 **Universidad Autónoma de Nuevo León**

**Facultad de Ingeniería Mecánica y Eléctrica**

LAB. CONTROLADORES Y MICROCONTROLADORES PROGRAMABLES

Práctica P8

Máquina de estados

Nombre o nombres de los integrantes junto con su matrícula:

Verónica Yazmín Gómez Cruz 1884224

Nahaliel Gamaliel Ríos Martínez 1884244

Ing. Jesús Daniel Garza Camarena

Semestre Febrero 2021 – Junio 2021

MN1N2

San Nicolás de los Garza, N.L. 19.05.2021

# Objetivo

Aplicar el funcionamiento de una máquina de estados finitos en un MCU

# Introducción.

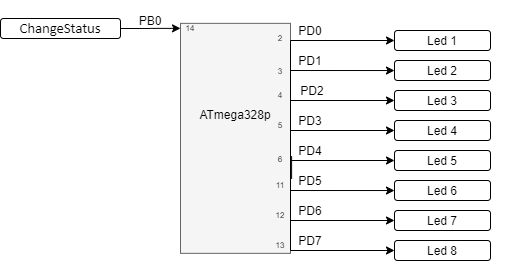
Las máquinas de estado finito, más conocidas por su acrónimo en inglés FSM (Finite State Machine), se utilizan ampliamente en el diseño de circuitos digitales (además de en otros ámbitos de la ingeniería, como la programación), para describir el comportamiento de un sistema según el valor de sus entradas y de cómo van cambiando en el tiempo.

Desde el punto de vista de las FSM, un sistema está compuesto de estados por los que va pasando el sistema, de señales de entrada que modifican esos estados y de señales de salida que pueden utilizarse para conocer el estado del sistema y actuar en consecuencia. Un ejemplo muy visual podría ser un semáforo, el cuál dispone de tres estados diferentes, uno para cada color.

Máquina de estado Moore.- Edward Moore escribió un ensayo en 1956 (Gedanken-experiments on Sequential Machines) y por lo tanto el estilo de la máquina lleva su nombre. El dice que la salida depende solo del estado, y el próximo estado es dependiente del estado actual (o salida), y la entrada. Puedes notar que no importa cuál será el estado de la entrada, la salida solo depende el estado actual contenido dentro del elemento de la memoria.

Máquina de estado Mealy.- George Mealy escribió un ensayo un año antes que Moore, titulado “A Method for Synthesizing Sequential Circuits”, en el cual entra en profundidad acerca de crear máquinas de estado desde funciones matemáticas, y describe esas salidas de máquinas de estado en términos de sus entradas.

**Diagrama de bloques**

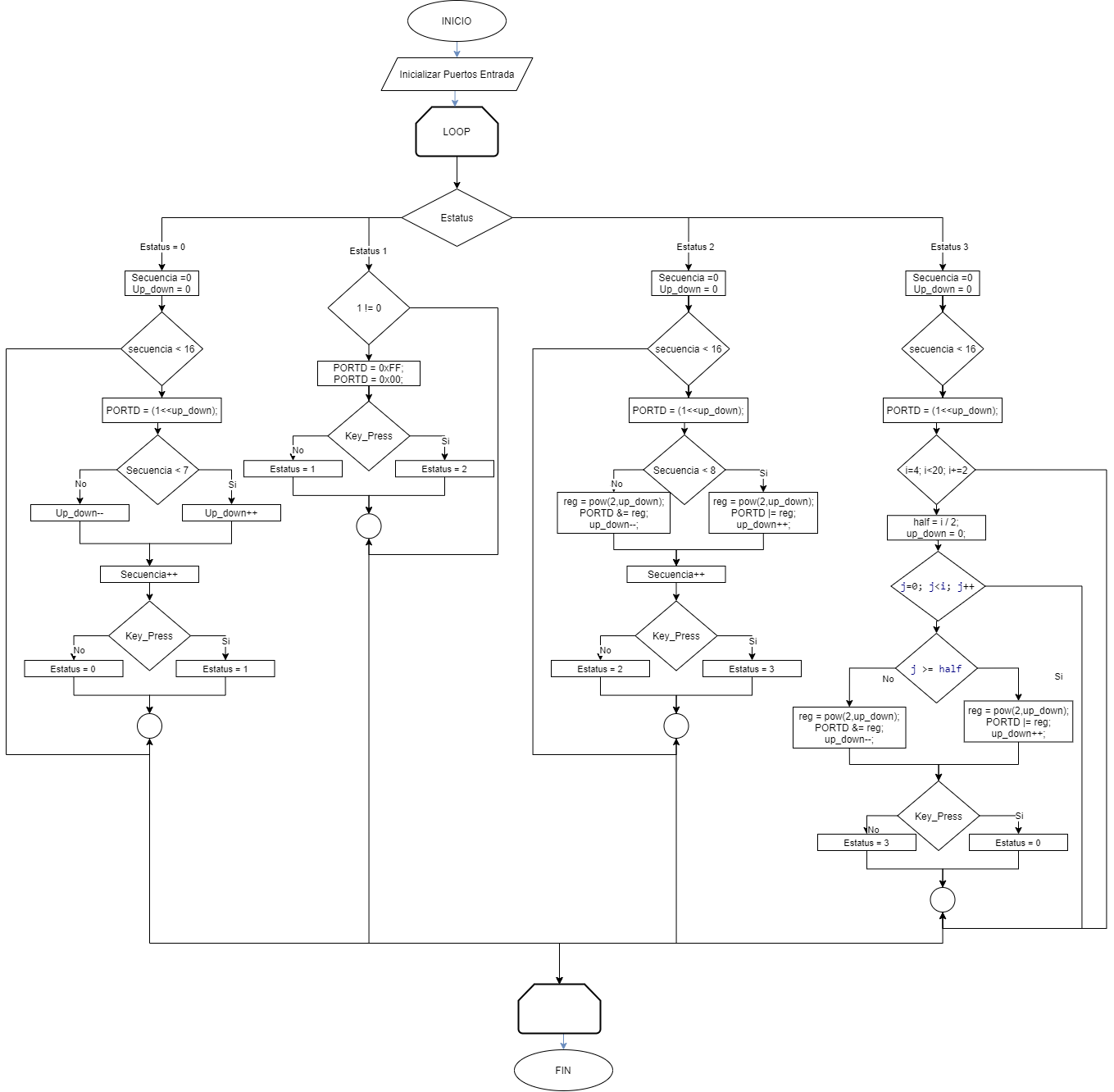
****

**Diagrama de transición**

**Diagrama

Descripción generada automáticamente**

# Diagrama de flujo.



# Materiales utilizados

1 ATMEGA328P

2 Push Button

8 resistencaias (220)

1 resistencaias (10K)

1 capacitores 1nF

8 Leds azules

# Código en Atmel.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* LLENAR ESTE ESPACIO CON LOS SIGUIENTES DATOS: \*

\* Nombre: Verónica Yazmín Gómez Cruz \*

\* Nahaliel Gamaliel Rios Martinez \*

\* Hora clase: N1-N2 \*

\* Día: M \*

\* N° de lista: 17, 18 \*

\* N° de Equipo: 7 \*

\* Dispositivo: ATMEGA328P \*

\* Rev: 1.0 \*

\* Propósito de la actividad: \*

\* Diseñe, efectué la simulación y construya un \*

\* prototipo de un sistema que controle 8 LEDs los \*

\* cuales son controlados por una máquina de estados\*

\* cada vez que se presiona un botón estos cambiaran\*

\* de modalidad. \*

\* \*

\* a) Modalidad 1: Efecto auto increíble los leds \*

\* se desplazan cada 0.5 segundos. \*

\* \*

\* b) Modalidad 2: Leds encienden y apagan cada 1 \*

\* segundo. \*

\* \*

\* c) Modalidad 3: Los leds se van encendiendo uno \*

\* a uno hasta terminar y después se apagan uno \*

\* en uno cada 1 segundo. \*

\* \*

\* d) Modalidad 4: Efecto libre a decisión del \*

\* diseñador. \*

\* \*

\* Fecha: 19.05.2021 \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*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 <math.h>

#include <avr/delay.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Macros y constantes\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define *F\_CPU* 1000000UL //1 Mhz

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Variables globales\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//--Espacio para declarar variables globales

//Salidas

#define led1 PIND0

#define led2 PIND1

#define led3 PIND2

#define led4 PIND3

#define led5 PIND4

#define led6 PIND5

#define led7 PIND6

#define led8 PIND7

//Entradas

#define ChangeStatus PINB0

#define Key\_Press bit\_is\_set (PINB,ChangeStatus)

enum states

{

state\_0, //0 - auto increíble

state\_1, //1 - encender y apagar cada 1 seg

state\_2, //2 - encender y apagar 1 a 1 cada 1 seg

state\_3 //3 - Efecto libre

} state;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Funciones\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//--Espacio para Establecer funciones

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Declaración de Funciones\*\*\*\*\*\*\*\*\*\*\*\*\*/

//--Espacio para declarar funciones

void initialize\_ports(void); // Inicializar puertos

void auto\_increible(void);

void ON\_OFF(void);

void ON\_OFF\_ONE\_BY\_ONE(void);

void FreeEfect(void);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Programa principal\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int main(void)

{

//--Inicialización

initialize\_ports();

state = state\_0; //Estado inicial

//--Ejecución

while (1) //loop infinito

{

switch (state)

{

//a) Modalidad 1: Efecto auto increíble los leds se desplazan cada 0.5 segundos.

case state\_0:

auto\_increible();

break;

//b) Modalidad 2: Leds encienden y apagan cada 1 segundo.

case state\_1:

ON\_OFF();

break;

//c) Modalidad 3: Los leds se van encendiendo uno a uno hasta terminar y después se apagan uno en uno cada 1 segundo.

case state\_2:

ON\_OFF\_ONE\_BY\_ONE();

break;

//d) Modalidad 4: Efecto libre a decisión del diseñador.

case state\_3:

FreeEfect();

break;

}

} // 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

DDRB &=~\_BV(ChangeStatus); //Boton

PORTB |= \_BV(ChangeStatus); //pull- Up activado

//--Salidas

DDRD |=\_BV(led1);

DDRD |=\_BV(led2);

DDRD |=\_BV(led3);

DDRD |=\_BV(led4);

DDRD |=\_BV(led5);

DDRD |=\_BV(led6);

DDRD |=\_BV(led7);

DDRD |=\_BV(led8);

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

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Descripcion de lo que hace la funcion: \*

//auto\_increible: Esta función enciende un led, lo apaga, \*

// enciende el siguiente y así sucesivamente hasta \*

// completar el recorrido \*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void auto\_increible(void)

{

*uint8\_t* up\_down;

*uint8\_t* secuencia;

//Reiniciamos variables

secuencia = 0;

up\_down = 0;

while (secuencia < 16){

if (Key\_Press)

{

state = state\_0;

}else{

state = state\_1;

*\_delay\_ms*(500);

break;

}

PORTD = (1<<up\_down);

*\_delay\_ms*(500);

if (secuencia < 7) {

up\_down++;

}else{

up\_down--;

}

secuencia++;

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Descripcion de lo que hace la funcion: \*

//ON\_OFF: Esta función enciende y apaga todos los leds \*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void ON\_OFF(void)

{

while(1 != 0){

if (Key\_Press)

{

state = state\_1;

}else{

state = state\_2;

*\_delay\_ms*(500);

break;

}

PORTD = 0xFF;

*\_delay\_ms*(500);

PORTD = 0x00;

*\_delay\_ms*(500);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Descripcion de lo que hace la funcion: \*

// ON\_OFF\_ONE\_BY\_ONE: Esta función enciende un led \*

// sucesivamente hasta tenerlos todos encendidos, después \*

// los apaga uno a uno \*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void ON\_OFF\_ONE\_BY\_ONE(void)

{

*uint8\_t* up\_down;

*uint8\_t* secuencia;

*uint8\_t* reg;

//Reiniciamos variables

secuencia = 0;

up\_down = 0;

while (secuencia < 16){

if (Key\_Press)

{

state = state\_2;

}else{

state = state\_3;

*\_delay\_ms*(500);

break;

}

reg = *pow*(2,up\_down);

if (secuencia <= 8) {

PORTD |= reg;

*\_delay\_ms*(1000);

if (up\_down == 8){

up\_down--;

}else{

up\_down++;

}

}else{

PORTD &= reg;

*\_delay\_ms*(1000);

up\_down--;

if (secuencia == 14)

{

up\_down = 0;

}

}

secuencia++;

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Descripcion de lo que hace la funcion: \*

// FreeEfect: Esta función enciende un led, lo apaga, \*

// enciende los siguientes 2, los apaga y así sucesivamente\*

// hasta completar el recorrido, para dar el efecto de una \*

// barra de sonido \*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void FreeEfect(void)

{

int up\_down;

*uint8\_t* reg;

*uint8\_t* half;

*uint8\_t* i;

*uint8\_t* j;

//Reiniciar Variable

up\_down = 0;

for (i=4; i<20; i+=2)

{

half = i / 2;

up\_down = 0;

for (j=0; j<i; j++)

{

if (Key\_Press)

{

state = state\_3;

}else{

state = state\_0;

*\_delay\_ms*(500);

break;

}

if (j >= half)

{

reg = *pow*(2,up\_down);

if (reg == 2)

{

reg = 1;

}

PORTD &= reg;

//\_delay\_ms(200);

up\_down--;

}else{

reg = *pow*(2,up\_down);

PORTD |= reg;

//\_delay\_ms(200);

if (j == (half-1)){

up\_down--;

}else{

up\_down++;

}

}

}

}

}

# Diagrama del circuito en PROTEUS.

Diagrama

Descripción generada automáticamente con confianza media

**Conclusión**

En esta práctica vimos como crear una maquina de estados finitos e implementarla en el microcontrolador, en este caso la maquina se trató de tener funcionalidades diferentes en 8 leds de salida, esto permite tener mayor control en los eventos que queremos disparar en nuestro circuito, pues podremos saber en todo momento la razón del estado en que se encuentra el microcontrolador y como llegó a dicho estado, creo que este tipo de implementaciones pueden ser muy útiles para más cosas por ejemplo controlar periféricos dependiendo de las entradas de un sensor y así poder automatizar procesos sin necesidad de nosotros controlar los estados del circuito.

**Bibliografía**

P. (2019, 18 junio). Máquinas de estado. MCI Capacitación. https://cursos.mcielectronics.cl/2019/06/18/maquinas-de-estado/

A., & A. (2017b, marzo 25). Máquinas de estado finito en VHDL. Digilogic. https://digilogicelectronica.wordpress.com/2017/03/25/maquinas-de-estado-finito-fsm-vhdl/#:%7E:text=Las%20m%C3%A1quinas%20de%20estado%20finito,entradas%20y%20de%20c%C3%B3mo%20van

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