INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

Introducción a los microcontroladores

Profesor: Aguilar Sánchez Fernando

Práctica 02 "Contador Ascendente/Descendente"

Integrantes del Equipo:

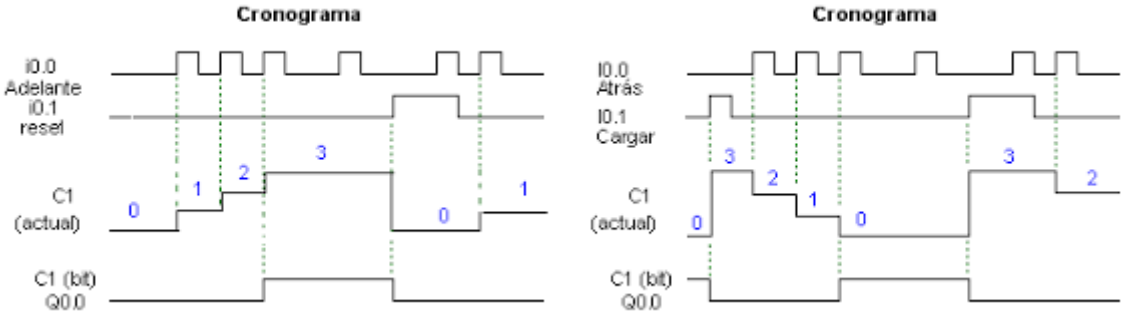
- **Arizmendi Alvarado Brian**
- **Rodríguez López Ricardo**

Objetivo

Al término de la sesión, los integrantes del equipo contarán con la habilidad de realizar un contador descendente y ascendente utilizando funciones de retardo.

Introducción Teórica

Los contadores ascendentes/descendentes son circuitos electrónicos que se utilizan para contar impulsos eléctricos en una dirección específica. El contador ascendente cuenta los impulsos en orden creciente, es decir, desde cero hasta un valor máximo establecido. Por otro lado, el contador descendente cuenta los impulsos en orden decreciente, es decir, desde un valor máximo establecido hasta cero.



Estos contadores son muy utilizados en la electrónica digital para llevar el control de eventos o procesos que necesitan ser contabilizados, como el número de piezas producidas en una línea de ensamblaje, el número de veces que se pulsa un botón, entre otros. Además, los contadores ascendentes/descendentes también se pueden utilizar en combinación con otros circuitos para implementar diversas funciones lógicas, como la multiplicación o la división.

Los contadores ascendentes/descendentes pueden ser implementados de diversas formas, desde circuitos discretos hasta sistemas integrados en un solo chip, como los microcontroladores. En general, estos circuitos utilizan flip-flops para almacenar y cambiar el valor del contador y puertas lógicas para controlar el flujo de los impulsos.

En esta práctica, se diseñará y programará un contador ascendente-descendente utilizando un microcontrolador Atmega8535.

El Atmega8535 es un microcontrolador de la familia AVR de Atmel, que cuenta con 8K de memoria flash programable, 512 bytes de memoria EEPROM y 512 bytes de memoria RAM. Este microcontrolador es muy popular debido a su bajo costo, facilidad de programación y amplia disponibilidad de herramientas de desarrollo.

Para implementar el contador ascendente-descendente, se utilizarán los puertos de entrada/salida del Atmega8535, y se programará en lenguaje C

Materiales y Equipo empleado

- CodeVision AVR
- AVR Studio 4
- Microcontrolador ATmega 8535
- 16 LED’s
- 16 Resistores de 330 Ω a 1⁄4 W

Código

```
// I/O Registers definitions
#include <mega8535.h>
#include <delay.h>
// Declare your global variables here

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
DDRA=(0<<DDA7) | (0<<DDA6) | (0<<DDA5) | (0<<DDA4) | (0<<DDA3) | (0<<DDA2) | (0<<DDA1) | (0<<DDA0);
// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
PORTA=(0<<PORTA7) | (0<<PORTA6) | (0<<PORTA5) | (0<<PORTA4) | (0<<PORTA3) | (0<<PORTA2) | (0<<PORTA1) | (0<<PORTA0);

// Port B initialization
// Function: Bit7=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out Bit1=Out Bit0=Out
DDRB=(1<<DDB7) | (1<<DDB6) | (1<<DDB5) | (1<<DDB4) | (1<<DDB3) | (1<<DDB2) | (1<<DDB1) | (1<<DDB0);
// State: Bit7=0 Bit6=0 Bit5=0 Bit4=0 Bit3=0 Bit2=0 Bit1=0 Bit0=0
PORTB=(0<<PORTB7) | (0<<PORTB6) | (0<<PORTB5) | (0<<PORTB4) | (0<<PORTB3) | (0<<PORTB2) | (0<<PORTB1) | (0<<PORTB0);

// Port C initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In
DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (0<<DDC3) | (0<<DDC2) | (0<<DDC1) | (0<<DDC0);
// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T
PORTC=(0<<PORTC7) | (0<<PORTC6) | (0<<PORTC5) | (0<<PORTC4) | (0<<PORTC3) | (0<<PORTC2) | (0<<PORTC1) | (0<<PORTC0);

// Port D initialization
// Function: Bit7=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out Bit1=Out Bit0=Out
DDRD=(1<<DDD7) | (1<<DDD6) | (1<<DDD5) | (1<<DDD4) | (1<<DDD3) | (1<<DDD2) | (1<<DDD1) | (1<<DDD0);
// State: Bit7=0 Bit6=0 Bit5=0 Bit4=0 Bit3=0 Bit2=0 Bit1=0 Bit0=0
PORTD=(0<<PORTD7) | (0<<PORTD6) | (0<<PORTD5) | (0<<PORTD4) | (0<<PORTD3) | (0<<PORTD2) | (0<<PORTD1) | (0<<PORTD0);
```

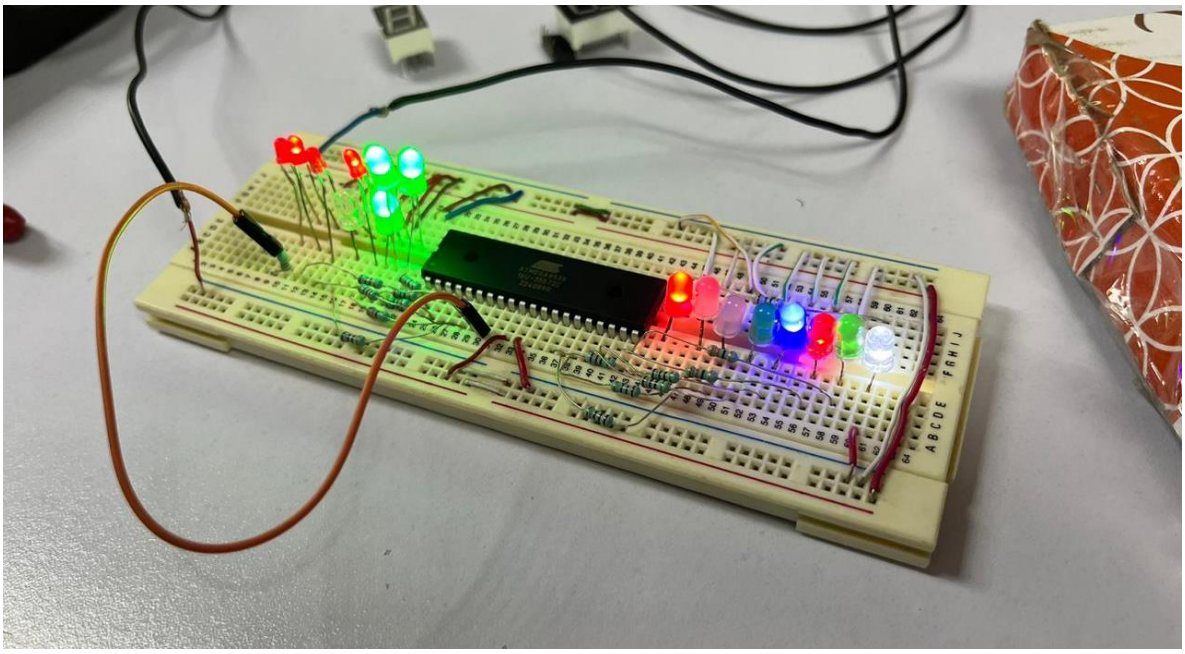
```
 // Port D initialization
 // Function: Bit7=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out Bit1=Out Bit0=Out
 DDRD=(1<<DDD7) | (1<<DDD6) | (1<<DDD5) | (1<<DDD4) | (1<<DDD3) | (1<<DDD2) | (1<<DDD1) | (1<<DDD0);
 // State: Bit7=0 Bit6=0 Bit5=0 Bit4=0 Bit3=0 Bit2=0 Bit1=0 Bit0=0
 PORTD=(0<<PORTD7) | (0<<PORTD6) | (0<<PORTD5) | (0<<PORTD4) | (0<<PORTD3) | (0<<PORTD2) | (0<<PORTD1) | (0<<PORTD0);

 // Timer/Counter 0 initialization
 // Clock source: System Clock
 // Clock value: Timer 0 Stopped
 // Mode: Normal top=0xFF
 // OC0 output: Disconnected
 TCCR0=(0<<WGM00) | (0<<COM01) | (0<<COM00) | (0<<WGM01) | (0<<CS02) | (0<<CS01) | (0<<CS00);
 TCNT0=0x00;
 OCR0=0x00;
```

```
TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) | (0<<COM1B0) | (0<<WGM11) | (0<<WGM10);
TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<WGM13) | (0<<WGM12) | (0<<CS12) | (0<<CS11) | (0<<CS10);
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// OC2 output: Disconnected
ASSR=0<<AS2;
TCCR2=(0<<WGM20) | (0<<COM21) | (0<<COM20) | (0<<WGM21) | (0<<CS22) | (0<<CS21) | (0<<CS20);
TCNT2=0x00;
OCR2=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=(0<<OCIE2) | (0<<TOIE2) | (0<<TICIE1) | (0<<OCIE1A) | (0<<OCIE1B) | (0<<TOIE1) | (0<<OCIE0) | (0<<TOIE0);
```

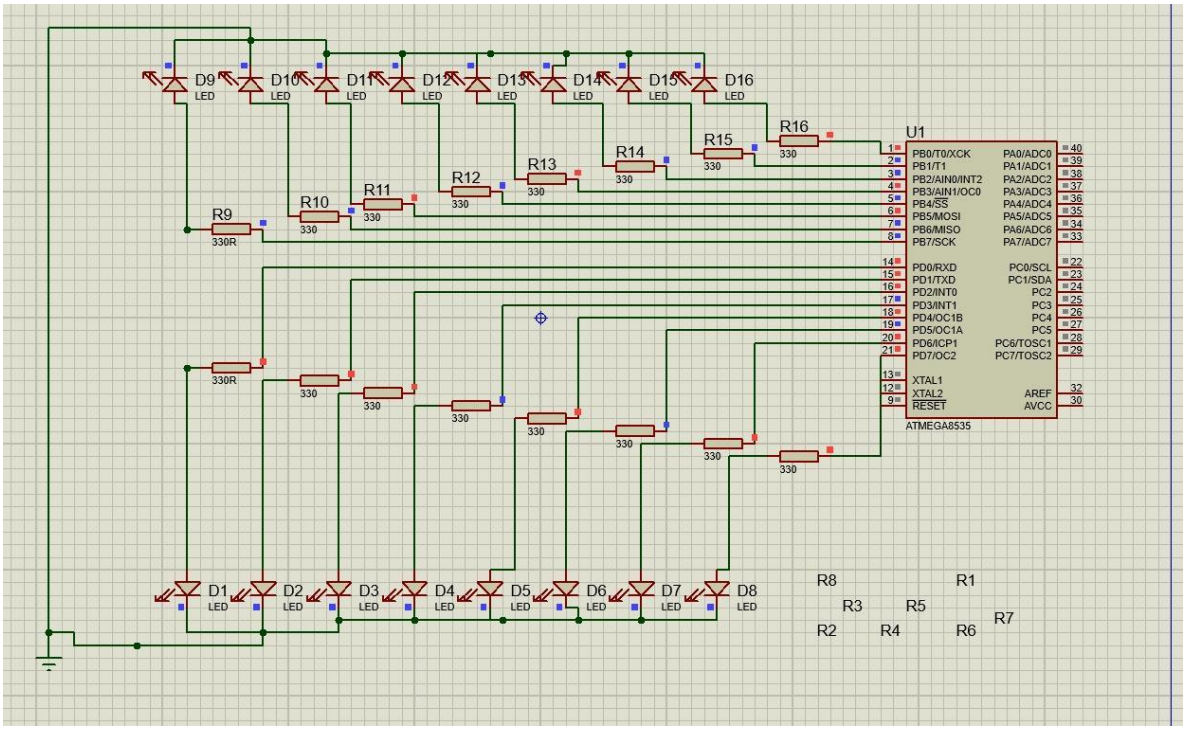
```
// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=(0<<OCIE2) | (0<<TOIE2) | (0<<TICIE1) | (0<<OCIE1A) | (0<<OCIE1B) | (0<<TOIE1) | (0<<OCIE0) | (0<<TOIE0);

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=(0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);
MCUCSR=(0<<ISC2);

// USART initialization
// USART disabled
UCSRB=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) | (0<<RXEN) | (0<<TXEN) | (0<<UCSZ2) | (0<<RXB8) | (0<<TXB8);

// Analog Comparator initialization
// Analog Comparator: Off
// The Analog Comparator's positive input is
// connected to the AIN0 pin
// The Analog Comparator's negative input is
// connected to the AIN1 pin
ACSR=(1<<ACD) | (0<<ACBG) | (0<<ACO) | (0<<ACI) | (0<<ACIE) | (0<<ACIC) | (0<<ACIS1) | (0<<ACIS0);
SFIOR=(0<<ACME);

// ADC initialization
// ADC disabled
ADCSRA=(0<<ADEN) | (0<<ADSC) | (0<<ADATE) | (0<<ADIF) | (0<<ADIE) | (0<<ADPS2) | (0<<ADPS1) | (0<<ADPS0);
```

```
// TWI initialization
// TWI disabled
TWCR=(0<<TWEA) | (0<<TWSTA) | (0<<TWSTO) | (0<<TWEN) | (0<<TWIE);

while (1)
      {
      // Place your code here
            delay_ms(500);

            PORTB++;
            PORTD--;
      }
}
```

Pruebas



Simulación



Conclusiones

Arizmendi Alvarado Brian

En conclusión, los contadores ascendentes/descendentes son circuitos electrónicos que se utilizan frecuentemente en la electrónica digital, son fundamentales para llevar el control de eventos y procesos.

El diseño y programación de estos contadores se puede realizar utilizando nuestro microcontrolador y un conjunto de instrucciones sencillas programadas en C.

Juegan un papel crucial en numerosas aplicaciones y sistemas.

Rodríguez López Ricardo

En esta práctica pudimos darnos cuenta de cómo se utilizan dos salidas diferentes que trabajan independientemente, lo cual es algo básico que tenemos que saber, ya que es la importancia o es para lo que fueron creados los microcontroladores, poder hacer cosas simultáneamente.

Referencias Bibliográficas:

Anónimo. (s.f.). CONTADORES. Unican. https://personales.unican.es/manzanom/planantiguo/edigitali/CONTG1.pdf

Muñoz P. (2020, 4 de Agosto). CONTADORES ASCENDENTES-DESCENDENTES. *Prezi.* https://prezi.com/p/lfdf4mi6fg04/contadores-ascendentes-descendentes/