



INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

Práctica 1 Puertos de Entrada y Salida

Alumno:

Amador Nava Miguel Ángel

Profesor:

Aguilar Sanchez Fernando

Grupo:

3CM5

Asignatura:

Introducción a los Microcontroladores



Índice general

Introducción	3
Objetivo	3
Material y equipo	3
Planteamiento del problema	4
Análisis teórico	4
Diseño del circuito	7
Análisis práctico	8
Comparaciones	8
Conclusiones	9
Código en lenguaje C	10
Referencias	13

Introducción

Los microcontroladores son un circuito integrado que tienen un microprocesador, memoria de programa y de datos así como unidades de entrada y salida (puertos paralelo, temporizadores, conversores A/D, puertos serie, etc). [1]

Las unidades de entrada y salida se refieren a los puertos que tiene el microcontrolador para recibir o enviar datos en forma serie o en forma paralela, y sirven para monitorizar y controlar otros dispositivos[2]. Cuentan además con módulos especiales para convertir señales analógicas a digitales o de digitales a analógicas.

Generalmente tienen arquitectura Harvard que es aquella en donde existen dos buses independientes para mejorar la velocidad de transferencia de información interna: el bus de datos y el bus de direcciones. El bus de datos puede ser de 8, 16, 32 bits y el de dirección depende de la cantidad de memoria del microcontrolador.

Los parámetros más importantes en un microcontrolador son:

- Bus de datos: 8, 16, 32 bits.
- Capacidad de memoria: Tamaño de la memoria RAM y de la memoria EEPROM en kilobytes.
- Velocidad: Número de instrucciones a ejecutar por segundo. Depende de la frecuencia del oscilador del microcontrolador.
- Puertos: Puertos de entrada y salida de forma paralela y serial para comunicación externa.
- Módulos: Para conversión A/D, D/A, PWM, USB, CAN, I2C, SPI, UART, USART, etc.[3]

Objetivo

Al término de la sesión, los integrantes del equipo contaran con la habilidad de programar los puertos como entrada y salida del Microcontrolador ATmega8535 usando las herramientas “Code Vision AVR” y “AVR Studio 4”.

Material y equipo

- CodeVision AVR
- AVR Studio 4
- Microcontrolador ATmega 8535
- 8 LED's
- 8 Resistores de 330Ω a $\frac{1}{4} W$
- 1 Dip switch u ocho Push Botón

Planteamiento del problema

Realiza un programa para programar el Puerto B como entrada y escribir la información en el Puerto D activándolo como salida, recuerde activar las resistencias de Pull-up del puerto B para colocar solo el Dipswitch.

Análisis teórico

Las resistencias "pull-up" y "pull-down" son resistencias normales, colocadas independientemente o en un grupo de resistencias del mismo valor, solo que tienen ese nombre por la función que cumplen: sirven para sumir un valor por defecto de la señal recibida en una entrada del circuito cuando por ella no se detecta ningún valor concreto (ni alto ni bajo), cuando la entrada no está conectada a nada. Este tipo de resistencias aseguran que los valores binarios recibidos no fluctúan sin sentido en ausencia de señal de entrada.[4]

Para la configuración pull up cuando el push botón no esté precionado, se tendrá el valor de Vcc (1 lógico), y cuando se precione será un 0 Volts (0 lógico). En la configuración de pull down es lo contrario que pull up.

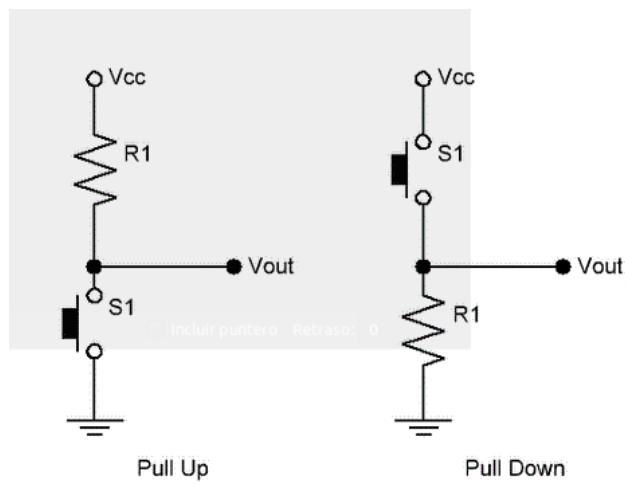


Figura 1: Circuito para resistencias pull up y pull down

Se pretende que los valores ingresados a través del dip switch en el puerto B se reflejen en el puerto D. Para activar las resistencias de pull up inicializamos en 1 y como entrada el puerto B.

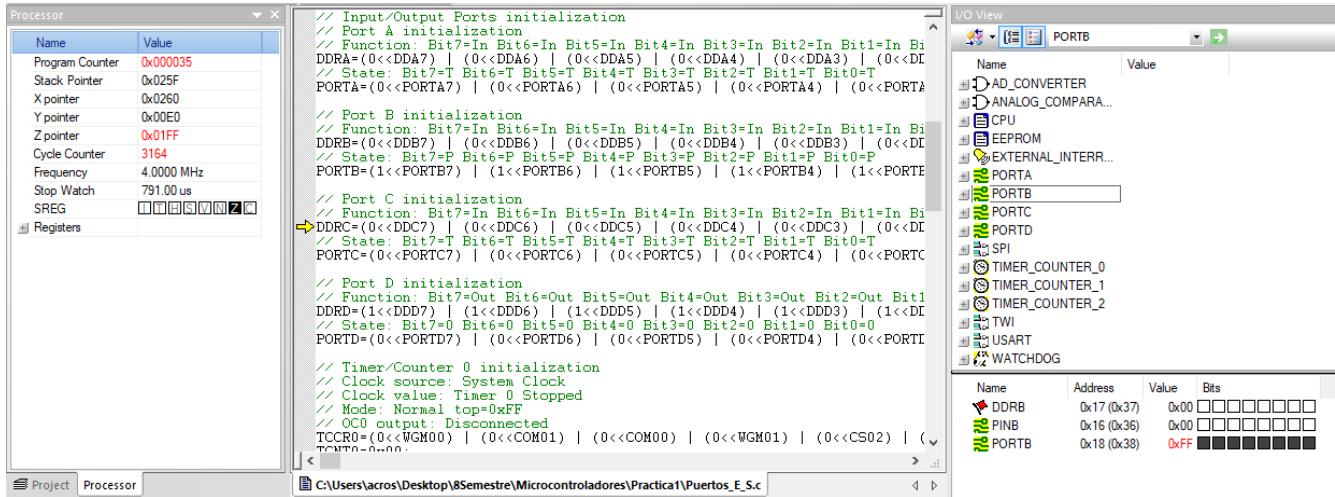


Figura 2: Inicialización del puerto B

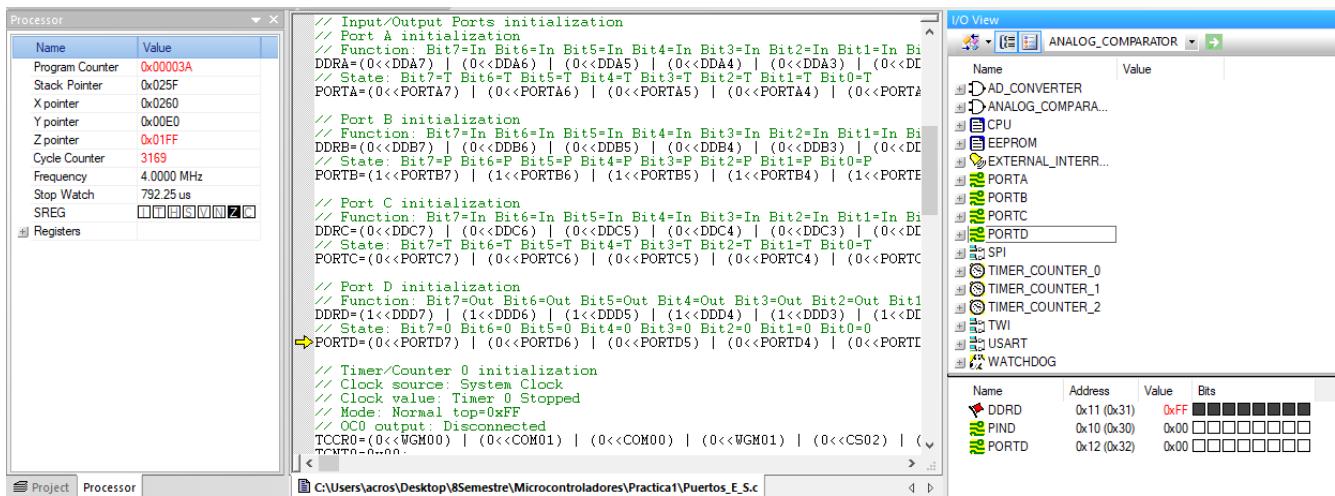


Figura 3: Inicialización del puerto D

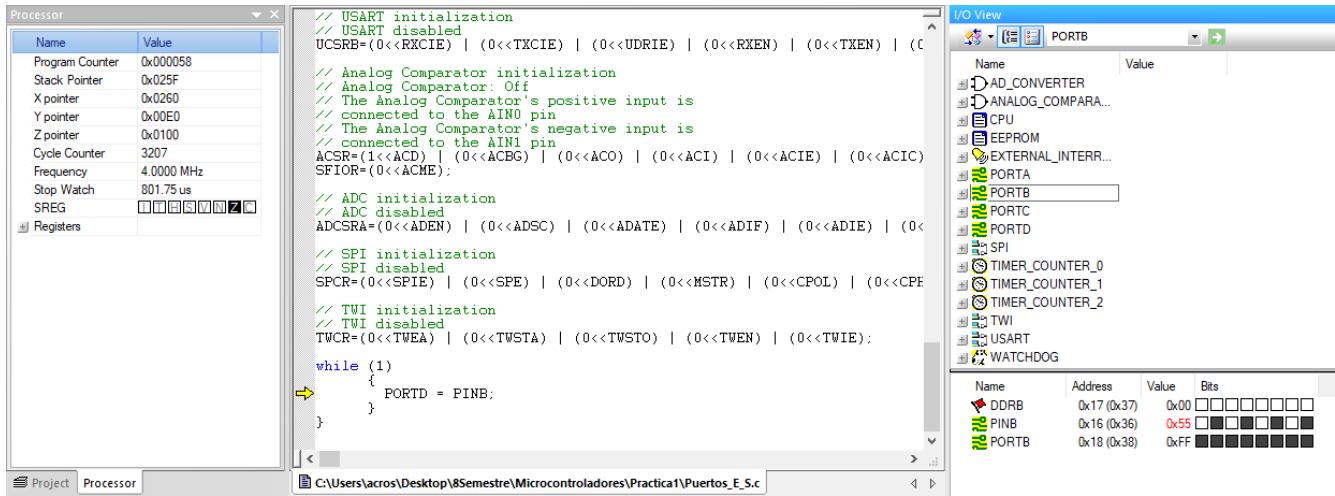


Figura 4: Ingreso de valores al PIND

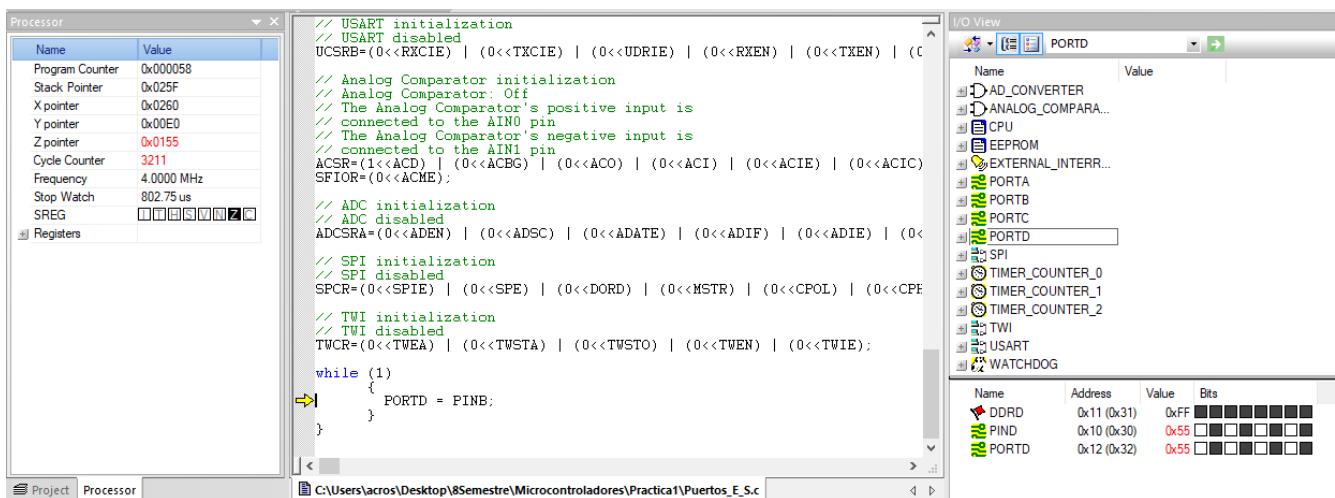


Figura 5: Reflejo de valores en PORTD

Diseño del circuito

En la Figura 6 se muestra el circuito para la implementación de la práctica.

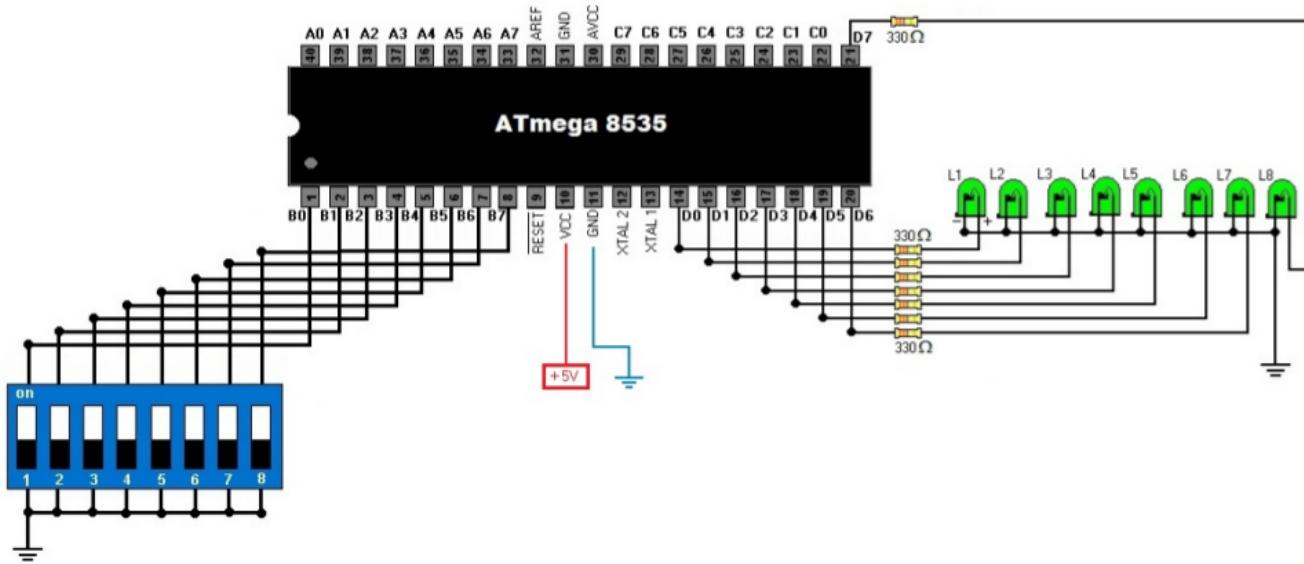


Figura 6: Diseño del circuito

Análisis práctico

En la Figura 7 se puede ver el circuito implementado, siendo alimentado con el grabador de AVR's, esta figura muestra una combinación controlada por el dip switch.

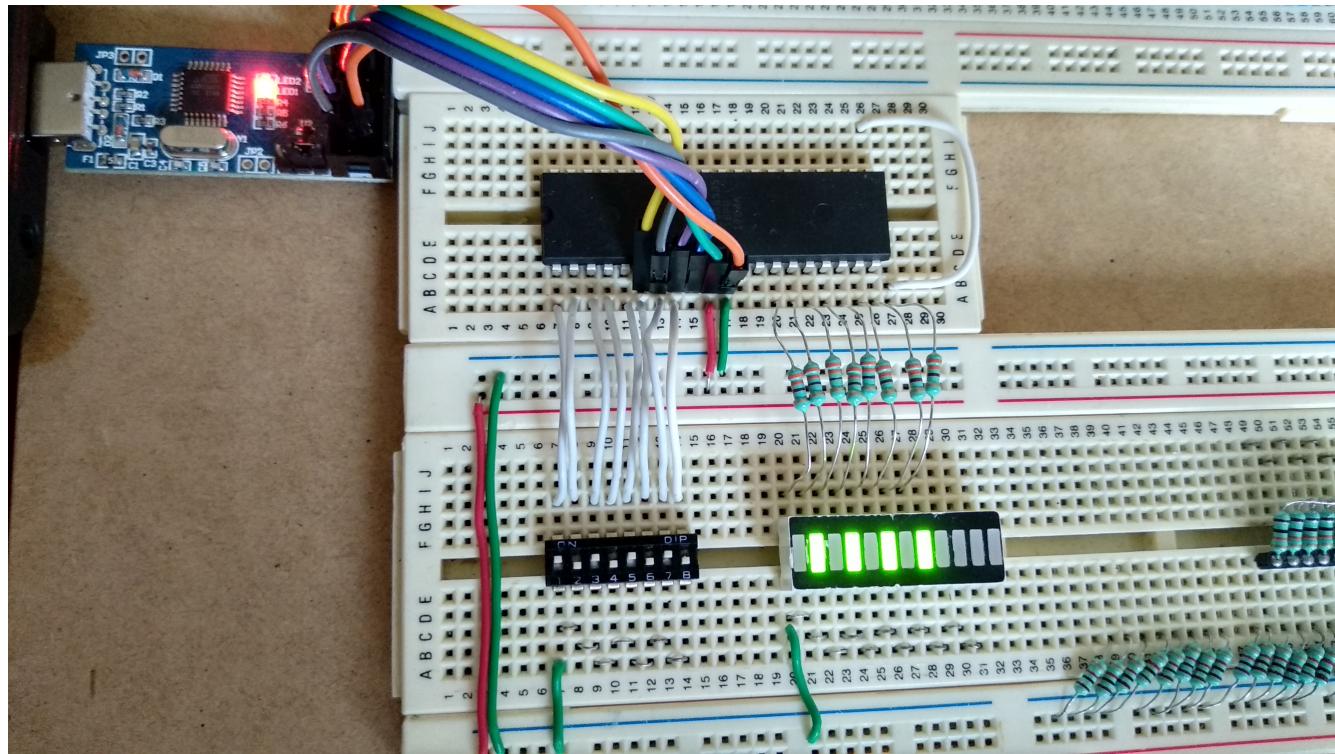


Figura 7: Circuito funcionando

Comparaciones

Los resultados obtenidos son iguales a los esperados en la simulación, lo que nos permite ver que lo que se ingresa en el puerto B, se refleja de la misma manera en el puerto D.

Conclusiones

Los puertos de entrada y salida sirven para recibir y enviar información, los datos de entrada se reciben con PINX donde X es la letra del puerto correspondiente, los datos de salida se reciben con PORX donde X es la letra del puerto correspondiente. Y estos servirán para manejar o recibir información de dispositivos y procesar esos datos.

Código en lenguaje C

```
1  ****
2 This program was created by the CodeWizardAVR V3.34
3 Automatic Program Generator
4 Copyright 1998-2018 Pavel Haiduc, HP InfoTech s.r.l.
5 http://www.hpinfotech.com
6
7 Project : Introduccin a los Microcontroladores
8 Version : 1.0
9 Date    : 25/01/2019
10 Author  : Amador Nava Miguel ngel
11 Company : Escuela Superior de Cmputo
12 Comments: Prctica 1 Puertos E/S
13
14
15 Chip type          : ATmega8535
16 Program type        : Application
17 AVR Core Clock frequency: 1.000000 MHz
18 Memory model        : Small
19 External RAM size   : 0
20 Data Stack size     : 128
21 ****
22
23 #include <mega8535.h>
24
25 // Declare your global variables here
26
27 void main(void)
28 {
29 // Declare your local variables here
30
31 // Input/Output Ports initialization
32 // Port A initialization
33 // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
34 DDRA=(0<<DDA7) | (0<<DDA6) | (0<<DDA5) | (0<<DDA4) | (0<<DDA3) | (0<<DDA2)
35 // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
36 PORTA=(0<<PORTA7) | (0<<PORTA6) | (0<<PORTA5) | (0<<PORTA4) | (0<<PORTA3) |
37
38 // Port B initialization
39 // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
40 DDRB=(0<<DDB7) | (0<<DDB6) | (0<<DDB5) | (0<<DDB4) | (0<<DDB3) | (0<<DDB2)
41 // State: Bit7=P Bit6=P Bit5=P Bit4=P Bit3=P Bit2=P Bit1=P Bit0=P
```

```
42 PORTB=(1<<PORTB7) | (1<<PORTB6) | (1<<PORTB5) | (1<<PORTB4) | (1<<PORTB3) |  
43  
44 // Port C initialization  
45 // Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In  
46 DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (0<<DDC3) | (0<<DDC2)  
47 // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T  
48 PORTC=(0<<PORTC7) | (0<<PORTC6) | (0<<PORTC5) | (0<<PORTC4) | (0<<PORTC3) |  
49  
50 // Port D initialization  
51 // Function: Bit7=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out Bit1=Out  
52 DDRD=(1<<DDD7) | (1<<DDD6) | (1<<DDD5) | (1<<DDD4) | (1<<DDD3) | (1<<DDD2)  
53 // State: Bit7=0 Bit6=0 Bit5=0 Bit4=0 Bit3=0 Bit2=0 Bit1=0 Bit0=0  
54 PORTD=(0<<PORTD7) | (0<<PORTD6) | (0<<PORTD5) | (0<<PORTD4) | (0<<PORTD3) |  
55  
56 // Timer/Counter 0 initialization  
57 // Clock source: System Clock  
58 // Clock value: Timer 0 Stopped  
59 // Mode: Normal top=0xFF  
60 // OC0 output: Disconnected  
61 TCCR0=(0<<WGM00) | (0<<COM01) | (0<<COM00) | (0<<WGM01) | (0<<CS02) | (0<<CS01)  
62 TCNT0=0x00;  
63 OCR0=0x00;  
64  
65 // Timer/Counter 1 initialization  
66 // Clock source: System Clock  
67 // Clock value: Timer1 Stopped  
68 // Mode: Normal top=0xFFFF  
69 // OC1A output: Disconnected  
70 // OC1B output: Disconnected  
71 // Noise Canceler: Off  
72 // Input Capture on Falling Edge  
73 // Timer1 Overflow Interrupt: Off  
74 // Input Capture Interrupt: Off  
75 // Compare A Match Interrupt: Off  
76 // Compare B Match Interrupt: Off  
77 TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) | (0<<COM1B0) | (0<<WGM11) |  
78 TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<WGM13) | (0<<WGM12) | (0<<CS12) | (0<<CS11)  
79 TCNT1H=0x00;  
80 TCNT1L=0x00;  
81 ICR1H=0x00;  
82 ICR1L=0x00;  
83 OCR1AH=0x00;  
84 OCR1AL=0x00;  
85 OCR1BH=0x00;
```

```
86  OCR1BL=0x00;  
87  
88 // Timer/Counter 2 initialization  
89 // Clock source: System Clock  
90 // Clock value: Timer2 Stopped  
91 // Mode: Normal top=0xFF  
92 // OC2 output: Disconnected  
93 ASSR=0<<AS2;  
94 TCCR2=(0<<WGM20) | (0<<COM21) | (0<<COM20) | (0<<WGM21) | (0<<CS22) | (0<<CS21)  
95 TCNT2=0x00;  
96 OCR2=0x00;  
97  
98 // Timer(s)/Counter(s) Interrupt(s) initialization  
99 TIMSK=(0<<OCIE2) | (0<<TOIE2) | (0<<TICIE1) | (0<<OCIE1A) | (0<<OCIE1B) | (0<<OCIE0)  
100  
101 // External Interrupt(s) initialization  
102 // INT0: Off  
103 // INT1: Off  
104 // INT2: Off  
105 MCUCR=(0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);  
106 MCUCSR=(0<<ISC2);  
107  
108 // USART initialization  
109 // USART disabled  
110 UCSRB=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) | (0<<RXEN) | (0<<TXEN) | (0<<UCSRB)  
111  
112 // Analog Comparator initialization  
113 // Analog Comparator: Off  
114 // The Analog Comparator's positive input is  
115 // connected to the AIN0 pin  
116 // The Analog Comparator's negative input is  
117 // connected to the AIN1 pin  
118 ACSR=(1<<ACD) | (0<<ACBG) | (0<<ACO) | (0<<ACI) | (0<<ACIE) | (0<<ACIC) | (0<<ACIE)  
119 SFIOR=(0<<ACME);  
120  
121 // ADC initialization  
122 // ADC disabled  
123 ADCSRA=(0<<ADEN) | (0<<ADSC) | (0<<ADATE) | (0<<ADIF) | (0<<ADIE) | (0<<ADPS)  
124  
125 // SPI initialization  
126 // SPI disabled  
127 SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) | (0<<CPOL) | (0<<CPHA) | (0<<CPOL)  
128  
129 // TWI initialization
```

```
130 // TWI disabled
131 TWCR=(0<<TWEA) | (0<<TWSTA) | (0<<TWSTO) | (0<<TWEN) | (0<<TWIE);
132
133 while (1)
134 {
135     PORTD = PINB;
136 }
137 }
```

Bibliografía

- [1] Aquitectura y organización de un microcontrolador menerico. [Online]. Available: http://www.exa.unicen.edu.ar/catedras/tmicrocon/Material/1_introduccion_a_los_ucontroladores.pdf
- [2] Puertos e/s. Universidad de Oviedo. [Online]. Available: <https://www.unioviedo.es/ate/alberto/TEMA4-puertos.pdf>
- [3] J. A. Polanía. Microcontroladores. CEDUVIRT. [Online]. Available: <https://www.ceduvirt.com/resources/1.Puertos%20de%20entrada-salida.pdf>
- [4] Las resistencias “pull-up” y “pull-down”. Myelectronic. [Online]. Available: <http://myelectronic.mipropia.com/Componentes/LAS%20RESISTENCIAS%20PULL-UP.pdf?i=1>