INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

"Practica 9 Mostrar caracteres POLI display"

Equipo:

García García Marcos Ricardo

Rodríguez Tarango Christopher Alberto

Zamorano Aparicio José Eduardo

Grupo:

3CM10

Materia:

Introducción a los Microcontroladores

Profesor:

Paz Rodríguez Héctor Manuel

Escuela Superior de Cómputo

Contenido

Introducción

El ATMEGA8535 es un microcontrolador de 8 bits basado en la arquitectura RISC, el núcleo AVR combina un gran conjunto de instrucciones con 32 registros de propósito general. Los 32 registros están directamente conectados con la unidad aritmética-lógica (ALU), permitiendo que dos registros sean accesados en una sola instrucción ejecutada en un ciclo de reloj. Esta arquitectura permite que el microcontrolador sea más de diez veces más rápido que los microcontroladores tradicionales (microcontroladores CISC). En la siguiente imagen se muestra el microcontrolador ATMEGA8535.

Características del AT MEGA8535

- 8K bytes de memoria flash programable
- Memoria SRAM interna de 512 bytes
- 512 bytes en EEPROM
- USART (Universal Synchronous and Asynchronous serial Receiver and Transmitter).
- 32 líneas de entrada/salida de propósito general. Repartidas en 4puertos de 8 bits cada uno.
- Temporizadores/contadores con modo de comparación entre ellos.
- 8 conversores analógico digital de 10 bits de resolución.
- Interrupciones internas y externas.
- Un puerto serie SPI.
- Se le llama binarios, llamados así porque el conteo se realiza en códigos binarios. Los contadores son circuitos lógicos secuenciales que llevan la cuenta de una serie de pulsos de entrada de los retardos.

Display de 7 segmentos

El visualizador de siete segmentos es una forma de representar números en equipos electrónicos. Está compuesto de siete segmentos que se pueden encender o apagar individualmente. Cada segmento tiene la forma de una pequeña línea.

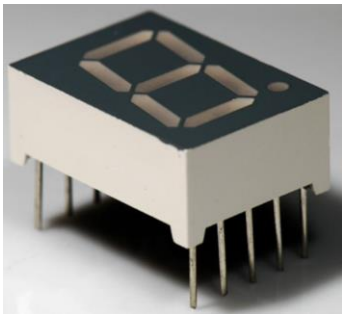

Fig. 1.0 Display de 7 segmentos.

Cada uno de los segmentos que forman la pantalla está marcado con ocho primeras letras del alfabeto ('a'-'g'), y se montan de forma que permiten activar cada segmento por separado, consiguiendo formar cualquier dígito numérico.

Los hay de dos tipos: ánodo común y cátodo común.

En los de tipo de cátodo común, todos los cátodos de los led`s o segmentos están unidos internamente a una patilla común que debe ser conectada a potencial negativo (nivel “0”). El encendido de cada segmento individual se realiza aplicando potencial positivo (nivel “1”) por la patilla correspondiente a través de una resistencia que límite el paso de la corriente.

Objetivo

Realizar un programa que muestre los caracteres POLI por medio de un display de 7 segmentos.

Material

- Tarjeta Pazuino.
- Computadora personal.
- Display de 7 segmentos de cátodo común.
- Alambre de conexión.
- Resistencia 100 ohms

Desarrollo

Comenzamos realizando una tabla para obtener el dato en hexadecimal de cada uno de los caracteres que vamos a mostrar en el display de 7 segmentos.

dp	g	f	e	d	c	b	a	Caracter	Dato
0	1	1	1	0	0	1	1	P	$73
0	0	1	1	1	1	1	1	O	$3F
0	0	1	1	1	0	0	0	L	$38
0	0	0	0	0	1	1	0	I	$06

Tabla 1. Dato hexadecimal de cada carácter que se mostrará en el display.

Código fuente

```
                ;Muestra Caracteres POLI

                .INCLUDE "M8535DEF.INC"
                .CSEG
                .ORG $0
                RJMP INICIO             ;SALTA A VECTORES DE INTERRUPCION
                .ORG $015
Inicio:
                LDI R16,LOW (RAMEND)        ;INICIALIZAMOS STACK POINTER
                OUT SPL, R16                ; INICIALIZAMOS STACK POINTER
                LDI R16, HIGH (RAMEND)      ; INICIALIZAMOS STACK POINTER
                OUT SPH,R16                 ; INICIALIZAMOS STACK POINTER

                LDI R16, $FF                ;R16 -> $FF
                OUT DDRB, R16               ;PUERTO B COMO SALIDA
Loop:
                LDI R16, $73                ;CARGAR DATO $73 EN R16 CARÁCTER P
                OUT PORTB, R16              ;MOSTRAR DATO EN EL PUERTOB
                RCALL DELAY                 ;LLAMAR AL RETARDO
                LDI R16, $3F                ;CARGAR DATO $73 EN R16 CARÁCTER P
                OUT PORTB, R16              ;MOSTRAR DATO EN EL PUERTOB
                RCALL DELAY                 ;LLAMAR AL RETARDO
                LDI R16, $38                ;CARGAR DATO $73 EN R16 CARÁCTER P
                OUT PORTB, R16              ;MOSTRAR DATO EN EL PUERTOB
                RCALL DELAY                 ;LLAMAR AL RETARDO
                LDI R16, $06                ;CARGAR DATO $73 EN R16 CARÁCTER P
                OUT PORTB, R16              ;MOSTRAR DATO EN EL PUERTOB
                RCALL DELAY                 ;LLAMAR AL RETARDO
                RJMP LOOP           ;SALTAR A LOOP PARA QUE LO HAGA CONTINUAMENTE
DELAY:
```

```
DEC R20                 ;DECREMENTAR R20
BRNE DELAY              ;LLAMAR AL RETARDO
DEC R21                 ; DECREMENTAR R21
BRNE DELAY              ;LLAMAR AL RETARDO
DEC R22                 ; DECREMENTAR R21
BRNE DELAY              ;LLAMAR AL RETARDO
RET                     ;FIN DE SUBRUTINA
```

Conclusión

Realizamos retardos por medio de instrucciones, decrementando registros de propósito general en vez de usar Timer0, y también aprendimos a enviar información o datos por el puerto b para mostrarlos en un display de 7 segmentos por medio de las combinaciones binarias de cada letra y enviándolas por hexadecimal.