# Manual de uso Banco de Pruebas EVR

Imagen que contiene motor

Descripción generada automáticamente

Contenido

[Manual de uso Banco de Pruebas EVR 1](#_Toc79496887)

[Preparación. 3](#_Toc79496888)

[Energización 3](#_Toc79496889)

[Uso 4](#_Toc79496890)

[Componentes. 4](#_Toc79496891)

[Interpretación de mediciones. 5](#_Toc79496892)

[Interpretación HMI laboratorio. 5](#_Toc79496893)

[Mensajes. 7](#_Toc79496894)

[Mensajes en PC. 7](#_Toc79496895)

[Mensajes en pantalla LCD. 10](#_Toc79496896)

[Mensajes lumínicos y/o sonoros. 10](#_Toc79496897)

[Anexo. 12](#_Toc79496898)

[Acceder a Monitor Serial 12](#_Toc79496899)

[Problemas de comunicación. 14](#_Toc79496900)

[Carga de programa cafProyect01 en Banco de Pruebas EVR V1. 14](#_Toc79496901)

[Código programa Banco de Pruebas EVR V1. 15](#_Toc79496902)

## Preparación.

Para realizar pruebas respecto al funcionamiento del equipo EVR Teloc 1500 de Hasler, es necesario contar con:

* Banco de prueba EVR V1
* Equipo Teloc 1500 a evaluar
* HMI, SIE y comunicación MVB del laboratorio CAF Chile.

El procedimiento de preparación para las pruebas consiste en los siguientes pasos:

1. Conectar los terminales del banco de pruebas EVR V1 con las correspondientes tarjetas del equipo Teloc 1500.
2. Realzar la comunicación MVB entre el equipo Teloc 1500 y la SIE del laboratorio

## Energización

El banco de pruebas necesita de 2 alimentaciones, una de 5VDC a 12VDC para la etapa lógica, y otra de 72VDC para la etapa de potencia y alimentación del equipo Teloc 1500 y HMI. Para alimentar la etapa lógica se puede utilizar la comunicación USB entre el banco de pruebas EVR V1 y el PC, o un cargador de teléfono, o una fuente de 12VDC.

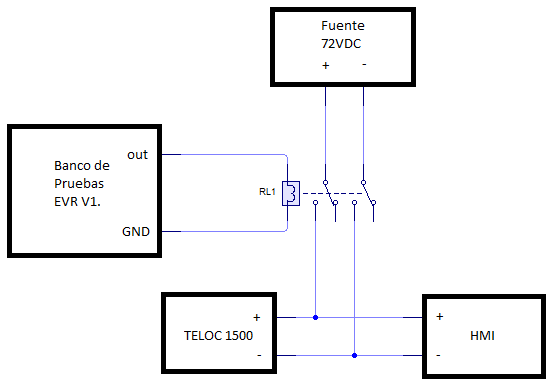
Los 72VDC son suministrados por cualquiera de las fuentes habilitadas en el laboratorio.

Ilustración 1, Diagrama de conexionado de alimentación para el banco de pruebas

## Uso

El Banco de pruebas EVR V1 controla la alimentación del equipo Teloc 1500, HMI, como también la generación de la velocidad, y evalúa en paralelo el estado de los relés de adquisición de velocidad del dispositivo a evaluar, en busca de alguna falla en estos.

Para iniciar las pruebas primero se deben alimentar la etapa de potencia y después la etapa lógica, una vez realizado esto, el Banco de pruebas de pruebas se encargará de realizar las pruebas en el siguiente orden:

1. Prueba de inicio: Al encender el Teloc 1500, evaluara que su encendido sea el correcto, esto hace alusión a que se enciendan primero los relés de las tarjetas X1 (0,5Km/h y 6km/h) y X6 (6km/h), si esto no sucede, indicara su correspondiente error.
2. Adquisición de velocidad (simulación de viaje): Se simula un viaje generando una señal cuadrada que aumentara en frecuencia, indicando la adquisición de velocidad, y vigilando que los relés de adquisición de velocidad del equipo Teloc 1500 cambien de estado, tanto en el proceso de aceleración como desaceleración, teniendo una pausa entre viajes donde se pueden agregar las ‘acciones de estación’.

La cantidad de ‘viajes’ que se le simulan a través del Banco de pruebas EVR V1 son 27, con una duración aproximada de 33,4 minutos. Luego de esto, el equipo Teloc 1500 permanecerá apagado por un tiempo de 1 minuto, para realizar nuevamente las pruebas 1 y 2. Este proceso se repetirá hasta que se estime conveniente.

A través de la pantalla HMI en la sección Control/Mantenimiento/Equipos/R2/EVR.VA podemos visualizar el estado del equipo Teloc 1500, la velocidad que se estará registrando, y el estado de alguna de las tarjetas.

## Componentes.

El Banco de pruebas EVR V1 es un sistema integrado de múltiples elementos que trabajan juntos, con tal de desarrollar y evaluar el estado de los equipos Teloc 1500, el cual está compuesto de la siguiente forma:

* **Pantalla LCD**: Se encarga de entregar los errores respecto a los relés de adquisición de velocidad, incluye un regulador de brillo.
* **Switch de Reset**: cuando está orientado hacia RESET anula la prueba y suspende la ejecución del programa, cuando está orientado a ‘POWER 12V’ la prueba reinicia desde cero.
* **Leds indicativos de adquisición de velocidad**: Estos leds indican en qué estado se encuentra los relés de adquisición de velocidad VT.5 y VT6 de las tarjetas X1, X6 y X7.
* **Sensor de velocidad**: carga que se alimenta con 12VDC a través de la tarjeta X4
* **Relé control encendido EVR**: como su nombre lo indica, controla el encendido del EVR
* **Alimentador 12VDC X1**: conector de 12VDC provenientes de la tarjeta X1 y que alimenta la generación de la señal cuadrada, y señal de error.
* **Conectores X1, X4, X5, X6, X7 y X8**: conectores que salen del banco de pruebas EVR V1, y que se conectan a sus correspondientes tarjetas en el Teloc 1500.
* Arduino Uno R3: Placa que se encarga de la realización y control de la prueba, gracias a su programa integrado.

## Interpretación de mediciones.

A partir de sus componentes se pueden interpretar fallos que no son detectados por el Arduino, pero si pueden ser detectados e interpretados por el usuario. Estos son algunos de los errores que se pueden detectar con el uso de mutitester digital, al medir voltaje:

* Sensor de velocidad: inexistencia de los 12VDC, esto implica problemas con la tarjeta X4, que es la que se encarga de alimentar dicho componente, o X8 que es parte de la condición para que pueda ser alimentada la carga.
* Alimentador 12VDC X1: inexistencia de los 12VDC, esto indica un error en la tarjeta X1, que impide la generación de la señal cuadrada.
* Relé control encendido EVR: al enclavarse, si sus contactos NA que pasan a NC al ocurrir el enclavamiento, no presentan 72VDC, implica problemas en la alimentación del Teloc 1500 y su funcionamiento.

También el equipo HMI nos brinda información pertinente al estado de las tarjetas de la siguiente manera:

* Sondas: las sondas 1 y 2 son las que están vinculadas a la tarjeta X8, y la sonda 3 a la tarjeta X1. Cuando alguna presenta algún fallo, el HMI las marcara con un 1 en la pantalla de despliegue del error. Si el fallo es 0XDA00000000 esto implica un error en la alimentación del sensor de velocidad (carga del banco de pruebas) y que requiere énfasis en la tarjeta X4 y/o X8
* Umbrales: estos están relacionados con los relés de adquisición de velocidad. Donde el umbral 1 es X6, umbral 2 es X7 y el umbral 3 para X1 respectivamente, aunque existen errores que pasan de ser percibidos en este punto, pero si son detectados por el banco de pruebas EVR V1.

## Interpretación HMI laboratorio.

Al dirigirnos a través del HMI a la ruta Control/Mantenimiento/Equipos/R2/EVR.VA podremos visualizar distintos estados en la pantalla que representan estados distintos del funcionamiento del equipo Teloc 1500.

Una captura de pantalla de un videojuego

Descripción generada automáticamente con confianza mediaEn la ilustración 2 vemos que la columna de variables se encuentra en rojo, esto indica un problema de alimentación del dispositivo Teloc 1500. Ya que este no enciende y por consecuente no tiene un inicio correcto en su primera etapa de preparación.

Ilustración 2, Teloc no energizado.

Una captura de pantalla de un videojuego

Descripción generada automáticamente con confianza mediaEn la ilustración 3 vemos un problema a través del HMI que tiene que ver con el conexionado del equipo Teloc 1500 al desplegar el detalle del error 0xDA000000 en la fila **EVR\_R2\_Bits\_estado\_TELOC**.

Ilustración 3, Problemas lazo de integridad.

Este tipo de error es debido a un mal funcionamiento de las sondas 1 y 2 (sensor de velocidad, uqe puede ser debido a un mal funcionamiento de la tarjeta X8 y/o X4)) que arrastra un mal funcionamiento de los umbrales 1 y 2 (que estan ligados a los reles de adquisición de velocidad de las tarjetas X6 y X7).

Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

Ilustración 4, inicio y estado correcto del EVR

En la ilustración 4 , vemos un estado de pantalla optimo del equipo Teloc 1500, al desplegar el detalle en su fila **EVR\_RE\_Bits\_Estado\_TELOC.**

Una captura de pantalla de un videojuego

Descripción generada automáticamente con confianza media

Ilustración 5, y estado correcto del EVR

En la ilustración 5 vemos el estado correcto de los parámetros del EVR en la pantalla del HMI, **no obstante eso no significa que el quipo este en buen estado, sino que su inicio ha sido correcto.** Esto debido a que la velocidad de actualización es más lenta que el monitoreo por parte del banco de pruebas EVR V1. Por lo tanto puede detectar errores que no logran ser percibidos instantáneamente por el sistema.

Es posible que se origine en el **EVR\_R2\_Bits\_Estado\_TELOC** el bit 0x00080000, este está vinculado al fallo de la variable **VACMA\_fallo\_3** en la adquisición de velocidad cuando se superan el umbral de los 0.5km/h. Este podría ser considerado un problema si al tener velocidad simulada igual a 0 siguiese activo. Ya que indicaría que uno de los relés de adquisición de velocidad podría aun estar conmutado.

## Mensajes.

El equipo tiene 3 formas te entregar mensajes de errores, vía pc mediante el Monitor Serial, del programa Arduino. Vía pantalla LCD donde se entrega mensajes referentes al error encontrado, y señalización lumínica y/o sonora. A continuación, se señalarán los mensajes en cada uno de los casos.

### Mensajes en PC.

Para la comunicación entre el PC y el banco de pruebas vía USB haciendo uso del Monitor Serial del programa Arduino nos encontraremos con los siguientes mensajes:

* ‘**EVR: ON**’ Este mensaje indica que se realiza la alimentación del sistema con 72VDC
* ‘**1 OK**’ Indica que el inicio del EVR ha sido correcto solo iniciando los relés X1\_VT\_05, X1\_VT\_6 y X6\_VT6.
* ‘**1 PRECAUCION**’ El mensaje nos indica una anomalía en la iniciación del sistema del TELOC con algún relé de adquisición de velocidad iniciando fuera de tiempo.
* ‘**2 OK**’ indica que todos los relés de adquisición de velocidad están en estado lógico 1

Texto

Descripción generada automáticamente

Ilustración 6, ejemplo del inicio correcto de un equipo Teloc 1500 en el Banco de pruebas EVR V1.

* ‘**Ciclo: n Viaje N° x**’ Ciclo representa en que prueba va, desde n=1 hasta que se estime conveniente, viaje representa a cada simulación de adquisición de velocidad con detenciones que se realiza, que van desde x=1 a x=27, es decir, cada en cada ciclo se realizan 27 viajes.

La cantidad de viajes puede ser cambiado alterando el límite de variable **k** en el ciclo **for** de la función **testEVR()** en el código de **Arduino.**

* ‘**Acciones en estación**’ pulsos de acciones disponibles que se pueden utilizar para registrar alguna actividad en detención, en el Teloc 1500.
* ‘**TREN sale de ESTACIÓN**’ indica el inicio de la adquisición de velocidad, acelerando con un crecimiento lineal.
* ‘**VELOCIDAD MAXIMA**’ indica que se ha alcanzado la velocidad máxima del vehículo, y empieza con la desaceleración virtual de este.
* ‘**TREN ha llegado a DESTINO**’ indica el final del viaje del tren, para comenzar nuevamente desde las ‘acciones en estación’.
* ‘**EVR: OFF**’ indica que se completaron los 27 viajes del ciclo, y este apagara todo el sistema por un periodo de 1 minuto.

Texto

Descripción generada automáticamente

Ilustración 7, ejemplo de visualización de avance de los viajes, y finalización de estos, hasta el apagado del equipo Teloc 1500 por parte del banco de Pruebas EVR V1.

#### Errores

Los errores con las tarjetas con relés de control de adquisición de velocidad están codificados[[1]](#footnote-1) de la siguiente manera:

* + ERROR\_011: X1\_VT\_05 = 1
  + ERROR\_021: X7\_VT\_05 = 1
  + ERROR\_031: X6\_VT\_05 = 1
  + ERROR\_012: la variable X1\_VT\_6 = 1
  + ERROR\_022: X7\_VT\_6 = 1
  + ERROR\_032: la variable X6\_VT\_6 = 1
  + ERROR\_013: X1\_VT\_05 = 0
  + ERROR\_023: X7\_VT\_05 = 0
  + ERROR\_033: X6\_VT\_05 = 0
  + ERROR\_014: X1\_VT\_6 = 0
  + ERROR\_024: X7\_VT\_6 = 0
  + ERROR\_034: X6\_VT\_6 = 0

Donde se nos muestra primero la tarjeta en la que se encontró el problema, el tipo de relé que tuvo problemas, para finalizar mostrando el estado que tuvo al momento de ser identificado el error. El mensaje error finaliza mostrando el estado del grupo de las tarjetas en el orden X1, X7, X6

Imagen que contiene Tabla

Descripción generada automáticamente

Ilustración 8, ejemplo de visualización de un mensaje error. El sistema muestra el mensaje de error vinculado a la primera tarjeta en la que lo detecta, y el estado de las tarjetas X1, X7, X6.

Existe también el mensaje de ‘**Errores encontrados**’, este es un contador, como su nombre lo indica, de errores encontrados en la prueba, por cada error que identifica este reinicia la prueba, con un máximo de 5 intentos, el cual puede ser modificado en el código de Arduino cambiando el valor de la variable **n** en la función **countError()**.

### Mensajes en pantalla LCD.

Los mensajes en la pantalla LCD son resumidos, y almacenados en su propio sistema. Los mensajes que pueden ser visualizados en la pantalla son los siguientes.

* ‘Banco de Pruebas EVR’ este mensaje por defecto se mantendrá al iniciar el sistema hasta que se encuentre algún error.

El listado de errores se interpreta de igual forma que los códigos de error que se muestran a través del PC

* ‘ERROR!!! ERROR\_011: X1\_VT\_05’
* ‘ERROR!!! ERROR\_021: X7\_VT\_05’
* ‘ERROR!!! ERROR\_031: X6\_VT\_05’
* ‘ERROR!!! ERROR\_012: X1\_VT\_6’
* ‘ERROR!!! ERROR\_022: X7\_VT\_6’
* ‘ERROR!!! ERROR\_032: X6\_VT\_6’
* ‘ERROR!!! ERROR\_013: X1\_VT\_05’
* ‘ERROR!!! ERROR\_023: X7\_VT\_05’
* ‘ERROR!!! ERROR\_033: X6\_VT\_05’
* ‘ERROR!!! ERROR\_014: X1\_VT\_6’
* ‘ERROR!!! ERROR\_024: X7\_VT\_6’
* ‘ERROR!!! ERROR\_034: X6\_VT\_6’
* ‘ERROR!!! ERROR\_001: preparación X1’
* ‘ERROR!!! ERROR\_002: preparación X6’

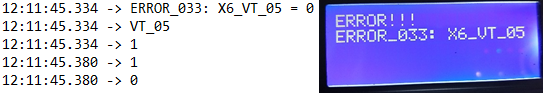


Ilustración 9, vemos el mismo error informado tanto por el Monitor Serial de Arduino, como por la pantalla LCD del Banco de pruebas.

### Mensajes lumínicos y/o sonoros.

Este tipo de mensaje visual y/o audible, son secuencias de on/off con duración de 0,5 segundos, y que se repiten cada 1 segundo. La tabla de secuencias se describe de la siguiente forma:

|  |  |  |
| --- | --- | --- |
| Secuencia | Intervalo | Error |
| 1 | 0 segundos | Problemas con tarjeta X1\_VT\_05 |
| 2 | 1 segundo | Problemas con tarjeta X1\_VT\_6 |
| 3 | 1 segundo | Problemas con tarjeta X7\_VT\_05 |
| 4 | 1 segundo | Problemas con tarjeta X7\_VT\_6 |
| 5 | 1 segundo | Problemas con tarjeta X6\_VT\_05 |
| 6 | 1 segundo | Problemas con tarjeta X6\_VT\_6 |
| 7 | 1 segundo | Problemas con X1 en Preparación |

# Anexo.

## Acceder a Monitor Serial

Para hacer uso del Monitor Serial de Arduino, es necesario conectar el banco de pruebas vía UBS a un computador con el programa. Una vez realizado esto, iniciar el banco de pruebas

Cada vez que se inicie el Monitor Serial implica un reinicio por parte del banco de pruebas, por lo tanto es aconsejable realizar esta acción previa iniciación de las pruebas, y evitar tanto el programa de Arduino como el Monitor Serial para evitar un reinicio al querer abrirlo nuevamente.

Una vez establecida la conexión, abrir el programa Arduino, haciendo clic en el icono de escritorio (1), esperar a que se abra la ventana correspondiente (2), en la cinta superior de opciones, entrar en ‘Herramientas’ y seleccionar ‘Monitor Serie’ o con las teclas Ctrl + Shift + M (3), se abrirá una nueva ventana en blanco en la que se mostraran cada uno de los mensajes de acciones que ocurren en el proceso (4).

Icono

Descripción generada automáticamente

1. Icono de la aplicación de Arduino.

Interfaz de usuario gráfica, Aplicación

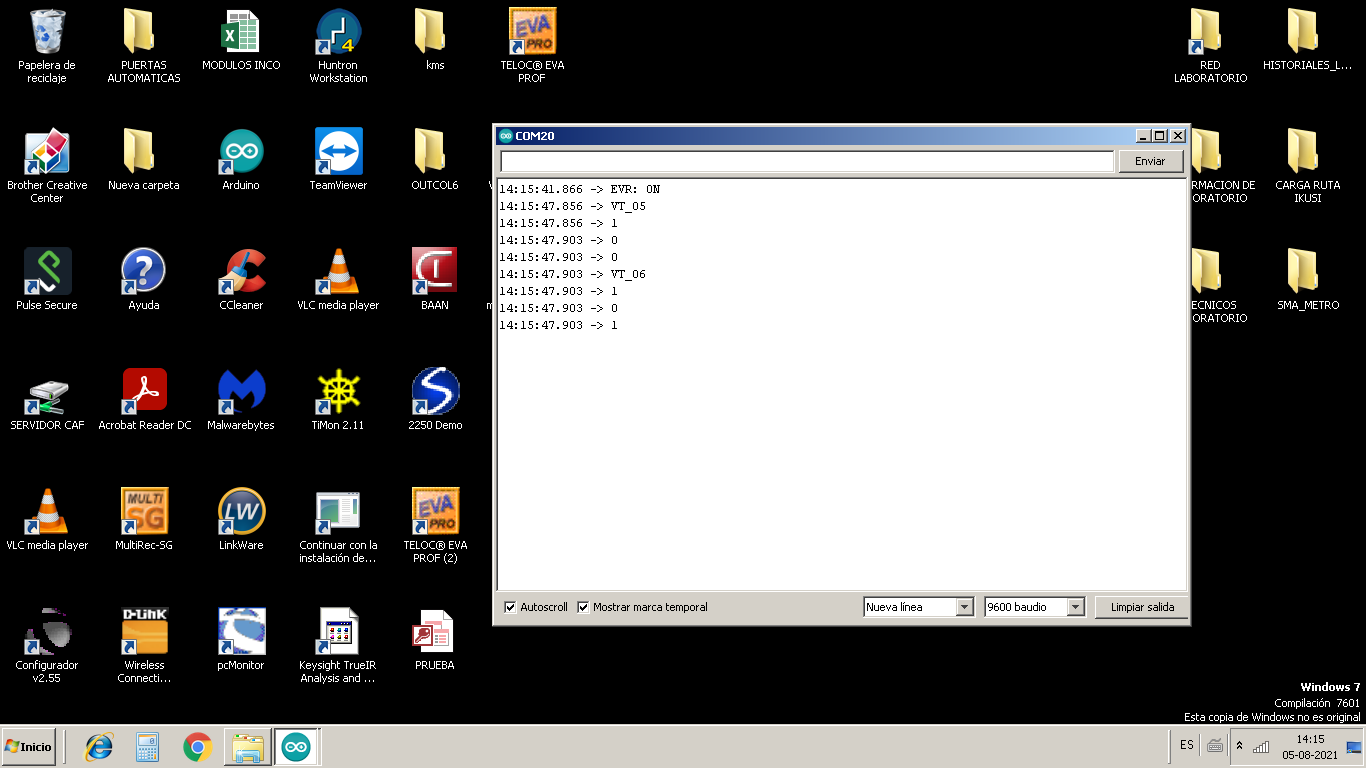
Descripción generada automáticamente

1. Ventana de Arduino.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

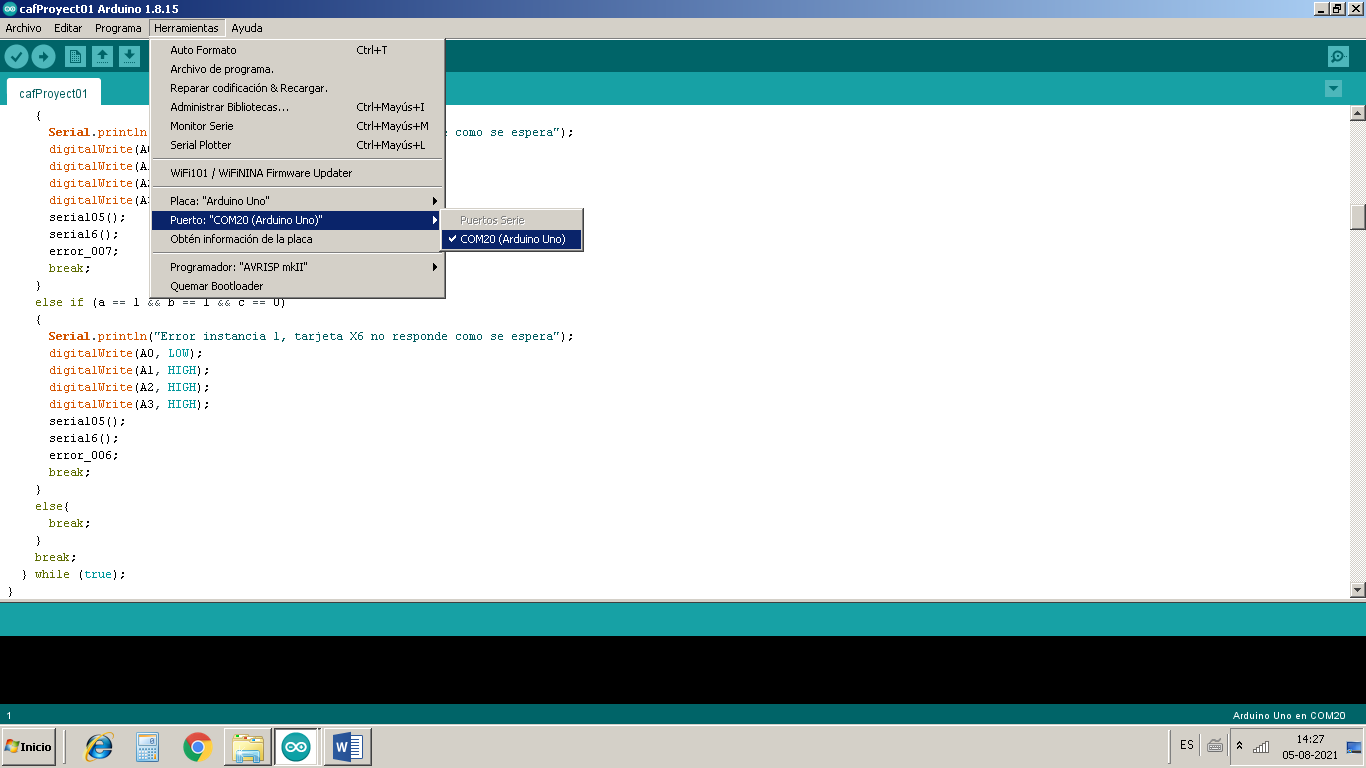
1. Acceder al Monitor Serial



1. Monitor Serial

## Problemas de comunicación.

En caso de que la comunicación falle, revisar que en la opción ‘Puerto’, en el menú desplegable de ‘Herramientas’ (3) este seleccionado el puerto COM20 (Arduino UNO) (5).



(5)

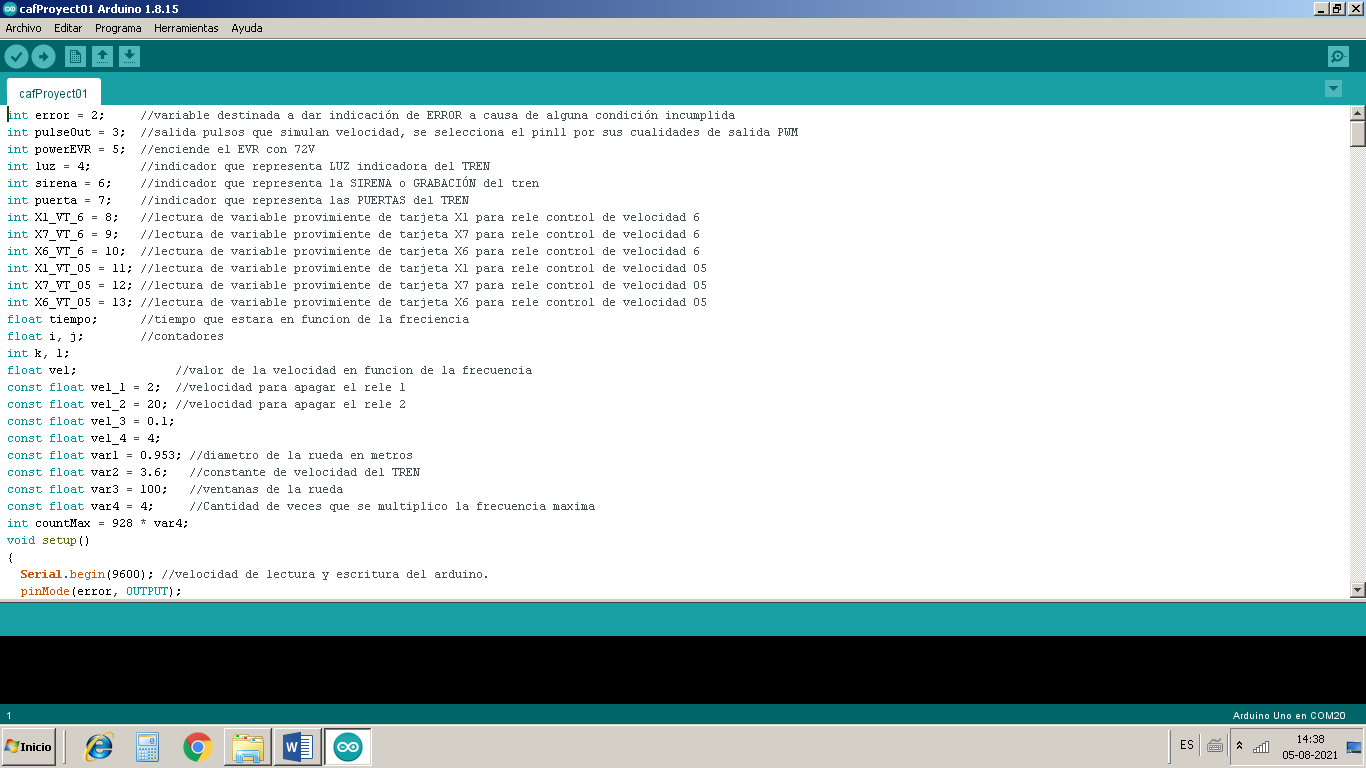
Para ejecutar el Monitor Serial no es necesario que el código que este en la ventana de la aplicación de Arduino sea el mismo que este cargado en el banco de pruebas.

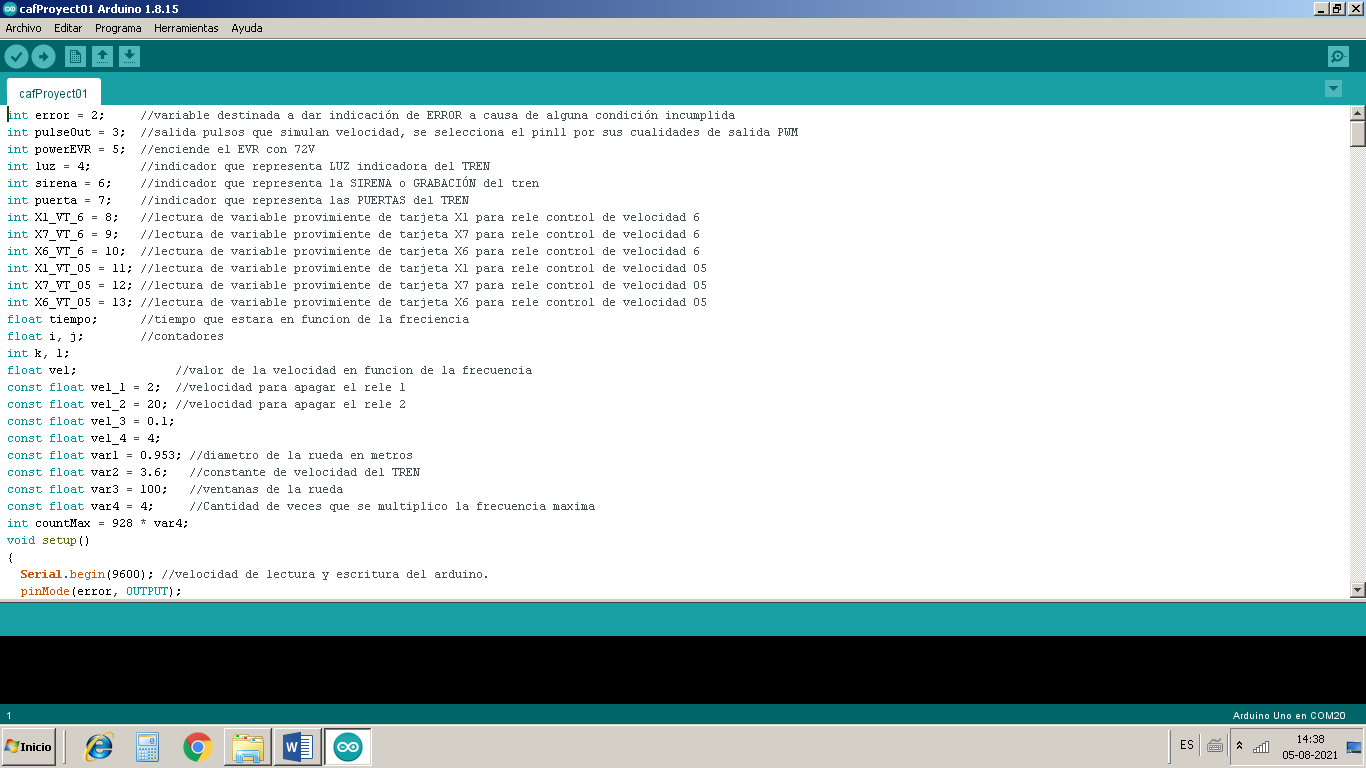
Si dicha opción no es solución al problema es posible que existan problemas relacionados con la placa Arduino que se encuentra al interior del Banco de pruebas EVR V1, en ese caso es recomendable reemplazar la placa Arduino UNO por una nueva y volver a cargar el problema cafPoyect01.ino en ella.

## Carga de programa cafProyect01 en Banco de Pruebas EVR V1.

Para modificar el programa del Arduino o volver a cargar el programa es necesario abrir su archivo ubicado en la siguiente Ruta:

[C:\Users\Juan Videla\Documents\IGNACIO\bancoEVR\_V1\cafProyect01](file:///C:\Users\Ignacio\Downloads\bancoEVR_V1\cafProyect01)

Una vez abierto, y conectado el banco de pruebas EVR V1 vía USB con el PC hacer las modificaciones necesarias en el código. Haya o no modificaciones en el código, los pasos siguientes son compilar el código y cargarlo en el banco de pruebas.

* Para compilar el código hay que hacer clic en el botón ‘Verificar’ o presionar las teclas Ctrl + R.
* Para cargar el código hay que hace clic en el botón ‘Subir’ o presionar las teclas Ctrl + U.

Una vez realizas estas acciones solo queda esperar que se empiece a ejecutar el programa, que puede ser monitorizado a través del Monitor Serial, o de la ejecución de las acciones programadas en el banco de pruebas EVR V1.

## Código programa Banco de Pruebas EVR V1.

Se adjunta el código que está incluido en el microcontrolador del banco de pruebas EVR V1. Con fecha del 10/08/2021. Este programa controla la energización del sistema según el diagrama del manual de uso. Simula la adquisición de velocidad del tren, son un aumento lineal de la velocidad, y vigila los estados lógicos de los relés de adquisición con relación con los umbrales de velocidad.

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 powerEVR = 5; //enciende el EVR con 72V

int luz = 4; //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

float tiempo; //tiempo que estara en funcion de la freciencia

float i, j; //contadores

int k, l, m, n;

float vel; //valor de la velocidad en funcion de la frecuencia

const float vel\_1 = 2; //velocidad para apagar el rele 1

const float vel\_2 = 20; //velocidad para apagar el rele 2

const float vel\_3 = 0.1;

const float vel\_4 = 4;

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 = 4; //Cantidad de veces que se multiplico la frecuencia maxima

int countMax = 928 \* var4;

void setup()

{

Serial.begin(9600); //velocidad de lectura y escritura del arduino.

pinMode(error, OUTPUT);

pinMode(pulseOut, OUTPUT);

pinMode(powerEVR, 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);

pinMode(A4, OUTPUT);

}

void loop()

{

testEVR();

}

void star()

{

displayOff();

displayOn();

}

void instancia01()

{

do

{

int a = digitalRead(X1\_VT\_05);

int b = digitalRead(X1\_VT\_6);

int c = digitalRead(X6\_VT\_6);

int d = digitalRead(X6\_VT\_05);

int e = digitalRead(X7\_VT\_6);

int f = digitalRead(X7\_VT\_05);

if (a == 1 && b == 1 && c == 1 && d == 0 && e == 0 && f == 0)

{

serial05();

serial6();

delay(14000);

Serial.println("1 OK");

break;

}

else if ((a == 1 && b == 1 && c == 1) && (d == 1 || e == 1 || f == 1))

{

serial05();

serial6();

delay(14000);

Serial.println("1 PRECAUCION");

break;

}

else if ((a == 0 || b == 0) && c == 1)

{

Serial.println("Error instancia 1, tarjeta X1 no responde como se espera");

digitalWrite(A0, HIGH);

digitalWrite(A1, LOW);

digitalWrite(A2, HIGH);

digitalWrite(A3, HIGH);

serial05();

serial6();

error\_007;

break;

}

else if (a == 1 && b == 1 && c == 0)

{

Serial.println("Error instancia 1, tarjeta X6 no responde como se espera");

digitalWrite(A0, LOW);

digitalWrite(A1, HIGH);

digitalWrite(A2, HIGH);

digitalWrite(A3, HIGH);

serial05();

serial6();

error\_006;

break;

}

else

{

break;

}

break;

} while (true);

}

void instancia02()

{

do

{

lecturasEnBajada05();

lecturasEnBajada6();

Serial.println("2 OK");

break;

} while (true);

}

//------------viaje-----------------------------------------------------------------------------------

void viaje()

{

digitalWrite(A4, HIGH);

//digitalWrite(cc, LOW);

enEstacion();

Serial.println("TREN sale de ESTACIÓN");

acelerar();

Serial.println("VELOCIDAD MAXIMA");

desacelerar();

Serial.println("TREN ha llegado a DESTINO");

}

//---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()

{

Serial.println("acciones en estacion");

digitalWrite(puerta, LOW);

delay(4500);

digitalWrite(luz, HIGH);

delay(2000);

digitalWrite(sirena, HIGH);

delay(2500);

digitalWrite(sirena, LOW);

digitalWrite(luz, LOW);

digitalWrite(puerta, HIGH);

delay(1000);

}

//---FUNCIONES\_POR\_ERRORES----------------------------------------------------------------------------

void error\_001()

{ //error 1 cuando X1\_VT\_05 debiera ser HIGH

int a = 0;

for (a; a < 10; a++)

{

digitalWrite(error, HIGH);

delay(500);

digitalWrite(error, LOW);

delay(500);

}

m = m+1;

countError();

}

void error\_002()

{ //error 2 cuando X1\_VT\_6 deberia ser HIGH

int a = 0;

for (a; a < 10; a++)

{

for (j = 0; j < 2; j++)

{

digitalWrite(error, HIGH);

delay(500);

digitalWrite(error, LOW);

delay(500);

}

delay(1000);

}

m = m+1;

countError();

}

void error\_003()

{ //error 3 cuando X7\_VT\_05 deberia ser HIGH

int a = 0;

for (a; a < 10; a++)

{

for (j = 0; j < 3; j++)

{

digitalWrite(error, HIGH);

delay(500);

digitalWrite(error, LOW);

delay(500);

}

delay(1000);

}

m = m+1;

countError();

}

void error\_004()

{ //error 4 cuando X7\_VT\_6 deberia ser HIGH

int a = 0;

for (a; a < 10; a++)

{

for (j = 0; j < 4; j++)

{

digitalWrite(error, HIGH);

delay(500);

digitalWrite(error, LOW);

delay(500);

}

delay(1000);

}

m = m+1;

countError();

}

void error\_005()

{ //error 5 cuando X6\_VT\_05 deberia ser HIGH

int a = 0;

for (a; a < 10; a++)

{

for (j = 0; j < 5; j++)

{

digitalWrite(error, HIGH);

delay(500);

digitalWrite(error, LOW);

delay(500);

}

delay(1000);

}

m = m+1;

countError();

}

void error\_006()

{ //error 6 cuando X6\_VT\_6 deberia ser HIGH

int a = 0;

for (a; a < 10; a++)

{

for (j = 0; j < 6; j++)

{

digitalWrite(error, HIGH);

delay(500);

digitalWrite(error, LOW);

delay(500);

}

delay(1000);

}

m = m+1;

countError();

}

void error\_007()

{ //error 7 Ninguna lectura cambia

int a = 0;

for (a; a < 10; a++)

{

for (j = 0; j = 7; j++)

{

digitalWrite(error, HIGH);

delay(500);

digitalWrite(error, LOW);

delay(500);

}

delay(1000);

}

m = m+1;

countError();

}

//---LECTURAS-------------------------------------------------------------------------------------------------------------------------------------------

void lecturasEnSubida05()

{

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: X1\_VT\_05 = 1");

serial05();

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: X7\_VT\_05 = 1");

serial05();

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: X6\_VT\_05 = 1");

serial05();

error\_005();

break;

}

else

{

break;

}

break;

}

}

void lecturasEnSubida6()

{

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 = 1");

serial6();

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: X7\_VT\_6 = 1");

serial6();

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 = 1");

serial6();

error\_006();

break;

}

else

{

break;

}

break;

}

}

void lecturasEnBajada05()

{

while (vel <= vel\_3)

{ //Si la velocidad calculada es MENOR a 0.1KM/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: X1\_VT\_05 = 0");

serial05();

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: X7\_VT\_05 = 0");

serial05();

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: X6\_VT\_05 = 0");

serial05();

error\_005();

break;

}

else

{

break;

}

break;

}

}

void lecturasEnBajada6()

{

while (vel <= vel\_4)

{ //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: X1\_VT\_6 = 0");

serial6();

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: X7\_VT\_6 = 0");

serial6();

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: X6\_VT\_6 = 0");

serial6();

error\_006();

break;

}

else

{

break;

}

break;

}

}

void powerOn()

{

Serial.println("EVR: ON");

digitalWrite(powerEVR, HIGH);

}

void powerOff()

{

delay(5000);

Serial.println("EVR: OFF");

digitalWrite(powerEVR, LOW);

delay(60000);

}

void serial05()

{

Serial.println("VT\_05");

Serial.println(digitalRead(X1\_VT\_05));

Serial.println(digitalRead(X7\_VT\_05));

Serial.println(digitalRead(X6\_VT\_05));

}

void serial6()

{

Serial.println("VT\_06");

Serial.println(digitalRead(X1\_VT\_6));

Serial.println(digitalRead(X7\_VT\_6));

Serial.println(digitalRead(X6\_VT\_6));

}

void infoViaje()

{

Serial.print("Ciclo: ");

Serial.print(l);

Serial.print(" viaje Nº ");

Serial.println(k);

}

void testEVR()

{

star();

for (l = 1; l; l++)

{

l=l+m;

powerOn();

delay(7000);

instancia01();

instancia02();

for (k = 1; k < 28; k++)

{

infoViaje();

viaje();

}

powerOff();

}

}

void countError()

{

do{

n=5;

Serial.print("Errores encontrados: ");

Serial.println(m);

if (m < n)

{

powerOff();

testEVR();

break;

}

else

{

endTestEVRForError();

break;

}

break;

}

while(true);

}

void displayOn()

{

digitalWrite(A0, LOW);

digitalWrite(A1, LOW);

digitalWrite(A2, LOW);

digitalWrite(A3, LOW);

digitalWrite(A4, HIGH);

delay(1000);

}

void displayOff()

{

digitalWrite(A0, LOW);

digitalWrite(A1, LOW);

digitalWrite(A2, LOW);

digitalWrite(A3, LOW);

digitalWrite(A4, LOW);

delay(1500);

}

void endTestEVRForError()

{

Serial.print("Revisar Historial\t");

Serial.print(m);

Serial.println("\terrores encontrados");

powerOff();

Serial.println("Fin de la prueba");

}

1. Al analizar el listado de las codificaciones de errores, observados lo siguientes: los errores finalizados en 1 y 2 tienen que ver con lo errores ocurridos en la aceleración, mientras que los errores que suceden en la desaceleración terminan su codificación en 3 y 4 [↑](#footnote-ref-1)