



INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO



Introducción a los microcontroladores 3CM16

Proyecto final

Juego de Ping-Pong

Equipo 7

Integrantes:

- ♥ Bocanegra Heziquio Yestlanezi
- ♥ Dominguez Durán Alan Axel
- ♥ Hernandez Mendez Oliver Manuel
- ♥ Martinez Cruz José Antonio

Profesor: Aguilar Sanchez Fernando



índice

Índice de imágenes.....	2
Objetivo.....	3
Introducción.....	3
Desarrollo	4
Conexión de la Matriz de LEDs.....	4
Conexión del Display de 7 segmentos.....	4
Conexión de los Push Buttons	4
Desarrollo del programa	4
Simulación – proteus.....	5
Circuito	5
Pruebas y ajustes.....	5
Codificación	6
Conclusiones	15
Bocanegra Heziquio Yestlanezi.....	15
Dominguez Durán Alan Axel.....	15
Hernandez Mendez Oliver Manuel	16
Martinez Cruz José Antonio	16
Referencias.....	17

índice de imágenes

Imagen 1 simulación – proteus.....	5
Imagen 2 circuito funcionando correctamente.....	6





Objetivo

Al término de este semestre los alumnos tendrán la capacidad para diseñar y elaborar un proyecto final.

Introduccion

Placa Arduino: Utilizaremos una placa Arduino como el controlador principal del juego y para controlar la matriz de LEDs y los botones. Puedes utilizar cualquier modelo de Arduino compatible [1].


Matriz de LEDs: Selecciona una matriz de LEDs de 7x5 o de 8x8, que sea compatible con la placa Arduino que estés utilizando. Asegúrate de tener una matriz de LEDs que admita control individual de cada LED [1].

Raqueta: La raqueta estará formada por dos puntos en la base de la matriz de LEDs. Estos puntos representarán la posición de la raqueta y se moverán de izquierda a derecha según los botones que se presionen [1].

Display de 7 segmentos: Utiliza un display de 7 segmentos para mostrar la puntuación del jugador. Cada vez que la pelota pase sin ser golpeada por la raqueta, se incrementará la puntuación y se mostrará en el display [1].

Push Buttons: Necesitarás dos botones para mover la raqueta hacia la derecha o hacia la izquierda. Conecta los botones a pines digitales de la placa Arduino [1].

Diseño y Diagramas eléctricos: Antes de comenzar a programar, es recomendable hacer un diseño y diagramas eléctricos del circuito. Esto te ayudará a tener una visión clara de cómo se conectan los componentes entre sí [1].





Desarrollo

Desarrollo Experimental

1.- Diseñe un Juego de Ping-Pong con las siguientes características armando su circuito final en "PLACA":

- ⇒ Use una Matriz de leds de 7x5 o de 8x8.
- ⇒ En la matriz de leds la pelota rebotará en las orillas y la raqueta estará formada por 2 puntos en la base de la matriz.
- ⇒ Se marcará un punto en el display de 7 segmentos por cada pelota que el jugador no alcance con la raqueta.
- ⇒ Los push button sirven para mover la raqueta de derecha a izquierda y viceversa.
- ⇒ Para la entrega del Proyecto se debe anexar un informe en el que debe incluir su diseño, diagramas eléctricos y código del programa aplicado.

Una vez contando con todos componentes, seguimos los siguientes pasos para el desarrollo del juego:

Conexion de la Matriz de LEDs

Conecta la matriz de LEDs a la placa Arduino siguiendo el esquema de conexiones correspondiente. Asegúrate de tener identificados los pines de datos (data), reloj (clock) y latch (latch) de la matriz de LEDs.

Conexion del Display de 7 segmentos


Conecta el display de 7 segmentos a la placa Arduino, utilizando pines digitales para controlar cada segmento. Consulta el datasheet del display para conocer la asignación de pines.

Conexion de los Push Buttons

Conecta los botones a pines digitales de la placa Arduino. Puedes utilizar resistencias pull-up o pull-down para asegurar que los pines tengan un valor lógico definido cuando los botones no están siendo presionados.

Desarrollo del programa

Utilizando el entorno de desarrollo de Arduino, programa la lógica del juego. Deberás controlar el movimiento de la pelota, detectar colisiones con las paredes y la raqueta, actualizar la puntuación en el display de 7 segmentos y mover la raqueta según los botones presionados.



Simulación – proteus

Con la información anterior y la investigación realizada para el proyecto final, realizamos la simulación mediante proteus, a continuación, en la imagen 1 podemos observar que la simulación funciona de manera correcta.

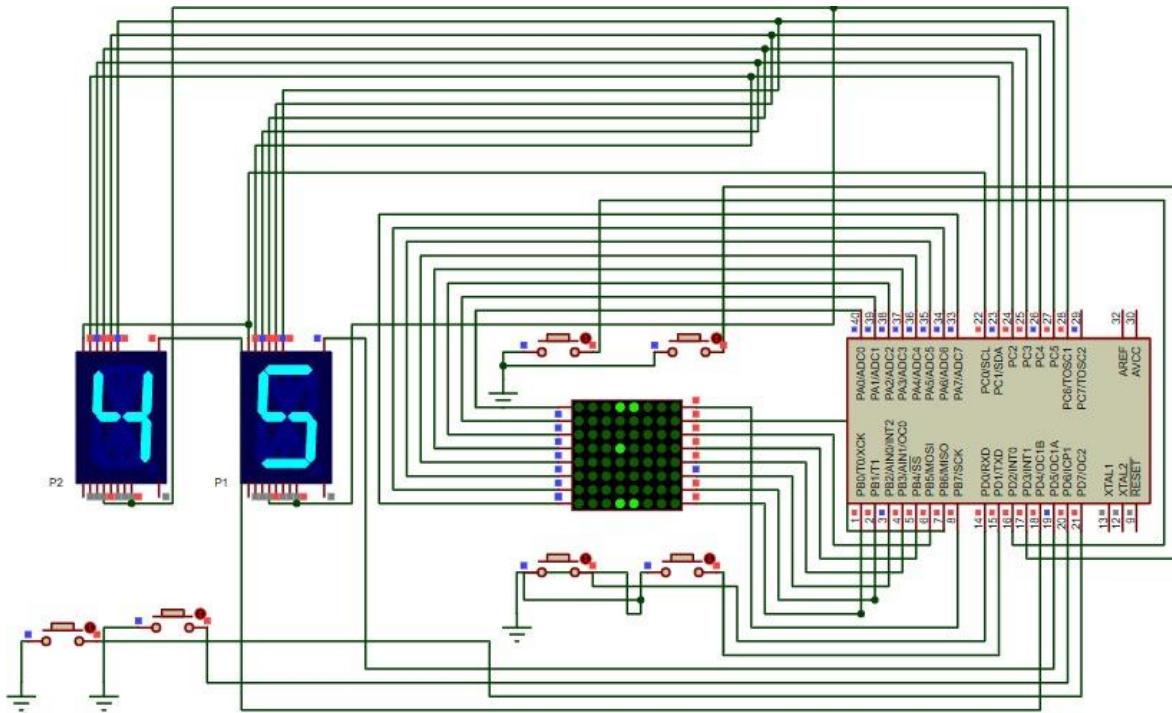


Imagen 1 simulación – proteus

Circuito

Pruebas y ajustes

Realizamos pruebas del juego para verificar su funcionamiento. Si es necesario, realizamos ajustes en el código para corregir errores o mejorar la jugabilidad. Pero en nuestro caso no tuvimos errores importantes, por lo que el código se mostrara mas adelante.

A continuación, en la imagen 2 podemos observar el correcto funcionamiento del circuito armado con pase en la imagen 1 el cual corresponde a la simulación mediante proteus.

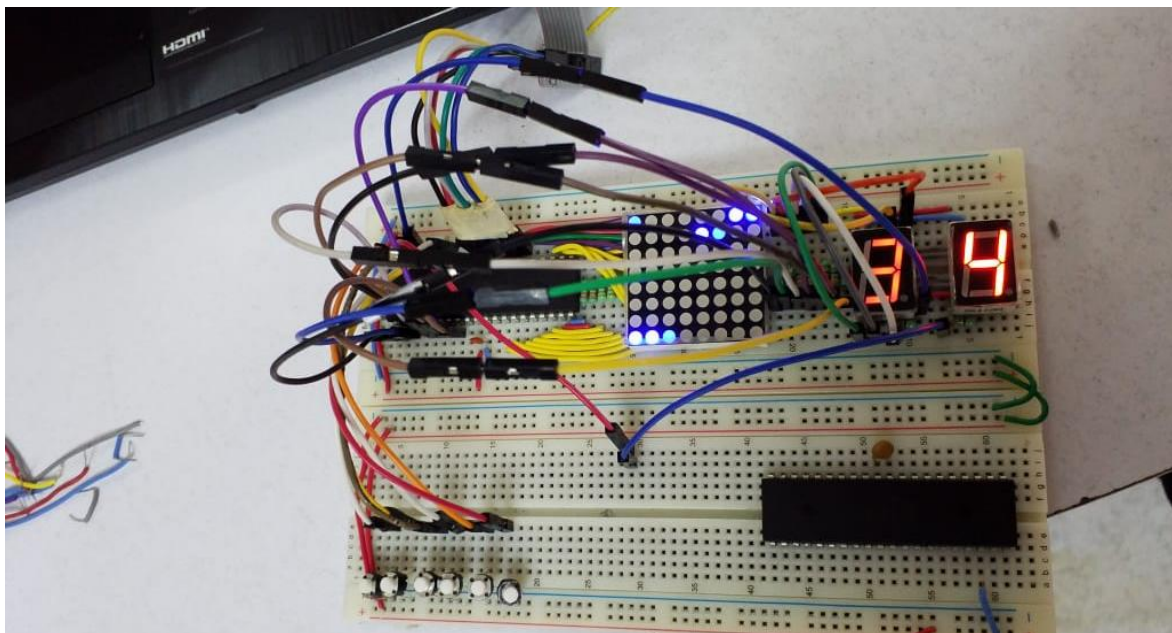


Imagen 2 circuito funcionando correctamente

Codificación

```

/*****
This program was created by the CodeWizardAVR V3.49a
Automatic Program Generator
♦ Copyright 1998-2022 Pavel Haiduc, HP InfoTech S.R.L.
http://www.hpinfotech.ro

Project : Proyecto - "Ping pong"
Version : 1.0
Date    : 24/05/2023
Author  : M A R D
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
*****/
```

```

***** /

// I/O Registers definitions
#include <mega8535.h>
#include <delay.h>
#include <stdlib.h>

#define inP1_right PINA.0 //pin 40
#define inP1_left PINA.1 //pin 39
#define inP2_right PINA.2 //pin 38
#define inP2_left PINA.3 //pin 37
#define in_reset PINA.4 //pin 36
#define in_play PINA.5 //pin 35

//Prototypes
void verVector(unsigned char myVector[7]);
void checa_boton (void);
void jugador1_move(int my_opc);
void jugador2_move(int my_opc);
void recorreVector_der();
void recorreVector_izq();
void pelota_move();
void verMarcador();

// Declare your global variables here
unsigned char num[7] = {0x06, 0x00, 0x00, 0x04, 0x00, 0x00, 0x0c};
bit botonp;
bit botona, game = 1;
bit dir; //direccion izq 0 - der 1
int opc, index_ball;
int dir_ball, timer; // 0 supizq / 1 sup / 2 supder / 3 infder / 4 inf / 5
indizq
unsigned char scoreP1, scoreP2;
const char tabla7segmentos[10]={0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7c,
0x07, 0x7f, 0x6f};

void main(void){
    // Declare your local variables here
    opc = -1;
    index_ball = 3;
    timer = 0;
    dir = 1;
    scoreP1 = 0;
    scoreP2 = 0;

```

```

dir_ball = 1 ;
// Input/Output Ports initialization
// Port A initialization
// Function: Bit7=Out Bit6=Out Bit5=In Bit4=In Bit3=In Bit2=In Bit1=In
Bit0=In
DDRA=(1<<DDA7) | (1<<DDA6) | (0<<DDA5) | (0<<DDA4) | (0<<DDA3) |
(0<<DDA2) | (0<<DDA1) | (0<<DDA0);
// State: Bit7=0 Bit6=0 Bit5=P Bit4=P Bit3=P Bit2=P Bit1=P Bit0=P
PORTA=(0<<PORTA7) | (0<<PORTA6) | (1<<PORTA5) | (1<<PORTA4) |
(1<<PORTA3) | (1<<PORTA2) | (1<<PORTA1) | (1<<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=Out Bit6=Out Bit5=Out Bit4=Out Bit3=Out Bit2=Out
Bit1=Out Bit0=Out
DDRC=(1<<DDC7) | (1<<DDC6) | (1<<DDC5) | (1<<DDC4) | (1<<DDC3) |
(1<<DDC2) | (1<<DDC1) | (1<<DDC0);
// State: Bit7=0 Bit6=0 Bit5=0 Bit4=0 Bit3=0 Bit2=0 Bit1=0 Bit0=0
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);

while (1){

    if(scoreP1 > 9 || scoreP2 > 9){
        game = 1;
    }//if

    checa_boton();

```



```
        if(game == 0){
            if(timer > 5){
                timer = 0;
                pelota_move();
            }else{
                timer++;
            }//if-else
        }//if
        verVector(num);
        verMarcador();

    }//while
} //main

void verVector(unsigned char myVector[7]){
    unsigned char i, v = 0x01;

    for(i = 0; i < 7; i++){
        PORTD = ~v;
        PORTC = myVector[i];
        delay_ms(3);
        v = v << 1;
    } //for
} //verVector

void checa_boton (void){
    if (inP1_right == 0){
        botona = 0;
        opc = 1;
    }else if(inP1_left == 0){
        botona = 0;
        opc = 2;
    }else if(inP2_right == 0){
        botona = 0;
        opc = 3;
    }else if(inP2_left == 0){
        botona = 0;
        opc = 4;
    }else if(in_reset == 0){
        botona = 0;
        opc = 5;
    }else if(in_play == 0){
```

```

        botona = 0;
        opc = 6;
    }else{
        botona=1;
    }//if-else

    if((botonp == 1) && (botona == 0)){ //hubo cambio de flanco de 1 a 0

        if(opc == 1 || opc == 2){
            jugador1_move(opc);
        }else if(opc == 3 || opc == 4){
            jugador2_move(opc);
        }else if(opc == 5){
            scoreP1 = 0;
            scoreP2 = 0;
            game = 1;
        }else{
            game = 0;
        }//if-else
        delay_ms(8); //40ms para eliminar rebotes
    }//if

    if((botonp == 0) && (botona == 1)){ //hubo cambio de flanco de 0 a 1
        delay_ms(8); //40ms para eliminar rebotes
    }//if
    botonp = botona;
    opc = -1;
} //checa_boton

void jugador1_move(int my_opc){
    unsigned char temp;

    if(my_opc == 1){
        temp = num[0] << 1;
        if(temp > 0x18){
            num[0] = 0x18;
        }else{
            num[0] = temp;
        }//if-else
    }else{
        temp = num[0] >> 1;
        if(temp < 0x03){
            num[0] = 0x03;
        }else{

```

```
num[0] = temp;
} //if-else

} //if-else

} //jugador1_move

void jugador2_move(int my_opc){
    unsigned char temp;

    if(my_opc == 3){
        temp = num[6] << 1;
        if(temp > 0x18){
            num[6] = 0x18;
        } else{
            num[6] = temp;
        } //if-else
    } else{
        temp = num[6] >> 1;
        if(temp < 0x03){
            num[6] = 0x03;
        } else{
            num[6] = temp;
        } //if-else
    } //if-else
} //jugador2_move

void recorrerVector_der(){
    unsigned int r_num = rand() % 6;
    if(index_ball + 1 == 6){
        dir = 0;
        if((num[index_ball + 1] ^ num[index_ball]) > num[index_ball + 1]){
            scoreP2++;
            num[index_ball] = 0x00;
            index_ball = 3;
            num[index_ball] = 0x04;
            //game = 1;
        } else{
            if(r_num < 3){
                dir_ball = r_num;
            } else{
```

```
        dir_ball = 1;
    }//if-else

} //if-else

}else{
    num[index_ball + 1] = num[index_ball];
    num[index_ball] = 0x00;
    index_ball++;
} //if-else

} //recorreVector_der

void recorrerVector_izq(){
    unsigned int r_num = rand() %6;
    if(index_ball - 1 == 0){
        dir = 1;
        if((num[index_ball - 1] ^ num[index_ball]) > num[index_ball - 1]){
            scoreP1++;
            num[index_ball] = 0x00;
            index_ball = 3;
            num[index_ball] = 0x04;
            //game = 1;
        }else{
            if(r_num >= 3){
                dir_ball = r_num;
            }else{
                dir_ball = 4;
            } //if-else
        } //if-else
    }else{
        num[index_ball - 1] = num[index_ball];
        num[index_ball] = 0x00;
        index_ball--;
    } //if-else
} //recorreVector_izq

void pelota_move(){
    unsigned char temp;

    if(dir == 1){ //abajo
        switch(dir_ball){
```

```
case 3:{
    temp = num[index_ball] >> 1;
    if(temp != 0x00){
        num[index_ball] = temp;
        recorreVector_der();
    }else{
        dir_ball = 5;
        recorreVector_der();
    }//if-else

    break;
}
case 4:{
    recorreVector_der();
    break;
}
case 5:{
    temp = num[index_ball] << 1;
    if(temp != 0x10){
        num[index_ball] = temp;
        recorreVector_der();
    }else{
        dir_ball = 3;
        recorreVector_der();
    }//if-else
    break;
}
default:
    dir_ball = 4;
    break;
} //switch

} else{
    switch(dir_ball){
        case 2:{
            temp = num[index_ball] >> 1;
            if(temp != 0x00){
                num[index_ball] = temp;
                recorreVector_izq();
            }else{
                dir_ball = 0;
                recorreVector_izq();
            }//if-else
            break;
        }
    }
}
```



```

        case 1:{
            recorreVector_izq();
            break;
        }
        case 0:{
            temp = num[index_ball] << 1;
            if(temp != 0x10){
                num[index_ball] = temp;
                recorreVector_izq();
            }else{
                dir_ball = 2;
                recorreVector_izq();
            }//if-else
            break;
        }
        default:
            dir_ball = 1;
            break;
    }//switch

} //if-else

} //pelota_move

void verMarcador(){
    PORTA.7 = 0;
    PORTB = 0x00;
    PORTA.6 = 1;
    PORTB = tabla7segmentos[scoreP1];
    delay_ms(2);

    PORTA.6 = 0;
    PORTB = 0x00;
    PORTA.7 = 1;
    PORTB = tabla7segmentos[scoreP2];
    delay_ms(2);
} //verMarcador

```



Conclusiones


Bocanegra Heziquio Yestlanezi

Del proyecto final puedo concluir que con base en nuestros conocimientos adquiridos en la materia realizando las practicas anteriores, el diseñar un juego de Ping-Pong utilizando una matriz de LEDs, un display de 7 segmentos y botones para controlar la raqueta es un proyecto interesante para mí ya que combina electrónica y programación. Durante el proceso de desarrollo, se requiere una planificación cuidadosa, que incluye la selección de los componentes adecuados y la conexión correcta de los mismos a la placa Arduino, pero en este caso tuvimos la opción de no utilizar la placa y montarlo directamente en el protoboard.

La matriz de LEDs permite representar visualmente la pelota y las paredes del juego, mientras que el display de 7 segmentos muestra la puntuación del jugador. Los botones se utilizan para controlar el movimiento de la raqueta, lo que añade una interacción física al juego.

Dominguez Durán Alan Axel

En esta última práctica, implementamos un conjunto de conceptos para lograr el objetivo principal de la práctica, primeramente, para el tablero de juego, se utilizó una matriz de leds para mostrar las 2 raquetas y la pelota del juego. Además de que se implementaron un total de 6 push buttons sin rebote y con interrupción para distintas funciones, 2 eran del primer jugador, para poder mover la raqueta que le corresponde de izquierda a derecha, al igual que el jugador 2; finalmente los otros dos botones se utilizaron para iniciar el juego y para reiniciarlo respectivamente. La implementación final fueron los displays, los cuales mostraban el puntaje de cada jugador, los cuales al rebasar el número nueve se reiniciaban.





Hernandez Mendez Oliver Manuel

En esta última práctica, se logró el objetivo principal mediante la implementación de varios elementos. En primer lugar, se utilizó una matriz de LEDs para representar las raquetas y la pelota del juego en el tablero. Además, se incorporaron un total de 6 pulsadores con interrupción y sin rebote para diversas funciones. Esta integración resultó en un juego interactivo y visualmente atractivo, brindando a los jugadores una experiencia completa y emocionante.

Martinez Cruz José Antonio

En esta última práctica, la cual es el tercer proyecto, se terminó aplicando todo lo que fue aprendido de las prácticas y parte de los proyectos anteriores. Cada uno de los módulos establecidos fue desarrollado de manera satisfactoria, tal como el uso de la matriz de led, el uso de interrupciones externas, asignación de push button para la interacción de los usuarios, así como el uso de displays para obtener una transmisión de información clara entre el usuario y el microprocesador.





Referencias

- [1] "Design and Implementation of BCD Counter with Digital Display using VHDL", en IEEE Xplore Digital Library, <https://ieeexplore.ieee.org/document/7601309>.

