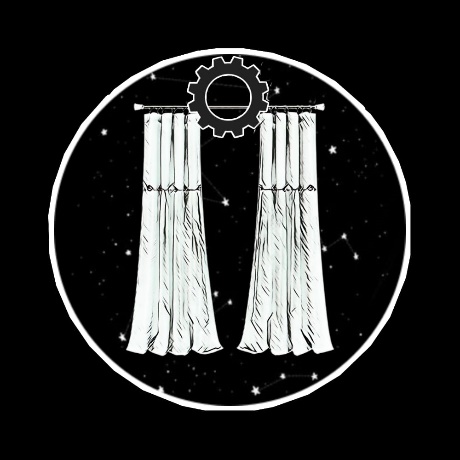


Manual Técnico



CORTINEX

Indice

1)Objetivos………………………………………………Pag 3

1.1) Objetivos específicos..Pag 3

2)Alcance…………………………………………………Pag 3

3) Requerimientos técnicos…………………….Pag 3

4)Herramientas utilizadas……………………….Pag 4

5)Instalacion y funcionamiento………….……Pag 4

6) Analisis…………………………………………………Pag 4

7) Desarrollo……………………………………………Pag 5

8)Diagrama del sistema…………………………Pag 10

1: OBJETIVOS

El propósito de este Manual es mostrar el proceso de diseño del sistema, y su correcto uso, para que este tenga el mantenimiento adecuado en caso de un fallo.

A grandes rasgos se diseñó y creo este sistema con el mero propósito de guiar al usuario cuando se encuentre con el mismo, y de informarle como se hizo, su proceso de instalación, etc.

1.1: OBJETIVOS ESPECIFICOS

* Desarrollo del sistema
* Requisitos para su funcionamiento

2: ALCANCE

Este manual es dirigido hacia: Técnicos

Conocimiento básico en Hardware

3: REQUERIMIENTOS TECNICOS

* Trozo de Carton
* Tela
* Cordel
* 2 Clips
* Pistola de silicona caliente
* Motor Dc 3v A 6v Caja reductora Auto inteligente Arduino
* CI MM53200M
* Controlador BA659AN
* 3 palos de brocheta
* Pegamento instantáneo
* Alambres delgados para hacer el conmutador
* Un poco de cables
* Enchufe (un cargador) con cables positivos y negativos

4: HERRAMIENTAS UTILIZADAS

* Regla
* Soldador
* Lapiz
* Compas
* Punzon
* Alicate
* Tijera
* Perforador

5: INSTALACION Y FUNCIONAMIENTO

Para la instalación del sistema solo basta con colocar el producto en la superficie que desee, si desea puede utilizar cinta doble faz sobre las esquinas para sellarlo contra la pared.

Una vez completado este procedimiento, tome el control remoto y mueva la palanca de derecha a izquierda para abrir y cerrar las cortinas, controlando el movimiento.

6: ANALISIS

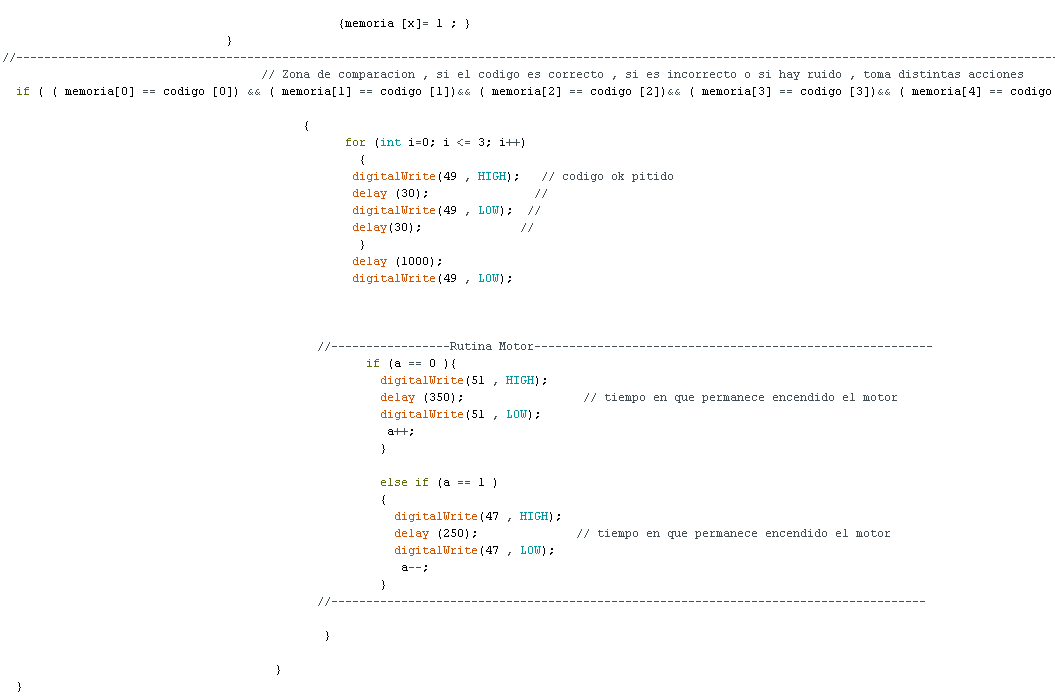
El sistema ha sido sometido a análisis, diseño, desarrollo y diferentes tipos de pruebas, y al mismo tiempo da soporte a las fallas que puedan existir en el mismo, de manera de brindar la mejor calidad y desempeño a todo aquel que quiera acceder a él.

CORTINEX surge a partir de una manera de ofrecer mayor comodidad, ofreciendo un producto económico y eficaz.

7: DESARROLLO

A continuación, se proporcionará el código de Arduino del sistema:

******

******

/\*

Decodificador IC KM 53200

R de IM, C 10 pf Oscilador

\*/

Int código ( ) = (0,0,0,0,0,1,0,1,1,1,0,0) ; // Ingreso del código emparejado con el IC KM 53200

// VARIABLES PARA CALCULOS

Unsigned int r, tiempo, a;

Int memoria (100);

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

Void setup() {

pinMode (47, OUTPUT); // Motor, giro directo

pinMode (51, OUTPUT); //Motor, giro inverso

pinMode (31, INPUT); // Entrada RX

pinMode (49, OUTPUT);//buffer

a = 0 ;

}

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

Void loop() {

digitalWrite(49, LOW) ;

digitalWrite(51, LOW) ;

tiempo= 0;

tiempo = pulseIn(31 , LOW , 9100) ; //Si el pulso del pin 31 es > que 7 segundos continua /if pulse > 7 miliseconds, continues

if ( tiempo > 7000) // detecta el pulso de descanso entre rata de datos, el pulso esta bajo aprox 8 ms

{

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

//lectura de los anchos de pulso, luego del pulso prolongado de aprox 8 ms procede a leer 12 pulsos en alto con sus respectivos anchos

//en microsegundos

For (int x=0;x<12;x++)

{ memoria(x) = pulseIn (31, HIGH); )

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

//revisa si en alguna de las posiciones del bco de memoria el ancho de pulso pertenece a un ruido de lo contrario convierte los anchos largo en 0 y corto en 1

for (int x-0;x<12;x++)

{

if (memoria (x) >350)

(memoria (x) = 0 ; }

Else

(memoria (x) = 1; )

}

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

//Zona de comparación, si el código es correcto, si es incorrecto o si hay ruido, toma distinas acciones

If ( (memoria (0) == código (0)) ¨ (memoria(1) == codigo (1)) ¨ (memoria (2) == codigo (2)) ¨ (memoria (3) == codigo (3)) ¨ (memoria (4) codigo (4))

{

For (int 1=0 ; 1 <= 3; 1++)

{

digitalWrite (49 , HIGH); // codigo ok pitido

delay (30); //

digitalWrite (49, LOW) ; //

delay (30) ; //

}

Delay (1000) ;

digitalWrite (49, LOW);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Rutina Motor\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

If (a == 0) {

digitalWrite (51 , HIGH) ;

delay (350) ; // tiempo en que permanece encendido el motor

digitalWrite (51, LOW) ;

a++;

}

Else if (a == 1)

{

digitalWrite (47, HIGH );

delay (250) ; // tiempo en que permanece encendido el motor

digitalWrite (47, LOW);

a--;

}

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

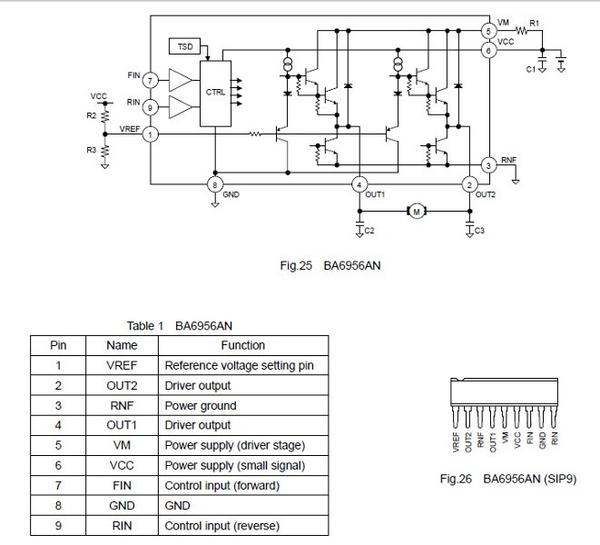
}

}

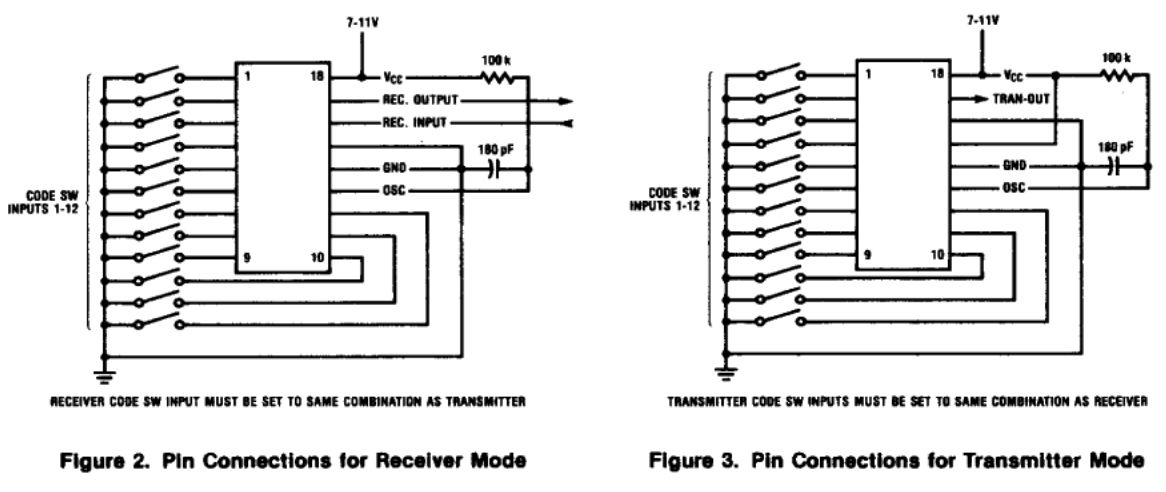
}

9: DIAGRAMA DEL SISTEMA

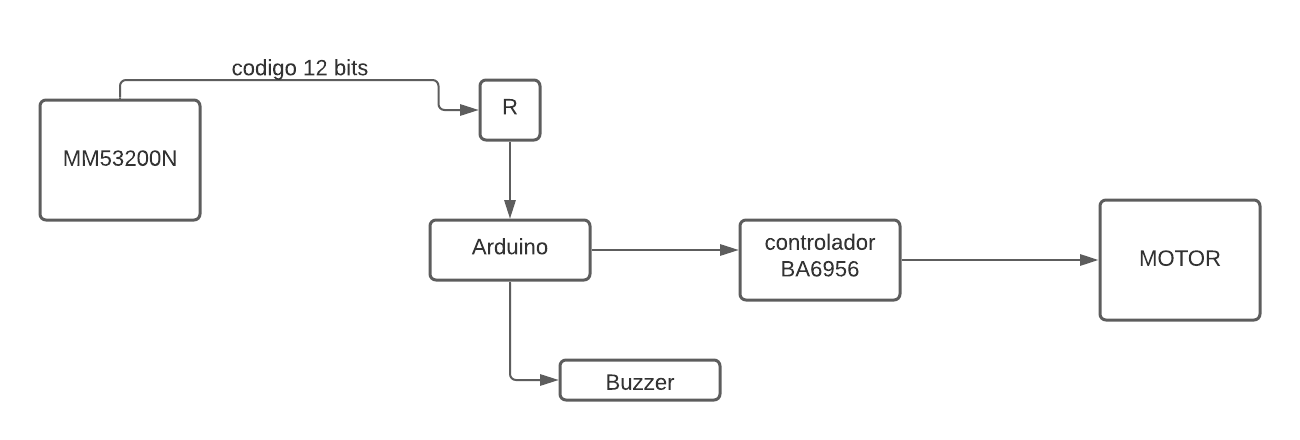
BA695AN



MM53200N



FUNCIONAMIENTO BASICO:



El control con un integrado MM53200N, a través de un transmisor enviara un código de 12 bits a un receptor con dos transitores que limpiara la señal y esa misma señal se enviara al Arduino y con una cierta programación el buzzer del receptor, con unos ciertos pitidos para saber si el código es correcto o incorrecto. Luego el controlador BA6956 controlara el funcionamiento del motor.