Universidad de Buenos Aires
Facultad de Ingeniería

86.07 - Laboratorio de microprocesadores

TP3: Puerto Serie

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

1er cuatrimestre de 2021

El siguiente trabajo práctico consiste en establecer comunicación bidireccional serie entre un microcontrolador AVR de 8 bits y una computadora de escritorio.

Índice

1. Introducción

El puerto serie es un periférico que necesita configuración antes de utilizarse. Tiene que haber un acuerdo entre ambos extremos sobre de qué modo se van a serializar los datos. Es decir, como mínimo:

- A qué velocidad transmitir
- De qué modo sincronizamos lo transmitido con lo recibido
- De qué tamaño es la unidad básica de información comunicable, o "carácter"

Para este Trabajo Práctico, se adoptó una velocidad de `9600 bits por segundo` o `baudrate` Esto significa que al dar la orden de transmitir o recibir un carácter, sabemos que los dígitos binarios se van a transmitir a razón de 1/9600 segundos.

Respecto a la sincronización, se adoptó el mecanismo más común, la comunicación asincrónica. Esto significa que no se tendrá una señal de clock que acompañe a los datos, sino una estructura de datos adicional que enmarca la información y permita sincronizar transmisor con receptor.

El enmarcado consiste en que, para cada carácter se antepone un bit de inicio `(start bit)`, que siempre vale "0" y luego de cada carácter se adiciona un bit de fin `(stop bit)` que siempre vale "1" y coincide con el estado en el que queda el canal cuando no se transmite nada.

En lo que respecta al largo del carácter, se adoptaron 8 bits de datos. En la literatura esto se encuentra como 8N1: 8 bits de datos, sin paridad y 1 bit de stop. El concepto de paridad tiene que ver con agregar un noveno bit a modo de redundancia y así detectar algunos errores del lado de receptor. En este Trabajo Práctico no se utilizó bit de paridad.

Si se suma el bit de `start`, los 8 bits de datos y el bit de `stop`, entonces, cada byte de información útil se convierte en 10 bits comunicados. Entonces, el tiempo de transmisión del byte será 10/9600 segundos.

2. Programa

El programa consiste en que al encender el microcontrolador, se deberá transmitir el siguiente texto:

```
*** Hola Labo de Micro ***
Escriba 1,2,3,4 para controlar los LEDs
```

Si en el terminal serie se aprieta la tecla '1', entones se enciende/apaga el LED 1 (toggle). Si se aprieta la tecla '2', ocurre lo propio con el LED 2 y así para los cuatro LEDs.

2.1. Conexión y configuración de los puertos

Se necesitó una placa Arduino UNO, 4 resistencias de 220 Ω, 4 LEDs, un protoboard y cables para el conexionado. Se utilizó el puerto B del microcontrolador ATMega 328P, siendo los puertos 8,9,10 y 11 del Arduino. Se puede observar en la siguiente figura:

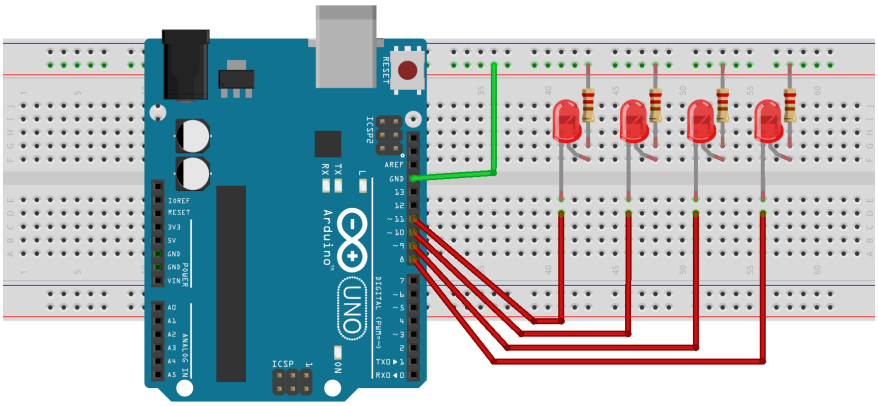

Figura 1: Diagrama esquemático

En este caso se utilizó una placa Arduino Uno (microcontrolador ATmega328p) y la funcionalidad de convertir a USB ya está disponible, dado que se utiliza el mismo hardware para el puerto serie y para reprogramar la placa por lo que no se necesitó comprar un conversor TTL a USB.

2.2. Desarrollo

Para la conexión del puerto serie y poder observarlo en una terminal, se instaló una extensión en el Atmel Studio 7:

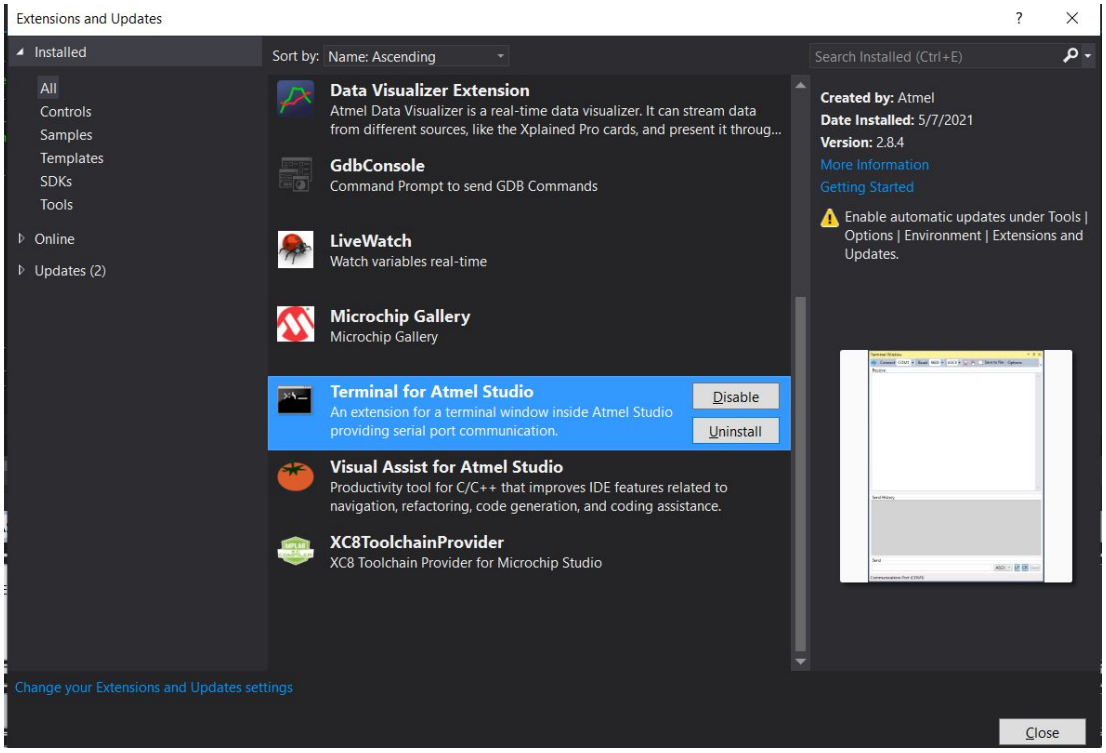

Figura 2: Extensión instalada para la terminal

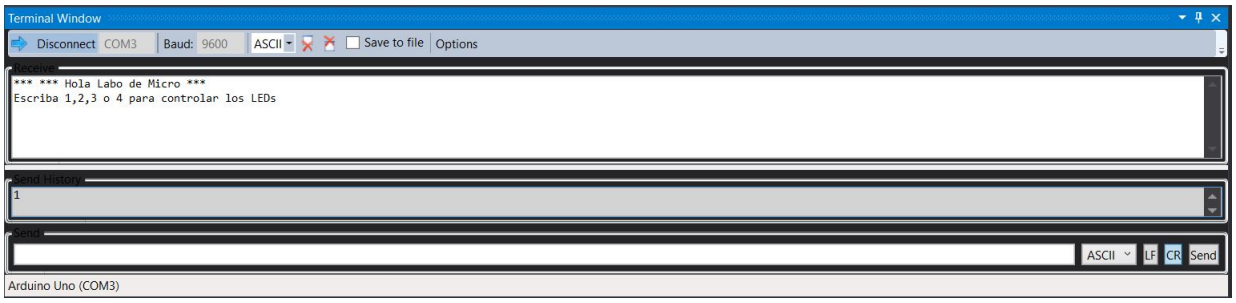

Figura 3: Ejemplo de uso de la terminal

En cuanto a la lógica del algoritmo se planteó la siguiente solución al problema:

Figura 4: Diagrama en bloques

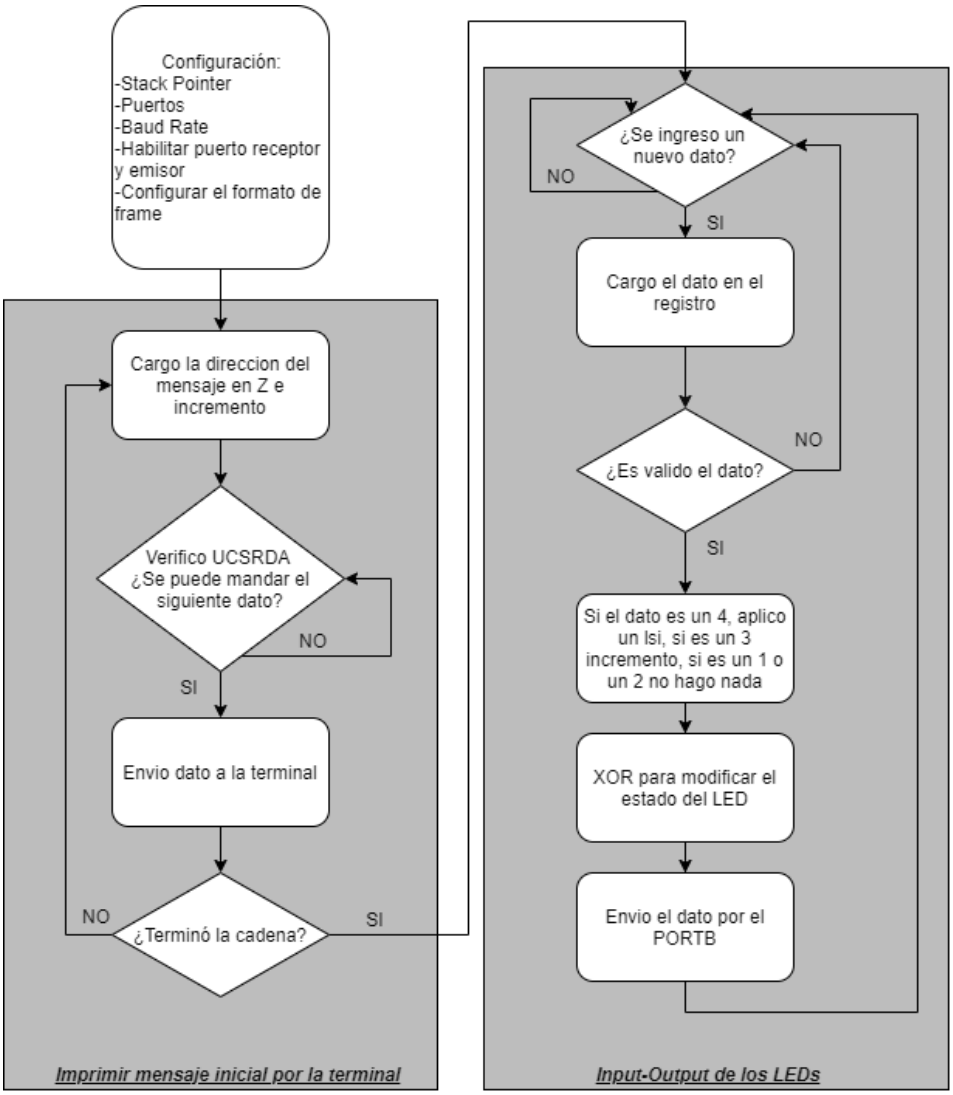

Figura 5: Diagrama de flujo

2.2.1. Macros utilizadas

Se modularizó el código utilizando macros de manera que el módulo principal sea más legible. Las macros se declararon en un archivo `macros.inc`. El resto se encuentra en `main.asm`

```
;Registros auxiliares
.def        aux1 = R17
.def        aux2 = R18
.def        aux3 = R19

;Configuracion Baud Rate
.equ        FOSC = 16*10^6 ;frecuencia de oscilacion 16MHz
.equ        BAUD = 9600
.equ        BPS = 103 ;((FCPU/16/BAUD) - 1)
```

Listing 1: Constantes y registros utilizados.

```
;Configuracion de los puertos ------------------------------------------------------------
.macro configport
        ldi             aux1, @1
        out             @0, aux1
.endmacro

;Configuracion Puerto Serie --------------------------------------------------------------
.macro configUBRR0
        ldi             aux1, 0x00
        sts             UBRR0H, aux1
        ldi             aux1, BPS
        sts             UBRR0L, aux1
.endmacro

.macro configUCSR0B
        ldi             aux1, (1<<RXEN0) | (1<<TXEN0)
        sts             UCSR0B, aux1
.endmacro

.macro configUCSR0C
        ldi             aux1, (1<<UCSZ01) | (1<<UCSZ00)
        sts             UCSR0C, aux1
.endmacro
```

Listing 2: Macros para la configuración de los puertos

```
.macro writeMessage
                        initZ       MENSAJE
                        ldi             aux2, 68

        Znext:          lpm             aux1, Z+

        transmit:   lds             aux3, UCSR0A
                        sbrs        aux3, UDRE0
                        rjmp        transmit
                        sts             UDR0, aux1
                        dec             aux2
                        brne        Znext
        clr             aux2
.endmacro
```

Listing 3: Macro para recibir

```
.macro receive_input

        receive:
                        ;Espero a recibir la info
                        lds             aux3, UCSR0A
                        sbrs        aux3, RXC0

                        ;Recibio data, sale del loop
                        rjmp        receive
                        lds             aux1, UDR0

                        ;Descarto bits con una mascara
                        andi        aux1, 0x0f

                        ;Si no es 1,2,3,4 no hago nada
                        cpi             aux1, 1
                        brlo        receive
                        cpi             aux1, 5
                        brcc        receive

                        ;Si es un 4
                        sbrc        aux1,2
                        lsl             aux1

                        ;Si es un 3
                        cpi             aux1,3
                        brne        not3
                        inc             aux1

                        not3:
                                ;Alto<->Bajo
                                eor             aux2, aux1
                                out             PORTB, aux2
.endmacro
```

Listing 4: Macros para transmitir

2.2.2. Código principal

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

.cseg
.org 0x0000
        jmp         config

.org INT_VECTORS_SIZE

;-----------------------------------------------------------------
;               Configuracion de los puertos
;-----------------------------------------------------------------

config:
        ; Inicializo el stack
        initSP

        ; Declaro PortB como salida (LEDs)
        configport DDRB,0xff

        ; Configuro BAUD RATE
        configUBRR0

        ; Habilito receptor y emisor
        configUCSR0B

        ; 8bit data, 1 stop bit, sin paridad
        configUCSR0C

;-----------------------------------------------------------------
;                       Main
;-----------------------------------------------------------------

main:
```

```
35          ;Escribo el mensaje inicial
36          writeMessage
37
38          ;Espero recibir un mensaje para prender los LEDs
39          loop:
40                  receive_input
41                  rjmp loop
42  ;-----------------------------------------------------------------------
43  ;                       Mensaje
44  ;-----------------------------------------------------------------------
45
46  MENSAJE:
47  .DB '*','*','*',' ','H','o','l','a',' ','L','a','b','o',' ','d','e','',
48      'M','i','c','r','o','','*','*','*','\n','E','s','c','r','i','b','a',' ','1',',','2',',','3',' ','o',
49      ' ','4',' ','p','a','r','a','','c','o','n','t','r','o','l','a','r','',
50      'l','o','s',' ','L','E','D','s'
```

3. Anexo

3.1. Repositorio Github

https://github.com/fiuba-labo-de-micro-miercoles/2021_1c_trabajos_practicos-lmberard/tree/master/TP3

3.2. Video funcionando

https://drive.google.com/file/d/1rgokJ9YrXRmbmJsIcj-VsvH60d5_zmCV/view?usp=sharing

3.3. Bibliografía

- AVR Datasheet
- AVR Manual de instrucciones
- Esquemático Arduino UNO

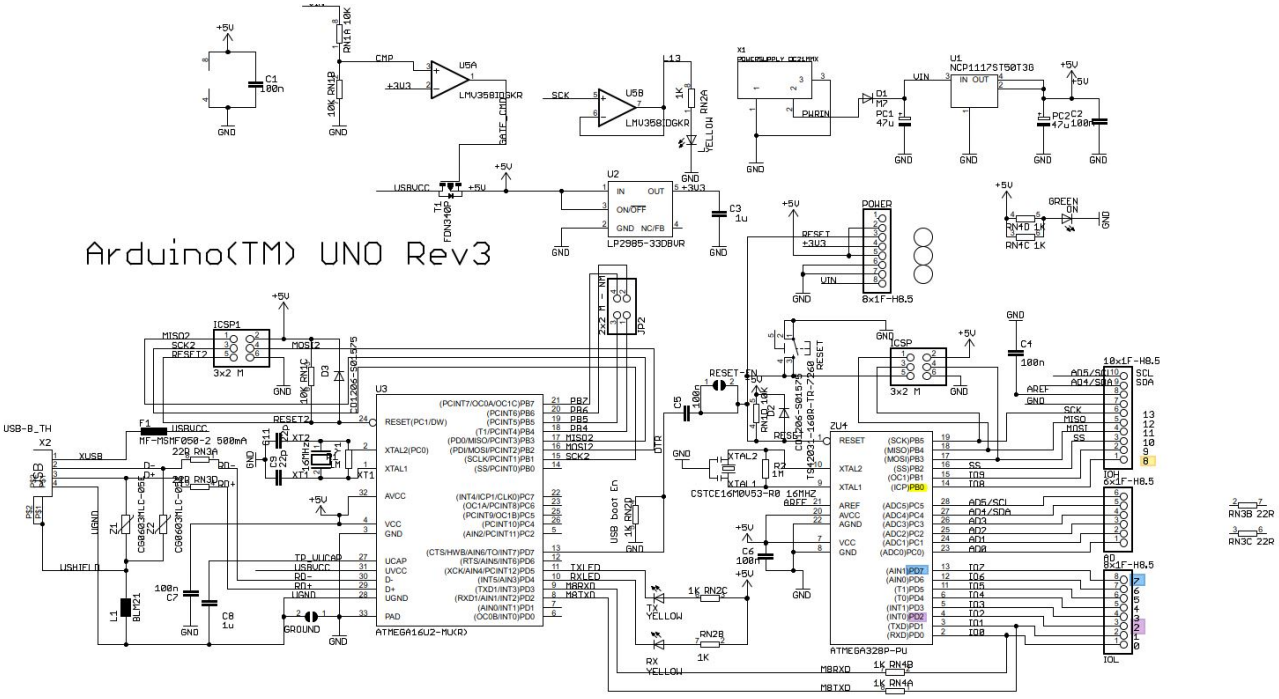

Figura 6: Esquemático Arduino UNO