Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Simulador de sensores de temperatura

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Daniel Hernández Escamilla

Tutor: Antonio Martí Campoy

Curso 2017/2018

Resumen

El proyecto consiste en el desarrollo de una aplicación de escritorio, mediante el lenguaje de programación C#, que simule un conjunto de sensores térmicos de forma gráfica para poder estudiar su funcionamiento de forma más sencilla. Para ello, la aplicación se comunicará con una tarjeta virtual, pasando información del estado de los sensores y permitiendo que esta información sea obtenida finalmente mediante un programa en C que comunique con la misma tarjeta virtual.

Palabras clave: sensores de temperatura, tarjeta de adquisición de datos, tarjeta virtual, simulador.

Índice

Índice Imágenes

1. Introducción

En esta memoria se va a describir el desarrollo del trabajo final de grado "Simulador de sensores de temperatura".

Como parte de las prácticas de laboratorio de la asignatura Informatización Industrial del Grado en Ingeniería en Tecnologías Industriales de la ETSIT de la Universitat Politècnica de València, los alumnos deben trabajar sobre una tarjeta de adquisición de datos, en concreto la tarjeta "NuDAQ / NuIPC 9112 Series" [9], junto a un conjunto de componentes de hardware que se pueden conectar a esta tarjeta. Los alumnos pueden actuar sobre estos componentes modificando por tanto los valores de entrada de la tarjeta y pueden observar que, tras obtener estos valores desde un programa informático de su propia creación, este era capaz de modificar las salidas de la tarjeta, modificando el comportamiento de los dispositivos hardware conectados a ella.

Como está tarjeta y el conjunto de componentes conllevan un coste excesivo para los alumnos si desean adquirirlos de manera propia para practicar en sus casas, se llevó a cabo el desarrollo de un sistema de simulación virtual de la tarjeta hardware. Esto se llevó a cabo mediante la realización de un Proyecto Final de Carrera.

El encargado de su realización fue el alumno Miguel Carro Pellicer en su proyecto final de carrera llamado "Simulador de tarjeta de adquisición de Datos "NuDAQ / NuIPC 9112 Series." [1] siendo este presentado en el año 2007. Miguel consiguió emular el funcionamiento completo de la tarjeta hardware en un proceso software, siendo este accesible para cualquier alumno de forma sencilla desde su propia casa, sin necesidad de la dependencia de disponibilidad del laboratorio de prácticas.

Este simulador está compuesto por 3 elementos principales:

- La tarjeta virtual de adquisición de datos que simula a la NuDAQ / NuIPC 9112 Series.
- Un simulador software mediante el cual se pueden modificar las entradas y obtener las salidas de la tarjeta.
- Librería de la tarjeta, modificada para su uso con la tarjeta virtual que permite al alumno realizar programas de usuario en C.

Basándonos en la simulación de la tarjeta hardware, este proyecto surge por la misma problemática desarrollada anteriormente, el coste y complejidad de utilizar múltiples sensores de temperatura reales.

2. Medida de Temperatura

El propósito principal de este proyecto es simular el funcionamiento de una serie de sensores de temperatura, para su mejor comprensión se va a desarrollar una pequeña explicación de cada uno de los sensores utilizados.

Todo el conjunto de este tipo de sensores tiene la misma finalidad, obtener la temperatura del medio en que se encuentran inmersos, pero son utilizados en diversos tipos de aplicaciones completamente dispares, ya que se utilizan desde el campo de la alimentación hasta en el control de la seguridad de elementos de vehículos o a su vez de forma casera o en maquinaria industrial.

Estos sensores tienen una característica principal en común, son elementos eléctricos que, en base a la temperatura a la que son sometidos, generan una diferente señal eléctrica como salida sobre el circuito en que están incluidos.

2.1 NTC

Un NTC [2][3][4][5] es un sensor de tipo termistor, es decir, su resistencia varía en función de la temperatura a la que son sometidos. Hay dos tipos de termistores, los PTC y los NTC, en el que el coeficiente de temperatura es positivo y negativo respectivamente, es decir, a más temperatura mayor o menor valor tendrá su resistencia interna respectivamente.

Por lo general, se usan en un rango de temperaturas poco amplio y con un margen de error menor de un grado Celsius.

2.2 LM335

Un sensor LM335[2][3][4][6] es un sensor de bajo coste y muy sencillo de integrar, siendo la principal característica de este sensor que está calibrado a un grado, es decir, su salida es lineal, variando por cada grado de temperatura Kelvin 10mV su valor de salida.

2.3 Termopar Tipo J

Los sensores de temperatura de tipo termopar [2][3][4][7] son más difíciles de utilizar, pero permiten medir rangos de temperatura de cientos de grados Celsius, por lo que se utilizan principalmente en la industria. Utilizan la unión de dos metales y la diferencia de temperatura entre sus extremos y la unión de ambos para calcular la diferencia de temperatura entre sus dos uniones.

Hay de varios tipos dependiendo el metal con el que son construidos, por ejemplo, para las temperaturas más bajas se suelen usar los Tipo K (Cromel/Alumel), Tipo E

(Cromel/Constantán), Tipo J (Hierro/Constantán), Tipo T(Cobre/Constantán) Tipo N (Nicrosil/Nisil), mientras que, para temperaturas más altas, se suele usar los que contienen platino, que son por lo tanto más caros, como los Tipo B, R o S (Platino/Rodio en diferentes proporciones).

2.4 RTD PT100

Como el NTC, el sensor RTD [2][3][4] es un sensor resistivo, es decir, cuando varia la temperatura varia el valor de su resistencia. Los RTD utilizan diversos materiales en su construcción para adecuarse a las necesidades del rango de temperatura que se necesita medir, siendo estos principalmente materiales conductores como el Cobre, Níquel o Platino entre otros.

En este caso se va a simular el RTD de tipo PT100, que como su nombre indica, estará compuesto de Platino, con una resistencia de 100 Ohmnios a 0ºC, pudiendo alcanzar temperaturas de medida cercanas a los 1000ºC gracias al platino.

3. Objetivos

El principal objetivo de este proyecto consistirá en la simulación de una serie de sensores de temperatura de los cuales se dispone en el laboratorio de prácticas, pero no son de simple obtención para los alumnos en sus domicilios para continuar con la practica sobre los mismos, además de la dificultad de obtener temperaturas específicas sobre los componentes hardware sensibles a la temperatura y ver los resultados.

Se partirá sobre dos componentes de los ya desarrollados, la tarjeta virtual y su librería. Se creará un proceso virtual o simulador nuevo, el cual se comunicará con la tarjeta virtual permitiendo el envío de datos, siendo el valor de los mismos generado por los alumnos desde la interfaz gráfica del propio proceso.

Por lo tanto, desde este proceso virtual, se podrán modificar los estados de estos sensores y desde un punto de medición establecido en cada sensor, se enviará a la tarjeta virtual el valor en voltios correspondiente a la temperatura de cada sensor implementado.

Para comprobar el correcto funcionamiento de este simulador y su interactuación con la tarjeta virtual, se proporcionará un programa que utilizando la librería de la tarjeta lea el valor de las entradas analógicas y muestre la temperatura de cada sensor. Este programa posteriormente deberá ser implementado por los alumnos de prácticas.

El funcionamiento del simulador consistirá en que el alumno establezca una temperatura a la cual serán sometidos los sensores térmicos, además de algunas de las características de los mismos, siendo posible así comprobar que los valores introducidos se corresponden a los valores obtenidos como resultado en el cálculo teórico de estos sensores.

4. Análisis de requisitos

En este apartado se tratará el propósito del proyecto, desde una descripción general hasta el análisis de cada una de sus funcionalidades.

Para ello, se utilizará como base el estándar IEEE 830-1998 de Especificación de Requisitos de Software [8]. En dicho documento se detalla claramente la forma en la que debe estar organizado un correcto análisis de requisitos para cualquier tipo de proyecto, adaptándolo para las necesidades concretas de cada aplicación como se hará en este caso.

4.1 Introducción

Este apartado servirá como una introducción sobre el objetivo del sistema, siguiendo como ya hemos indicado el formato del estándar IEEE 830-1998.

4.1.1 Propósito

El propósito general de este apartado es el de proporcionar una definición de las restricciones y funcionalidades principales de la aplicación a desarrollar para el lector de el documento actual.

4.1.2 Ámbito del sistema

Como el objetivo de este proyecto no es crear un sistema que sea comercializable, sino un complemento para los alumnos, no es del todo necesario la denominación del mismo, pero por ejemplo lo podremos llamar como el título de este proyecto, “Simulador de sensores de temperatura”.

El ámbito principal del sistema será la implementación de una aplicación que actué como simulador de una serie de sensores de temperatura. Esta simulación consistirá en la visualización de la variación de los parámetros de estos sensores cuando se modifica la temperatura a la que están sometidos. El usuario podrá interaccionar mediante la interfaz de la aplicación para modificar tanto como los parámetros base de los sensores simulados como la temperatura ambiente a la que se van a encontrar estos sensores.

Con esto en cuenta, el principal propósito u objetivo de la aplicación es el de permitir al usuario comprobar sencillamente y de manera visual si el resultado de sus cálculos es correcto. Añadiendo, además, mediante la simulación de la tarjeta virtual “NuDAQ / NuIPC 9112 Series” y su correcta comunicación, la posibilidad de que el alumno mediante el desarrollo de un programa capaz de comunicar con esta misma tarjeta pueda obtener los valores y utilizarlos.

4.1.3 Definiciones, Acrónimos y Abreviaturas

- **ERS**: Especificación de requisitos del sistema.
- **Tarjeta Virtual**: Tarjeta "NuDAQ / NuIPC 9112 Series" simulada virtualmente.
- **Windows Named Pipes (Pipes):** Sistema Cliente/Servidor que proporciona una comunicación entre distintos procesos en la misma máquina o en la misma red.
- **C#:** Lenguaje de programación.
- **Usuario:** El individuo que interactúe con la aplicación, en la mayoría de las veces se referirá al alumno.
- **NTC:** Sensor de temperatura de tipo Termistor con coeficiente de temperatura negativo.
- **LM335:** Sensor de temperatura de bajo coste.
- **Termopar:** Sensor de Temperatura basado en la diferencia de temperatura entre sus puntos.
- **RTD:** Sensor de temperatura con gran margen de temperatura.

4.1.4 Referencias

- IEEE 830-1998 de Especificación de Requisitos de Software según el estándar IEEE Std.830-1998, 22 de octubre de 2008.

4.1.5 Visión General del Documento

Como se pude apreciar, una ERS está dividida en 3 partes principales, la primera o actual, que es la encargada de proporcionar una visión global de la especificación, otra que es la encargada de la explicación o descripción de la funcionalidad a implementar y una final, donde se definen los requisitos que se deben cumplir para poder decir que tiene el funcionamiento correcto.

4.2 Descripción General

En esta sección se proporcionará un contexto de los factores que afectan al proyecto a desarrollar.

4.2.1 Perspectiva del Producto

El principal elemento sobre el que se basa la estructura de este proyecto es la Tarjeta Virtual, a partir de ella se efectuarán las comunicaciones entre los distintos componentes, podemos ver una representación gráfica del esquema de comunicaciones en la Imagen 1.

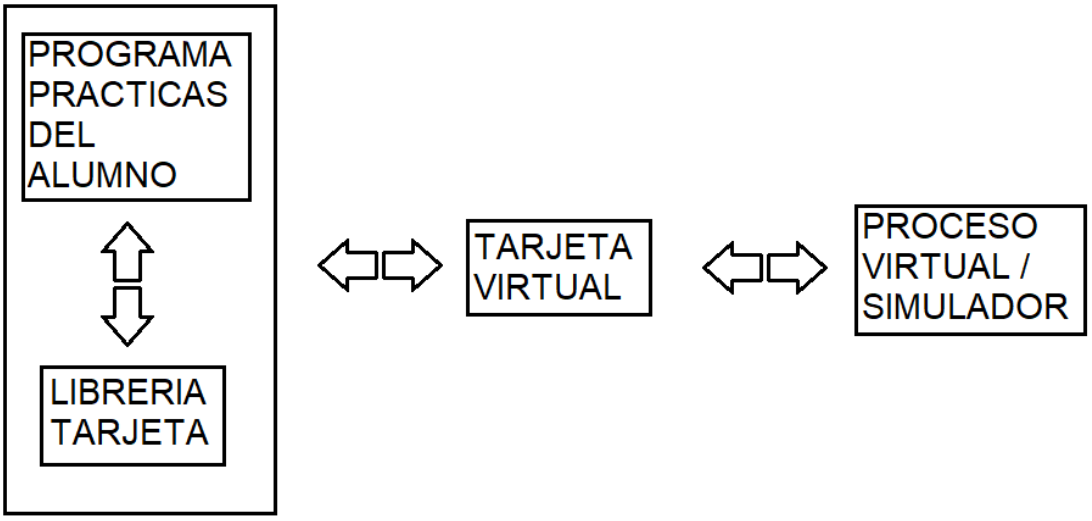

Imagen 1: Esquema estructura

Desde el simulador, que es la parte que se va a implementar en este proyecto, el alumno o usuario será el responsable de la interacción con los componentes visuales, los cuales a su vez desencadenarán una serie de comunicaciones de entrada a la tarjeta virtual, almacenando los valores correctos para el estado del conjunto de componentes visuales en ese momento.

4.2.2 Funciones del Producto

En este apartado se va a indicar las funciones necesarias que debe tener este simulador para que se pueda decir que se ha implementado correctamente.

- Se realizará una comunicación con la tarjeta virtual para la entrada de datos, pudiendo así realizar una actualización sobre los mismos.
- Se realizará una recreación de una serie de sensores de temperatura, simulando su inclusión en un circuito eléctrico sometido a una determinada temperatura ambiente y el comportamiento de los mismos ante esa temperatura.
- Se realizará una recreación de un selector de temperatura, simulando la temperatura ambiente a la que se encontrarán sometidos los sensores.
- Se debe contar con un usuario que interaccione con el sistema, siendo este el encargado de proporcionar los valores variables de los elementos del simulador.
- Se realizará una comprobación del correcto funcionamiento del sistema.

4.2.3 Características de los Usuarios

Por norma general, y por tanto así se supondrá en este documento, el usuario que interactúa con la aplicación serán los alumnos que utilicen la tarjeta física en las prácticas de la asignatura. Estos serán los encargados de seleccionar los parámetros del simulador, al mismo tiempo serán los que obtendrán los beneficios del mismo, por lo

que deberán tener conocimientos previos sobre los sensores de temperatura que se simulan además de la capacidad de crear programas sencillos que comuniquen con la tarjeta virtual para obtener los datos.

4.2.4 Restricciones

Como restricciones principales en el desarrollo de la aplicación se pueden enumerar las siguientes:

- Se debe respetar el sistema de comunicación que utiliza la tarjeta virtual para la comunicación entre los distintos procesos.
- El Simulador debe ser el encargado de abrir y cerrar el proceso de la tarjeta virtual.
- Se debe proporcionar información técnica sobre los sensores de temperatura incluidos en el simulador.
- La aplicación debe visualizarse en todo tipo de pantallas.
- Será necesaria una ejecución sobre una máquina con un S.O. Windows.

4.3 Requisitos Específicos

En este apartado se especificarán los requisitos de una forma más detallada, entrando más en profundidad en cada uno y dividiéndolos por su tipo.

4.3.1 Interfaces Externas

Número de requisito	1
Nombre del requisito	Visualización y redimensión de la ventana de aplicación.
Descripción del requisito	La aplicación deberá poder ser re-escalable para poder visualizarse en todos los tipos de pantalla de forma correcta

4.3.2 Funciones o Funcionalidad

Número de requisito	2
Nombre del requisito	Ejecutar y Cerrar la Tarjeta Virtual.
Descripción del requisito	Cuando se inicie la aplicación se deberá ejecutar la tarjeta virtual como un proceso paralelo, debiendo cerrarse este proceso también cuando se finalice la ejecución de la aplicación

Número de requisito	3
Nombre del requisito	Comunicación con la Tarjeta Virtual
Descripción del requisito	Se utilizará un sistema basado en los Named Pipes de Windows para la creación de un canal de conexión desde la aplicación hacia la tarjeta virtual y viceversa.

Número de requisito	4
Nombre del requisito	Visualización de temperatura ambiente
Descripción del requisito	Se visualizará la temperatura ambiente a la que serán sometidos la serie de sensores que se incluyen dentro de la aplicación.

Número de requisito	5
Nombre del requisito	Modificación de temperatura ambiente
Descripción del requisito	Se modificará la temperatura ambiente a la que serán sometidos la serie de sensores que se incluyen dentro de la aplicación.

Número de requisito	6
Nombre del requisito	Obtención tensión de salida en circuito de instrumentación con sensor NTC
Descripción del requisito	Se obtendrá mediante los correspondientes cálculos la tensión de salida del circuito de instrumentación que contenga el sensor de temperatura NTC en base a la temperatura a la que está sometido.

Número de requisito	7
Nombre del requisito	Modificación valor resistencia auxiliar en circuito NTC
Descripción del requisito	Se modificará el valor de la resistencia auxiliar en el circuito de instrumentación que contiene el sensor NTC.

Número de requisito	8
Nombre del requisito	Obtención valor de la resistencia interna del sensor NTC
Descripción del requisito	Se obtendrá mediante los correspondientes cálculos el valor de la resistencia del sensor de temperatura NTC en base a la temperatura a la que está sometido.

Número de requisito	9
Nombre del requisito	Obtención tensión de salida en circuito de instrumentación con sensor LM335
Descripción del requisito	Se obtendrá mediante los correspondientes cálculos la tensión de salida del circuito de instrumentación que contenga el sensor de temperatura LM335 en base a la temperatura a la que está sometido.

Número de requisito	10
Nombre del requisito	Obtención tensión de salida en circuito de instrumentación con sensor Termopar de Tipo J
Descripción del requisito	Se obtendrá mediante los correspondientes cálculos la tensión de salida del circuito de instrumentación que contenga el sensor de temperatura Termopar de Tipo J en base a la temperatura a la que está sometido.

Número de requisito	11
Nombre del requisito	Obtención tensión de salida en circuito de instrumentación con sensor RTD PT100
Descripción del requisito	Se obtendrá mediante los correspondientes cálculos la tensión de salida del circuito de instrumentación que contenga el sensor de temperatura RTD PT100 en base a la temperatura a la que está sometido.

Número de requisito	12
Nombre del requisito	Modificación valor resistencias auxiliares en circuito RTD PT100
Descripción del requisito	Se modificará el valor de las resistencias auxiliares en el circuito de instrumentación que contiene el sensor RTD PT100.

Número de requisito	13
Nombre del requisito	Obtención valor de la resistencia interna del sensor RTD PT100
Descripción del requisito	Se obtendrá mediante los correspondientes cálculos el valor de la resistencia del sensor de temperatura RTD PT100 en base a la temperatura a la que está sometido.

4.3.3 Requisitos de Rendimiento

Número de requisito	14
Nombre del requisito	Comunicación instantánea
Descripción del requisito	La aplicación se comunicará con la tarjeta virtual de forma instantánea, sin tiempos de espera.

Número de requisito	15
Nombre del requisito	Comunicación sin cortes
Descripción del requisito	La aplicación debe poder comunicarse y transmitir datos sin error, no deben producirse cortes en las comunicaciones y que se pierdan datos en el proceso.

4.3.4 Atributos del Sistema

Número de requisito	16
Nombre del requisito	Autentificación no necesaria
Descripción del requisito	No se debe pedir autentificación a los usuarios, se debe suponer que con haber obtenido el fichero ejecutable de la aplicación ya están autorizados a utilizarla.

Número de requisito	17
Nombre del requisito	Portabilidad
Descripción del requisito	La aplicación debe ser lo más sencillamente portable posible para aumentar la comodidad del usuario para su instalación en distintas maquinas.

Número de requisito	18
Nombre del requisito	Mantenibilidad o Interacción con el usuario
Descripción del requisito	La aplicación debe permitir al usuario que modifique elementos base de la aplicación (temperatura, valores de los sensores…) para que los sensores se adecuen a sus necesidades.

5. Casos de uso

Mediante los casos de uso se puede mostrar la funcionalidad de la aplicación desde un punto de vista externo a la aplicación, es decir, los casos de uso contendrán toda la serie de acciones que los usuarios de la aplicación podrán llevar a cabo, sin entrar en su diseño o implementación.

Los usuarios que pueden interactuar con los casos de uso serán denominados actores pudiendo ser estos de varios tipos. Para este proyecto como no está regido por diferentes permisos solo habrá un actor, el usuario o el alumno que utiliza la aplicación, Este podrá realizar 3 acciones, modificar la temperatura ambiente a la que están sometidos los sensores, modificar las resistencias variables en el circuito de instrumentación del sensor de temperatura NTC y modificar las resistencias variables en el circuito de instrumentación del sensor de temperatura RTD PT100.

La forma más común de expresar lo que se acaba de describir es mediante un diagrama de casos de uso modelado mediante UML, obteniendo el diagrama resultante visible en la Imagen 2.

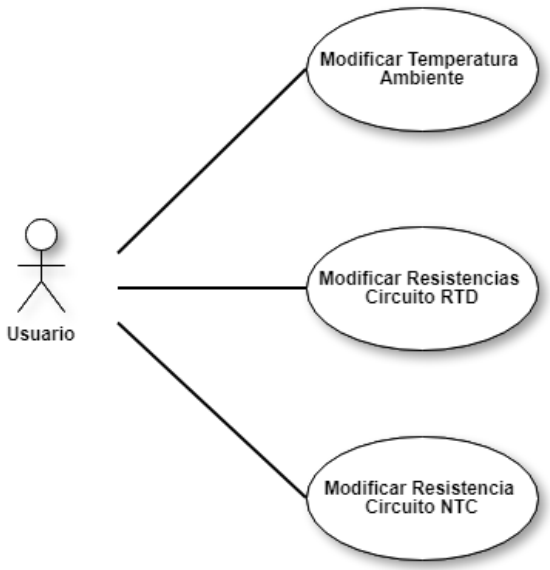

Imagen 2: Diagrama casos de uso

6. Diseño del proyecto

Este apartado está enfocado en el proceso de diseño del proyecto o aplicación, dividiéndose este en otros tres apartados secundarios: la integración y comunicación con la tarjeta virtual, la estructuración de los canales a utilizar por cada sensor y por último el diseño de la interfaz y del funcionamiento del sistema.

6.1 Comunicación con la Tarjeta Virtual

Como la aplicación está tomando como base la Librería y Tarjeta Virtual que desarrolló en su proyecto final de carrera Miguel Carro, se va a utilizar el mismo sistema de comunicación que utilizó él en la creación de dichos componentes: los Named Pipes de Windows.

Para ello, lo más sencillo tanto por facilidad y comodidad, como por la posibilidad de tener una mejor compatibilidad con este sistema de comunicación de procesos, se va a utilizar tecnologías Windows para el desarrollo de la aplicación.

Como nos indica Miguel Carro en la memoria del proyecto de la tarjeta virtual, la trama será una simple cadena de caracteres separando sus parámetros por el carácter de control "#", teniendo como máximo 20 caracteres en total. La estructura que se utiliza para las tramas de comunicación es la siguiente.

"[CODIGO_OPERACION # DIRECCIÓN o CANAL # VALOR]"

Se pueden definir los valores de la siguiente manera:

- El **CODIGO_OPERACION** especifica el código de la función de la librería que ha enviado dicha trama, pueden ser los siguientes:
  - **DO**
    - Una trama que comience con este código de operación, será emitida desde la tarjeta virtual y recibida por la aplicación o proceso virtual, conteniendo en ella el estado de las salidas digitales de la tarjeta.
  - **AO**
    - Una trama que comience con este código de operación, será emitida desde la tarjeta virtual y recibida por la aplicación o proceso virtual, conteniendo en ella el valor de las salidas analógicas de la tarjeta.

- **XZ**
  - Esta trama será emitida desde la aplicación o proceso virtual y recibida por la tarjeta virtual, le indica a la tarjeta virtual que el valor digital se ha actualizado, por lo que debe actualizarlo.

- **XX**
  - Esta trama será emitida desde la aplicación o proceso virtual y recibida por la tarjeta virtual, le indica a la tarjeta virtual que el valor analógico se ha actualizado, por lo que debe actualizarlo.

- El campo **DIRECCIÓN o CANAL** especifica el puerto al que se hace referencia si estamos en una operación digital, mientras que indicará el canal si se trata de una entrada/salida analógica. Estos valores pueden ir del 0 al 16.

- El campo **VALOR** es donde se almacenan los datos de la trama. Si se trata de una trama digital, el campo valor estará compuesto por un número entero, mientras que, si se trata de una trama analógica, se tratará de un numero en formato decimal o real.

  Además del formato de las tramas, para poder realizar la conexión con la tarjeta virtual, se tendrán que utilizar los anteriormente nombrados Pipes ya definidos en la librería virtual, se utilizarán dos en concreto, uno para las operaciones de salida y otro para las de entrada

  .

- **Tubolib**: Este tubo o pipe es el encargado de la comunicación entre la tarjeta virtual y la librería, por lo que no nos afectará en el desarrollo de la aplicación.

- **Tubotest**: Este tubo o pipe se encargará de procesar las operaciones de salida de la tarjeta virtual, es decir, las de lectura, siendo de una sola dirección.

- **Tubotest2**: Este tubo o pipe se encargará de procesar las operaciones de entrada sobre la tarjeta virtual, es decir, las de escritura, siendo este tubo también de carácter unidireccional.

El motivo principal de la utilización de dos tubos distintos, uno para escritura y otro para lectura es para evitar un posible bloqueo en la comunicación entre la tarjeta virtual y la aplicación, ya que si se realiza al mismo tiempo una llamada de entrada y otra de salida es posible que se produzca un exceso de información simultánea, se exceda el tamaño máximo del buffer y la tarjeta virtual quede bloqueada. Podemos ver cómo sería la estructura principal de comunicación entre los elementos en la Imagen 3.

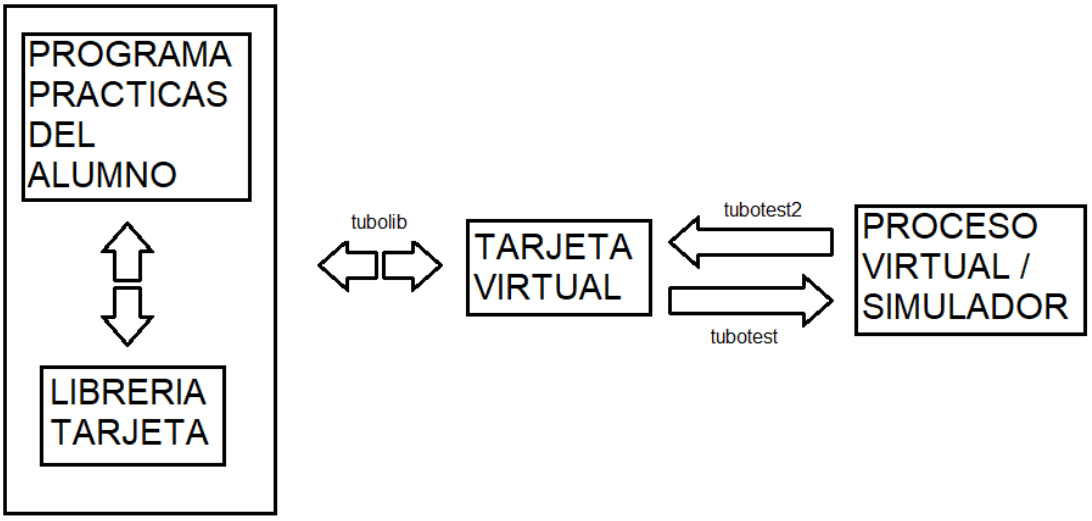

Imagen 3: Esquema comunicaciones

6.2 Canales de comunicación

Para obtener la información de cada sensor, se va a tomar un valor de salida por cada uno, es decir, se escribirá un voltaje de salida proporcionado por la medida en un punto de cada sensor implementado en la aplicación.

Un sensor no debe sobrescribir los datos emitidos por otro, deben ser datos completamente independientes, para ello, es necesario asociar cada sensor a un canal de entrada distinto de la tarjeta virtual. Para posibilitar al alumno la comprobación de estos valores, mediante un programa que se conecte a la tarjeta virtual, los sensores y sus canales asociados deben ser los mismos para cada ejecución de la aplicación.

Una vez definido el canal que utilizará cada sensor, se tienen dos posibilidades a la hora de enviar los datos, las variables de tipo analógico y digital. Como el objetivo es enviar un valor decimal de la medición del voltaje en un determinado punto del sensor, este valor se enviará siempre como una variable analógica, dejando la variable digital de ese canal para que actué como un campo de carácter Booleano indicando si ese sensor se encuentra en una situación de activo y está emitiendo un valor a la tarjeta virtual o por el contrario está inactivo y el valor almacenado no es válido. Este campo de situación solo es posible generarlo ya que se trata de una simulación en la cual se saben las temperaturas mínimas y máximas soportadas por cada sensor proporcionadas por el fabricante y la temperatura ambiente a la que se encuentran sometidos, por lo solo servirá a modo de ayuda, siendo imposible su cálculo fuera de la simulación.

Podemos establecer entonces, que los canales de comunicación para cada sensor sean los siguientes:

- **Sensor NTC**: Canal 1 de la tarjeta virtual.
- **Sensor LM335**: Canal 2 de la tarjeta virtual.
- **Sensor Termopar de Tipo J**: Canal 3 de la tarjeta virtual.
- **Sensor RTD de tipo PT100**: Canal 8 de la tarjeta virtual.

Estos canales, como ya se ha dicho anteriormente, serán los mismos tanto para el campo analógico como para el digital.

6.3 Interfaz y funcionalidad del sistema

6.3.1 Interfaz

Como bien se ha dicho en varias ocasiones, la aplicación simulará sensores de temperatura, por lo que los elementos principales del diseño serán tanto esos sensores, como el selector de temperatura.

- **Selector de temperatura**

El selector de temperatura estará accionado por parte del usuario o alumno, es decir, solo el podrá modificar su valor que deberá ser el mismo que actué para todos los sensores. Como todos los sensores tienen un pequeño margen de error, este valor de temperatura será redondeado a un valor entero para su mayor comodidad tanto en la visualización como en la interacción con él.

Teniendo esto en cuenta, lo más óptimo es que se sitúe en la parte superior de la aplicación y ofrezca distintas posibilidades para la modificación de su valor. Se puede apreciar un boceto inicial en la Imagen 4.

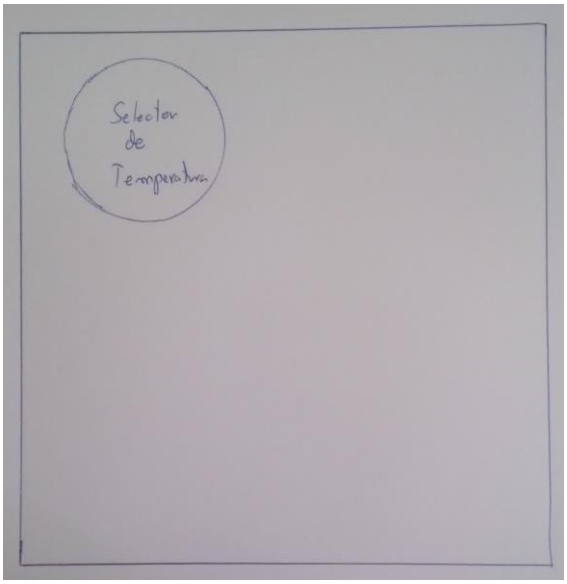

Imagen 4: Boceto con selector de temperatura

- **Sensores de Temperatura**

Los sensores de temperatura es la parte principal del proyecto, cada uno de ellos estará integrado en un tipo de circuito de instrumentación distinto, por lo que su diseño no será el mismo, ya que no todos se pueden utilizar de la misma forma, pero podemos suponer que todos ellos tendrán unos elementos en común, el sensor de temperatura y el punto de medida, de donde saldrá el valor de la tensión que se enviará a la tarjeta virtual.

Se puede ver un boceto inicial en la Imagen 5:

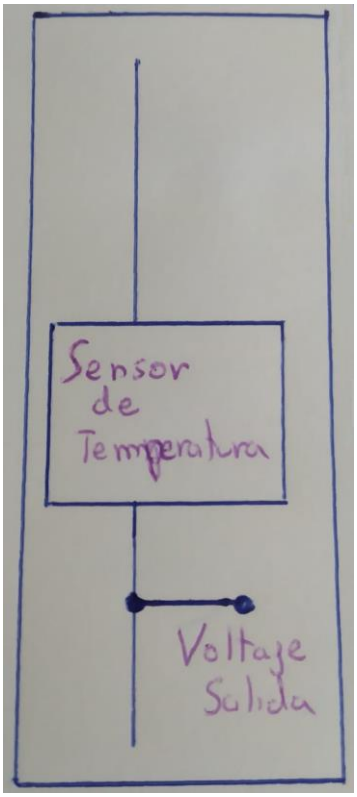

Imagen 5: Boceto sensor de temperatura

Una vez que se saben los elementos visuales a incluir en el sistema, hay dos formas de hacerlo, introduciendo todos los sensores a desarrollar en la misma pantalla, o tener una pantalla independiente para cada uno de ellos. Tras estudiarlo, se decide que lo óptimo para esta aplicación será alinear todos los sensores incluidos en el sistema en la misma pantalla, para poder ver como el cambio de temperatura modifica de forma distinta la salida de cada uno de ellos de un solo vistazo.

En base a esta decisión se podrá definir un boceto final sobre cómo deben estar organizados los elementos en la interfaz de la aplicación, este boceto se puede ver en la Imagen 6.

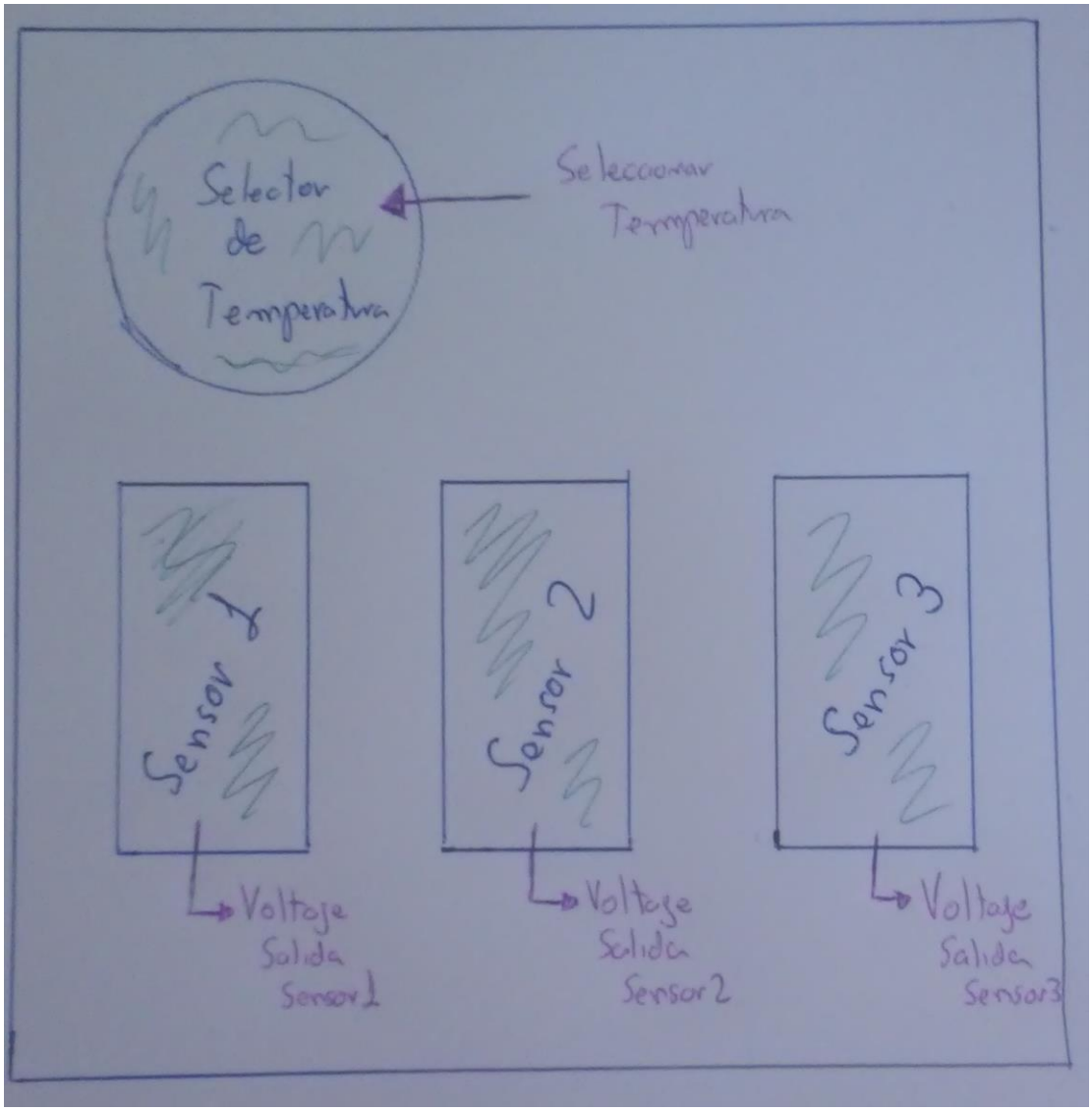

Imagen 6: Boceto final aplicación

6.3.2 Funcionalidad

La aplicación debe tener dos funcionalidades principales, ambas incluidas dentro del marco académico de la asignatura. Servirá tanto para la comprobación de resultados de problemas teóricos como para la integración de los sensores en algoritmos creados por los alumnos.

Es decir, supongamos que un alumno tiene un problema teórico en el que debe resolver a que temperatura ambiente se encuentra un sensor NTC cuando sabe el valor tanto de la tensión de salida como de los componentes variables del circuito, el valor de la resistencia del sensor, por ejemplo. El alumno resolverá este problema teóricamente, pero una vez resuelto, deberá venir al simulador, configurar el sensor con los parámetros adecuados y la temperatura ambiente con la obtenida teóricamente y

comprobar que el voltaje de salida que se escribirá en la tarjeta virtual es el mismo que se le proporcionaba en el enunciado del problema teórico.

Por otro lado, supongamos que el alumno quiere integrar el sensor en su propio algoritmo, configurando la temperatura ambiente, gracias a la comunicación con la tarjeta virtual, podría usar la tensión de salida del sensor en su algoritmo como activador de funciones.

Sabiendo los objetivos de utilización explicados anteriormente, se puede definir como debe ser la funcionalidad de la aplicación. Se seleccionará una temperatura ambiente, cada sensor comprobará si esta temperatura está dentro del rango soportado, si esta temperatura se encuentra fuera del rango, escribirá en la tarjeta virtual en el correspondiente canal del sensor el valor digital de desactivado, por el contrario, si la temperatura es válida, realizará los cálculos necesarios para obtener la tensión de salida del sensor, escribiendo el valor decimal de esta tensión en la variable analógica de su canal y marcando el valor digital como activado, pudiendo ser estos valores digitales 0 y 1 para indicar si esta desactivado o activado respectivamente.

7. Implementación

En este apartado se ahondará en los detalles de la aplicación implementada, haciendo un resumen tanto de las tecnologías utilizadas, como de la forma en la que se han utilizado y el motivo de ello.

7.1 Tecnologías utilizadas

En este apartado se detallarán las tecnologías utilizadas para el desarrollo de la aplicación. Todas ellas tienen versión gratuita o para estudiantes

- **Microsoft Visual Studio 2012**

Para el desarrollo de la aplicación, se ha decidido la utilización del entorno de desarrollo de Windows, Visual Studio. El principal motivo para su utilización es que al ser de Windows ya incorpora todas las librerías necesarias para poder realizar la comunicación con la tarjeta virtual, es decir, las librerías para poder conectar los Pipes de Windows.

El lenguaje de programación utilizado, por el mismo motivo que el anterior, el de tener la máxima compatibilidad posible, ha sido **C# el lenguaje de programación** de Windows para sus aplicaciones de escritorio. Creando para ello en este entorno de desarrollo un proyecto de tipo **Windows Forms** y siguiendo una estructura de desarrollo separado en dos puntos, **interfaces visuales y clases** dentro de librerías de código.

- **Adobe Photoshop**

Para la creación gráfica de los componentes de la aplicación se ha utilizado el programa editor de gráficos de Adobe, el Adobe Photoshop, ya que, para evitar posibles problemas de utilización de imágenes de terceros, se han creado completamente todos los componentes gráficos que se muestran en la aplicación excepto los logos de la universidad.

- **Borland C++ Compiler**

Para la compilación del programa de pruebas que deberá realizar tanto el alumno, como el desarrollador para la comprobación del correcto funcionamiento de la aplicación se ha utilizado Borland C++ Compiler, este se trata de un compilador muy rápido y sencillo utilizado mediante la consola de Windows.

7.2 Conexión y Comunicación con la Tarjeta Virtual

Como ya se ha comentado en el apartado anterior, la aplicación está dividida en dos componentes principales, las interfaces y las clases con el código. En base a eso se ha creado una clase llamada `TarjetaVirtual`, estando dentro de ella todas las funciones que se encargarán de la comunicación con la tarjeta virtual. En este apartado se va a proceder a la explicación de las funciones incluidas en esta clase.

Lo primero que se debe hacer es comprobar que no existe ninguna tarjeta virtual en ejecución por parte de otro programa, ya que esto llevaría a errores pudiendo no leer/actualizar en la misma tarjeta nuestra aplicación y el programa de pruebas del alumno. Para ello se ha implementado la función estática:

```
public static void MatarTarjetas()
```

Esta función se encargará de buscar todos los procesos que contengan el nombre "TarjetaVirtual" y ejecutar sobre ellos la función Kill de Windows, es decir, cerrarlos y gracias a su declaración como una función estática no será necesario que tengamos inicializada nuestra tarjeta virtual para utilizarla.

Una vez se hayan cerrado otras conexiones que puedan afectar a la simulación, crearemos una nueva, para ello, ahora sí, inicializaremos la clase `TarjetaVirtual`, al inicializarla se llevaran a cabo dos funciones principales, la de iniciar la tarjeta virtual y la de la conexión de los tubos o pipes, para ello, se llamará a la función:

```
public void StartTubos()
```

Esta función será la encargada de inicializar ambos tubos de comunicación con la tarjeta virtual, "tubotest" y "tubotest2". Cuando estas funciones hayan finalizado su ejecución se podrá decir que ya tenemos establecida una conexión estable y se podrá empezar a utilizar la aplicación.

Para ello, se ha creado una función general, a la que llamaremos pasando como parámetros la información correspondiente cuando se necesite realizar una operación de escritura sobre la tarjeta.

```
public void WriteEvent(String codoperacion,int canal,Decimal valor)
```

Como se puede suponer por el nombre de los parámetros, en el parámetro codoperacion especificaremos el código de operación, el parámetro canal será el canal de la tarjeta y el parámetro valor será el campo valor. Este método se encargará de la creación de la trama en base a los parámetros recibidos, y una vez creada y formateada, será enviada mediante el evento "Write" del Pipe de Windows a la tarjeta virtual.

Con estos métodos ya podríamos tener completo el funcionamiento de la tarjeta necesario para el correcto funcionamiento de la aplicación, pero además de eso, se va a añadir una función que será llamada cuando se dispare el evento de cierre de aplicación,

esta función será la encargada de realizar la función de *Garbage Collector*, es decir, cerrar todos los posibles hilos, pipes y procesos que todavía sigan abiertos, para no dejar recursos ocupados en la maquina donde se ejecute la aplicación al cerrar la misma.

```
public void CloseConnection()
```

7.3 Sensores térmicos

En este apartado se explicará el diseño y creación de las clases utilizadas para los sensores. Como la aplicación contiene 4 sensores distintos, se crearán 4 clases, siendo estas lo más común entre ellas posible. Para la explicación de estas clases se mostrará tanto las cabeceras de las mismas, como la forma de la realización del cálculo del valor tensión a escribir en la tarjeta virtual.

7.3.1 NTC

```
public class NTC
    {
        public const int Canal = 1;             //Canal de la tarjeta virtual que
utilizará el sensor
        public const decimal MinT = -40;          //Temperatura mínima a la que
puede funcionar el sensor
        public const decimal MaxT = 150;          //Temperatura máxima a la que
puede funcionar el sensor

        public const decimal Vin = 5;           //Tensión de entrada del circuito
        public const double  RTo = 10000;       //Valor de la resistencia a To
        public const double Beta = 3977;        //Valor Beta
        public const double To = 25;              //Valor temperatura utilizado
como calibrador

        public  decimal  RFixed  {  get;  set;  }        //Valor  de  la  resistencia
personalizable por el alumno
        public decimal RVar { get; set; }       //Valor de la resistencia del NTC
        public decimal Vo { get; set; }         //Tensión de salida, enviada a la
tarjeta virtual

    }
```

Aquí podemos ver todas las variables que compondrán nuestra clase para el sensor NTC. Para el cálculo de la tensión de salida, tendremos que ejecutar dos cálculos principales. Primero se calculará el valor de la resistencia del NTC, esto se realiza mediante la siguiente formula, siendo el parámetro temperatura la temperatura ambiente a la que se encuentra el sensor, establecida por el usuario mediante el selector de temperatura.

```
    RVar    =    (decimal)(RTo    *    Math.Exp(Beta    *    ((1    /    (temperatura    +
Constantes.KelvinVar)) - (1 / (To + Constantes.KelvinVar)))));
```

Una vez calculada la resistencia del NTC, ya se podrá calcular la tensión de salida a enviar a la tarjeta.

```
    Vo = Math.Round(Vin * (RFixed / (RFixed + RVar)), 4);
```

7.3.2 LM335

```
    public class LM335
    {
        public const int Canal = 2;          //Canal de la tarjeta virtual que
utilizará el sensor
        public const decimal MinT = -40;     //Temperatura mínima a la que puede
funcionar el sensor
        public const decimal MaxT = 100;     //Temperatura máxima a la que puede
funcionar el sensor

        public decimal Vo { get; set; }      //Tensión de salida, enviada a la
tarjeta virtual
```

En el caso del LM335 no es necesario la declaración de ninguna variable auxiliar para el cálculo de la tensión de salida. En un LM335 calibrado la tensión de salida varia 10mV por grado Kelvin, por lo que su cálculo es el más sencillo de los sensores implementados, tan solo habrá que aplicar la siguiente formula, siendo el valor temperatura la temperatura ambiente a la que se encuentra el sensor:

```
        Vo = Math.Round((decimal)(((temperatura+Constantes.KelvinVar)*10) /1000),
4);
```

7.3.3 Termopar Tipo J

```
    public class Termopar
    {
        public const int Canal = 3;                  //Canal de la tarjeta
virtual que utilizará el sensor
        public const decimal MinT = 95;              //Temperatura mínima a la
que puede funcionar el sensor
        public const decimal MaxT = 760;             //Temperatura máxima a la
que puede funcionar el sensor
        public const double TempFrio = 100;          //Temperatura referencia


        public decimal Vo { get; set; }              //Tensión de salida, enviada
a la tarjeta virtual
        public double Vtunion_0 { get; set; }        //Tensión de temperatura en
union
        public double Vtunion_tfrio { get; set; }    //Diferencia de tensión
entre uniones
        public double Vtfrio_0 { get; set; }         //Tensión de temperatura
referencia
        public Dictionary<double, double> TablaTermopar { get; set; } //Tabla con
valores almacenados de tensión para cada temperatura
```

Como en los sensores anteriores, en el anterior código se pueden apreciar las variables necesarias para el cálculo de la tensión de salida del sensor. Se debe rellenar el diccionario de valores de tensión en base a la temperatura mediante la tabla de valores proporcionada por el fabricante en base a 0 grados. Con la tabla correctamente rellenada se obtendrán los valores correspondientes para la temperatura de referencia de la unión fría y la temperatura de la unión, siendo esta ultima la temperatura ambiente a la que se encuentra el sensor. Esta temperatura de referencia de la unión fría se ha establecido a

100º Celsius para añadir algo más de complejidad a la hora de la realización del cálculo. Siendo por lo tanto el cálculo de la tensión de salida realizado con la siguiente formula.

```
Vtunion_tfrio = Vtunion_0 - Vtfrio_0;

Vo = (decimal)Math.Round(Vtunion_tfrio, 4);
```

7.3.4 RTD PT100

```
public class RTD
{
    public const int Canal = 8;                 //Canal de la tarjeta virtual
que utilizará el sensor
    public const decimal MinT = -200;           //Temperatura mínima a la que
puede funcionar el sensor
    public const decimal MaxT = 950;            //Temperatura máxima a la que
puede funcionar el sensor
    public const decimal RFria = 100;           //Valor resistencia fija de
calibración
    public const decimal V = 5;                 //Tensión de entrada del
circuito
    public const double Alpha = 0.003850;       //Valor Alpha


    public decimal Vo { get; set; }             //Tensión de salida, enviada a
la tarjeta virtual
    public decimal RVar { get; set; }           //Valor de la resistencia del
RTD
    public decimal RFixed { get; set; }         //Valor resistencias
personalizables por el alumno
    public decimal U1 { get; set; }             //Tension calculada rama 1
    public decimal U2 { get; set; }             //Tension calculada rama 2
```

En el RTD, como en el caso del NTC, al ser ambos sensores de temperatura de carácter resistivo, el paso inicial para el cálculo de la tensión de salida será el cálculo del valor variable de la resistencia del sensor en base a la temperatura a la que se encuentra sometido. Esto se realiza mediante la siguiente formula.

```
RVar = RFria * (1 + (decimal)Alpha * (decimal)temperatura);
```

Una vez obtenido el valor de la resistencia variable del sensor este se puede incluir en una variedad de circuitos de instrumentación, considerando el más adecuado para el cálculo de la tensión la implementación en un puente de Wheatstone, mediante el cual se obtendrá la diferencia de tensión entre ambas ramas del puente, siendo esta diferencia el valor de la tensión de salida del sensor calculado mediante la siguiente formula.

```
U1 = V * (RFria /(RFria + RFixed));
U2 = V * (RVar /(RVar + RFixed));

Vo = Math.Round(U2 - U1,4);
```

7.4 Interfaz gráfica

Por último, vamos a ver el proceso para la implementación de la interfaz gráfica de la aplicación. Para la implementación de esta interfaz se han utilizado los *Forms* de Windows, desde ellos se pueden crear sencillamente tanto los componentes gráficos como sus eventos para la comunicación con el código de la aplicación.

Como la estructura y la disposición de los componentes ya se han explicado en el apartado de diseño, no volveremos a explicar el porqué de los componentes sino su apariencia y su funcionalidad. Empezaremos por el selector de temperatura.

7.4.1 Selector de temperatura

El selector de temperatura estará compuesto de 3 partes, el texto plano con el valor de la temperatura en grados Celsius, un selector numérico desde el cual se podrá o poner la temperatura de forma manual escribiendo el valor exacto, o modificarla con sus botones de aumentar o bajar el valor del parámetro y un selector *TrackBar*. Podemos ver estos componentes en la Imagen 7.

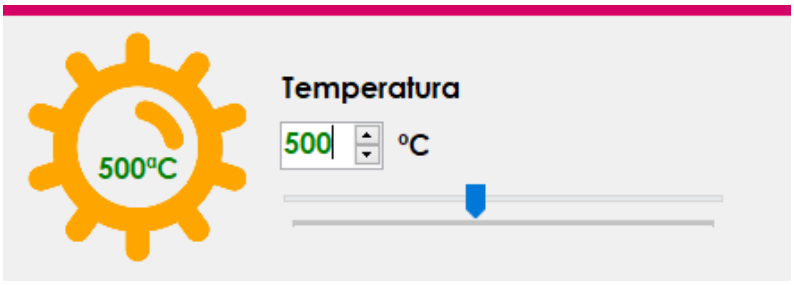

Imagen 7: Selector temperatura media

Este selector tendrá una longitud algo inferior a la mitad de la pantalla de la aplicación, situado en la parte superior izquierda de la misma como ya hemos dicho, siendo los valores mínimos y máximos seleccionables, el valor mínimo más bajo de todos los sensores y el valor máximo más alto de los sensores respectivamente.

Cuando se modifique la temperatura desde cualquiera de sus formas, se disparará un evento que hará que se actualicen los valores de salida de todos los sensores térmicos, pero, además, cuando la temperatura se fije en un valor inferior al 25% de los valores disponibles o superior al 75% de los mismos, el componente visual cambiará para indicar al usuario que es una temperatura baja o alta como se puede apreciar en las siguientes imágenes.

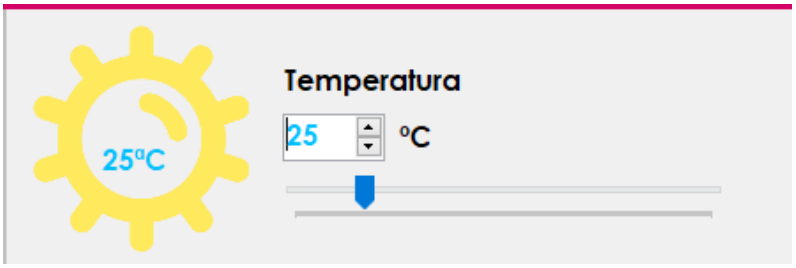

Imagen 8: Selector temperatura baja

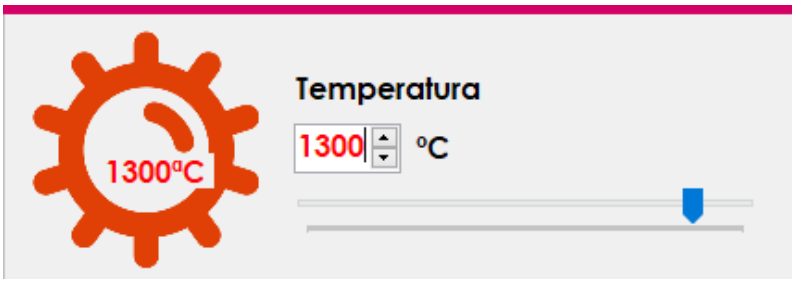

Imagen 9: Selector temperatura alta

La implementación gráfica de los sensores consistirá tanto en el diseño de los circuitos de implementación de los mismos como en la creación de una leyenda o notas con la información base de cada sensor. Estando todos estos situados en la parte central de la aplicación ocupando casi todo el tamaño de la ventana.

7.4.2 NTC

En el caso del NTC, el circuito estará formado por una tensión de entrada, el sensor NTC con el valor de su resistencia, estando este campo deshabilitado para la modificación por parte del usuario siendo un valor solo de lectura calculado en base a la temperatura ambiente a la que está sometido el sensor, una resistencia de control modificable por el usuario, la masa y el punto de medición de la tensión de salida, situado este entre ambas resistencias, obteniendo como resultado el circuito apreciable en la Imagen 10.

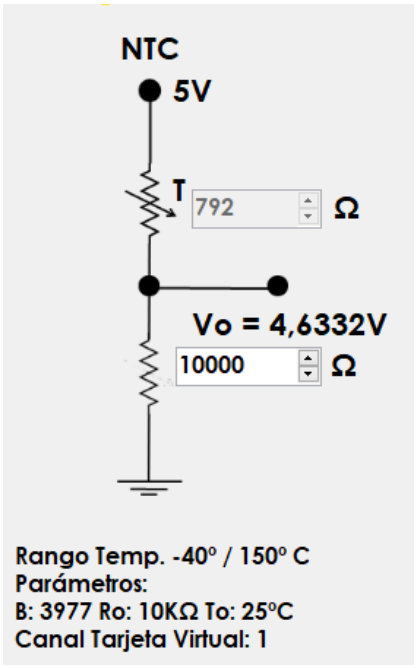

Imagen 10: Circuito NTC

Además del circuito, debajo del mismo se incluirá unos campos a modo de leyenda siendo su valor el asignado como constante en la declaración de la clase NTC por lo que estos valores serán fijos durante cualquier ejecución.

7.4.3 LM335

El circuito de instrumentación del sensor LM335 será muy similar al del NTC, estará compuesto por una tensión de entrada, una resistencia fija de control para simular un circuito más real, el diodo que identificará al sensor, la masa y el punto de medición de la tensión de salida, situado entre la resistencia de control y el sensor. Obteniendo como resultado el diseño del circuito mostrado en la Imagen 11.

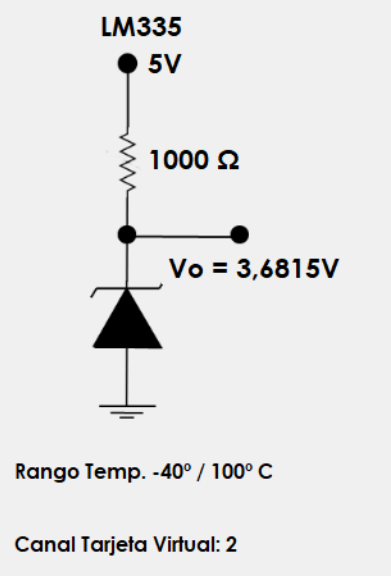

Imagen 11: Circuito LM335

Además del circuito, debajo del mismo se incluirá unos campos a modo de leyenda siendo su valor el asignado como constante en la declaración de la clase LM335 por lo que estos valores serán fijos durante cualquier ejecución.

7.4.4 Termopar Tipo J

En este caso, el diseño del circuito de instrumentación será algo diferente en comparación a los anteriores, el principal cambio será que como el Termopar actúa como un generador de tensión no será necesaria una tensión de entrada, por lo que solo contendrá el diseño del sensor, constando este de la indicación de donde se realizará la medición de la temperatura de la unión fría y donde la de la unión caliente, el punto de medición de la tensión de salida y la masa. Como en los casos anteriores se puede apreciar el resultado en la Imagen 12.

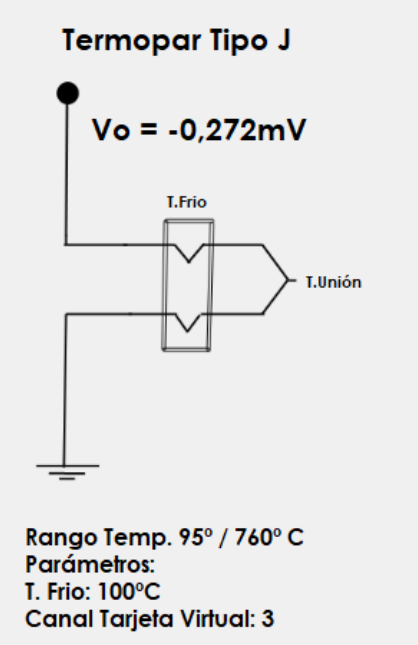

Imagen 12: Circuito Termopar

Además del circuito, debajo del mismo se incluirá unos campos a modo de leyenda siendo su valor el asignado como constante en la declaración de la clase Termopar por lo que estos valores serán fijos durante cualquier ejecución.

7.4.5 RTD PT100

Para el sensor de tipo RTD, en la creación de su circuito de instrumentación se utilizará un diseño ya establecido, conocido como puente de Wheatstone, este circuito está compuesto por dos brazos o ramas en paralelo con dos resistencias cada uno, posicionando en una de ellas el sensor de tipo resistivo RTD, teniendo en la rama contraria una resistencia con un valor fijo que actuará como resistencia de control, sirviendo las dos resistencias variables restantes para la adecuación a las distintas tensiones a las que se verá sometido. En nuestro caso como la tensión de entrada será fija estas resistencias solo servirán para aportar credibilidad al circuito. Mediante la medición de la tensión en el punto intermedio de ambas ramas y sacando su diferencia se obtendrá valor de la tensión de salida de nuestro circuito. Este circuito basado en el puente de Wheatstone se puede apreciar en la Imagen 13.

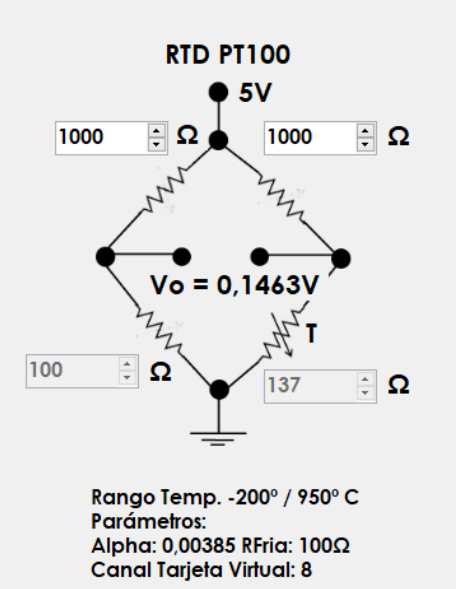

Imagen 13: Circuito RTD

Además del circuito, debajo del mismo se incluirá unos campos a modo de leyenda siendo su valor el asignado como constante en la declaración de la clase RTD por lo que estos valores serán fijos durante cualquier ejecución.

Además, como parte común a todos los sensores, en la leyenda de cada sensor como se puede apreciar en la Imagen 14, cuando un sensor se encuentre sometido a una temperatura inválida para él mismo, aparecerá una línea de texto más en color rojo, indicando que hay parámetros incorrectos y no se va a realizar el cálculo de la tensión de salida.

Imagen 14: Leyenda T. Inválida

8. Pruebas de funcionamiento

En este apartado se va a analizar el funcionamiento del simulador desarrollado, para poder evaluar el comportamiento tanto gráfico como funcionalmente. El proceso detallado de las pruebas realizadas se muestra en la Imagen 15.

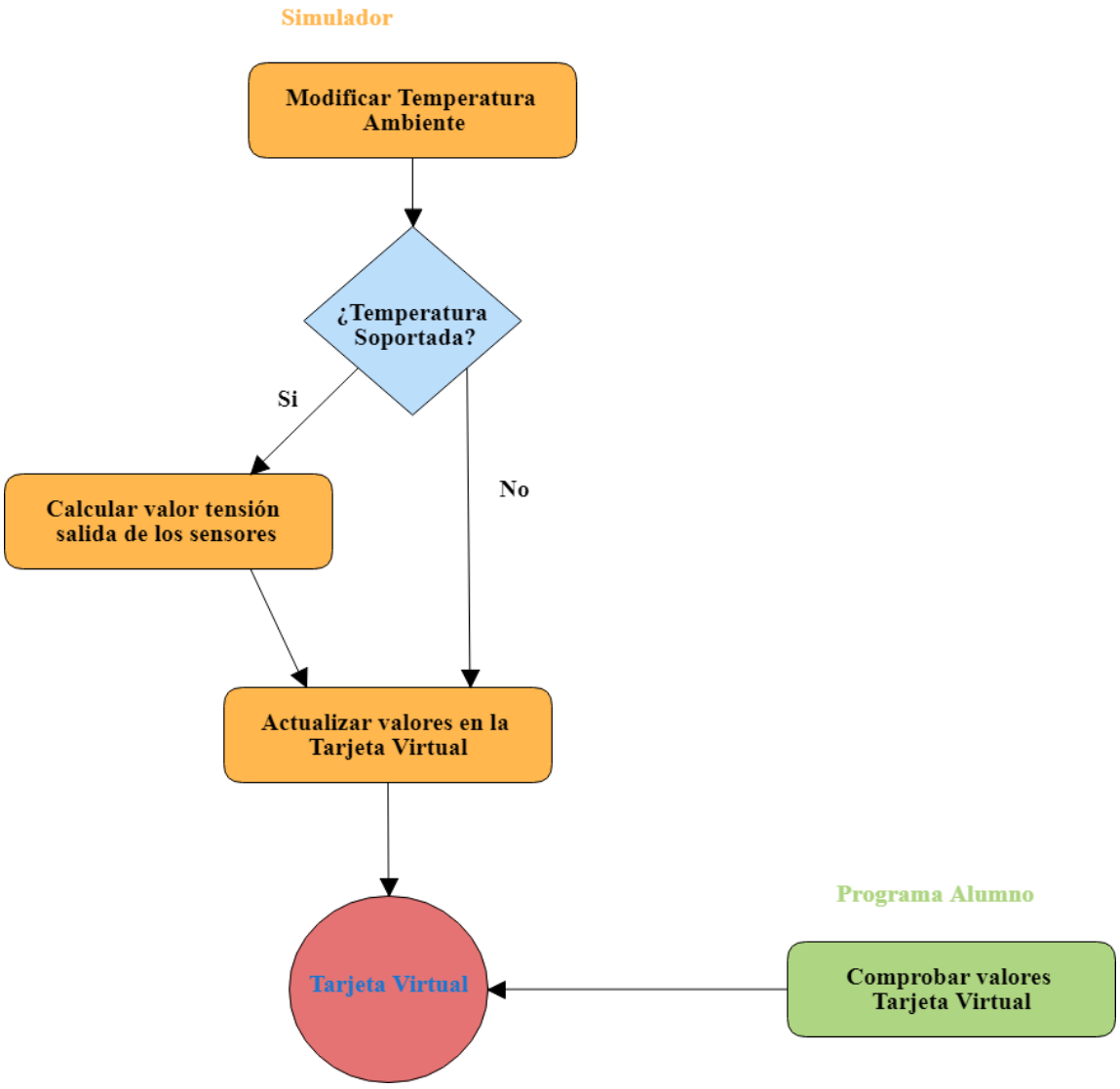

Imagen 15: Diagrama testeo funcionamiento

8.1 Carga Inicial

Cuando se inicia la aplicación, se parte con una temperatura por defecto. Se ha elegido una que entre en el rango de temperaturas válido de todos los sensores implementados, siendo esta, por ejemplo, 95 grados Celsius. Podemos ver el estado inicial de la aplicación en la Imagen 16.

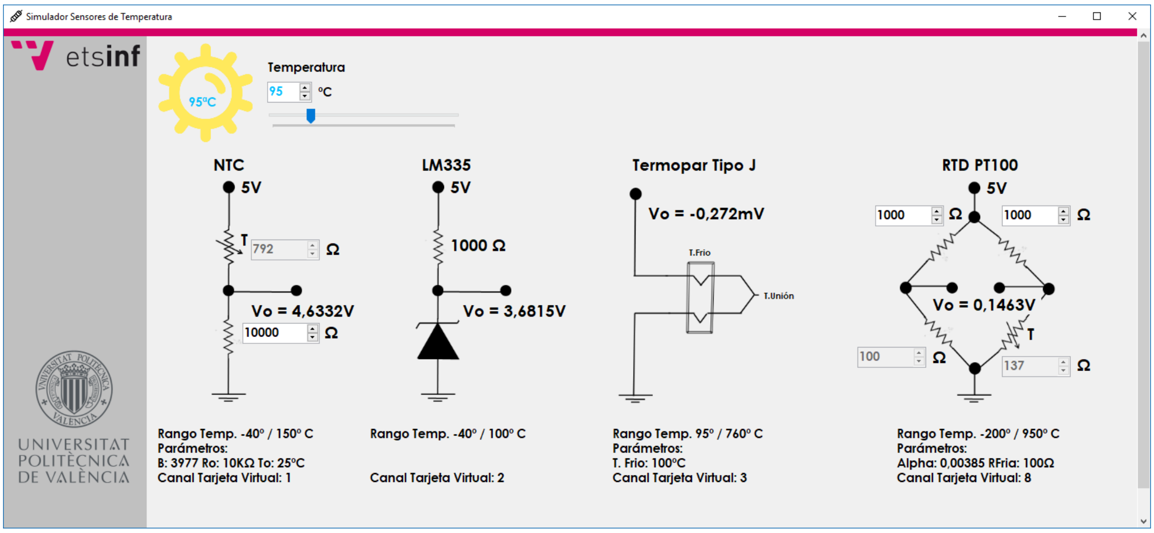

Imagen 16: Estado inicial del simulador

Para comprobar si la comunicación es correcta, se ha implementado un programa en C, como los que deberá hacer el alumno en un futuro, sirviendo este únicamente a modo de pruebas y para observar que los valores almacenados en la tarjeta se modifican correctamente. Una vez iniciado el simulador, y por lo tanto la tarjeta virtual, se podrá proceder a la ejecución de este programa de pruebas. La ejecución de este programa proporciona el resultado mostrado en la Imagen 17, donde puede verse el voltaje leído en las entradas analógicas de la tarjeta virtual.

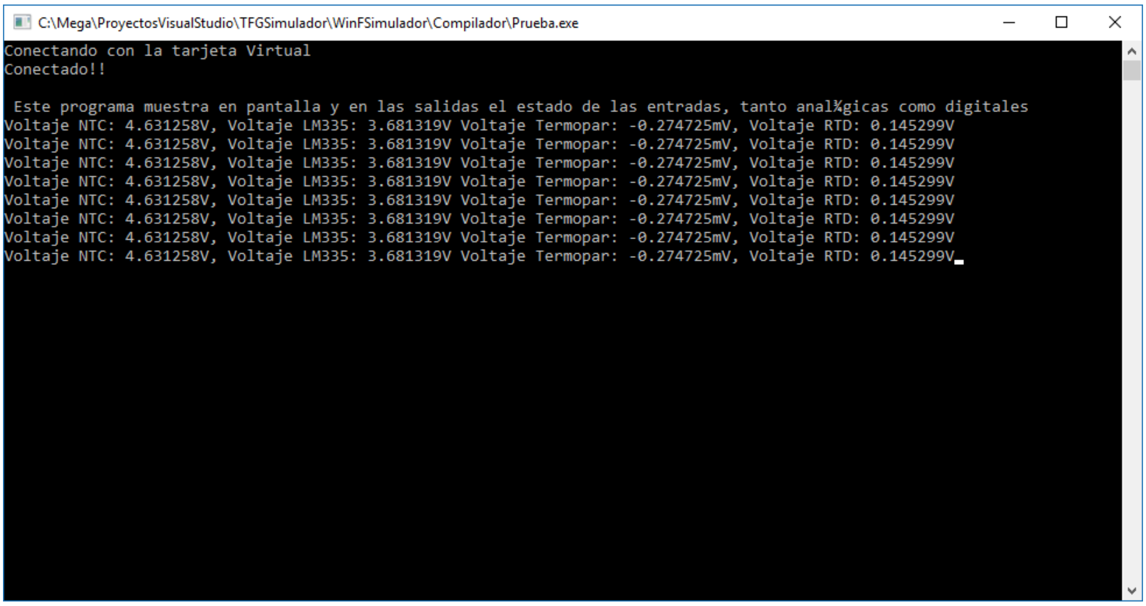

Imagen 17: Estado inicial consola

Se puede apreciar que hay un ligero margen de error por parte de la tarjeta virtual. Este error es intencionado, y simula el error de una tarjeta de adquisición de datos real.

8.2 Temperatura inválida

El paso siguiente será modificar la temperatura para ver cómo reaccionan los sensores y la tarjeta virtual ante ello, teniendo dos posibles respuestas dependiendo de si es una temperatura soportada o no. Se puede apreciar el cambio a una temperatura inválida en la Imagen 18.

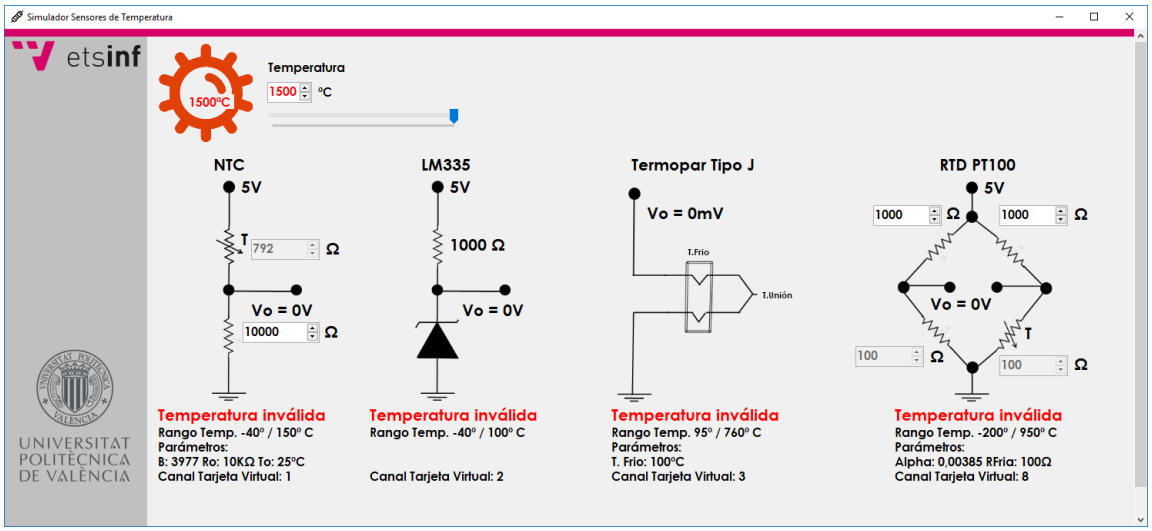

Imagen 18: Simulador temperatura inválida

Como se puede apreciar, debajo de cada sensor aparecerá la señal de aviso "Temperatura inválida", además se seleccionará como su tensión de salida el valor de 0 voltios. Como se ha visto anteriormente, se puede apreciar este estado desde la consola generada por el programa de pruebas en la Imagen 19.

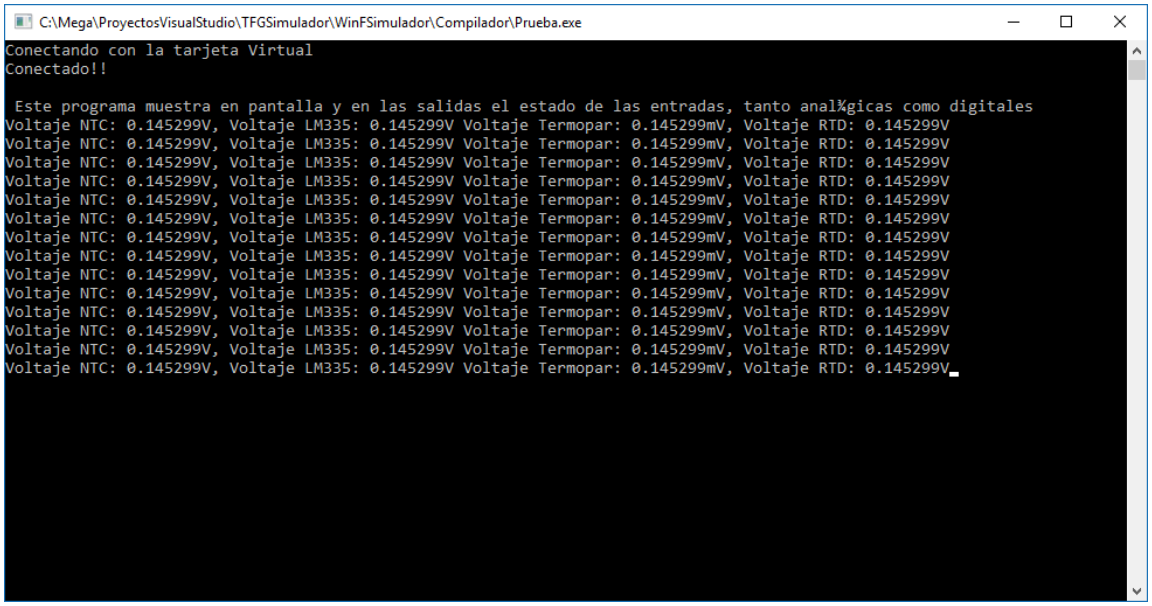

Imagen 19: Estado temperatura inválida consola

8.3 Temperatura válida

Sin embargo, cuando la modificación de temperatura es entre valores aceptados o validos la tensión de salida se calculará para cada sensor, mostrando el resultado de esos cálculos en la pantalla y enviándolos a la tarjeta virtual. Esto se puede ver en la Imagen 20.

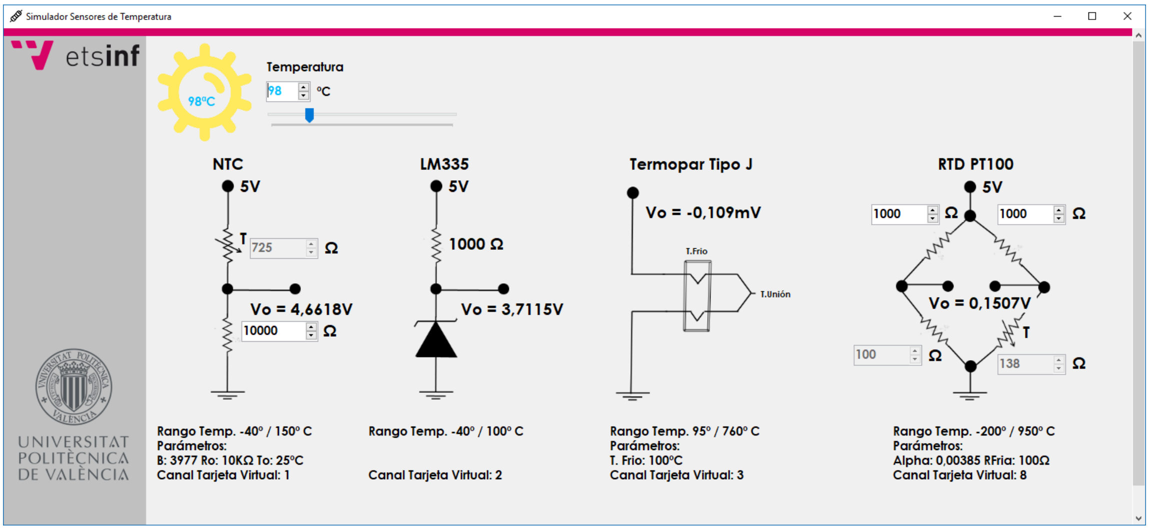

Imagen 20: Simulador temperatura válida

Si mientras se realiza este cambio, se tiene la aplicación de consola en ejecución, se puede apreciar el momento exacto en el que se han modificado los valores de las tensiones en la tarjeta virtual. Pudiendo apreciar este momento en la Imagen 21, indicado con una línea de color rojo el momento en el que sucede el cambio de temperatura y el programa lee el nuevo valor.

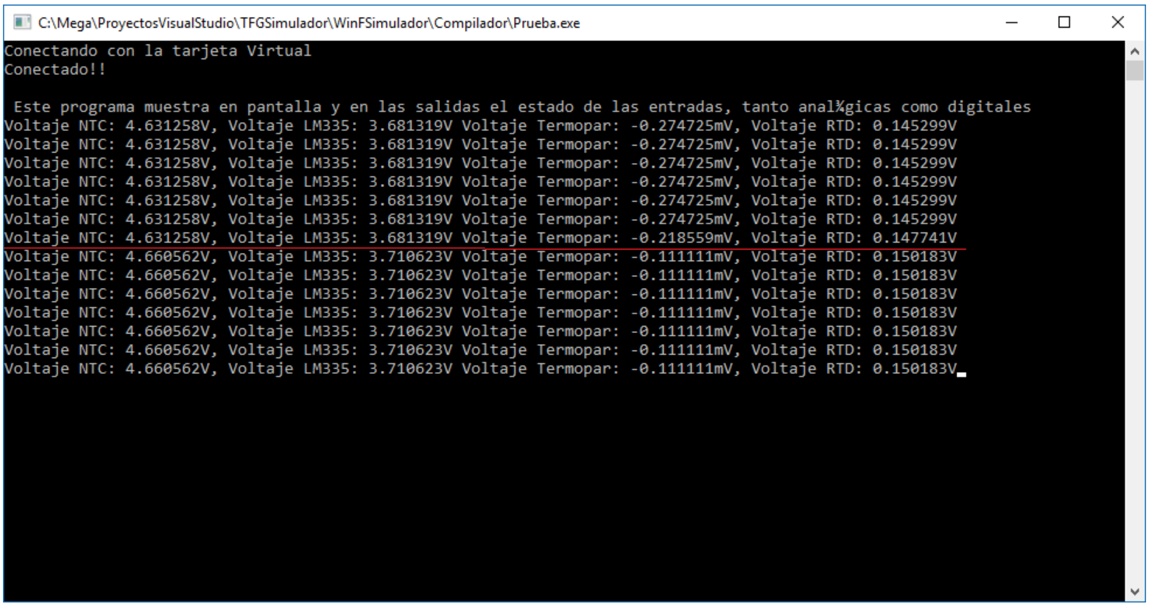

Imagen 21: Estado temperatura válida consola

Como se ha visto, las pruebas han sido correctas respecto al funcionamiento esperado por parte de la aplicación, por lo que se podría decir que se ha logrado el objetivo del proyecto.

9. Conclusiones y futuras ampliaciones

Como punto final de esta memoria se encuentra este apartado, en el cual se explicarán las conclusiones sobre el mismo, siendo estas de carácter técnico, personal o la posibilidad de futuras ampliaciones a incorporar en un futuro, pudiendo estas plantearse como una ampliación del actual proyecto o como sugerencias para la realización de un nuevo proyecto.

Los retos principales que se plantearon al inicio de este proyecto fueron tanto la correcta comunicación con la tarjeta virtual ya existente como el estudio de los sensores de temperatura más comunes y útiles para el futuro usuario de la aplicación, decidiendo cuáles serían las más adecuados, además del diseño e implementación de los circuitos de instrumentación de los mismos.

Se puede afirmar que estos retos han sido correctamente superados y que se ha logrado completar el desarrollo de una aplicación consistente, con un funcionamiento correcto y que se adecua a los requisitos establecidos al inicio del proyecto, quedando siempre esta aplicación abierta a posibles mejoras y ampliaciones en un futuro.

Una de estas posibles ampliaciones que comentamos podría ser la inclusión de una mayor cantidad de sensores o capacidad de modificación sobre los mismos, es decir, poder dar a alumno una mayor capacidad de interactuación con la aplicación pudiendo este cambiar valores base de los sensores, que en la aplicación actual están basados en las especificaciones técnicas de un fabricante concreto, pudiendo buscar otros valores técnicos validos de otro fabricante. Además, también sería una buena futura implementación la posibilidad de permitir al alumno el desarrollo de su propio circuito de instrumentación en base al sensor térmico a utilizar.

En el aspecto personal me encuentro muy satisfecho con el resultado final del trabajo, el cual me ha permitido durante su realización iniciar o mejorar mis conocimientos en los diversos temas que trata, sobre todo en el aspecto de instrumentación electrónica y sensores de temperatura, además de la posibilidad de ver de primera mano cómo sería la realización completa de una aplicación pasando por todas las fases de su desarrollo. Considerándome después de la realización del mismo, además de todo lo adquirido mediante las clases recibidas en estos años, con los conocimientos y experiencias necesarias para tener una buena base con la que salir al mundo laboral y poder afrontar cualquier desafío que me proponga.

10. Bibliografía

- [1] Miguel Carro Pellicer: **Simulador de tarjeta de adquisición de datos Nudaq/Nuipc9112 Series** [Proyecto final de carrera] ETSINF-UPV, 2007

- [2] Miguel A. Pérez García, Juan C. Álvarez Antón, Juan C. Campo Rodríguez, Fco. Javier Ferrero Martin, Gustavo J. Grillo Ortega. **Instrumentación Electrónica**. Thompson.

- [3] Ramón Pallàs-Areny, Jhon G. Webster. **Sensor and signal conditioning, second edition.**

- [4] David Arboledas Brihuega**. Electrónica Básica.**

- [5] Vishay: **Computation spreadsheet for VISHAY BC components of reference NTCC100/ NTCLE100/ NTCLE203 series** [Hoja de datos del fabricante]

- [6] Texas Instruments. **LMx35, LMx35A Precision Temperature Sensors** [Hoja de datos del fabricante]

- [7] Ise, Inc. **ITS-90 Table for Type J Thermocouple (Ref Junction 0ºC)** [Hoja de datos del fabricante]

- [8] IEEE Std. 830-1998**, Especificación de Requisitos según el estándar de IEEE 830**, 22 de octubre de 2008.

- [9] AdLink, **NuDAQ / NuIPC 9112 Series Multi-function DAS Cards for PCI / 3U CompactPCI User's Manual**

11. Anexo: Manual de Usuario

Como el objetivo de la aplicación desarrollada es de carácter docente, se ve la necesidad de la inclusión a modo de anexo en esta memoria de un manual de usuario para facilitar al alumno lo más posible la utilización de la misma.

Lo primero que se debe hacer es iniciar el programa, para ello será necesario la ejecución del fichero SimuladorSensoresTemperatura.exe (Imagen 22).

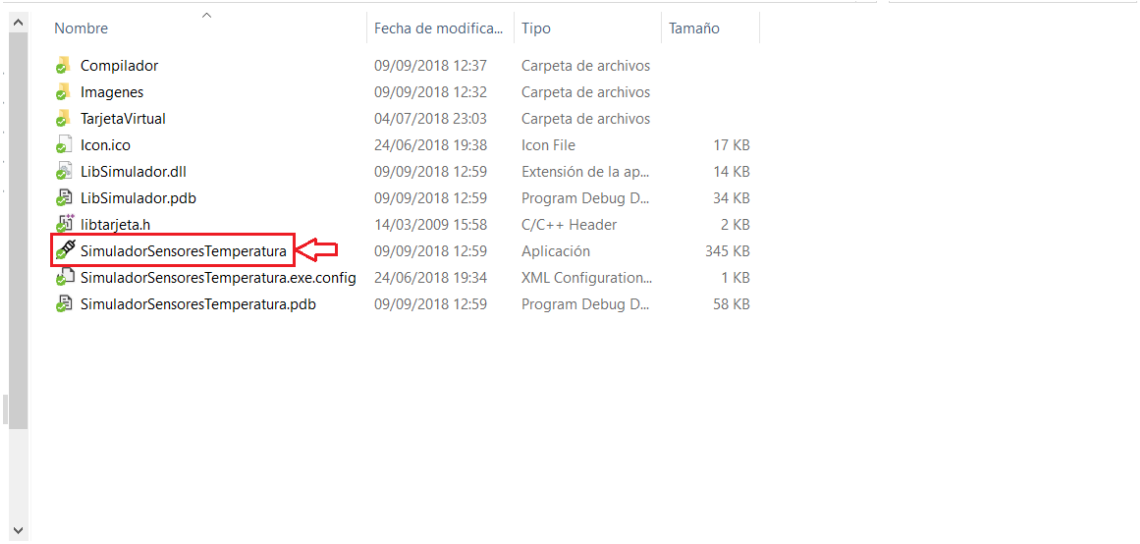

Imagen 22: Manual: Localización fichero .exe

Una vez ejecutado aparecerá la pantalla inicial de la aplicación. (Imagen 23)

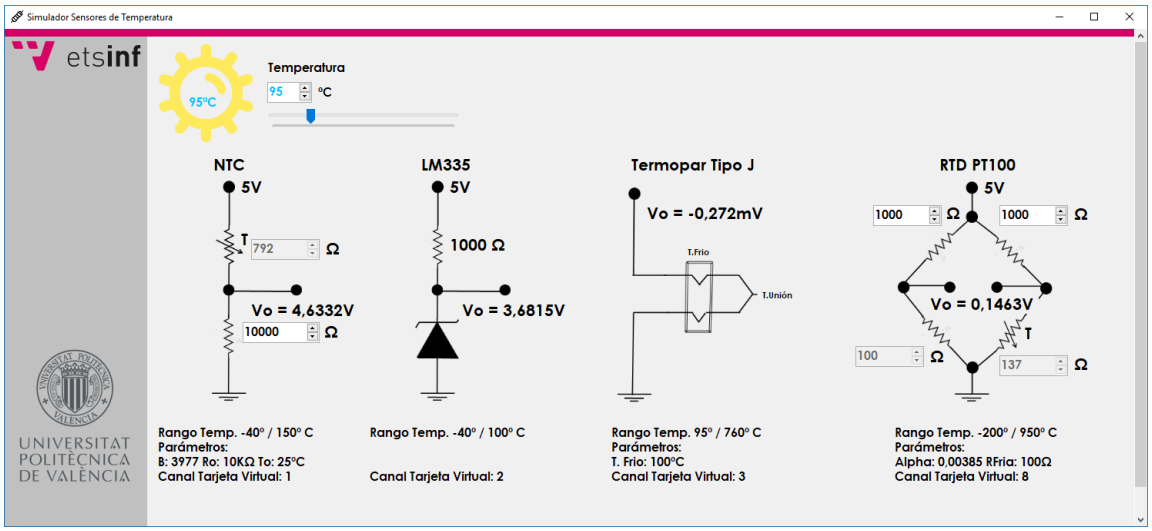

Imagen 23: Manual: ventana inicial

En esta pantalla, nos encontraremos con 3 elementos modificables, el selector de temperatura ambiente y las resistencias variables de los sensores NTC y RTD, indicadas respectivamente en las Imágenes 24 y 25.

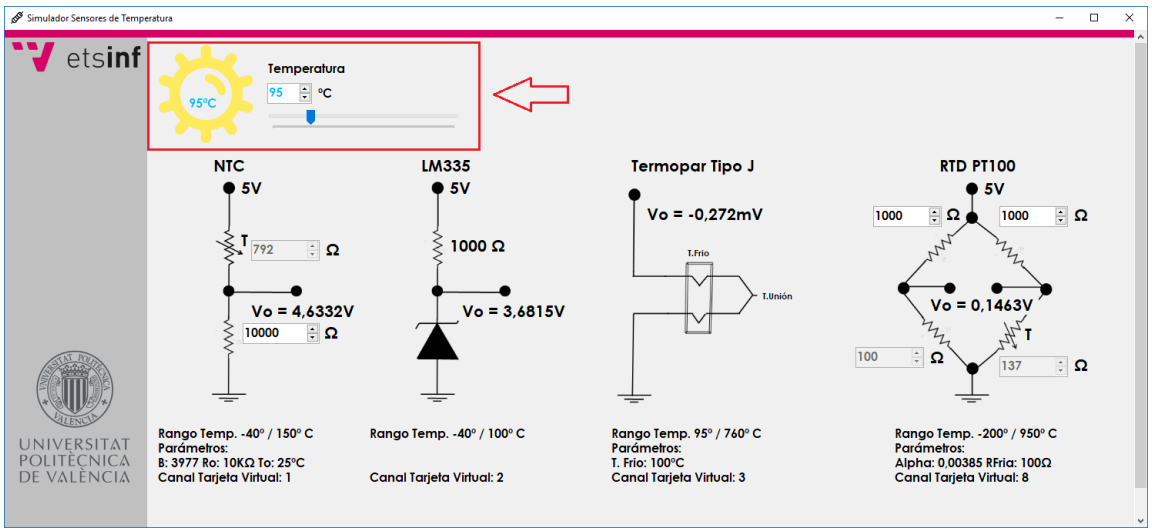

Imagen 24: Manual: Selector temperatura

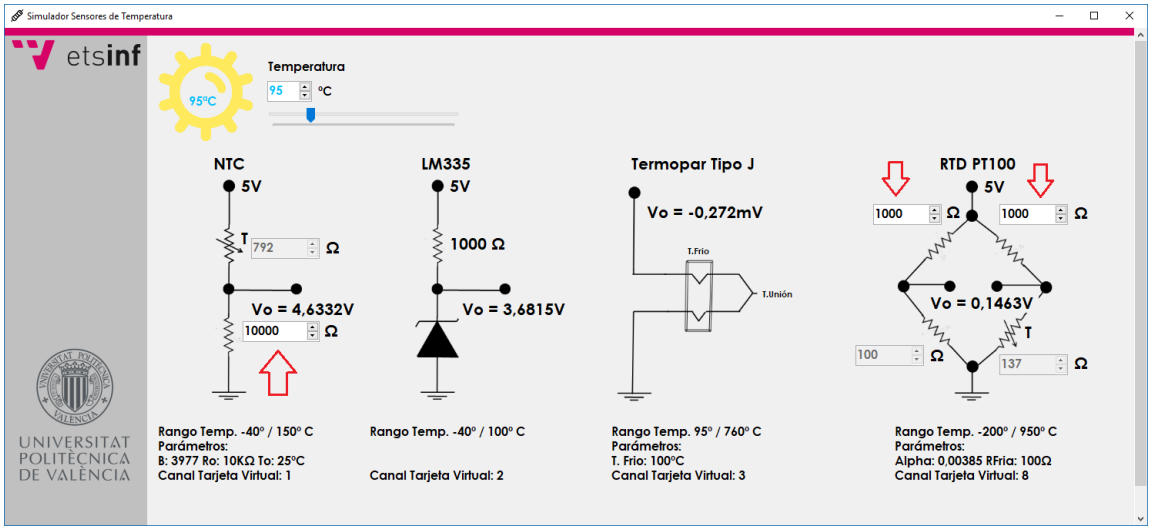

Imagen 25: Manual: Selector valor resistencias

Con la modificación de estos valores se obtendrán resultados distintos en la tensión de salida de los circuitos de instrumentación. El valor de esta tensión de salida será apreciable desde el simulador (Imagen 26) y obtenible desde la tarjeta virtual.

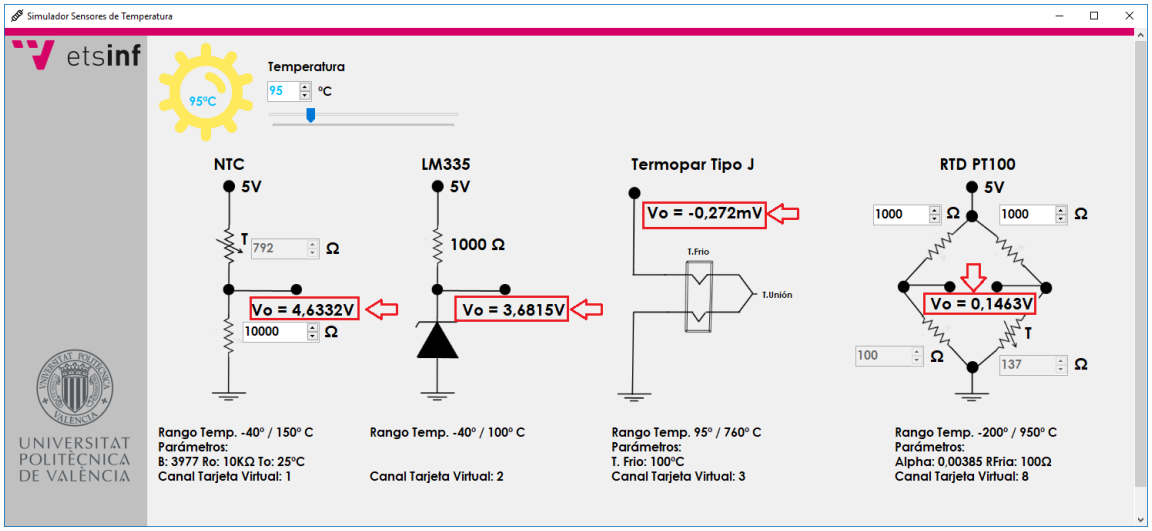

Imagen 26: Manual: Tensión de salida

Para obtener el valor de la tarjeta virtual será necesario la creación de un programa en C, este programa se puede crear con cualquier editor de texto y debe ser compilado por cualquier compilador de C para su posterior ejecución, como una de las posibles recomendaciones se encuentran el editor Notepad ++ y el Compilador Borland C++, pero eso ya queda a elección del usuario.

Este programa deberá incluir la librería de la tarjeta virtual, proporcionada junto al ejecutable del simulador (Imágenes 27 y 28).

```
#include <windows.h>
#include "libtarjeta.h"
#include <stdio.h>
#include <conio.h>
```

Imagen 27: Manual: Librerías Necesarias

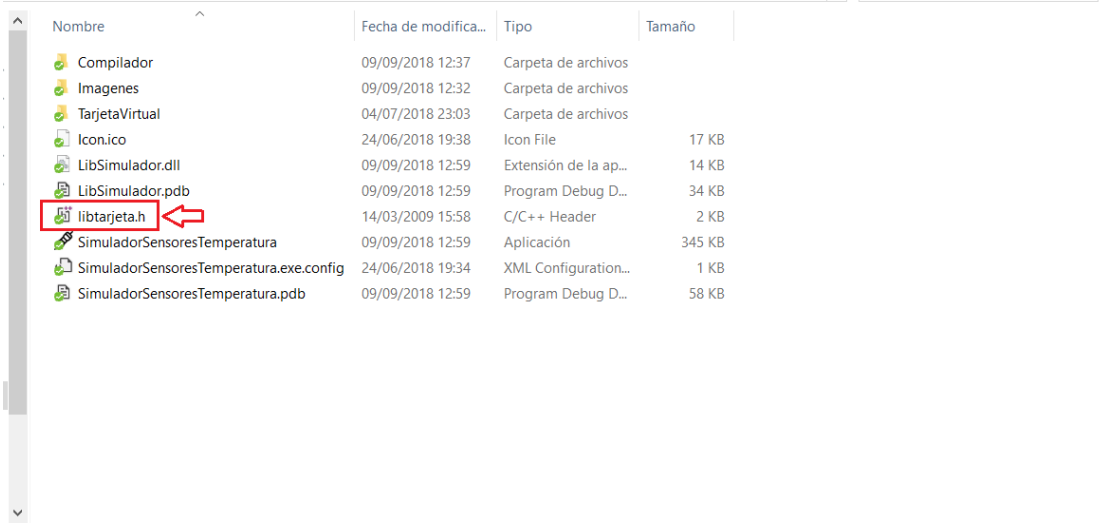

Imagen 28: Manual: Localización librería tarjeta

Si el programa en C se ha diseñado y compilado correctamente, al ejecutarlo se obtendrán dos posibles resultados, dependiendo de si se ha ejecutado antes o no el simulador de los sensores el cual inicia la tarjeta virtual, en los cuales se mostrará un error al no estar la tarjeta virtual encendida (Imagen 29) o se mostrará el resultado del programa implementado (Imagen 30).

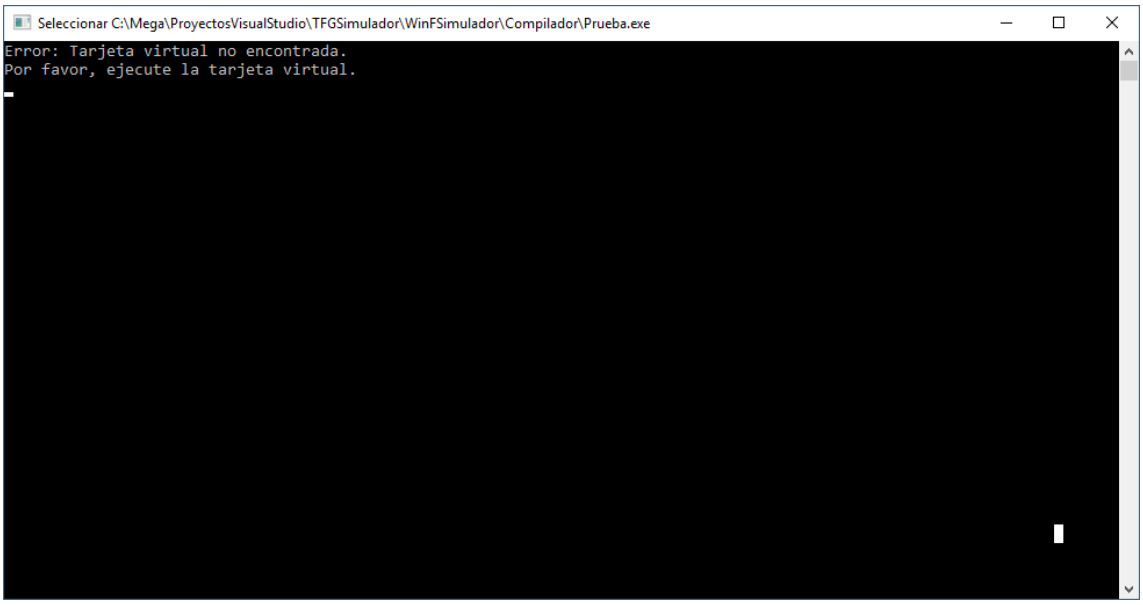

Imagen 29: Manual: Error tarjeta virtual no iniciada

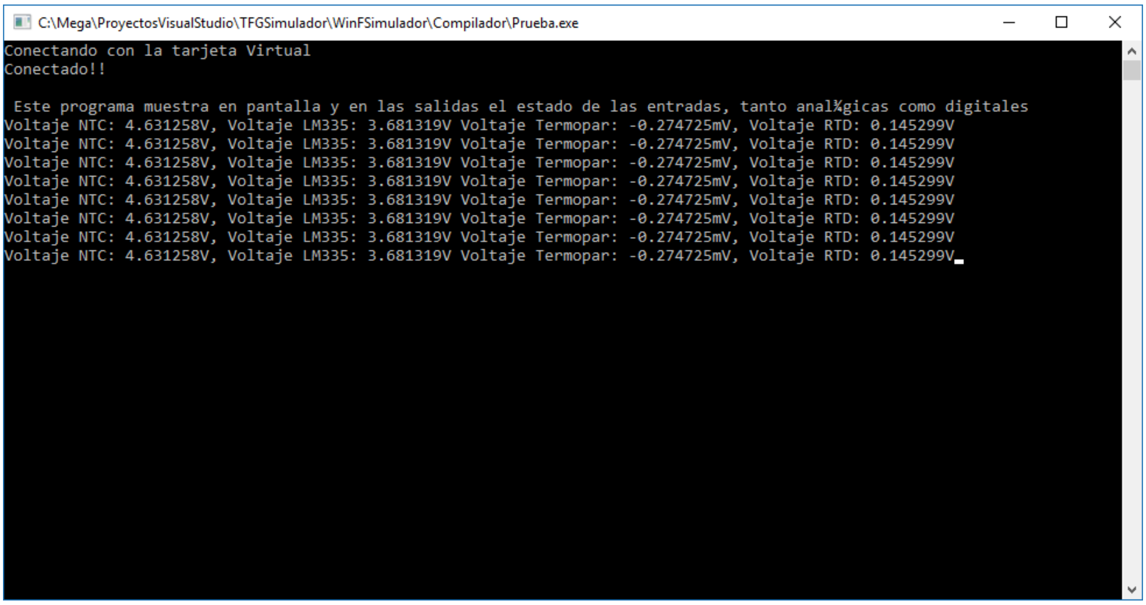

Imagen 30: Manual: Lectura correcta

Para finalizar con la ejecución, bastara con cerrar el simulador de los sensores, encargándose este de cerrar la conexión con la tarjeta virtual y cerrar su proceso.