Tabla de contenido

[Proyecto Banco de pruebas EVR. 2](#_Toc70513705)

[Conexiones. 3](#_Toc70513706)

[Visualizar ejecución del programa en el PC. 3](#_Toc70513707)

[Errores detectables (Arduino). 5](#_Toc70513708)

[Anexos. 6](#_Toc70513709)

[Programa Arduino (simulador de pulsos). 6](#_Toc70513710)

[Programa Display PIC16F628A 12](#_Toc70513711)

[Etapa de control (Salidas del Arduino). 15](#_Toc70513712)

[Etapa de Potencia (Entradas del Arduino). 16](#_Toc70513713)

[Etapa de Comunicación. 17](#_Toc70513714)

[Control de velocidad 18](#_Toc70513715)

[Registrador de eventos 19](#_Toc70513716)

# Proyecto Banco de pruebas EVR.

El banco de pruebas para EVR (registrador de eventos o HASLER) está diseñado con el objetivo de simular un recorrido de viaje en tren para que el EVR registre datos de velocidad y realice la acción de activar o desactivar de manera virtual los relés de control de velocidad de las tarjetas x1, x6 y x7, enviando pulsos simulados a la velocidad creciente y decreciente de un tren a las tarjetas x1, x4 y x8. Detectara y dará información visual en caso de presentarse errores en caso de que los estados lógicos en las salidas del EVR (que representan las activaciones o desactivaciones de los relés de velocidad) no cambien cuando sea debido.

En primera instancia el programa comienza simulando estar en una estación detenido, realizando:

1. Apertura de Puertas (señal luminosa correspondiente APAGADA).
2. Espera de 4,5 segundos.
3. Luz indicadora cierre de puertas (señal luminosa correspondiente ENCENDIDA).
4. Espera de 2 segundos.
5. Sirena indicadora cierre de puertas (señal luminosa correspondiente ENCENDIDA).
6. Espera de 2.5 segundos
7. Luz indicadora cierre de puertas (señal luminosa correspondiente APAGADA).
8. Sirena indicadora cierre de puertas (señal luminosa correspondiente APAGADA).
9. Cierre de Puertas (señal luminosa correspondiente APAGADA).
10. Bloqueo de las puertas (señal luminosa correspondiente ENCENDIDA).
11. Espera de 1 segundo.

Nota: Cada vez que el ciclo se reinicia la variable correspondiente al ‘Bloqueo de las puertas’ cambia de estado a APAGADO. Esta función toma alrededor de 10 segundos aproximadamente.

El programa, continua con el movimiento del tren, este se divide en dos partes: **Aceleración y desaceleración.** El cambio entre ambas subfunciones ocurre al llegar a la velocidad máxima virtual establecida (100km/h aproximadamente) en estas funciones el Arduino está evaluando los estados de las salidas del EVR, que estos sean los que corresponde en las situaciones establecidas (cuando la velocidad sea mayor a 0.5km/h y luego mayor a 6km/h para la aceleración, y efecto contrario para la desaceleración).

Respecto a las entradas y salidas del sistema, las salidas del Arduino (que van desde los pines 2 hasta 7) correspondes a la etapa de control, donde se indica el error en caso de haber alguno, se envía la señal cuadrada que representa la velocidad de movimiento del tren e indicadores del proceso que se realiza al estar en estación. Se consideran como entradas los pines 8 hasta 13, siendo estos los receptores de pulsos provenientes de EVR para los relés de control de velocidad.

# Conexiones.

El sistema cuenta con dos entradas de tensión, una de 12v que se utiliza para alimentar el Arduino y los componentes que participan en la etapa de control. Mientras que la otra entrada es de 72v que se destina para alimentar el EVR y las tarjetas, como también los componentes que participan en la etapa de potencia. Desde el sistema salen 6 conectores que corresponden a cada una de las tarjetas del registrador de eventos (X1, X4, X5, X6, X7, X8) y que están confeccionados según el diagrama de conexión de estas tarjetas. Cada conector debe ir con su tarjeta correspondiente. Considerando que el sistema no tiene un botón de inicio, y apenas el sistema es alimentado con 12V comenzara a funcionar hasta que el sistema sea apagado.

Se debe considerar que el ciclo del programa es, en teoría, de ejecución infinita, y que será detenido si y solo si encuentra algún error, quedando estático en dicho punto del programa, y este solo podrá reiniciarse en ejecución presionando el botón Reset (que solo se usa cuando no se encuentra error y es solo para reiniciar el programa del Arduino) o apagando y encendiendo la etapa de control, que volverá tanto el Display como el Arduino al inicio de sus correspondientes programas. Esto se diseño de esta manera considerando casos en que la memoria del Arduino forzara un reinicio del programa, el Display siempre mostrara el error en el que quedo registrado el Arduino.

# Visualizar ejecución del programa en el PC.

Para visualizar la ejecución del programa y así ver el punto en el que se encuentra, es necesario tener instalado en el equipo el compilador de Arduino que lleva el mismo nombre.

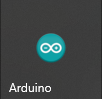
Una vez en el programa, sea un archivo nuevo o un trabajo en proceso, es conectar el Arduino mediante un cable USB correspondiente al modelo de la placa, y verificar en *‘Herramientas/Puerto/COM’X’’* (Modelo Arduino), donde X es el número del puerto *(Ilustración 2)*. Encaso que no este seleccionado, realizar dicha acción.

Ilustración 1, Icono del   
programa compilador   
de Arduino.

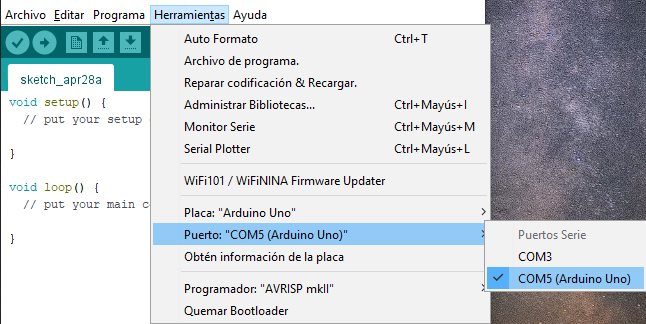


Ilustración 2

Una seleccionado, para poder visualizar la ejecución del programa es necesario seleccionar el ‘Monitor Serie’ en ***‘Herramientas/Monitor Serie’*** o presionando las teclas ***‘Ctrl + Mayús + m’*** abriéndose la siguiente ventana donde se puede apreciar los mensajes y la ejecución del programa *(Ilustración 3)*.

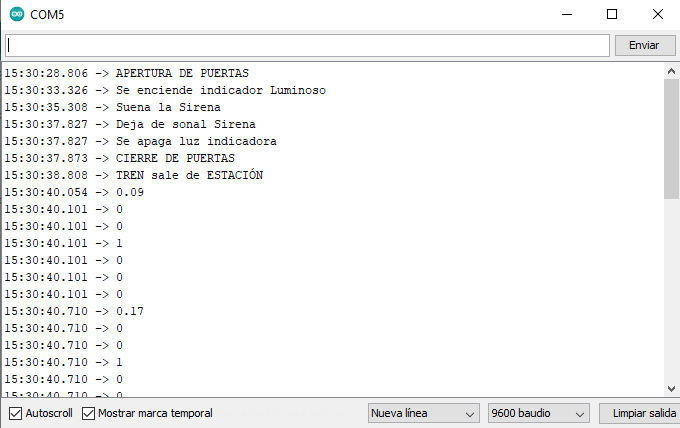


Ilustración 3, visualización de los valores establecidos para mostrarse en la ventana de monitor serial.

De esta manera es posible visualizar la ejecución de programa directamente en el pc con sus correspondientes marcas temporales (en caso de que este seleccionada la opción en la ventana).

En un buen funcionamiento, la ejecución de estas funciones es continuo y se repetirá hasta que el Arduino detecte alguna anomalía en las salidas del EVR, que podrán ser visualizadas directamente de 3 formas:

1. Led de error correspondiente al pin digital 2 del Arduino, las que tendrán ciclos de intermitencia, separados por un segundo
2. Display, que mostrara el error y la tarjeta afectada, según indique el Arduino.
3. Led que indican el estado en el que se encuentran los relés de velocidad.

A continuación, se mostrarán los errores que detectara el Arduino:

# Errores detectables (Arduino).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Error | Señal Luminosa (cantidad de veces que cambia de estado, con espera de 1 segundo) | Salida Arduino - Entrada PIC (A0, A1, A2, A3). | Tarjeta Afectada. | Mensaje de error | Momento |
| 001 | 1 | 1000 | **X1** | ERROR\_011: la variable X1\_VT\_05 | En Aceleración |
| 1110 | ERROR\_013: la variable X1\_VT\_05 | En Desaceleración |
| 002 | 2 | 0010 | **X1** | ERROR\_012: la variable X1\_VT\_6 | En Aceleración |
| 0101 | ERROR\_014: la variable X1\_VT\_6 | En Desaceleración |
| 003 | 3 | 0100 | **X7** | ERROR\_021: la variable X7\_VT\_05 | En Aceleración |
| 0001 | ERROR\_023: la variable X7\_VT\_05 | En Desaceleración |
| 004 | 4 | 1010 | **X7** | ERROR\_022: la variable X7\_VT\_6 | En Aceleración |
| 1101 | ERROR\_024: la variable X7\_VT\_6 | En Desaceleración |
| 005 | 5 | 1100 | **X6** | ERROR\_031: la variable X6\_VT\_05 | En Aceleración |
| 1001 | ERROR\_033: la variable X6\_VT\_05 | En Desaceleración |
| 006 | 6 | 0110 | **X6** | ERROR\_032: la variable X6\_VT\_6 | En Aceleración |
| 0011 | ERROR\_034: la variable X6\_VT\_6 | En Desaceleración |
| 007 | 7 | Sin Definir | **Sin Definir** | Sin Definir | Sin Definir |

# Anexos.

## Programa Arduino (simulador de pulsos).

//---DEFINICION\_PINES--------------------------------------------------------------------  
int error = 2;//variable destinada a dar indicación de ERROR a causa de alguna condición incumplida  
int pulseOut = 3;//salida pulsos que simulan velocidad, se selecciona el pin11 por sus cualidades de salida PWM  
int cc = 4;//Confirmación Cierre de Puertas.  
int luz = 5;//indicador que representa LUZ indicadora del TREN  
int sirena = 6;//indicador que representa la SIRENA o GRABACIÓN del tren  
int puerta = 7;//indicador que representa las PUERTAS del TREN  
int X1\_VT\_6 = 8;//lectura de variable provimiente de tarjeta X1 para rele control de velocidad 6  
int X7\_VT\_6 = 9;//lectura de variable provimiente de tarjeta X7 para rele control de velocidad 6  
int X6\_VT\_6 = 10;//lectura de variable provimiente de tarjeta X6 para rele control de velocidad 6  
int X1\_VT\_05 = 11;//lectura de variable provimiente de tarjeta X1 para rele control de velocidad 05  
int X7\_VT\_05 = 12;//lectura de variable provimiente de tarjeta X7 para rele control de velocidad 05  
int X6\_VT\_05 = 13;//lectura de variable provimiente de tarjeta X6 para rele control de velocidad 05  
//---DEFINICION\_VARIABLES----------------------------------------------------------------  
int countMax = 1160; //es 1.25 veces la frecuencia máxima  
float tiempo;//tiempo que estara en funcion de la frecuencia  
float i, j;//contadores  
float vel;//valor de la velocidad en funcion de la frecuencia  
const float vel\_1 = 0.5;//velocidad para apagar el rele 1  
const float vel\_2 = 6;//velocidad para apagar el rele 2  
const float var1 = 0.953;//diametro de la rueda en metros  
const float var2 = 3.6;//constante de velocidad del TREN  
const float var3 = 100;//ventanas de la rueda  
const float var4 = 1.25;//Cantidad de veces que se multiplico la frecuencia máxima

void setup()  
{  
 Serial.begin(9600);  
 pinMode(pulseOut, OUTPUT);  
 pinMode(puerta, OUTPUT);  
 pinMode(luz, OUTPUT);  
 pinMode(sirena, OUTPUT);  
 pinMode(X1\_VT\_6, INPUT);//se define como entrada para lectura de reaccion del REGISTRADOR DE EVENTOS  
 pinMode(X1\_VT\_05, INPUT);//se define como entrada para lectura de reaccion del REGISTRADOR DE EVENTOS  
 pinMode(X7\_VT\_6, INPUT);  
 pinMode(X7\_VT\_05, INPUT);  
 pinMode(X6\_VT\_6, INPUT);  
 pinMode(X6\_VT\_05, INPUT);  
 pinMode(error, OUTPUT);  
 pinMode(A0, OUTPUT);//Se usaran las entradas analogicas como salidas digitales  
 pinMode(A1, OUTPUT);  
 pinMode(A2, OUTPUT);  
 pinMode(A3,OUTPUT);  
}

void loop()  
{  
 digitalWrite(cc, LOW);  
 enEstacion();  
 Serial.println("TREN sale de ESTACIÓN");  
 acelerar();  
 Serial.println("VELOCIDAD MAXIMA");  
 desacelerar();  
 Serial.println("TREN ha llegado a DESTINO");  
}  
//------------FUNCIONES------------------------------------------------------------------  
//---SALIDA\_SEÑAL\_CUADRADA---------------------------------------------------------------

void senalOut()  
{//Funcion que se encarga de generar los estados de los semiciclos correspondientes.  
 tiempo = (1000 / (i / (var4)));//tiempo de duracion de cada semiCiclo en milisegundos  
 digitalWrite(pulseOut, HIGH);  
 delay(tiempo / 2);//semiciclo positivo  
 digitalWrite(pulseOut, LOW);  
 delay(tiempo / 2);//semiciclo negativo  
}  
//---ACELERACION\_DEL\_TREN----------------------------------------------------------------  
void acelerar()  
{  
 for (i = 1; i < countMax; i++)  
 {//se calculo que aproximadamente los 928hz se alcanzan los 100km/h  
 senalOut();  
 velMax();  
 lecturasEnSubida05();  
 lecturasEnSubida6();  
 }  
 }  
//---DESACELERACION\_DEL\_TREN-------------------------------------------------------------  
void desacelerar()

{  
 for (i = countMax; i >= 1; i--)  
 {  
 senalOut();  
 velMax();  
 lecturasEnBajada6();  
 lecturasEnBajada05();  
 }  
}  
//---VISUALIZAR\_VELOCIDAD----------------------------------------------------------------  
void velMax()  
{  
 vel = ((i / var4) \* PI \* var1 \* var2 / var3);  
 Serial.println(vel);  
}  
//---ACCIONES\_EN\_ESTACION----------------------------------------------------------------  
void enEstacion()  
{  
 digitalWrite(puerta, LOW);  
 Serial.println("APERTURA DE PUERTAS");  
 delay(4500);  
 digitalWrite(luz, HIGH);  
 Serial.println("Se enciende indicador Luminoso");  
 delay(2000);  
 digitalWrite(sirena, HIGH);  
 Serial.println("Suena la Sirena");  
 delay(2500);  
 digitalWrite(sirena, LOW);  
 Serial.println("Deja de sonal Sirena");  
 digitalWrite(luz, LOW);  
 Serial.println("Se apaga luz indicadora");  
 digitalWrite(puerta, HIGH);  
 Serial.println("CIERRE DE PUERTAS");  
 digitalWrite(cc, HIGH);  
 delay(1000);  
}  
//---FUNCIONES\_POR\_ERRORES---------------------------------------------------------------  
void error\_001()  
{ //error 1 cuando X1\_VT\_05 debiera ser HIGH  
Serial.println("");  
for (;;)  
 {  
 digitalWrite(error, HIGH);  
 delay(500);  
 digitalWrite(error, LOW);  
 delay(500);  
 }  
}  
void error\_002()  
{ //error 2 cuando X1\_VT\_6 deberia ser HIGH  
 for (;;)  
 {  
 for (j = 0; j < 2; j++)  
 {  
 digitalWrite(error, HIGH);  
 delay(500);  
 digitalWrite(error, LOW);  
 delay(500);  
 }  
 delay(1000);  
 }  
}

void error\_003()  
{ //error 3 cuando X7\_VT\_05 deberia ser HIGH  
 for (;;)  
 {  
 for (j = 0; j < 3; j++)  
 {  
 digitalWrite(error, HIGH);  
 delay(500);  
 digitalWrite(error, LOW);  
 delay(500);  
 }  
 delay(1000);  
 }  
}

void error\_004()  
{ //error 4 cuando X7\_VT\_6 deberia ser HIGH  
 for (;;)  
 {  
 for (j = 0; j < 4; j++)  
 {  
 digitalWrite(error, HIGH);  
 delay(500);  
 digitalWrite(error, LOW);  
 delay(500);  
 }  
 delay(1000);  
 }  
}

void error\_005()  
{ //error 5 cuando X6\_VT\_05 deberia ser HIGH  
 for (;;)  
 {  
 for (j = 0; j < 5; j++)  
 {  
 digitalWrite(error, HIGH);  
 delay(500);  
 digitalWrite(error, LOW);  
 delay(500);  
 }  
 delay(1000);  
 }  
}

void error\_006()  
{ //error 6 cuando X6\_VT\_6 deberia ser HIGH  
 for (;;)  
 {  
 for (j = 0; j < 6; j++)  
 {  
 digitalWrite(error, HIGH);  
 delay(500);  
 digitalWrite(error, LOW);  
 delay(500);  
 }  
 delay(1000);  
 }  
}

void error\_007()  
{ //error 7 Ninguna lectura cambia  
 for (;;)  
 {  
 for (j = 0; j = 7; j++)  
 {  
 digitalWrite(error, HIGH);  
 delay(500);  
 digitalWrite(error, LOW);  
 delay(500);  
 }  
 delay(1000);  
 }  
 }  
//---LECTURAS----------------------------------------------------------------------------  
void lecturasEnSubida05()  
{  
 Serial.println(digitalRead(X1\_VT\_05));  
 Serial.println(digitalRead(X7\_VT\_05));  
 Serial.println(digitalRead(X6\_VT\_05));  
 while (vel > vel\_1)  
 { //Cuando la velocidad calculada es MAYOR a 0.5KM/h y...  
 if (digitalRead(X1\_VT\_05) == HIGH)  
 { //Si la lectura es 0 digital, se ejecuta el error 1  
 digitalWrite(A0,HIGH);  
 digitalWrite(A1,LOW);  
 digitalWrite(A2,LOW);  
 digitalWrite(A3,LOW);  
 Serial.println("ERROR\_011: la variable X1\_VT\_05 debia arrojar un UNO, pero recibio un CERO\nERROR CONTROL DE VELCIDAD RELE 0.5KM/H");  
 error\_001();  
 break;  
 }  
 else if (digitalRead(X7\_VT\_05) == HIGH)  
 { //Si la lectura es 0 digital, se ejecuta el error 3  
 digitalWrite(A0,LOW);  
 digitalWrite(A1,HIGH);  
 digitalWrite(A2,LOW);  
 digitalWrite(A3,LOW);  
 Serial.println("ERROR\_021: la variable X7\_VT\_05 debia arrojar un UNO, pero recibio un CERO\nERROR CONTROL DE VELCIDAD RELE 0.5KM/H");  
 error\_003();  
 break;  
 }  
 else if (digitalRead(X6\_VT\_05) == HIGH)  
 { //Si la lectura es 0 digital, se ejecuta el error 5  
 digitalWrite(A0,HIGH);  
 digitalWrite(A1,HIGH);  
 digitalWrite(A2,LOW);  
 digitalWrite(A3,LOW);  
 Serial.println("ERROR\_031: la variable X6\_VT\_05 debia arrojar un UNO, pero recibio un CERO\nERROR CONTROL DE VELCIDAD RELE 0.5KM/H");  
 error\_005();  
 break;  
 }  
 else  
 {  
 break;  
 }  
 break;  
 }  
 }  
 void lecturasEnSubida6()  
 {  
 Serial.println(digitalRead(X1\_VT\_6));  
 Serial.println(digitalRead(X7\_VT\_6));  
 Serial.println(digitalRead(X6\_VT\_6));  
 while (vel > vel\_2)  
 { //Cuando la velocidad calculada es MENOR a 6km/h y...  
 if (digitalRead(X1\_VT\_6) == HIGH)  
 { //Si la lectura es 1 digital, se ejecuta el error 2  
 digitalWrite(A0,LOW);  
 digitalWrite(A1,LOW);  
 digitalWrite(A2,HIGH);  
 digitalWrite(A3,LOW);  
 Serial.println("ERROR\_012: la variable X1\_VT\_6 debia arrojar un UNO, pero recibio un CERO\nERROR CONTROL DE VELCIDAD RELE 6KM/H");  
 error\_002();  
 break;  
 }  
 else if (digitalRead(X7\_VT\_6) == HIGH)  
 { //Si la lectura es 1 digital, se ejecuta el error 4  
 digitalWrite(A0,HIGH);  
 digitalWrite(A1,LOW);  
 digitalWrite(A2,HIGH);  
 digitalWrite(A3,LOW);  
 Serial.println("ERROR\_022: la variable X7\_VT\_6 debia arrojar un UNO, pero recibio un CERO\nERROR CONTROL DE VELCIDAD RELE 6KM/H");  
 error\_004();  
 break;  
 }  
 else if (digitalRead(X6\_VT\_6) == HIGH)  
 { //Si la lectura es 1 digital, se ejecuta el error 6  
 digitalWrite(A0,LOW);  
 digitalWrite(A1,HIGH);  
 digitalWrite(A2,HIGH);  
 digitalWrite(A3,LOW);  
 Serial.println("ERROR\_032: la variable X6\_VT\_6 debia arrojar un UNO, pero recibio un CERO\nERROR CONTROL DE VELCIDAD RELE 6KM/H");  
 error\_006();  
 break;  
 }  
 else  
 {  
 break;  
 }  
 break;  
 }  
 }  
 void lecturasEnBajada05()  
 {  
 Serial.println(digitalRead(X1\_VT\_05));  
 Serial.println(digitalRead(X7\_VT\_05));  
 Serial.println(digitalRead(X6\_VT\_05));  
 while (vel <= vel\_1)  
 { //Si la velocidad calculada es MENOR a 0.5KM/h y...  
 if (digitalRead(X1\_VT\_05) == LOW)  
 { //Si la lectura es 1 digital, se ejecuta el error 1  
 digitalWrite(A0,HIGH);  
 digitalWrite(A1,HIGH);  
 digitalWrite(A2,HIGH);  
 digitalWrite(A3,LOW);  
 Serial.println("ERROR\_013: la variable X1\_VT\_05 debia arrojar un CERO, pero recibio un UNO\nERROR CONTROL DE VELCIDAD RELE 0.5KM/H");  
 error\_001();  
 break;  
 }  
 else if (digitalRead(X7\_VT\_05) == LOW)  
 { //Si la lectura es 1 digital, se ejecuta el error 3  
 digitalWrite(A0,LOW);  
 digitalWrite(A1,LOW);  
 digitalWrite(A2,LOW);  
 digitalWrite(A3,HIGH);Serial.println("ERROR\_023: la variable X7\_VT\_05 debia arrojar un CERO, pero recibio un UNO\nERROR CONTROL DE VELCIDAD RELE 0.5KM/H");  
 error\_003();  
 break;  
 }  
 else if (digitalRead(X6\_VT\_05) == LOW)  
 { //Si la lectura es 1 digital, se ejecuta el error 5  
 digitalWrite(A0,HIGH);  
 digitalWrite(A1,LOW);  
 digitalWrite(A2,LOW);  
 digitalWrite(A3,HIGH);  
 Serial.println("ERROR\_033: la variable X6\_VT\_05 debia arrojar un CERO, pero recibio un UNO\nERROR CONTROL DE VELCIDAD RELE 0.5KM/H");  
 error\_005();  
 break;  
 }  
 else  
 {  
 break;  
 }  
 break;  
 }  
 }  
 void lecturasEnBajada6()  
 {  
 Serial.println(digitalRead(X1\_VT\_6));  
 Serial.println(digitalRead(X7\_VT\_6));  
 Serial.println(digitalRead(X6\_VT\_6));  
 while (vel <= vel\_2)  
 { //Si la velocidad calculada es MENOR a 6km/h y...  
 if (digitalRead(X1\_VT\_6) == LOW)  
 { //Si la lectura es 1 digital, se ejecuta el error 2  
 digitalWrite(A0,LOW);  
 digitalWrite(A1,HIGH);  
 digitalWrite(A2,LOW);  
 digitalWrite(A3,HIGH);  
 Serial.println("ERROR\_014: la variable X1\_VT\_6 debia arrojar un CERO, pero recibio un UNO\nERROR CONTROL DE VELCIDAD RELE 6KM/H");  
 error\_002();  
 break;  
 }  
 else if (digitalRead(X7\_VT\_6) == LOW)  
 { //Si la lectura es 1 digital, se ejecuta el error 2  
 digitalWrite(A0,HIGH);  
 digitalWrite(A1,HIGH);  
 digitalWrite(A2,LOW);  
 digitalWrite(A3,HIGH);  
 Serial.println("ERROR\_024: la variable X7\_VT\_6 debia arrojar un CERO, pero recibio un UNO\nERROR CONTROL DE VELCIDAD RELE 6KM/H");  
 error\_004();  
 break;  
 }  
 else if (digitalRead(X6\_VT\_6) == LOW)  
 { //Si la lectura es 1 digital, se ejecuta el error 2  
 digitalWrite(A0,LOW);  
 digitalWrite(A1,LOW);  
 digitalWrite(A2,HIGH);  
 digitalWrite(A3,HIGH);  
 Serial.println("ERROR\_034: la variable X6\_VT\_6 debia arrojar un CERO, pero recibio un UNO\nERROR CONTROL DE VELCIDAD RELE 6KM/H");  
 error\_006();  
 break;  
 }  
 else  
 {  
 break;  
 }  
 break;  
 }  
 }

//---NOTAS-------------------------------------------------------------------------------

//considerar que la grafica de velocidad que se dibuja (o comprende) es lineal, por lo tanto, su aceleración es constante mientras que a través  
//de una ecuacion de posición en funcion del tiempo es parabolico, esto considerando el M.R.U.A. (movimiento rectilineo uniforme acelerado).  
//El tiempo aproximado desde la salida del tren hasta la llegada a la estacion toma aproximadamente 1:09 minutos.  
//Se cambio la proporcion de la frecuencia de 2.5 a 1.25 veces la frecuencia, esto debido a que el programa toma mas tiempo en las iteraciones.  
//los casos de errores evalua una tarjeta a la vez, queda pendiente versionar casos de pruebas para multiples combinaciones de tarjetas.

## Programa Display PIC16F628A

#include <16F628A.h>  
#use delay(clock = 4000000) /\* OSCILADOR INTERNO \*/  
#FUSES NOWDT /\* No Watch Dog Timer\*/  
#FUSES PUT /\* Power Up Timer\*/  
#FUSES NOPROTECT /\* Code not protected from reading\*/  
#FUSES NOBROWNOUT /\* No brownout reset\*/  
#FUSES NOMCLR /\* Master Clear pin disabled\*/  
#FUSES NOLVP /\* No low voltage prgming, B3(PIC16) or B5(PIC18) used for I/O\*/  
#FUSES NOCPD  
#FUSES INTRC //Activacion uso cristal interno

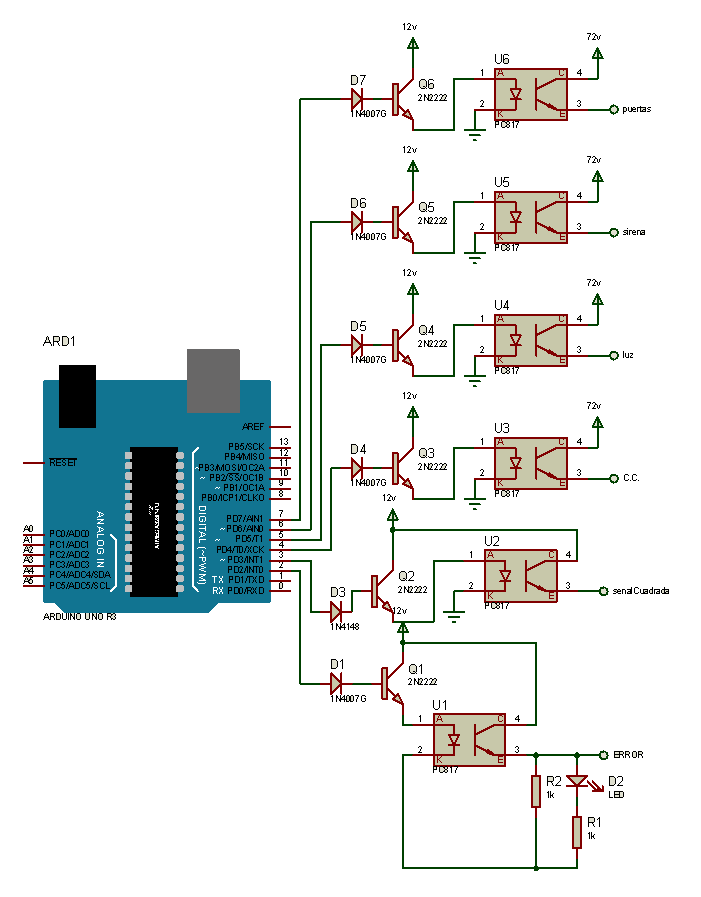
#include <lcd.c> /\*libreria para usar el lcd\*/  
#define LCD\_ENABLE\_PIN PIN\_B0 /\* ASIGNAMOS EL PIN "B0" COMO SALIDA PARA E \*/  
#define LCD\_RS\_PIN PIN\_B1 /\* ASIGNAMOS EL PIN "B1" COMO SALIDA PARA RS\*/  
#define LCD\_RW\_PIN PIN\_B2 /\* ASIGNAMOS EL PIN "B2" COMO SALIDA PARA RW\*/  
#define LCD\_DATA4 PIN\_B4 /\* ASIGNAMOS EL PIN "B4" COMO SALIDA PARA D4\*/  
#define LCD\_DATA5 PIN\_B5 /\* ASIGNAMOS EL PIN "B5" COMO SALIDA PARA D5\*/  
#define LCD\_DATA6 PIN\_B6 /\* ASIGNAMOS EL PIN "B6" COMO SALIDA PARA D6\*/  
#define LCD\_DATA7 PIN\_B7 /\* ASIGNAMOS EL PIN "B7" COMO SALIDA PARA D7\*/

void main()

{  
 output\_high(pin\_a4);  
 delay\_ms(500);  
 output\_low(pin\_a4);  
 lcd\_init();  
 lcd\_putc("\f");  
 do{  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc,"Esperando ERROR");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc, "Registrador de Eventos");  
 //funcion de arduino lecturasEnSubida05  
 if (input(PIN\_A0) == 1 && input(PIN\_A1)==0 && input(PIN\_A2)==0 && input(PIN\_A3)==0)

{  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc,"ERROR\_011: la variable X1\_VT\_05");  
 break;  
 }  
 else if (input(PIN\_A0) == 0 && input(PIN\_A1)==1 && input(PIN\_A2)==0 && input(PIN\_A3)==0)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc, "ERROR\_021: la variable X7\_VT\_05");  
 break;  
 }  
 else if (input(PIN\_A0) == 1 && input(PIN\_A1)==1 && input(PIN\_A2)==0 && input(PIN\_A3)==0)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc, "ERROR\_031: la variable X6\_VT\_05");  
 break;  
 }  
 //funcion de arduino lecturasEnSubida6  
 else if (input(PIN\_A0) == 0 && input(PIN\_A1)==0 && input(PIN\_A2)==1 && input(PIN\_A3)==0)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc,"ERROR\_012: la variable X1\_VT\_6");  
 break;  
 }  
 else if (input(PIN\_A0) == 1 && input(PIN\_A1)==0 && input(PIN\_A2)==1 && input(PIN\_A3)==0)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc, "ERROR\_022: la variable X7\_VT\_6");  
 break;  
 }  
 else if (input(PIN\_A0) == 0 && input(PIN\_A1)==1 && input(PIN\_A2)==1 && input(PIN\_A3)==0)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc, "ERROR\_032: la variable X6\_VT\_6");  
 break;  
 }  
 //funcion de arduino lecturasEnBajada05  
 else if (input(PIN\_A0) == 1 && input(PIN\_A1)==1 && input(PIN\_A2)==1 && input(PIN\_A3)==0)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc, "ERROR\_013: la variable X1\_VT\_05");  
 break;  
 }  
 else if (input(PIN\_A0) == 0 && input(PIN\_A1)==0 && input(PIN\_A2)==0 && input(PIN\_A3)==1)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc,"ERROR\_023: la variable X7\_VT\_05");  
 break;  
 }  
 else if (input(PIN\_A0) == 1 && input(PIN\_A1)==0 && input(PIN\_A2)==0 && input(PIN\_A3)==1)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc,"ERROR\_033: la variable X6\_VT\_05");  
 break;  
 }  
 //funcion de arduino lecturasEnBajadas6  
 else if (input(PIN\_A0) == 0 && input(PIN\_A1)==1 && input(PIN\_A2)==0 && input(PIN\_A3)==1)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc, "ERROR\_014: la variable X1\_VT\_6");  
 break;  
 }  
 else if (input(PIN\_A0) == 1 && input(PIN\_A1)==1 && input(PIN\_A2)==0 && input(PIN\_A3)==1)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc,"ERROR\_024: la variable X7\_VT\_6");  
 break;  
 }  
 else if (input(PIN\_A0) == 0 && input(PIN\_A1)==0 && input(PIN\_A2)==1 && input(PIN\_A3)==1)  
 {  
 lcd\_putc("\f");  
 lcd\_gotoxy(1,1);  
 printf(lcd\_putc, "Error en:");  
 lcd\_gotoxy(1,2);  
 printf(lcd\_putc,"ERROR\_034: la variable X6\_VT\_6");  
 break;  
 }  
 }while(true);  
}

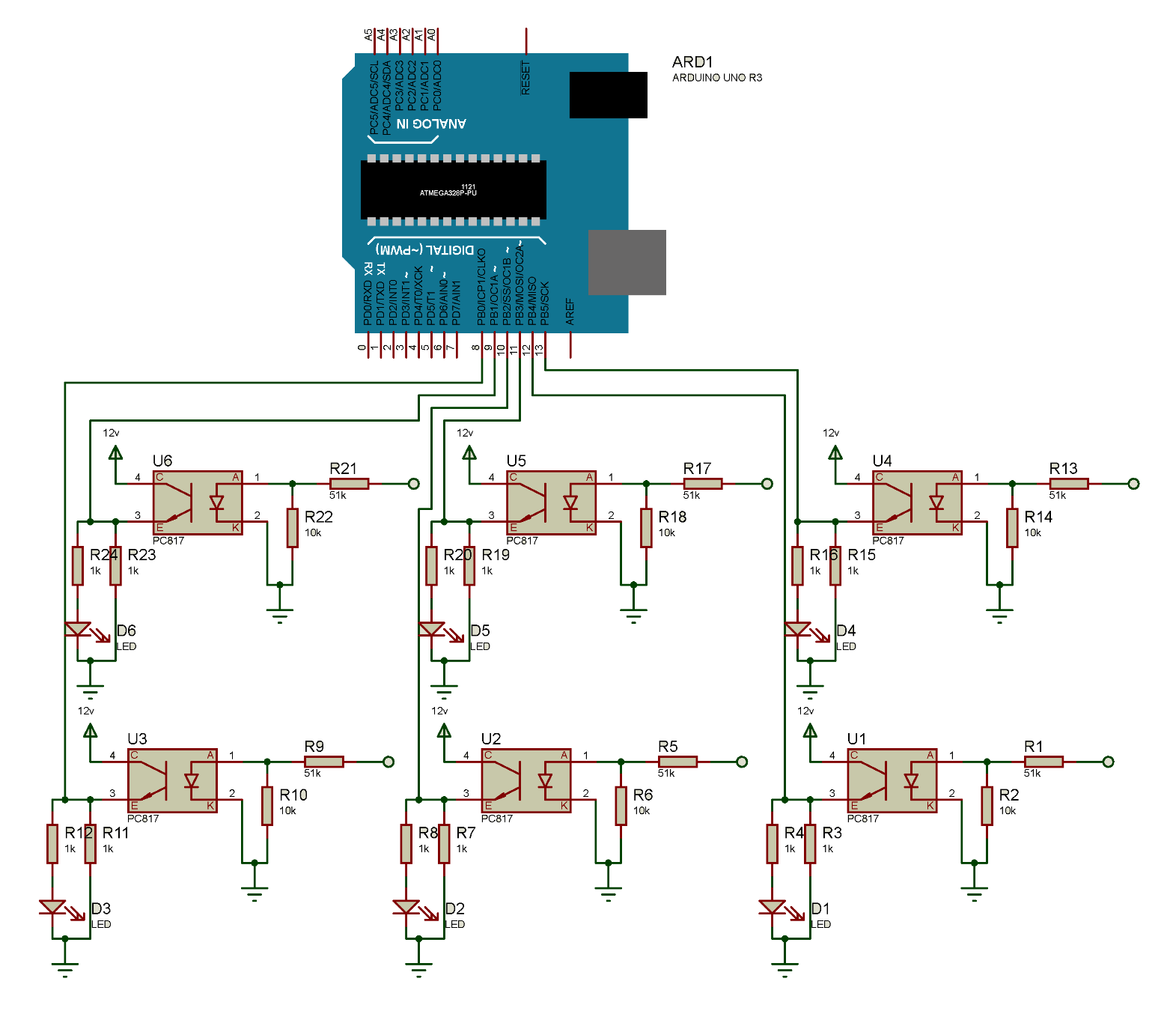
## Etapa de control (Salidas del Arduino).



Salida indicadores/Entradas al EVR.

\*El Arduino se alimenta con 12v

## Etapa de Potencia (Entradas del Arduino).



Entradas al Arduino, señales provenientes del EVR (72v).

X1.RELÉ.6

X1.RELÉ.05

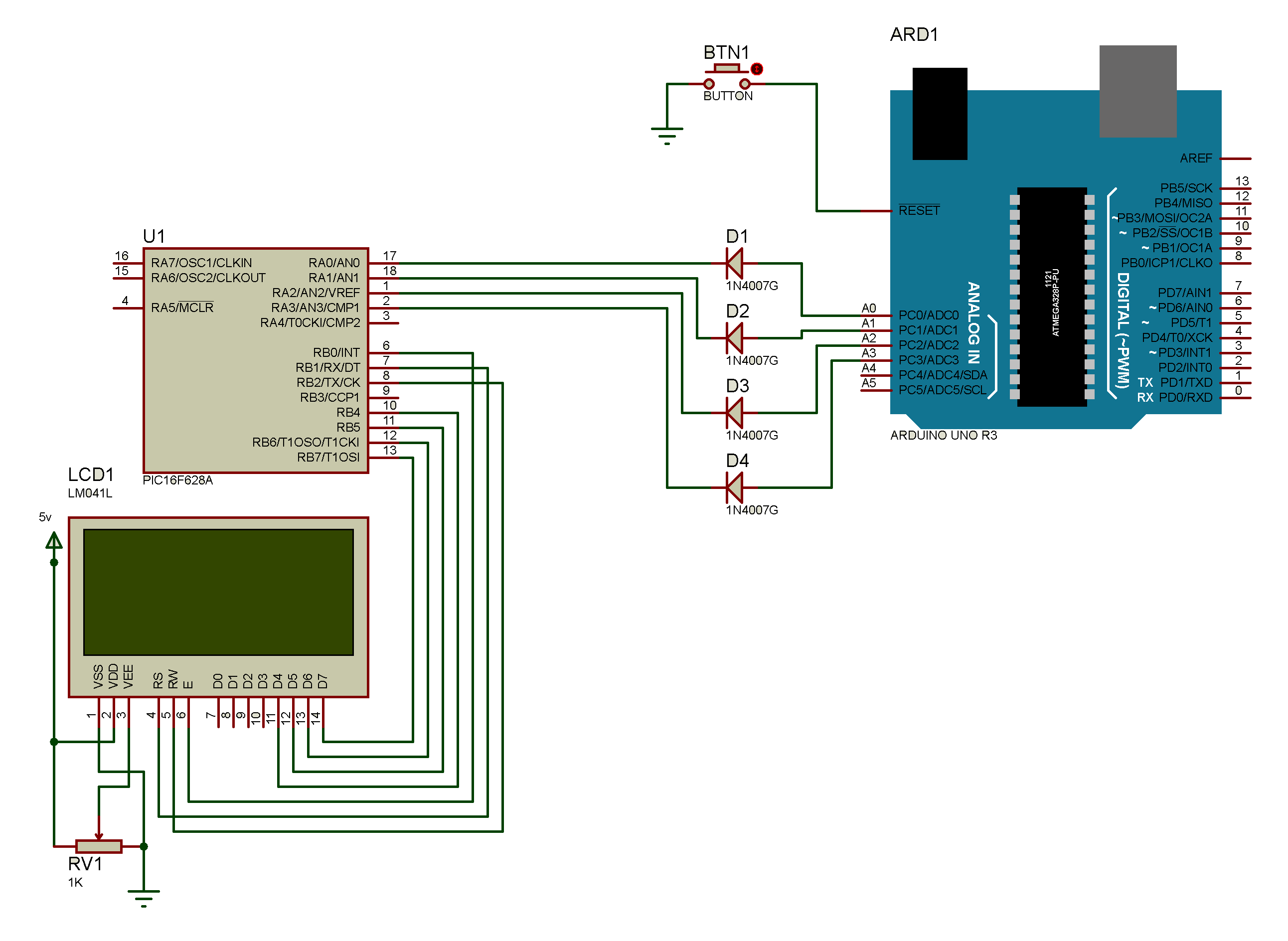
X7.RELÉ.05

X6.RELÉ.05

X6.RELÉ.6

X7.RELÉ.6

## Etapa de Comunicación.



El PIC16f628A y el Display se alimentan directamente de los 5v proporciona directamente dicha salida el Arduino.

El botón de Reset solo reinicia el funcionamiento del Arduino, no así el de la pantalla.

## Control de velocidad

## Registrador de eventos