Laboratorio de Principios de Mecatrónica

11 de febrero de 2022

Práctica #1

Microcontroladores

Grupo:

L001

Estudiante:

- Bermúdez Guillen Alejandro
- Caballero Enciso Carla
- Montes de Oca Villa Luciano

Profesor:

Benito Granados–Rojas

Índice

1. Introducción

Un microcontrolador es un dispositivo programable, o un circuito integrado programable, capaz de realizar distintas tareas. Además, esta es la parte principal de una máquina o aplicación embedida. Básicamente, esta puede ser considerada una " pequeña computadora" capaz de controlar dispositivos de entrada y salida.

Los microcontroladores tienen 7 partes básicas que son: la memoria RAM, ROM, convertidores de datos, puertos de entrada y de salida, un CPU, comunicación serial y un temporizador. Desde nuestro punto de vista las partes más importantes son la memoria RAM y el CPU.

En el caso de este laboratorio vamos a usar el microcontrolador Arduino, el cual es uno de los más famosos y fácil de programar. En este caso, Arduino, es un proyecto de librería abierta, lo que hace que el software sea gratis y el hardware sea muy económico.

2. Experimentos y Simulaciones

2.1. Experimento 2.1 BLINK

Para este ejercicio, tenemos una entrada y una salida (input y output). Nuestro input es un botón (push button) y nuestro output es un LED. En este ejercicio lo que hicimos fue programar el arduino para que esté parpadeando el LED 1 segundo sí, 1 segundo no. Al momento de presionar el botón, lo que va a hacer es parpadear más lento, para esto lo que cambiamos fue el tiempo que hay entre el on y off a 2 segundos (.5 Hz). y una vez que se dejara de presionar el botón regresará a parpadear 1 segundo si, 1 segundo no.

2.2. Experimento 2.2 BLINK ASM

Al analizar el código notamos que el programa tiene la misma funcionalidad que en el ejercicio anterior, pero en lugar de usar Arduino se usa el lenguaje ensamblador. Es un programa que hace parpadear a un LED aproximadamente cada 0.18 segundos. Una diferencia notable es que el lenguaje ensamblador es mucho más complejo y se necesitan más instrucciones en comparación con Arduino.

Primero iniciamos el loop, luego con sbi se establece un bit en el registro de Entradas/Salidas que son el 5 y el 7. Después con call llamamos una subrutina. Así que vamos a la subrutina tiempo e indica que carguemos inmediatamente en el registro 22 el valor 45. Luego define tres bucles: el tercero, carga inmediatamente el valor 255 al registro 21; el segundo, carga inmediatamente el valor 255 al registro 20; y el primero, decrementa en una unidad el valor del registro 20.

Después, con BNRE se vuelve a regresar a LOOP1 hasta que valga cero y luego pasa a la siguiente instrucción. Y así sucesivamente con LOOP2 y LOOP3. Decrementa LOOP2 hasta que valga cero para después pasar a LOOP3 y, de igual forma, decrementarla hasta que también valga cero. Al final, se regresa al "programa principal" y se sale de la subrutina. Luego borra el bit en el registro de Entradas/Salidas

que son el 5 y el 7. Vuelve a llamar a la subrutina tiempo y cuando regresa salta a main.

Como el ATmega2560 opera a 16MHz, entonces su periodo de oscilaciòn es de 1/16MHz = 6.25x10 segundos. Como la subrutina hace 2,926,125 iteraciones (255x255x45), entonces el retardo es la multiplicación de las iteraciones por el periodo y obtenemos 0.1828 segundos.

A continuación mostramos el código que realizamos del LED con el botón en ensamblador.

```
void setup()
{
DDRB = DDRB | B10000000; // Data Direction Register B: Inputs 0-6, Output 7
}
void loop()
{
asm (
"inicio: \n\t"
"sbi 0x05,0x07 \n\t"            //push boton IN-5, LED OUT-7
\IN r0, 0x05 \n\t"              //lee el valor y carga en r0
\LDI r1, 0 \n\t"                //cargo Reg0 con 0
\cmp r0, r1 \n\t"              //compara los valores
\LDI r22, 90 \n\t"              //asigna 90 (si son iguales será mas lento)
"sbi 0x07 \n\t"                 //prende LED
\jne else \n\t"                 //sino se va a else y asigna 45 sera mas rapido
"call tiempo \n\t"
"cbi 0x07 \n\t"                 //apaga LED
"call tiempo \n\t"
"jmp main \n\t"
\else: \n\t"
\LDI r22, 45 \n\t"
"ret \n\t"

"tiempo: \n\t"
"LOOP_3: \n\t"
"LDI r21, 255 \n\t"
"LOOP_2: \n\t"
"LDI r20, 255 \n\t"
"LOOP_1: \n\t"
"DEC r20 \n\t"
"BRNE LOOP_1 \n\t"
"DEC r21 \n\t"
"BRNE LOOP_2 \n\t"
"DEC r22 \n\t"
"BRNE LOOP_3 \n\t"
"ret \n\t"
);
}
```

2.3. Experimento 2.3

Para este último ejercicio, conectamos los 3 LEDs para lograr armar los dos semáforos digitales y simular la sincronización de colores vista en la práctica.

Obtuvimos seis puertos digitales de salida y realizamos la programación del semáforo en lenguaje Arduino. El código documentado que lleva por nombre "Semaforoej3"se encuentra en GitHub (véase Ligas).

Esta sincronización de estados simulaba probablemente un cruce entre calles ya que, como podemos observar en la figura de más arriba, cuando el semáforo 1 permanece en rojo por 6 segundos, al mismo tiempo el semáforo 2 se mantiene en verde por 5 segundos y posteriomente en amarillo por 1 segundo. Lo anterior corresponde a los estados de los dos semáforos durante los primeros 6 segundos. Posteriormente, en los próximos 6 segundos, ocurre lo contrario ya que el semáforo 1 pasa a color verde durante 5 segundos y luego 1 segundo en amarillo. De manera simultánea, el semáforo 2 pasa a rojo durante los 6 segundos.

3. Conclusiones

La realización de esta práctica nos ayudó bastante a entender la importancia de los microcontroladores en la Mecatrónica pues un simple experimento como puede ser el prender un LED o algo más elaborado como armar un par de semáforos, aporta bastante conocimiento y permite conocer el funcionamiento de estos microcontroladores, ayudados por la programación de alto y bajo nivel. También, esta práctica nos sirvió para entender el inmenso mundo y las ramas que conforman a la Mecatrónica ya que esta rama se constituye de la Computación, los Circuitos Lógicos y de Control, la Electrónica, las Matemáticas, entre muchas otras.

4. Ligas

`https://github.com/Lucio27MV/Lab-Principios-Meca`

Referencias

HETPRO. (2020). Microcontrolador – qué es y para que sirve. 2022/02/09, de Hetpro Sitio web: https://hetpro-store.com/TUTORIALES/microcontrolador/

RJHcoding (2018) Working With I/O Registers. 2022/02/10, de RJHcoding Sitio web: http://www.rjhcoding.com/avr-asm-io.php

Rojo Café (2021). Delay en Ensamblador. 2022/02/10, de RJHcoding Sitio web: https://www.youtube.com/watch?v=zQ3av4G9Hzs

Support. (2021). Qué es un microcontrolador. 2022/02/09, de GSL industrias Sitio web: https://www.industriasgsl.com/blog/post/que$_e s_u n_m icrocontrolador$