Laboratorio de Microprocesadores - 86.07

Trabajo Práctico Obligatorio N°2

Profesor:			Ing. Guillermo Campiglio								
Cuatrimestre/Año:			1°/2020								
Turno de las clases prácticas			Miércoles								
Jefe de trabajos prácticos:			Ing. Pedro Ignacio Martos								
Docente guía:			Ing. Fabricio Baglivo, Ing. Fernando Pucci								
Autores			Seguimiento del proyecto								
Nombre	Apellido	Padrón									
Ivan Eric	Rubin	100 577									

Observaciones:

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

Fecha de aprobación				Firma J.T.P

Coloquio	
Nota final	
Firma profesor	

1. Objetivo del trabajo

El objetivo de este trabajo es mantener un LED encendido mientras se mantiene presionado un botón utilizando puertos de entrada y salida del microprocesador.

2. Descripción del trabajo

Se logrará el objetivo definiendo el pin del botón como pin de entrada y el pin del LED como pin de salida. Se utilizarán ciclos para observar si se presiona el boton o si se deja de presionar, prendiendo y apgando el LED respectivamente.

3. Diagrama de conexiones en bloques

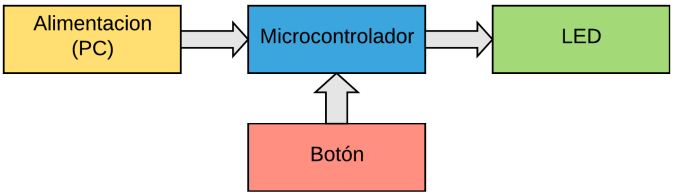

Figura 3.1: Diagrama de bloques.

4. Circuito Esquemático

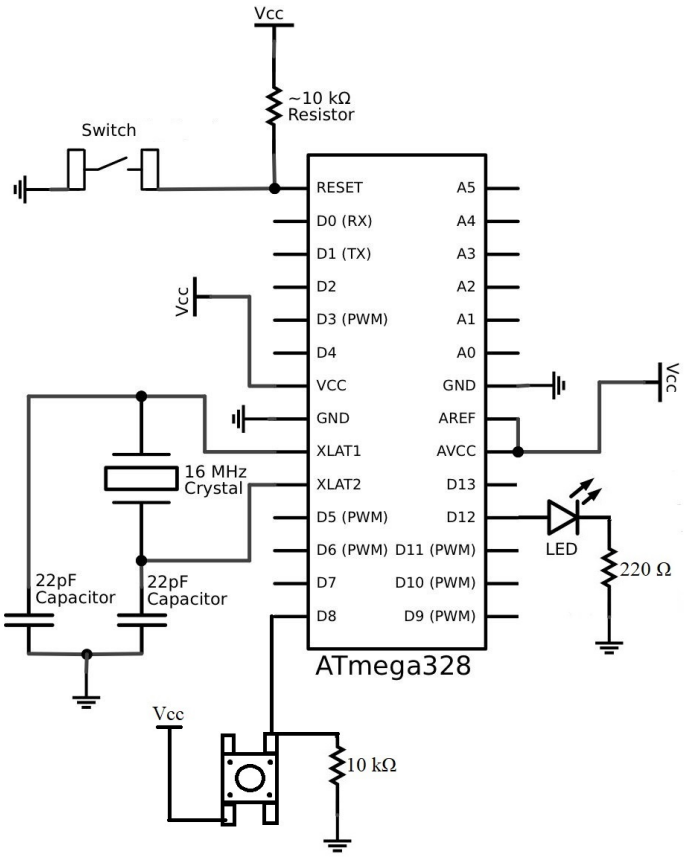

Figura 4.1: Circuito esquemático.

5. Listado de componentes

- Arduino UNO + Conector PC (864$).
- Microprocesador ATMEGA 328P (Integrado en Arduino).
- Resistencia de $220\Omega \pm 1\,\%$ y otra de $10k\Omega \pm 1\,\%$. (10$).
- Diodo Emisor de Luz (LED) (10$).
- Un pulsador (20$).
- Cables Macho-Macho (40 cables x 100$).
- Protoboard (200$).

6. Diagrama de flujo del software

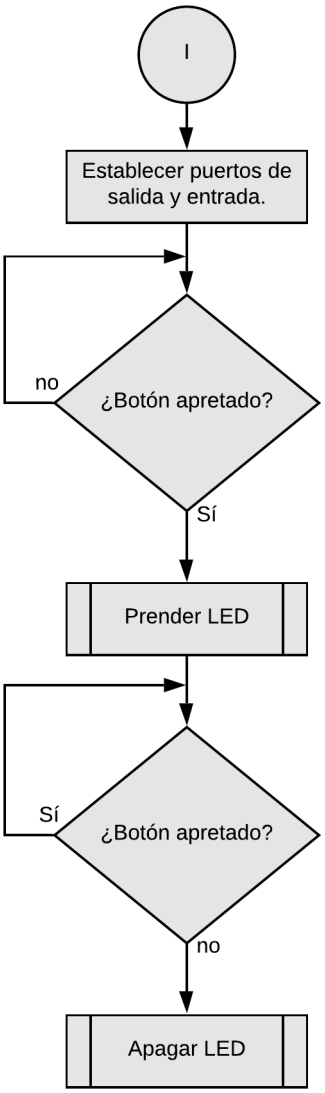

Figura 6.1: Diagrama de flujo.

7. Código de programa

```
/*
 *
   Trabajo Práctico Obligatorio 2
   Autor: Ivan Eric Rubin
 *
 */

.include "m328pdef.inc"
 /*
   Pin del botón (B) y Pin del LED (L).
 */
.EQU B = 0  ; Botón en el pin B0 (8 de Arduino)
.EQU L = 4  ; LED en el pin B4 (12 de Arduino)
/*
   Puertos
*/
.EQU PUERTO_SALIDA = PORTB
.EQU PUERTO_ENTRADA = PINB
.EQU CONF_PUERTO = DDRB

.CSEG ; A partir de aquí hay código

   JMP MAIN

.ORG INT_VECTORS_SIZE

MAIN:
/*
   Establezco entradas y salidas en el puerto
*/
   LDI R16, 0x30        ; En B5 quiero que el LED del Arduino esté apagado entonces
    lo voy a establecer como pin de salida
   OUT CONF_PUERTO, R16 ; Seteo los pines B4 y B5 como salida y el resto de los
   pines como entrada
/*
   Si se detecta un flanco ascendente se enciende el LED.
*/

ALTO:
   SBIS PUERTO_ENTRADA, B  ; Si se apreta el botón saltea la siguiente instrucción.
   JMP ALTO
   SBI PUERTO_SALIDA, L ; Se enciende el LED.
/*
   Luego de encenderse el LED se espera un flanco descendente.
*/
BAJO:
   SBIC PUERTO_ENTRADA, B  ; Si se suelta el botón saltea la siguiente instrucción.
   JMP BAJO
   CBI PUERTO_SALIDA, L ; Se apaga el LED.
   JMP ALTO          ; Se vuelve a la detección de flanco ascendente.
```

8. Resultados

Se logró detectar los flancos ascendentes y descendentes producidos al apretar el botón conectado al pin B0 del microprocesador para encender y apagar el LED conectado al pin B4.

Si se quisiera utilizar la resistencia de pullup interna de los puertos el pin de entrada estaría siempre

en un estado 1 lógico y al apretar el botón pasaría a low (0 lógico). Por lo tanto, en el programa habría que detectar los flancos descendentes para prender el LED y los ascendentes para apagarlo. También, habria que activar el pullup utilizando la instrucción SBI (Set Bit) en el pin de entrada del botón (En este caso B0).

Por otro lado, circuitalmente habría que cambiar (respecto al esquema anterior) Vcc por tierra en el botón y eliminar la resistencia de $10k\Omega$ ya que ya habrá una conectada a Vcc internamente. Esto resulta más económico ya que nos ahorrariamos esta resistencia para poder armar el circuito. Entonces quedaría el pin conectado al botón y este a tierra.

9. Conclusiones

Se puede destacar que en este trabajo se aprendió como es posible ahorrar una resistencia utilizando la interna de pullup del puerto para lograr un circuito mas económico y eficiente. También se llegó a un entendimiento mas profundo del uso de puertos y pines para poder detectar flancos y utilizar esa información para controlar otros pines de salida.