------------------------------------------------------------------
19
20   .org INT_VECTORS_SIZE
21
22   main:
23          ; Configuro en los puertos
24           configports
25
26          ; Configuro el stack
27          initSP
28
29          ; Configuro las interrupciones externas
30          configINT
31
32          ; Inicializo el display
33          initDisplay
34          showDigit
35
36          ; Loop principal
37          loop:
38                  showDigit
39                  nop
40                  jmp                 loop
41
42  ;-----------------------------------------------------------------------
43  ;                                       Rutina de interrupción - INT0
44  ;-----------------------------------------------------------------------
45  INT0action:
46          rcall        delay
47          rcall        decrement
48  INT0end:
49          reti
50
51  INT1action:
52          rcall        delay
53          rcall        increment
54  INT1end:
55          reti
56
57  ;-----------------------------------------------------------------------
58  ;                                                   Decremento del contador
59  ;-----------------------------------------------------------------------
60  decrement:
61          cpi                 counter,MIN_VALUE
62          breq        return
63          dec                 counter
64  ret
65  ;-----------------------------------------------------------------------
66  ;                                                   Incremento del contador
67  ;-----------------------------------------------------------------------
68  increment:
69          cpi                 counter,MAX_VALUE
70          breq        return
71          inc                 counter
72  ret
73
74  return:
75          ret
76
77  ;-----------------------------------------------------------------------
78  ;                                 Delay temporal para evitar rebotes
79  ;-----------------------------------------------------------------------
80  delay:
```

```
81
82          ldi dummyreg1, DELAY_TIME
83
84          loop1:
85                  sbic        PIND, PD3
86                  jmp                 INT0end
87                  sbic        PIND, PD2
88                  jmp                 INT1end
89                  ldi                 dummyreg2, DELAY_TIME
90
91          loop2:
92                  ldi                 dummyreg3, DELAY_TIME
93
94          loop3:
95                  dec                 dummyreg3
96                  brne        loop3
97                  dec                 dummyreg2
98                  brne        loop2
99                  dec                 dummyreg1
100                 brne        loop1
101                 ret
102 ret
103
104
105 ;-----------------------------------------------------------------------
106 ;                                   Tabla ROM: Display 7 Segmentos
107 ;-----------------------------------------------------------------------
108  .org 0x0100
109
110  TABLE_DISPLAY:
111         .db 0x3F,0x06,0x9B,0x8F,0xA6,0xAD,0xBD,0x07,0xBF,0xA7
```

2.3. Preguntas

- **¿ Si en vez de colocar siete resistores se coloca uno solo en el nodo común, qué ocurre?**
  Al tratarse de un display de 7 segmentos, se recomienda tener los circuitos separados para cada led. Utilizando un solo resistor la corriente va a variar dependiendo del numero que este en el display, dando por ejemplo una diferencia de brillo entre el valor '1' con respecto al '8' o errores en el display de los mismos. Colocando 7 resistencias se asegura que la corriente para cada segmento sea la misma, sin importar el número que se quiera mostrar, evitando posibles errores y diferencias de tension/corriente entre cada segmento.

- **¿ Cuánta corriente puede proveer cada PIN y un puerto completo ? ¿Es la misma corriente que se provee en estado alto que en estado bajo ?**

  La corriente máxima de cada pin es de 40mA pero la suma máxima total de corriente de salida combinada entre todos los pines de Entrada/Salida es de 200 mA. Sin embargo para el estado bajo se recomienda no exceder la corriente más de 100mA y para el estado alto 150mA.

- **¿ Si en el programa se eliminase el manejo por la interrupción del pin PD2 y se quisiera conseguir, no obstante, que el programa siga funcionando de la misma forma, cómo lo modificaría ?**

  En caso de no utilizar el manejo por interrupción se tendría que verificar constantemente si se pulsó el botón para cambiar el dígito y se podría perder el momento en el cual efectivamente se hace. Esto se conoce como método de encuesta o consulta Polling. La ventaja justamente de utilizar el método de interrupción es evitar que se pierdan eventos, tratar determinados eventos con más prioridad que otros y responder a ellos más rápido y sin latencia.

3. Conclusiones

En el desarrollo del trabajo práctico se implementó la utilización de macros, las mismas facilitaron la comprensión y lectura del código principal y resultaron de gran utilidad.

Con respecto al problema en sí, plantearlo mediante interrupciones es beneficioso con respecto al método Polling pero la implementación es más complicada. En principio se tuvo problemas con respecto al conexionado de los switches ya que no se podía obtener las señales, como solución se le agregaron resistencias para mejorar el funcionamiento del circuito y se modificó la configuración de las interrupciones.

En cuanto al algoritmo principal de incrementar o decrementar el contador en un principio se habia implementado con macros y de una manera mas rebuscado. Se simplificó la implementación utilizando rutinas. De esta manera las interrupciones solo se encargan de aumentar o disminuir el contador, para mostrar el digito se hace directamente en el loop principal del main.

4. Anexo

4.1. Repositorio Github

`https://github.com/fiuba-labo-de-micro-miercoles/2021_1c_trabajos_practicos-lmberard/tree/master/TP2`

4.2. Video funcionando

`https://drive.google.com/file/d/1vfZtPwBk2HKssDjtMk0XCI31s4u58eiK/view?usp=sharing`

4.3. Bibliografía

- AVR Datasheet
- AVR Manual de instrucciones
- Teórica 5 - Laboratorio de Microprocesadores (86.07)
- Esquemático Arduino UNO

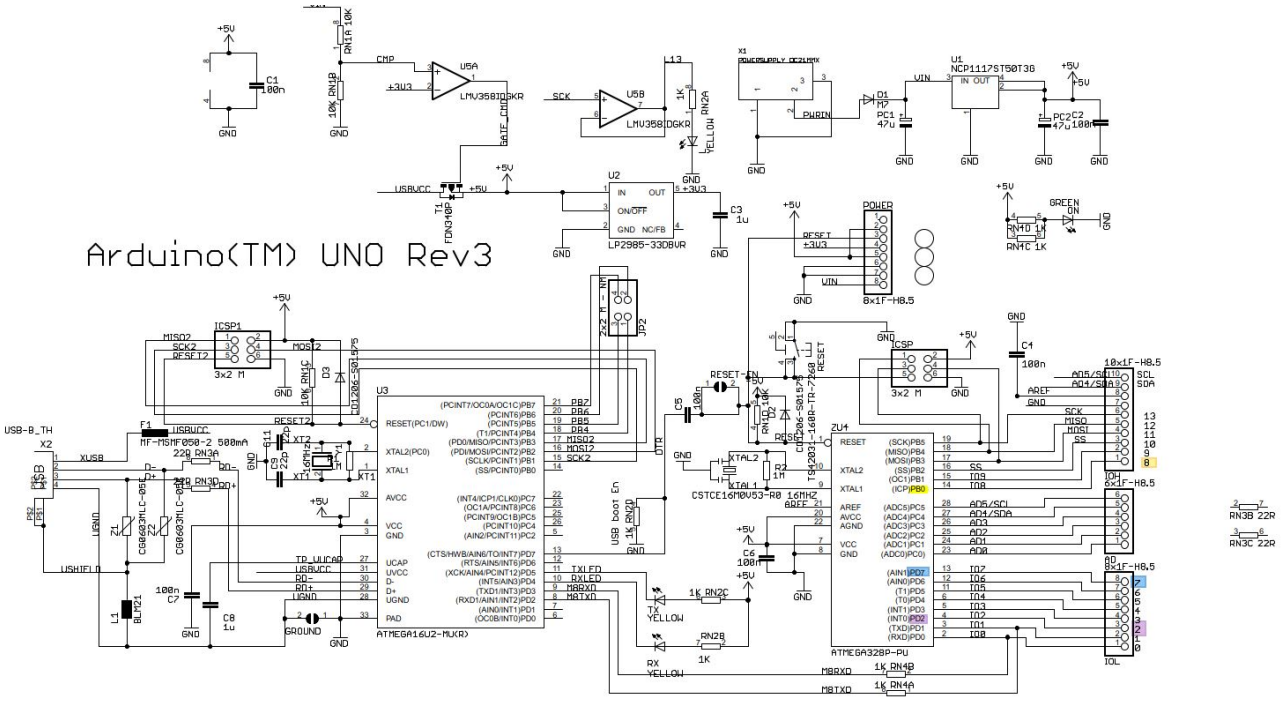

Figura 8: Esquemático Arduino UNO