Universidad de Buenos Aires
Facultad de Ingeniería
Año 2020 - 1^{ero} cuatrimestre

Laboratorio de Microprocesadores (86.07)

TRABAJO PRÁCTICO OBLIGATORIO 3

ESTUDIANTE:
Pintos Gaston 99711
gmpintos@fi.uba.ar

Índice

1. Objetivos

El trabajo práctico consiste en encender de manera progresiva una tira constituida por 6 LED. El encendido se realiza de manera única desde el bit menos significativo hacia el de ma significación. Luego se realiza la tarea contraria hasta encender el LED correspondiente al de menos significación. Se utilizo el microcontrolador *atmega 328P* de la placa *Arduino*. A lo largo del desarrollo se presentaran las herramientas utilizadas para la realización del proyecto.

2. Desarrollo:

2.1. Herramientas de Software:

Para el siguiente proyecto se utilizo la herramienta *AtmelStudio* para desarrollar el código en lenguaje *Assembly*. La misma herramienta permite simular el código escrito con lo cual fue posible realizar pruebas a lo largo del desarrollo. Además se utilizo la herramienta *AVRdude* para poder montar el programa en el microcontrolador.

2.1.1. Código de Montaje:

Se utilizo el siguiente código para cumplir la tarea asignada:

```
.device ATmega328P
.cseg
.org 0x0000

main:
    LDI R16,HIGH(RAMEND)
        OUT SPH,R16
        LDI R16, LOW(RAMEND)
        OUT SPL, R16

    LDI R17,0X00
    OUT PORTB,R17 ;Limpio el puerto B

    LDI R16,0XFF
    OUT DDRB,R16 ;SETEO PUERTO B COMO SSALIDA

    LDI R17,0b000001 ;Enciendo el primer led
    OUT PORTB,R17
    CALL DELAY

SHIFT:
     CALL LEFT_SHIFT ;Desplazo el bit hacia izquierda
     CALL RIGHT_SHIFT ;Desplazo el bit hacia derecha
     RJMP SHIFT              ;Reinicio el proceso

LEFT_SHIFT:
     LDI R18,5               ;Cantidad de desplazamientos
LOOP1:
     LSL R17                 ;Movimiento hacia la izquierda
     OUT PORTB,R17
     CALL DELAY
     DEC R18
     BRNE LOOP1
     RET

RIGHT_SHIFT:
     LDI R18,5               ;Cantidad de desplazamientos
LOOP2:
     LSR R17                 ;Movimiento hacia la derecha
     OUT PORTB,R17
```

```
        CALL DELAY
        DEC R18
        BRNE LOOP2
        RET

    DELAY:
        LDI R21, 200
    D1: LDI R22, 200
    D2: LDI R23, 50
    D3: NOP
        NOP
        DEC R23
        BRNE D3
        DEC R22
        BRNE D2
        DEC R21
        BRNE D1
        RET
```

Se utilizo la instrucción *.DEVICE* para identificar el tipo de microcontrolador utilizado. La directiva *.CSEG* indica la locación de memoria utilizada para escribir el código. Con la directiva *.ORG* se seteo el *stack pointer* al inicio de la memoria flash donde se ubican las directivas del programa.

Se ingresa el valor 0x00 en el puerto B utilizando el registro 17. Con esto se busca limpiar cual valor previo del puerto. Luego se ingresa el valor 0xFF en el DDRB con el uso del registro 16. Esto setea al puerto B como salida de información.

El programa inicia ingresando en el puerto el valor 0b000001. Con esto se logra encender el LED correspondiente al pin 0 del puerto utilizado. Luego se ingresa en el loop "SHIFT.ªsignado a desplazar el bit 1 contenido en el puerto. Para esto se utilizan las funciones "$LEFT\_SHIFT$" con la instrucción LSL y "$RIGHT\_SHIFT$" con la instrucción LSR. Ambas consisten de un loop controlado por el registro 18 que contiene la cantidad de desplazamientos requeridos.

Se utiliza un llamado a la función "DELAY.ªntre los encendido de los LED para poder identificar tal operación.

Todas las instrucciones utilizadas para la escritura del código se encuentran en el manual de instrucciones del microcontrolador. El mismo puede encontrarse en la bibliografía al finalizar el trabajo.

2.1.2. Calculo del retardo:

Para el siguiente código se utilizo el siguiente retardo: para una frecuencia de clock característica de $16MHz$:

```
    DELAY:
        LDI R21, 200     1 ciclo
    D1: LDI R22, 200     1 ciclo
    D2: LDI R23, 50      1 ciclo
    D3: NOP              1 ciclo
        NOP              1 ciclo
        DEC R23          1 ciclo
        BRNE D3          1 ciclo si no cumple/ 2 si cumple
        DEC R22          1 ciclo
        BRNE D2          1 ciclo si no cumple/ 2 si cumple
        DEC R21          1 ciclo
        BRNE D1          1 ciclo si no cumple/ 2 si cumple
        RET
```

$$tiempo = \frac{ciclos}{frecuencia} = \frac{(((50 * 5) + 3) * 200 + 3) * 200 + 3}{16000000} = 0{,}632s \quad (1)$$

2.1.3. Montaje del código:

Paecuencia dera el montaje del código con la herramienta *AVR DUDE* a través del *ATMEL STUDIO* se utilizaron los siguientes comandos:
"C:9-Laboratorio de Microcomputadoras-6.3-mingw32(1).exe"-C
"C:9-Laboratorio de Microcomputadoras-6.3-mingw32(1).conf" -v
-p atmega328p -c arduino -P COM3 -D
-U flash:w:"$(MSBuildProjectDirectory)$(Configuration)$(OutputFileName).hex":i

Con esta herramienta fue posible que el codigo escrito fuera procesado por el microcontrolador para poder ejecutar las tareas programadas.

2.1.4. Diagramas de flujo:

Para una comprensión de manera gráfica se presenta a continuación un diagrama de flujo del código previamente explicado.

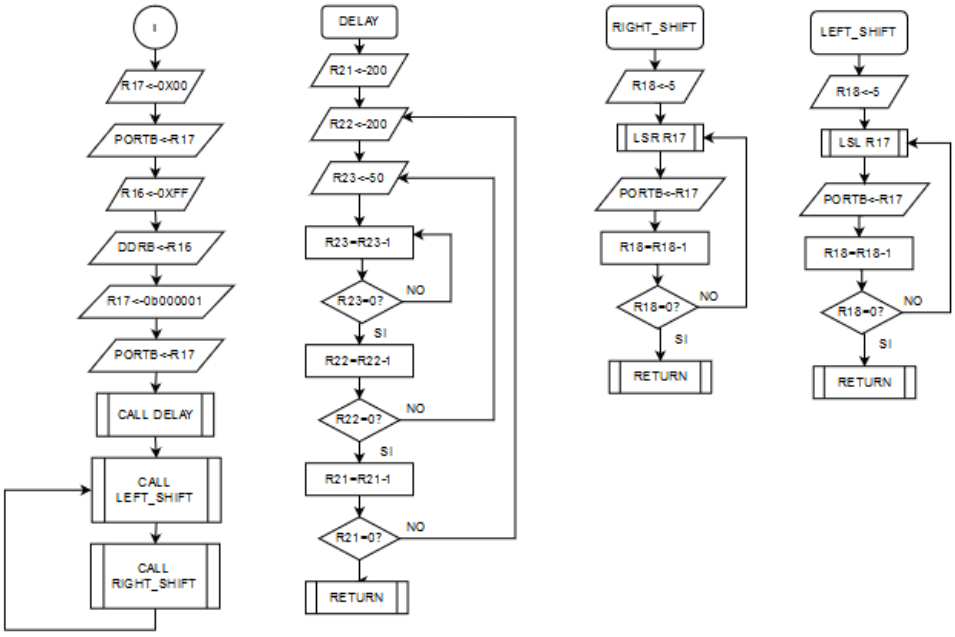

Figura 1: Diagrama de flujo de código.

2.2. Herramientas de Hardware:

Para realizar el proyecto se utilizaron el microcontrolador de la placa *Arduino Uno*, un protoboard donde se montaron todos los elementos, 6 diodos led y 6 resistencias de 220Ω. Para la alimentación y trasmisión del código se utilizo el puerto USB que viene con la placa. Para interconexión entre la placa y el protoboard se utilizaron cables macho-macho. Se puede visualizar la conexión de los elementos en la siguiente imagen.

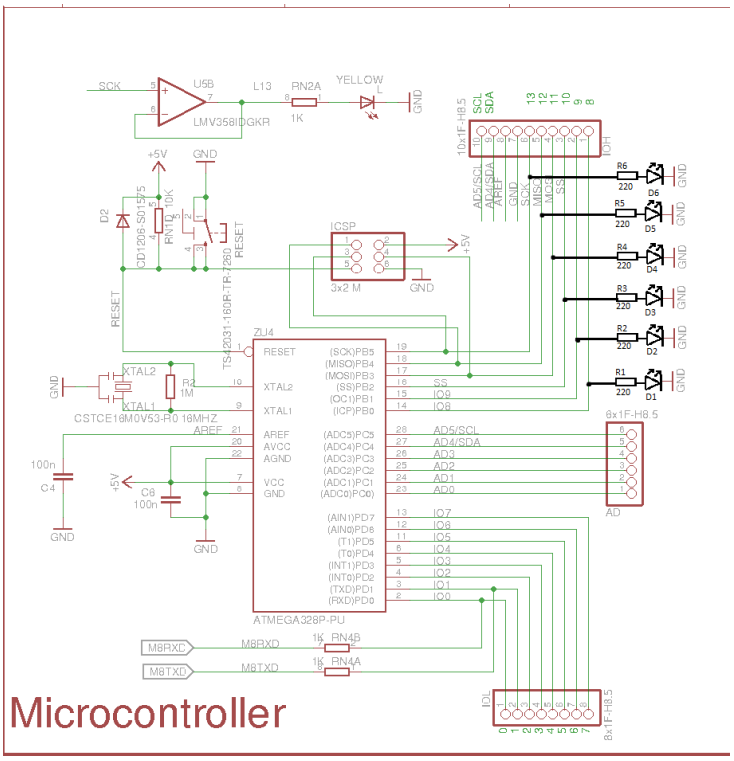

Figura 2: Diagrama de conexión.

La placa *Arduino Uno* cuenta con un microcontrolador *ATmega328p* . El ATmega328 proporciona comunicación serie UART TTL (5V), que está disponible en los pines digitales 0 (RX) y 1 (TX), Dentro de sus conexiones cuenta con 14 pines digitales de entrada/salida. Tiene una memoria flash de 32KB, una SRAM de 2KB y una EEPROM de 1KB. La velocidad del reloj es de 16MHz.

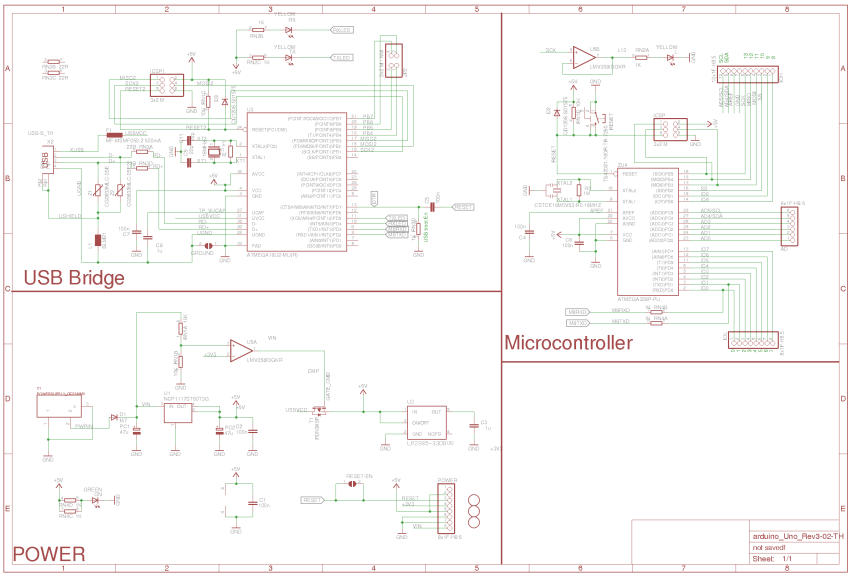

Figura 3: Diagrama de conexión de la placa.

2.2.1. Listado de componentes

Para este trabajo practico fueron utilizados los siguientes elementos. Se estima un coste total de 1500$.

Artículo	Precio
Arduino Uno	$800 - 1500$\$
Led	$20 - 30$\$
Resistencia	$5 - 10$\$

Tabla 1: Precios de componentes.

2.3. Resultados

En el momento de realizar el código se busco introducir instrucciones que puedan desplazar el bit dentro del numero 0b000001 para poder encender los diodos LED. Para esto fueron incluidas las instrucciones LSR y LSL. Esto permitió reducir considerablemente el código ante una posible inicialización de valores de manera unitaria en cada led con instrucciones como OUT cargando un numero binario por cada desplazamiento.
Se implemento el código utilizando la instrucción *call* en cada llamado a la función encargada de desplazar el bit segun fuera necesario. Se evito utilizar la instrucción*rjmp* en la llamada a funciones para evitar problemas en el stack.
Se utilizo la instrucción *rjmp* como consecuencia lógica al necesitar desplazar el bit a derecha una vez que este se encontró en el ultimo LED de la izquierda y viceversa creando un loop en el main.
Se introdujeron directivas al compilador para poder informar el dispositivo utilizado, el segmento de memoria, definir variables y setear el stack pointer.

2.4. Calculo de potencia:

Considero que la corriente que suministra cada pin en estado alto es como máximo la corriente IOH que entrega el microcontrolador.La potencia total consumida es solo la de un único LED ya que estos se encienden de manera unitaria. Entonces, considerando que la caída de tensión característica para un LED de color rojo es $V_{LED} = 1,6V$ y que la corriente IOH es igual a $20mA$ según la hoja de datos, verifico la potencia en ambos casos:

$$Pot_{IOH} = Vcc * IOH = 5v * 20mA = 100mW \tag{2}$$

$$Pot_{LED} = Vcc * I_{LED} = Vcc * \frac{\Delta V}{R} = 5v * \frac{5v - 1,6v}{220} = 77,3mW \tag{3}$$

r Por lo cual podemos verificar que es un consumo acorde para el micrcontrolador utilizado.

3. Conclusiones

Para este trabajo practico se pudo utilizar instrucciones para rotar el valor alto de un numero binario para encender distintos led cumpliendo el objetivo del trabajo. Se desarrollo un código a fin que pudo sintetizar la tarea sin tener que ingresar los números requeridos para que encienda particularmente un LED por lo que genero un código es mas legible, amigable y eficiente.

4. Bibliografía

- AVR Microcontroller and Embedded Systems: Using Assembly and C, 1stEdition, Muhammad Ali Mazidi, Pearson
- "Digital Design, Principles and Practices", 3rdEdition, J.Wakerly, Prentice-Hall
- Set de instrucciones AVR de 8 bits
- Execution cycle of the AVR architecture
- Guía de Trabajos Prácticos
- ATmega328P Datasheet