EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Saltillo

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE SALTILLO

INGENIERÍA EN

SISTEMAS COMPUTACIONALES

SISTEMAS PROGRAMABLES

INTERFAZ DE MOTOR DE PASOS

PRESENTA

EIMY LU-RUHAMA CRUZ RODRÍGUEZ (19051114)

MARIO ORLANDO TORRES SALAZAR (19051220)

BRANDON ALEXIS PRADO CASTRO (19051178)

INSTITUTO TECNOLÓGICO DE SALTILLO

Contenido

Objetivo

Desarrollar una aplicación/programa de computadora en cualquier lenguaje de programación que permita comunicarse al puerto serial del Arduino y poder controlar por medio de un menú el motor de pasos, a su vez leer la información proveniente del puerto serial para poder graficar la posición del motor y todo este procedimiento guardarlo en un documento de texto o en una base de datos. Todo lo anterior para poder lograr el objetivo de crear una interfaz de usuario amigable y funcional que puede enviar y recibir datos para poder interpretarlos de forma visual para el usuario final.

Marco teórico

- Arduino es una plataforma de creación de electrónica de código abierto, la cual está basada en hardware y software libre, flexible y fácil de utilizar para los creadores y desarrolladores. Esta plataforma permite crear diferentes tipos de microordenadores de una sola placa a los que la comunidad de creadores puede darles diferentes tipos de uso.
- Visual Basic es un lenguaje de programación orientado a objetos desarrollado por Microsoft. El uso de Visual Basic agiliza y simplifica la creación de aplicaciones .NET con seguridad de tipos. El lenguaje Visual Basic .NET es totalmente diferente a sus antecesores, permite crear aplicaciones de escritorio, Web y móviles. Brinda un completo número de características para hacer que el desarrollo de aplicaciones sea realmente rápido.
- Oracle Database es un sistema de gestión de base de datos de tipo objeto-relacional, desarrollado por Oracle Corporation, la empresa estadounidense de hardware y software. Este tipo de sistema mejora la gestión de grandes bases de datos y también aumenta el nivel de seguridad.
- Un programa de computadora, aplicación o software, son un conjunto de instrucciones en forma secuencial, llamado código, que, a través de su interpretación por el sistema operativo o hardware, le permiten desarrollar una acción específica a una computadora portátil o de mesa.
- La interfaz de usuario es el medio con que el usuario puede comunicarse con una máquina, equipo, computadora o dispositivo, y comprende todos los puntos de contacto entre el usuario y el equipo.
- Una gráfica, una representación gráfica o un gráfico es un tipo de representación de datos, generalmente cuantitativos, mediante recursos visuales, para que se manifieste visualmente la relación matemática o correlación estadística que guardan entre sí.

Desarrollo de la práctica

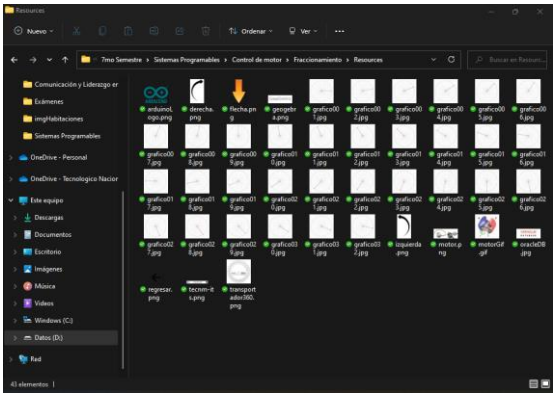
Para el desarrollo de la presente práctica, se tomo como base los trabajos anteriores (Motor de pasos controlado por Bluetooth y Motor de pasos simple), el único cambio que recibió el código de Arduino fue que conforme se va ejecutando el código, este mismo envía la información de la variable **i** del ciclo **for** hacia el puerto serial para que la aplicación lo pueda leer interpretar, como podemos ver a continuación en el circulo rojo:

```
void PasoSimple() {
  for (i = 0; i < pasos; i++) {
    // Paso 1
    digitalWrite(INA, HIGH);
    digitalWrite(INB, LOW);
    digitalWrite(INC, LOW);
    digitalWrite(IND, LOW);
    delay(tiempo_espera);

    // Paso 2
    digitalWrite(INA, LOW);
    digitalWrite(INB, HIGH);
    digitalWrite(INC, LOW);
    digitalWrite(IND, LOW);
    delay(tiempo_espera);
```

```
    // Paso 3
    digitalWrite(INA, LOW);
    digitalWrite(INB, LOW);
    digitalWrite(INC, HIGH);
    digitalWrite(IND, LOW);
    delay(tiempo_espera);

    // Paso 4
    digitalWrite(INA, LOW);
    digitalWrite(INB, LOW);
    digitalWrite(INC, LOW);
    digitalWrite(IND, HIGH);
    delay(tiempo_espera);

    Serial.println(i);
  }
}
```

Para el desarrollo de la aplicación o programa de computadora, se utilizo el IDE de Visual Studio 2019 con la versión de .NET 4.8.1 y la Base de datos Oracle 11g XE, con las librerías DLL de Oracle Data base en su versión 4.xx para guardar la información de ejecución.

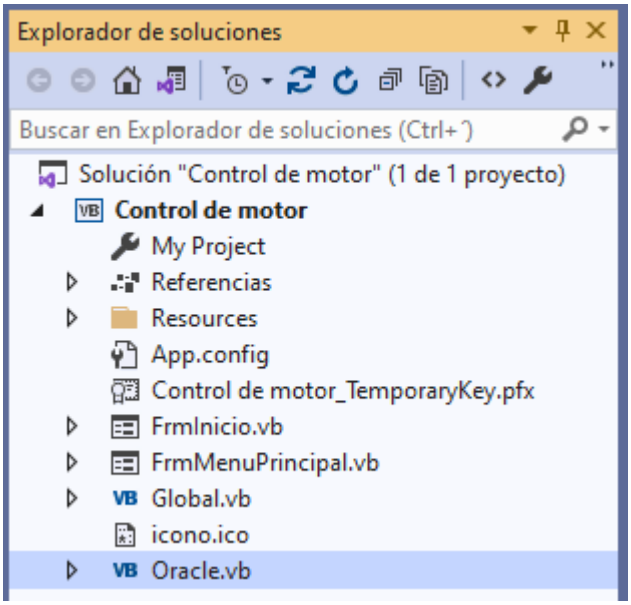
El grafico se elaboró en Geogebra y se exporto en 32 imágenes.



La base de datos consta de la siguiente tabla y vista, además de sus atributos:

```
Run SQL Command Line

Copyright (c) 1982, 2014, Oracle.  All rights reserved.

SQL> CONNECT motor/motor
Connected.
SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
EJECUCION                      TABLE
VISTA_TODAS_EJECUCIONES        VIEW

SQL> DESCRIBE EJECUCION;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID_EJECUCION                              NOT NULL NUMBER(10)
 PASO_REALIZADO                            NOT NULL VARCHAR2(30)
 FECHA_HORA_INICIO                         NOT NULL DATE
 FECHA_HORA_TERMINO                                 DATE

SQL> DESCRIBE VISTA_TODAS_EJECUCIONES;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                        NOT NULL NUMBER(10)
 PASO REALIZADO                            NOT NULL VARCHAR2(30)
 FECHA/HORA EJECUCIËN                               VARCHAR2(27)
 FECHA/HORA T╔RMINO                                VARCHAR2(27)
 SEGUNDOS DE EJECUCIËN                              NUMBER

SQL>
```

Estructura del proyecto en Visual Studio:



El proyecto consta de aproximadamente más de 1000 líneas de código.

La conexión del Arduino con el motor y la computadora es idéntica a la primera práctica realizada con el motor, solo con una versión modificada del código:

```
//  8*64 = 512 pasos
int pasos = 512;
int i = 0;
//IN1
int INA = 8;
//IN2
int INB = 9;
//IN3
int INC = 10;
//IN4
int IND = 11;
//Tiempo que pasa para la activación de la bobina
int tiempo_espera = 11;
//Tiempo que pasa para cada secuencia
int tiempo = 500;
//Variable con la funcion a desarrollar en el motor
char funcion = ' ';

void setup() {
  //Velocidad del sereal
  Serial.begin(9600);
  pinMode(INA, OUTPUT);  //pines como salida
  pinMode(INB, OUTPUT);
  pinMode(INC, OUTPUT);
  pinMode(IND, OUTPUT);
  delay(tiempo);
}

void loop() {
  if (Serial.available()) {//>0
    funcion = Serial.read();
  }

  if (funcion == '1') {
    PasoSimple();
    funcion = ' ';
  }

  if (funcion == '2') {
    PasoSimpleInverso();
    funcion = ' ';
  }

  if (funcion == '3') {
    PasoDoble();
    funcion = ' ';
  }

  if (funcion == '4') {
    PasoDobleInverso();
    funcion = ' ';
  }

  if (funcion == '5') {
    MedioPaso();
    funcion = ' ';
  }

  if (funcion == '6') {
    MedioPasoInverso();
    funcion = ' ';
  }

  if (funcion == '7') {
    Apagar();
    funcion = ' ';
  }
}

void PasoSimple() {
  for (i = 0; i < pasos; i++) {
    // Paso 1
    digitalWrite(INA, HIGH);
    digitalWrite(INB, LOW);
    digitalWrite(INC, LOW);
    digitalWrite(IND, LOW);
    delay(tiempo_espera);

    // Paso 2
    digitalWrite(INA, LOW);
    digitalWrite(INB, HIGH);
    digitalWrite(INC, LOW);
    digitalWrite(IND, LOW);
    delay(tiempo_espera);

    // Paso 3
    digitalWrite(INA, LOW);
    digitalWrite(INB, LOW);
    digitalWrite(INC, HIGH);
    digitalWrite(IND, LOW);
    delay(tiempo_espera);

    // Paso 4
    digitalWrite(INA, LOW);
    digitalWrite(INB, LOW);
    digitalWrite(INC, LOW);
    digitalWrite(IND, HIGH);
    delay(tiempo_espera);
    Serial.println(i);
  }
}

void PasoSimpleInverso() {
  for (i = 0; i < pasos; i++) {
    // Paso 1
    digitalWrite(INA, LOW);
    digitalWrite(INB, LOW);
    digitalWrite(INC, LOW);
    digitalWrite(IND, HIGH);
    delay(tiempo_espera);
```

```
  // Paso 2
  digitalWrite(INA, LOW);
  digitalWrite(INB, LOW);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 3
  digitalWrite(INA, LOW);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 4
  digitalWrite(INA, HIGH);
  digitalWrite(INB, LOW);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  Serial.println(i);
 }
}

void PasoDoble() {
 for (i = 0; i < pasos; i++) {
  // Paso 1
  digitalWrite(INA, HIGH);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 2
  digitalWrite(INA, LOW);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 3
  digitalWrite(INA, LOW);
  digitalWrite(INB, LOW);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);

  // Paso 4
  digitalWrite(INA, HIGH);
  digitalWrite(INB, LOW);
  digitalWrite(INC, LOW);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);

  Serial.println(i);
 }
}

void PasoDobleInverso() {
 for (i = 0; i < pasos; i++) {
  // Paso 1
  digitalWrite(INA, HIGH);
  digitalWrite(INB, LOW);
  digitalWrite(INC, LOW);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);

  // Paso 2
  digitalWrite(INA, LOW);
  digitalWrite(INB, LOW);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);

  // Paso 3
  digitalWrite(INA, LOW);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 4
  digitalWrite(INA, HIGH);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  Serial.println(i);
 }
}

void MedioPaso() {
 for (i = 0; i < pasos; i++) {
  // Paso 1
  digitalWrite(INA, HIGH);
  digitalWrite(INB, LOW);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 2
  digitalWrite(INA, HIGH);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

   // Paso 3
  digitalWrite(INA, LOW);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);
```

```
  // Paso 4
  digitalWrite(INA, LOW);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 5
  digitalWrite(INA, LOW);
  digitalWrite(INB, LOW);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 6
  digitalWrite(INA, LOW);
  digitalWrite(INB, LOW);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);

  // Paso 7
  digitalWrite(INA, LOW);
  digitalWrite(INB, LOW);
  digitalWrite(INC, LOW);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);

  // Paso 8
  digitalWrite(INA, HIGH);
  digitalWrite(INB, LOW);
  digitalWrite(INC, LOW);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);
  Serial.println(i);
 }
}
void MedioPasoInverso() {
 for (i = 0; i < pasos; i++) {
  // Paso 1
  digitalWrite(INA, HIGH);
  digitalWrite(INB, LOW);
  digitalWrite(INC, LOW);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);

  // Paso 2
  digitalWrite(INA, LOW);
  digitalWrite(INB, LOW);
  digitalWrite(INC, LOW);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);

  // Paso 3
  digitalWrite(INA, LOW);
  digitalWrite(INB, LOW);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, HIGH);
  delay(tiempo_espera);
```

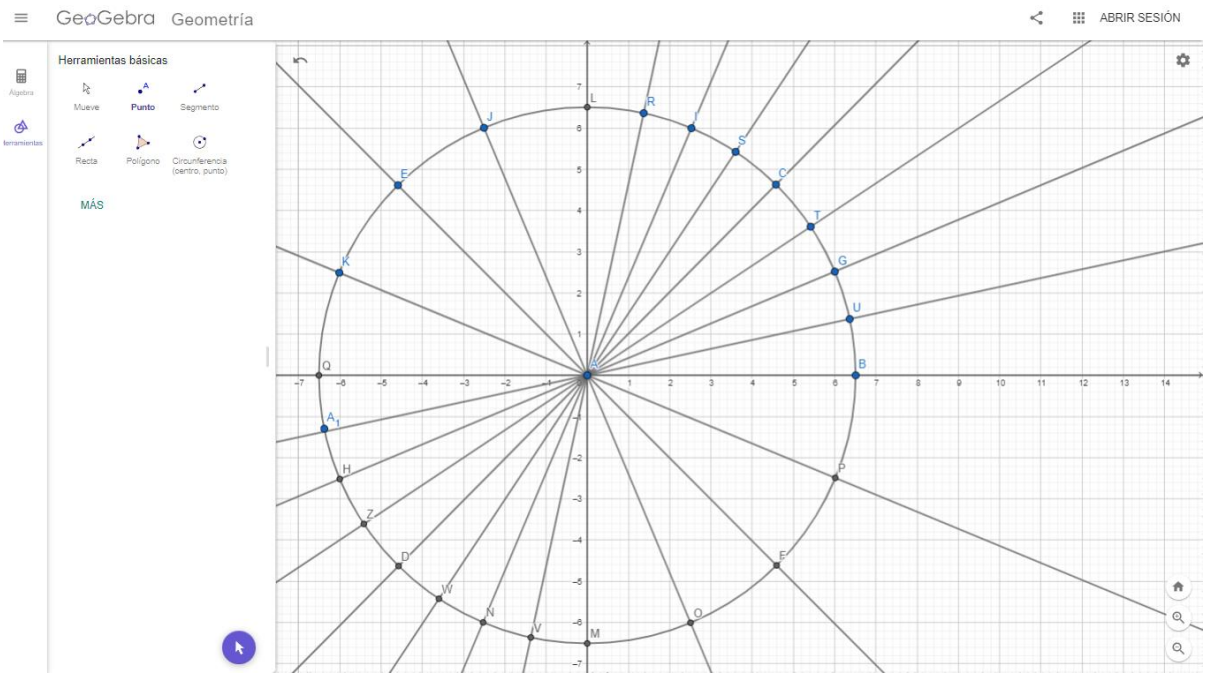
```
  // Paso 4
  digitalWrite(INA, LOW);
  digitalWrite(INB, LOW);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // paso 5
  digitalWrite(INA, LOW);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, HIGH);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 6
  digitalWrite(INA, LOW);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 7
  digitalWrite(INA, HIGH);
  digitalWrite(INB, HIGH);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  // Paso 8
  digitalWrite(INA, HIGH);
  digitalWrite(INB, LOW);
  digitalWrite(INC, LOW);
  digitalWrite(IND, LOW);
  delay(tiempo_espera);

  Serial.println(i);
 }
}

void Apagar() {
 digitalWrite(INA, LOW);
 digitalWrite(INB, LOW);
 digitalWrite(INC, LOW);
 digitalWrite(IND, LOW);
}
```

Script para la creación de la base de datos es la siguiente, contiene creación de tabla, creación de trigger, procedimientos almacenados para insertar, eliminar y actualizar, y una vista:

```
CONNECT system/admin
CREATE USER motor IDENTIFIED BY motor;
GRANT connect, resource TO motor;
GRANT create session, create table, create view, create trigger, create procedure TO motor;
CONNECT motor/motor

CREATE TABLE Ejecucion (
  id_ejecucion NUMERIC(10) NOT NULL,
  paso_realizado VARCHAR(30) NOT NULL,
  fecha_hora_inicio DATE NOT NULL,
  fecha_hora_termino DATE NULL,
  CONSTRAINT pk_Ej_idEjecucion PRIMARY KEY (id_ejecucion),
  CONSTRAINT ck_Ej_fechaHoraIT CHECK(fecha_hora_termino > fecha_hora_inicio)
);

--SECUENCIA PARA GENERAR ID EN LA TABLA Ejecucion
CREATE SEQUENCE SecuenciaEjecucion_ID
START WITH 1
INCREMENT BY 1
ORDER;

-- CREACION DE TRIGGER DE TABLA DE Ejecucion
CREATE OR REPLACE TRIGGER Ejecucion_Autoincrementa
BEFORE INSERT ON Ejecucion
FOR EACH ROW
BEGIN
  SELECT SecuenciaEjecucion_ID.NEXTVAL INTO :NEW.id_ejecucion FROM DUAL;
END;
/

--PROCEDIMIENTOS ALMACENADOS DE ENTRADA

------INSERT-----
--INSERT TABLA Ejecucion
CREATE OR REPLACE PROCEDURE Ejecucion_Insertar (paso_realizadoD IN VARCHAR, fecha_hora_inicioD IN DATE)
            AS
            BEGIN
                        INSERT INTO Ejecucion (paso_realizado, fecha_hora_inicio)
                        VALUES (paso_realizadoD, fecha_hora_inicioD);
            END;
/

------UPDATE-----
--UPDATE TABLA Ejecucion
CREATE OR REPLACE PROCEDURE Ejecucion_Actualizar (id_ejecucionD IN NUMERIC, fecha_hora_terminoD IN DATE)
            AS
            BEGIN
                        UPDATE Ejecucion SET fecha_hora_termino = fecha_hora_terminoD
                        WHERE id_ejecucion = id_ejecucionD;
            END;
/

------DELETE-----
--DELETE TABLA Ejecucion
CREATE OR REPLACE PROCEDURE Ejecucion_Eliminar (id_ejecucionD IN NUMERIC)
            AS
            BEGIN
                        DELETE Ejecucion WHERE id_ejecucion = id_ejecucionD;
            END;
/

--VISTAS
```

```
--VISTA DE Ejecucion
CREATE OR REPLACE VIEW Vista_Todas_Ejecuciones AS
SELECT id_ejecucion As ID, paso_realizado As "PASO REALIZADO", TO_CHAR(fecha_hora_inicio, 'DD/MON/YY HH24:MI:SS') As
"FECHA/HORA EJECUCIÓN",
TO_CHAR(fecha_hora_termino, 'DD/MON/YY HH24:MI:SS') As "FECHA/HORA TÉRMINO",
TRUNC((fecha_hora_termino - fecha_hora_inicio) * (60 * 60 * 24)) AS "SEGUNDOS DE EJECUCIÓN"
FROM Ejecucion ORDER BY fecha_hora_inicio DESC;

COMMIT;
```

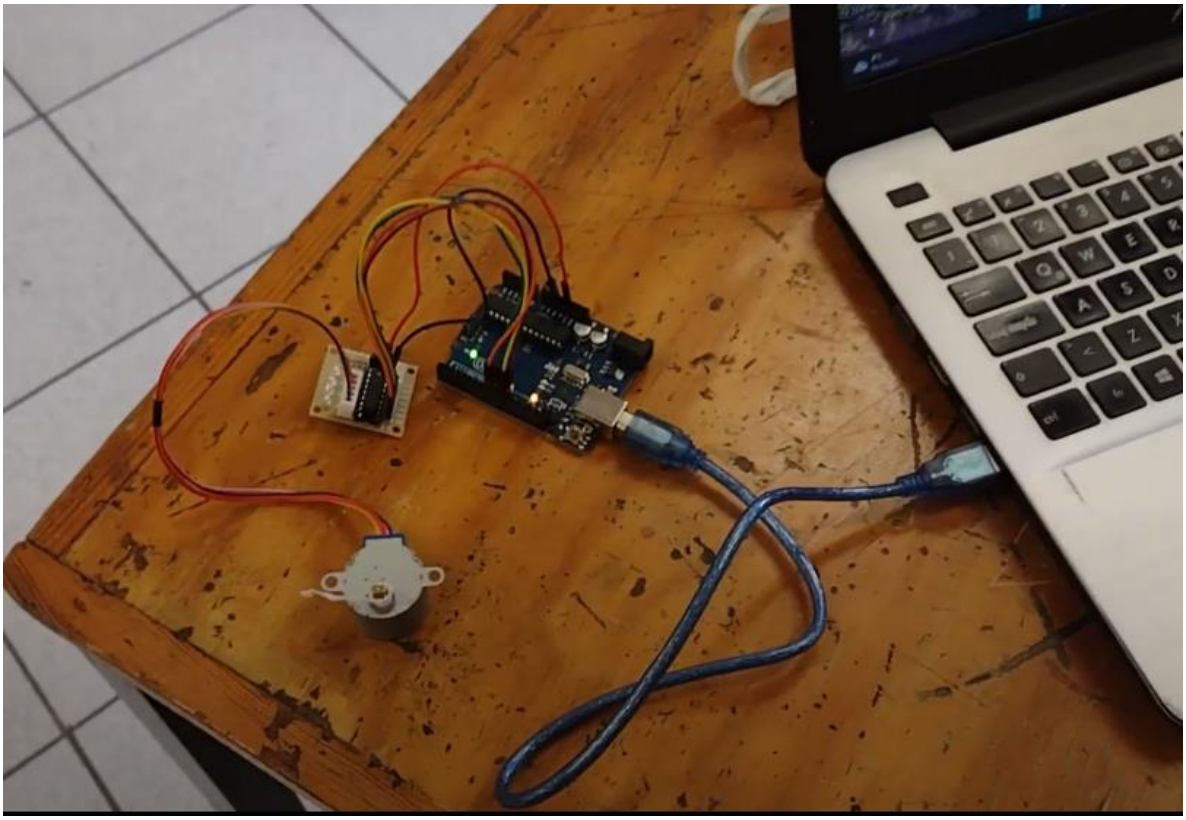
Creación del grafico en Geogebra

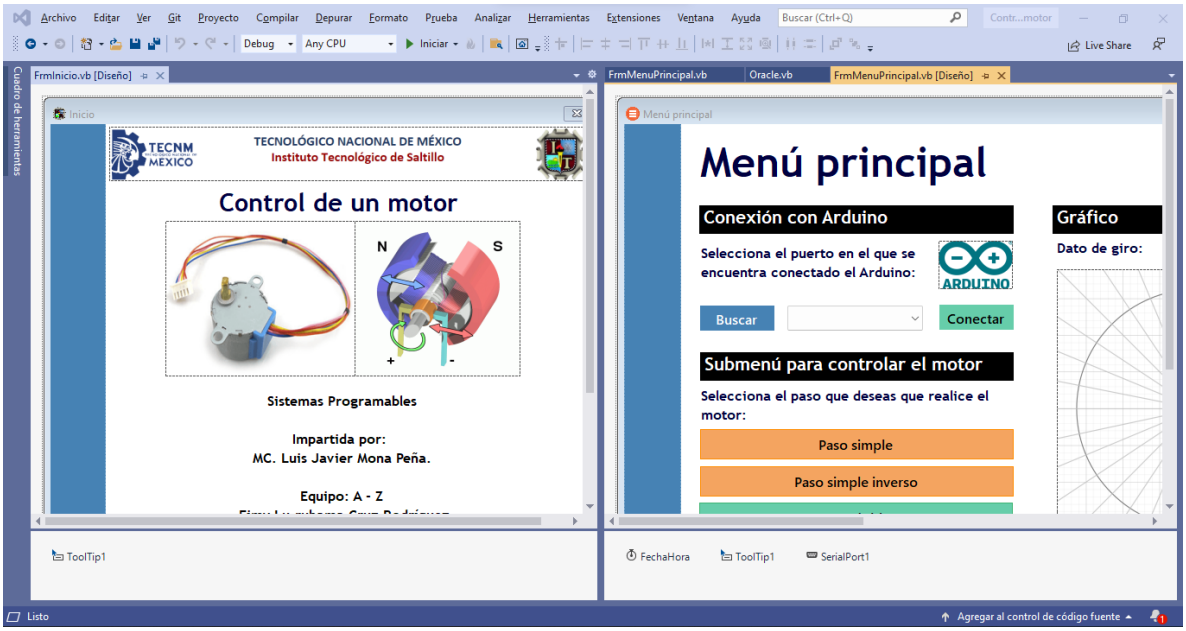


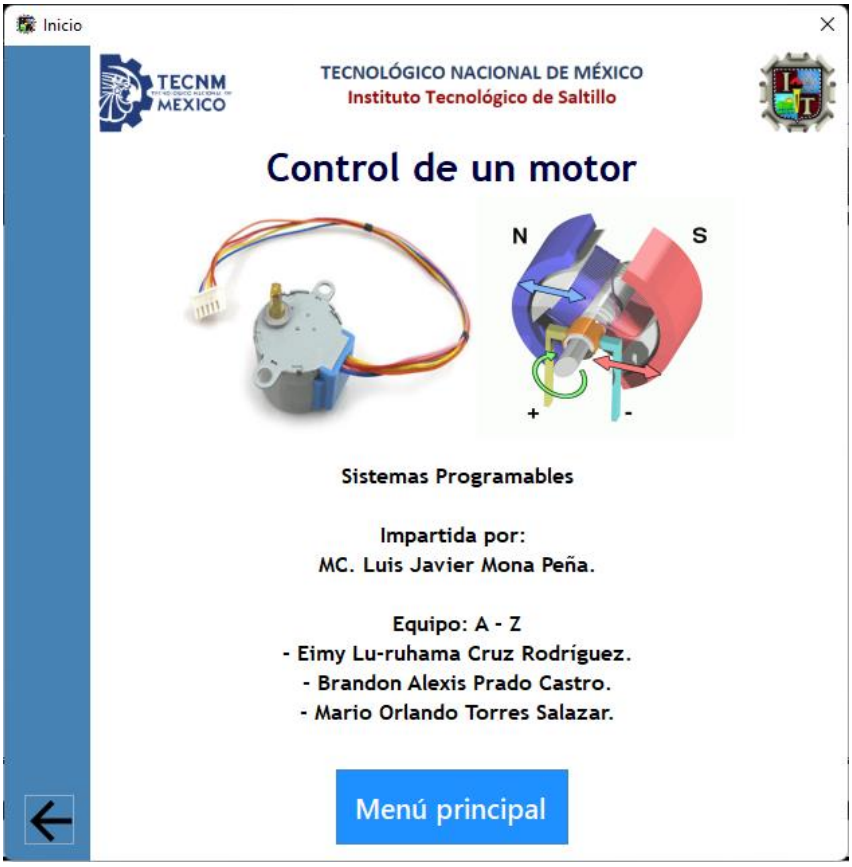
El material utilizado para esta práctica es el siguiente:

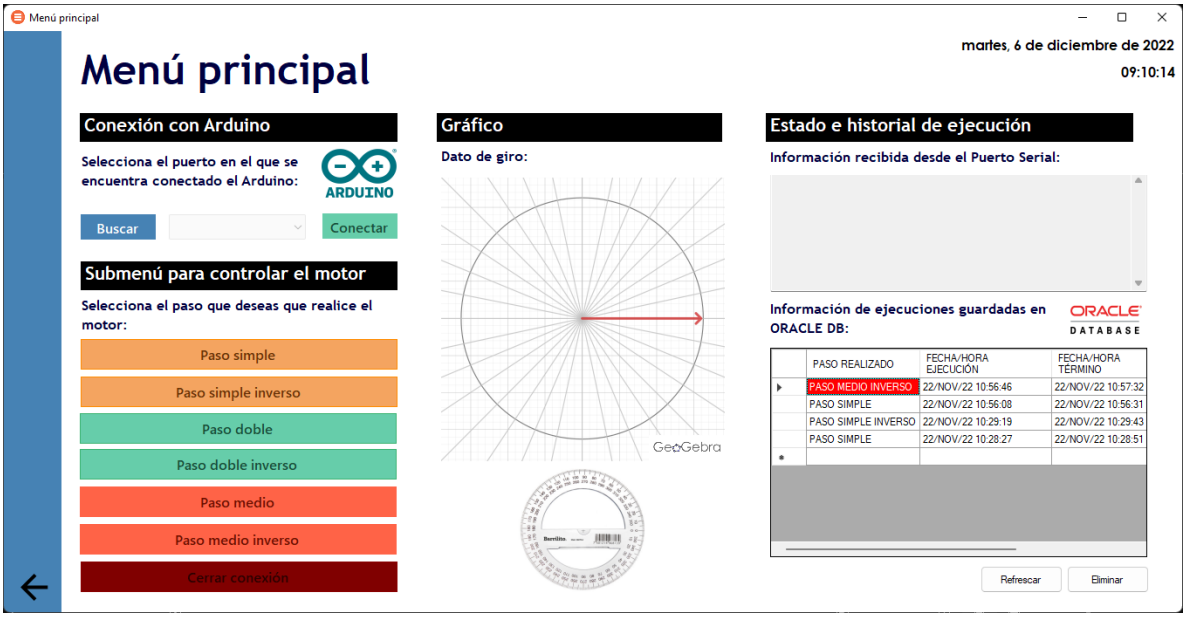
- Arduino
- Motor
- Cable de conexión para Arduino – Computadora
- Cables para hacer los puentes de conexión
- Computadora
- IDE Visual Studio con el lenguaje Visual Basic
- Aplicación de escritorio para controlar el motor
- Herramientas de Geogebra
- BD de Oracle 11g XE

Resultados




Archivo Editar Ver Git Proyecto Compilar Depurar Formato Prueba Analizar Herramientas Extensiones Ventana Ayuda
Debug
Any CPU
Iniciar
Live Share
FrmInicio.vb [Diseño]
FrmMenuPrincipal.vb
Oracle.vb
FrmMenuPrincipal.vb [Diseño]
Cuadro de herramientas
Inicio
TECNM MÉXICO
TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Saltillo
Control de un motor
N
S
+
-
Sistemas Programables
Impartida por:
MC. Luis Javier Mona Peña.
Equipo: A - Z
ToolTip1
Menú principal
Menú principal
Conexión con Arduino
Selecciona el puerto en el que se encuentra conectado el Arduino:
ARDUINO
Buscar
Conectar
Submenú para controlar el motor
Selecciona el paso que deseas que realice el motor:
Paso simple
Paso simple inverso
Gráfico
Dato de giro:
FechaHora
ToolTip1
SerialPort1
Listo
Agregar al control de código fuente





Evidencia en video

https://tecnmsaltillo-my.sharepoint.com/:v:/g/personal/19051178_saltillo_tecnm_mx/EUr4ku6swT1FpBoOh4TDlZcBXJvbI36_ARtWpqVwsX9hhA?e=ywcbg2

Materia adicional

https://tecnmsaltillo-my.sharepoint.com/:f:/g/personal/19051178_saltillo_tecnm_mx/EhqNSNC2CqxNksX9H2kvc1wBkElVMKtCmoeQzQazGmJtgg?e=YWNJf6

Conclusiones

En base a la presente práctica se puede concluir que la elaboración de una aplicación o programa de escritorio abre grandes posibilidades para el manejo de componentes tanto digitales como análogos en sectores de la industria mecánica, mecatrónica, ensambladora y todas aquellas en las cuales se utilizan componentes para realizar tareas, en esta ocasión se pudo ver de forma gráfica el comportamiento de un motor, de acuerdo a los diferentes pasos que se ejecutaban desde el programa desarrollado en Visual Basic, también resulta interesante el uso de Base de Datos para guardar información, en este caso de ejecución de los procesos respectivos para que en un momento determinado se puedan analizar los datos y posiblemente recomendar soluciones a deficiencias detectadas.

Bibliografía

(s.f.). Obtenido de https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno

(s.f.). Obtenido de https://www.oracle.com/database/

(s.f.). Obtenido de https://www.euroinnova.mx/blog/que-es-un-programa-de-computadora

(s.f.). Obtenido de https://rockcontent.com/es/blog/interfaz-de-usuario/

(s.f.). Obtenido de https://pro.arcgis.com/es/pro-app/latest/help/analysis/geoprocessing/charts/what-is-a-chart-.htm

(s.f.). Obtenido de https://hetpro-store.com/TUTORIALES/instalacion-visual-studio-con-arduino/