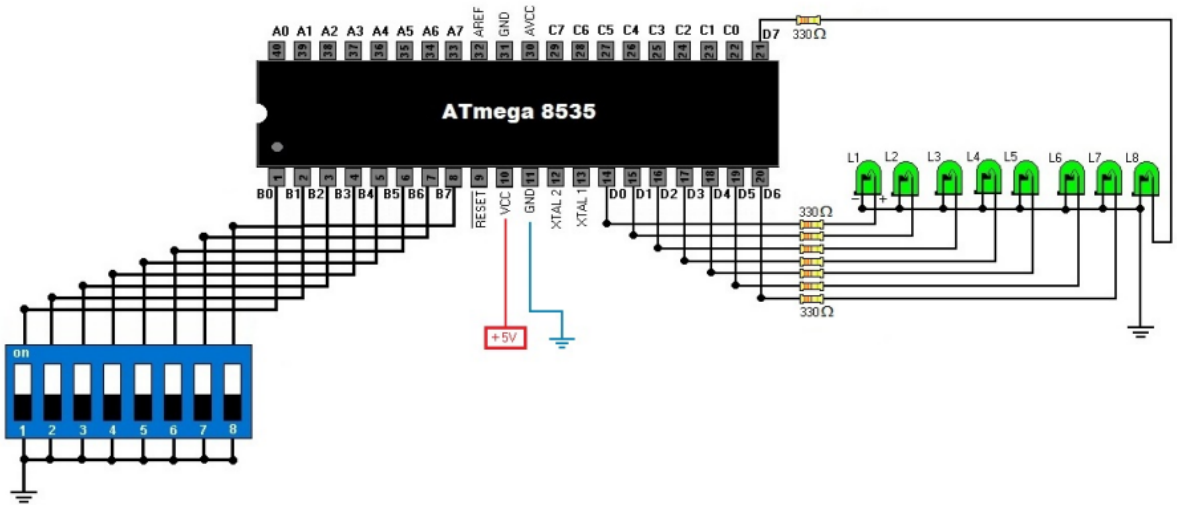
INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

SISTEMAS EN CHIP

PRACTICA No. 01

"Puertos de E/S"



ALUMNO: RENTERIA ARRIAGA JOSUE.

GRUPO: 6SCM1

PROFESOR: FERNANDO AGILAR SÁNCHEZ.

23/OCTUBRE/ 2021

I. OBJETIVO GENERAL:

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”

II. MATERIAL Y EQUIPO EMPLEADO

- ✓ Code Vision AVR
- ✓ AVR Studio 4
- ✓ Microcontrolador ATmega 8535
- ✓ 8 LED’s
- ✓ 8 resistores de 330 Ω a ¼ W
- ✓ 1 Dip switch u ocho Push Botón

III. INTRODUCCIÓN TEÓRICA

En esta práctica podremos saber utilizar correctamente y aplicar la implementación del código en el lenguaje de programación C de los conceptos de Entrada y Salida, codificación y creación del circuito para poder dar solución a la práctica.

En esta práctica se estudiará la implementación de un Programa y Circuito para programar el puerto B como entrada y escribir la información en el puerto D activándolo como salida. Los datos de entrada están dados por un DipSwitch. En este caso de la práctica se ocuparán algunos conceptos obtenidos anteriormente en el curso como lo es la implementación de código en C para poder programar nuestro microcontrolador, algunos conceptos de Fundamentos de Diseño Digital y de Electrónica Analógica para poder realizar correctamente el armado de nuestro circuito y que este funcione correctamente.

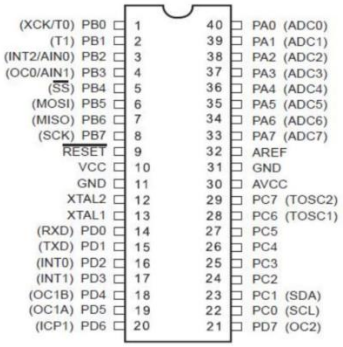
Ocuparemos algunos elementos como lo dice el formato de la practica ya que ocuparemos el simulador Code Vision para poder implementar el código para después programar nuestro ATMega8535, en mi caso se ocupará el programa Khazama AVR con su programador para poder programar nuestro ATMega con el archivo .hex que se generó anteriormente. Para la parte del armado del circuito será algo sencillo ya que solo se implementarán Leds para los puertos de salida y un DipSwitch para el puerto de las entradas.

Algunos conceptos que se necesitan para poder entender algunos conceptos de la practica son los siguientes:

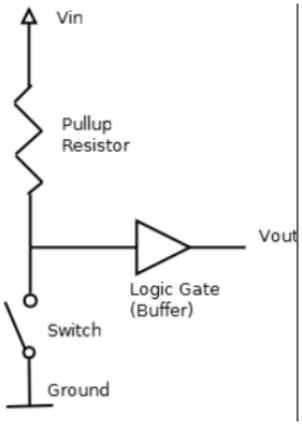
Configuración de entrada: Si algún pin de este puerto está configurado como entrada, entonces actúa como si “flotara”, es decir, la entrada tiene una resistencia de entrada ilimitada y un potencial indeterminado **[1]**.

Configuración de salida: Cuando el pin se configura como una salida, entonces actúa como un "drenaje abierto". Al aplicar un 0 lógico a un bit de puerto, el pin correspondiente se conectará a tierra (0 V) y, al aplicar el 1 lógico, la salida externa seguirá "flotando".

Para aplicar la lógica 1 (5V) en este pin de salida, es necesario construir una resistencia pullup externa **[1]**.



Las resistencias pull-up y pull-down se utilizan para polarizar correctamente las entradas de las puertas digitales y evitar que floten de forma aleatoria cuando no hay ninguna condición de entrada. Las puertas lógicas digitales pueden utilizarse para conectarlas a circuitos o dispositivos externos, pero hay que tener cuidado de que sus entradas o salidas funcionen correctamente y proporcionen la condición de conmutación esperada. Las puertas lógicas digitales modernas, los circuitos integrados y los microcontroladores contienen muchas entradas, llamadas «pines», así como una o más salidas, y estas entradas y salidas tienen que estar correctamente configuradas, ya sea en ALTO o en BAJO para que el circuito digital funcione correctamente **[2]**.



Las resistencias pull-up son resistencias de valor fijo utilizadas entre la conexión de una fuente de voltaje y un pin particular en un circuito lógico digital. Más comúnmente emparejado con interruptores, su propósito es asegurar que la tensión entre Ground y Vcc se controle activamente cuando el interruptor está abierto. Además, no afectando el estado del circuito al hacerlo también. Debes tener en cuenta que si no hay resistencias pull-up, resultará en un cortocircuito, que no es ideal **[2]**.

Esta práctica nos ayudara a entender cómo se pueden implementar los circuitos con un microcontrolador y todo el proceso que conlleva realizar la programación del microcontrolador.

Con esta introducción se podrá comprender y realizar el desarrollo de la práctica que se desarrolla más adelante.

IV. 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.

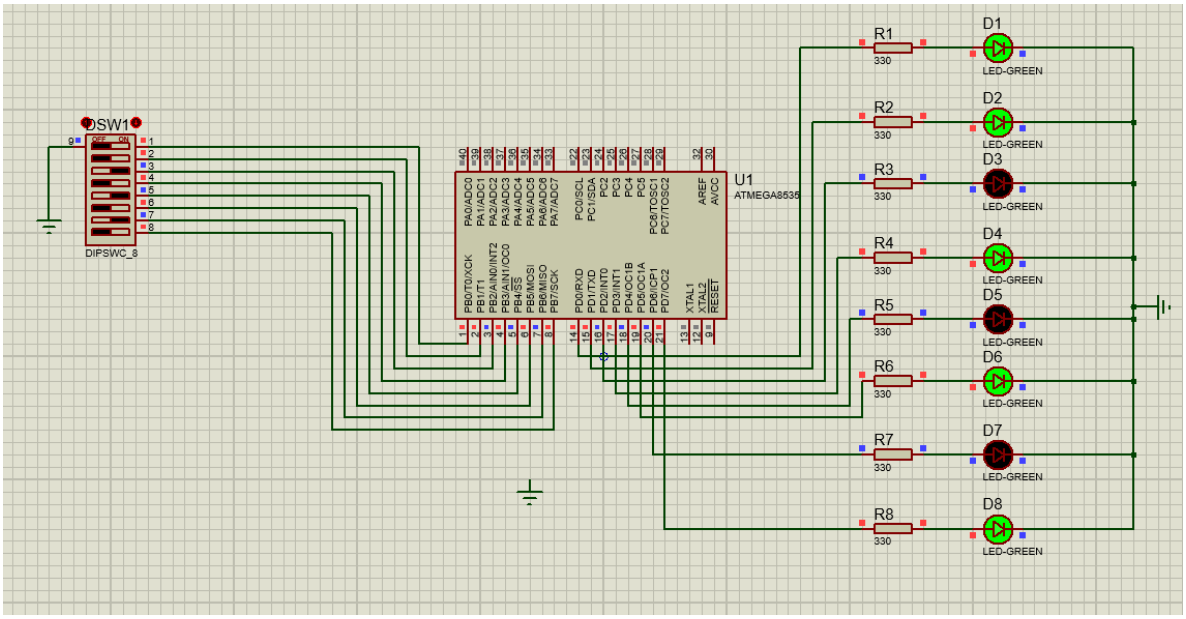


Fig.1_*Circuito Armado y Simulado en Proteus.*

CIRCUITO EN FISICO ARMADO.

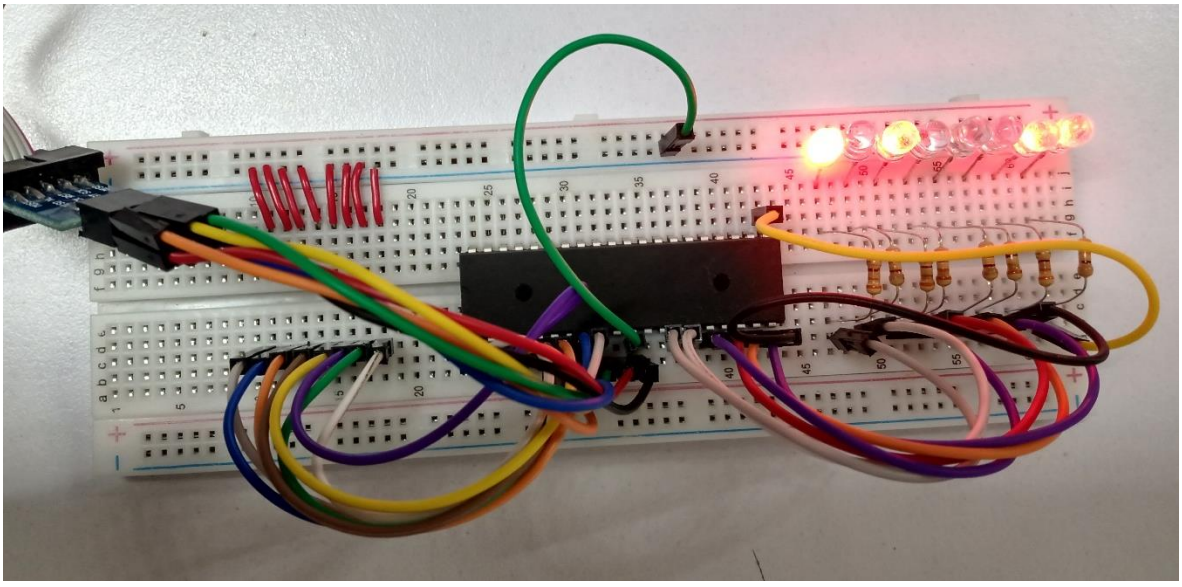


Fig.2_*Circuito Armado en Fisico.*

V. CÓDIGO

```
// I/O Registers definitions
#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;
      }
}
```

VI. OBSERVACIONES Y CONCLUSIONES

Gracias a la realización de la práctica podemos observar cómo se utiliza e implementa un circuito con un microcontrolador (ATMega8535) y como gracias a su programador pudimos logra programar el integrado con nuestro código en lenguaje C. Esta practica no fue tan difícil, pero, es muy importante para poder familiarizarnos con el entorno de programación y como este funciona. Durante la practica si se tuvieron muchos problemas, mas que nada porque no conocía la forma de trabajar con estos elementos físicos.

Tras la investigación y explicación de la práctica previa me quedo un poco más claro como poder implementar y programar el ATMega8535. También conocí el concepto de resistencias pull-up y como utilizarlas en nuestro entorno de programación.

Para concluir esta práctica podemos decir que fue una experiencia interesante ya que nunca había armado circuitos en físico, solo en simulaciones y es muy diferente.

VII. BIBLIOGRAFÍA

- ✓ CodeVisionAVR. (2018). CodeVisionAVR. 20/Septiembre/2021, de CodeVisionAVR Sitio web: http://www.hpinfotech.ro/cvavr-download.html

- ✓ Descubre Arduino. (2012). Resistencias pull-up y pull-down en Arduino. 20/Septiembre/2021,Sitio web: https://descubrearduino.com/resistencias-pull-up-y-pull-down-en-arduino/#:~:text=Las%20resistencias%20pull-up%20son%20resistencias%20de%20valor%20fijo,se%20controle%20activamente%20cuando%20el%20interruptor%20est%C3%A1%20abierto.

- ✓ Microprocesador. (2006). Microcontroladores 8051 Puertos de entrada y salida. 20/Septiembre/2021,Sitio web: https://isolution.pro/es/t/microprocessor/microcontrollers-8051-input-output-ports/microcontroladores-8051-puertos-de-entrada-y-salida#:~:text=Los%20microcontroladores%208051%20tienen%204%20puertos%20de%20E,el%20microcontrolador%20se%20conecte%20con%20los%20dispositivos%20perif%C3%A9ricos.