Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica

CONTROLADORES Y MICROCONTROLADORES PROGRAMABLES

Actividad fundamental 4
ADC y Timer

Nombre o nombres de los integrantes junto con su matrícula:
Nahaliel Gamaliel Ríos Martínez 1884244

Ing. Jesus Daniel Garza Camarena

Semestre Febrero 2021 – Junio 2021

MN1N2

San Nicolás de los Garza, N.L. 18.05.2021

Objetivo

Comprender los periféricos internos del microcontrolador como el ADC y los timer

Introducción.

Un conversor de señal o ADC es un sistema que puede transformar señales de tipo analógico en otras de tipo digital.

Un sistema de procesamiento digital de la señal traduce primero una señal analógica que varía de manera continua a una serie de niveles discretos. Esta serie de niveles sigue las variaciones de la señal analógica y se asemeja a una escalera.

Según la función de un conversor de señal pueden establecerse numerosas utilidades. Estos conversores o ADC pueden encontrarse en dispositivos como los smartphones, termómetros, micrófonos y, en general todos los lugares donde una señal analógica de tipo físico deba ser convertida en otra digital. Esto ha sido especialmente relevante y ha tenido un papel esencial en la llamada Revolución 4.0 donde la industria ha sido digitalizada en gran parte gracias a la precisión y ventajas que otorgan. Son de hecho los responsables de que pueda llevarse un control pormenorizado y centralizado de todas las variables físicas que intervienen en los diferentes procesos de producción y sistemas al convertirlos en señales binarias de tipo digital.

Existen fundamentalmente tres tipos de conversores:

- Conversores de aproximaciones sucesivas: Se emplean para realizar mediciones de alta velocidad. En concreto, con el empleo de comparadores establece un rango para precisar el rango de voltaje de entrada. Es decir, compara sucesivamente el voltaje de entrada con el de salida para llevar a cabo una aproximación sucesiva. Con ello, finalmente consigue la resolución óptima deseada.
- Conversores de rampa: La función de un convertidor de señal de rampa está enfocada sobre todo en conseguir una buena linealidad en detrimento de la velocidad. El nombre de rampa viene por producir ondas de sierra que suben y bajan para alcanzar el valor cero. Un temporizador comienza a contar y cuando el voltaje de dicha rampa alcanza el voltaje de entrada un comparador graba el valor registrado por el temporizador.

Los registros utilizados en el manejo de las entradas analógicas en el ATMEGA328p son:

- ADMUX: ADC Multiplexer Selection Register. Selector del canal del multiplexor del ADC y el voltaje de referencia.
- ADCSRA: ADC Control and Status Register A. Control del ADC y su estado.
- ADCSRB: ADC Control and Status Register B.
- ADCL: ADC Data Register Low. Cuando la conversión ADC ha finalizado, el resultado se deja en estos dos registros.

- ADCH: Data Register High
- DIDR0: Digital Input Disable Register 0. Para deshabilitar la entrada digital de los pines analógicos. Página 326.

Diagrama de bloques

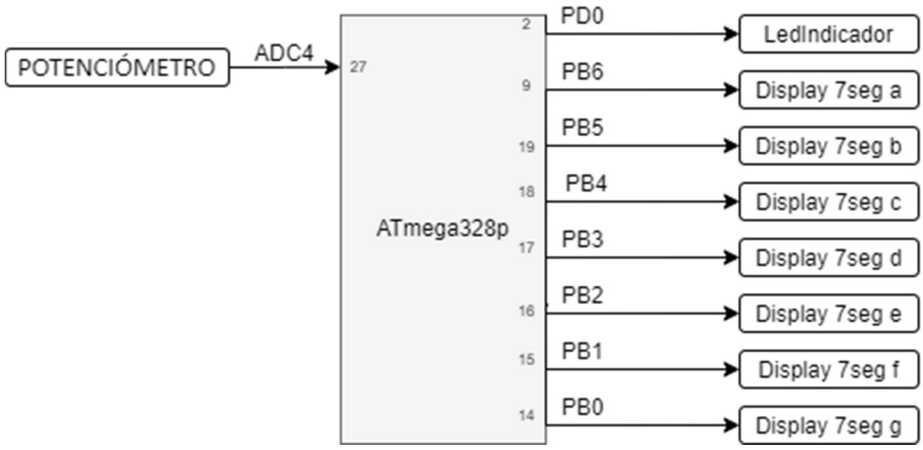
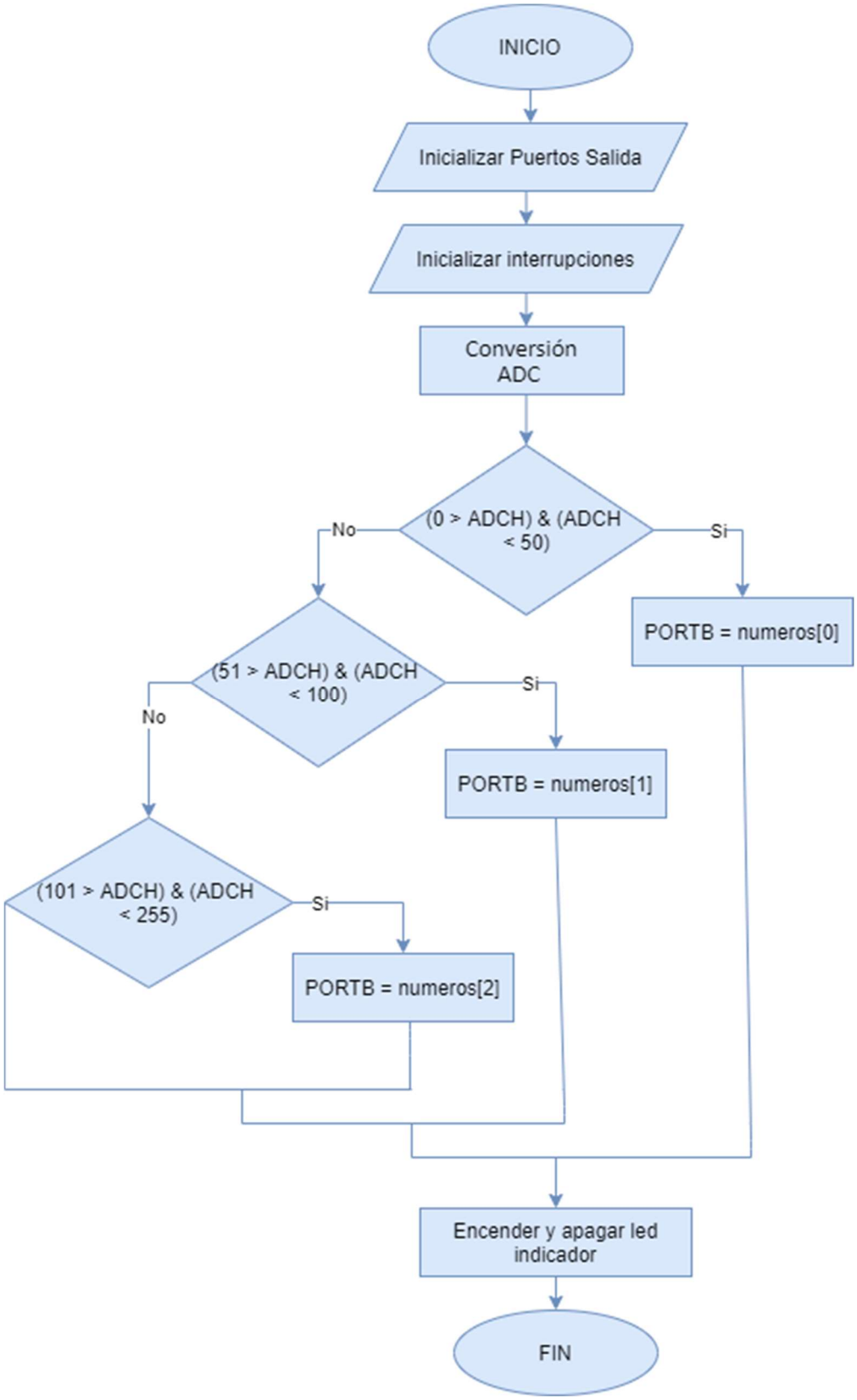


Diagrama de flujo.



Materiales utilizados

1 ATMEGA328P
1 Push Button
1 Led Azul
1 Potenciómetro
1 resistencaia (10K)
1 resistencia (220)
3 capacitores
1 Display 7 seg

Código en Atmel.

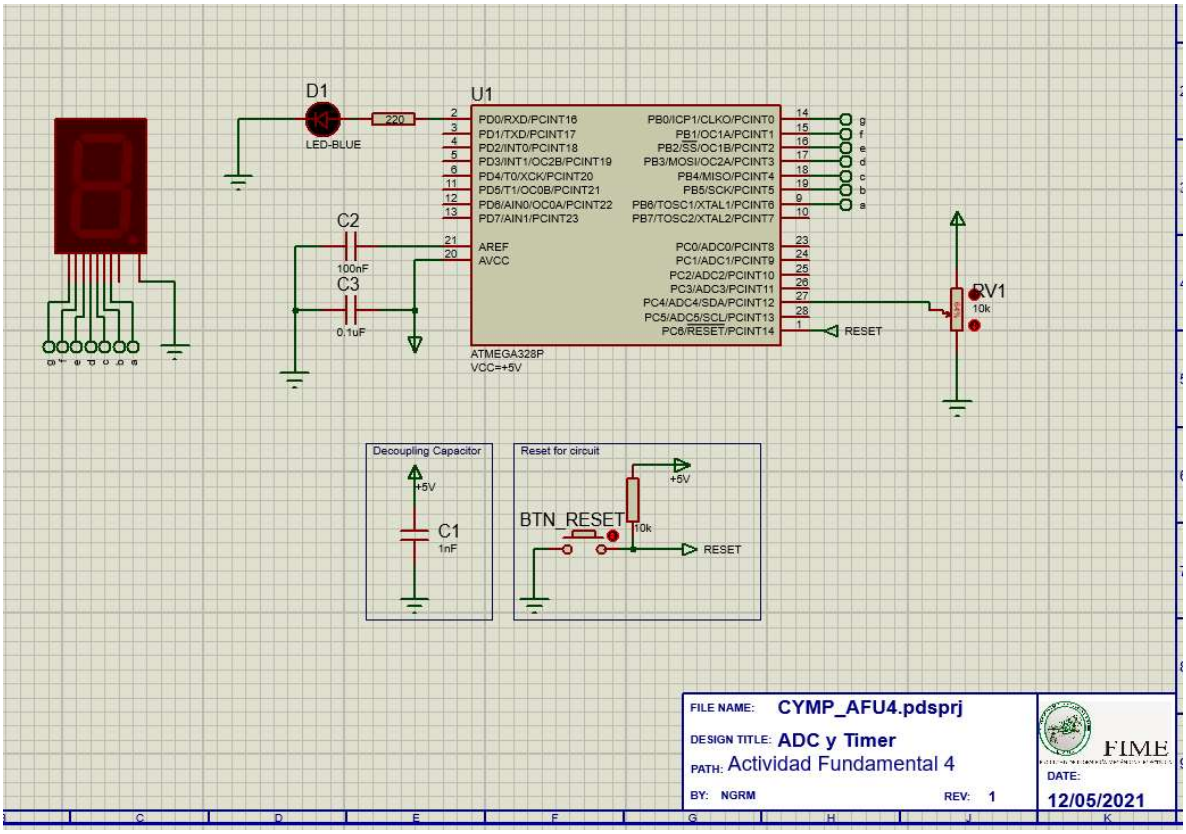
```
/*************************************************
 * LLENAR ESTE ESPACIO CON LOS SIGUIENTES DATOS:    *
 * Nombre: Nahaliel Gamaliel Rios Martinez          *
 * Hora clase: N4                                   *
 * Día: LMV                                         *
 * Nº de lista: 33                                  *
 * Dispositivo: ATMEGA328P                          *
 * Rev: 1.0                                         *
 * Propósito de la actividad:                       *
 * Implementar un detector de rango el cual muestre *
 * las siguientes letras en un display dependiendo  *
 * de los niveles existentes en la entrada del      *
 * microcontrolador utilizando un potenciómetro     *
 * (resistencia variable) en la entrada siendo      *
 * leído por el ADC para convertir el nivel de      *
 * voltaje mostrado en un número binario.           *
 *                                                  *
 * - 0 a 50 Letra "L" de Low                        *
 * - 51 a 100 ? Letra "K" de OK                     *
 * - 101 a 255 ? Letra "H" de High                  *
 *                                                  *
 * Añade un led que encienda y apague a 0.1s        *
 * para que se visualice que el programa esa        *
 * corriendo mediante un timer                      *
 *                                 Fecha: 18.05.2021 *
 *************************************************/
/*atmega328P PIN - OUT*/
/*       PIN - OUT
         atmega328P
          -------
    PC6  |1     28| PC5
    PD0  |2     27| PC4
    PD1  |3     26| PC3
    PD2  |4     25| PC2
    PD3  |5     24| PC1
    PD4  |6     23| PC0
    VCC  |7     22| GND
    GND  |8     21| AREF
    PB6  |9     20| AVCC
    PB7  |10    19| PB5
    PD5  |11    18| PB4
    PD6  |12    17| PB3
```

```
    PD7  |13    16| PB2
    PB0  |14    15| PB1
          --------
*/
/*atmega328P PIN FUNCTIONS*/
 /*
 atmega328P PIN FUNCTIONS
 pin   function              name    pin     function              name
 1     !RESET/PCINT14        PC6     15      PCINT1/OC1A           PB1
 2     RxD/PCINT16           PD0     16      PCINT2/OC1B/SS        PB2
 3     TxD/PCINT17           PD1     17      PCINT3/OC2A/MOSI      PB3
 4     INT0/PCINT18          PD2     18      PCINT4/MISO           PB4
 5     INT1/PCINT19/OC2B     PD3     19      PCINT5/SCK            PB5
 6     PCINT20               PD4     20      ANALOG VCC            AVCC
 7     +5v                   VCC     21      ANALOG REFERENCE      AREF
 8     GND                   GND     22      GND                   GND
 9     XTAL1/PCINT6          PB6     23      PCINT8/ADC0           PC0
 10    XTAL2/PCINT7          PB7     24      PCINT9/ADC1           PC1
 11    PCINT21/OC0B          PD5     25      PCINT10/ADC2          PC2
 12    PCINT22/OC0A/AIN0     PD6     26      PCINT11/ADC3          PC3
 13    PCINT23/AIN1          PD7     27      PCINT12/ADC4/SDA      PC4
 14    PCINT0/AIN1           PB0     28      PCINT13/ADC5/SCL      PC5
 */
/*******************Bibliotecas*****************************/
#include <avr/io.h>//se incluyen las Bibliotecas de E/S del AVR atmega328P
#include <avr/interrupt.h>// librería de interrupciones
#include <avr/delay.h>

/*****************Macros y constantes**********************/
#define F_CPU 1000000UL //1 Mhz

/*******************Variables globales*********************/
//--Espacio para declarar variables globales
#define a PINB0
#define b PINB1
#define c PINB2
#define d PINB3
#define e PINB4
#define f PINB5
#define g PINB6

#define LedIndicador PIND0

uint8_t numeros[3] = {
       //gfedcba
       0b0111000, //L
       0b1110110, //K
       0b1110100, //h
};


/***********************Funciones*************************/
//--Espacio para Establecer funciones
/*********************Declaración de Funciones************/
//--Espacio para declarar funciones
void initialize_ports(void); // Inicializar puertos
void ADC_init(void);
void ADC_on(void);
```

```c
/******************Programa principal*******************/
int main(void)
{
//--Inicialización
    cli();
    initialize_ports(); // va hacía la inicialización
    ADC_init();
    sei();
    ADC_on();

//--Ejecución
    while (1) //loop infinito
    {

    } // END loop infinito

} // END MAIN
/******************Definición de funciones*****************/
//*********************************************************
//Descripcion de lo que hace la funcion:                   *
//initialize_ports : inicializa los puertos de entrada o   *
//salida                                                   *
//*********************************************************
void initialize_ports(void)
{
       //--Entradas

    //--Salidas
       DDRB |=_BV(a);
       DDRB |=_BV(b);
       DDRB |=_BV(c);
       DDRB |=_BV(d);
       DDRB |=_BV(e);
       DDRB |=_BV(f);
       DDRB |=_BV(g);

       DDRD |=_BV(LedIndicador);

       PORTD = 0x00; //-Por seguridad iniciamos en 0
       PORTB = 0x00; //-Por seguridad iniciamos en 0

}
//*********************************************************
//Descripcion de lo que hace la funcion:                   *
//ADC_init : Habilitamos la interrupción y configuramos    *
// el ADC                                                  *
//*********************************************************
void ADC_init(void)
{
       //Avcc como pin de referencia
       ADMUX &=~ (1<<REFS1);
       ADMUX |=  (1<<REFS0);

       //8 bits
       ADMUX |= (1<<ADLAR);

    //PIN ADC4
```

```
    ADMUX &=~ (1<<MUX3);
    ADMUX |=  (1<<MUX2);
    ADMUX &=~ (1<<MUX1);
    ADMUX &=~ (1<<MUX0);

       //Freeruning
       ADCSRA |= (1<<ADATE);

       //Habilitar interrupción
       ADCSRA |= (1<<ADIE);

       //velocidad de muestreo
       // 1 MHz clock / 8 = 125 kHz ADC clock debe de estar entre 50 - 200Khz
       ADCSRA &=~ (1<<ADPS0);
       ADCSRA |=  (1<<ADPS1);
       ADCSRA |=  (1<<ADPS2);
}
//*************************************************************
//Descripcion de lo que hace la funcion:                      *
//ADC_init : Leer y convertir señal análoga                   *
//*************************************************************
void ADC_on(void)
{
       //Encendemos el ADC
       ADCSRA |= (1<<ADEN);
       _delay_ms(10);
       // Iniciar la conversión
       ADCSRA |= (1 << ADSC);
}
ISR(ADC_vect)
{
       //0 a 5V -> 0 a 255bits

       //0 a 50 Letra "L" de Low
       if ( (ADCH >= 0) && (ADCH <= 50) )
       {
              PORTB = numeros[0];

       //51 a 100 ? Letra "K" de OK
       }else if((ADCH >= 51) && (ADCH <= 100)){
              PORTB = numeros[1];

       //101 a 255 ? Letra "h" de High
       }else if((ADCH >= 101) && (ADCH <= 255)){
              PORTB = numeros[2];
       }

       PORTD ^= (1<<LedIndicador); //Encender y apagar led indicador

}
```

Diagrama del circuito en PROTEUS.



Conclusión

En esta práctica vimos mas sobre el funcionamiento y utilidad de los convertidores ADC y de como estos han sido de gran relevancia en lo que llamamos la industria 4.0, pues gracias a este mecanismo somos capases de recopilar y explotar una gran cantidad de información.
Para realizar esta práctica utilizamos un potenciómetro de resistencia variable y a través de las interrupciones del microcontrolador leemos el valor que nos esta regresando para posteriormente mostrar un indicador en un display de 7 segmentos. También agregamos un led indicador el la rutina ISR para mostrar que el código está en ejecución

Bibliografía

Floyd, T. L. (2006). Fundamentos de sistemas digitales. Pearson Educación.

Rubio, A. (2019, 5 abril). ¿Cuál es la función de un convertidor de señal? - Instrumentación Digital. Paneles digitales y analizadores de red. https://www.instrumentaciondigital.es/cual-es-la-funcion-de-un-convertidor-de-senal/

JECRESPOM (2017b, septiembre 5). ADC –. Aprendiendo Arduino. https://aprendiendoarduino.wordpress.com/tag/adc/#:%7E:text=El%20ADC%20puede%20trabajar%20en,luego%20comienza%20con%20la%20siguiente.

ATmega328P. 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash. DATASHEET. https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf