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

Laboratorio de Microprocesadores (86.07)

TRABAJO PRÁCTICO OBLIGATORIO 1

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

Índice

1. Objetivos

El siguiente trabajo practico constituye una introducción a la programación de un microcontrolador. El trabajo practico consiste en utilizar el lenguaje de programación *Assembly* para llevar a cabo del encendido de un led a través de instrucciones al microcontrolador. Para este trabajo se utilizo el microcontrolador *atmega 328P* de la placa *Arduino* como fue recomendado por la cátedra. 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. Codigo de Montaje:

- Para la tarea de encendido de un led a través de la utilización de el puerto serie B se utilizo el siguiente código:

```
;Codigo para TP1

;Activar un led usando todo el puerto

START:
            LDI R16,0x00
            OUT PORTB,R16
            LDI R16,0xFF
            OUT DDRB,R16
    BLINK: LDI R16,0x01
            OUT PORTB,R16
             call DELAY
            LDI R16,0x00
            OUT PORTB,R16
             call DELAY
            RJMP BLINK

    DELAY:
      ldi   r18, 17
      ldi   r19, 60
      ldi   r20, 204
    L1:
      dec   r20
      brne L1
      dec   r19
      brne L1
      dec   r18
      brne L1
      ret
```

Podemos observar que la función principal es START. Esta comienzo con la instrucción LDI la cual carga al registro R16 con el valor 0. Este valor es direccionado al PORTB bajo la instrucción OUT con la intención de borrar cualquier valor anterior que contenga el PORTB. Luego se carga al registro R16 con el valor 0xFF lo cual completa cada bit del registro con un uno. Se direcciona ese valor al DDRB para setearlo como puerto de salida.
La notación "BLINK:"marca el inicio del loop de encendido y apagado en donde se envía un valor al puerto para el encendido y apagado. Se genera un retardo de tiempo para cada situación con la instrucción DELAY. La instrucción DELAY consta de dos registros cargados con un valor, los cuales son decrementados a cada llamada del loop y comparados con la instrucción BREC. Esta instrucción corrobora si su valor es nulo y en caso contrario llama al comando a su derecha.

- Para la tarea de encendido de un led a través de la utilización de un led en particular se utilizo el siguiente código:

```
        LDI R16,0x00
        OUT PORTB,R16
        LDI R16,0xFF
        OUT DDRB,R16
BLINK:  SBI PORTB,5
        call DELAY
        CBI PORTB,5
        call DELAY
        RJMP BLINK

DELAY:
    ldi   r18, 17
    ldi   r19, 60
    ldi   r20, 204
L1:
    dec   r20
    brne  L1
    dec   r19
    brne  L1
    dec   r18
    brne  L1
    ret
```

  Podemos apreciar que la funcionalidad y escritura del código es similar al caso anterior salvo por dos instrucciones. A diferencia de utilizar un registro para poder setear el valor del puerto se utilizaron los comando CBI y SBI. Estos comando fueron utilizados para poder modificar precisamente el led en cuestión.

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. En este caso se utilizo el microcontrolador *ATMEGA 328p*.

2.1.2. Calculo del retardo:

Considerando que la placa cuenta con una velocidad de reloj de 16MHz se procede a estimar el tiempo de retardo. La cantidad de ciclos realizada por el delay esta en 198604 ciclos ya que la instrucción LDI=1 ciclo, BRNE= 1ciclo si no cumple y 2 ciclos si cumple, RET=1 ciclo y NOP=1 ciclo.

```
DELAY:
    ldi   r18, 17          ; 1 ciclo
    ldi   r19, 60          ; 1 ciclo
    ldi   r20, 204         ; 1 ciclo
L1:
    dec   r20              ; 1 ciclo
    brne  L1               ; 1 ciclo si no cumple y 2 ciclos si cumple
    dec   r19              ; 1 ciclo
    brne  L1               ; 1 ciclo si no cumple y 2 ciclos si cumple
    dec   r18              ; 1 ciclo
    brne  L1               ; 1 ciclo si no cumple y 2 ciclos si cumple
    ret
```

Esto conlleva a calcular que:

$$16 * 256 * [(256 * 3) - 1 + 3] + 59 * [(256 * 3) - 1 + 3] + [(204 * 3) - 1 + 3] + 36 = 3200000 ciclos \quad (1)$$

Considerando un cristal de *clock* con frecuencia característica de $16MHz$, se estima un tiempo de retardo de:

$$t = \frac{3200000}{16000000} = 200ms \quad (2)$$

2.1.3. Montaje del código:

Para 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.

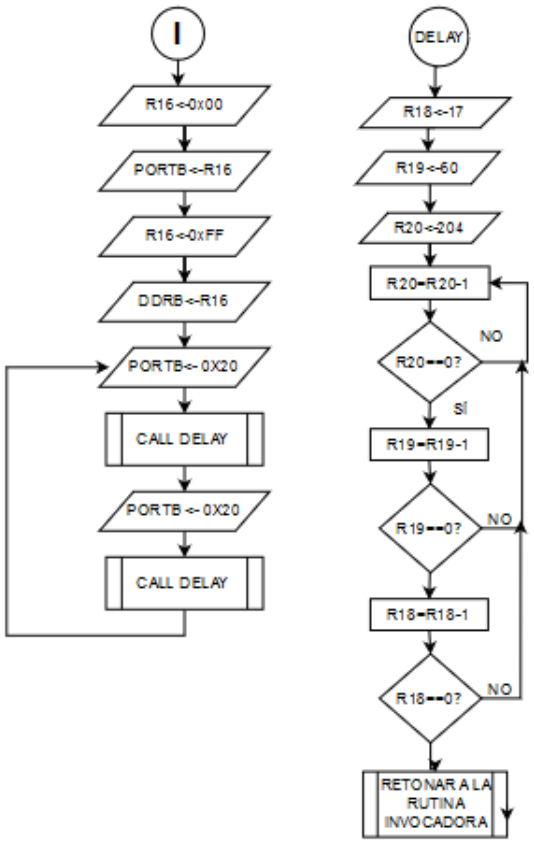

Figura 1: Diagrama de flujo de código utilizado para el puerto

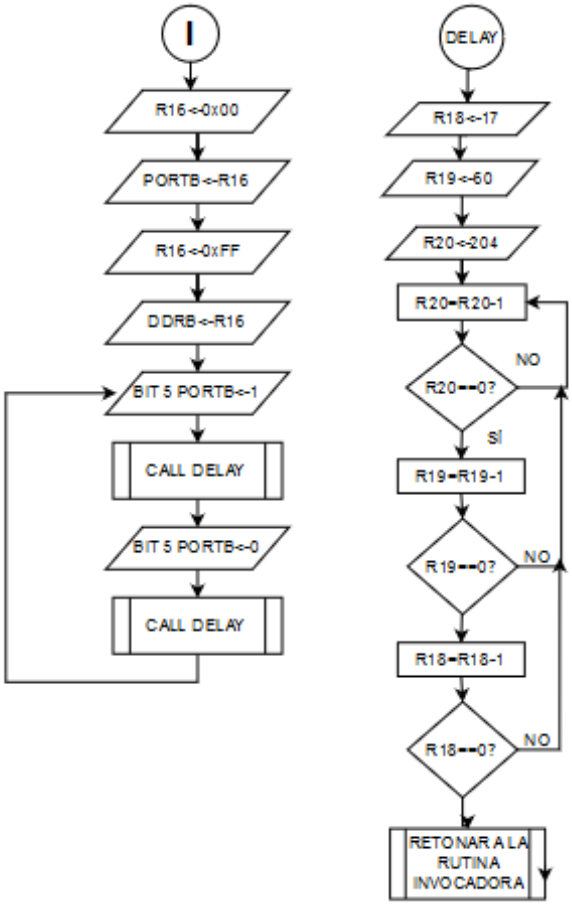

Figura 2: Diagrama de flujo de código utilizado para el bit.

2.2. Herramientas de Hardware:

Para realizar el proyecto se utilizaron el microcontrolador de la placa *Arduino Uno* y un led incorporado en la misma. Para la alimentación y trasmisión del código se utilizo el puerto USB que viene con la placa. Se puede visualizar la conexión interna de la placa en la siguiente imagen.

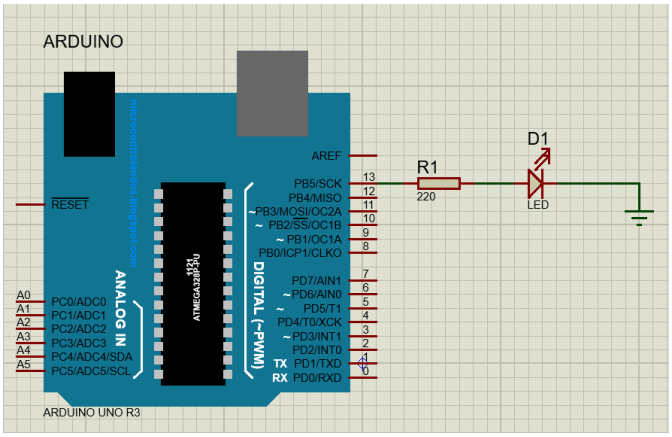

Figura 3: Diagrama de conexión del led.

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.

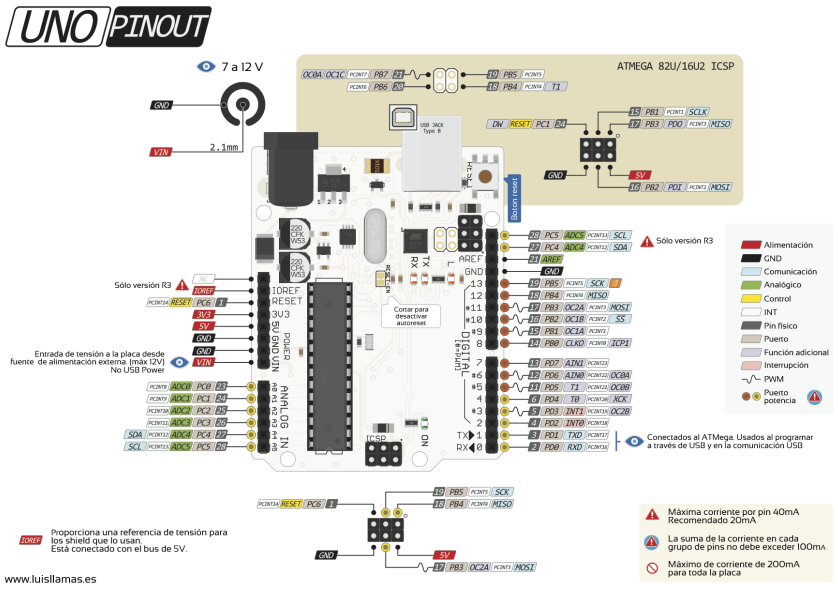

Figura 4: Diagrama de conexión de la placa.

2.2.1. Listado de componentes

Para este trabajo practico solo fue utilizad un cuyo costo en el mercado esta dentro del rango de $800 - 1500$.

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

Tabla 1: Precios de componentes.

2.3. Resultados

Para realizar la corrección se utilizo la instrucción CALL para generar el delay entre el encendido y apagado. El problema en el uso de RJMP es que el microcontrolador efectivamente salta a la direccion especificada, pero sin guardar nada en el stack. Luego, al terminar la subrutina con RET se popean 2 bytes del stack para obtener la direccion de retorno. El prolema es que esa direccion de retorno nunca se pusheo previamente, con lo cual se genera un problema con el stack, produciendo que la direccion de retorno sea aleatoria

3. Conclusiones

El principal problema surgió en el montaje del código a la placa, requiriendo de muchas horas de investigación de forma autodidacta al respecto. Pasado este problema, el código fue sencillo de realizar dada la bibliografía que explica de manera muy detallada los pasos a seguir en cuanto a la programación con *Assembly*. Con los objetivos del trabajo cumplidos, el mismo deja como enseñanza la tarea de montar un programa en un microcontrolador.

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