**UNL**

***U****niversidad****N****acional de****L****oja*

Facultad de la Energía,

Las Industrias y los Recursos

Naturales no Renovables

“Implementación HARDWARE Y SOFTWARE DE UNA CERRADURA digital”

Autor:

Deiby Patricio Calva Pintado.

Docente:

Ing. Ángel Freddy Ganazhapa.

Periodo Académico

OCTUBRE-MARZO

1. **INTRODUCCIÓN**

La seguridad es la segunda necesidad más importante del ser humano, se ubica sólo por debajo de la necesidad de subsistencia, esta necesidad: ha impulsado desde tiempos antiguos a los hombres para desarrollar elementos, técnicas y mecanismos de protección para resguardar sus pertenencias.

Este proyecto fue elaborado para contar con un sistema de seguridad electrónico, por medio de una cerradura. En este campo de aplicación fue basada en la puerta de una casa que se lleva a cabo con el propósito de no necesitar de una llave para este tipo de seguridad, ye que es muy frecuente perder u olvidar las llaves.

Nuestra práctica conlleva en elaborar un dispositivo que permita controlar la seguridad de una cerradura realizado en Lenguaje C en el IDE Atmel Studio. En este proceso se utiliza la placa **Atmega2560**, que a través de la configuración de sus puertos se lo va a implementar, en la cual se podrá ingresar mediante una contraseña programable. Existirán 2 contraseñas, una contraseña maestra la cual no será posible cambiar, y otra la cual será posible cambiar mediante la contraseña maestra. Solo se podrá ingresar con la segunda contraseña con la que podamos tener mayor seguridad en nuestro hogares, cuartos y sistemas en dónde se requiera seguridad.

1. **DISEÑO DE ALTO NIVEL**

**2.1. Justificación del Proyecto.**

El proyecto que aquí se plantea surge como una solución a los problemas más comunes que presentan las puertas convencionales y a los inconvenientes que poseen en cuanto a seguridad. Aprovechando las ventajas de las cerraduras inteligentes y, en conjunto con las bondades que actualmente brindan los microcontroladores de ATMEL, se desarrolló una solución práctica, innovadora y de bajo consumo. Donde se pretende demostrar que, mediante un sistema electrónico acoplado a una cerradura se puede mejorar la seguridad, resolviendo de esta manera algunas de las carencias de los sistemas actuales.

**2.2. Descripción del Hardware y Software**

Para el desarrollo de la aplicación se utiliza el IDP **Atmel Studio 7.** En esta plataforma de desarrollo integrado es donde se creará el código que permite manipular los pines de la placa Atmega2560. Se utiliza librerías que permite imprimir mensajes, tiempo de retardo entre otras librerías.

Para empezar a programar lo primero que se debe hacer es definir las variables que se va a emplear. Para la actividad se empleará una frecuencia master de 16000000 Hz con un tiempo retarda de 1000 ms. Es necesario incluir en la programación y especificar el valor de cristal de cuarzo que se utilizará como reloj en el microcontrolador; se incluye las librerías para la comunicación de la pantalla LCD para mostrar los mensajes que el microcontrolador envía, se incluye la librería del teclado matricial a utilizar (en este caso se usó un teclado de cuatro filas y tres columnas). Por último, se declara la dirección de memoria EEPROM (los pines de salida-entrada) y se guarda en la memoria una contraseña Padre o Maestra que será la clave para luego ser ingresada una nueva clave y puede ser cambiada las veces que desee el usuario para poder tener acceso.

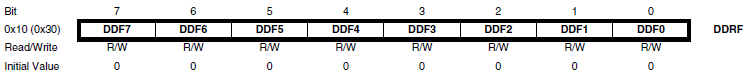
Seguidamente el microcontrolador estará siempre leyendo el teclado, esperando a que una tecla sea pulsada. Si la contraseña es ingresada correctamente el microcontrolador tomará este dato como una clave ingresada y este se almacenará en la EEPROM para ser posteriormente comparada con la clave guardada en el banco de memoria; si esta clave es correcta se envía una señal que activará el servomotor en el DDRB=(1<<PB5) y al Led de luz verde PORTF=0x04, a la vez enviará un mensaje que será visualizado en la pantalla LCD; seguidamente el microcontrolador esperará una señal que será enviada por el sensor magnético a uno de los pines de este DDRB=(1<<PB5), para que el servomotor tome la posición de abierto y después de unos 5000 ms volverá coger la posición actual del servomotor Si la clave llegase a ser errónea, el microcontrolador detectará esto como INCORRECTO, y se activara el sonido del buzzer y Led Rojo de alerta incorrecto en el PORTF=0x03.

Al momento de ingresar la contraseña Maestra el microcontrolador pedirá que ingrese una nueva clave actual para poder cambiar la contraseña. Se comparan las claves, si fuese correcto se pedirá ingresar la nueva clave a almacenar en la memoria EEPROM y se reiniciará el sistema con la clave nueva guardada, si llegase a ser la clave incorrecta el microcontrolador volverá a descartar la apertura de la Cerradura siendo así un ciclo While.

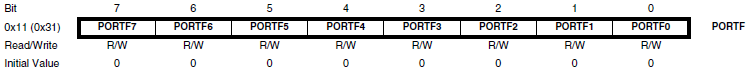
**2.3. Normas Técnicas.**

A continuación, se describe todos los registros utilizados en el programa según la hoja de datos del microprocesador **Atmega 2560**.

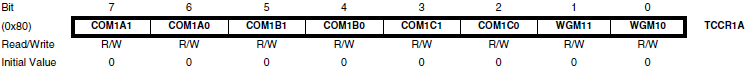
**2.3.1. Descripción del registro de los Puertos de E/S**

* **DDRF**

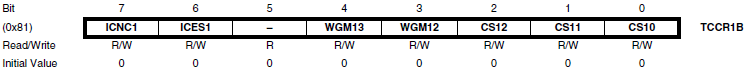
*Ilustración 1: Registro de Dirección de Datos del Puerto F*

* **PORTF**

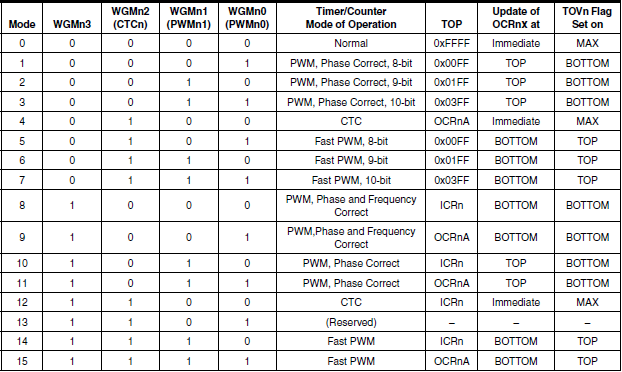
*Ilustración 2: Registro de datos del Puerto F*

* **TCCR1A**

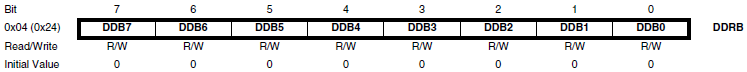
*Ilustración 3: Registro de control A, Timer/Counter 1*

* **TCCR1B**

*Ilustración 4: Registro de control B, Timer/Counter 1*

* WGM11 del registro TCCR1A y WGM12, WGM13 del registro TCCR1B nos ayudan a configurar las formas de onda para el timer de acuerdo a la siguiente tabla:

*Ilustración 5: Los modos 1,2,3,5,6,7 corresponden a los 6 modos PWM estándar*

* **DDRB**

*Ilustración 5: Registro de Dirección de Datos del Puerto B*

**2.4. Patentes**

|  |  |
| --- | --- |
|  | Esta publicado por Instructables, es una comunidad de usuarios que crean recursos educativos de manera que nos permite aprender desde arte hasta tecnología, navegando paso a paso donde nos mostrarán una secuencia de imágenes por cada paso cómo realizar proyectos, y además encontraremos notas en las mismas para facilitarnos mejor la compresión. |
| **Theme** | Keypad Door Lock |
| **Link** | <https://www.instructables.com/id/Keypad-Door-Lock/> |
| **Link\_Archivo\_Implementado**  **GitHub** | <https://github.com/DeibyCalva/ProyectoFinal_Dise-oDigital> |

*Tabla 1: Patente del proyecto Cerradura Digital.*

1. **PROYECTACIÓN HARDWARE**

* **ARDUINO ATMEGA 2560:** es una placa de desarrollo basada en el microcontrolador ATmega2560. Tiene 54 entradas/salidas digitales (de las cuales 15 pueden ser usadas como salidas PWM), 16 entradas analógicas, 4 UARTs, un cristal de 16Mhz, [conexión USB](https://www.mcielectronics.cl/shop/product/cable-usb-a-macho-b-macho-6ft-1-82m-9602), jack para [alimentación DC](https://www.mcielectronics.cl/shop/product/transformador-ac-dc-9v-650ma-regulado-jack-dc-5-5x2-1mm-9591), conector ICSP, y un botón de reseteo. La placa Mega 2560 es compatible con la mayoría de shields compatibles para Arduino UNO. (cl, s.f.)

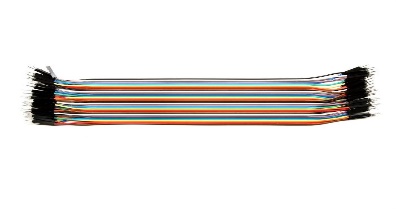


*Figura 1: -ARDUINO ATMEGA 2560*

* **Buzzer:** Un zumbador o mejor conocido como buzzer (en ingles) es un pequeño transductor capaz de convertir la energía eléctrica en sonido. Para hacerlos funcionar solo basta conectar el positivo con el + y la tierra o negativo con el – de una batería o cualquier fuente de corriente directa**.**

*Figura 2: -* *Buzzer*

* **Cables dupont:** Los cables jumpers dupont facilitan la conexión en prototipos, sensores y otros dispositivos electrónicos. Son perfectos para usar con micro controladores como Arduino, Raspberry, AVR entre otros.

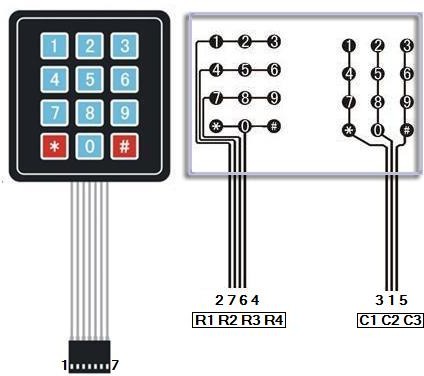


*Figura 3: -* *Cables dupont*

* **key pad (teclado) de 4x3:** Un teclado no es más que una colección de botones, a cada uno de los cuales le asignamos un símbolo o una función determinada.

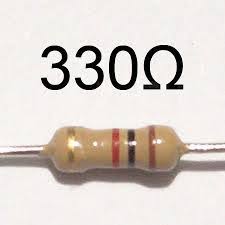
Leer botones es algo que ya no tiene secretos para nosotros, pero si conectáramos cada tecla a un pin digital de nuestro Arduino, pronto estaríamos en apuros.El teclado de nuestro ordenador suele ser de alrededor de 106 teclas, así que el método de fuerza bruta va a entrar en apuros rápidamente. Necesitamos otra solución.Y como el mundo está lleno de gente ingeniosa se les ocurrió una solución de lo más elegante, una matriz de teclas.

Vamos a ver un ejemplo con un pequeño teclado numérico de 16 teclas tipo los de los teléfonos móviles o los de los cajeros automáticos.



*Figura 4: -* *key pad (teclado) de 4x3*

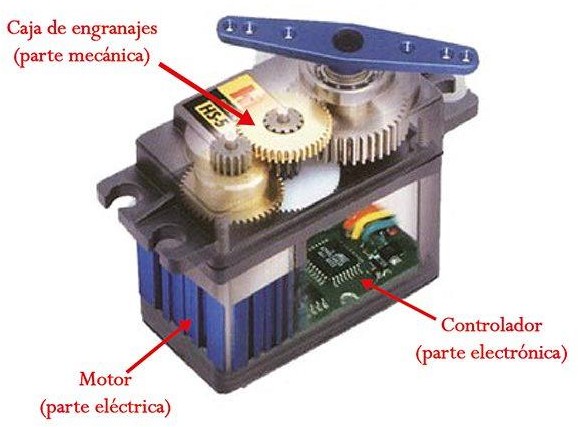
* **Resistencia**: Las resistencias son uno de los tipos básicos de componentes electrónicos. Tienen dos terminales y un semiconductor, está formada por carbón y otros elementos resistivos.



*Figura 5: -* *Resistencia*

* **Servomotor:** Un servomotor es un tipo especial de motor que permite controlar la posición del eje en un momento dado. Está diseñado para moverse determinada cantidad de grados y luego mantenerse fijo en una posición. El motor en el interior de un servomotor es un motor DC común y corriente. El eje del motor se acopla a una caja de engranajes similar a una transmisión. El circuito electrónico es el encargado de manejar el movimiento y la posición del motor.

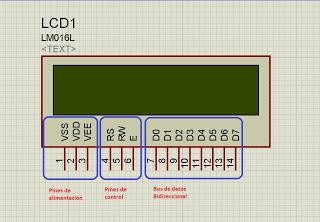
La presencia del sistema de engranajes como el que se muestra en la figura hace que cuando movemos el eje motor se sienta una inercia muy superior a la de un motor común y corriente. Observando las imágenes que hemos presentado nos podemos dar cuenta que un servo no es un motor como tal, sino un conjunto de partes (incluyendo un motor) que forman un sistema.



*Figura 6: Servomotor*

* **Display LDC 16x2:** El LCD(Liquid Crystal Dysplay) o pantalla de cristal líquido es un dispositivo empleado para la visualización de contenidos o información de una forma gráfica, mediante caracteres, símbolos o pequeños dibujos dependiendo del modelo. Está gobernado por un microcontrolador el cual dirige todo su funcionamiento.

En este caso vamos a emplear un LCD de 16x2, esto quiere decir que dispone de 2 filas de 16 caracteres cada una. Los píxeles de cada símbolo o carácter, varían en función de cada modelo.



*Figura 7: -Display LDC 16x2*

**Pines de alimentación:**

VSS: Gnd

VDD: +5 voltios

VEE: corresponde al pin de contraste, lo regularemos con un potenciómetro de 10K conectado a Vdd.

**Pines de control:**

RS: Corresponde al pin de selección de registro de control de datos (0) o registro de datos(1). Es decir el pin RS funciona paralelamente a los pines del bus de datos. Cuando RS es 0 el dato presente en el bus pertenece a un registro de control/instrucción. y cuando RS es 1 el dato presente en el bus de datos pertenece a un registro de datos o un carácter.

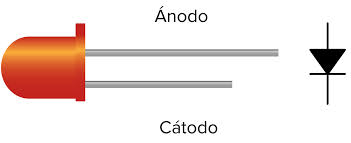
RW: Corresponde al pin de Escritura(0) o de Lectura(1). Nos permite escribir un dato en la pantalla o leer un dato desde la pantalla.

E: Corresponde al pin Enable o de habilitación. Si E(0) esto quiere decir que el LCD no esta activado para recibir datos, pero si E(1) se encuentra activo y podemos escribir o leer desde el LCD.

**Pines de Bus de datos:**

El Bus de datos bidireccional comprende desde los pines D0 a D7. Para realizar la comunicación con el LCD podemos hacerlo utilizando los 8 bits del bus de datos(D0 a D7) o empleando los 4 bits mas significativos del bus de datos(D4 a D7). En este caso vamos a explicar la comunicación con el bus de 4 bits.

* **Led:** es un dispositivo semiconductor que emite luz incoherente de espectro reducido cuando se polariza de forma directa la unión PN en la cual circula por él una corriente eléctrica.



*Figura 8: - Led*

* **Potenciómetro:** Un potenciómetro es un resistor eléctrico con un valor de resistencia variable y generalmente ajustable manualmente. Los potenciómetros utilizan tres terminales y se suelen utilizar en circuitos de poca corriente, para circuitos de mayor corriente se utilizan los reostatos. En muchos dispositivos eléctricos los potenciómetros son los que establecen el nivel de salida. Por ejemplo, en un altavoz el potenciómetro ajusta el volumen; en un televisor o un monitor de ordenador se puede utilizar para controlar el brillo.



*Figura 9: - Potenciómetro*

* **Protoboard:** Es una placa de pruebas para electrónica que contiene numerosos orificios en los que es posible insertar cables y otros elementos electrónicos para montar circuitos provisionales.

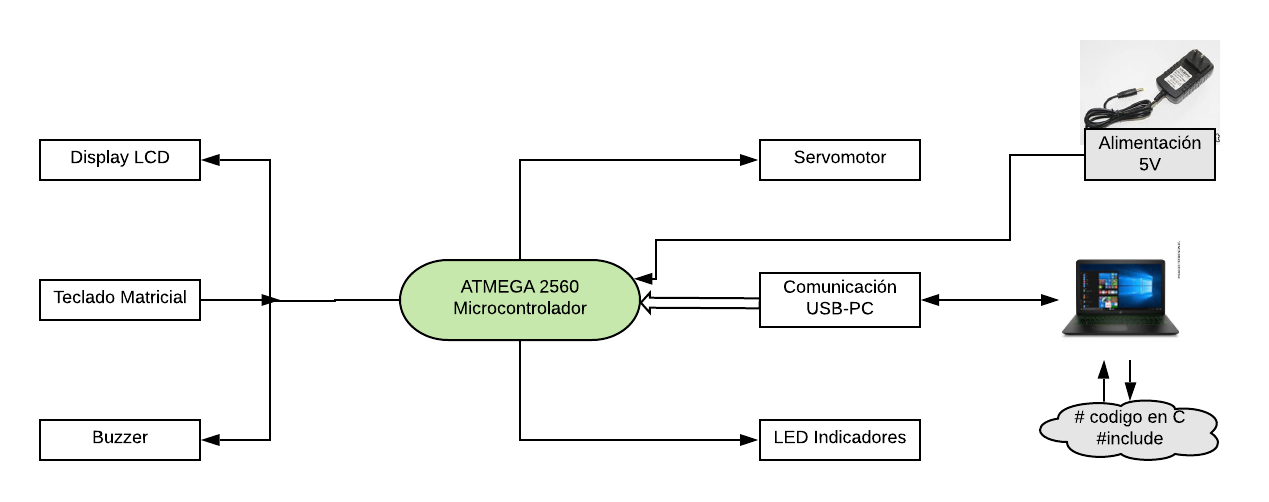


*Figura 10: - Protoboard*

* **Costo Materiales Hardware.**

|  |  |
| --- | --- |
| Materiales | Precio $ |
| * 2 leds, green, red * 1 Display LDC 16x2 * 1 ServoMotor * 2 resistores de 330 Ω * 1 keyPad(Teclado Matricial) 4x3 * Cables dupont * 1 buzzer * 1 Potenciómetro 5kΩ * 1 Arduino Admega 2560 * 1 Protoboard | - 0.20ctv   * - 4 $ * - 5 $ * - 0.20ctv * - 4 $ * - 1 $ * - 0.50ctv * - 0.50ctv * - 25$ * - 4 $ |
| **TOTAL:** | **44.40 $** |

*Tabla 2: Costo de materiales.*

1. **DIAGRAMA DE BLOQUES.**

*Ilustración 6: Diagrama de Bloques*

La figura nos muestra el diagrama de bloques del proyecto, en donde se presenta lo siguiente:

1. **Comunicación por USB-PC o Alimentación 5v.** Este módulo permite la interconexión entre la caja y el computador y una vez esté conectado cargar el código al Atmega 2560 para luego poder también dar uso solo de la alimentación de 5v.
2. **Acceso por teclado.**  Luego de que haya sido cargado el código al Microcontrolador se hace uso de un teclado matricial, las cuales al ser presionadas son detectadas por un microcontrolador y mostradas en display LCD como “números enteros”. Si la contraseña ingresada es distinta a la guardada en la memoria del microcontrolador, se mostrará *“contraseña incorrecta”* activando el puerto del buzzer emitiendo un sonido junto con el LED ROJO en cambio, si es igual se mostrará *“CORRECTO”,* para luego ingresar una nueva contraseña permitiendo controlar al Servomotor Esta última mostrará seguidamente el mensaje *“ABIERTO”* y a su vez activa un LED verde como indicador. Posteriormente el circuito mandará una señal que active el servomotor haciendo que gire 90° y abriendo la caja para luego de 5000 ms se vuelva a cerrar.
3. **DISEÑO DE SOFTWARE**

**Conceptos**

**Atmel Sudio:** es la plataforma de desarrollo integrado (IDP) para desarrollar y depurar todas las aplicaciones de microcontroladores AVR® y SAM. El IDP de Atmel Studio  brinda un entorno transparente y fácil de usar para escribir, compilar y depurar sus aplicaciones escritas en C / C ++ o código de ensamblaje. También se conecta sin problemas a los depuradores, programadores y kits de desarrollo que admiten dispositivos AVR ® y SAM. (MicroChip, s.f.)

**Directiva #define:** La directiva define se usa para definir un identificador y una cadena que el compilador sustituirá por el identificador cada vez que se encuentre en el archivo fuente.  (Enrique Vicente Bonet Esteban)

**Directiva #include:** La directiva #include fuerza al compilador a incluir otro archivo fuente en el archivo que tiene la directiva #include, y a compilarlo. El nombre del archivo fuente a incluir se colocará entre comillas dobles o entre paréntesis de ángulo.

**#include:** contiene los prototipos de las funciones, macros, y tipos para manipular datos de entrada y salida. (José Norbey Sánchez F.)

**#include <avr/io.h>:** Este archivo de encabezado incluye las definiciones de E / S apropiadas para el dispositivo que ha sido especificado por el -mmcu=interruptor de línea de comandos del compilador. (MicroChip, s.f.)

**#include <avr / eeprom.h>:** Este archivo de encabezado declara la interfaz a algunas rutinas de biblioteca simples adecuadas para manejar los datos EEPROM contenidos en los microcontroladores AVR. La implementación utiliza una interfaz de modo de sondeo simple. Las aplicaciones que requieran acceso EEPROM controlado por interrupción para garantizar que no se pierda tiempo en spinloops tendrán que implementar su propia implementación.

**Implementación Del Diseño de Software**

/\*

\* ProyectoFinal\_DeibyCalva.c

\*

\* Created: 27/2/2020 17:48:59

\* Author : Deiby Calva

\*/

#define F\_CPU 16000000UL // Frecuencia de CPU en Hz.

#include <avr/io.h> // contiene la definicion de registros y sus respectivos bits, nombres logicos de cada nombre PIN (OCR0A, OCIE0A)

#include <stdlib.h> // contiene funciones basicas es necesaria para poder utilizar la funcion itoa(), que transforma un valor entero en una cadena de caracteres

#include <stdbool.h> // permite usar bool como un tipo de datos booleano. true evalúa como 1 y false evalúa como 0

#include <util/delay.h>// Funciones convenientes para bucles de retardo de espera ocupada

#include <avr/eeprom.h>// Esta biblioteca le permite leer y escribir esos bytes

#include "lcd.c" //libreria o mas bien es un codigo desarrollado que contiene algunas de las funciones mas cotidianas para el uso del LCD

#include "mat\_kbrd.c" // se carga la libreria del teclado

#include "uart.c" //Trasmisor receptor asincrono universal controla los puertos y dispositivos en serie

int main(void)

{

/\* Replace with your application code \*/

void Wait() // Realiza una espera para que el servomotor pueda girar

{

*uint8\_t* i;

for(i=0;i<10;i++)

{

//Espera un tiempo

*\_delay\_loop\_2*(0);

*\_delay\_loop\_2*(0);

*\_delay\_loop\_2*(0);

}

}

DDRF=0x07; // Puertos de salida

PORTF=0x00; // Inician en Cero

#define eeprom\_true 0

#define eeprom\_data 1

*FILE* uart\_str = *FDEV\_SETUP\_STREAM*(uart\_putchar, uart\_getchar, *\_FDEV\_SETUP\_RW*);

char claveMaestra[5]= "1\*26"; // Clave maestra que sirve para cambiar la contraseña hija

bool Bmaestra = false; // Sirve para saber si se ha ingresado la clave maestra

eeprom\_write\_word(1, claveMaestra[0] - '0') ; // se guarda la clave maestra en la eeprom

eeprom\_write\_word(2, claveMaestra[1] - '0') ;

eeprom\_write\_word(3, claveMaestra[2] - '0') ;

eeprom\_write\_word(4, claveMaestra[3] - '0') ;

char cl[4]; // SE recupera de la eeprom la clave Maestra

cl[0] = eeprom\_read\_word(1) + '0';

cl[1] = eeprom\_read\_word(2) + '0';

cl[2] = eeprom\_read\_word(3) + '0';

cl[3] = eeprom\_read\_word(4) + '0';

char claveNueva[4]; // se recupera de la eeprom la clave que ingreso el usuario

claveNueva[0] = eeprom\_read\_word(6) ;

claveNueva[1] = eeprom\_read\_word(7) ;

claveNueva[2] = eeprom\_read\_word(8) ;

claveNueva[3] = eeprom\_read\_word(9) ;

char key; // variable de caracter

int i=0; // inicia en cero

lcd\_init(LCD\_DISP\_ON); // se inicia o activa la LCD

kbrd\_init(); // Se inicia el teclado

lcd\_home(); // Esa ubicación al enviar el texto posterior a la pantalla.

lcd\_clrscr(); //Limpiar la pantalla del LCD

lcd\_puts("INGRESE CLAVE MA: ");//Imprime en la LCD el mensaje

*\_delay\_ms*(50); // 50 ms de retraso

char cont[5]="00000"; // variable de contador de caracteres en un arreglo

int x=0;

while (1)

{

key = kbrd\_read(); // LEE EL TECLADO

if (key != 0)

{

PORTF=0x01; // Se activa el puerto del buzzer para que suene al momento de digitar cada tecla

*\_delay\_ms*(50);// 50 ms de retraso

PORTF=0x00; // se desactiva el puerto del buzzer

lcd\_gotoxy(i, 1); //Mueve el cursor a la posicion x=i y=0

cont[i]= key; // i se almacena en la variable contador

lcd\_putc(cont[i]);// imprime el caracter i en la posicion 0 y esto va ir

i++; // incrementando hasta que llegue las teclas del matricial

if (!Bmaestra)

{

if (i==4 && Bmaestra == false)

{

if (cont[0] == cl[0] &&cont[1] == cl[1] && cont[2] == cl[2] && cont[3] == cl[3])// compara la contraseña ingresada por el usuario

// con la contraseña maestra

{

lcd\_clrscr(); //Limpiar la pantalla del LCD

lcd\_puts("CORRECTO");//Imprime en la LCD el mensaje

PORTF=0x04; // Se activa el puerto del buzzer y LED en PORTF2

*\_delay\_ms*(1000); //1000 ms de retraso

PORTF=0x00; //se desactiva el puerto del buzzer y del LED

Bmaestra = true;

i=0;

x=0;

lcd\_clrscr(); //Limpiar la pantalla del LCD

lcd\_puts("NUEVA CLAVE: ");//Imprime en la LCD el mensaje

*\_delay\_ms*(2000); //2000 ms de retraso

}

else

{ //Compara la contraseña ingresada ingresada con la contraseña normal

if (cont[0] == claveNueva[0] &&cont[1] == claveNueva[1] && cont[2] == claveNueva[2] && cont[3] == claveNueva[3])

{

lcd\_clrscr(); //Limpiar la pantalla del LCD

lcd\_puts("ABIERTO");//Imprime en la LCD el mensaje

PORTF=0x04; // Se activa el puerto del buzzer y LED en PORTF2

*\_delay\_ms*(1000); //1000 ms de retraso

PORTF=0x00; //se desactiva el puerto del buzzer y del LED

TCCR1A|=(1<<COM1A1)|(1<<COM1B1)|(1<<WGM11); //NON Inverted PWM

TCCR1B|=(1<<WGM13)|(1<<WGM12)|(1<<CS11)|(1<<CS10); //PRESCALER=64 MODE 14(FAST PWM)

ICR1=4999; //fPWM=50Hz (Period = 20ms Standard).

DDRB=(1<<PB5); //PWM Pins as Out

OCR1A=120; //0 degree

Wait();

ICR1=0;

*\_delay\_ms*(5000);

ICR1=4999;

OCR1A=425; //90 degree

Wait();

ICR1=0;

i=0;

lcd\_clrscr();//Limpiar la pantalla del LCD

lcd\_puts("INGRESE CLAVE: ");//Imprime en la LCD el mensaje

}

else

{

i=0;

lcd\_clrscr();//Limpiar la pantalla del LCD

lcd\_puts("INCORRECTO");//Imprime en la LCD el mensaje

PORTF=0x03; // Se activa el puerto del buzzer y LED en PORTF0 PORTF0 Encendiendo el LED rojo y sonido de alerta

*\_delay\_ms*(2000);//2000 ms de retraso

PORTF=0x00; //se desactiva el puerto del buzzer y del LED

lcd\_clrscr(); //Limpiar la pantalla del LCD

lcd\_puts("INGRESE CLAVE: "); //Imprime en la LCD el mensaje

}

}

}

}

}

if (Bmaestra )

{

if (i==4)

{ // se guarda en la eeprom la nueva contraseña

eeprom\_write\_word(6, cont[0] ) ;

eeprom\_write\_word(7, cont[1] ) ;

eeprom\_write\_word(8, cont[2] ) ;

eeprom\_write\_word(9, cont[3] ) ;

// la va recuperar de la eeprom la contraseña recientemente cambiada

claveNueva[0] = eeprom\_read\_word(6) ;

claveNueva[1] = eeprom\_read\_word(7) ;

claveNueva[2] = eeprom\_read\_word(8) ;

claveNueva[3] = eeprom\_read\_word(9) ;

lcd\_clrscr(); //Limpiar la pantalla del LCD

lcd\_puts("SE HA CAMBIADO"); //Imprime en la LCD el mensaje

PORTF=0x04; // Se activa el puerto del buzzer y LED en PORTF2

*\_delay\_ms*(3000); // 3000 ms de retraso

PORTF=0x00; //se desactiva el puerto del buzzer y del LED

lcd\_clrscr(); //Limpiar la pantalla del LCD

lcd\_puts("INGRESE CLAVE: ");//Imprime en la LCD el mensaje

*\_delay\_ms*(50); //2000 ms de retraso

i=0;

x=0;

Bmaestra =false;

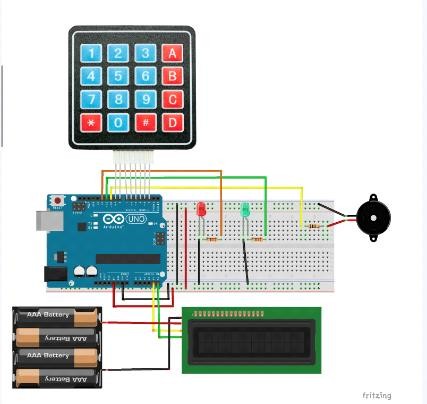
}

}

}

}

1. **RESULTADOS OBTENIDOS.**

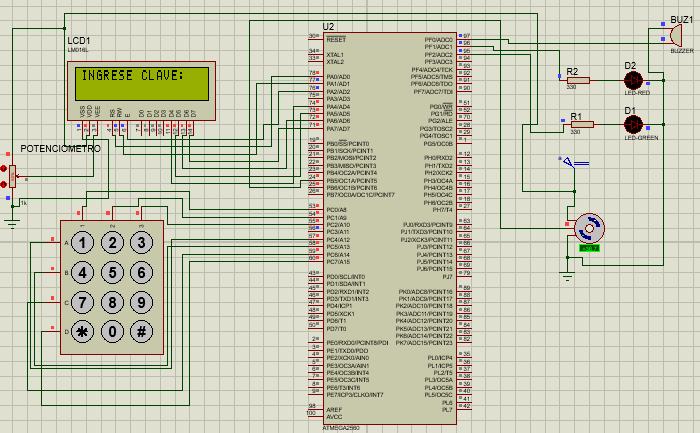
 **- Conexión Dispositivo.**

*Ilustración 7: Conexión Dispositivo*

**- Precisión**

Luego de concluir la programación para el microcontrolador, fue necesario el uso de un software de simulación PROTEUS (ver *Ilustración 8*) con la finalidad de revisar paso a paso la programación, conexión y evitar futuros errores de compilación. Posteriormente se realizaron pruebas en un Protoboard conectándolo ya en físico.

Luego fue necesario el uso de una caja pequeña de MDF y cartón para realizar la maqueta de la Cerradura Digital en la parte frontal (tapa) se hizo orificios para poder colocar la pantalla LCD, el teclado matricial tipo membrana, los leds indicadores y un pequeño jalador para poder abrir y cerrar la puerta (Ver ANEXOS). En la parte posterior hicimos otro agujero para colocar la fuente de alimentación para todos los componentes. Después se decoró toda la parte exterior de la caja, se fijó todos los componentes en lugares previstos con anterioridad.

****En la parte de ANEXOS se puede observar cómo quedó la maqueta en la parte exterior e interior respectivamente después de haber concluido con la implementación final del prototipo.

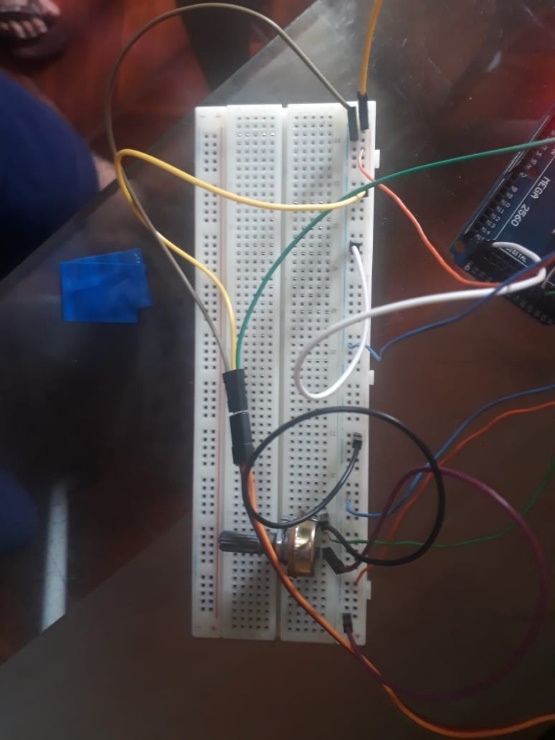
*Ilustración 8: Simulación en Proteus.*

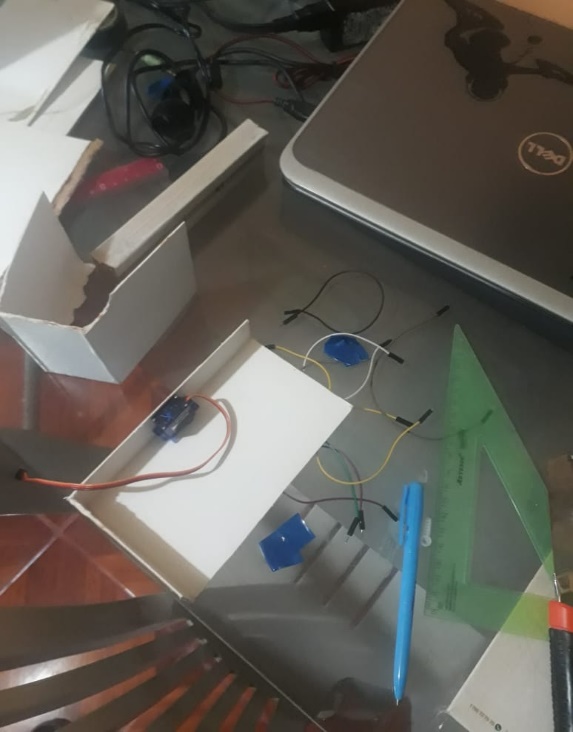
1. **CONCLUSIONES**

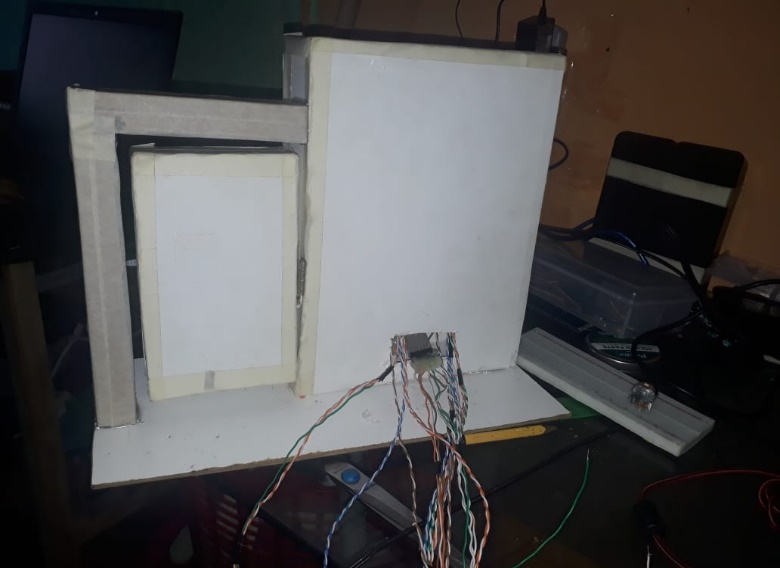
* Se desarrolló una Cerradura Digital para un prototipo similar a una caja fuerte, integrando nuevas tecnologías para controlar la apertura y el cerrado. Estas son de manera manual (teclado matricial) y remota (conectada a un microcontrolador).
* Con el desarrollo del presente proyecto se logró implementar en este todos los conocimientos adquiridos a lo largo del presente ciclo. Así como la inclusión de nuevos conocimientos.
* Durante la elaboración del proyecto se generaron dudas, las cuales se fueron resolviendo mediante consultas, y de esta manera lograr asimilar de mejor manera el conocimiento del tema.

1. **RECOMENDACIONES**

* Para realizar el código en AtmelStudio se necesita un previo estudio de sus comandos más comunes o investigación de los ya utilizados.
* Es necesario revisar las hojas de especificaciones y el diagrama de pines del atmega 2560 para evitar confusiones al momento de realizar las diferentes conexiones.

1. **ANEXOS**







# Bibliografía

|  |  |
| --- | --- |
| [1] | C. A. Barrionuevo Infante1, «Cerradura electrónica controlada por un prototipo de caja fuerte,» Pueblo Continente, 2019. |
| [2] | MicroChip. [En línea]. Available: https://www.microchip.com/mplab/avr-support/atmel-studio-7. |
| [3] | A. cl. [En línea]. Available: http://arduino.cl/arduino-mega-2560/. |
| [4] | . C. Atmel, «DATASHEET,» USA, 2014. |
| [5] | Comunidad, «Instructables,» [En línea]. Available: https://www.instructables.com/id/Keypad-Door-Lock/. [Último acceso: 10 03 2020]. |