INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

PRÁCTICA 1: USO DE PUERTOS E/S

ALUMNO: MALAGON BAEZA ALAN ADRIAN
MARTINEZ CHAVEZ JORGE ALEXIS

GRUPO: 6CM3

U.A: SISTEMAS EN CHIP

PROFESOR: FERNANDO AGUILAR SÁNCHEZ

FECHA DE ENTREGA: 15 DE ABRIL DE 2023

OBJETIVO

Al término de la sesión, los integrantes del equipo contaran con la habilidad de programar los puertos como entrada y salida del Microcontrolador ATmega8535 usando las herramientas “Code Vision AVR” y “AVR Studio 4”.

INTRODUCCIÓN TEÓRICA

MICROCONTROLADOR

Es un equipo con las mismas características de una computadora, solo que su tamaño es más pequeño. Tiene un CPU (Central Processing Unit) por sus siglas en inglés, una memoria RAM y una memoria ROM. Es el cerebro de un sistema informático y el motor que activa el funcionamiento de un equipo.[1]

Su utilidad está presente en muchas áreas de la vida cotidiana, en la industria cumple una tarea fundamental, ya que es utilizado como complemento en la automatización de diversas operaciones.[1]

Con un microcontrolador tenemos la posibilidad de realizar múltiples tareas, tales como la administración de entrada y salida en un proceso informático determinado. En el sector industrial es frecuente ver su aplicación en controladores y otros sistemas de automatización que detallaremos más adelante.[1]

Por otro lado, en la mayoría de los dispositivos tecnológicos que usamos está presente el microcontrolador. Ellos hacen posible el funcionamiento de ordenadores, celulares, calculadoras, laptop, relojes, alarmas, entre otros equipos.[1]

MICROCONTROLADOR ATMEGA8535

Es un microcontrolador fabricado por la empresa Microchip Technology Inc. Posee las siguientes características:[2]

- Microcontrolador AVR de 8 bit de alto rendimiento y bajo consumo.
- Arquitectura RISC avanzada.
    - 130 instrucciones. La mayoría de un simple ciclo de clock de ejecución.
    - 32 x 8 registros de trabajo de propósito general.
    - Capacidad de procesamiento de unos 16 MIPS a 16 MHz.
    - Funcionamiento estático total.
    - Multiplicador On-Chip de 2 ciclos
- I/O y encapsulados
    - 32 líneas de I/O programables.
    - PDIP de 40 pines, TQFP y MLF de 44 pines.
- Tensiones de funcionamiento.
    - 2.7 - 5.5V (ATmega8535L).
    - 4.5 - 5.5V (ATmega8535).
- Niveles de velocidad.
    - 0 - 8 MHz (ATmega8535L).
    - 0 - 16 MHz (ATmega8535).
- Características especiales del microcontrolador.

- Reset de Power-on y detección de Brown-out programable.
- Oscilador RC interno calibrado.
- Fuentes de interrupción externas e internas.
- 6 modos de descanso: Idle, reducción de ruido ADC, Power-save, Power-down, Standby y Standby extendido.

CODEVISIONAVR

CodeVisionAVR es un IDE (Entorno de Desarrollo Integrado) desarrollado por la empresa HP InfoTech. Integra un compilador de lenguaje C para los microcontroladores Microchip AVR. Además, integra un generador automático de programa (CodeVisionWizard) para los chips AVR8, AVR8X, AVR DA, AVR DB y XMEGA, el cual nos ayuda a configurar los puertos de estos microcontroladores para su correcto funcionamiento sin que el usuario tenga que programar el comportamiento de los puertos.[3]

CodeVisionAVR nos es útil para poder desarrollar programas en lenguaje C para microcontroladores con el fin de obtener el archivo de tipo hexadecimal (.hex) el cuál podremos utilizar para cargarlo a nuestro microcontrolador o bien cargarlo a un circuito simulado por computadora en simuladores como Proteus para probar su funcionamiento previamente antes de armar el circuito físico.

MATERIALES Y EQUIPO EMPLEADO

- ✓ CodeVision AVR
- ✓ AVR Studio 4
- ✓ Microcontrolador ATmega8535
- ✓ 8 LED’s
- ✓ 8 resistores de 330Ω a ¼ W
- ✓ 1 Dip switch u 8 Push Botón

DESARROLLO EXPERIMENTAL

1. Realiza un programa para programar el Puerto B como entrada y escribir la información en el Puerto D activándolo como salida, recuerde activar las resistencias de Pull-up del puerto B para colocar solo el Dipswitch.

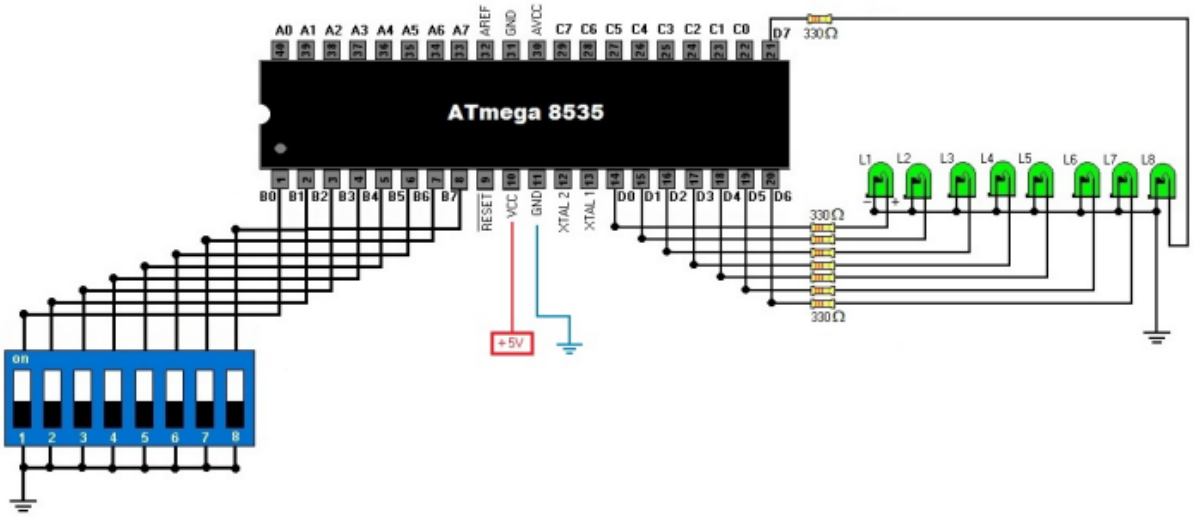

Figura 1. Puerto B como entrada, Puerto D como salida

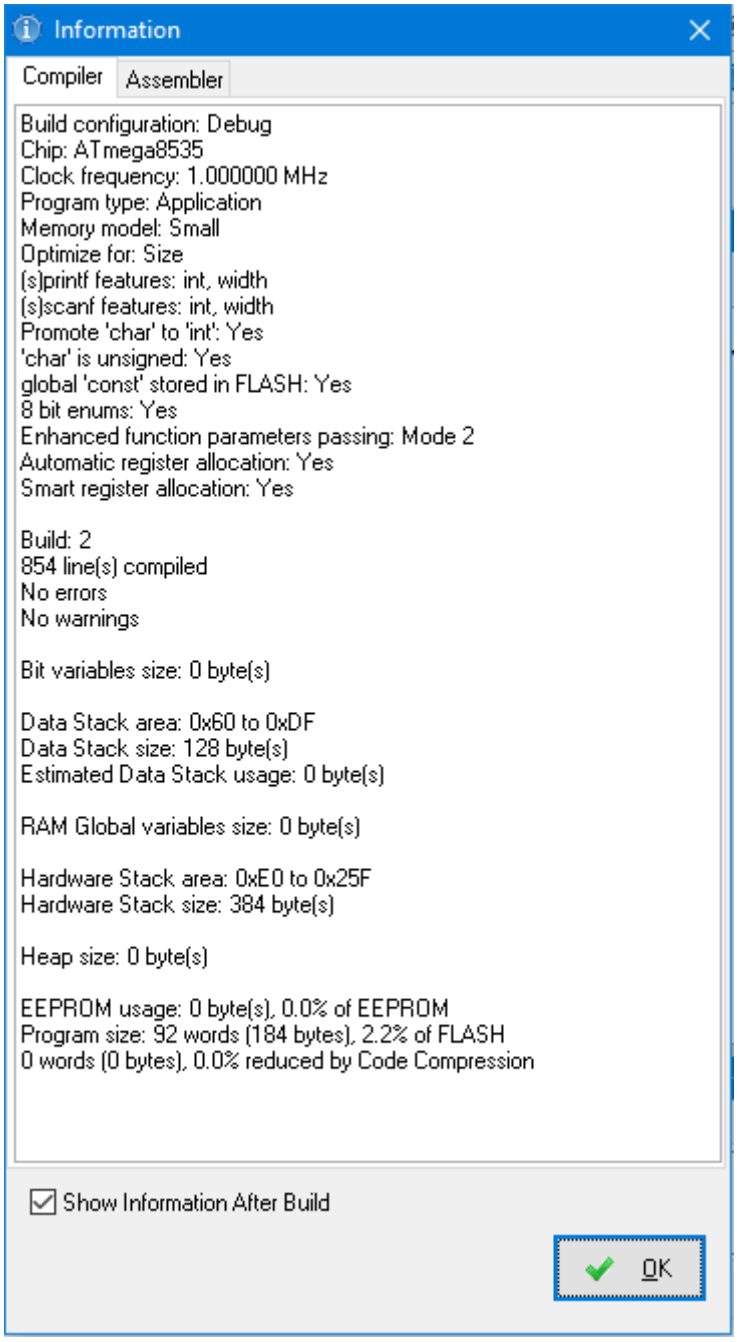

Figura 2. Compilación exitosa del programa desarrollado en CodeVisionAVR

CÓDIGO GENERADO POR CODEVISION

```
/*****************************************************
This program was created by the CodeWizardAVR V3.47
Automatic Program Generator
© Copyright 1998-2021 Pavel Haiduc, HP InfoTech S.R.L.
http://www.hpinfotech.ro

Project :
Version :
Date    :
Author  :
Company :
Comments:


Chip type               : ATmega8535
Program type            : Application
AVR Core Clock frequency: 1.000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 128
*****************************************************/

#include <mega8535.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=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In
Bit0=In
DDRB=(0<<DDB7) | (0<<DDB6) | (0<<DDB5) | (0<<DDB4) | (0<<DDB3) |
(0<<DDB2) | (0<<DDB1) | (0<<DDB0);
// State: Bit7=P Bit6=P Bit5=P Bit4=P Bit3=P Bit2=P Bit1=P Bit0=P
PORTB=(1<<PORTB7) | (1<<PORTB6) | (1<<PORTB5) | (1<<PORTB4) |
```

```
(1<<PORTB3) | (1<<PORTB2) | (1<<PORTB1) | (1<<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);

// 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;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Disconnected
// OC1B output: Disconnected
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
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);

// 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);

// SPI initialization
```

```
// SPI disabled
SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) | (0<<CPOL) |
(0<<CPHA) | (0<<SPR1) | (0<<SPR0);

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

while (1)
      {
      // Place your code here
        PORTD = PINB;
      }
}
```

CIRCUITO ELECTRICO EN PROTEUS

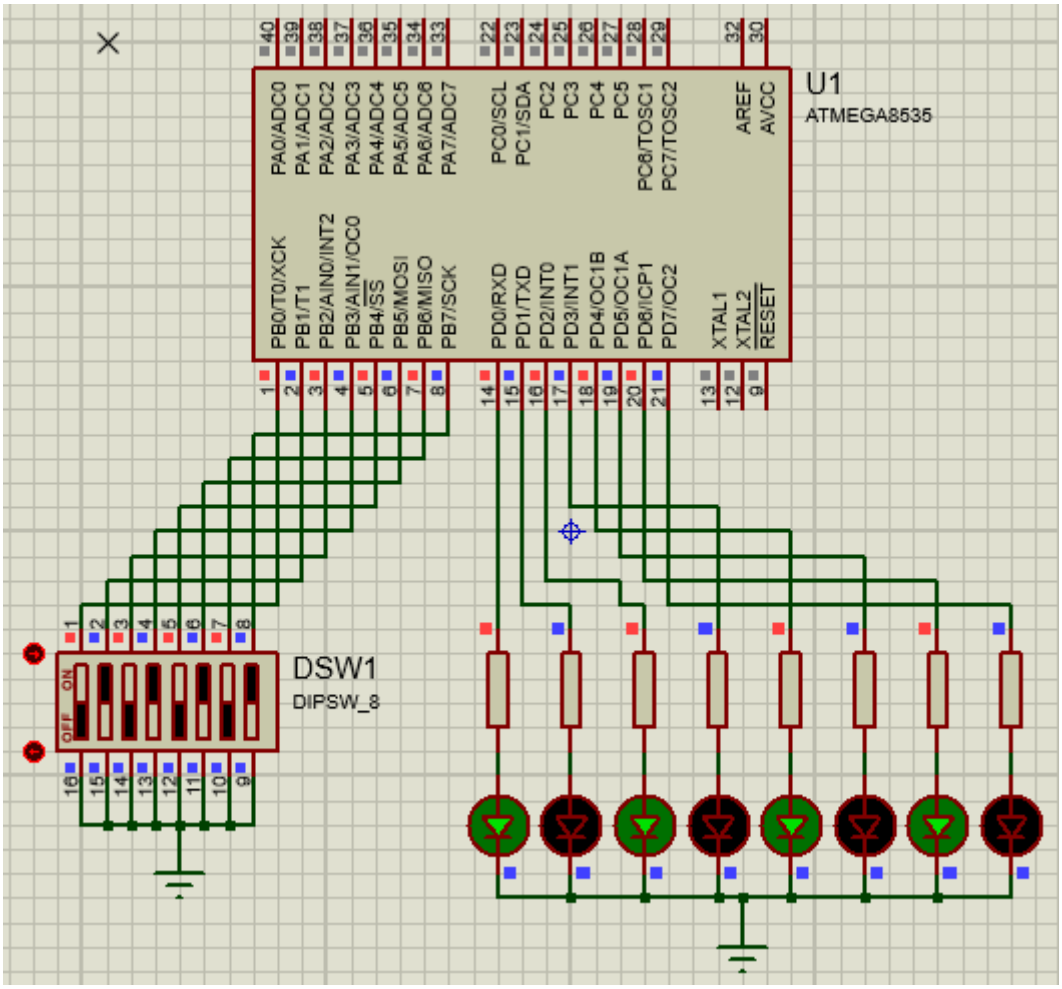

Figura 3. Circuito simulado en Proteus

CIRCUITO EN EL PROTO ARMADO

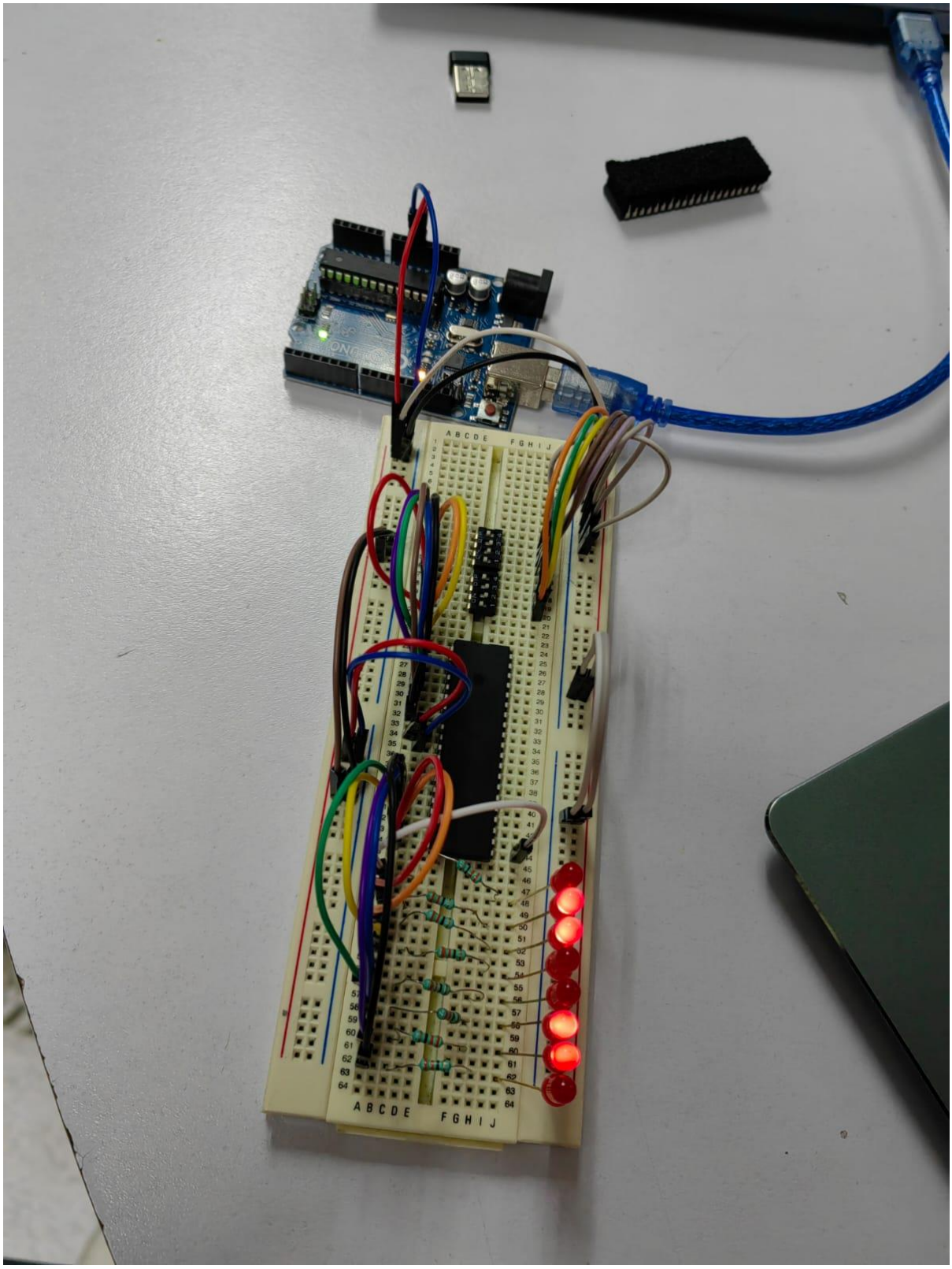

Figura 4. Circuito armado en la onoi

OBSERVACIONES Y CONCLUSIONES INDIVIDUALES

Malagón Baeza Alan Adrián

Los microcontroladores son dispositivos que juegan un papel muy importante en las máquinas con las que interactuamos día con día; ya sea el refrigerador, el estéreo del carro, la lavadora, o incluso nuestros audífonos inalámbricos; los microcontroladores son los dispositivos capaces de dar las instrucciones necesarias a los dispositivos que lo integran para poder realizar sus tareas de forma correcta. En el mercado existen muchas marcas y modelos de microcontroladores los cuales tienen distintas especificaciones; es trabajo de nosotros el investigar qué microcontrolador es el adecuado para la realización del proyecto que tengamos en mente, puesto que de nada sirve tener un microcontrolador caro y potente si la aplicación que necesitamos no requiere de un alto procesamiento. El microcontrolador ATMEGA8535 posee la característica de tener un alto rendimiento y un bajo consumo, además de tener un costo bajo, lo que lo hace el microcontrolador idóneo para poder realizar muchas aplicaciones sin gastar mucho dinero.

Martínez Chávez Jorge Alexis

Los microcontroladores requieren de un programa desarrollado en lenguaje C y compilado en un archivo de tipo hexadecimal (.hex) para poder cargar las instrucciones que se requieren para el funcionamiento del microcontrolador. Para poder desarrollar y compilar el código de nuestros programas podemos hacer uso del IDE CodeVisionAVR, el cual nos ofrece un generador automático de programa para poder configurar los puertos del microcontrolador con el fin de facilitar el trabajo del programador. Utilizar CodeVisionAVR es sumamente fácil y muy intuitivo, lo que ayuda a un fácil desarrollo de programas para el microcontrolador que vamos a utilizar a lo largo del semestre.

El interactuar entre CodeVisionAVR y Proteus facilita el uso virtual del microcontrolador ATMEGA8535, lo cual será de mucha ayuda para desarrollar todas las prácticas programadas para este semestre.

BIBLIOGRAFÍA

[1] Industrias GSL, “Industriasgsl.com / Venta de Suministros Industriales,” *Industrias GSL*, 2021. https://www.industriasgsl.com/blog/post/que_es_un_microcontrolador (accessed Feb. 06, 2022).
[2] “ATmega8535,” *Sc.ehu.es*, 2022. http://www.sc.ehu.es/sbweb/webcentro/automatica/web_avr/archivos/Otros%20AVRs/ATmega/ATmega8535.htm (accessed Feb. 06, 2022).
[3] HP InfoTech, “HP InfoTech - CodeVisionAVR C Compiler,” *www.hpinfotech.ro*, 2021. http://www.hpinfotech.ro/ (accessed Feb. 06, 2022).