FACULTAD DE INGENIERIA

Universidad de Buenos Aires

Laboratorio de Microprocesadores - 86.07

Trabajo Práctico N°8:

Puerto Serie

Profesor:			Ing. Guillermo Campiglio								
Cuatrimestre/Año:			1°/2020								
Turno de las clases prácticas			Miércoles								
Jefe de trabajos prácticos:			Ing. Pedro Ignacio Martos								
Docente guía:			-								
Autor			Seguimiento del proyecto								
Francisco	Rossi	99540									

Observaciones:

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

Fecha de aprobación				Firma J.T.P

Coloquio	
Nota final	
Firma profesor	

Índice

1. Introducción **2**
1.1. Objetivo . . . 2
1.2. Descripción . . . 2

2. Materiales **2**

3. Diagrama en Bloques **2**

4. Esquemático **3**

5. Diagrama de flujo **4**
5.1. Polling . . . 4
5.2. Interrupciones . . . 5

6. Código **6**
6.1. UBRR0 . . . 6
6.2. Polling . . . 6
6.3. Interrupciones . . . 9

7. Resultados **11**

8. Conclusiones **11**

1. Introducción

En el siguiente informe se explica el diseño de un programa escrito en lenguaje Assembler el cual establece una comunicación bidireccional serie entre un microcontrolador `ATMEGA328P` y una computadora utilizando el protocolo USART.

1.1. Objetivo

El objetivo es utilizar el puerto serie del micro controlador y el conversor de puerto serie a USB del arduino para comunicar ambos dispositivos.

1.2. Descripción

Se conectan ambos dispositivos y el microcontrolador imprime en la terminal del puerto serie un mensaje de bienvenida, dando opciones de cuatro acciones distintas sobre cuatro luces `LEDs` conectadas al mismo, cada vez que se ingrese '1','2','3' o '4', desde la consola de la computadora cada uno de los cuatro LEDs, respectivamente, conectados al puerto B permutaran su estado teniendo en cuenta su estado actual: de encendido a apagado o de apagado a encendido.

2. Materiales

Se utilizaron los siguiente materiales para el proyecto:

a. 4 LEDs (80$ (Pesos Argentinos))

b. 4 Resistores de 220 Ω (16$ (Pesos Argentinos))

c. 1 Microcontrolador ATmega328p (Utilizando el integrado con el Arduino Uno) (700$ (Pesos Argentinos))

3. Diagrama en Bloques

En la **Fig. 1** se muestra un diagrama en bloques del circuito.

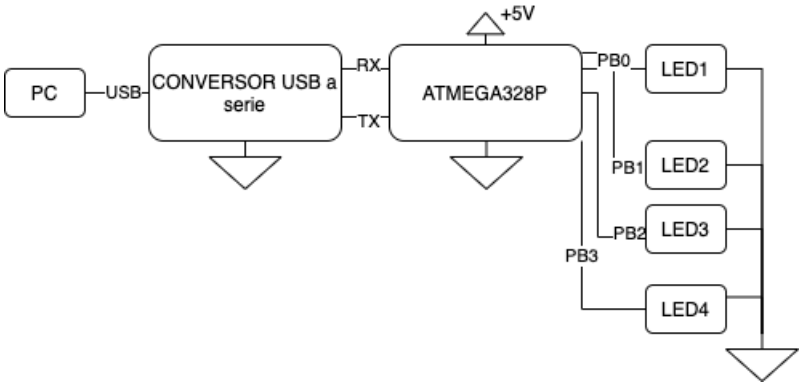

Figura 1: Diagrama en bloques.

4. Esquemático

En las **Fig. 3** y **Fig. 2** se muestra como se conectó el arduino con el LED, y los switches.

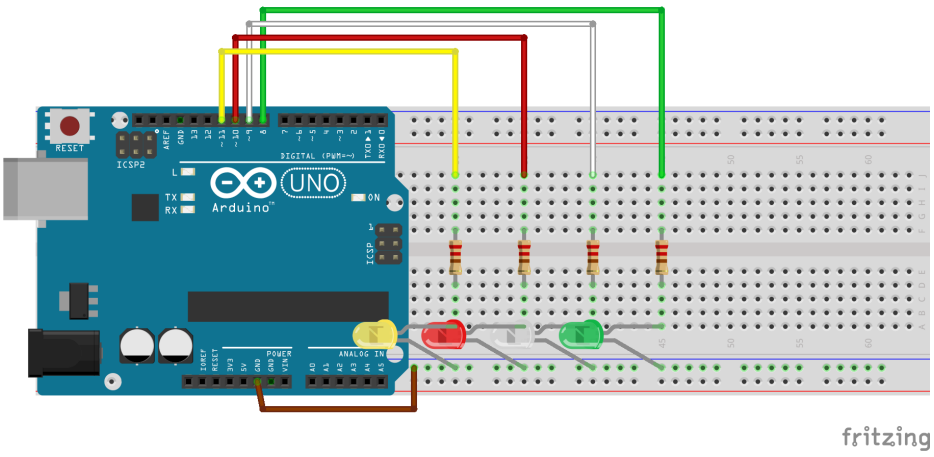

Figura 2: Conección entre el arduino, los Switches y el LED

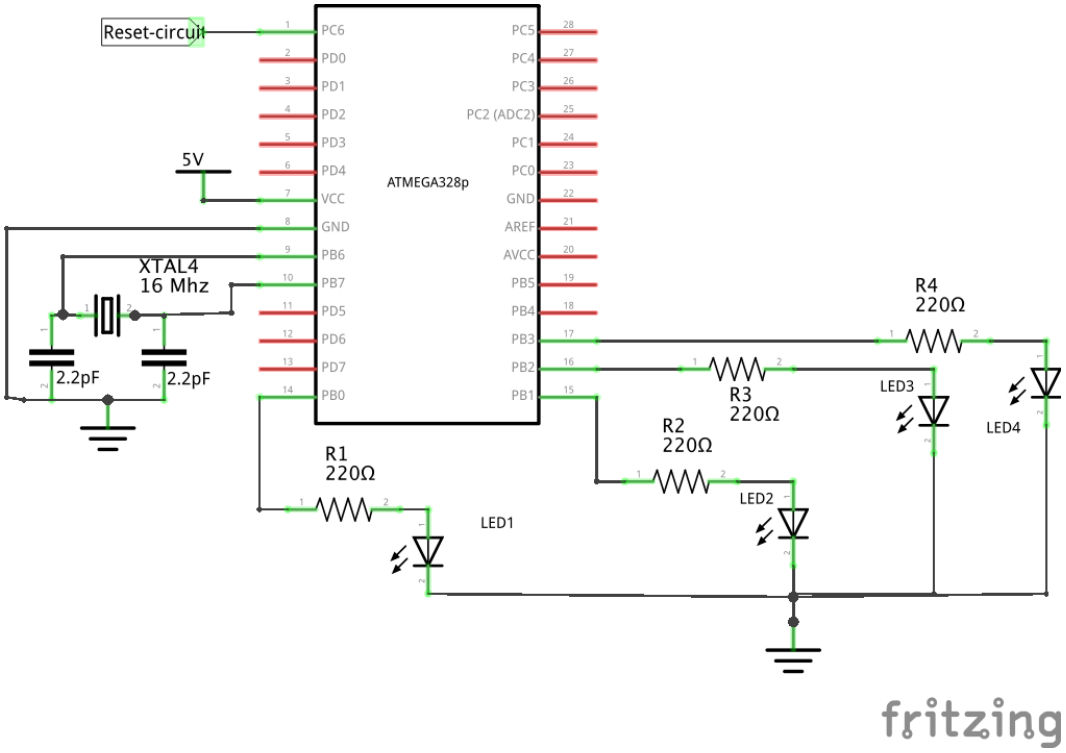

Figura 3: Esquemático del circuito implementado.

5. Diagrama de flujo

En la **Fig. 4** se muestra el diagrama de flujo del programa a partir de polling y en la **Fig. 5** se muestra el diagrama de flujo del programa a partir de interrupciones.

5.1. Polling

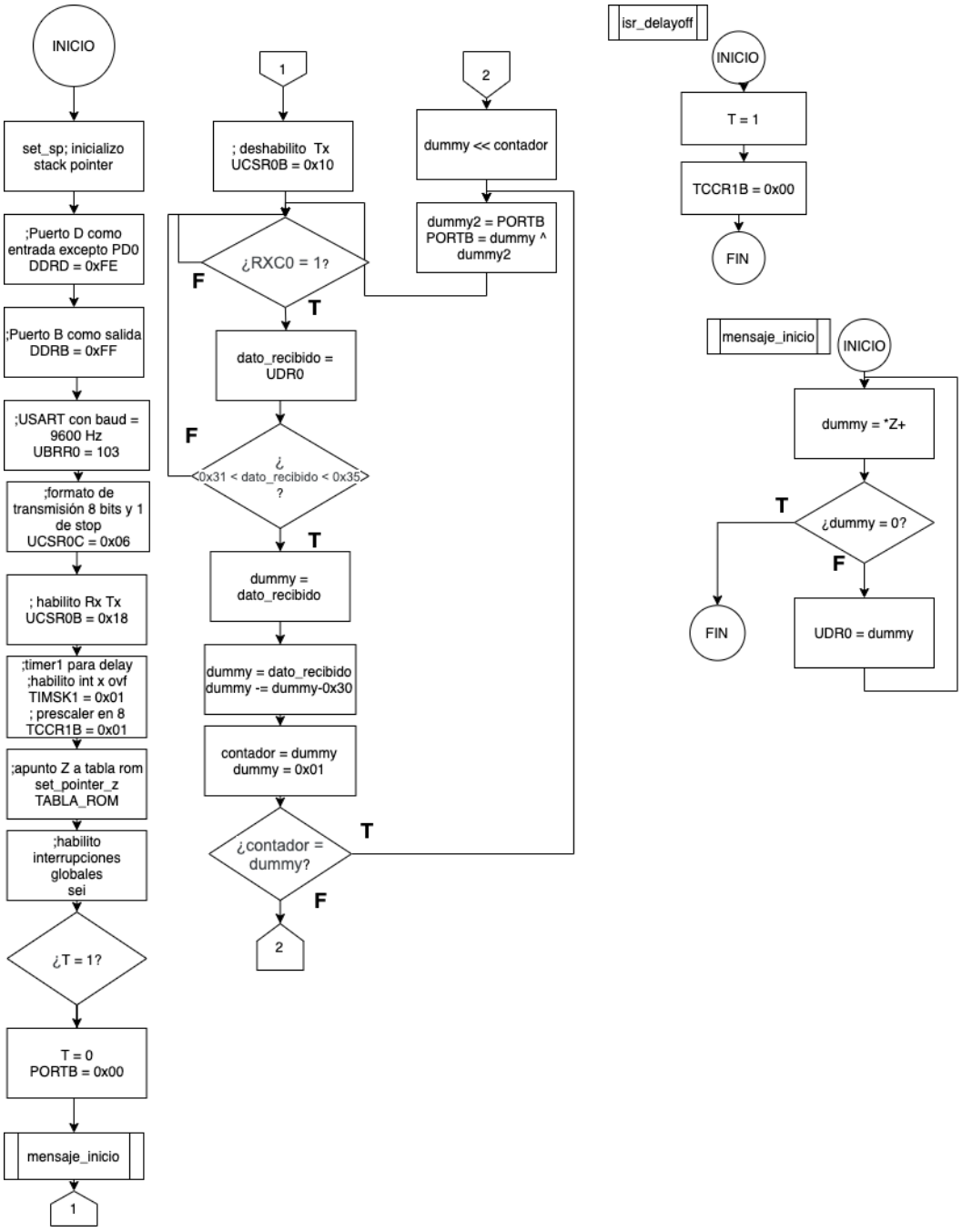

Figura 4: Diagrama de flujo del programa a partir de polling.

5.2. Interrupciones

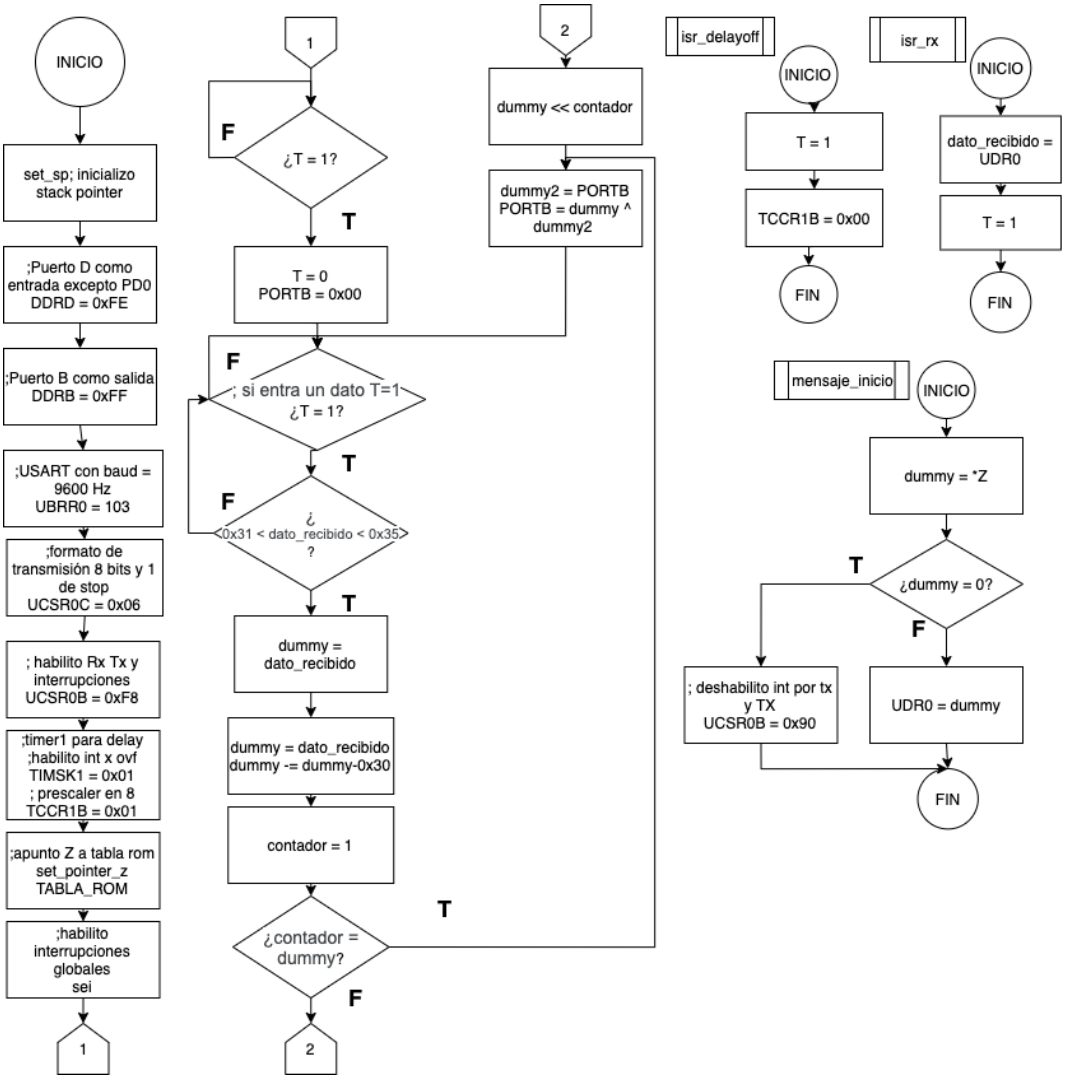

Figura 5: Diagrama de flujo del programa a partir de interrupciones.

6. Código

En los siguientes códigos se configura el micro de manera tal de generar una conexión mediante el protocolo `USART` con una computadora de escritorio. Para esto se configuran los registros `UBRR0`, `UCSR0B` y `UCSR0C`. Y se utiliza el puerto B para conectar cuatro LEDs a ser controlados desde instrucciones de la computadora.

Los registros se configuran de la siguiente forma:

6.1. UBRR0

Los registros que componen `UBRR0` (`UBRR0H` y `UBRR0L`) son configurados de manera que el baudrate sea $baud = 9600 \iff UBRR0 = 103$ con un error de $0.2\,\%$.

6.2. Polling

El siguiente código corresponde a la versión a partir de polling.

```
; Autor: Francisco Rossi
; Padron: 99540
; 86.07 Laboratorio de Microprocesadores - FIUBA
; Catedra: Miercoles
; Fecha: 19 de agosto de 2020

.include "m328pdef.inc"

.def poll_buff = r17
.def dato_recibido = r18
.def contador = r19
.def dummy2 = r24
.def dummy = r25

.macro set_sp
  ldi dummy, low(RAMEND)
  out spl, dummy
  ldi dummy, high(RAMEND)
  out sph, dummy
.endm


.macro set_port_as_out
  ldi dummy, 0xFF
  out @0, dummy
.endm


.macro set_pointer_z
  ldi ZL, low(@0 << 1)
  ldi ZH, high(@0 << 1)
.endm

.cseg

.org 0x0000
  jmp config

.org OVF1addr
  jmp isr_delayoff

; EXT INT
.org INT_VECTORS_SIZE

config:
```

```
46    ; inicializo el stack pointer
47    set_sp
48
49    ; Rx PD0
50    ; Tx PD1
51    ; Leds en PB0,1,2,3
52
53    ;Puerto D como salida excepto PD0
54
55    ldi   dummy,0xFE
56    out   DDRD,dummy
57
58    ; puerto B como salida
59    set_port_as_out DDRB
60
61    ; Configuro USART en baud 9600
62
63    clr   dummy
64    sts   UBRR0H, dummy
65    ldi   dummy, 103
66    sts   UBRR0L, dummy
67
68    ; seteo el formato 8 bits y 1 de stop
69    ldi   dummy, (1 << UCSZ01 | 1 << UCSZ00)
70    sts   UCSR0C, dummy
71
72    ; habilito el envio y la recepci n de datos
73    ldi   dummy,(1 << RXEN0 | 1 << TXEN0 )
74    sts   UCSR0B, dummy
75
76    ; inicializo puntero Z en la tabla con el mensaje a enviar
77        set_pointer_z TABLA_ROM
78
79    ; config timer para delay
80    ldi   dummy,0x01
81    sts   TIMSK1, dummy
82    ldi   dummy, 0x02 ; prescaler de 8
83    sts   TCCR1B, dummy
84
85    ; habilito interrupciones globales.
86    sei
87  main:
88
89  ; delay inicial para dar tiempo a la consola
90  delay:
91    brts  delay_off
92    rjmp  delay
93
94  delay_off:
95    clt
96    ; limpio el puerto B
97    clr   dummy
98    out   PORTB,dummy
99
100   call  mensaje_inicio
101   ; desactivo la transmisi n
102   ldi   dummy,(1 << RXEN0 | 0 << TXEN0 )
103   sts   UCSR0B, dummy
104
105 leer_entrada:
106   ; polling hasta que halla datos a recibir
107   lds   dummy,UCSR0A
108   sbrs  dummy,RXC0
109   rjmp  leer_entrada
110   ; guardo el dato
111   lds   dato_recibido, UDR0
112
113 refresh_leds:
114   ; reviso que el dato recibido este en los valores que tienen acci n
```

```
115   cpi    dato_recibido, 0x31
116   brlo   no_es_valido
117   cpi    dato_recibido, 0x35
118   brsh   no_es_valido
119   ; si es 1 2 3 o 4 ingresa
120 match:
121
122   mov    dummy,dato_recibido
123   subi   dummy,0x30
124   mov    contador, dummy
125   ; ahora contador contiene el numero de led a permutar 1,2,3,4
126
127   ; si es 1 directamente encendo el mismo
128   ldi    dummy,0x01
129   cp     dummy,contador
130   breq   sigo
131   ; si no sigo
132   dec    contador
133   ; corro la mascara (dummy) a la posici n en PortB del led a encender
134 loop:
135   lsl    dummy
136   dec    contador
137   breq   sigo
138   rjmp   loop
139
140 sigo:
141   ; permuto el led correspondiete determinado por la mascara en dummy
142   in     dummy2, PORTB
143   eor    dummy2, dummy
144   out    PORTB, dummy2
145
146 no_es_valido:
147   rjmp   leer_entrada
148
149 isr_delayoff:
150
151   set
152   clr    dummy
153   ; apago el timer
154   sts    TCCR1B, dummy
155   reti
156
157 mensaje_inicio:
158   ; cargo proximo byte a transmitir en el registro dummy
159   lpm    dummy, Z+
160
161   cpi    dummy, 0
162   breq   end ; si leo 0 es que termine de transmitir
163
164   ; espero a que el buffer de transmision este libre para cargarlo
165 buffer_empty_poll:
166   lds    poll_buff, UCSR0A
167   sbrs   poll_buff, UDRE0
168   rjmp   buffer_empty_poll
169
170   ; si esta libre transmito dato
171   sts    UDR0, dummy
172   rjmp   mensaje_inicio
173 end:
174   ret
175
176 .org 0x500
177 TABLA_ROM: .db "*** Hola Labo de Micro ***",'\r','\n',"Escriba 1, 2, 3 o 4 para controlar los
      LEDs",'\r','\n',0
```

6.3. Interrupciones

El siguiente código corresponde a la versión a partir de interrupciones.

```
; Autor: Francisco Rossi
; Padron: 99540
; 86.07 Laboratorio de Microprocesadores - FIUBA
; Catedra: Miercoles
; Fecha: 19 de agosto de 2020

.include "m328pdef.inc"

.def dato_recibido = r18
.def contador = r19
.def dummy2 = r24
.def dummy = r25

.macro set_sp
  ldi dummy, low(RAMEND)
  out spl, dummy
  ldi dummy, high(RAMEND)
  out sph, dummy
.endm

.macro set_port_as_out
  ldi dummy, 0xFF
  out @0, dummy
.endm

.macro set_pointer_z
  ldi ZL, low(@0 << 1)
  ldi ZH, high(@0 << 1)
.endm

.cseg

.org 0x0000
  jmp config

; habilito interrupciones dadas por RXC0 y UDRE0
.org URXCaddr
  jmp isr_rx
.org UDREaddr
  jmp mensaje_inicio
.org OVF1addr
  jmp isr_delayoff
; EXT INT
.org INT_VECTORS_SIZE

config:
  set_sp

  ; Rx PD0
  ; Tx PD1
  ; Leds en PB 0,1,2,3

  ; Config Puerto D como salida excepto PD0

  ldi    dummy,0xFE
  out    DDRD,dummy

  ; puerto B como salida
  set_port_as_out DDRB

  ; Configuro USART en baud 9600

  clr    dummy
  sts    UBRR0H, dummy
```

```
65    ldi    dummy, 103
66    sts    UBRR0L, dummy
67
68    ; seteo el formato 8 bits y 1 de stop
69    ldi    dummy, (1 << UCSZ01 | 1 << UCSZ00)
70    sts    UCSR0C, dummy
71
72    ; config de interrupciones de USART 0 y enable de transmisi n y recepci n
73    ldi    dummy,(1 << UDRIE0|1 << RXCIE0 | 1 << TXCIE0 | 1 << RXEN0 | 1 << TXEN0 )
74    sts    UCSR0B, dummy
75
76
77
78
79    ; config timer para delay
80    ldi    dummy,0x01
81    sts    TIMSK1, dummy
82    ldi    dummy, 0x02 ; prescaler de 8
83    sts    TCCR1B, dummy
84
85    ; inicializo puntero Z
86    set_pointer_z TABLA_ROM
87    ; habilito interrupciones globales.
88    sei
89  main:
90
91
92  ; delay inicial para dar tiempo a la consola
93  delay:
94    brts   delay_off
95    rjmp   delay
96
97  delay_off:
98    clt
99    ; limpio el puerto B
100   clr    dummy
101   out    PORTB,dummy
102
103 hold:
104   brts     refresh_leds
105   rjmp     hold
106
107 refresh_leds:
108   ; reviso que el dato recibido este en los valores que tienen acci n
109   cpi    dato_recibido, 0x31
110   brlo   no_es_valido
111   cpi    dato_recibido, 0x35
112   brsh   no_es_valido
113   ; si es 1 2 3 o 4 ingresa
114
115 match:
116
117   mov    dummy,dato_recibido
118   subi   dummy,0x30
119   mov    contador, dummy
120   ; ahora contador contiene el numero de led a permutar 1,2,3,4
121
122   ldi    dummy,0x01
123   cp     dummy,contador
124   breq   sigo
125   dec    contador
126   ; corro la mascara (dummy) a la posici n en PortB del led a encender
127
128 loop:
129   lsl    dummy
130   dec    contador
131   breq   sigo
132   rjmp   loop
133
```

```
134 sigo:
135   ; permuto el led correspondiete determinado por la mascara en dummy
136   in     dummy2, PORTB
137   eor    dummy2, dummy
138   out    PORTB, dummy2
139 
140   clt
141 
142 no_es_valido:
143   rjmp   hold
144 
145 isr_delayoff:
146   set
147   clr    dummy
148   sts    TCCR1B, dummy
149   reti
150 
151 mensaje_inicio:
152   ; rutine de interrupcion para transmitir el mensaje
153   ; cada vez que el buffer de transmision este libre para cargarlo se entra a al rutina
154   ; cargo porximo byte a transmitir en dummy
155   lpm    dummy, Z+
156   cpi    dummy, 0
157   breq   end ; si leo 0 es que termine de transmitir
158 
159   sts    UDR0, dummy
160   reti
161 end:
162   ; deshabilito interupciones y transmisi n cuando termino de transimitir el mensaje
163   ldi    dummy,(0 << UDRIE0|1 << RXCIE0 |0 << TXCIE0 | 1 << RXEN0 | 0 << TXEN0 )
164   sts    UCSR0B, dummy
165   reti
166 
167 isr_rx:
168   ; si recibo un dato marco el mismo con el bit t
169   lds    dato_recibido, UDR0
170   set
171   reti
172 
173 .org 0x500
174 TABLA_ROM: .db "*** Hola Labo de Micro ***",'\r','\n',"Escriba 1, 2, 3 o 4 para controlar los
      LEDs",'\r','\n',0
```

7. Resultados

Se logró diseñar un programa para el microcontrolador ATMEGA328p con el cual se genera una conexión bilateral entre el microcontrolador y la computadora, enviando (el mensaje de inicio) y recibiendo información y actuando en función de la misma apagando y prendiendo los LEDs conectados al puerto B.

8. Conclusiones

Se logró generar un programa que establezca la conexión bilateral con la computadora y efectivamente maneje los LEDs en función de las instrucciones enviadas por la computadora.