A picture containing icon

Description automatically generatedLogo

Description automatically generatedINSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

**PRÁCTICA 13: CONVERTIDOR ANALÓGICO DIGITAL**

ALUMNOS: MALAGON BAEZA ALAN ADRIAN  
 MARTINEZ CHAVEZ JORGE ALEXIS

GRUPO: 6CM3

U.A: SISTEMAS EN CHIP

PROFESOR: FERNANDO AGUILAR SÁNCHEZ

FECHA DE ENTREGA: 28 DE MAYO DE 2023

**OBJETIVO**

Al término de la sesión, los integrantes del equipo contaran con la habilidad de hacer uso del convertidor analógico digital del microcontrolador.

**INTRODUCCIÓN TEÓRICA**

**CONVERSOR DE SEÑAL ANALÓGICA A DIGITAL**

Un conversor o convertidor de señal analógica a digital (Conversor Analógico Digital, CAD; Analog-to-Digital Converter, ADC) es un dispositivo electrónico capaz de convertir una señal analógica, ya sea de tensión o corriente, en una señal digital mediante un cuantificador y codificándose en muchos casos en un código binario en particular. Donde un código es la representación unívoca de los elementos, en este caso, cada valor numérico binario hace corresponder a un solo valor de tensión o corriente.1

En la cuantificación de la señal se produce pérdida de la información que no puede ser recuperada en el proceso inverso, es decir, en la conversión de señal digital a analógica y esto es debido a que se truncan los valores entre 2 niveles de cuantificación, mientras mayor cantidad de bits mayor resolución y por lo tanto menor información perdida.1

Se utiliza en equipos electrónicos como computadoras, grabadores de sonido y de vídeo, y equipos de telecomunicaciones.1

Diagram

Description automatically generated

Figura 1. Procesos de la conversión AD

**CARACTERÍSTICAS DEL ATMEGA8535 PARA USO DEL ADC**

Podemos observar en la hoja de especificaciones del ATMEGA8535 que cuenta con un convertidor ADC de 10 bits.

Text

Description automatically generated



Figura 2. Características periféricas del ATMEGA8535

También, en la descripción de lo pines, podemos observar que el ATMEGA8535 cuenta con un pin para el voltaje de referencia (AREF) y otro para el voltaje de entrada (AVCC).

A picture containing text

Description automatically generated

Figura 3. Descripción de los pines AVCC y AREF del ATMEGA8535

Además, contamos con algunas características proporcionadas por la marca que debemos de considerar al momento de usar el ADC del microcontrolador.

Text

Description automatically generated

Figura 4. Características del ADC del ATMEGA8535

**MATERIALES Y EQUIPO EMPLEADO**

* CodeVision AVR
* AVR Studio 4
* Microcontrolador ATmega 8535
* 8 LEDS
* 8 Resistores de 330 Ω a 1⁄4 W
* 1 Potenciómetro de 10k

**DESARROLLO EXPERIMENTAL**

1. Realice una conversión de 8 bits sobre el canal 0 del ADC, con un Vref = 5V, muestre el resultado con leds en el Puerto B.

Diagram

Description automatically generated

Figura 5. Circuito para el convertidor analógico digital

Text, letter

Description automatically generated

Figura 6. Compilación exitosa en CodeVision

**CÓDIGO GENERADO POR CODEVISION**

|  |
| --- |
| /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  This program was created by the CodeWizardAVR V3.48b  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>  #include <delay.h>  #define ADC\_VREF\_TYPE ((0<<REFS1) | (1<<REFS0) | (1<<ADLAR)) //Voltaje de referencia en pin AVCC  // Lectura de los 8 bits más significativos del ADC  **unsigned** **char** **read\_adc**(**unsigned** **char** adc\_input){  ADMUX=adc\_input | ADC\_VREF\_TYPE;  delay\_us(**10**);  ADCSRA|=(**1**<<ADSC);  **while** ((ADCSRA & (**1**<<ADIF))==**0**);  ADCSRA|=(**1**<<ADIF);  **return** ADCH;  }  // 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=In Bit6=In Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In Bit0=In  DDRD=(**0**<<DDD7) | (**0**<<DDD6) | (**0**<<DDD5) | (**0**<<DDD4) | (**0**<<DDD3) | (**0**<<DDD2) | (**0**<<DDD1) | (**0**<<DDD0);  // State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=T Bit2=T Bit1=T Bit0=T  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);  // ADC initialization  // ADC Clock frequency: 500.000 kHz  // ADC Voltage Reference: AVCC pin  // ADC High Speed Mode: Off  // ADC Auto Trigger Source: ADC Stopped  ADMUX=ADC\_VREF\_TYPE;  ADCSRA=(**1**<<ADEN) | (**0**<<ADSC) | (**0**<<ADATE) | (**0**<<ADIF) | (**0**<<ADIE) | (**0**<<ADPS2) | (**0**<<ADPS1) | (**1**<<ADPS0);  SFIOR=(**1**<<ADHSM) | (**0**<<ADTS2) | (**0**<<ADTS1) | (**0**<<ADTS0);  // 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**)  {  PORTB = read\_adc(**0**);  delay\_ms(**10**);  }  } |

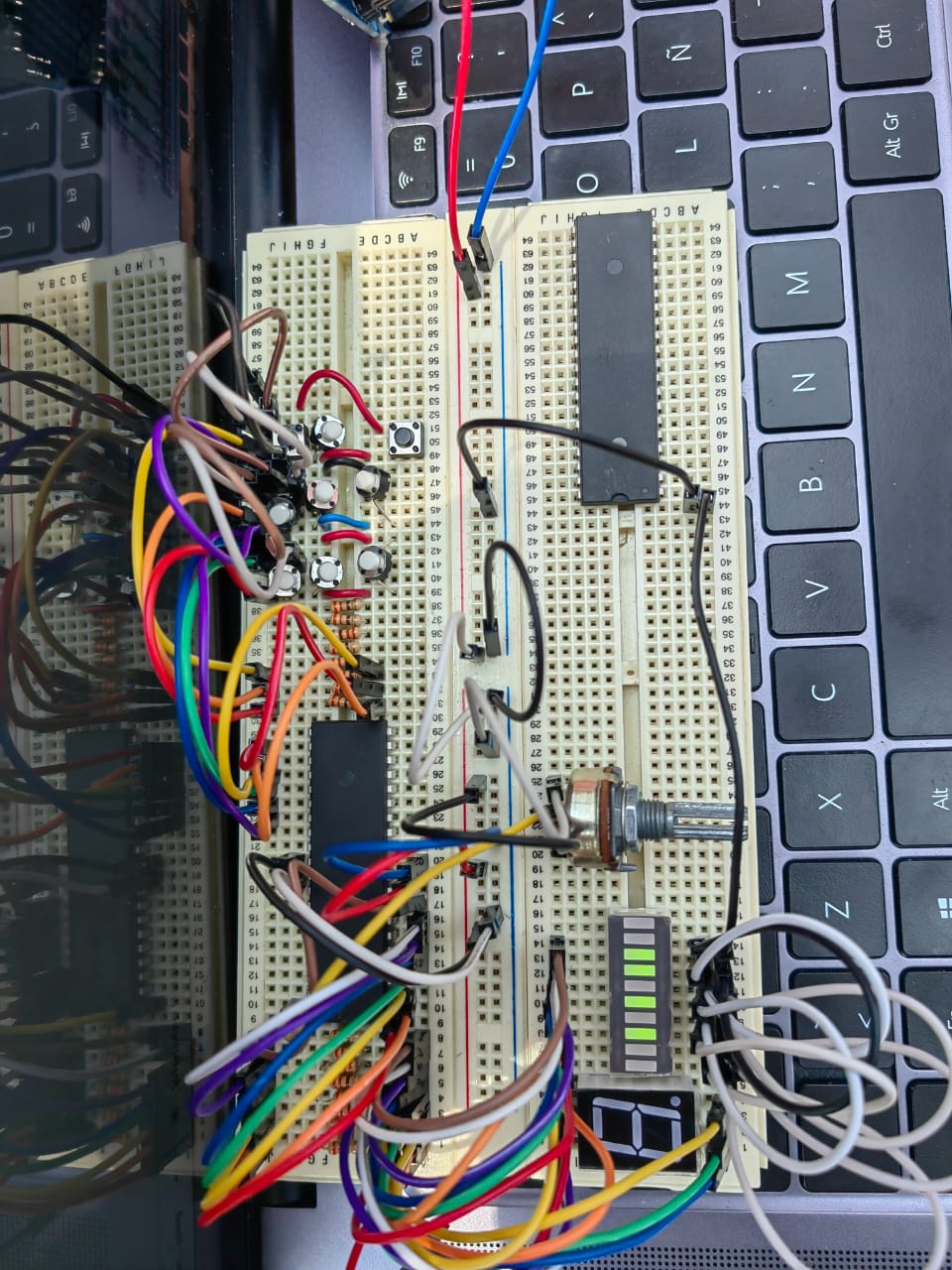
**CIRCUITO ELECTRICO EN PROTEUS**

Chart

Description automatically generated with low confidence

Figura 7. Circuito simulado en Proteus

**CIRCUITO EN EL PROTO ARMADO**

****

**OBSERVACIONES Y CONCLUSIONES INDIVIDUALES**

Malagon Baeza Alan Adrian

El convertidor analógico a digital es un dispositivo que nos sirve para obtener la representación digital (en 0’s y 1’s) de una señal analógica que entra en el convertidor. Estos dispositivos se pueden implementar en nuestros circuitos utilizando el ADC0804 o cualquier otro equivalente comercial. Sin embargo, si utilizamos un microcontrolador como el ATMEGA8535 podemos utilizar el ADC que se encuentra dentro del microcontrolador.

Martínez Chávez Jorge Alexis

Para poder implementar el ADC de nuestro microcontrolador es necesario leer las especificaciones que nos ofrece el fabricante en su hoja de datos, para conocer los bits de conversión, los pines para la entrada de los voltajes de entrada y de referencia, entre otros datos que pueden ser de utilidad para la construcción del circuito con el ADC.

Para poder leer el voltaje de entrada y realizar la conversión del voltaje es muy sencillo; el código de desarrollado para realizar la lectura y conversión consta de 6 líneas de código, que son generadas por CodeVision para poder ser utilizadas en nuestro programa realizando la llamada a la función “read\_adc”.

**BIBLIOGRAFÍA**

[1] Colaboradores de los proyectos Wikimedia, “Conversor de señal analógica a digital,” *Wikipedia.org*, May 03, 2005. <https://es.wikipedia.org/wiki/Conversor_de_se%C3%B1al_anal%C3%B3gica_a_digital>

(accessed May 06, 2022).

[2] alldatasheet.com, “ATMEGA8535 Datasheet(PDF) - ATMEL Corporation,” *www.alldatasheet.com*. <https://www.alldatasheet.com/datasheet-pdf/pdf/164169/ATMEL/ATMEGA8535.html> (accessed May 08, 2022).