Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería Electrónica Industrial y Automática

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

Autor:
Vera Chan, Ken

Tutor:
Zamarreño Cosme, Jesús María
Departamento Ingeniería de Sistemas y Automática

Valladolid, Julio 2018.

AGRADECIMIENTOS:

En primer lugar, a mi tutor, Jesús Zamarreño Cosme cuyo esfuerzo, superación y ganas de aprender han supuesto todo un buen ejemplo para mí.

En segundo lugar, a mi familia, sin los cuales no habría llegado a donde he llegado, por sus esfuerzos para que haya alcanzado esta última de mis metas de estudiante.

Y por último a mis amigos, presentes en los buenos y malos momentos.

PALABRAS CLAVE:

COMUNICACIÓN OPC, LABVIEW, PID, DAQ

RESUMEN:

En el laboratorio de Ingeniería de Sistemas y Automática de la sede Mergelina de la EII se dispone de varias plantas piloto con distintos objetivos de control, todas ellas instrumentadas y conectadas a ordenadores personales para su supervisión y control. Para el desarrollo de este proyecto se ha dispuesto de una planta de control de nivel de agua de un depósito y la plataforma de desarrollo de software LabView de National Instruments.

El objetivo del proyecto se basa en el desarrollo de un software que permita el diseño, implementación y operación de PIDs contemplando todas las etapas de diseño y estableciendo la comunicación con la planta a través del estándar OPC. Con todo ello el alumno podrá realizar prácticas para el desarrollo de sus conocimientos en automática referente a estudios en lazo abierto, lazo cerrado y cálculos de los parámetros de un PID de manera experimental.

ÍNDICE GENERAL POR BLOQUES:

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

I - INTRODUCCIÓN Y OBJETIVOS

TRABAJO FIN DE GRADO – KEN VERA CHAN
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

INDICE DE CONTENIDOS:

1. INTRODUCCIÓN Y OBJETIVOS

1.1. INTRODUCCIÓN:

Desde que James Watt inventara su primer regulador realimentado mecánico dando vida así al control automático Industrial hasta hoy, los métodos de control han variado muchísimo.

El control automático es una disciplina dedicada al estudio y diseño de dispositivos capaces de poder mantener una serie de variables de un sistema dentro de unos márgenes establecidos, sin que con ello implique la intervención humana en el proceso.

Una de las formas más populares de mantener un sistema controlado es a través del controlador PID. Un PID es un controlador automático basado en tres parámetros matemático-teóricos: una parte proporcional, una integral y otra derivativa, que expondremos detalladamente, más adelante.

El correcto diseño de un PID permite automatizar una planta de proceso de manera continúo pudiendo sustituir a una persona física que pudiera realizar ese control de forma manual. Así, la aplicación de estos dispositivos en el mundo está bastante expandida siendo algunas de las aplicaciones más comunes las siguientes:

- Lazos de temperatura (aire acondicionado, calentadores, refrigeradores, etc.)
- Lazos de nivel (nivel en tanques de líquidos como agua, lácteos, mezclas, crudo, etc.)
- Lazos de presión (para mantener una presión predeterminada en tanques, tubos, recipientes, etc.)
- Lazos de flujo (mantienen la cantidad de flujo dentro de una línea o tubo)
- Domótica
- Ingeniería aeroespacial (seguimiento de radiación solar de satélites artificiales)
- Energías limpias (eólico seguimiento de la dirección del viento, solar: seguimiento de la dirección de incidencia del sol)

1.2. JUSTIFICACIÓN DEL PROYECTO:

La motivación de la realización de este proyecto reside en la necesidad de poder diseñar e implementar un controlador PID en un laboratorio de control para las asignaturas de Automática estableciendo la conexión mediante un servidor OPC para la transferencia de datos entre la tarjeta de adquisición de datos (SAD) y la plataforma sobre la que se desarrolla el software diseñado, LabView de National Instruments.

De esta forma se podrá comprender el análisis que se tenía teorizado acerca de este controlador de manera experimental, procediendo como sigue a continuación:

- Se realizará una previa calibración de datos que se reciba de una determinada planta de estudio.
- Seguidamente se realizará una experimentación en lazo abierto o lazo cerrado cuyos parámetros obtenidos serán usados posteriormente para el diseño del PID.
- Por último, con los parámetros mencionados y los métodos existentes para el diseño del PID se finalizará calibrando el controlador hasta que su funcionamiento sea óptimo para la aplicación directa sobre un sistema.

1.3. OBJETIVOS:

Como se mencionó anteriormente, para poder realizar el correcto y óptimo diseño del controlador PID será necesario seguir los pasos que se concretaron. Aquí se detallarán con mayor precisión:

En primer lugar, se comenzará con una calibración de los datos recibidos por el sistema analizado. El proceso se basa en recolectar una determinada cantidad de datos a través de la tarjeta de adquisición de datos (SAD) recibidos desde un sensor, el cual convierte una variable física en valores de voltaje. Dicha recolección permitirá al usuario encontrar una ecuación matemática que relacione las entradas de medición con unas salidas de tensión, pudiendo trabajar con ésta última en algún punto del programa; calculo parámetros lazo abierto, cerrado y del diseño del PID.

Posteriormente, se procederá a realizar la experimentación en lazo abierto o la de lazo cerrado, ambas independientes entre sí, con el fin de poder extraer los parámetros correspondientes a ambas experimentaciones: la ganancia (K), el retardo (d) y la constante de tiempo (tau), para el lazo abierto y para el lazo

cerrado se tienen los parámetros: la ganancia crítica (Kc) y período crítico (Tc). Ambos métodos se explicarán con mayor detalle en el capítulo 3.

Después de ello se proporcionará un manual de usuario para el empleo del programa de cara a conexiones, calibración, experimentaciones y diseño de PID con la planta en cuestión que se pretenda analizar y controlar.

Al final de todo ello, se podrá diseñar el dispositivo PID correspondiente según cuál haya sido el tipo de experimentación que se había realizado previamente. Para el tipo de experimentación que se haya hecho se tendrá un conjunto de regulaciones de control hallados por combinación de los parámetros calculados L.A. y L.C. todos ellos con la finalidad de calcular: ganancia proporcional (P), tiempo integral (I) y tiempo derivativo (D). Para optimizar el diseño del controlador, el usuario podrá determinar tras el cálculo de los parámetros PID, qué tipo de configuración numérica es la más deseada modificando, de ser necesario, los mismos según la respuesta observada por el sistema.

1.4. ESTRUCTURA DE LA MEMORIA:

1.4.1. INTRODUCCIÓN Y OBJETIVOS:

Aquí se presentará una introducción a la temática del proyecto y se tratarán tanto la justificación del mismo, como la necesidad de su realización. Además de ello, se detallarán los objetivos que se pretenden conseguir en su desarrollo.
A continuación, se presentará la estructura de la memoria, citando las partes que posee y explicando brevemente su contenido.

1.4.2. SOFTWARE Y HERRAMIENTAS:

En este capítulo lo que se pretenderá es explicar la plataforma de programación que se ha empleado (LabView de National Instruments) para poder desarrollar el software del diseño del PID, así como las herramientas que se han utilizado, el hardware implicado y la tarjeta de adquisición de datos (SAD).

1.4.3. FUNDAMENTOS TEÓRICOS:

A lo largo de este capítulo se expondrán los conocimientos teóricos y matemáticos que conforman la automática que reside en el programa, se detallarán los cálculos precisos para poder extraer los parámetros implicados en la experimentación en lazo abierto y el significado físico que

poseen de cara al tipo de planta en la cual se realicen las determinadas prácticas y simulaciones. Además, se expondrán los desarrollos matemáticos para el caso de lazo cerrado con sus parámetros y sus respectivos significados físicos. Por último, se expondrán con amplio detalle los métodos de sintonía que existen para el cálculo de los parámetros del correspondiente PID según qué elección se haya seleccionado; en lazo abierto o en lazo cerrado.

1.4.4. DESARROLLO DEL SOFTWARE:

Primeramente, se expondrán todas las directrices que posea la interfaz del usuario a fin de que cualquier persona pueda comprender con facilidad en lo referente al empleo de las opciones que se proponen en la obtención de los parámetros que permitirán diseño del controlador PID. Los puntos que se seguirán para poder cumplir con este cometido serán; una primera calibración, hallando sus parámetros, que posteriormente serán utilizados en la experimentación en lazo abierto y lazo cerrado, pudiendo prescindir del cálculo de uno de ellos para proceder con la sintonía del PID, última de las tareas de control que se podrán diseñar con este software.

A continuación, se procederá con el planteamiento que se ha seguido para llegar a desarrollar el software en cuestión detallando la arquitectura que se ha creado para tener una guía sobre la que se establecerá el sistema programado.

Por último, lo que se ha añadido muy detalladamente, ha sido el manual del programador para que se pueda contemplar las diversas funciones que se han creado, bloques, subrutinas etc, que se diseñaron para componer el programa.

1.4.5. EJEMPLO PRÁCTICO:

En este capítulo se expondrán los detalles de la planta que había sido utilizada para el desarrollo del software del presente proyecto. A medida que se expongan los detalles de la planta utilizada se irán haciendo mención de lo referente a la calibración de los datos de llegada, la comunicación mediante un servidor OPC para la transferencia de datos, del software diseñado y de las herramientas empleadas.

1.4.6. CONCLUSIONES Y LINEAS FUTURAS:

Se expondrán las conclusiones obtenidas de este proyecto, tanto en la documentación precedente al desarrollo, como en el transcurso del proyecto, exponiendo en último lugar las líneas futuras de ampliación.

1.4.7. BIBLIOGRAFÍA:

En este último capítulo se expondrá la bibliografía que se ha empleado para realizar el presente proyecto, archivos de consulta, páginas web oficiales entre otros documentos.

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

II – SOFTWARE Y HERRAMIENTAS

TRABAJO FIN DE GRADO – KEN VERA CHAN
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

INDICE DE CONTENIDOS:

2. SOFTWARE Y HERRAMIENTAS

2.1. COMUNICACIÓN OPC:

El OPC es un estándar de comunicación en el campo del control y supervisión de procesos industriales, cuya especificación clásica se basaba en la tecnología Microsoft, estando las nuevas especificaciones (OPC UA) basadas en estándares abiertos. El estándar ofrece una interfaz común para comunicación que permite que componentes de software individuales interactúen y compartan datos. La comunicación OPC se realiza a través de una arquitectura Cliente-servidor. El servidor OPC es la fuente de datos (como un dispositivo hardware a nivel de planta) y cualquier aplicación basada en OPC puede acceder a dicho servidor para leer/escribir cualquier variable que ofrezca el servidor. Es una solución abierta y flexible al clásico problema de los drivers propietarios.

El Servidor OPC hace de interfaz comunicando por un lado con una o más fuentes de datos utilizando sus protocolos nativos (típicamente PLCs, DCSs, básculas, Módulos I/O, controladores, etc.) y por el otro lado con Clientes OPC (típicamente SCADAs, HMIs, generadores de informes, generadores de gráficos, aplicaciones de cálculos, etc.)

En una arquitectura Cliente OPC/ Servidor OPC, el Servidor OPC es el esclavo mientras que el Cliente OPC es el maestro. Así como se ha argumentado antes, las comunicaciones entre el Cliente OPC y el Servidor OPC son bidireccionales (los Clientes pueden leer y escribir en los dispositivos a través del Servidor OPC) Existen cuatro tipos de servidores OPC definidos por la OPC Foundation, y son los siguientes:

- **Servidor OPC DA (Acceso de Datos):** especialmente diseñado para la transmisión de datos en tiempo real.
- **Servidor OPC HDA (Acceso a Datos Históricos):** Provee al Cliente OPC HDA de datos históricos.
- **Servidor OPC A&E Server (Servidor de Alarmas y Eventos):** Transfiere Alarmas y Eventos desde el dispositivo hacia el Cliente OPC A&E.
- **Servidor OPC UA (Arquitectura Unificada):** Basado en el set más nuevo y avanzado de la OPC Foundation, permite a los Servidores OPC trabajar con cualquier tipo de datos.

En conjunto, los tres primeros tipos de Servidores OPC se conocen como Servidores OPC "Clásicos" para distinguirlos de OPC UA que se convertirá en la base de las futuras arquitecturas OPC.

Figura 2.1. Comunicación OPC Server

Fuente: https://www.matrikonopc.es/opc-servidor/index.aspx

Del anterior diagrama (figura 2.1) se pueden explicar detalladamente los siguientes apartados numerados:

- **En el punto primero:** Se detallan las Comunicaciones Cliente OPC / Servidor OPC (Servidor OPC DA, Servidor OPC HDA, Servidor OPC A&E)

  Los Servidores OPC clásicos utilizan la infraestructura COM/DCOM de Microsoft Windows para el intercambio de datos. Lo que significa que esos Servidores OPC deben instalarse bajo el Sistema Operativo de Microsoft Windows. Un Servidor OPC puede soportar comunicaciones con múltiples Clientes OPC simultáneamente.

- **En el punto segundo:** Se trata de un servidor OPC - Traducción de Datos/Mapping.
  La principal función de un Servidor OPC es el traducir datos nativos de la fuente de datos en un formato OPC que sea compatible con una o más especificaciones OPC mencionadas anteriormente (ejemplo: OPC DA para datos en tiempo real). Las especificaciones de la OPC Foundation solo definen la porción OPC de las comunicaciones del Servidor OPC, así que la eficiencia y calidad de traducción del protocolo nativo a OPC y de OPC al protocolo nativo dependen enteramente de la implementación del desarrollador del Servidor OPC.

- **En el punto tercero:** Se trata de un servidor OPC –Comunicación Fuente de Datos.

  Los Servidores OPC comunican nativamente con las fuentes de datos, por ejemplo: dispositivos, controladores y aplicaciones. Las especificaciones de la OPC Foundation no especifican como el Servidor OPC se debe comunicar con la fuente de datos porque hay una gran variedad de fuentes de datos disponibles en el mercado. Cada PLC, DCS, controlador, etc. tiene su propio protocolo de comunicación o API que a su vez permiten la utilización cualquier cantidad de conexiones físicas (serial RS485 o RS232, Ethernet, wireless, redes propietarias, etc.).

- Las ventajas de emplear servidores OPC son las siguientes:

a) Integración de distintas tecnologías de diferentes fabricantes dentro de un mismo sistema. La industria no tendrá que trabajar con un solo Sistema Propietario o Sistemas SCADA o DCS específicos.

b) Costos de desarrollo de sistemas de aplicación menores, dado que se está trabajando en una plataforma universal (OLE/COM), por lo que se evita duplicidad de esfuerzos. La adquisición de sistemas de aplicación y de Upgrade será más económica y además se podrá desarrollar sistemas específicos, según nuestras necesidades y no las que nos ofrezcan los fabricantes.

c) Permite integración de múltiples plataformas (Windows, Linux, Unix, suSe) mediante la utilización de COM, DCOM, Active X y Entire X. Permite un desarrollo comunicacional dentro de redes LAN (Local Access

Network) y WAN (World Access Network), como así también exportar datos a Internet.

d) Comunicación en línea libre, eficaz y flexible desde el nivel de procesos hasta el nivel de Gestión. Por medio de Softwares industriales específicos se puede lograr un mayor control dentro del proceso productivo, y optimizar Materias Primas, Recursos, Costos, etc.

- Las desventajas de emplear servidores OPC son las siguientes:

OPC parece ser un sistema industrial ideal, pero al ser tan transparente en el ámbito de las aplicaciones como así también interoperable en distintas plataformas presenta inherentemente problemas de Seguridad en estos aspectos.

a) Los fabricantes de Buses de Campo (fieldbus), han desarrollado Gateways para interconectar sus protocolos propietarios a redes Ethernet. Personas inescrupulosas podrían acceder desde el nivel de gestión hasta el nivel de proceso, dañando potencialmente todo el sistema comunicacional.

b) Hoy en día existe la capacidad de desarrollar programas ejecutables en base a OLE/COM/DCOM, e ingresar a sistemas de red de diferentes sistemas operativos a nivel WAN, mediante Entire X y Active X. La red de comunicaciones OPC puede estar expuesta a escala mundial y accesible desde cualquier plataforma operativa.

Para evitar estos problemas, se suele recomendar acrecentar los niveles de seguridad, usando Softwares de Encriptación de datos (para el tráfico de Datagramas) y Cortafuegos para proteger la Red de datos de otras externas.

En resumen y como se pueden ver en las figuras siguientes (figura 2.2 y figura 2.3), la comunicación OPC ofrece una interfaz común para transferencia de datos entre componentes de softwares individuales. Si bien en la figura 2.2, existe un problema de hilos de comunicación diferentes entre sí por cada componente que se conecte a un servidor:

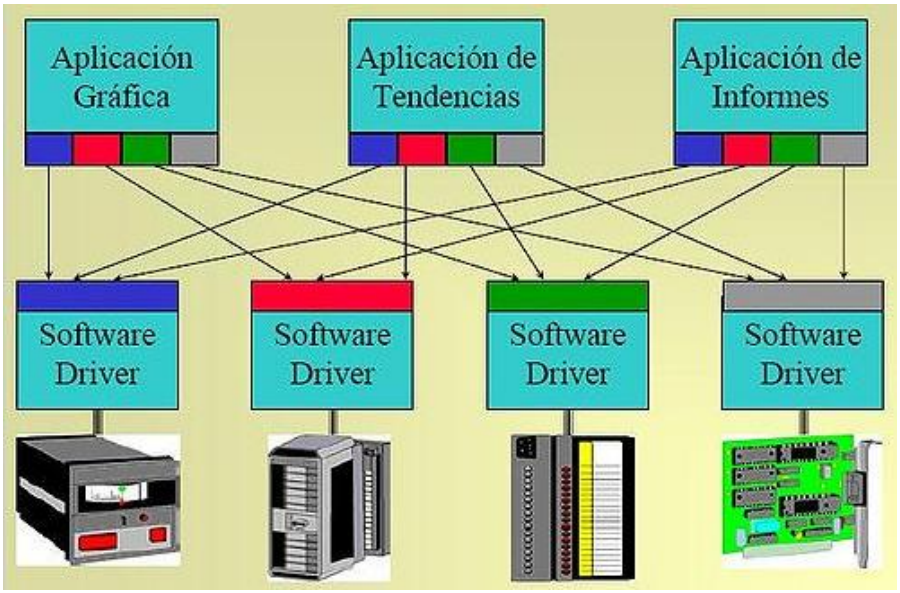


Figura 2.2. Problema de comunicación sin servidor OPC
(https://opcfoundation.org/about/opc-technologies/opc-ua/)

en la Figura 2.3, se propone la solución de crear una comunicación OPC estructurado como un Bus de campo común entre varios softwares que pretenden una transferencia de información:

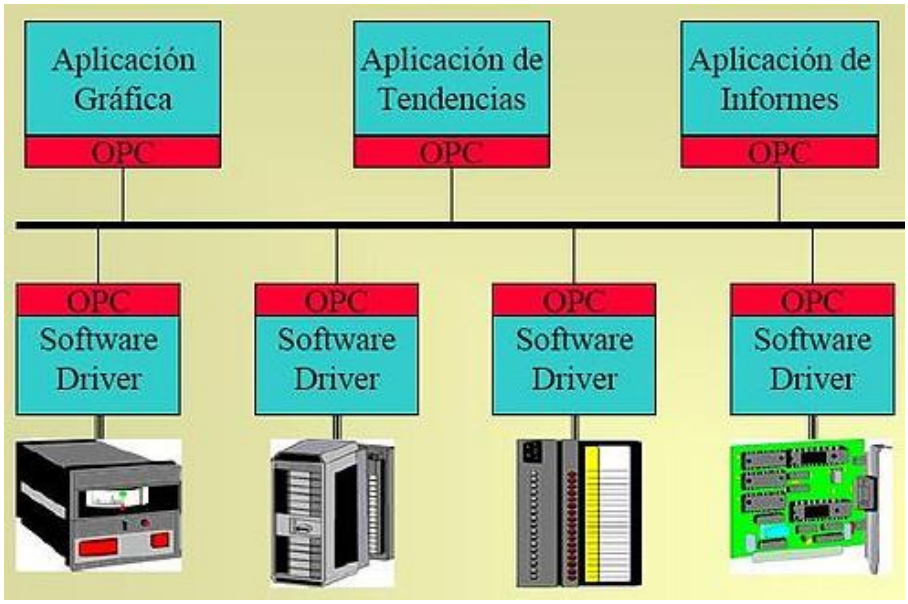


Figura 2.3. Solución de comunicación con servidor OPC
(http://lnx.enerxia.net/portal/index.php?option=com_content&view=article&id=245&catid=48&Itemid=113)

Así pues, de entre todos los tipos de servidores que se han comentado en este epígrafe, el tipo de comunicación OPC empleado en este proyecto está

basado en un **Servidor OPC DA (Acceso de Datos)** que como ya se comentó permite la transferencia de datos en tiempo real.

2.2. PLATAFORMA DEL ENTORNO DE DESARROLLO LABVIEW:

2.2.1. HISTORIA Y DEFINICIÓN DEL SOFTWARE:

En este apartado lo que se pretende es explicar la plataforma de programación que se ha empleado; LabView de National Instruments, para poder desarrollar el software del diseño del PID, así como las herramientas que se han utilizado.

Para explicar detenidamente el software sobre el cuál se desarrolla el presente proyecto, se argumentarán los siguientes aspectos fundamentales de dicha plataforma:

- LabView fue creada por la empresa National Instruments en 1986 para funcionar principalmente en máquinas MAC, la cual se define como la línea de computadoras personales diseñada, desarrollada y comercializada por la empresa Apple Inc
- Se define LabView como un software de ingeniería diseñado para aplicaciones que requieren pruebas, medidas y control con acceso rápido a información de datos y hardware.
- El desarrollo grafico de LabView permite reducir los tiempos de ejecución de los proyectos, y su completo entorno de depuración permite minimizar riesgos en la puesta en marcha de los sistemas.
- LabView trabaja con dos ventanas de programación, una de las cuales tiene como destino el diseño de las distintas secuencias de programación en forma de diagramas de bloques unidos mediante “hilos” a través de los cuales se transmite la información que se tratará en distintas zonas de dicha ventana. En la otra ventana se proporciona un entorno visual de interacción con el usuario que podrá accionar y visualizar una serie de sucesos que el programa, tras la previa ejecución, le proporcionará en base a lo que en la anterior ventana se había diseñado con un fin concreto, generalmente tratamiento de datos recibidos de manera externa y/o interna: por parte del usuario en cuestión. Posteriormente, se detallará el funcionamiento de dichas ventanas y la sincronización que ambas poseen a lo largo de la ejecución de un programa.

2.2.2. ENTORNO DE APLICACIÓN DEL SOFTWARE:

LabView es ampliamente utilizado en el ámbito industrial para el control y medida, especializado en la adquisición y análisis de datos, permite programar test automáticos y control de instrumentos.

Es posible ejecutar LabView en diferentes entornos: Windows, CE, Linux, Mac, Sistemas operativos en tiempo real, y también en diferentes dispositivos PC, PXI, FPGA y microcontroladores (véase Figura 2.4.)
LabView está presente en todo tipo de industrias: desde la industria pesada hasta el control electrónico, incluyendo grandes aplicaciones de control crítico. Entre los destinos a los que se dedica el programa en cuestión, también está el empleo académico para poder estudiar y analizar un tipo de programación visual práctico y menos abstractivo que utiliza algoritmos comunes del lenguaje C textual.

Como se ha contemplado en el epígrafe anterior acerca de las diversas formas de conexión Software- Hardware, LabView permite que diversos tipos de dispositivos puedan establecer una comunicación para una concreta trasferencia de datos, de entre ellos se pueden mencionar los que se exponen en el siguiente diagrama:

Figura 2.4. Elementos Hardware que abarca LabView

2.2.3. INTERFAZ DEL PROGRAMADOR:

Como se comentó en el apartado anterior, LabView es un software de programación visual que emplea dos ventanas de operación. La primera de ellas, sobre fondo blanco, permite al usuario hacer uso de las herramientas lógicas de programación para encajar rutinas y subrutinas que compongan los engranajes de un programa completo. Las diferentes herramientas que se utilizan en esta área son de diferentes tipos, de entre los cuales se encuentran los siguientes:

- **Lógicas**: aquellas que trabajan con variables booleanas.
- **Aritméticas:** aquellas que emplean números de tipo entero, decimal, hexadecimal… para trabajar con herramientas matemáticas del cálculo numérico.
- **Operandos Aritméticos**: referente a operaciones con base matemática de diferente índole y destino, para tratar datos numéricos y proporcionar un resultado.
- **Textuales**: son las que se destinan a procesamiento de datos de tipo string.
- **Arrays**: colección de datos ordenados en forma vectorial, si es de una dimensión, matricial si es bidimensional o una estructura de array para dimensiones mayores a dos.
- **Sentencias de programación**: de entre ellas while, do, for, case… que ejecutan un compendio de ordenes según la funcionalidad que posean y la relación que tengan con otras.
- **Comunicación**: en este caso las herramientas que dispone LabView de entre ellas, la que usa el presente proyecto basado en variables de tipo Socket para la comunicación OPC, protocolos TCP, colas y variables de tipo local y global para la transferencia de datos de cualquier tipo entre varios SUBVI, concepto que se comentará posteriormente.
- **Gestión de tiempos**: la utilidad de poder gestionar el tiempo permite al programador realizar tareas que requieran controlar un transcurso temporal a un ritmo diferente al ordinario, detención de la ejecución de rutinas durante un instante definido o meramente la conversión del tiempo en magnitudes mayores; horas, días, etc.
- **Gestión de ficheros de entrada y salida:** LabView permite trabajar con ficheros en donde se pueden almacenar datos que han sido tratados a lo largo del programa o cargar información para procesarlo en alguna rutina determinada.

- **Herramientas de control:** permite al usuario diseñar sistemas automáticos para proyectos de rutinas comunes de recepción, tratamiento y salida de datos.
- **Herramientas de interface y de diálogo con el usuario:** son aquellas que permiten a LabView interactuar con el usuario y están diseñadas para imitar una funcionalidad similar a la de los programas comunes de proposición de varias opciones y elección de la más conforme en base a una serie de consecuencias que muestre el software diseñado.
- **Herramientas de sincronización de procesos:** son aquellas que permite priorizar tareas, trabajar conjuntamente o compartir una base de datos entre varios procesos que usen una pila de información que puede ser fijo o variar de tamaño en función de la evolución del programa.

Todas las herramientas que se han expuestos antes pueden ser utilizadas al mismo tiempo, o interactuar entre varias rutinas VI a través de variables globales, conformando un software de la índole que corresponda según las utilidades empleadas. El panel que proporciona la iconografía de dichas herramientas tiene el siguiente aspecto:

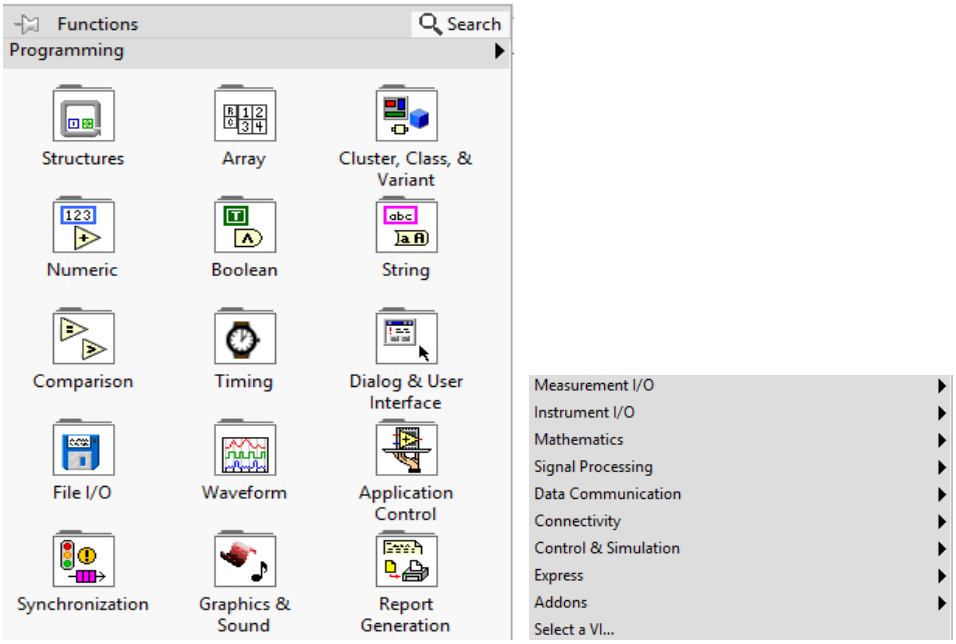


Figura 2.5. Panel herramientas: ventana del programador.

Como se ha señalado anteriormente durante la explicación de las herramientas, se mencionó la terminología de SUBVI y VI, los cuales son bloques de programa que considera LabView como unidad de programa que se

ejecutará de manera secuencial. Como antes se comentó existen un tipo de variable que se puede conectar con otros VI cuando aparezcan de manera icónica en la ejecución del programa, éstas son las variables globales.

Se denomina interfaz del programador a la zona de LabView en donde se diseña el conjunto de instrucciones del programa que realizará una o varias tareas empleando bloques de código conexionados entre sí formando un diagrama esquemático de ejecución. En la siguiente ilustración se pueden contemplar los bloques que se han mencionado, junto con la ventana que usará el programador para diseñar el diagrama ejecutable correspondiente:

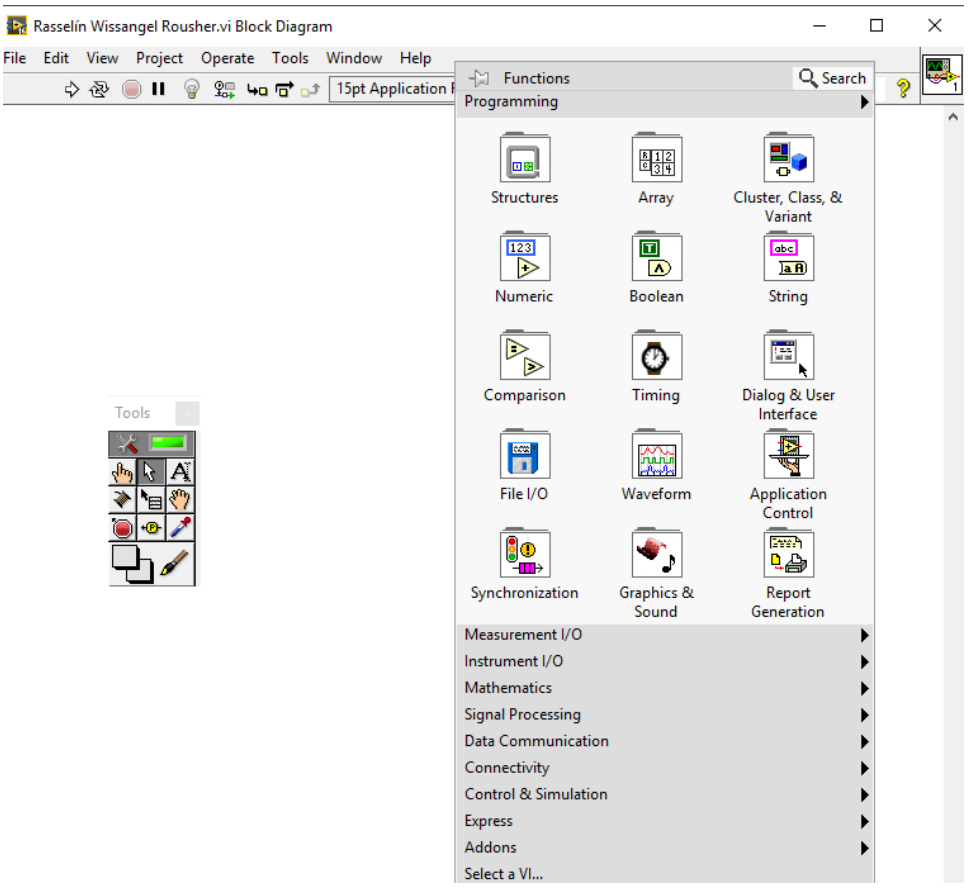


Figura 2.6. Ventana del programador LabView

A continuación de mostrará un ejemplo de programa realizado en esta plataforma de LabView, en donde se podrá observar el tipo de bloques empleados y la sincronización que se ha asignado para realizar una tarea. En este caso la tarea que ejecuta el programa se trata en ir mostrando los valores numéricos en una salida asignada que el usuario irá introduciendo indefinidamente hasta el instante en que el valor que haya introducido sea el cero, en cuyo caso lo mostrará por una segunda salida quedando registrado y

saliendo del bucle *do while* que se observa englobando todo el programa interior:

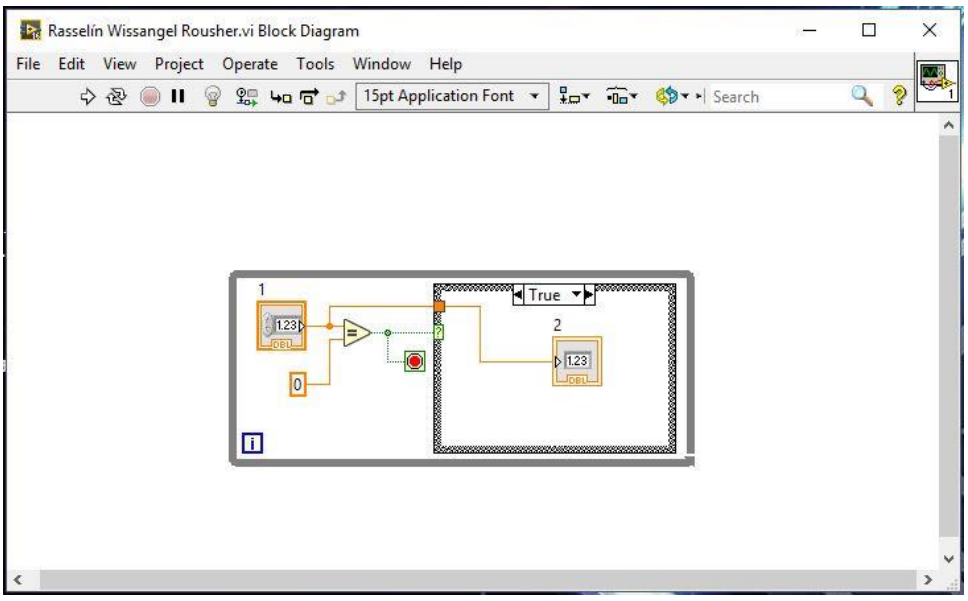


Figura 2.7. Ejemplo de programa realizado en LabView

En la ventana del programador se puede observar una serie de botones referentes al control de la reproducción del programa como si de un dispositivo multimedia se tratara. Dichos botones son los mostrados a continuación:

Figura 2.8. Botones de control de ejecución del programa.

Explicados de izquierda a derecha, el primero de ellos permite al usuario iniciar la ejecución del programa, si hubiera algún error de compilación, saldría una ventana emergente argumentando las observaciones pertinentes referentes a la discrepancia contemplada indicando el lugar, en el diagrama del programa, en donde se contraría dicho error. El segundo de ellos permite ejecutar el programa indefinidamente, aunque éste haya llegado al final del diagrama diseñado. El tercer botón detiene la ejecución del programa. Por último, el botón de las barras verticales permite al usuario detener el programa en un punto de su ejecución sin detenerlo por completo, pudiendo reanudarlo de volver a accionarlo por segunda vez.

El botón de la bombilla, en la ilustración, es único en esta ventana del programador ya que facilita a éste; el poder encontrar errores, verificar el funcionamiento de la trasferencia de datos a través de las conexiones por las líneas extendidas (hilos de colores para conectar bloques de programación) entre otras facilidades, todo ello debido a que al estar encendida la bombilla el

programa incluye en su ejecución un seguimiento en tiempo real de los bloques que están siendo utilizados, mostrando al mismo tiempo, una leyenda de las variables que están siendo tratadas al clicar sobre alguna de las líneas que trasporta los datos correspondientes.

En las explicaciones anteriores se mencionaron las líneas de transferencia de datos, de entre ellas se pueden diferenciar algunas de las más importantes que se muestran a continuación (Figura 2.9.):

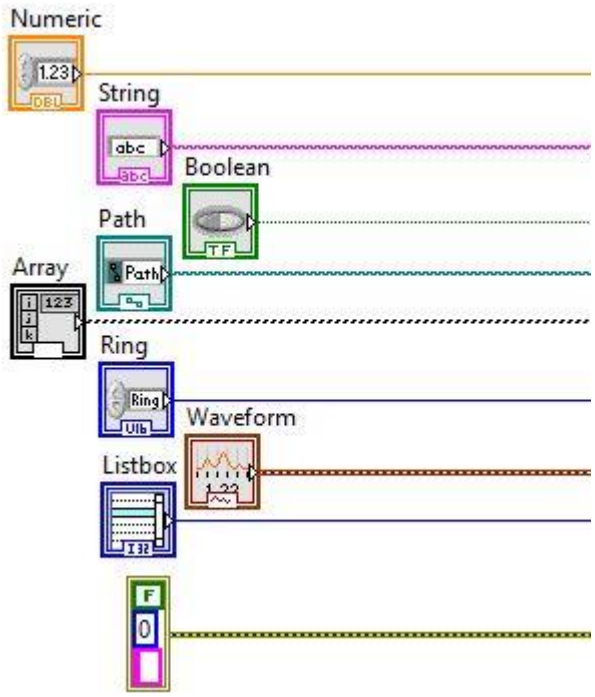


Figura 2.9. Tipos de datos en LabView clasificados por colores.

En la anterior ilustración (se describirán en descenso) se pueden observar los siguientes tipos de datos:

- **Numeric (hilo naranja):** Este tipo de variable transfiere datos numéricos enteros o con decimales. Su representación es de tipo DBL.
- **String (hilo rosa):** Es la variable que se encarga de transferir datos de tipo textual.
- **Boolean (hilo verde)**: Con esta variable se podrán transferir datos binarios, un solo dígito sea cero o uno.

- **Path (hilo verde azulado)**: Con esta variable se podrá transferir datos de tipo URL de directorios de Windows como rutas de situación de ficheros o carpetas donde cargar o guardar un elemento concreto.

- **Array (hilo negro)**: Las variables de tipo array son estructuras que almacenan cualquier tipo de datos de manera ordenada. Los Arrays pueden tener varias dimensiones, siendo un vector si es de una dimensión, matriz si fuera de dos dimensiones y array, en general, si es mayor de dos. En la ilustración anterior este array aparece en color negro porque aún no se ha destinado el tipo de variable que lo usará.

- **Ring (hilo azul):** Este tipo de variable es especial en LabView y su uso está basado en una estructura que el usuario podrá ver como un menú desplegable de opciones para elegir, pudiendo ser de tipo texto o numérico lo que el desplegable de opciones mostrará por pantalla.

- **Waveform (hilo granate):** Esta variable transporta datos generalmente de tipo numérico, enviados a través de un cluster de varias líneas de datos numéricos, para poder mostrarlos en un gráfico que el usuario visualizará por pantalla.

- **Listbox (hilo azul oscuro):** Es una variable que almacena una lista de datos como si de una "caja" se tratara, que después de procesará para fines comunes a las anteriores variables.

- **Error cluster constant (hilo amarillo):** Este último conjunto de variables de tipo: numeric; cuyo valor es cero, string; cuyo valor es un caracter vacío y boolean; cuyo valor es false. Todas ellas sirven para proceder al reinicio en caso de que surja algún error durante la ejecución del programa diseñado.

En ocasiones se utiliza también el hilo azul para variable de tipo numérico entero que se puede observar en bucles que poseen índices que irán recorriendo el mismo, cuyos números estarán definidos por números enteros positivos. Así mismo, el hilo amarillo, antes comentado, también sirve para verificar el tipo de error que surgirá en un bloque cuando el programa los vaya recorriendo y hubiera alguna discrepancia en concreto, que genere un mal funcionamiento como causa del efecto dominó de acarrear ese error en los bloques sucesores al bloque que avisó de la existencia de un fallo.

2.2.4. INTERFAZ DEL USUARIO:

En la parte de la ventana del usuario, el programador podrá colocar un conjunto de accionamientos, desplegables, gráficas de representación de datos numéricos, etc., que posibilitarán al usuario comprender la dinámica del sistema programado con facilidad y usarlo para el fin por el cual fue diseñado. En la ilustración siguiente (Figura 2.10) se puede observar la ventana del usuario cuando el programador empieza a construir la correspondiente interfaz de interactuación:

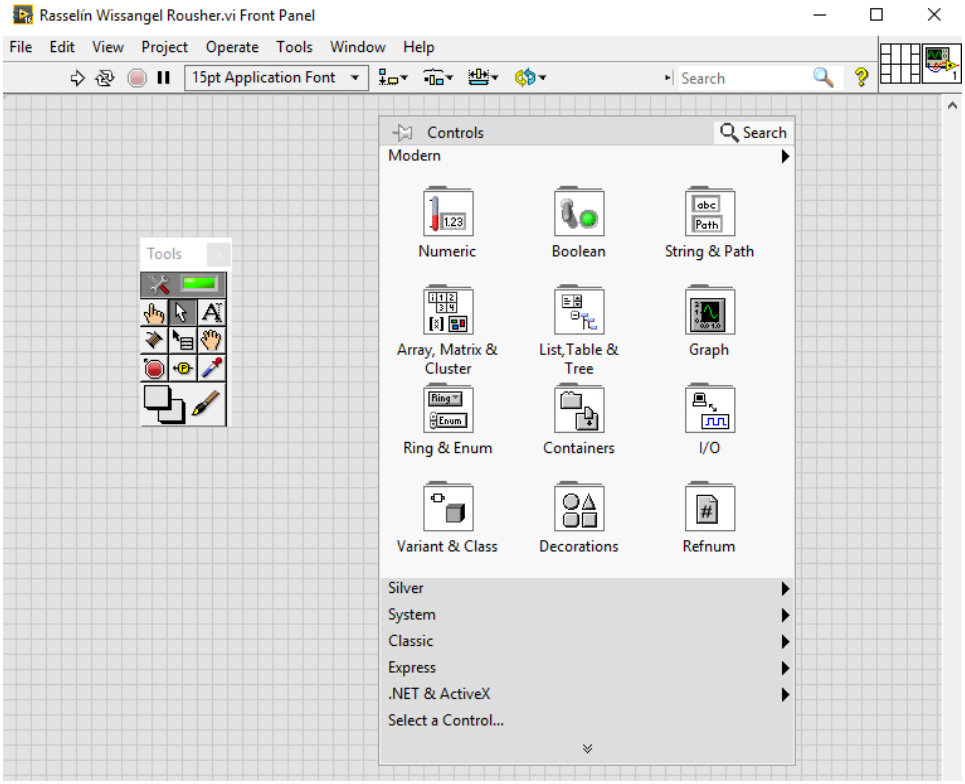


Figura 2.10. Panel herramientas: ventana del usuario.

En la imagen anterior se puede apreciar el menú de bloques funcionales bastante similar a la que se podía visualizar en la interfaz del programador, esto es debido a que en LabView, las variables que se crean en esta ventana se sincronizan con la otra ventana de fondo blanco debido a que existen tres tipos variables sincronizadas entre ambas ventanas que son las que se exponen a continuación:

- **De control:** aquellas que el usuario puede cambiar el valor que envíen a los bloques del diagrama del programa antes y/o durante su ejecución.

- **De indicación:** son las variables que muestran al usuario un valor que no podrá modificar, pero si visualizar con la intención de comprobar un resultado como consecuencia de haber tratado, en el transcurso del programa, una serie de variables en los diversos bloques funcionales.
- **Constantes:** Son aquellas variables que el usuario no verá en la interfaz que se le proporciona para interactuar con el programa pero que existen solo para que el programador fije valores que no variarán ni antes ni durante la ejecución del correspondiente programa.

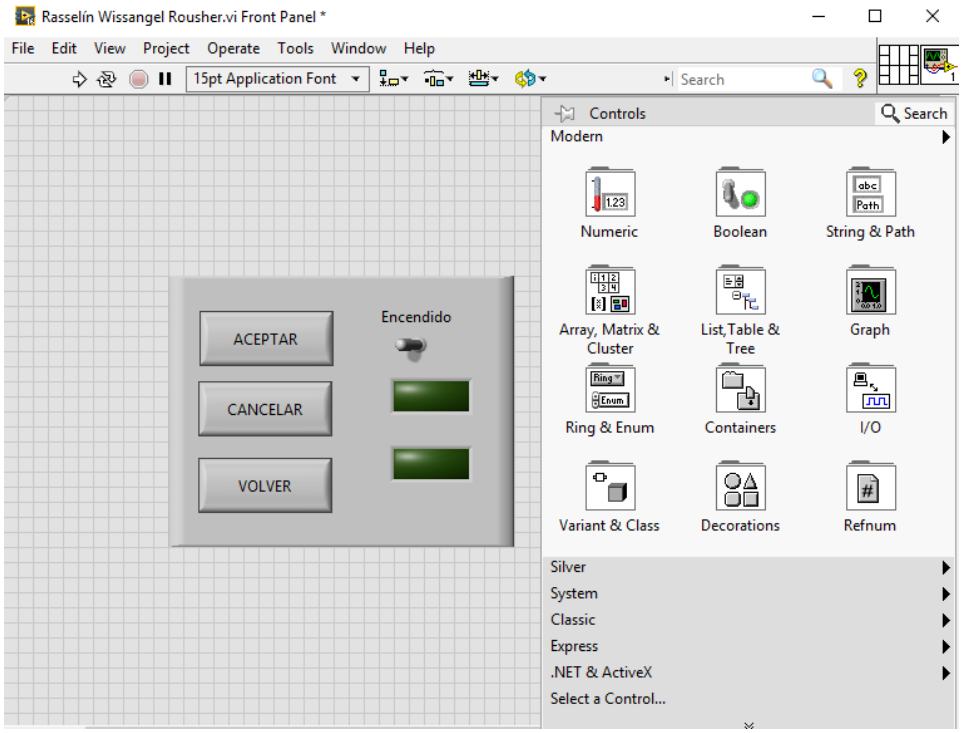


Figura 2.11. Ejemplo de interfaz de usuario en LabView.

Cuando se procede al diseño de la interfaz de usuario, se pretende que éste sea lo más sencillo posible, de manera que se emplearán juegos de botones que indiquen con palabras sencillas la dirección que tomaría el programa en curso. En la Figura 2.11 se puede observar un ejemplo de programa durante el proceso constructivo del mismo. De entre los elementos que aparecen en el panel interactivo, se puede comentar que el programador ha trabajado con datos de tipo Booleana distribuidos entre variables de control e indicación.

En el ejemplo comentado, cabe mencionar que algunas de las variables involucradas pueden ser utilizadas en otros SubVis. Obsérvese en la ilustración el botón de "VOVER", accionamiento muy común cuando se habla de menús secundarios emergentes de uno principal. La variable Booleana que existe

detrás de dicho botón esta sincronizada con otro programa VI de LabView. La manera en que están conexionados se puede contemplar la siguiente ilustración (Figura 2.12.a y Figura 2.12.c) en ambas imágenes se puede ver como un programa secundario se transforma en una "caja negra" en cuyo interior se aloja parte de un programa, realizando unas determinadas acciones. Dicho bloque tendrá asignadas unas variables cuyas funciones son dos: unas que se empleará en el interior y otras que tendrán como objetivo proporcionar los resultados que se generaron dentro del SubVI definido.
Para poder configurar este SubVI, se tiene que entrar en las configuraciones que ofrece la pantalla del usuario para asignar qué variables son entrantes y salientes (Figura 2.12.b) adjudicando un hueco cuadrado o rectangular (según convenga la disposición de entradas y salidas) según el tipo de variable: controlador o indicador. Generalmente los huecos de la izquierda estarán destinados para las variables que introducen información al bloque, mientras que los de la derecha serán para mostrar al usuario la información que el bloque ya ha procesado. Véase Figura 2.12.c.

Entre otras modificaciones que se pueden realizar está cambiar el nombre del bloque en cuestión con un fondo de concepción ilustrativo de las operaciones que éste ejecuta en su interior. Véase Figura 2.12.b cuadrado de la derecha.

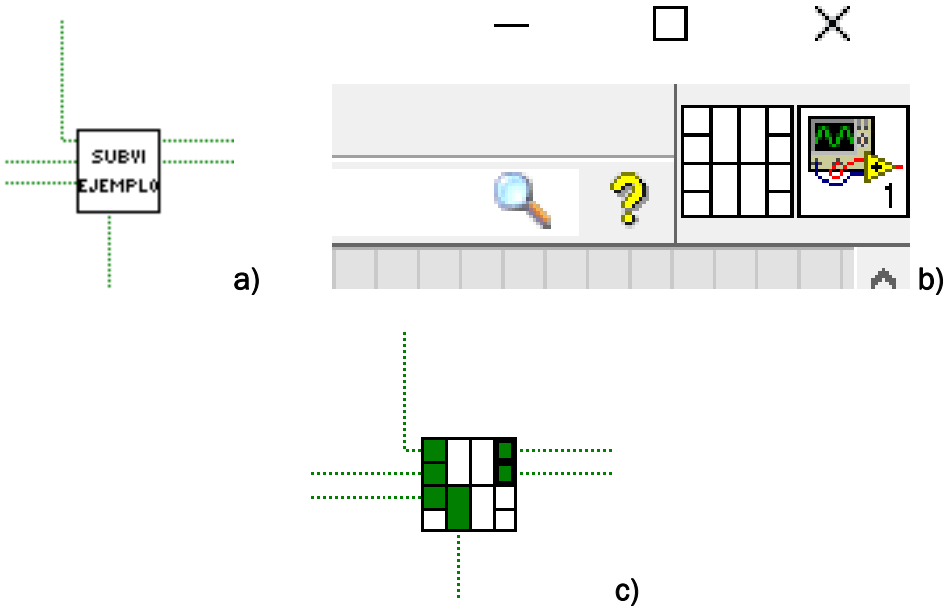


Figura 2.12. Iconos de interconexión para la sincronización de SubVIs.

La ventaja que este tipo de solución que propone LabView con la creación de SubVI se basa en la optimización visual de comprender la dinámica del programa diseñado, separando por zonas, la serie de operaciones que, durante la ejecución, el programa recorre usando y mostrando la información que

necesita y resulta respectivamente. Aparte de esta ventaja, está aquella basada en mostrar al usuario aquellas ventanas que únicamente sea conveniente que visualice, de cara a usar el programa de control diseñado.

En lo que respecta al presente proyecto, todas las indicaciones que se han comentado en este capítulo se pondrán en práctica de cara a exponer las ideas y las estructuras diseñadas para la realización del software en referente a su arquitectura. Los detalles explicados con las ilustraciones adjuntas permitirán con mayor facilidad comprender el compendio de ventanas de programación que se han tenido en cuenta para el software, entendiéndose como tal SubVIs que se sincronizan para poder realizar una determinada acción.
A lo largo del capítulo cuarto de esta memoria se especificarán que áreas han sido diseñadas para el usuario y que otras para el programador, pudiéndose identificar, sin mayor dificultad, qué acciones ejecuta el programa en SubVIs concretos, en función de los datos que le llegan y transmite.

Al ser bastante intuitiva la clase de accionamientos que se han incluido en la ventana del usuario, el lector podrá seguir al detalle la dirección de las argumentaciones que se expondrán minuciosamente.

Hay que tener en cuenta que todo el programa diseñado gira en torno a los conceptos teóricos del control automático para poder estudiar y analizar la evolución de los sistemas a los que se destine su aplicación. Así pues, a lo largo del siguiente capítulo se explicarán todos los conceptos necesarios para la realización del programa.

2.3. CREACIÓN, LECTURA Y ESCRITURA DE FICHERO XML:

En este capítulo, además, se comentarán los detalles del fichero de tipo XML que se ha empleado a lo largo de la ejecución del programa diseñado en LabView, permitiendo guardar y/o cargar los datos del diseño en curso de un controlador PID. Primeramente, se argumentará lo referente al concepto fichero XML.

XML es una especificación para diseñar lenguajes de marcado, que permite definir etiquetas personalizadas para descripción y organización de datos. Tiene su utilidad para representar información estructurada en la web (todos documentos), de modo que esta información pueda ser almacenada, transmitida, procesada, visualizada e impresa, por muy diversos tipos de aplicaciones y dispositivos.

Las ventajas que tiene son:

- Fácilmente procesable.
- Separa radicalmente el contenido y el formato de presentación.
- Diseñado para cualquier lenguaje y alfabeto.

Las características que tiene son:

- XML es un subconjunto de SGML (Standar Generalised Mark-up Language) que incorpora las tres características más importantes de este: Extensibilidad, Estructura y Validación.
- Basado en texto.
- Orientado a los contenidos no presentación.
- Las etiquetas se definen para crear los documentos, no tienen un significado preestablecido.
- No es sustituto de HTML.
- No existe un visor genérico de XML.

Las aplicaciones que tiene son:

- Publicar e intercambiar contenidos de bases de datos.
- Formatos de mensaje para comunicación entre aplicaciones (B2B)
- Descripción de meta contenidos.

Las consideraciones que tiene como formato de archivo son:

- Conjunto de datos con sus respectivas etiquetas de marcado XML.
- Se almacena como texto en archivo con extensión .xml.
- Un documento XML puede incluir cualquier flujo de datos basado en texto: un artículo de una revista, un resumen de cotizaciones de bolsa, un conjunto de registros de una base de datos, etc.

La estructura que tiene el documento XML:

Un documento XML está formado por **datos de caracteres** y **marcado**, el marcado lo forman las etiquetas. Para poder comprender con eficacia su estructura, se adjunta un modelo de fichero XML (Véase figura 2.16.):

Prologo
```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<! DOCTYPE persona SYSTEM "persona.dtd"
```
Cuerpo
```
<persona>
    <nombre>Luis</nombre>
    <apellidos>Pérez</apellidos>
</persona>
```

Figura 2.13. Ejemplo de código de fichero XML

Durante la escritura del fichero XML lo que se consigue es un orden en la organización de los datos escritos en el documento, de manera que cualquier persona, tras abrirlo, pueda observar un orden de prioridad de la información que se ha recopilado del experimento realizado con el software diseñado en este proyecto. En lo que respecta en la forma de colocar la información, se siguen las siguientes reglas para evitar errores de compilación:

Figura 2.14. Estructura del código XML

Lo que se ha contemplado en la figura 2.14 se comentará a continuación:

Los componentes de un fichero XML:

- Elementos: Pieza lógica del marcado, se representa con una cadena de texto(dato) encerrada entre etiquetas. Pueden existir elementos vacíos (
). Los elementos pueden contener atributos.
- Instrucciones: Ordenes especiales para ser utilizadas por la aplicación que procesa.
- Las instrucciones XML: Comienzan por <? Y terminan por ?>.
- Comentarios: Información que no forma parte del documento. Comienzan por <!– y terminan por –>.
- Declaraciones de tipo: Especifican información acerca del documento.
- Secciones CDATA: Se trata de un conjunto de caracteres que no deben ser interpretados por el procesador: <![CDATA[Aquí se puede meter cualquier carácter, como <, &, >, … Sin que sean interpretados como marcación]]>

La sintaxis de un fichero XML:

- Representa las normas a seguir para la construcción de documentos XML.
- Todo elemento tiene que tener su correspondiente etiqueta de inicio y de cierre, o una sola etiqueta vacía.
- Todo documento, debe haber un elemento (llamado raíz de documento) que contenga a los demás.

- Todos los elementos deberán estar correctamente anidados.
- Todos los valores de los atributos deberán ir entre comillas.

Las normas de construcción correcta de un fichero XML:

- La primera letra de los nombres se escribirá en mayúscula
- Los nombres compuestos se escribirán juntos o separados por guión bajo SacaCorchos
- Los elementos han de comenzar por un carácter o "_" no numérico.

Ahora que se conoce el formato de fichero XML y el modo en que se podía construir para almacenar información, se procederá a conocer cómo LabView puede crear y escribir sobre un documento de esta índole. Por orden de operatividad se expondrán los bloques que han podido crear el fichero XML, escribir sobre él y poder cerrarlo posteriormente con la posibilidad de reabrirlo y reeditarlo:

Figura 2.15. Bloques LabView: Crear y Cerrar Fichero XML

Estos dos bloques se suelen colocar justo en los extremos de la programación que se construirá del software de cara a trabajar con fichero XML dado que el primero de ellos, el de la izquierda, crea el fichero permitiendo incluir datos, borrarlos o reeditarlos mientras que el bloque de la derecha cierra el fichero XML que se ha reeditado o creado, imposibilitando que se puedan hacer más cambios sobre el mismo, a posteriori.

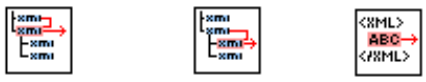


Figura 2.16. Bloques LabView: Edición de código XML

Estos tres bloques, figura 2.16, trabajan en lo que se refiere al código interno de colocación de la información en el fichero XML. De izquierda a derecha, el primero de ellos proporciona la información de un apartado en el código (declaración; en referencia a la variable que guarda), el segundo de ellos extrae la información de un subapartado, permitiendo cargar la información que pueda estar relacionada con un apartado anterior que no esté relacionado con otros apartados, por ejemplo, si un

apartado está declarado como “persona”, posibles subapartados podrían ser la edad, peso, altura, etc, que no tendrían nada que ver con otro apartado que fuera “puesto de trabajo” o “domicilio particular”. El último de ellos extrae información de un nodo en concreto de alguna declaración que se haya hecho en el código, tras abrirlo, por ejemplo, de los subapartados anteriores, edad, puede ser un número concreto de tipo “integer” o “float”, peso, otra variable de la misma índole que la anterior e ídem con la variable altura, mientras que en el apartado puesto de trabajo, el nodo que se extraería sería de tipo “string”, una frase o palabra que declarara el tipo de puesto asignado que se le ha otorgado; secretario, director jefe, notario, etc.

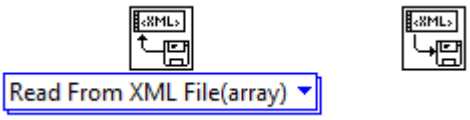


Figura 2.17. Bloques LabView: Lectura y Escritura

Los bloques que aparecen en la figura 2.17, permite leer o escribir en el fichero XML sin llegar a cerrar el mismo más son útiles cuando, durante la ejecución de un programa, se necesita que el fichero esté abierto para consultar algún dato del documento XML o actualizar información de algún experimento repetido no conforme con los resultados que se habían obtenido. El bloque de la izquierda lee de un fichero XML, cargando la información que existe en él mientras que el de la derecha lo que hace es escribir datos en el mismo fichero, ya sean nuevos para rellenar los campos de documento XML vacíos o actualizar algún dato concreto en uno o varios apartados. El tipo de variable que se pueda guardar en el fichero XML puede ser de cualquiera que se necesite almacenar sin restricciones ni limitaciones por posibles fallos en referente a guardar o abrir un documento concreto.

Figura 2.18. Bloques LabView: Guardar y Cargar datos

Por último, estos dos bloques permiten programar la manera en que los datos adquiridos se guardan o se cargan para ser almacenados o utilizados, respectivamente.

Antes se comentó que los bloques que se expusieron inicialmente; crear o cerrar un fichero XML, eran los extremos en la programación de una rutina en LabView relacionada con tratamiento de ficheros de este tipo,

pero en ocasiones se pretende terminar una rutina de un programa no solo cerrando el fichero escrito, sino que además guardando las modificaciones que pudieran haberse realizado en éste. Solo en aquellos que el fichero XML tenga un destino de mera consulta, no será necesario emplear el bloque de la derecha de la figura 2.18. más una consulta no precisa de realizar modificaciones.

Téngase en cuenta que, en el caso de lectura de un fichero, para poder consultar alguna información sí que precisaría del bloque de la izquierda para cargar los datos que se consultarán posteriormente.

Así pues, para tratamientos de ficheros XML, los bloques que se han expuesto representan puntos clave para el programador en el entorno de LabView, pudiendo crear, editar, modificar y guardar datos relevantes a meros cálculos o como respecta a este proyecto a una recolección de los datos de la experimentación en lazo abierto, lazo cerrado y el diseño de sistemas de control con PIDs.

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

III – FUNDAMENTOS TEORICOS

TRABAJO FIN DE GRADO – KEN VERA CHAN
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

INDICE DE CONTENIDOS:

3. FUNDAMENTOS TEÓRICOS

A la hora de analizar un sistema físico basado en ecuaciones que definen su comportamiento, en ocasiones no resulta posible ni práctico extraer la función de transferencia que se obtenga de sus expresiones matemáticas debido a su complejidad en el cálculo o a los principios teóricos a los que se fije dicho sistema. Ante ello se emplea una alternativa eficiente basado en modelos dinámicos empíricos que poseen las siguientes características:

- Los modelos que se obtienen han sido estudiados de manera experimental contemplando el sistema como un proceso que posee una serie de salidas y entradas.
- El modelo que resulta del previo análisis experimental logra describir de manera aproximada la relación que existe entre ciertas variables determinadas de entrada y salida.
- Al ser un modelo lineal obtenido de manera experimental, sucede que sólo será válido para una región limitada en torno a unas condiciones iniciales.
- El modelo empírico resultante incluye la dinámica de todos los elementos existentes entre la variable de entrada que se perturba y la variable de salida, que será la señal medida que se irá registrando en tiempo real.

Gracias a toda esta información que proporcionan los elementos descritos anteriormente se podrá diseñar con eficacia un sistema de control.

Para poder realizar el diseño de un sistema de control, se parte de la función de transferencia que proporciona el modelo lineal. Dicha función de transferencia es una fidedigna representación de la respuesta del modelo experimental desde la entrada hasta la salida ante una entrada escalón, sin ser necesario el conocimiento de la construcción interna del sistema.

Con la mencionada entrada escalón y a partir de la curva de reacción resultante que proporcione el modelo experimental se podrá medir la
constante de tiempo y el valor en estado estacionario, y a partir de éstos se puede calcular la función de transferencia. Con lo comentado anteriormente hablaríamos, entonces, de la identificación de un sistema de primer orden con retardo que se detallará a continuación como sistema de control en lazo abierto.

3.1. EXPERIMENTACIÓN EN LAZO ABIERTO (L.A.):

Los sistemas en los cuales la salida no tiene efecto sobre la acción de control se denominan “sistemas de control en lazo abierto”. En otras palabras, en un sistema de control en lazo abierto no se mide la salida ni se realimenta para compararla con la entrada.

En cualquier sistema de control en lazo abierto, la salida no se compara con la entrada de referencia. Debido a esta configuración, a cada entrada de referencia le corresponde una condición de operación fija, y como consecuencia de ello, la precisión del sistema depende de la calibración previa. Ante la presencia de perturbaciones, un sistema de control en lazo abierto no realiza una tarea deseada. Así pues, en base a lo argumentado, un sistema en lazo abierto tiene la siguiente representación en forma de diagrama de bloques:

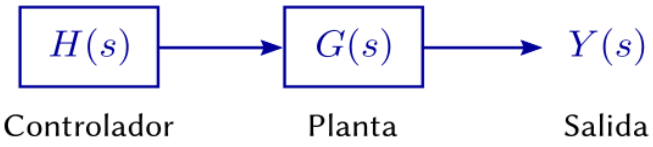


Figura 3.1. Diagrama de bloques sistema en lazo abierto

En la práctica, el control en lazo abierto, sólo se usa si se conoce la relación entre la entrada y la salida y si no hay perturbaciones internas ni externas. Ante estas condiciones, es evidente que estos sistemas no son de control realimentado, por ello la acción de control se calcula conociendo la dinámica del sistema, las consignas y estimando las perturbaciones. Esta estrategia de control puede compensar los retrasos inherentes del sistema anticipándose a las necesidades del usuario. Sin embargo, el lazo abierto generalmente es insuficiente, debido a los errores del modelo y a los errores en la estimación de las perturbaciones. Por ello, es común la asociación de lazo cerrado-lazo abierto, de modo que el lazo cerrado permite compensar los errores generados por el lazo abierto.

En la experimentación existe una serie de parámetros basados en un ajuste empírico para estudiar la evolución de una planta de procesos. La mayoría de dichos procesos poseen una respuesta monótona creciente estable ante una entrada escalón. En la siguiente ilustración (figura 3.1.) se pueden contemplar cuatro procesos representativos:

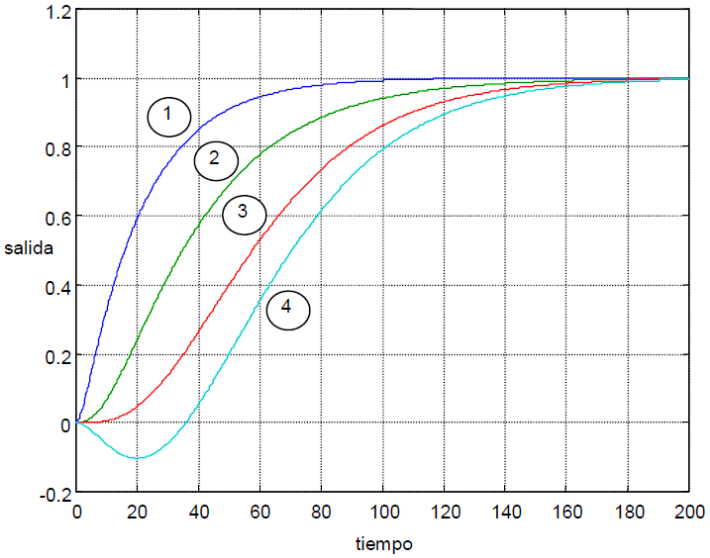


Figura 3.2. Procesos representativos de control en lazo abierto

Los parámetros, antes mencionados, describirán un modelo simple y lo óptimo que pueda aproximarse a las características de la respuesta. Dicho modelo suele ser uno aproximado a un sistema de primer orden con un retardo. Lo que se pretende en todos los sistemas que posean análisis empíricos es identificar el orden del mismo para poder extraer su función de transferencia y con ello aproximarlo a un sistema de primer orden con un retardo determinado, con el fin último de poder extraer una serie de parámetros que definirán la experimentación en lazo abierto cuyas características se pueden apreciar en el siguiente trazado (figura 3.2.):

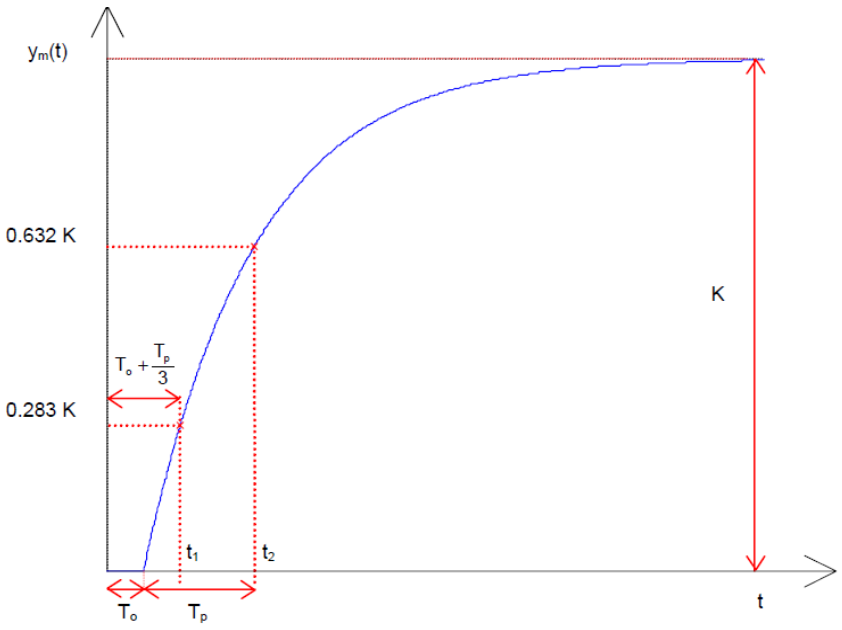


Figura 3.3. Modelo aproximado primer orden más retardo

De la ilustración se pueden extraer los parámetros: Retardo (d), Constante de tiempo (τ: tau), Ganancia del sistema (K), obtenido como cociente entre la respuesta producida por el sistema y el salto introducido en la entrada.

Para poder extraer los parámetros del modelo aproximado de primer orden con retardo se procederán a emplear dos de los posibles métodos:

- **Método de diferencia de tiempos:**
  Diferencia de tiempo entre el punto alcanzado que representa el 28% y el 63% del tiempo de respuesta de la curva de reacción.
- **Método de la pendiente máxima**:
  Dada la curva de reacción se pretende encontrar la recta tangente a dicha curva cuya pendiente sea la más pronunciada de todas las posibles que haya a lo largo de la curvatura de la respuesta a una entrada a escalón.

El primero de los métodos se describe como diferencia de tiempos, el tiempo en que la curva alcanza el 28% y el 63% de la respuesta. De manera gráfica se puede observar en la figura 3.3 mostrada a continuación:

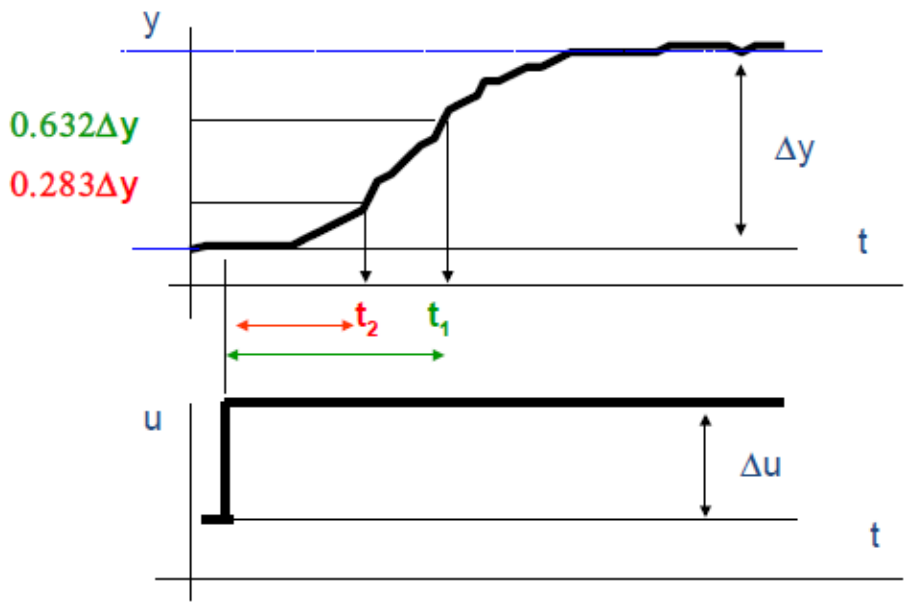


Figura 3.4. Modelo aproximado primer orden más retardo: método de la diferencia de tiempos.

Para poder realizar el estudio, se tiene que tener en cuenta las siguientes consideraciones durante el experimento:

- Control en manual.
- Esperar hasta que la salida esté en estado estacionario.

- Provocar salto en la variable manipulada.
- Registrar la salida (variable controlada) hasta que alcance el nuevo estado estacionario.
- Obtener K como el cociente entre cambios; numerador será la salida que proporcione la variable controlada y en el denominador, el salto específico provocado en la variable manipulada.
- Medir instantes de tiempo en segundos: t_1 al 28.3% y t t_2 al 63.2%.
- Obtener parámetro constante de tiempo (τ: tau): $1.5 \cdot (t\ t_2\text{-}t_1)$
- Obtener parámetro de retardo temporal (d): $t_2 - \tau$

En el segundo de los métodos emplea la inserción de una recta tangente a la curva de reacción en el punto de inflexión, cuya pendiente representa la constante de tiempo. El punto en que la recta tangente corta al eje horizontal de la gráfica de estudio (valor inicial de la variable), representa el retardo que posee el sistema ante la entrada salto que se había introducido en la experimentación. En la siguiente figura 3.4. se puede visualizar los parámetros de experimentación que se han comentado.

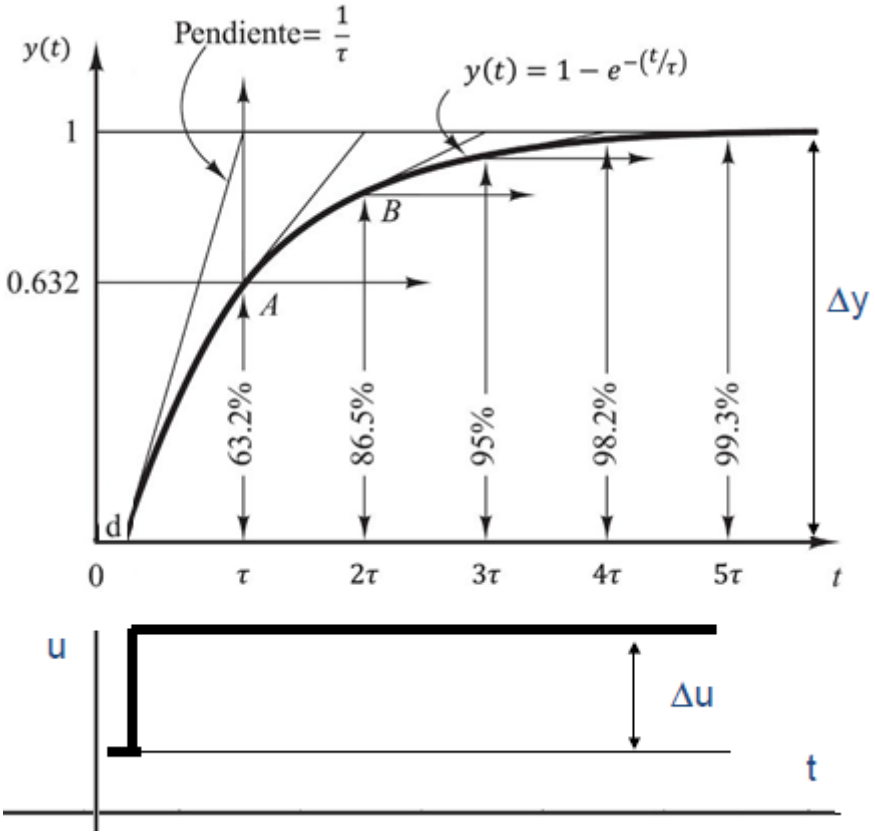


Figura 3.5. Modelo aproximado primer orden más retardo: método de la pendiente máxima.

El inconveniente de todo ello se basa en el trazado de la comentada recta tangente de máxima pendiente, especialmente en aquellos casos en los que la señal correspondiente a la variable de salida posea mucho ruido o presente un comportamiento extremo; pendiente muy grande para sistemas con excesivo ruido y pendiente muy pequeña para sistemas que carezcan del mismo o que esta señal sea mínima.

Después de realizar, cualquiera de los dos anteriores métodos sobre la curva de reacción se procederá a describir la ecuación de primer orden que seguirá el siguiente modelo matemático:

$$G(s) = \frac{K}{\tau S + 1} e^{-dS}$$

Figura 3.6. Función transferencia del sistema de primer orden con retardo.

Con los parámetros calculados anteriormente, se podrá diseñar un posterior controlador automático PID, basado en la experimentación en lazo abierto que emplea dichas medidas (K, Tau, d)

3.2. EXPERIMENTACIÓN EN LAZO CERRADO (L.C.):

Son sistema de control en lazo cerrado aquellos en los que la señal de salida del sistema (variable controlada) tiene efecto directo sobre la acción de control (variable de control).

La acción de control se calcula en función del error medido entre la variable controlada y la consigna deseada. Las perturbaciones, aunque sean desconocidas son consideradas indirectamente mediante sus efectos sobre las variables de salida. Este tipo de estrategia de control puede aplicarse sea cual sea la variable controlada. La gran mayoría de los sistemas de control que se desarrollan en la actualidad son en lazo cerrado.

Para comprender el funcionamiento del sistema de control en lazo cerrado se definirá el concepto de: realimentación.

La realimentación se define como la operación que en presencia de perturbaciones tiende a reducir la diferencia entre la salida de un sistema y alguna entrada de referencia. Esta reducción se logra manipulando alguna variable de entrada del sistema, siendo la magnitud de dicha variable de entrada función de la diferencia entre la variable de referencia y la salida del sistema. Ante esta explicación se procederá a detallar los conceptos de las

siguientes partes de un sistema de control en lazo cerrado, cuya representación es la siguiente:

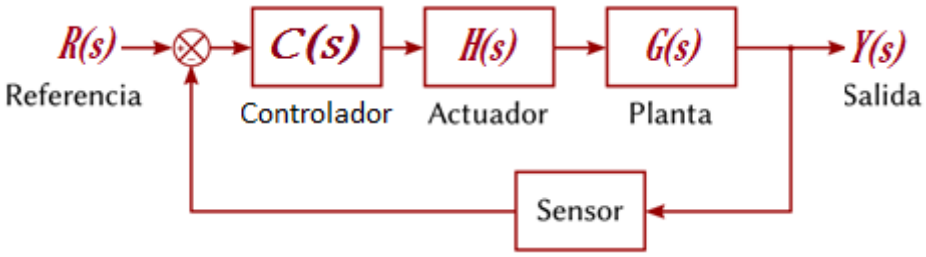


Figura 3.7. Diagrama de bloques sistema en lazo cerrado.

- **Variable de entrada:** variable del sistema tal que una modificación de su magnitud o condición puede alterar el estado del sistema.
- **Variable de salida:** variable del sistema cuya magnitud o condición se mide.
- **Perturbación:** señal que tiende a afectar el valor de la salida de un sistema. Si la perturbación se genera dentro del sistema se la denomina interna, pero si es una perturbación externa, la cual se genera fuera del sistema, constituiría una entrada.
- **Sistemas de control:** conjunto de dispositivos que actúan juntos para lograr un objetivo de control.
- **Señal de referencia:** señal que se calibra en función del valor a la salida del sistema.
- **Sensor:** Dispositivo que permite conocer los valores de las variables de medidas del sistema. En la realimentación envía la señal de la salida al actuador junto con la referencia.
- **Planta:** Es el conjunto de componentes o piezas que van a tener un determinado objetivo de control.
- **Transductor:** Dispositivo que transforma un tipo de energía en otro más apto para su utilización. Si la energía trasformada es en forma eléctrica se llama sensor. Por ser el instrumento encargado de detectar la señal de salida para utilizarla de nuevo en el proceso de realimentación se le llama en los sistemas de control captador.

En la experimentación, para la obtención de los parámetros de control: Ganancia crítica (Kc) y Período crítico (Pc) se procederá con el método que se expone a continuación: **Método de oscilación de Ziegler-Nichols.**

Es importante saber que este método es válido sólo para plantas estables en lazo abierto dado que las oscilaciones que se producirán en la obtención de la ganancia crítica dependen de su estabilidad; en el caso de que no fuera estable en lazo abierto, una primera iteración en la búsqueda de la ganancia,

modificando ésta, provocaría la inestabilidad del sistema en cuestión, generando unas oscilaciones crecientes.

El procedimiento para la obtención de los mencionados parámetros es el argumentado a continuación:

- Aplicar a la planta sólo control proporcional con ganancia Kp pequeña.
- Aumentar el valor de Kp hasta que el lazo comience a oscilar. La oscilación debe ser lineal y debe detectarse en la salida del controlador.
- Registrar la ganancia resultante como ganancia crítica Kp = Kc. Si se siguiera aumentando la ganancia del sistema, se provocarían oscilaciones crecientes, desestabilizando el sistema.
- El período de oscilación se obtendrá cuando habiendo calculado la ganancia crítica se mida el tiempo que existe entre dos crestas o dos puntos a la misma altura en la respuesta que proporciona el sistema, este valor corresponderá con el período crítico de la oscilación Pc a la salida del controlador.

3.3. EL CONTROLADOR PID:

En este apartado se expondrán los elementos esenciales para el diseño de un controlador automático de tipo PID y la relación que existen entre sí de cara a las respuestas que pueda proporcionar un sistema en el que se aplique el controlador.

Un controlador PID se coloca en un sistema de control en lazo cerrado para poder realizar una serie de acciones sobre un actuador, recibiendo una señal de error que se pretende corregir dinámicamente. En el siguiente diagrama se podrá observar como el controlador PID se coloca en el sistema y dónde para poder comprender el funcionamiento del mismo:

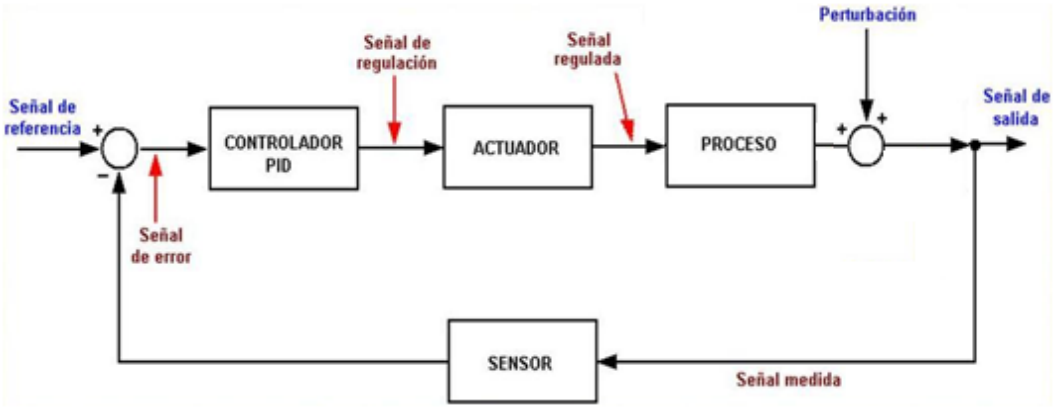


Figura 3.8. Diagrama de bloques con un controlador PID

En el anterior diagrama se puede contemplar los diferentes elementos del sistema que se sincronizarán para que el controlador pueda mantener la señal de referencia dentro de unos márgenes establecidos a pesar de las posibles perturbaciones que reciba el proceso existente. EL controlador PID posee tres parámetros, los cuales tienen efecto sobre la respuesta del sistema de la siguiente manera ilustrativa (figura 3.9)

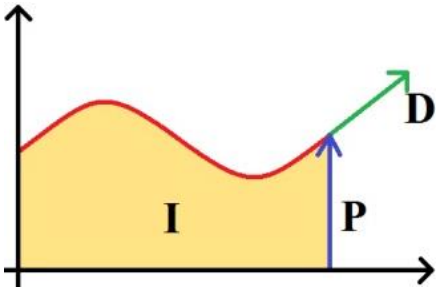


Figura 3.9. Efecto de cada parámetro del controlador PID en la respuesta.

La parte P, supone el parámetro proporcional del controlador y se encarga de reducir el error estacionario, aunque no logra eliminarlo. Obsérvese en la imagen la flecha azul que indica cómo reduce el error estacionario que posee el sistema ante perturbaciones en el proceso.

La parte I, hace referencia al tiempo integral del controlador y tiene como función eliminar el error estacionario. Para un valor pequeño implica una acción integral grande y, por el contrario, un tiempo integral grande implica una acción integral pequeña. La acción integral proporciona una corrección para compensar las perturbaciones y mantener la variable controlada en el punto de consigna. Hay que tener en cuenta que un tiempo integral muy pequeño podría inestabilizar el sistema de control diseñado.

La parte D, hace referencia al tiempo derivativo, el cual es un tiempo requerido para que la acción proporcional contribuya a la salida del control en una cantidad igual a la acción derivativa. Un tiempo derivativo pequeño implica una acción derivativa pequeña, sin embargo, de ser grande este tiempo implicaría una acción derivativa grande. La acción derivativa anticipa el efecto de la acción proporcional para estabilizar más rápidamente la variable controlada después de cualquier perturbación.

Los tres parámetros pueden incluirse en una ecuación general que describiría la expresión matemática de un controlador PID teórico. Dicha ecuación es la que se muestra a continuación (figura 3.10):

$$u(t) = K_p e(t) + \frac{K_p}{Ti} \int_0^t e(\tau)\, d\tau + K_p T_d \frac{de(t)}{dt}$$

Figura 3.10. Ecuación matemática de un controlador PID

Con la anterior ecuación se puede observar que, si se considerara un tiempo integral bastante grande, el controlador dejaría de tener la acción integral en su diseño. En la experimentación se concebirá el fenómeno que produce el tiempo para sistemas con carga de ruido, en donde la respuesta se verá afectada cuanto mayor sea el ruido del sistema que se pretende estudiar ya que su estabilidad cambiará empeorándose. Otro de los comentarios que puede añadirse es que la constante proporcional acompaña a los tres operandos que se puede ver en la ecuación.

En la tabla siguiente se puede ver como se interrelacionan estos tres parámetros en lo que respecta el diseño del controlador PID, afectando en su estabilidad, velocidad de respuesta, error estacionario, entre otras.

	Kp aumenta	Ti disminuye	Td disminuye
Estabilidad	Se reduce	Disminuye	Aumenta
Velocidad	Aumenta	Aumenta	Aumenta
Error estacionario	No Eliminado	Eliminado	No Eliminado
Área del error	Se Reduce	Disminuye hasta cierto punto	Se reduce
Perturbación control	Aumenta bruscamente	Aumenta Gradualmente	Aumenta muy bruscamente

A continuación, se procederá a exponer la forma en que están dispuestos cada uno de estos tres parámetros del controlador en el sistema en lazo cerrado. El bloque encerrado por la línea discontinua representará el PID analizado, que en este sistema se encuentra colocado en serie (figura 3.11):

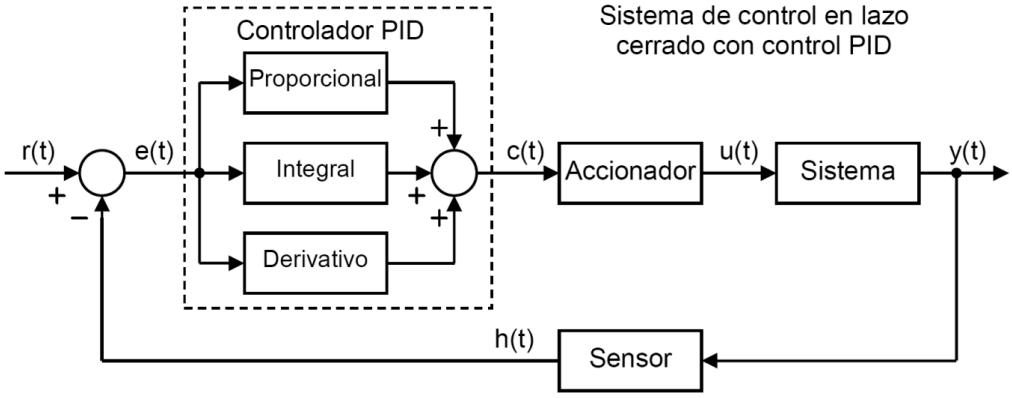


Figura 3.11. Colocación parámetros del controlador PID en un lazo cerrado.

A continuación, se expondrá los diferentes métodos de sintonía que existen para poder diseñar un controlador automático en base a los datos calculados de la experimentación ya sea en lazo abierto o cerrado:

3.4. MÉTODOS DE SINTONÍA DE PIDS:

En este capítulo se desarrollarán los métodos de sintonía que existen en función de los parámetros de las dos experimentaciones (Lazo Abierto y Lazo Cerrado) que se dispongan. Como se mencionó anteriormente, la finalidad de estas sintonías es encontrar el óptimo diseño del control automático PID. Cada una de las sintonías posee un compendio de ecuaciones matemáticas para la obtención de la ganancia proporcional, tiempo integral y el tiempo derivativo, siendo este último el que será más sensible a ruidos que pueda tener la planta durante el diseño. Por ello y debido a la naturaleza de la planta que se detallará en futuros capítulos, el valor numérico del tiempo derivativo se pone a cero, en las posteriores configuraciones manuales para mejorar los obtenidos con las ecuaciones establecidas por la teoría de sintonización. Los métodos de sintonía serán los siguientes:

- Método sintonía en lazo abierto:
  - Ziegler Nichols.
  - Rovira.
  - Cohen Coon.
  - Lopez y otros.
- Método sintonía en lazo cerrado:
  - Ziegler Nichols.

Antes de entrar en los tipos de sintonía que existen se procederá a definir los parámetros que se pretenden diseñar del controlador.

Como se había comentado el controlador PID viene determinado por tres parámetros: el proporcional, el integral y el derivativo. Dependiendo de la modalidad del controlador alguno de estos valores puede ser nulo, por ejemplo, un controlador proporcional, denominado tipo P, tendrá el tiempo integral infinito y el tiempo derivativo a 0, un controlador tipo PI solo el derivativo será 0, etc.

Cada uno de estos parámetros influye en mayor medida sobre alguna característica de la salida (tiempo de establecimiento, sobreoscilacion, …) pero también influye sobre las demás, por lo que por mucho que ajustemos no encontraríamos un PID que redujera el tiempo de establecimiento a 0, la sobreoscilacion a 0, el error a 0, … sino que se trata más de ajustarlo a un término medio cumpliendo las especificaciones requeridas.

A continuación, se detallará cómo habrá que sintonizar cada uno de los parámetros del PID para conseguir un sistema de control automatizado. Recuérdese lo que se comentó en el apartado anterior, cuando se introdujeron los conceptos de constante proporcional, tiempo integral y tiempo derivativo:

- Acción proporcional:

La respuesta proporcional es la base de los tres modos de control, si los otros dos, control integral y control derivativo están presentes, éstos son sumados a la respuesta proporcional. “Proporcional” significa que el cambio presente en la salida del controlador es algún múltiplo del porcentaje del cambio en la medición. Este múltiplo es llamado “ganancia” del controlador. Para algunos controladores, la acción proporcional es ajustada por medio de tal ajuste de ganancia, mientras que para otros se usa una “banda proporcional”. Ambos tienen los mismos propósitos y efectos. Para efectuar su sintonía, correctamente se deberá:

  - Poner el tiempo integral (TI), a su máximo.
  - Tiempo derivativo (TD), a su mínimo valor.
  - Empezando con una ganancia muy pequeña se va aumentando hasta obtener las características de respuesta deseadas.

- Acción integral:

La acción integral da una respuesta proporcional a la integral del error. Esta acción, como ya se había comentado, elimina el error en régimen estacionario, provocado por el modo proporcional. Por contra, se obtiene un mayor tiempo

de establecimiento, una respuesta más lenta y el periodo de oscilación es mayor que en el caso de la acción proporcional. Para poder sintonizar este tiempo se procederá con los siguientes pasos:

- Reducir el tiempo integral (TI) hasta anular el error en estado estacionario, aunque la oscilación sea excesiva.
- Disminuir ligeramente la ganancia hasta obtener las características de respuesta deseadas.
- Repetir hasta obtener las características de respuesta deseadas.

- Acción derivativa:

La acción derivativa da una respuesta proporcional a la derivada del error (velocidad de cambio del error). Añadiendo esta acción de control a las anteriores se disminuye el exceso de sobre oscilaciones. Para poder sintonizar este tiempo se procederá del modo que se expone a continuación:

- Mantener ganancia (P) y tiempo integral (TI), obtenidos anteriormente.
- Aumentar el tiempo derivativo (TD), hasta obtener características similares, pero con la respuesta la respuesta más rápida.
- Aumentar ligeramente la ganancia (P) si fuera necesario.

Existen diversos métodos de ajuste para controladores PID, pero ninguno de ellos nos garantiza que siempre encuentre un PID que haga estable el sistema. Por lo que el más usado sigue siendo el método de prueba y error, probando parámetros del PID y en función de la salida obtenida variando estos parámetros. Dichos métodos serán los siguientes, clasificados según que experimentación se haya realizado previamente; en lazo abierto o en lazo cerrado.

3.4.1. MÉTODO SINTONÍA EN LAZO ABIERTO (L.A.)

Para el método de sintonía en lazo abierto, como se mencionó líneas más arriba, se tienen varios tipos de métodos que se expondrán a continuación:

- Método de Ziegler Nichols:

Este método permite combinar los tres parámetros de la experimentación en lazo abierto de forma que se pueda diseñar un controlador de tipo P, PI y PID serie. Téngase en cuenta que, si la planta no contiene integradores ni polos dominantes complejos conjugados, la curva de respuesta escalón unitario puede tener forma de S (Véase Figura 3.4). Si la respuesta no exhibe una curva con forma de S, este método no es pertinente. Tales curvas de respuesta

escalón se generan experimentalmente o a partir de una simulación dinámica de la planta. Las combinaciones, que antes se habían mencionado, se muestran en la siguiente tabla según qué tipo de controlador se desee diseñar:

Tipo	Ganancia K_p	Tiempo integral	Tiempo derivativo
P	τ / (K d)		
PI	0.9τ /(K d)	3.33 d	
PID serie	1.2τ /(K d)	2 d	0.5 d

Figura 3.12. Tabla sintonía Ziegler Nichols Lazo Abierto.

- **Método de Rovira:**

En el método de Rovira, a diferencia del anterior, los controladores que se diseñan son de tipo paralelo o no iterativo exclusivamente controladores PI y PID. Se diseñan para sistemas con cambios de consigna. Lo que se pretende con ellos es minimizar la integral del error |e| en el caso de la configuración MIAE y |e|· t para el caso de la configuración de MITAE. Los modelos que se diseñan con este método tienen que ser procesos monótonos, es decir, el cociente del valor del retardo entre la constante de tiempo sea siempre menor que la unidad (osea, d/Tau<1), siendo ambos valores los que se extraen de la curva de reacción ante una entrada escalón. En la siguiente tabla (figura 3.12) se proporcionan una serie de valores para las constantes “a” y “b” correspondientes a los criterios de sintonía MIAE y MITAE:

PI paralelo

Criterio	Proporcional	Integral	Derivativo
MIAE	a=0.758 b=-0.861	a=-0.323 b=1.020	
MITAE	a=0.586 b=-0.916	a=-0.165 b=1.030	

PID Paralelo

MIAE	a=1.086 b=-0.869	a=-0.130 b=0.740	a=0.348 b=0.914
MITAE	a=0.965 b=-0.855	a=-0.147 b=0.796	a=0.308 b=0.929

$$K_p K = a\left(\frac{d}{\tau}\right)^b$$

$$\frac{\tau}{T_i} = a\left(\frac{d}{\tau}\right) + b$$

$$\frac{T_d}{\tau} = a\left(\frac{d}{\tau}\right)^b$$

Figura 3.13. Tabla sintonía Rovira Lazo Abierto.

- **Método de Cohen Coon:**

En el método de Cohen Coon lo que propone es una variante del método de Ziegler Nichols en lazo abierto. Dado que la curva de reacción, según el ajuste de Ziegler Nichols, es muy sensible para valores que corresponden con el cociente entre el retardo y la constante de tiempo (es decir, d/Tau). Ante esta limitación Cohen y Coon desarrollaron una tabla modificada (ver figura 3.13) para mejorar esta limitación empleando datos del mismo ensayo:

Tipo de Controlador	Ganancia K_c	Tiempo Integral T_i	Tiempo Derivativo T_c
P	$\frac{\tau}{Kd}\left(1+\frac{d}{3\tau}\right)$		
PI	$\frac{\tau}{Kd}\left(0.9+\frac{d}{12\tau}\right)$	$d\frac{30+3d/\tau}{9+20d/\tau}$	
PID	$\frac{\tau}{Kd}\left(1.333+\frac{d}{4\tau}\right)$	$d\frac{32+6d/\tau}{13+8d/\tau}$	$d\frac{4}{11+2d/\tau}$

Figura 3.14. Tabla sintonía Cohen Coon Lazo Abierto.

Empleando esta tabla se podrá hacer que la respuesta sea más homogénea que la que proporcionaría Ziegler Nichols, a pesar de que aun sea ligeramente sensible al cociente d/Tau.

- **Método de López y Otros:**

Para el método de López y Otros, el diseño estará destinado para controladores PID en paralelo, con la misión de rechazar perturbaciones y minimizar el error. Y al igual que en la sintonía de Rovira se tienen las configuraciones de MIAE como |e| y MITAE como |e|·t. Pero en este método se tiene una modalidad más, denominada MISE que sería e^2. En este método también se disponen de dos parámetros “a” y “b” que tendrán unos valores determinados para cada configuración que se pretenda fijar, además, de la exigencia que se establece acerca del cociente del retardo entre la constante de tiempo de la curva de reacción, en lazo abierto, ante una entrada escalón, que debe ser menor que la unidad (d/Tau<1) debido a ser válido para procesos monótonos. En la siguiente tabla (figura 3.14) se puede observar las diversas sintonías que se pueden obtener en función de los valores de los parámetros “a” y “b” y de la elección que se tome en el diseño de un controlador de tipo PI o PID, ambos paralelos.

Reguladores PI paralelo

Criterio	Proporcional	Integral	Derivativo
MIAE	a=0.984 b=-0.986	a=0.608 b=-0.707	
MISE	a=1.305 b=-0.959	a=0.492 b=-0.739	
MITAE	a=0.859 b=-0.977	a=0.674 b=-0.68	

Reguladores PID paralelo

Criterio	Proporcional	Integral	Derivativo
MIAE	a=1.435 b=-0.921	a=0.878 b=-0.749	a=0.482 b=1.137
MISE	a=1.495 b=-0.945	a=1.101 b=-0.771	a=0.560 b=1.006
MITAE	a=1.357 b=-0.947	a=0.842 b=-0.738	a=0.381 b=0.995

$$K_p K = a\left(\frac{d}{\tau}\right)^b$$

$$\frac{\tau}{T_i} = a\left(\frac{d}{\tau}\right) + b$$

$$\frac{T_d}{\tau} = a\left(\frac{d}{\tau}\right)^b$$

Figura 3.15. Tabla sintonía López y Otros Lazo Abierto.

3.4.2. MÉTODO SINTONÍA EN LAZO CERRADO (L.C.)

Este método de sintonía de Ziegler Nichols en lazo cerrado es un método más preciso que el de lazo abierto, pero requiere de experimentaciones de prueba y error para alcanzar su optimización. Su sintonía puede inestabilizar el sistema y comprometer su integridad, pero ello depende de la experimentación en lazo cerrado que se había estudiado del sistema previamente, es decir, en el cálculo de la ganancia crítica, Kc y del período crítico, Tc.

Recuérdese que la ganancia crítica se obtenía conectando el regulador en modo "P", es decir, con el tiempo integral, Ti, de valor infinito y tiempo derivativo, Td, con valor a cero.

Posteriormente, se va aumentando la ganancia de manera que el sistema comience a responder con crecientes oscilaciones hasta un punto en que la respuesta se convertía en oscilatorio uniforme, es decir, con un período definido y una amplitud constante. En ese instante la ganancia que provocaba dicha respuesta se convertía en ganancia crítica, Kc, por encima de la cual el sistema comenzaría a oscilar crecientemente mostrando una inestabilidad evidente.

Si se mantenía esa ganancia crítica, podía extraerse el valor del período que la respuesta oscilante tendría, siendo ésta la que representaría el período crítico, Tc.

Para el diseño de controladores de tipo P, PI, PID serie y paralelo se deberán conocer la ganancia crítica y el período crítico calculados anteriormente. En la siguiente tabla de podrán contemplar las ecuaciones de los mencionados controladores que se pueden diseñar (figura 3.14):

Tipo	Ganancia K_p	Tiempo integral	Tiempo derivativo
P	$0.5\ K_c$		
PI	$0.45\ K_c$	T/1.2	
PID paralelo	$0.75\ K_c$	T/1.6	T/10
PID serie	$0.6\ K_c$	T/2	T/8

Figura 3.16. Tabla sintonía Ziegler-Nichols en Lazo Cerrado.

Ahora que se han expuesto todos los conceptos teóricos que han sido empleados en el diseño del presente software del proyecto, en el siguiente capítulo se procederá a integrar lo detallado en el anterior y en éste, en referente al programa LabView utilizado y a la teoría de diseño de sistemas de control, respectivamente.

Ambos capítulos mencionados permitirán comprender como están programados los bloques que se contemplarán en la arquitectura del software pudiendo visualizar las tareas de control que se han analizado en este capítulo en base al lenguaje de programación visual que emplea LabView; empezando por la calibración de los datos recibidos, la realización de las experimentaciones en lazo abierto y cerrado y, por último, los procedimientos para sintonizar un controlador PID determinado.

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

IV – DESARROLLO DEL SOFTWARE

TRABAJO FIN DE GRADO – KEN VERA CHAN
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

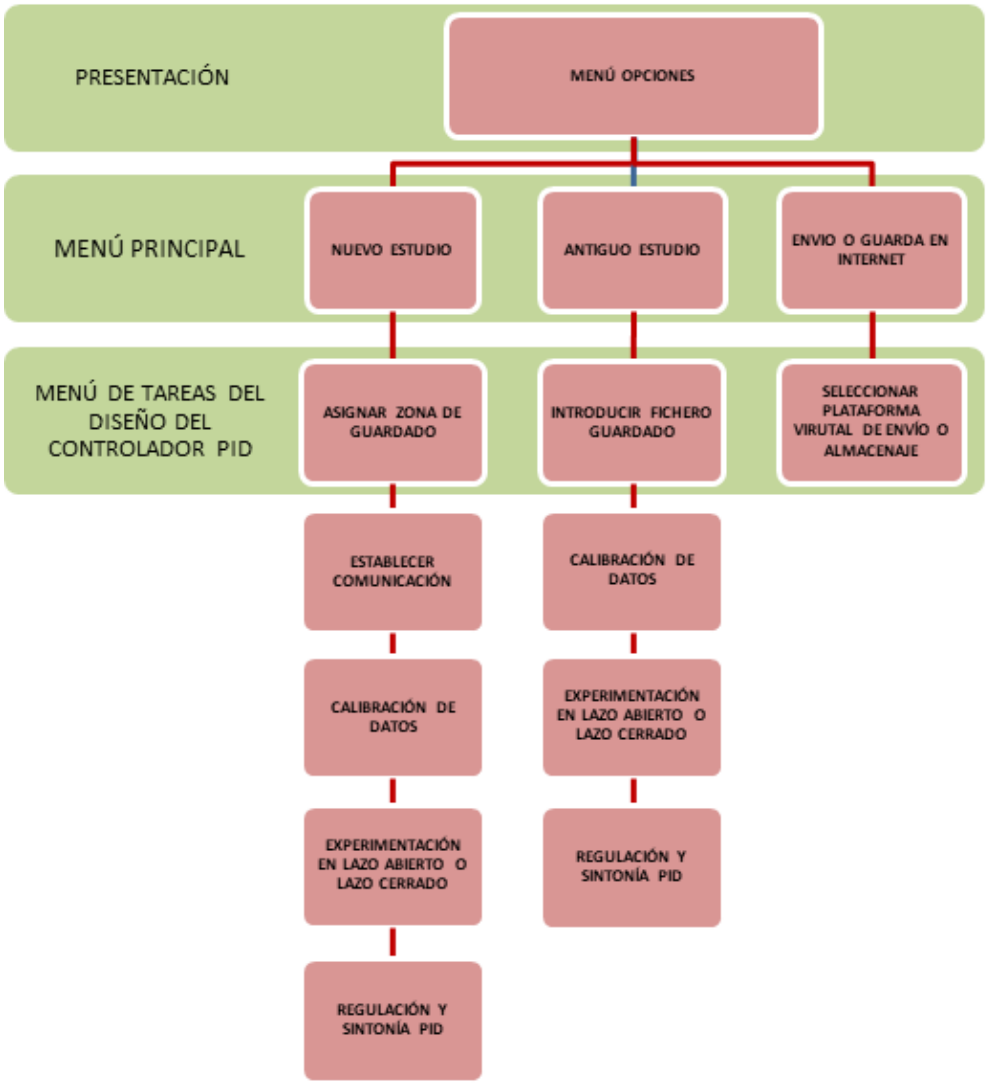
INDICE DE CONTENIDOS:

4. DESARROLLO DEL SOFTWARE

4.1. MANUAL DEL USUARIO

Aquí se expondrán todos los puntos por los cuales pasará el usuario motivado por la realización y diseño de un controlador PID. Como se mencionó en la imagen de la figura 4.1. sobre el panel principal de opciones disponibles para iniciar uno de ambos estudios, el usuario deberá elegir entre las dos que se ofrecen para comenzar el diseño.

Para comprender los puntos importantes que se recorrerán a lo largo de todo el software se incluirá un amplio esquema referente a todas las tareas que se proponen para poder diseñar un controlador PID concretando que puntos son los más influyentes en este cometido:



En el esquema anterior se pueden ver todas las tareas que se han programado para que el usuario elija que camino a seguir en función, primeramente, de si fue un estudio interrumpido o nuevo el que tenga intención de reanudar o iniciar respectivamente. Téngase en cuenta que el software puede dividirse en dos grandes bloques basados en un nuevo estudio con todas las opciones que se han comentado y en especial el registro del fichero XML vacío a emplear, así como el de antiguo estudio con igual similitud de opciones que el anterior, aunque en especial con el registro del correspondiente fichero XML que se leerá para extraer la información que posea en su interior.

Cabe pensar que las tareas correspondientes a la calibración, a ambas experimentaciones y a la sintonía del controlador PID se comparten en operatividad, todas las funciones que en ellas se ejecutan y en el siguiente capítulo dedicado al programador se verá con certeza este hecho para optimizar la carga de procesamiento y mejorar el rendimiento del software.

Como se comentó el usuario tiene la posibilidad de realizar un nuevo estudio o modificar alguno antiguo además de enviar algún fichero XML generado como consecuencia de haber accionado cualquiera de ambas opciones. Además de ello se proporcional el accionamiento común en todos los programas que permite abandonar la ejecución y salir del programa en cuestión. Como se puede ver en la imagen mostrada a continuación (figura 4.1.) se ha ejecutado el programa iniciándose en el menú principal, observándose que el botón de envío del fichero o almacenaje en internet esta deshabilitado:

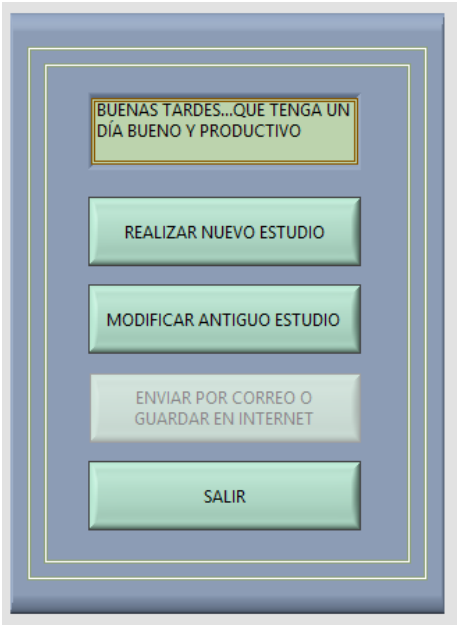


Figura 4.1. Interfaz usuario panel principal de opciones.

4.1.1. PROGRAMA DE EJECUCIÓN NUEVO ESTUDIO:

Seguidamente, cuando el usuario haya accionado el comienzo con el programa basado en la realización de un nuevo estudio lo que se le propondrá serán una serie de accionamientos de los cuales sólo el primero de ellos estará habilitado ya que previamente se ha de conocer el fichero XML que se tendrá que utilizar para guardar todos los datos generados.

Figura 4.2. Interfaz usuario panel de opciones realización nuevo estudio.

Obsérvese, además, que el panel que se muestra en la imagen anterior (figura 4.2.) posee un panel donde se registrará la información que se obtendrá de las diferentes tareas en tiempo real.

4.1.1.1. ASIGNAR ZONA DE GUARDADO:

En este panel, el usuario designará una zona en el explorador de Windows en donde creará, abrirá y editará el fichero XML. En el panel de opciones se podrá ver como el usuario puede abandonar la operación antes de iniciarla, accionando el botón de “volver” o reanudar su cometido escribiendo un nombre que se le otorgará al futuro fichero XML, que cuando esté conforme con ello podrá accionar el botón de “aceptar nombre” en donde acto seguido, se abrirá una ventana para poder facilitar el acceso a la ruta en donde querría alojar el documento en cuestión.

Si el usuario está conforme con el nombre asignado se podrá ver en los espacios inferiores la URL del explorador en donde se encontrará dicho fichero además del nombre de la carpeta en donde se encontrará. Todo ello se podrá visualizar en la imagen que se proporciona a continuación (figura 4.3.):

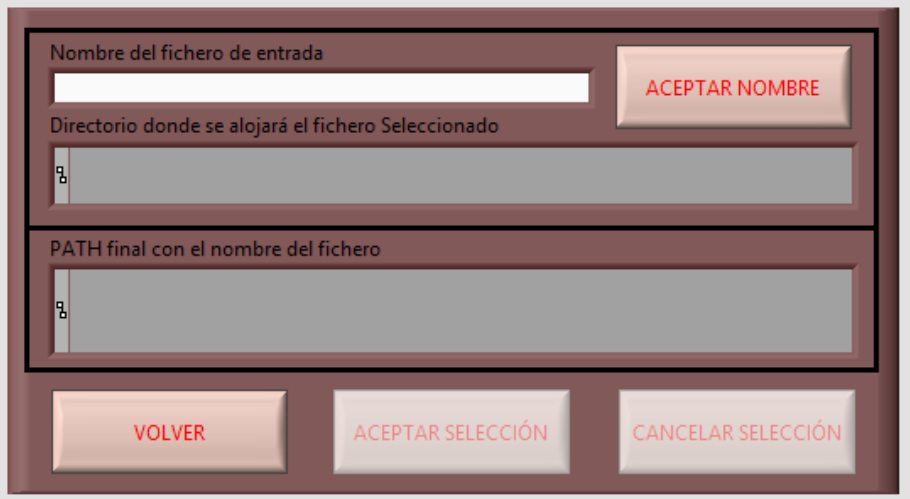


Figura 4.3. Interfaz usuario panel de asignación zona de guardado.

Si el usuario está satisfecho con los resultados de asignación de la zona de guardado y el nombre del fichero, podrá accionar el botón de “aceptar” y proseguir con las siguientes tareas. En otro caso accionará el botón de “cancelar” y saldrá igualmente al menú anterior, pero seguirán estando inhabilitadas todas las demás opciones correspondientes con las tareas de calibración, experimentación y sintonía del controlador PID.

4.1.1.2. ESTABLECER COMUNICACIÓN:

Al entrar en el panel de comunicación mediante un servidor OPC, el usuario tendrá la opción de abandonar la realización de la tarea accionando el botón de “volver” o iniciar con ella, pulsando el botón de “seleccionar URL de lectura” que es una de las que serán necesarias para poder continuar con las tareas sucesoras. La URL de lectura se encargará de leer los datos recibidos por el sensor instalado en la planta de estudio, mientras que la URL de escritura, la que se solicitará posteriormente, será aquella que se encargará de la variable manipulada del sistema. En la siguiente ilustración (figura 4.4.) se podrá ver el panel correspondiente a esta tarea:

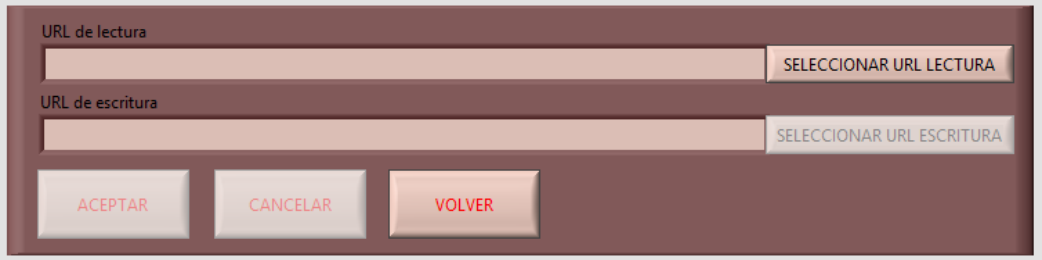


Figura 4.4. Interfaz usuario panel de comunicación.

Cuando se pretenda encontrar las URLS correspondientes a la lectura y escritura se podrá observar la siguiente ventana que mostrará los servidores OPC disponibles en esos instantes (Véase figura 4.5.):

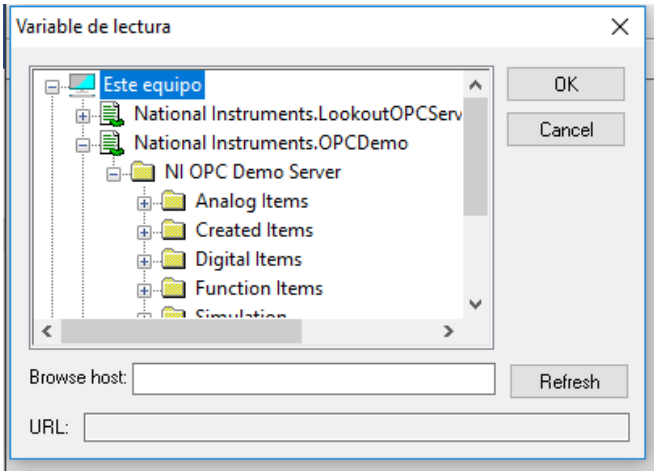


Figura 4.5. Interfaz usuario ventana de servidores OPC.

Al elegirse las respectivas URLs conseguidas de la ventana emergente de Windows para la selección, se obtendrá el siguiente panel (Véase figura 4.6.), otorgando al usuario la decisión de "aceptar" o "cancelar" la operación ejecutada. En el caso de "cancelar" la operación el sistema no podrá continuar con las posteriores tareas al no existir las correspondientes URLS que se usarían para leer del sensor y escribir sobre la variable manipulada.

Figura 4.6. Interfaz usuario panel de comunicación con URLS de lectura y escritura asignados.

4.1.1.3. CALIBRACIÓN DE DATOS:

Si se han concretado las URLS en la tarea anterior, en ésta se deberá calibrar los datos de entrada recibidos por el sensor de medida para poder convertirlos a unidades ingenieriles de la medida que se pretende estudiar en las posteriores tareas. El panel que el usuario se encontrará dividido en dos partes (figura 4.7. y figura 4.8.) para poder facilitar la explicación que se expondrá próximamente, pero en lo que respecta la interfaz diseñada en LabView, ambas imágenes estarán unidas formando una unidad para el análisis de los datos calibrados:

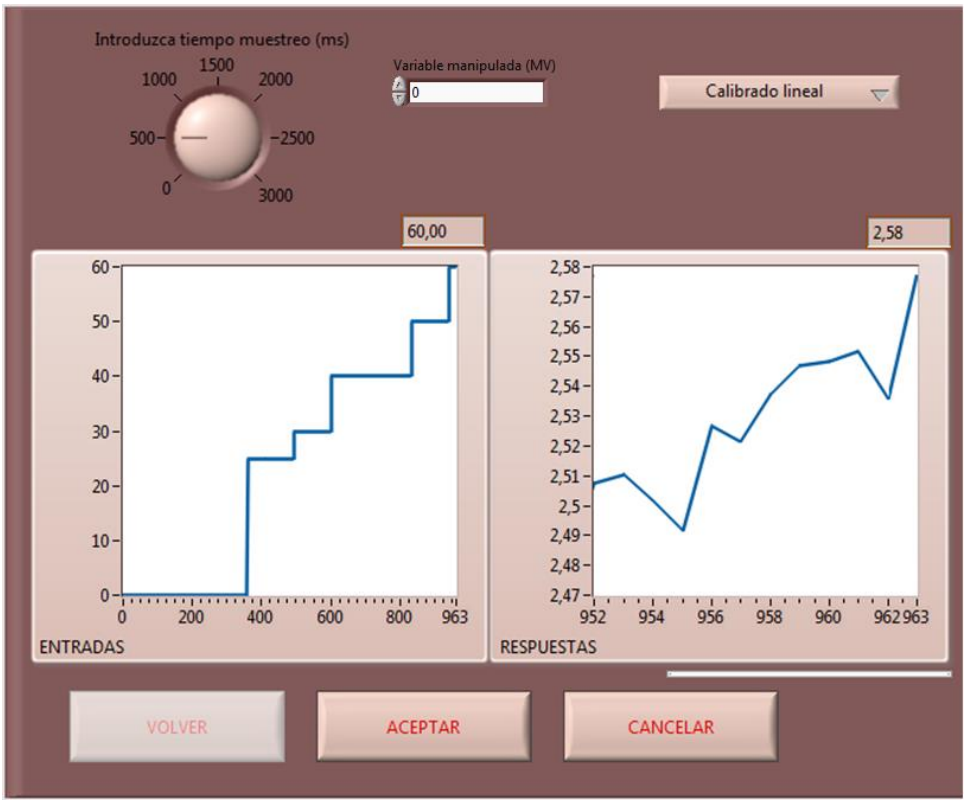


Figura 4.7. Interfaz usuario panel de entradas salto, respuestas del sistema, tiempo de muestreo y tipo de calibración de datos.

Para la primera de las imágenes (figura 4.7.) se puede observar que el usuario dispone de un tiempo de muestreo, el cual puede variar entre 0 y 3000 milisegundos, sirviéndole al usuario para sistemas que posean un período de obtención de datos inferior o superior a 500 milisegundos o 1000 milisegundos que sería un período de muestreo común para sistemas convencionales.

Otro de los aspectos es el tipo de calibración que se dispone para poder realizar la tarea correspondiente: Calibrado lineal, polinómica, potencia, exponencial y logarítmica. La primera de ellas es la que se ha usado en este proyecto, pero

en cualquiera de los casos se pueden emplear las restantes según convenga la planta de procesos.

La gráfica de la izquierda representa las entradas de tipo escalón que se podrán dar al sistema durante la calibración de los datos, siendo la gráfica de la derecha la correspondiente a la respuesta que presenta el sistema no calibrada, pero es útil para que el usuario pueda almacenar estos puntos concretos para su posterior calibrado.

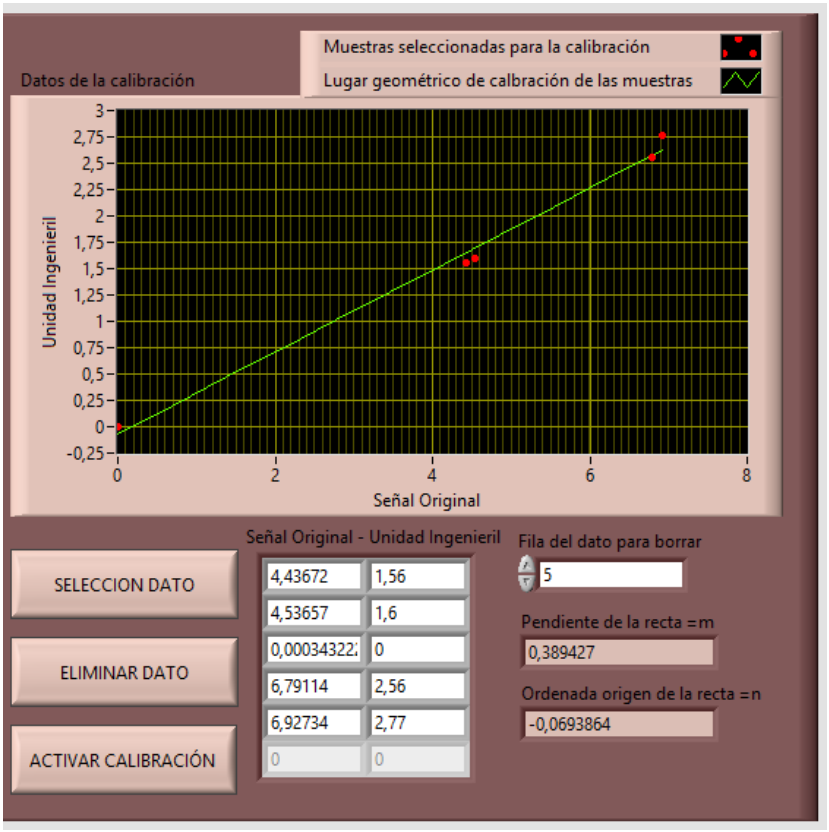


Figura 4.8. Interfaz usuario panel de respuesta calibrada, recolección de los puntos para la calibración y parámetros de la calibración obtenida.

En la imagen anterior (figura 4.8.) se puede ver la gráfica de una prueba que se ejecutó con éxito, obteniendo una calibración lineal como puede observarse en color verde sobre fondo negro con puntos rojos referentes a los datos no calibrados que se escogieron para el calibrado. Además de todo eso, se verán los datos que han sido seleccionados para la operación que esta tarea exige. Cada vez que se añada un dato se añadirá una fila vacía para que la matriz de los datos pueda contener todo lo que se pretenda analizar posteriormente, sin embargo, si se hubiera introducido un dato por error sea la razón que sea con los accionamientos que se disponen justo a la izquierda de la matriz de datos permitirán corregir la selección y volver a elegir de nuevo aquellos datos que se

consideren oportunos. Así, al final de dicha tarea, el usuario podrá ver los parámetros de la calibración que se hayan conseguido en función de los datos seleccionados y del tipo de calibrado impuesto.

Si está conforme con la calibración ejecutada, el usuario podrá accionar el botón de "aceptar" y ver que os resultados se guardarán en el fichero cuando vea posteriormente, en el panel de información los parámetros registrados.

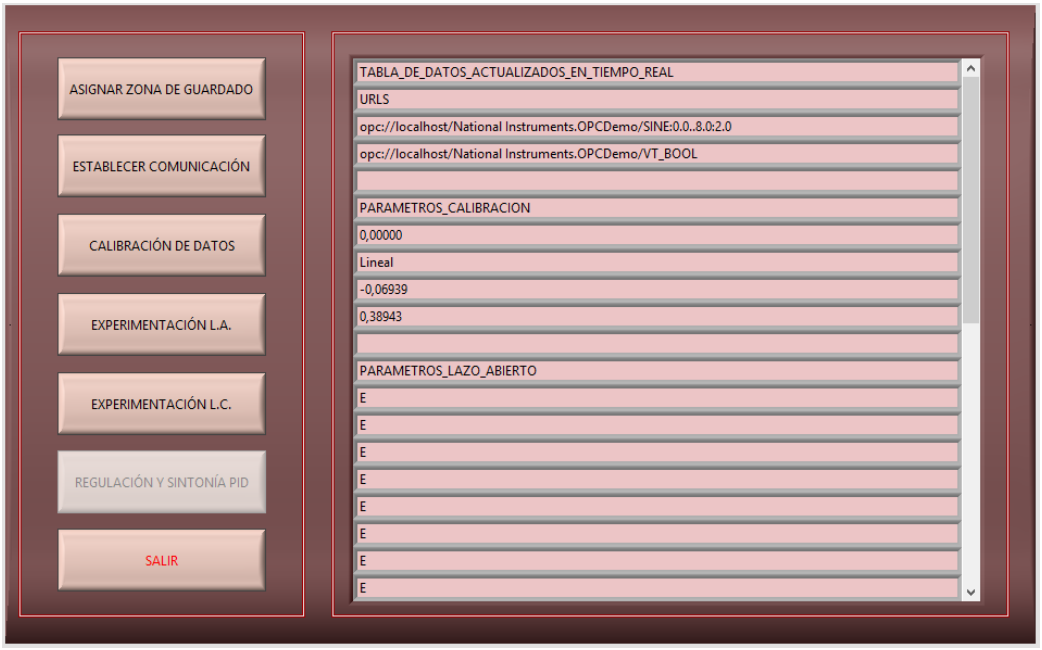


Figura 4.9. Interfaz usuario panel de datos actualizados.

Como se comentó (Véase la figura 4.9.) el usuario verá en el panel de información generada los datos que se han actualizado que serán los mismos que se podrán ver en el fichero XML que se creó para almacenar el progreso obtenido. Como se comentó al principio el panel adjunto aparecerá con unos títulos para que el usuario sepa a qué tarea pertenecen.

Si no se ha ejecutado una tarea, la casilla correspondiente a la colocación del dato que generaría ésta, se encontrará sustituida por una letra "E" cuyo significado será "EMPTY" referente a valor no encontrado o casilla vacía del inglés.

4.1.1.4. EXPERIMENTACIÓN EN LAZO ABIERTO:

Tras la calibración, el usuario podrá adentrarse en cualquiera de ambas experimentaciones que se proponen; en lazo abierto y en lazo cerrado, téngase en cuenta que, como se mencionó en apartados anteriores, el controlador PID

no necesita tener ambas experimentaciones realizadas más con una de ellas bastaría para el diseño expreso.

Para el caso de la experimentación en lazo abierto, se tendrán en cuenta el empleo de las URLs de lectura y escritura además de la calibración hecha en el apartado anterior, ya que, con los datos convertidos en unidades de unidades de ingeniería, el proceso será más eficiente de cara a la obtención de los parámetros de esta experimentación. Como se puede ver a continuación (figura 4.10.) se han hecho uso de tres gráficas, de las cuales la primera situada en la zona superior izquierda de la imagen corresponde con las entradas de tipo escalón que se introducirán para generar con ello, la curva de reacción que tendrá el sistema estudiado. La gráfica de la derecha en la imagen, zona superior, serán las respuestas calibradas que el sistema produce como consecuencia de los escalones introducidos.

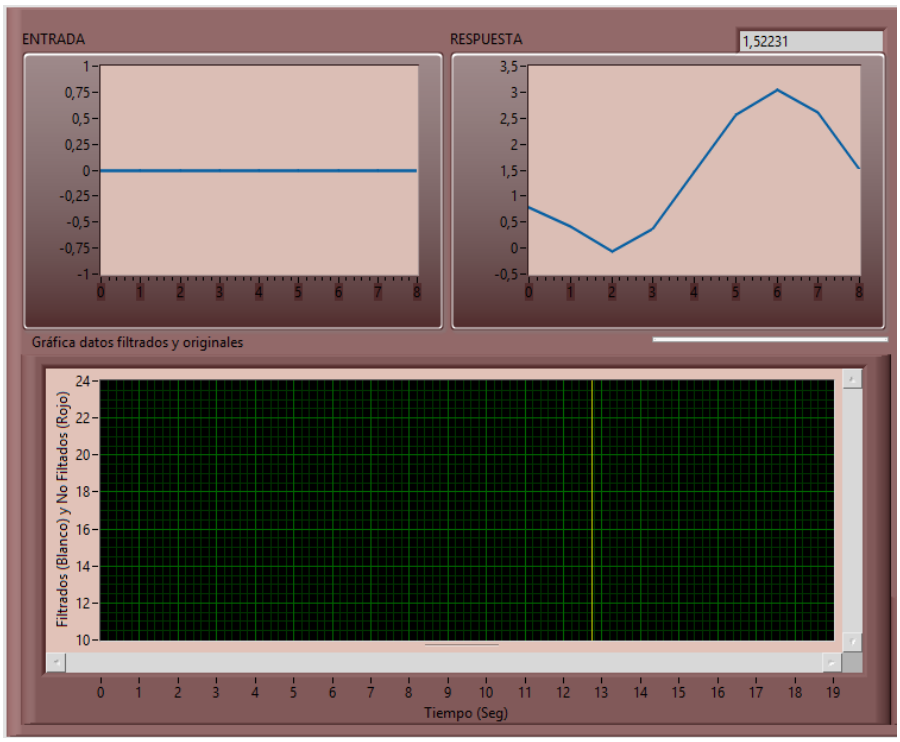


Figura 4.10. Interfaz usuario panel de gráficas; entrada salto, respuesta calibrada y curva de reacción de la experimentación en lazo abierto.

La gráfica inferior con fondo oscuro será la encargada de representar la curva de reacción, la cual se explicó en profundidad en el capítulo tercero sobre "fundamentos teóricos", así como la manera en que se podrán extraer los parámetros en lazo abierto según como quede representada la curvatura. El usuario podrá definir previamente el tiempo de muestreo que será la más efectiva según qué sistema se esté analizando. Además de ello podrá generar

las correspondientes entradas escalón para poder construir la curva de reacción de la que luego se analizarán sus parámetros.

Todo el compendio de gráficas vendrá definido por los datos que se introducirán y se generarán en la siguiente parte del panel fragmentado (figura 4.11.) ya que cuando se haya ejecutado la tarea que en esta subrutina se encomienda, el usuario podrá hacer uso de un cursor que habilitado, se podrá desplazar de izquierda a derecha y de la zona superior hacia la inferior para poder confirmar los parámetros obtenidos desde la propia gráfica obtenida. Dichos parámetros de esta experimentación quedarán reflejados bajo los títulos que los definen, siendo los valores de fondo sombreado los obtenidos de la curva de reacción y los de fondo claro los obtenidos de implementar los saltos en la entrada:

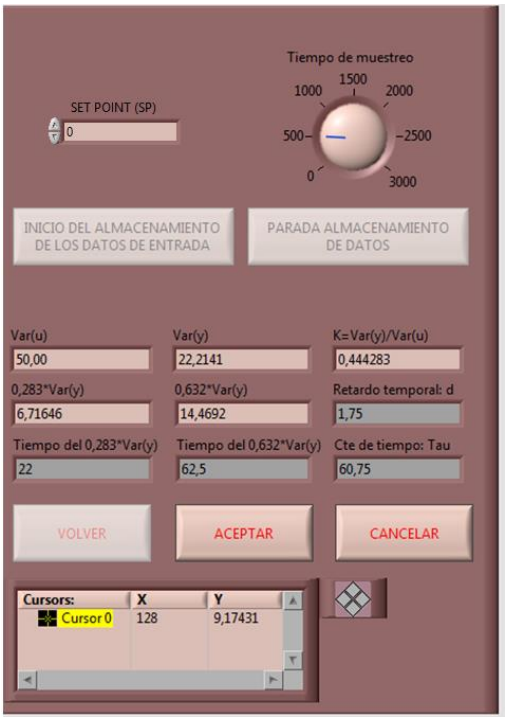


Figura 4.11. Interfaz usuario panel de generación de los parámetros de la experimentación en lazo abierto junto con las entradas escalón.

En la imagen anterior se puede ver que el usuario podrá almacenar los datos calibrados que considere oportuno para poder ejecutar la experimentación en lazo abierto. Introducirá las entradas escalón que vea oportunas para poner al sistema en una preparación previa de cara a iniciar con la experimentación. De manera que podrá imponer los saltos que desee y de la magnitud que necesite para su posterior análisis. Justo cuando se sienta conforme con el almacenamiento de los datos podrá detener el proceso y contemplar la curva de reacción generada, los parámetros de la experimentación calculados y por

supuesto los accionamientos de “aceptación” o “cancelación”, momento en que se decidirá si guardar o no lo obtenido en el fichero XML.

4.1.1.5. EXPERIMENTACIÓN EN LAZO CERRADO:

En el caso de la experimentación en lazo cerrado, el usuario podrá obtener los dos parámetros correspondientes a esta tarea comprobando la evolución del sistema mediante las gráficas que se muestran a continuación (figura 4.12.) En ellas se podrá contemplar la respuesta que proporciona la planta de proceso cuando se cambia el valor numérico del setpoint. Se verá el efecto causado sobre la variable controlada que aparece en la misma gráfica que en la de las entradas salto. La evolución de la variable controlada quedará impuesta por el cambio automático del valor de la variable manipulada que intentará mantener el valor numérico establecido por la referencia introducida en el sistema. Téngase en cuenta que como ya se había comentado en anteriores ocasiones, el usuario calculará primeramente la ganancia crítica pudiendo conseguir que la respuesta que genere la variable controlada sea oscilatoria con una ganancia y un período definido, aumentándola hasta conseguir tal hecho momento en que, si se incrementa más su valor, el sistema respondería con oscilaciones crecientes adentrándose en un estado de inestabilidad evidente.

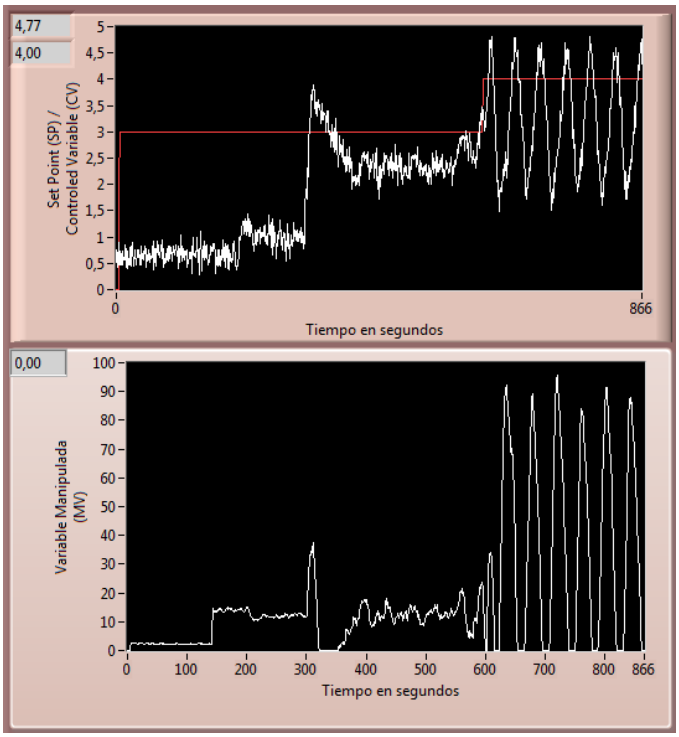


Figura 4.12. Interfaz usuario de gráfica superior de las variables controladas y setpoint y gráfica inferior variable manipulada.

Para poder realizar las tareas que en esta sección se encomienda el usuario dispondrá de un tiempo de muestreo como en todas las anteriores tareas programadas. Este tiempo de muestreo deberá considerarse el mismo que se decidió colocar para las tareas precedentes por recomendación con el fin de que los parámetros no queden falseados por los distintos tiempos de muestreo asignados. Seguidamente, el usuario podrá hacer uso del control numérico que corresponde con la ganancia introducida en el sistema junto con el valor de las entradas escalón que se podrán establecer para conseguir ejecutar correctamente la experimentación en este tipo de lazo cerrado.

Como en la ocasión anterior, se podrá almacenar un conjunto de datos que el usuario considere con el fin de calcular la ganancia crítica. El intervalo seleccionado estará fijado por los accionamientos “almacenar gráfica de la señal de entrada” como inicio del almacenamiento y “parada de almacenaje gráfica señal de entrada” como el final de la recolección de los datos.

Si el usuario decide accionar el último de los accionamientos anteriores podrá ver como el sistema se detiene por completo generando en el gráfico de la imagen siguiente (figura 4.13.) el resultado de la recolección de los datos que quedaron dentro del intervalo que impuso el usuario con el juego de botones que permite iniciar y finalizar el almacenamiento de los datos.

Si el usuario decidió detener el sistema es de suponer que consiguió que el sistema mostrara una respuesta oscilante, como se argumentó líneas más arriba, instante en que la ganancia que impuso se convertirá en crítica; el primero de los parámetros que se pretendió hallar en esta experimentación.

Posteriormente, el usuario verá que además de detenerse el sistema, podrá ver la respuesta oscilatoria y llegados a este punto podrá calcular el correspondiente período crítico que existirá entre dos de las crestas o valles que se pueda visualizar de la gráfica adjunta.

Para poder comprender el modo en que se calcula dicho período, el usuario deberá fijarse en la gráfica que se proporciona en la imagen (figura 4.13.) en donde el eje de abscisas (eje X) representan los segundos que se tienen representados de la curva de la respuesta del sistema en el intervalo que se obtuvo del almacenamiento de los datos anteriores. Al disponer de dos cursores, uno rojo y otro azul, el usuario podrá colocar ambos de manera que estén a la misma altura (o coordenada en el eje Y) de manera que sobre la curva de la respuesta oscilatoria podrá ver el período calculado como resta del valor en segundos del tiempo del cursor más a la derecha con el que se encuentre más a la izquierda, obteniéndose el correspondiente período, que una vez generado y bajo la conformidad del usuario se podrá accionar el, entonces, habilitado accionamiento “aceptar”, guardándose ambos

parámetros correspondientes a esta experimentación. Por el contrario, si decidiera prescindir de los datos generados simplemente tendrá que accionar el botón "cancelar", que no guardará lo calculado en esta tarea.

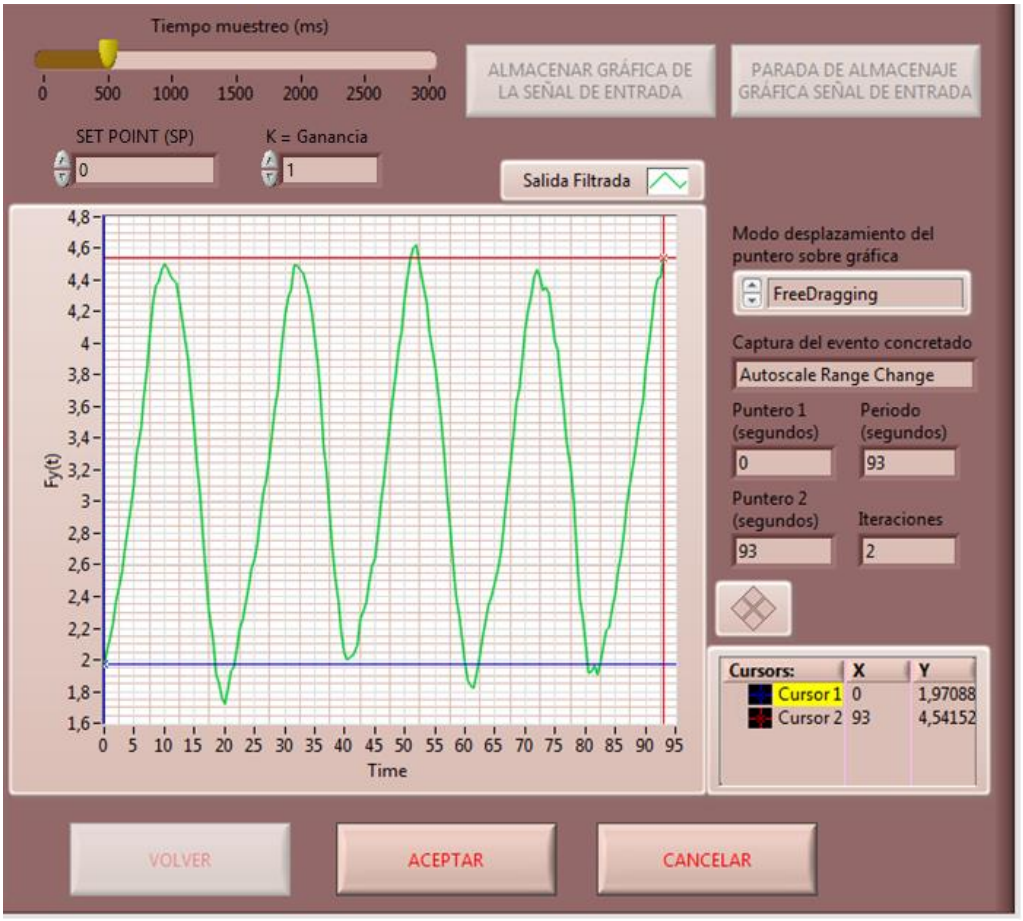


Figura 4.13. Interfaz usuario la obtención numérica de la ganancia crítica y de la obtención gráfica del período crítico.

4.1.1.6. REGULACIÓN Y SINTONÍA DEL PID:

En lo que respecta a esta última tarea, el usuario podrá diseñar un controlador PID automático y el modo en que podrá desarrollar el sistema de control será el siguiente (Véase figura 4.14.):

- **Menú de tipo de control:** En el desplegable superior izquierdo, el usuario podrá elegir qué tipo de experimentación emplear para el diseño del PID, pero téngase en cuenta que la posibilidad de elegir no se encontrará disponible si alguna de las experimentaciones no se realizó con éxito. Ante esta situación quedará fijado aquella experimentación que si que se realizó no pudiendo desplegar el menú de opciones referente a estas dos experimentaciones.

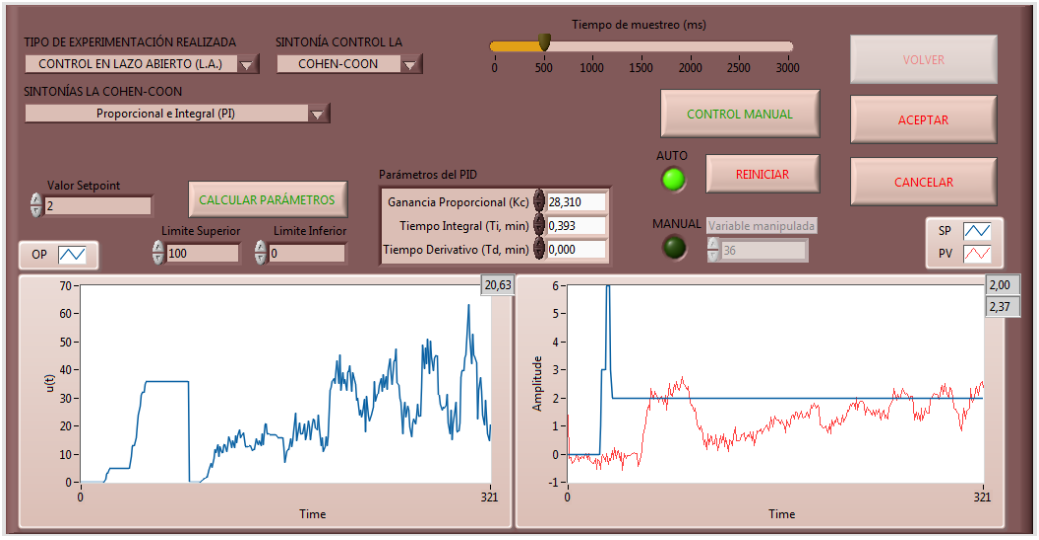


Figura 4.14. Interfaz usuario del diseño del controlador PID.

- **Menú del tipo de sintonía:** El desplegable que se encuentra justo debajo del desplegable anterior permitirá al usuario conocer qué tipo de controlador podrá diseñar en base a los parámetros de la experimentación que si habían estado disponibles. Así para los parámetros en lazo abierto se tendrán las siguientes sintonizaciones:

REGIÓN DE SINTONÍA DEL CONTROLADOR DE LAZO ABIERTO POR EL MÉTODO DE ZIEGLER-NICHOLS.

Tipo	Ganancia K_p	Tiempo integral	Tiempo derivativo
P	$\tau / (K\, d)$		
PI	$0.9\tau /(K\, d)$	3.33 d	
PID serie	$1.2\tau /(K\, d)$	2 d	0.5 d

Figura 4.15. Tabla de sintonía del controlador de lazo abierto por el método de Ziegler-Nichols.

REGIÓN DE SINTONÍA DEL CONTROLADOR DE LAZO ABIERTO POR EL MÉTODO DE ROVIRA.

PI paralelo

Criterio	Proporcional	Integral	Derivativo
MIAE	a=0.758 b=-0.861	a=-0.323 b=1.020	
MITAE	a=0.586 b=-0.916	a=-0.165 b=1.030	

PID Paralelo

MIAE	a=1.086 b=-0.869	a=-0.130 b=0.740	a=0.348 b=0.914
MITAE	a=0.965 b=-0.855	a=-0.147 b=0.796	a=0.308 b=0.929

$$K_p K = a\left(\frac{d}{\tau}\right)^b$$

$$\frac{\tau}{T_i} = a\left(\frac{d}{\tau}\right) + b$$

$$\frac{T_d}{\tau} = a\left(\frac{d}{\tau}\right)^b$$

Figura 4.16. Tabla de sintonía del controlador de lazo abierto por el método de Rovira.

REGIÓN DE SINTONÍA DEL CONTROLADOR DE LAZO ABIERTO POR EL MÉTODO DE COHEN-COON.

Tipo de Controlador	Ganancia K_c	Tiempo Integral T_i	Tiempo Derivativo T_c
P	$\frac{\tau}{Kd}\left(1+\frac{d}{3\tau}\right)$		
PI	$\frac{\tau}{Kd}\left(0.9+\frac{d}{12\tau}\right)$	$d\frac{30+3d/\tau}{9+20d/\tau}$	
PID	$\frac{\tau}{Kd}\left(1.333+\frac{d}{4\tau}\right)$	$d\frac{32+6d/\tau}{13+8d/\tau}$	$d\frac{4}{11+2d/\tau}$

Figura 4.17. Tabla de sintonía del controlador de lazo abierto por el método de Cohen-Coon.

REGIÓN DE SINTONÍA DEL CONTROLADOR DE LAZO ABIERTO POR EL MÉTODO DE LOPEZ Y OTROS.

Reguladores PI paralelo

Criterio	Proporcional	Integral	Derivativo
MIAE	a=0.984 b=-0.986	a=0.608 b=-0.707	
MISE	a=1.305 b=-0.959	a=0.492 b=-0.739	
MITAE	a=0.859 b=-0.977	a=0.674 b=-0.68	

$$K_pK = a\left(\frac{d}{\tau}\right)^b$$

$$\frac{\tau}{T_i} = a\left(\frac{d}{\tau}\right)^b$$

$$\frac{T_d}{\tau} = a\left(\frac{d}{\tau}\right)^b$$

Reguladores PID paralelo

Criterio	Proporcional	Integral	Derivativo
MIAE	a=1.435 b=-0.921	a=0.878 b=-0.749	a=0.482 b=1.137
MISE	a=1.495 b=-0.945	a=1.101 b=-0.771	a=0.560 b=1.006
MITAE	a=1.357 b=-0.947	a=0.842 b=-0.738	a=0.381 b=0.995

$$K_pK = a\left(\frac{d}{\tau}\right)^b$$

$$\frac{\tau}{T_i} = a\left(\frac{d}{\tau}\right)^b$$

$$\frac{T_d}{\tau} = a\left(\frac{d}{\tau}\right)^b$$

Figura 4.18. Tabla de sintonía del controlador de lazo abierto por el método de López y Otros.

Como se ha podido ver, se han expuestos todas las sintonías en lazo abierto que se programó para que el usuario pudiera elegir qué tipo de controlador, en estas circunstancias, diseñaría. En el apartado del diagrama del programador se mencionará cómo irán establecidas las sintonías sin argumentarse como serán cada una de ellas con sumo detalle, dado que tiene mayor prioridad la técnica empleada para poder

estructurar tanto los desplegables de los diversos menús, hasta la coordinación de los diferentes accionamientos para posibilitar la decisión de una entre varias sintonías disponibles diferenciando entre parámetros correspondientes al lazo abierto o cerrado, existiendo además, otra condición que se tendrá que incluir, basada en los modos manual o automático disponibles que podrán establecerse. En el caso de la sintonía en lazo cerrado, únicamente se tendrá el correspondiente al de Ziegler Nichols (Véase figura 4.19.):

REGIÓN DE SINTONÍA DEL CONTROLADOR DE LAZO CERRADO POR EL MÉTODO DE ZIEGLER-NICHOLS.

Tipo	Ganancia K_p	Tiempo integral	Tiempo derivativo
P	$0.5\ K_c$		
PI	$0.45\ K_c$	T/1.2	
PID paralelo	$0.75\ K_c$	T/1.6	T/10
PID serie	$0.6\ K_c$	T/2	T/8

Figura 4.19. Tabla de sintonía del controlador de lazo cerrado por el método de Ziegler-Nichols.

- **Mostrador de parámetros calculados del PID:**

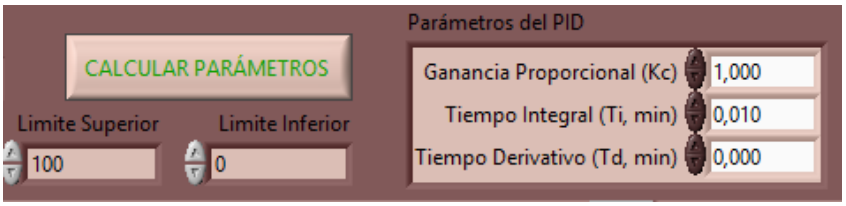


Figura 4.20. Interfaz usuario mostrador de parámetros calculados y delimitación de los valores en el controlador PID.

En la imagen anterior, el usuario podrá ver los parámetros del PID calculados según qué sintonía haya elegido para el diseño del mismo. Podrá, además, modificar los mismos si procediera con éxito una mejora en los datos proporcionados. Además de ello, el usuario podrá establecer qué límites de actuación tendrá el controlador que pretende diseñar a lo largo de esta tarea.

- **Activación del modo manual o automático:**

Como se comentó anteriormente, se podrá elegir entre el modo manual o el automático, siendo en el último de ellos la que aplica las diferentes sintonías disponibles para esta tarea del diseño del controlador PID. En el caso de que el usuario elija el modo manual, podrá interactuar con la variable manipulada directamente para cumplir con la exigencia de la referencia que se pretende seguir.

Una vez hecho esto el usuario podrá optar por cambiar al modo automático para que la planta siga la referencia sin que el usuario tenga que actuar sobre ella.

En el caso de que el efecto de la sintonía del controlador PID resulte ineficiente por causas diversas, se podrá reiniciar accionando el botón que lleva su causa provocando que el controlador comience desde un valor inicial nulo a funcionar para llevar al sistema a la referencia que el usuario estableció previamente.

Figura 4.21. Interfaz usuario selector de modo manual o automático con habilitación de la variable manipulada en modo manual.

Si bien es cierto que para poder ver el efecto eficiente del controlador PID, se necesita alguna gráfica, a continuación, se muestran dos de ellas (Véase figura 4.22.). En la de la derecha se verá la salida mostrada por el sistema con el controlador PID incluido en el funcionamiento y las respectivas entradas salto implicadas para contemplar el automatizado de la planta de procesos. La gráfica de la izquierda muestra la señal que genera el sistema como consecuencia de intentar seguir a la referencia que se impuso, intentando lograr con ello que el error estacionario sea prácticamente despreciable. Como se puede ver en la ilustración adjunta, ambas gráficas se colocaron una al lado de otra para poder facilitar al usuario la comprobación visual de los datos a medida que se ejecuta la tarea en cuestión.

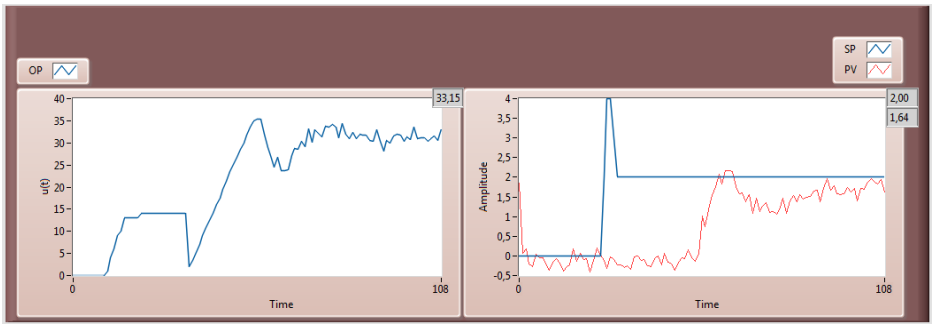


Figura 4.22. Interfaz usuario gráficas de la respuesta generada por el controlador, entradas saltos añadidos y señal OP generada.

Como en las anteriores tareas, el usuario dispondrá de un juego de accionamientos basados en "volver" si no se quisiera continuar con esta tarea sin haber hecho ningún cálculo previo, "cancelar" si habiendo calculado todo lo referente a esta tarea se presenta la suficiente disconformidad como para no guardar ningún dato obtenido. Si el usuario decidiera guardar los datos, únicamente debería accionar el botón de "aceptar" y se abandonaría la interfaz del diseño del controlador PID, regresando al menú principal ya sea nuevo estudio o antiguo estudio.

4.1.2. PROGRAMA DE EJECUCIÓN ANTIGUO ESTUDIO:

En esta zona del programa lo que se pretende es poder continuar con un trabajo que no había sido finalizado en su momento por las razones que fueran. El usuario podrá reanudar el estudio antiguo desde el punto donde lo dejó sin acabar e inclusive rehacer alguna tarea cuyos resultados no hayan sido satisfactorios. Téngase en cuenta que todas las tareas que el usuario encontrará en esta sección serán idénticas a las que se encontró en el menú del nuevo estudio excepto aquella en la que deberá introducir el fichero XML generado en una ocasión anterior. Dicha tarea se expondrá a continuación con todos los detalles para que el usuario pueda comprender el funcionamiento programado que se diseñó con detenimiento.

Para ello deberá existir un fichero XML en donde se hayan guardado los avances del diseño de un controlador PID de la índole que sea para ser ejecutado en este software.

4.1.2.1. INTRODUCCIÓN DEL FICHERO GUARDADO:

Aunque en lo que respecta a esta tarea, se mostrarán diferentes capturas de pantalla de la interfaz de la subrutina propuesta para el usuario, dichas imágenes se encuentran juntas en la ejecución del programa. Así en la primera de ellas el usuario deberá accionar el botón que se le proporciona, con el icono de una carpeta amarilla (Véase figura 4.23.) generando seguidamente una ventana con los posibles directorios que ofrece el explorador de Windows. En esa situación el usuario deberá elegir la carpeta en donde se encuentre el fichero XML y seleccionarlo para poder cargar los datos que contiene.

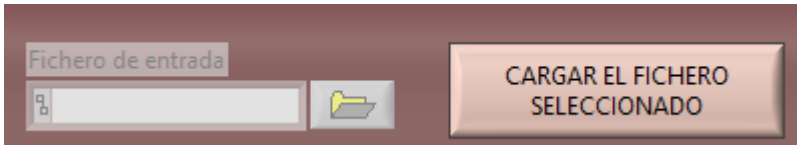


Figura 4.23. Interfaz usuario introducir fichero XML para leer y cargar los datos guardados de su interior.

Así pues, una vez cargados, habiendo accionado el botón que se le otorga justo a un lado, el usuario podrá ver el nombre del fichero registrado, el directorio a que pertenece y tras leído su contenido, recibir una sugerencia de comenzar en la tarea posterior a la última realizada (Véase figura 4.24.)

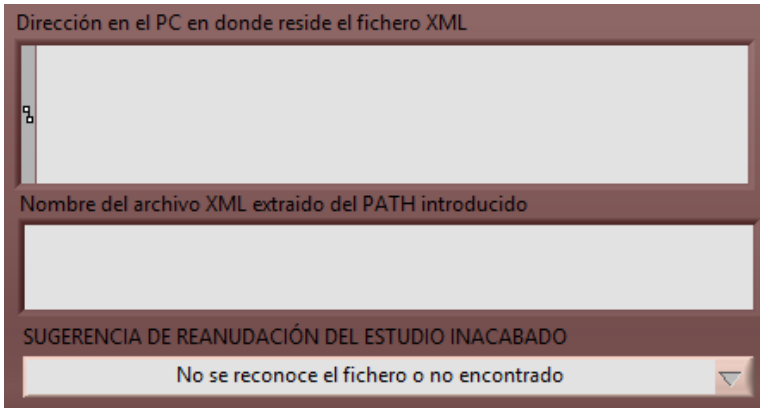


Figura 4.24. Interfaz usuario introducir fichero XML para cargar sus datos.

Al lado de estas ventanas indicadoras de la imagen anterior aparece una tabla con los datos cargados del fichero introducido anteriormente, mostrando aquellos aspectos que el usuario podrá considerar para futuras decisiones en lo que respecta a las tareas de calibración, experimentación y diseño del controlador PID. De manera que, tras haber cargado los datos del fichero, el

usuario podrá reanudar su estudio accionando el botón de "aceptar". En el caso de que el fichero muestre datos erróneos o simplemente tengan una estructura desordenada con un contenido que no obedece a la forma en que se guardaron previamente, el usuario podrá accionar el botón de "fichero no encontrado" como parte de la consideración asignada a fallo en la lectura del documento en cuestión. Todo ello se puede observar en la siguiente imagen (Véase figura 4.25.):

Figura 4.25. Interfaz usuario tabla de datos cargados del fichero XML introducido y accionamientos para decidir el destino de los mismos.

4.1.3. ENVIAR POR CORREO O GUARDAR EN INTERNET:

En lo que respecta a esta última tarea, el usuario podrá elegir de entre todas las páginas web que se proponen aquella que mejor le convenga. El funcionamiento que se propone en esta tarea será explicado, en posteriores puntos, acerca de la programación que se llevó a cabo en su diseño. Aquí se argumentará el modo en que el usuario podrá ejecutar la tarea otorgándosele una pantalla guía que le sugerirá seleccionar uno de los botones que se ofrecen para almacenar el fichero XML o enviarlo a través del medio elegido. La pequeña pantalla que le dará idea al usuario de cómo dar uso a esta tarea se muestra a continuación (Véase figura 4.26.):

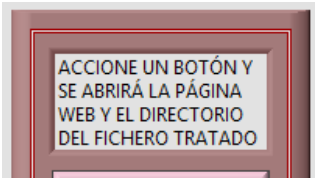


Figura 4.26. Interfaz usuario pantalla de información sobre el funcionamiento de la tarea de compartición y almacenaje en internet.

Seguidamente se mostrará la serie de accionamientos que al ser activados se mostrará la página web correspondiente al nombre del botón en cuestión, abriéndose en paralelo el directorio en donde se encuentra alojado el fichero XML, facilitando así al usuario dicha búsqueda para poder ejecutar el envío o almacenaje en la zona seleccionada de internet. La serie de accionamientos se muestran en la siguiente imagen (figura 4.27.):

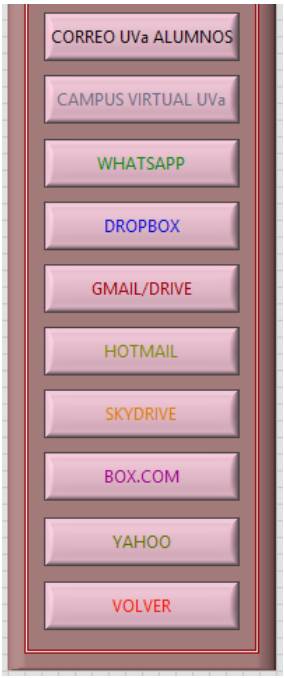


Figura 4.27. Interfaz usuario panel de páginas web de internet en donde el usuario podrá guardar o enviar el fichero XML.

Una vez enviado el fichero XML la tarea habrá quedado finalizada y se volverá al menú principal sin necesidad de accionar ningún botón de “aceptar” o “cancelar” debido a la corta duración que posee esta subrutina, cuyo motivo es el acceso a internet desde una carpeta del explorador de Windows para poder tratar el fichero en cuestión, en alguna de las páginas que eligió sea para almacenarlo en un espacio privado o enviarlo por correo electrónico según cual haya sido la plataforma social escogida.

4.2. MANUAL DEL PROGRAMADOR

En esta sección, se procederá a exponer las especificaciones del software diseñado en este proyecto. Se comentará la arquitectura que se ha diseñado en detalle para poder exponer el conjunto de ideas que han podido construir cada parte del programa en cuestión. Posteriormente se argumentarán los diagramas en el manual del programador para facilitar la comprensión de la estructura empleada en el código que rige su funcionamiento. Por último, al haberse añadido el manual del usuario previamente a este apartado, ello permitirá al lector la sencilla comprensión de las diversas pantallas desarrolladas y su función, que es un compendio de instrucciones para que el programa pueda ser utilizado de cara a las experimentaciones y diseño de controladores PID.

4.2.1. PLANTEAMIENTO, ARQUITECTURA Y DESARROLLO:

Seguidamente a lo expuesto anteriormente se expondrán el planteamiento, arquitectura y desarrollo del software implicado en este proyecto:

4.2.1.1. PLANTEAMIENTO:

El cometido del diseño de este software con base en el lenguaje de programación visual de LabView, reside en la búsqueda de conocer experimentalmente la teoría de controladores automáticos y la aplicación de éstos en plantas de proceso previamente analizados para su posterior automatización.

Ante la posibilidad de poder automatizar un sistema, primeramente, se destinó un conjunto de subrutinas para:

- Poder establecer la comunicación a través de un servidor OPC.
- Poder calibrar los datos de entrada de la planta en cuestión, los cuales son recibidos por parte de una tarjeta de adquisición de datos (DAQ)
- Creación, edición y carga de datos de un fichero de tipo XML que el usuario únicamente indicará el nombre y el lugar en el ordenador donde se destinará el correspondiente guardado. El programador podrá contemplar la idea acerca de que el usuario no será consciente del guardado automático que el software realiza en el fichero XML para facilitarle dicha tarea de actualización de datos creados durante la ejecución del programa.

El resto de los subprogramas se destinarán exclusivamente al diseño del controlador automático e implantación en la planta de estudio que se eligió inicialmente. El planteamiento es consecutivo, esto es, el usuario podrá seleccionar entre experimentación en lazo abierto (L.A.) o lazo cerrado (L.C.) para poder concluir con el diseño del controlador de tipo PID con sus correspondientes configuraciones, quedando toda aquella información resultante en el fichero XML.

Cabe comentar que el software tiene doble función para realizar las tareas que en su interior se sugieren. La primera de ellas se basa en un nuevo estudio, referente al inicio de la ejecución del programa sin interrupciones en la obtención de los datos necesarios para el uso a posteriori, no obstante, supóngase que por múltiples razones el usuario no pudiera completar el desarrollo del diseño del controlador PID, ante esto la segunda funcionalidad que se pretende argumentar se basa en ello, en el empleo de la información que se guardó en el fichero XML para poder reanudar la serie de tareas pendientes para finalizar el diseño de controlador PID. Si se ejecutara esta segunda funcionalidad, se estaría hablando de un estudio antiguo, que es como se denomina en el software programado.

A parte de todo lo comentado anteriormente se ha añadido una subrutina adicional que permite al usuario abrir en internet, según una previa selección, un portal web donde guardar el fichero XML generado además de abrirse una ventana de Windows en donde se encuentra el fichero actualizado en cuestión.

4.2.1.2. ARQUITECTURA:

En lo que se refiere arquitectura del programa se comentarán los puntos más esenciales en la construcción del software, aquellos que en el siguiente apartado se desarrollarán en mayor medida. Como ya se comentó en el segundo capítulo, en el diseño del programa en cuestión se ha empleado el uso de SubVIs, de los cuales los que han sido de vital importancia para poder exponer el funcionamiento del mismo son los que se comentan a continuación:

- **VI inicio:** Menú principal.
- **SubVIs principales:** Programa principal y Programa secundario.
- **SubVIs secundarios:** Crear fichero XML, establecer comunicación OPC, calibración, experimentación en lazo abierto, experimentación en lazo cerrado, sintonía PID y lectura del fichero XML.

Todos ellos se interrelacionan del siguiente modo que se muestra a continuación:

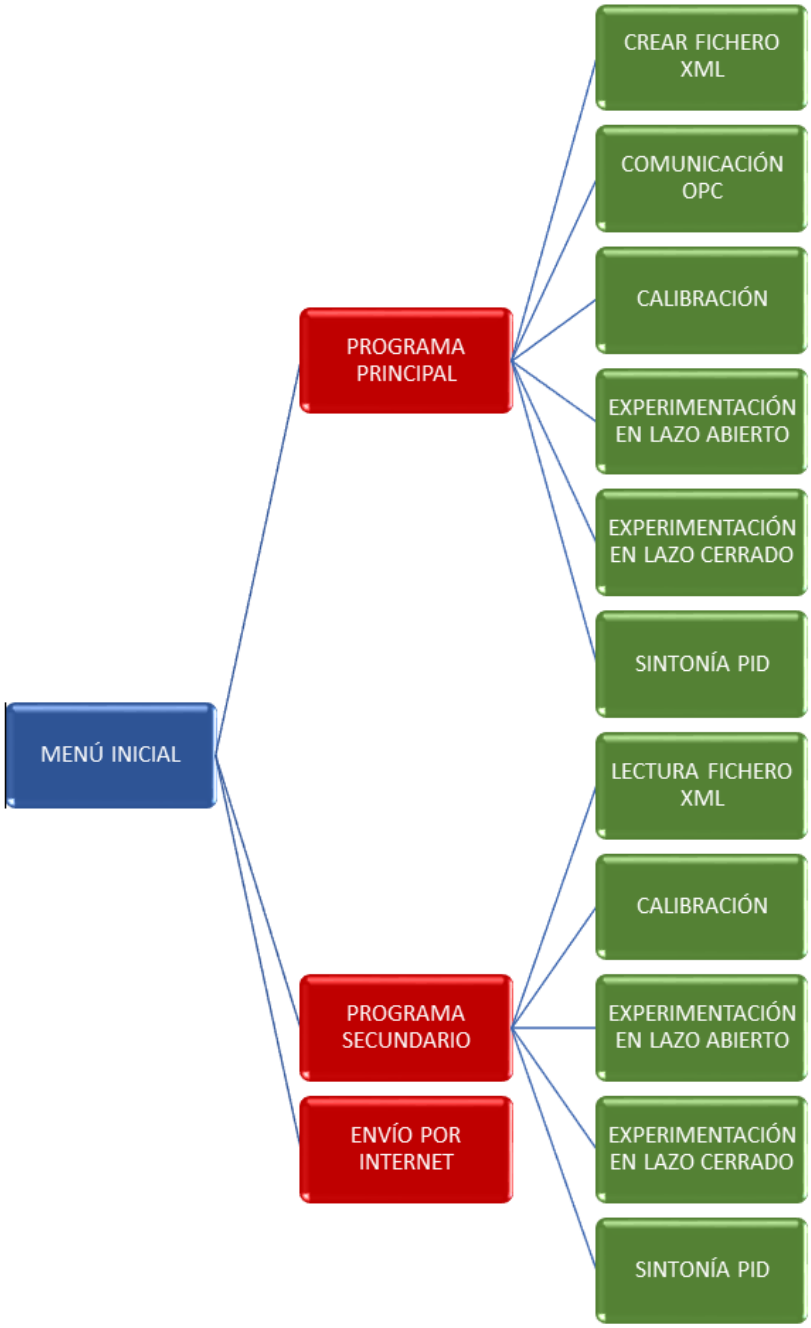


Figura 4.28. Arquitectura del software del programa

Como puede contemplarse en el diagrama, los bloques de color rojo son los que portan con los bloques verdes que contienen a su vez las tareas que son de objeto en este proyecto. Los tres bloques rojos se encargan de la estructura principal del programa, poseyendo cada uno de ellos una función que se expone a continuación:

- Programa principal:

  Es el encargado de generar un nuevo estudio con un nuevo fichero XML en donde se guardarán todos los datos generados durante la ejecución del programa.

- Programa secundario:

  Es el encargado de reanudar un estudio antiguo que ha quedado sin completar, cargando los datos del fichero XML que se almacenaron previamente permitiendo al usuario continuar la ejecución del programa desde la tarea donde quedó interrumpido su estudio. Una vez cargada la información que empleará posteriormente el usuario podrá volver a ejecutar tareas que ya había hecho con el fin de corregir o mejorar las tareas realizadas.

- Envío por Internet:

  Como ya se comentó anteriormente, este apartado únicamente permite al usuario abrir la ventana de Windows en donde se aloja el fichero XML además de la página web, según qué decisión haya establecido el usuario, en donde surgirá un portal privado de almacenamiento en internet para guardar o enviar el documento.

A continuación, se expondrán las características de las subrutinas que existen en este programa y el modo en que se sincronizan entre sí para la transferencia de datos o establecer una consecución de ejecución.

4.2.1.2.1. SUBRUTINAS COMPARTIDAS:

El programa principal y el secundario comparten subrutinas (bloques verdes, Figura 5.1.) de entre los cuales son los que se exponen a continuación:

- **Calibración:** En ella se realiza la calibración de los datos recibidos a través del servidor OPC, para convertirlos a datos con la unidad del sistema internacional que corresponde con la medida que se realizó.

- **Experimentación en Lazo Abierto (L.A.):** en esta zona, como se argumentó en el capítulo 3 sobre “fundamentos teóricos”, en el lazo abierto se hallarán los respectivos parámetros que se extraerán de la curva de reacción después de haber introducido una entrada escalón al sistema.

- **Experimentación en Lazo Cerrado (L.C.):** de la misma manera, acudiendo al capítulo 3, se puede comentar que en esta experimentación se podrán obtener los correspondientes parámetros después de buscar la conveniente ganancia crítica.

- **Sintonía de un controlador PID:** en la parte que corresponde esta tarea, también puede mencionarse que sus fundamentos teóricos se hallan en el capítulo que anteriormente se ha indicado, no obstante, en este capítulo se desarrollará el compendio de elementos que se juntaron para posibilitar el diseño de un controlador PID en función de qué experimentación se hizo previamente (Lazo Abierto o Lazo Cerrado o ambas). En este caso dicha selección según qué información quede registrada por parte del usuario se argumentará en el siguiente apartado, lo tendrá en cuenta el software diseñado de manera automática.

Sin embargo, existen, como se argumentó al principio de este capítulo, otras subrutinas que no se comparten entre SubVIs. Éstas se analizarán a continuación.

4.2.1.2.2. SUBRUTINAS PROPIAS DE UN SUBVI CONCRETO:

En este apartado se comentará todo acerca de aquellos SubVIs que no comparten nada acerca de la información que se trabaja en su ejecución. Con ello se hará referencia a tales subrutinas como: “programa principal” y “programa secundario”, existiendo en el primero de los cuales los programas siguientes:

- **Creación de fichero XML:** En donde se generará un nuevo documento de esta índole capacitado para almacenar los datos que se configuren en cada una de las tareas que se realizarán.

- **Comunicación OPC:** En esta subrutina lo que se pretenderá será el establecimiento de la comunicación OPC para poder recibir la información que se adquirirá de la fuente OPC (en nuestro ejemplo a través de la tarjeta de adquisición de datos DAQ)

Mientras que en el segundo de ellos la única subrutina que se encuentra diseñada es la relacionada con el fichero XML creado:

- **Lectura del fichero XML:** En este caso, el programa lo que hará es leer un fichero XML cuya URL en Windows haya introducido el

usuario previamente y permitirle reanudar la ejecución del programa desde donde dejó pendiente la obtención de datos que posteriormente se almacenarán, pudiendo actualizar aquellos datos de las tareas que haya hecho anteriormente.

4.2.1.2.3. SUBRUTINAS DE ALMACENAMIENTO DE DATOS:

Existen otro tipo de subrutinas que tratan el fichero XML, con el cual trabaja el programa del proyecto. Son aquellas que apenas son notorias para los usuarios debido a que carecen de panel interactivo, pero sí para el programador que visualizará el diagrama en LabView. Se trata de aquellos SubVIs que cargan, guardan y actualizan la información que se mueve por todas las tareas del software diseñado. Ello es debido a tres razones de peso que se explican a continuación:

- La primera de ellas está relacionada directamente con el fichero XML debido a que estas subrutinas guardan la información tratada en dicho fichero de manera automática sin consultar al usuario, proporcionándole a éste toda esa información actualizada en cada instante que lo necesite.

- La segunda de ellas se basa en la actualización de los datos mostrados en cada uno de los dos paneles de registro que se le otorga al usuario durante la ejecución de las sucesivas tareas a lo largo de un nuevo o antiguo estudio. Dichos paneles de información podrían considerarse como si fuera el propio fichero XML mostrando el contenido que posee en un concreto instante habiendo o no cambiado algún dato en alguna tarea.

- La última de ellas se trataría de la subrutina correspondiente al envío o almacenamiento del fichero XML en alguna página en concreta de internet tal y como se comentó líneas más arriba.

4.2.1.3. DESARROLLO:

Llegados a este punto, lo que se comentará en el desarrollo del programa son todos los detalles esenciales que posibilitaron el diseño de software con una sólida arquitectura, comentada anteriormente. Así pues, para facilitar el seguimiento de las rutinas detalladas se comenzará describiendo desde el programa inicial que contiene a todos los SubVIs, finalizando en los subprogramas que no contengan anidaciones en el interior de su

configuración esquemático de código, como si de un árbol jerárquico se tratara la exposición del código completo.

4.2.1.3.1. PROGRAMA INICIO:

En el programa inicio lo que se plantea es la colocación de los primeros botones que accionarán la selección que se pretenda hacer en función de si el usuario pretende realizar un nuevo estudio, un antiguo estudio o envío por internet (si es la primera ejecución de programa, este accionamiento quedará inhabilitado para el usuario, debido a que no existe fichero alguno con el que se haya trabajado previamente) Se programó además un botón adicional de salida para concluir con la ejecución del software. La representación de este panel de botones se muestra en la siguiente ilustración (figura 4.29.):

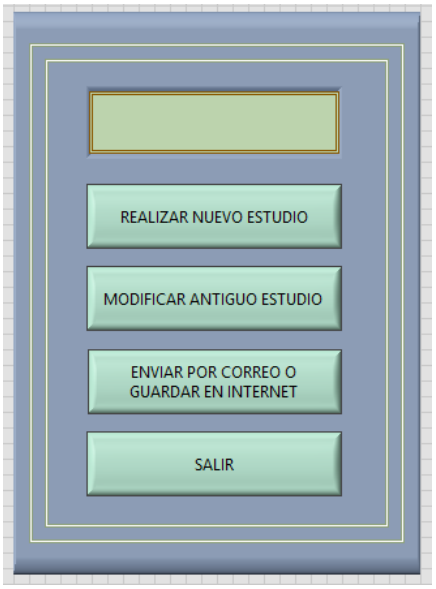


Figura 4.29. Panel usuario LabView del primer menú

Al abrir el interior de este panel interactivo, se puede contemplar el siguiente código programado (figura 4.30.):

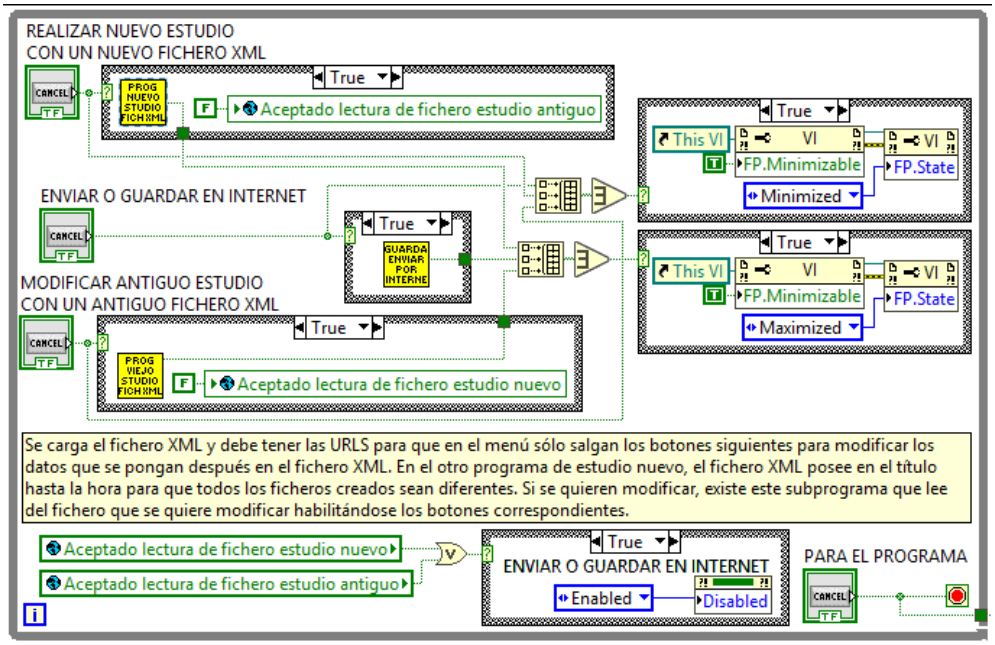


Figura 4.30. Código LabView del primer menú del Software diseñado.

En el cual se pueden apreciar las variables booleanas que se han utilizado para poder establecer qué accionamiento permite al usuario decidir entre varias opciones de inicio de un nuevo estudio o reanudación de uno anterior. Cabe mencionar que el código de los case que extraen los valores booleanos simplemente permiten al usuario centrarse en una única ventana emergente que será la parte del programa que este ejecutando actualmente quedando todas las demás minimizadas por eficacia visual.

Obsérvese que se han agregado una serie de comentarios relacionados con la explicación del motivo por el cual se decidió elegir esa óptima configuración de cara al método de programación. El cometido de dichos comentarios es facilitar al programador las razones de proceder de aquel modo en el instante de programar la tarea que en ese punto se sugiere. Así a lo largo de las posteriores explicaciones con las ilustraciones adjuntas, se contemplará lo argumentado en este párrafo.

Juntamente con la parte programada para el menú que antes se había mostrado se añadió en la misma estructura de la secuencia el reinicio de todas las variables globales que se han empleado a lo largo del completo software del proyecto. Todas ellas comparten datos de subrutinas entre sí y con ficheros XML por ello cada vez que se vuelve a ejecutar el programa desde el principio se debe reiniciar estas variables para evitar que exista información residual.

En la siguiente imagen (figura 4.31.) se puede contemplar las diferentes variables globales que se han empleado en el software:

- Variable global de tipo String: Dedicada al tratamiento de texto.
- Variable global de tipo Booleana: Destinada al valor único 0 o 1.
- Variable global de tipo DBL: Destinada a valores numéricos.

Cabe comentar que ya se había hablado de estas variables en el capítulo 2 de manera que para mayor información a lo largo de las posteriores explicaciones acerca de éstas consideraciones, consúltese dicho capítulo.

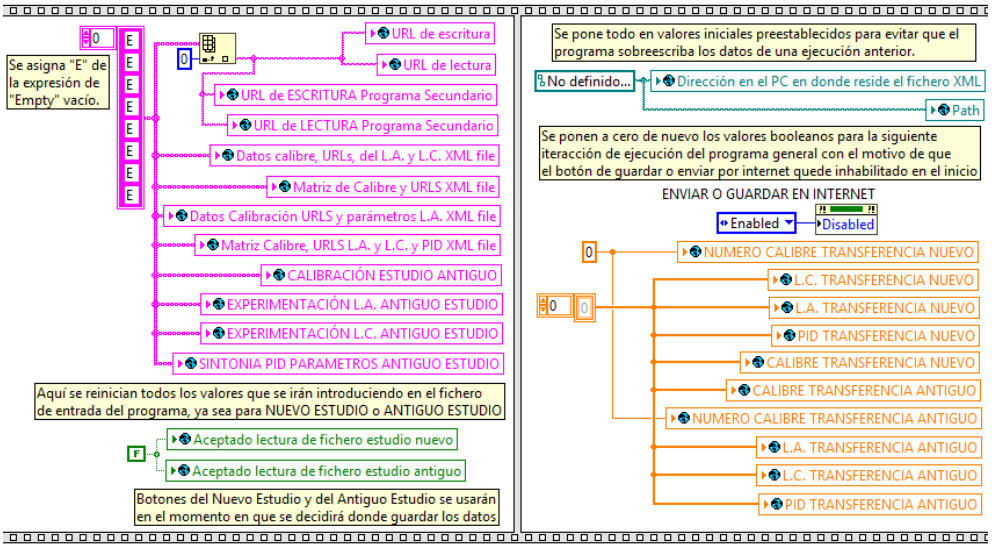


Figura 4.31. Código LabView del reinicio de todas variables globales.

En esta parte del programa se diseñó también una parte peculiar acerca de interactuar con el usuario con un simple saludo en función de la hora a la que ejecute el programa, ya sea por la mañana, por la tarde, por la noche o incluso de madrugada.

4.2.1.3.2. PROGRAMA PRINCIPAL NUEVO ESTUDIO:

Se programó la subrutina del nuevo estudio para que el usuario pudiera comenzar desde el punto que se argumentó en los apartados anteriores, es decir, desde la creación del fichero XML hasta el diseño del controlador PID. Para acceder al nuevo estudio el programador ha de tener en cuenta la siguiente ilustración (figura 4.32.):

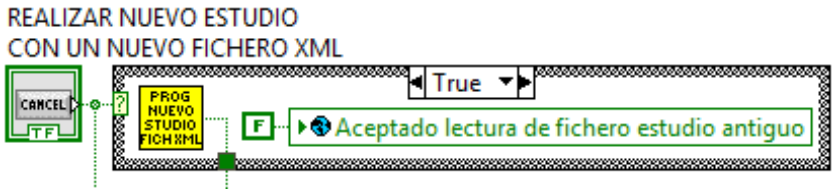


Figura 4.32. Bloque de acceso al nuevo estudio.

En ella se puede ver como el bloque amarillo ejecuta la entrada a la realización del nuevo estudio junto con una variable booleana con un estado de "falso". El hecho de que la variable booleana sea falsa evita que los SubVIs que se comparten en la ejecución; calibración, ambas experimentaciones y el diseño del controlador PID, envíen información de una variable global que no corresponda con las variables globales que estén directamente relacionados con el fichero XML ya sea nuevo o antiguo, en referente a guardar los datos que porten en su interior.

Ante esto cabe comentar una aclaración en referente a los ficheros XML y las variables globales. El programa está diseñado para que en función del tipo de fichero XML que se trate, antiguo o nuevo, poseerá unas exclusivas variables globales. Debido a esto, se tendrá que asignar el instante en que unas variables globales entren en juego y otras queden inoperativas, de ahí que se haya agregado una variable de tipo booleana que pueda representar un equivalente semáforo para organizar intervención de cada variable en un momento y tarea determinada.

Una última de las cuestiones que se tienen que tener en cuenta es que en la imagen anterior se ve como el bloque amarillo genera una única salida de tipo booleano que se emplea para poder minimizar el menú que este bloque dejó maximizado pudiendo crear confusión de ventanas abiertas al usuario que esté utilizando el presente software. Con esa variable booleana, cuyo mecanismo se empleará en todas las tareas de este programa, se podrá evitar que hayan más de una ventana abierta que tenga relación con la ejecución del software, existiendo maximizada aquella en la que el usuario este interactuando, permaneciendo las demás minimizadas hasta nueva acción que releve esta cualidad a otra subrutina.

Al accionar el acceso a un nuevo estudio, este SubVI posee un panel de datos actualizados adjunto a la serie de accionamientos que ejecutan las tareas de experimentación y configuración de los parámetros del PID. La ilustración siguiente muestra lo que el usuario vería al adentrarse en la realización de un nuevo estudio (Figura 4.33.):

Figura 4.33. Panel usuario LabView del menú estudio nuevo

Todos los accionamientos que se contemplan en la imagen anterior (figura 4.33.) poseen tres botones básicos para que el usuario pueda navegar por todas las zonas del software; aceptar, cancelar y volver. Estos botones se comentarán en mayor medida a posteriori para guiar al usuario en todos los pasos que debiera dar para utilizar este software con la mayor rectitud posible.

En la siguiente imagen (figura 4.34.) se vuelven a introducir las variables globales reiniciadas. El motivo de mostrar dicha ilustración se debe al modo en que se respondería a la posibilidad de que un usuario saliera del bloque “nuevo estudio” y decidiera seguidamente adentrarse en el bloque contiguo denominado “antiguo estudio” sin necesidad de haber apagado el ordenador o cerrado la aplicación de este software o viceversa.

El problema reside en que si no se incorporara lo que se programó en la ilustración de la figura 4.34. del reinicio de todas las variables globales, las mismas poseerían información que no formaría parte del estudio actual sea nuevo o antiguo que se pretende iniciar o reanudar respectivamente, existiendo confusión de datos que habrían guardadas en algunas variables globales que no deberían tener contenido alguno, del tipo que sea al no haberse ejecutado la correspondiente tarea.

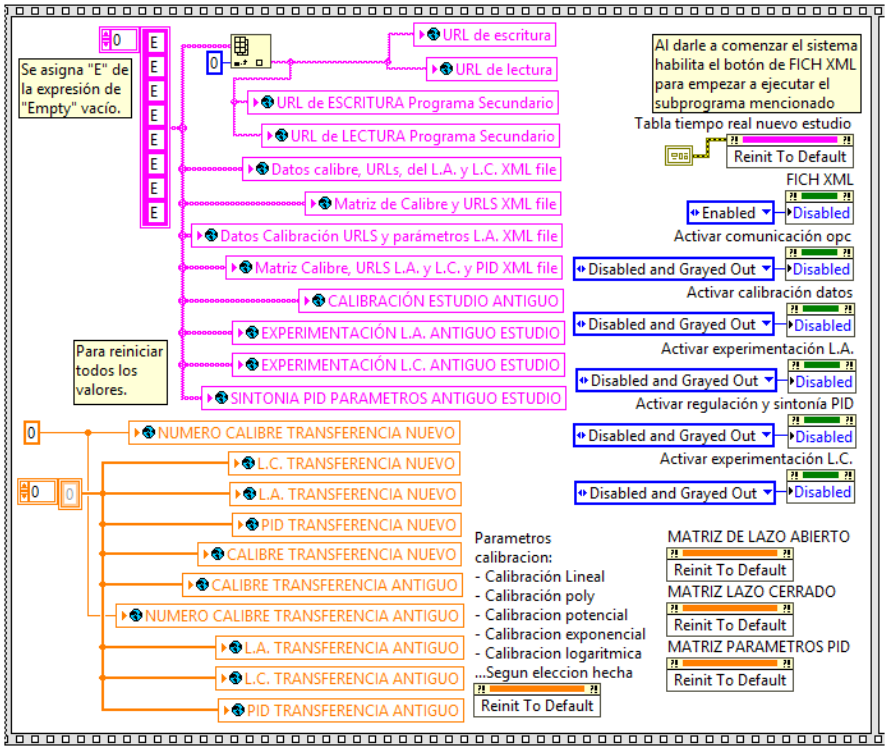


Figura 4.34. Código LabView del reinicio de variables estudio nuevo.

En la imagen también se puede visualizar las constantes de habilitación de los accionamientos booleanos en azul referente a cada tarea que se puede ejecutar. Aparecen cerca del margen derecho de la misma junto con las variables locales en la zona inferior, referente a matrices reiniciadas a un valor nulo por defecto. Téngase en cuenta que en cada SubVI se reiniciarán todas las variables para evitar confusión en el guardado de los datos en el fichero XML proveniente de las variables globales implicadas en cada tarea propuesta del software diseñado. A continuación, se expondrán los apartados que se tienen programados para cada una de las tareas que se han diseñado en este proyecto:

4.2.1.3.2.1. ASIGNAR ZONA DE GUARDADO:

En este módulo se programará la creación del fichero XML en donde se escribirán los avances realizados de todas las tareas que se completen. Obsérvese en la ilustración (Figura 4.35.) que el bloque naranja es el que permite el acceso al usuario a la creación del mencionado documento. El bloque se acciona exclusivamente si se acciona el botón booleano denominado "FICH XML" proporcionando en su salida la dirección en el explorador de Windows en dónde residirá el fichero XML. Esta información se guarda en una variable global que se compartirá posteriormente para poder actualizar los

datos que se vayan almacenando de las tareas que se vayan ejecutando. En la ilustración puede verse además que mientras no se accione el botón de aceptación de la creación y localización del fichero XML, el usuario no podrá acceder a la siguiente tarea del programa principal, quedando el accionamiento de las posteriores tareas inhabilitados para su acceso.

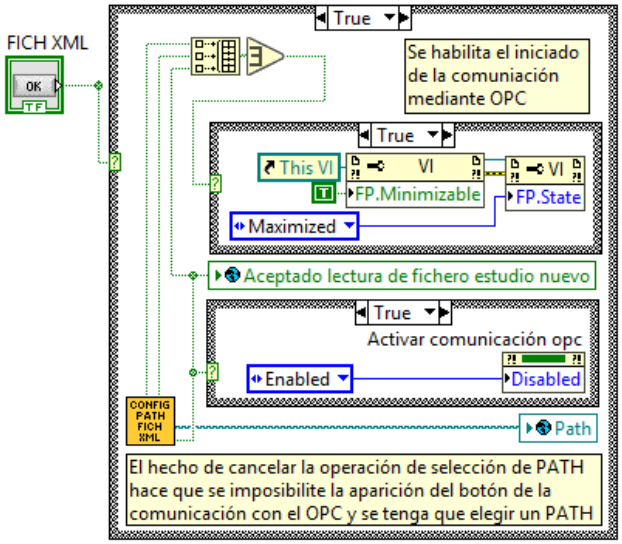


Figura 4.35. Código LabView asignación zona de guardado.

Como se puede observar aquí, además del bloque naranja de la creación del fichero XML, se ha diseñado dos estructuras de tipo “case” para minimizar la ventana del menú principal si se ejecuta esta subrutina y maximizar la que se encuentra dentro del bloque, de igual forma que si se finaliza con el programa interno, el case mostrado se ejecutará maximizando el presente menú.

Si el usuario accede al bloque anterior, mostrado en la imagen, podrá contemplar el código de LabView (que podrá visualizar con mayor detalle en EL Anexo C del proyecto) en donde cabe comentar los siguientes aspectos constructivos de la subrutina descritos de izquierda a derecha dentro de la ventana de diseño del software:

- Reinicio de todas las variables implicadas en esta subrutina.

- Código encargado de exigir al usuario que introduzca un nombre al fichero XML que contenga más de tres caracteres alfanuméricos, evitando así que se pueda introducir un nombre tan simplificado que luego pueda crear confusión para el usuario además de que, infringiendo el mínimo de tres caracteres implementados en el nombre, no se podrá habilitar los posteriores accionamientos de aceptación o cancelación de la creación del fichero XML. Si el usuario no cumple con

ello lo único que podría optar sería por accionar el botón de “volver”, regresando al menú anterior (Véase figura 4.36.):

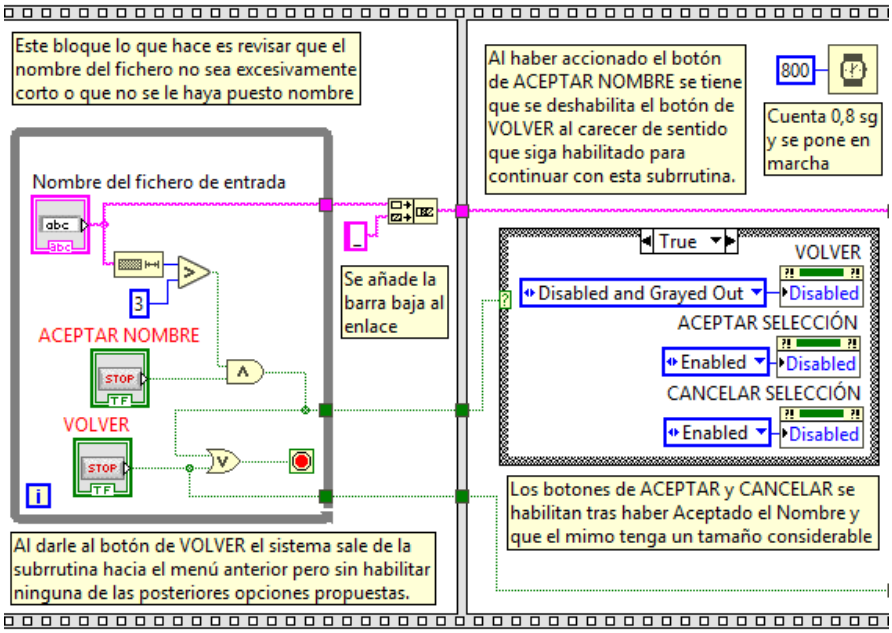


Figura 4.36. Código LabView asignación nombre fichero XML y habilitación de los botones de aceptación y cancelación.

- Adjunta la fecha y hora de la creación del fichero, que quiere decir que en el caso de que no se haya accionado el botón de “volver” (habiéndose accionado el de “aceptar” nombre del fichero XML) la estructura “case” del código de LabView que se puede contemplar en la siguiente ilustración (figura 4.37.) reanudará la ejecución de la subrutina asignándole al fichero, una fecha de creación junto con la hora, empleando concatenación de variables tipo “string”. Tras esta acción el programa generará una ventana que el usuario podrá ver, gracias el bloque “file dialog” que permite interactuar con él, exigiéndole que indique el directorio donde se guardarán todos los datos generados en el fichero durante la ejecución del programa, el cual siempre accederá a la URL que se genere tras este mencionado bloque.

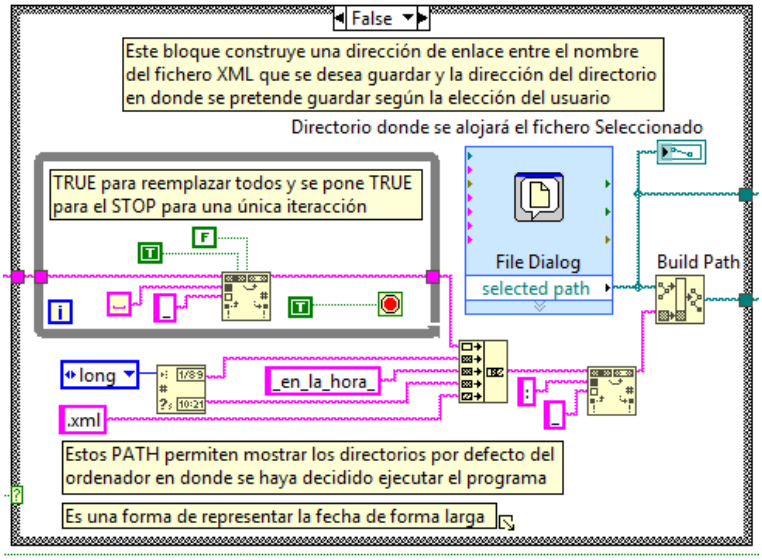


Figura 4.37. Código LabView asignación fecha creación del fichero XML.

- Posteriormente se programó la opción propuesta al usuario de aceptar o cancelar la operación para que la dirección URL generada del explorador de Windows se guarde o no, respectivamente, en el fichero. Así pues, esta subrutina tendrá como salida la URL de la dirección en el explorador de Windows de dónde se guardará el documento XML.

4.2.1.3.2.2. ESTABLECER COMUNICACIÓN.:

En el establecimiento de la comunicación, como se comentó en capítulos anteriores, se realiza mediante un servidor OPC. En él se recogerá la URL de lectura, variable que proporcionará la información medida, en este caso, en forma de tensión del sensor conectado al sistema (en otros casos con otros sensores la correspondiente medida podría ser a través de una señal de corriente), así como la URL de escritura, la cual permitirá al usuario cambiar el valor numérico de la variable manipulada (en la planta de este proyecto representará la potencia de la bomba hidráulica) Como se puede contemplar en la siguiente imagen (Figura 4.38.) para acceder a la subrutina de comunicación se tiene que accionar el botón booleano de comunicación OPC, accediendo al bloque interior naranja. Un aspecto que se debe contemplar en la imagen siguiente es la idéntica estructura de código implantado en el "case" que contiene al bloque de la comunicación a través de un servidor OPC y es que vuelve a aparecer la invocación del nodo que hace referencia a maximizar la ventana del menú principal de tareas propuestas que se activará cuando el usuario haya terminado de utilizar el bloque de comunicación con el servidor OPC.

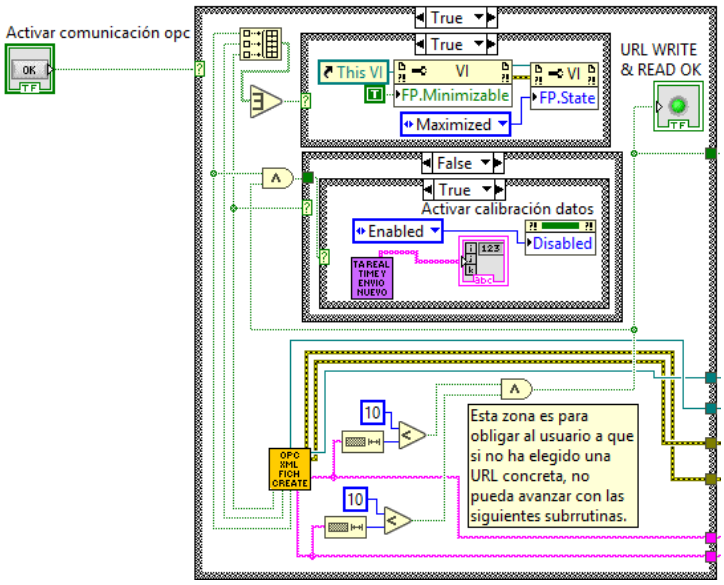


Figura 4.38. Código LabView establecimiento de la comunicación.

Obsérvese que además de este bloque existe otro adyacente de un color morado que es de diseño único y se empleará en las restantes tareas del programa, cuya misión es rellenar el panel de los datos que podrá ver el usuario a medida que complete las mismas. Ambas URLS se obtendrán en esta subrutina bajo las siguientes restricciones:

- El programa obligará al usuario a seleccionar ambas URLS evitando que alguna quede sin registrar, de cara al correcto funcionamiento de las posteriores tareas que hagan uso de estas direcciones (Véase figura 4.39.):

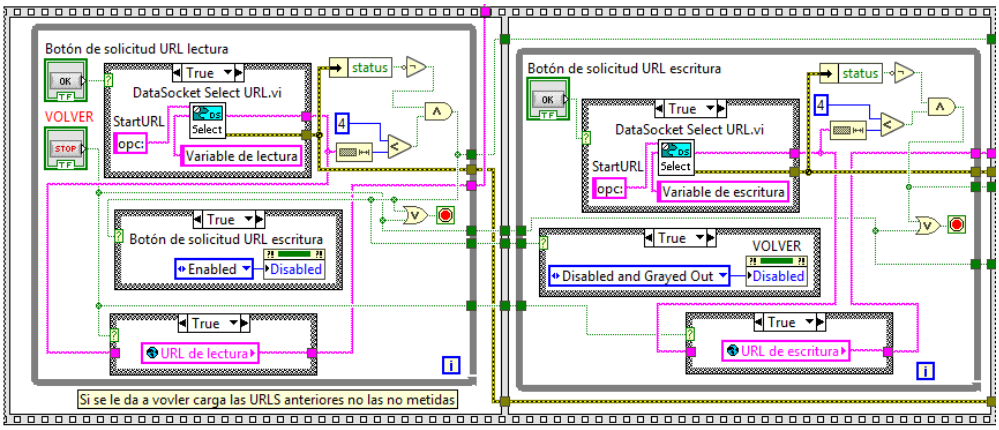


Figura 4.39. Código LabView obtención URL de lectura y escritura.

- El programa reconocerá errores de conexión que generen URLs aleatorias carentes de sentido. Para ello lo que se diseñó fue emplear

el estado del registro de errores que genere el bloque "Select" de la estructura del "case" en donde se podrá elegir la correspondiente URL ya sea para lectura como de escritura. Ante esto, si el bloque genera un error de la indistinta naturaleza que posea, quedará registrado usándose para volver a preguntarle al usuario por la correspondiente URL anulando, posteriormente, dicho error que se generó. Puede verse en la imagen anterior (figura 4.39.) lo comentado y que en el caso de que el usuario cancele la selección de alguna de las URLS que deberá escoger, el programa volverá a solicitarle dicha URL a menos que accione el botón de "volver" que se proporciona en el panel inferior de los indicadores textuales de las URLs.

- Guardará en el fichero XML ambas URLs que se han registrado en el programa para el posterior estudio de experimentación y diseño de un controlador PID. Téngase en cuenta a partir de ahora, que el bloque de color rojo que se mostrará en la imagen siguiente (figura 4.40.) se empleará minuciosamente para poder cargar los datos del fichero XML que se habían guardado en él además de guardar los datos que se consideren oportunos.

  Si bien es cierto que ambas decisiones, de carga y almacenaje, podrían colapsar de cara a una primera idea de programación de este bloque para carga y almacenaje, se resolvió aquel detalle empleando dos constantes booleanas de valor fijo para cada ocasión que corresponda una acción u otra; si la constante es TRUE su acción será la de guardar los datos y si es FALSE su acción será la de carga:

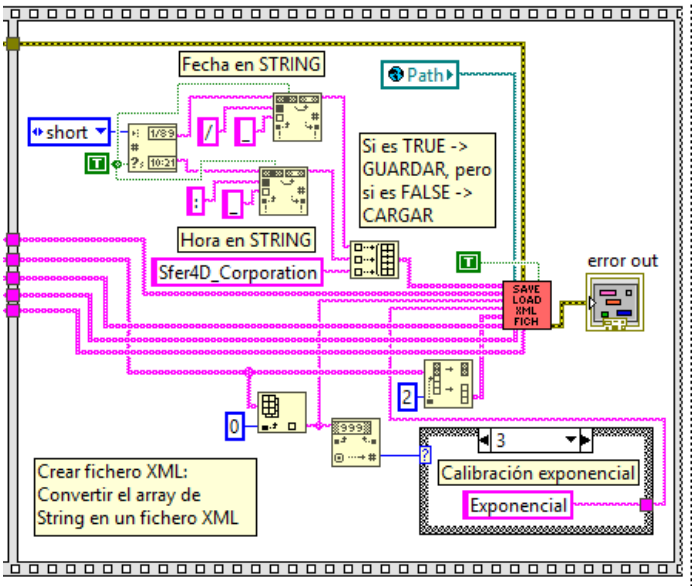


Figura 4.40. Código LabView carga y almacenaje en fichero XML.

Los cinco hilos gruesos de color rosa situados en el margen izquierdo de la imagen corresponden con la información guardada en una matriz de strings que desembocarán en el bloque de color rojo antes mencionado. La subrutina alojada en dicho bloque posee una entrada booleana y como bien se comenta en el recuadro que tiene encima, si la variable envía el valor "TRUE", el programa interno habilitará el modo de almacenaje de los datos que reciba, sin embargo, su el valor enviado es "FALSE" el bloque ejecutará el modo de carga de datos del fichero XML, motivo por el cual se necesite la dirección URL del explorador de Windows para poder realizar cualquiera de ambas ejecuciones. Se puede ver, además de todo lo comentado, que en la imagen se agrega la fecha, hora y nombre de la empresa que lo diseñó. Estas configuraciones sólo aparecerán en esta subrutina, más en el resto de ellas quedará implícito y el programa sólo se limitará a trabajar con los datos.

En lo que respecta a las posteriores tareas, sólo se expondrán en esta zona porque en la zona del programa secundario (cuando el usuario ejecute la subrutina que le permita reanudar el estudio que dejó sin terminar, cargando previamente los datos de un introducido fichero XML) todas las tareas que a continuación se detallarán serán idénticas más así se implementó para simplificar las posibles configuraciones que pudieran existir en el futuro.

4.2.1.3.2.3. CALIBRACIÓN DE DATOS:

En lo que se refiere a calibración de datos, es la primera de las subrutinas que se utiliza para poder comenzar a hacer las correspondientes experimentaciones en lazo abierto, lazo cerrado y sintonía PID. Es importante realizar una calibración lo más eficiente posible debido a la dependencia de las demás tareas en ésta. Si el calibrado es erróneo, el diseño del controlador PID será bastante complicado que se ajuste a la planta cuyo funcionamiento se pretende analizar y controlar. En la siguiente imagen se podrá contemplar el código de la parte correspondiente a la calibración de los datos recibidos (figura 4.41.). Para poder entender el código que se ha programado en esta zona se analizarán los siguientes parámetros:

- La presente subrutina sólo se ejecutará cuando se tenga constancia de haber establecido una previa comunicación con un servidor OPC y adquirido las respectivas URLS de lectura y escritura. Para poder realizar aquella tarea se empleó una simple puerta AND con las variables booleanas de comunicación y activación de la calibración involucradas en el resultado de dicha operación lógica.
- La calibración necesitará tener adjunto el bloque morado que antes se comentó el destino que poseía de cara a mostrar información al usuario en tiempo real.

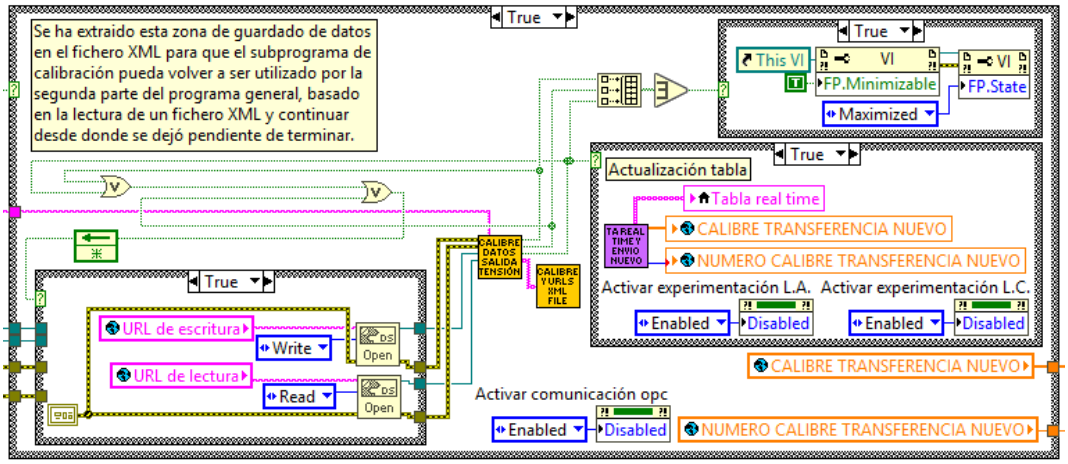


Figura 4.41. Código LabView calibración de datos.

- En la imagen anterior (zona superior derecha) se puede ver el mismo "case" referente al tratamiento de las ventanas maximizadas, que como se comentó anteriormente, estará presente en todas las tareas.

- El programador puede haberse dado cuenta del detalle de que en la imagen (figura 4.41) existen tres bloques, dos naranjas y uno morado, si bien el software se pretende que sea simétrico respecto a la forma de editar de manera similar las tareas que han sido tratadas, la imagen en cuestión irrumpe esa pretensión respecto a lo programado para el establecimiento de la comunicación (figura 4.38.) y ello es debido a que la tarea de calibración (y las posteriores) son bloques de LabView que se utilizan tanto para comenzar un estudio nuevo como para reanudar uno antiguo y es por esto que el bloque extra que aparece junto al de calibración es el que representa el almacenaje de los datos en el fichero XML. El problema estuvo en cómo programar si es un estudio nuevo el que se está ejecutando o uno antiguo. La solución ante esta problemática se encuentra en ese tercer bloque que será imperativamente diferente para cada una de ambas ocasiones; estudio nuevo y estudio antiguo. En la siguiente imagen se podrá ver con más detalle el conjunto de los tres bloques mencionados:

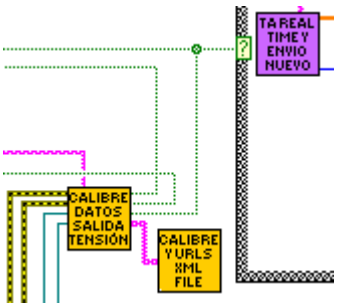


Figura 4.42. Código LabView bloques tratamiento de datos.

El bloque "calibre y urls xml file" (figura 4.42.) tendrá las mismas características que todos los bloques que acompañen a una tarea en concreto ya sea, calibración, experimentación en lazo abierto, experimentación en lazo cerrado o diseño del controlador PID y que además sea de un nuevo estudio con un nuevo fichero XML o uno antiguo con un fichero XML que ya posea datos en su interior de alguna de las tareas realizadas en este software.

Los detalles de este bloque se comentarán en apartado de "SUBVI DE ALMACENAMIENTO Y CARGA DE DATOS" junto con las explicaciones del código que reside en el bloque de color rojo que se mencionó líneas más arriba (figura 4.40.) cuyas funcionalidades de ejecutar los similares procedimientos es bastante análogo al bloque naranja adicional anterior.

Al acceder al interior del bloque de calibración se tienen los siguientes aspectos a considerar para comprender la programación existente. De entre ellos se citan los siguientes:

- Cuando se van adquiriendo una serie de datos obtenidos del sensor, el usuario podrá ir accionando el botón de "selección de elemento" para registrar aquellos datos de interés junto con su correspondiente valor en las unidades del sistema internacional a las cuales pertenezca. A medida que vaya registrando esos valores de entrada (estructura del case de la izquierda en la imagen, figura 4.43.), un array irá llenándose con los mismos dejando un espacio en blanco para el siguiente valor introducido. En este proceso un usuario puede introducir un valor con el que no esté conforme, en ese momento podrá hacer uso del borrado del elemento introducido (estructura del case de la derecha en la imagen, figura 4.43.) Se ha de tener en cuenta que no se podrán borrar más filas que las que se habían introducido y ello lo contabiliza la llamada a la función "NumRows" mostrada como bloque naranja con rótulo azul. Otro de los posibles problemas que podría existir sería ejecutar consecutivamente el borrado de las filas, provocando que se obtenga filas negativas como consecuencia de una "resta" en la sustracción de filas numéricamente hablando, de manera que ante esta situación el valor por defecto siempre será cero para que no se alcancen números negativos en esta subrutina.

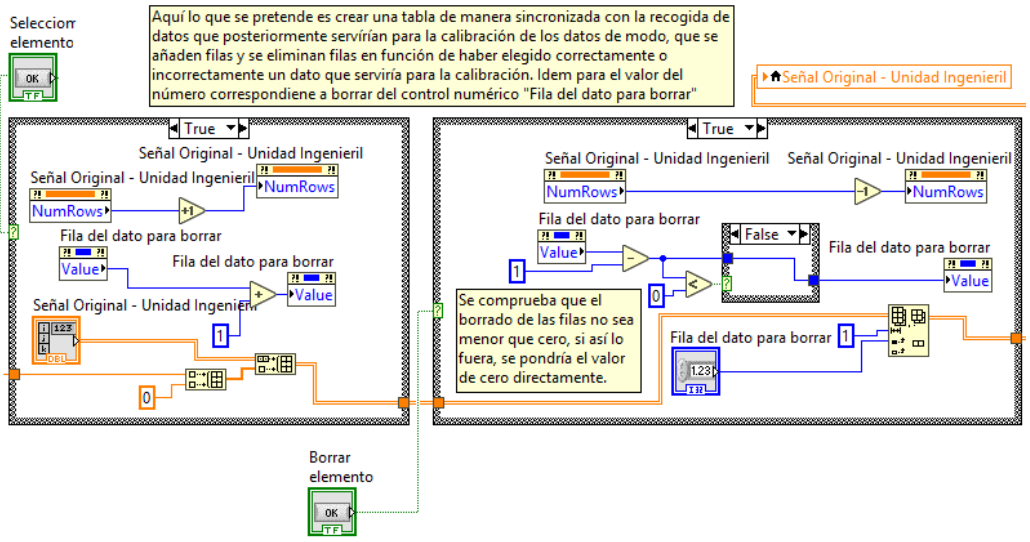


Figura 4.43. Código LabView añadir eliminar filas en toma de datos.

- Otro de los aspectos esenciales para poder programar la calibración de los datos sea el tipo de estudio que sea, ya que depende bastante del tipo de fenómeno físico estudiado en una planta concreta, fue la de establecer un tipo de calibración mediante una variable que portaba un menú desplegable de opciones, de entre ellas: calibrado lineal, polinómica, potencial, exponencial y logarítmica.

  Cada vez que se selecciona un tipo de calibración el usuario verá una gráfica diferente junto con los parámetros propios de dicha calibración haciendo que visualmente el resto de las variables pertenecientes a los resultados obtenidos en otras calibraciones desaparezcan del campo de visión del usuario, no obstante, si no desaparecieran los valores serían nulos todos ellos. Los indicadores correspondientes a una calibración específica únicamente se mostrarán cuando se elija el tipo de calibrado realizado.

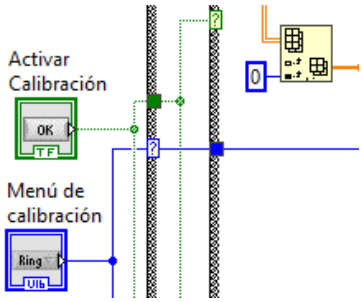


Figura 4.44. Código LabView sincronización de los accionamientos de la activación de la calibración y del tipo de calibración implementada.

- El último de los aspectos que se tiene que considerar es la forma en que se guardan los datos en un posterior fichero XML. Todos los valores de la calibración se convierten a tipo string con el fin de poder emplear la variable global, matriz que también es de tipo string, que corresponda con el estudio nuevo o antiguo que se haya realizado, de ahí que se encuentre en esta imagen (Véase figura 4.45.) la variable booleana que gobierne sobre el tipo de estudio que se había realizado dadas las circunstancias de compartir este bloque en las subrutinas del nuevo y antiguo estudio. La misma variable booleana global fue mencionada anteriormente en donde se explicaba la importancia de su implicación en las diferentes tareas de este proyecto.

  Aquí lo que se verifica es diferenciar entre un estudio y otro empleando un case que contiene una variable matricial global tipo string para cada caso; si es TRUE será un estudio nuevo y en el caso de que sea FALSE, el estudio será antiguo (figura 4.45). Al salir de case, la matriz de datos tipo string enviará la información al bloque naranja exterior que actualizará el fichero XML creado.

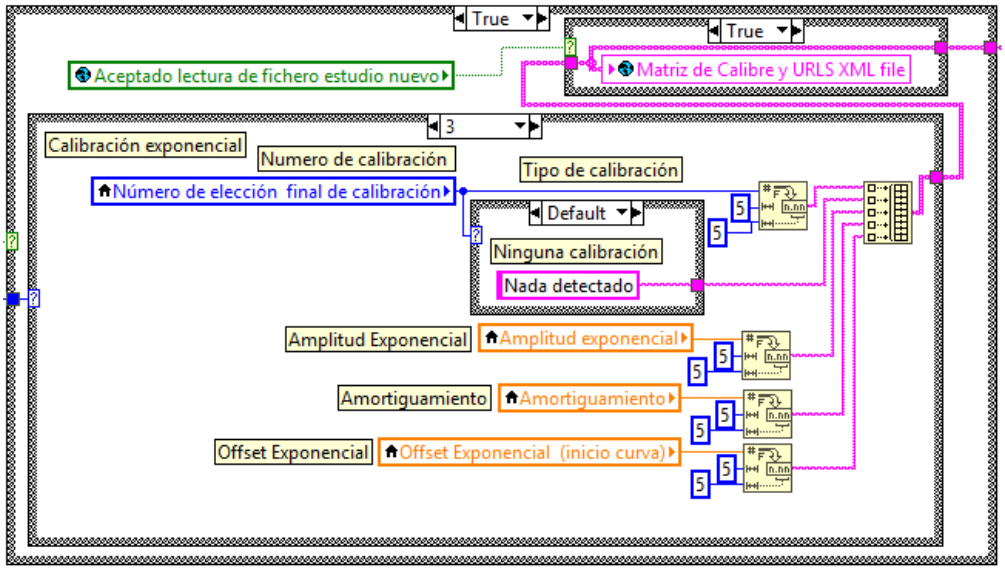


Figura 4.45. Código LabView guardado de datos de la calibración.

4.2.1.3.2.4. EXPERIMENTACIÓN EN LAZO ABIERTO:

En la experimentación en lazo abierto (L.A.) se comentarán los puntos más importantes en el desarrollo de la subrutina en donde bajo la teoría de control referente a la experimentación en lazo abierto, sugiere estudiar el sistema con la curva de reacción tras introducirle una entrada escalón, mediante la cual se

podrán extraer todos los parámetros referentes a este estudio. Recuérdese del capítulo tercero referente a la teoría de esta experimentación en donde los parámetros que se obtendrán son; ganancia (K), constante de tiempo (tau), retardo temporal (d).

Para poder exponer el código que se diseñó para poder llevar a cabo esta tarea primeramente se comentará el bloque que activa esta sección, el cual puede verse en la siguiente imagen (figura 4.46.):

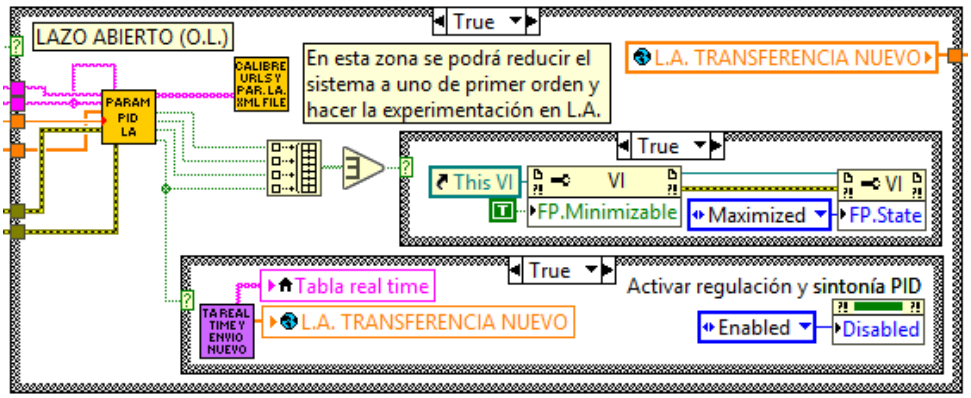


Figura 4.46. Código LabView experimentación en lazo abierto (L.A.).

En donde la similitud con la presentación del bloque de calibración es bastante clara (consúltese figura 4.41.), el procedimiento es simple, más una vez terminada la tarea de la experimentación en lazo abierto se podrá habilitar el diseño del controlador PID, teniendo en cuenta que no hará falta habilitar la tarea de experimentación en lazo cerrado debido a que es indiferente qué configuración haya sido estudiada si bien es cierto que el controlador se podrá diseñar con tener calculadas una de ambas experimentaciones.

Dentro del bloque correspondiente a la experimentación en lazo abierto se tendrán en cuenta las siguientes consideraciones para poder programar los procedimientos que optimizarán la funcionalidad de dicha tarea:

- Como es habitual se reiniciarán a un valor por defecto generalmente nulo para variables numéricas y carencia de caracteres para variables de tipo string, todas las variables que estén implicadas en esta subrutina.

- Otro de los aspectos es la técnica empleada para poder almacenar en una matriz de datos numéricos en tiempo real desde el momento en que el usuario decide comenzar con dicho almacenaje hasta el instante en que queda conforme y decide detener el proceso de dicho guardado, momento en que la curva de reacción se puede estudiar tras haber

introducido una entrada escalón, cuyos valores de dicho salto están dentro del intervalo que el usuario registró de los datos almacenados para la experimentación en lazo abierto. En la siguiente imagen (figura 4.47.) se puede contemplar que para poder programar la decisión que tomaría el usuario para comenzar a guardar los datos de la curva de reacción hasta que concluye dicho almacenamiento se empleó una variable booleana que en la propia imagen se puede visualizar denominándose "INICIO ALMACENAMIENTO DE LOS DATOS DE ENTRADA", la cual se va almacenando su valor en cada iteración que dará la estructura "do while", pudiéndose observar que en la matriz generada de valores booleanos, sólo aparecerá el TRUE una vez seguido de varios FALSE dada las consecutivas iteraciones posteriores. De la misma manera se llenará otra matriz de valores lógicos que independientemente a la anterior habrá un instante en que el valor FALSE que se repetirá sucesivamente cambiará a TRUE, momento en que el usuario decidió concluir con el almacenamiento de los datos de la curva de reacción.

El motivo de ambas matrices de valores booleanos reside en poder conocer en qué iteración se obtuvo el primer valor TRUE de inicio de almacenaje de los datos y en qué otra ocasión se observó que volvió a salir el valor de TRUE de fin de almacenaje, de manera que cuando se sepan en qué índices de las respectivas matrices se posicionaron los valores TRUE, el programador podrá recortar de la matriz general de datos recogidos el rango de todos ellos que se encuentran entre los intervalos en que los índices de las variables TRUE se han posicionado.

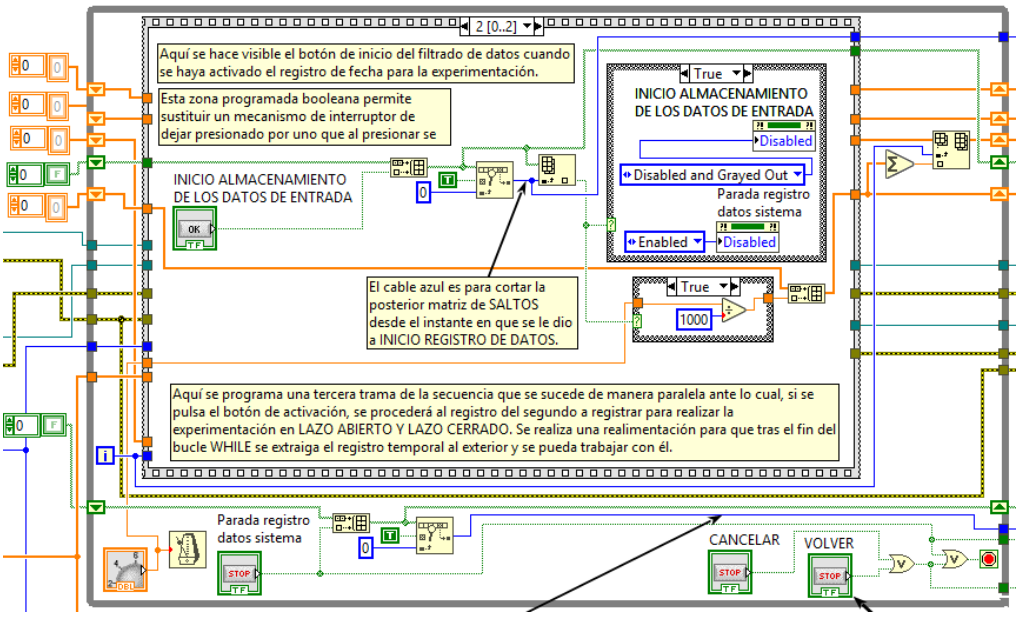


Figura 4.47. Código LabView almacenaje de datos de la curva de reacción.

Cuando se tenga el intervalo de los datos de la curva de reacción el siguiente paso será tratar la entrada escalón que se introdujo.

- En la siguiente imagen (figura 4.48.) se podrá observar cómo se extrae la magnitud del escalón introducido. Hay que tener en cuenta que el usuario ha podido introducir varios escalones para poder acomodar la resultante curva de reacción a un punto numérico inferior a uno concreto por razones de lentitud para alcanzarlo directamente o por simplificar la estimación proporcionando al sistema un escalón cuyo intervalo se sitúa entre dos valores numéricos pequeños. Ante eso el programa tiene que saber los valores correspondientes al escalón que decidió dar al sistema al cual pertenece la curva de reacción de los datos que se habían almacenado previamente. Para ello se considera que, tras comenzar el almacenaje de los datos de la curva de reacción, el usuario podrá dar los escalones que desee siendo el último dado, aquel que servirá de estudio en la experimentación en lazo abierto.

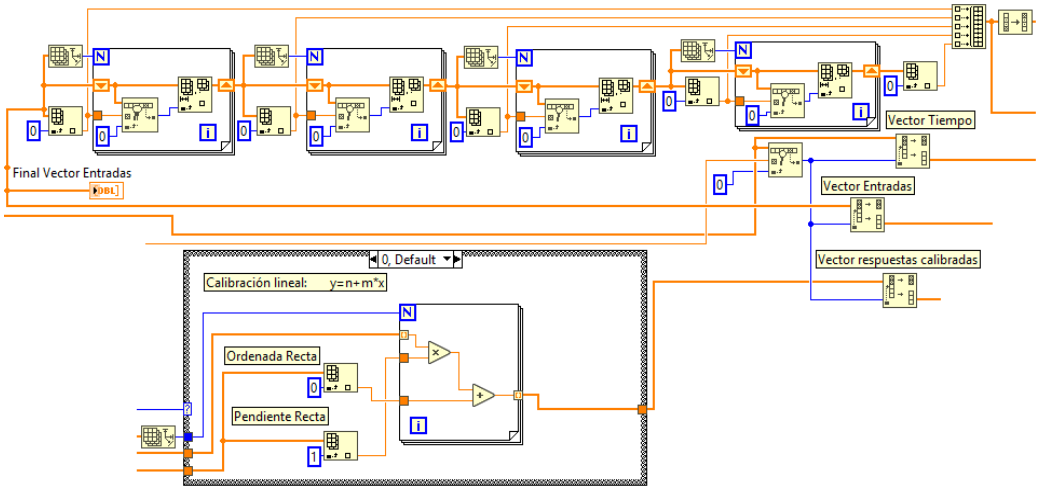


Figura 4.48. Código LabView intervalo curva reacción y aplicación calibrado.

Después de ello el programa fue diseñado para que los datos de la curva de reacción quedaran modificados por la aplicación directa de los parámetros de la calibración que se realizó anteriormente. En la imagen se puede ver que la estructura “case” permite modificar los datos que entran por la izquierda proporcionando en su salida los datos cambiados por el calibre.

- Después de tener los datos de la curva de reacción y los valores que se tienen de la entrada escalón lo siguiente será proceder con el cálculo de los parámetros de la experimentación en lazo abierto. Dichos parámetros son calculados con meras operaciones aritméticas junto con el tratamiento de matrices. Para poder extraer los tiempos

correspondientes al 28,3% y 63,2% de la salida que otorga el sistema se ha procedido a diseñar la siguiente ilustración (figura 4.49.):

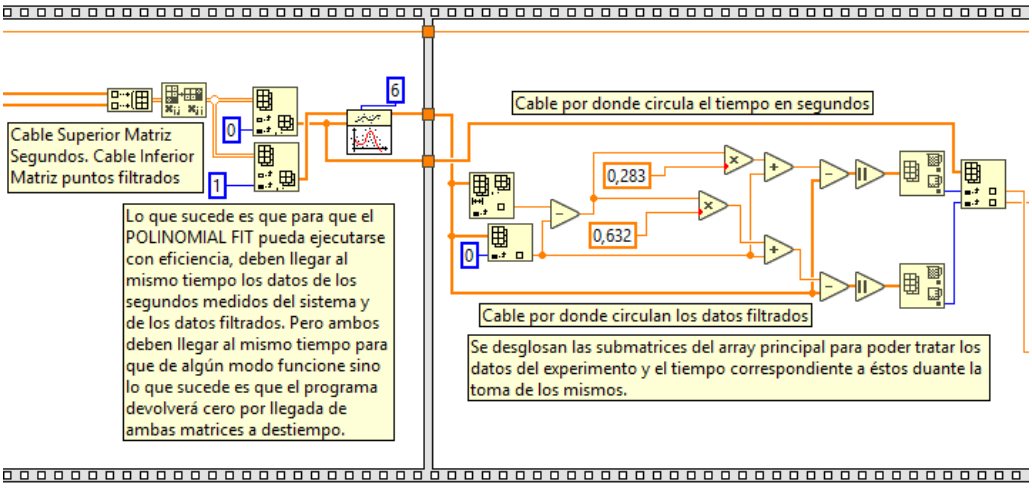


Figura 4.49. Código LabView extracción de los tiempos 28% y 63% de la salida del sistema para el cálculo de los parámetros en lazo abierto.

En la imagen se puede ver como el tiempo y los valores tratados con la calibración para la construcción de la curva de reacción, se complementan para poder componer una unidad con la que, en la segunda trama de la estructura secuencial, se obtienen los tiempos estimados cuando se alcanza respectivamente e 28,3% y el 63,2% de la respuesta que proporciona el sistema en su salida.

- Antes de que la subrutina pueda extraer los parámetros de la experimentación en lazo abierto, se tiene que diseñar un código en LabView para que los datos obtenidos por la planta queden perfectamente filtrados para eliminar las posibles señales de ruido que han interferido en la toma de datos y que de no existir dicho filtrado podrían interferir en la obtención de los parámetros de la experimentación. En la siguiente imagen se puede ver como se ha configurado el diseño de este código para el filtrado de los datos coleccionados con posibles datos provenientes de una señal de ruido generado por la planta.

Figura 4.50. Código LabView filtrado de la señal de entrada calibrada.

El bloque que filtra dicha señal de entrada es el que tiene el código alojado en el cuadrado verde azul claro, cuyo código interior es el que se muestra en la ilustración siguiente (figura 4.51.), lo que se pretende es el suavizado de los datos obtenidos realizando la media de los tres últimos datos registrados, denominado como media móvil:

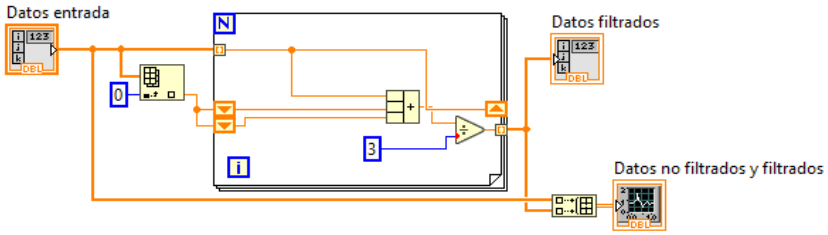


Figura 4.51. Código LabView interior bloque filtro de señales ruidosas.

- Por último, para poder guardar los datos correspondientes al estudio que se haya realizado, nuevo o antiguo, se procederá de la misma manera que se hizo en la forma de almacenar la información extraída en la calibración, pero en este caso con los datos numéricos obtenidos en esta experimentación (consúltese figura 4.45.)

4.2.1.3.2.5. EXPERIMENTACIÓN EN LAZO CERRADO:

En lo que respecta a la experimentación en lazo cerrado cabe comentar lo que se analizó en el capítulo tercero de este proyecto en referente a los fundamentos teóricos que se desarrollaron con la finalidad de poder conseguir en esta tarea la ganancia crítica Kc, y con ello el período crítico Tc, para ello el usuario ha de ejecutar el bloque que se puede visualizar en la siguiente imagen (figura 4.52.) Como se ha hablado en el caso de la experimentación en lazo abierto y dada la similitud del código en LabView, los tres presentes bloques tienen la funcionalidad que en su caso se argumentó con detalle, siendo el más colocado a la izquierda, el que se explicará a continuación:

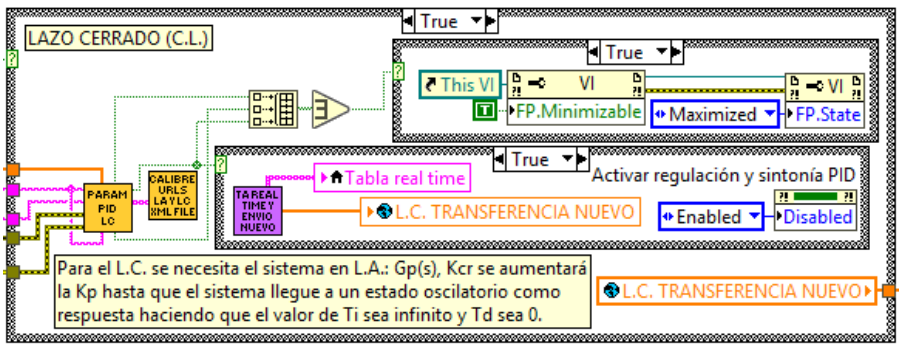


Figura 4.52. Código LabView experimentación en lazo cerrado (L.C.).

Si el usuario se introduce en la experimentación en lazo cerrado podrá ver el código en LabView que se ha diseñado y ha permitido conseguir programar esta tarea. Dicho código se detalla comenzando con la siguiente imagen (figura 4.53.):

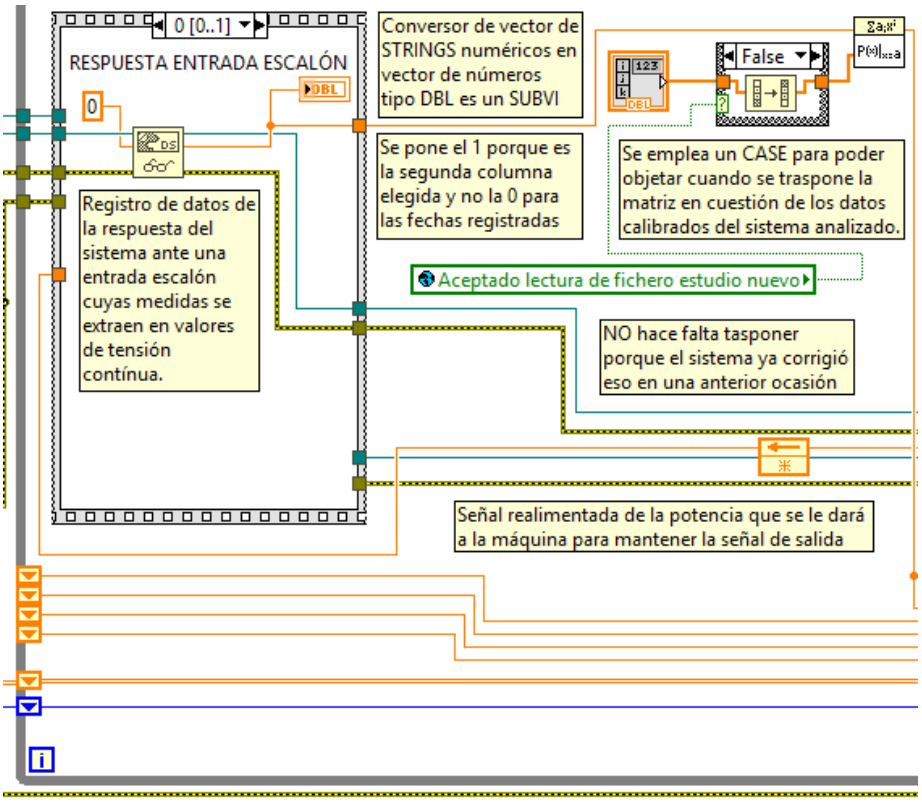


Figura 4.53. Código LabView recepción de las URLS de lectura y escritura, toma de datos y aplicación de los parámetros de la calibración.

Previamente a esta imagen, y como se hacen en todas las subrutinas, todas las variables locales se reinician a un valor por defecto para posibles iteraciones posteriores. En lo que se respecta a la ilustración, en el margen izquierdo se puede ver unos hilos de color verde oscuro que hacen referencia a la entrada de las URLS de lectura y escritura, empleados con el fin de poder generar una colección de datos en tiempo real en función de los escalones introducidos para poder obtener la ganancia y el período crítico. Obsérvese que al ser un bucle “do while” el programa recogerá todos los datos que se necesiten para llevar a cabo la tarea en cuestión hasta el momento en que el usuario decida detener la recolección de los mismos, instante en que se podrá calcular el período de manera visual. La estructura de ejecución simultánea, derivada de una secuencia, permite obtener los datos no calibrados de la planta al mismo tiempo que generar la entrada escalón, habiendo esperado hasta el instante en que ésta se haya realizado.

Obsérvese además que aquí vuelve a aparecer de nuevo la variable booleana referente a qué tipo de estudio había sido activado, el nuevo o el antiguo. El motivo de incluirla aquí reside en que para cada uno de ambos estudios se podría haber elegido un tipo de calibración diferente con unos parámetros de calibración, así mismos, distintos de entre sí. Dada la naturaleza de ambos estudios, los parámetros aparecen colocados en la matriz, que los porta, de una forma u otra pudiendo elegir de manera automática el valor que se le asignará a la estructura "case" para que se le aplique la operación de lectura de la información desde el último dato hasta el primero, recorriendo toda la matriz:

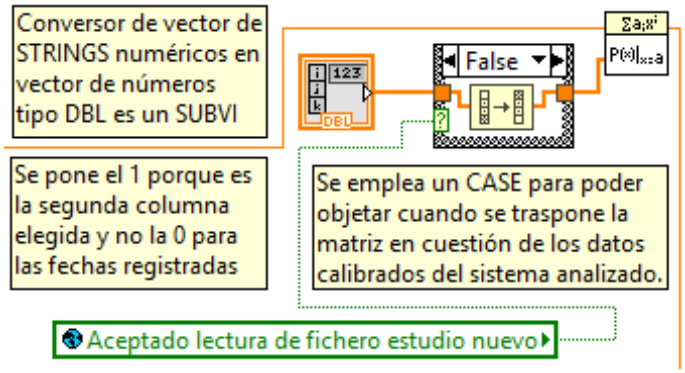


Figura 4.54. Código LabView variable booleana que dictamina que estudio, nuevo o antiguo, se está ejecutando actualmente.

Debido a esto, si no se colocara la variable en cuestión, el sistema daría por validado el hecho de recibir los parámetros de una calibración que no pertenecen al estudio actual pudiendo crear problemas en la obtención de los parámetros que se pretenden conseguir en esta tarea. Así pues y como anécdota del detallado código de programación, uno de los problemas que cundió en el diseño del presente software fue que el programa recibió los parámetros de una calibración lineal cuyos valores numéricos estaban errados en la recepción de manera que lo correspondiente a la ordenada lo consideraba pendiente (negativa en su caso) y viceversa, provocando que ante entradas escalón pequeñas, generara una respuesta rápida e inestable, pudiéndose creer que el sistema era sensible ante perturbaciones mínimas, por tal error no considerado previamente. De manera que tras varios análisis en el código se planteó tal método simple de elección del tipo de estudio que se llevaría a cabo por parte de la elección del usuario. Seguidamente, se planteó el poder conseguir una ganancia, la más exacta posible, para poder lograr que el sistema tuviera una respuesta oscilante, momento en que se podrá detener la recolección de datos provenientes de la planta de procesos y proceder con la obtención del período crítico. Para poder programar esta sección de la tarea en concreto, primeramente, se mostrará el código en LabView que se diseñó percatándose el lector de los siguientes puntos de dicha programación:

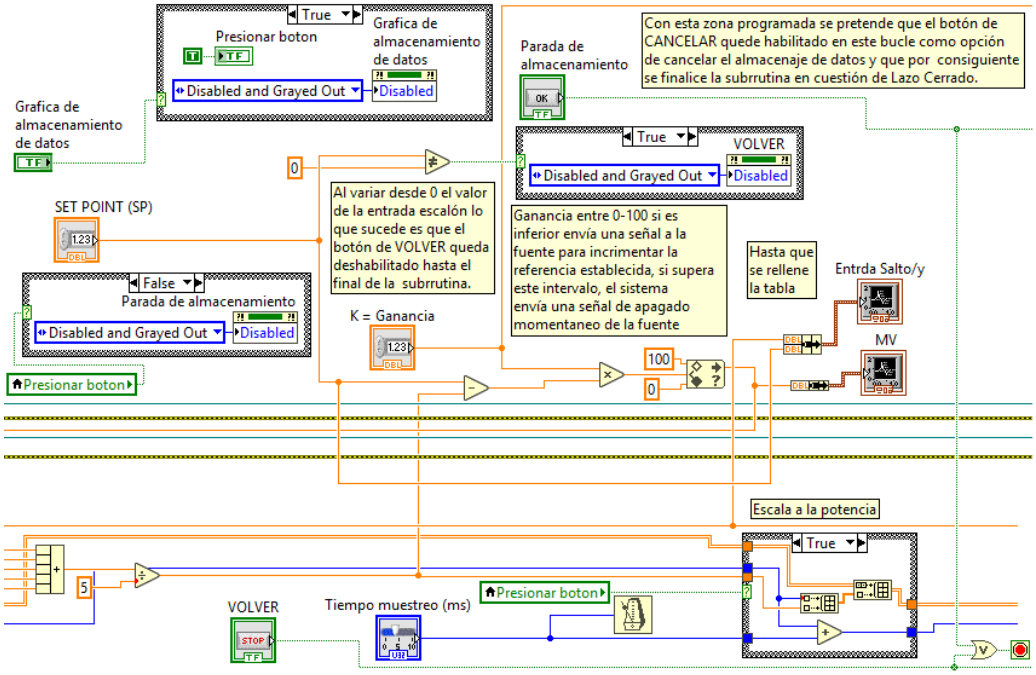


Figura 4.55. Código LabView obtención de la ganancia crítica.

En la zona superior izquierda (Véase figura 4.55.), las estructuras "case" únicamente inhabilitan los accionamientos correspondientes al de almacenaje de datos provenientes de la planta estudiada. Hasta ese botón, el resto de los accionamientos permanecerán inoperativos, siendo objeto que después de activar éste, se detenga esta tarea para proceder con la obtención del período crítico. Para poder comprender el motivo por el cual se consigue que el sistema alcance una respuesta oscilante, se empleó el uso del código del bloque "In Range and Coerce Function" (Véase figura 4.56.) Cabe mencionar que para mejorar los valores numéricos como respuesta del sistema ante varias entradas escalón, se ha empleado la técnica de la media móvil con los últimos cinco valores obtenidos en el bucle while que engloba a todo el código mostrado en la figura 4.55 anterior:

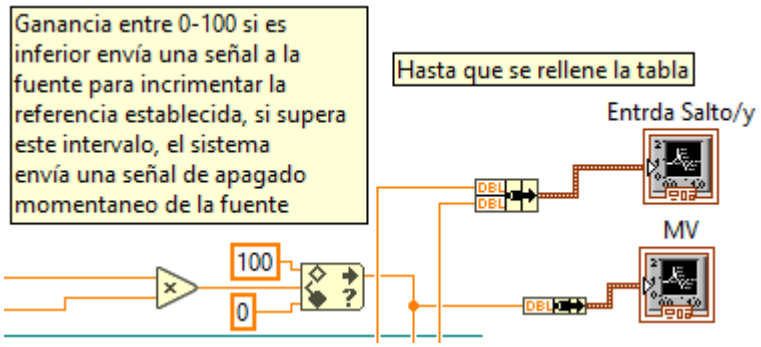


Figura 4.56. Código LabView función rango y obligatoriedad de mantener los datos calibrados de entrada entre dos constantes numéricas.

Dicho bloque permite controlar los datos de entrada calibrados dentro de un rango de dos valores establecidos constantes de la manera en que, si el primer valor supera el límite superior o rebasa el límite inferior, el programa detendrá la acción de la variable manipulada o introducirá un valor concreto en la misma para devolver al sistema al rango establecido, respectivamente.

Así pues, el valor que el software introducirá en la variable manipulada dependerá de la ganancia que el usuario haya colocado en el sistema de manera que, si dicho valor es elevado, la variable manipulada tendrá un valor grande y devolverá, con bastante rapidez, los datos de entrada al rango establecido. Al contrario de lo mencionado, si la ganancia introducida es pequeña lo que se conseguirá es, que tras superar alguno de ambos límites, el programa otorgará a la variable manipulada un valor pequeño, provocando que sea lento el proceso de devolver los datos al rango mencionado anteriormente. Cada uno de ambos procesos se podrá visualizar en las gráficas que se adjuntan seguidamente del bloque que se ha analizado en este apartado.

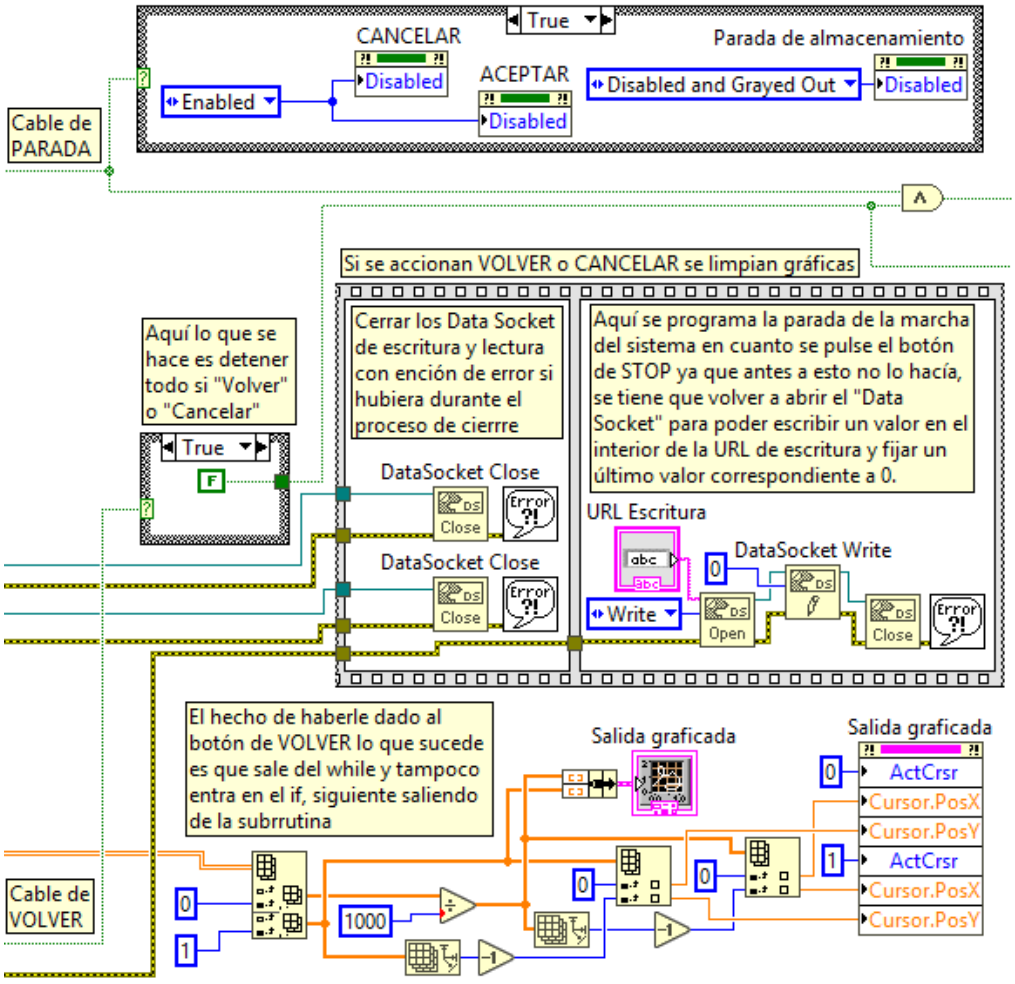


Figura 4.57. Código LabView de detención la recolección de datos.

Posteriormente, en la siguiente parte en que se procede a detener la recolección de datos y fijar la ganancia, entonces crítica, el usuario procederá a conseguir el período crítico con la gráfica de datos ya representada de manera oscilatoria. El código de LabView que representa el punto argumentado es el representado en la imagen anterior (figura 4.57.) en donde el bloque "case" superior ejecuta la decisión de continuar con la experimentación en lazo cerrado al mismo tiempo que se habilitan los posteriores botones de "aceptar" y "cancelar" el proceso una vez se haya calculado el período crítico.

En el caso de haber accionado el botón de "volver" la subrutina se habrá terminado según se comenzó, esto permite al usuario a regresar al menú anterior sin que tenga que accionar ninguno de los botones que esta tarea ofrece en su interfaz.

En la estructura secuencial el programa detiene el estado operativo de lo que la variable manipulada controle mediante variables de tipo socket y por consiguiente lo que realiza es generar el gráfico junto con los ejes de coordenadas X e Y, posibilitando además que la propia gráfica se centre numéricamente en la respuesta oscilatoria (zona inferior de la figura 4.57.). Este hecho facilita al usuario encontrar el período crítico con mayor precisión ya que el efecto zoom ya estará aplicado automáticamente junto con el centralizado oportuno que necesitaría la respuesta para continuar con la experimentación en lazo cerrado. En lo que corresponderá la estructura de tipo "time case" sucederán tres instantes al mismo tiempo, dos de ellos protagonizarán el estado de la planta cuando el usuario decida accionarlos; "aceptar" (Véase en la figura 4.58.) y "cancelar" que no se ve en la imagen pero que corresponde con otro caso dentro de dicha estructura.

Se ha dicho que mientras no se accionen esos botones, la subrutina no se detendrá, de manera que el "case" con el que trabajará hasta ese momento de finalización, corresponderá con el que recogerá los datos de modo que al detenerse el programa se "capturará" el evento completo recopilando todos los puntos que fueron almacenados entre dos instantes de tiempo propuestos por el accionamiento de anteriores botones que el usuario pudo ejecutar a su gusto.

Debido a esta estructura de "time case" el programador podrá comprender su importancia en lo referente al funcionamiento de la recopilación de datos, aceptación de los mismos (con su posterior cálculo automático del período crítico) o, por el contrario, la cancelación de dicha operación, sin que, con ello, queden guardados en el fichero XML lo trabajado en esta tarea de experimentación.

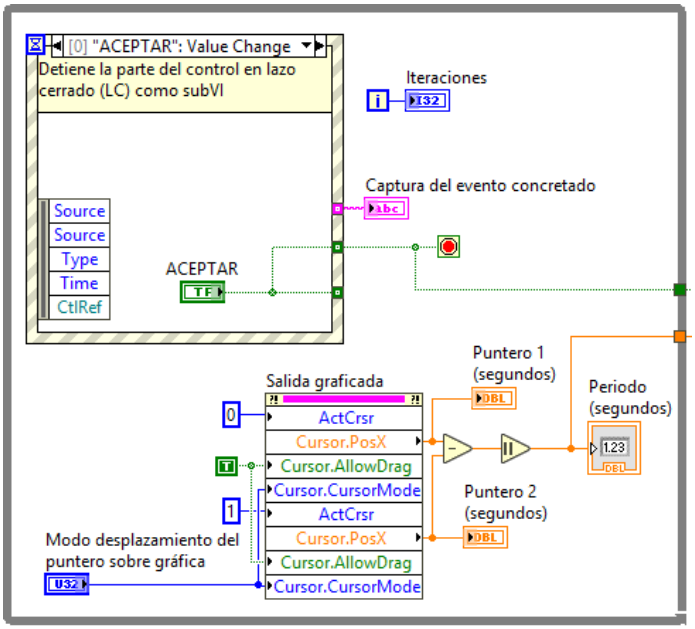


Figura 4.58. Código LabView de obtención del período crítico.

Por último, esta sección de la tarea correspondiente al lazo cerrado se encargará de, obtenida la gráfica de la respuesta oscilatoria, poder conseguir el valor numérico del período crítico. Simplemente hay que mencionar que, tras la detención, el usuario podrá ir recorriendo las sucesivas crestas de la respuesta mientras se registran las iteraciones que se hicieron para poder conseguir el período en cuestión. El recorrido lo harán dos punteros de colores diferentes para la identificación permitiendo al usuario ver cómo, al ser movidos, se verá la magnitud en segundos que tendrán en cada punto de la respuesta observándose que su diferencia siempre será, si están perfectamente colocados, el período crítico buscado.

La última parte de esta subrutina será exactamente igual que el resto de las que se vieron anteriormente (Ver figura 4.59.) en donde cabe mencionar de nuevo la diferencia que se ejecuta con la variable booleana en referente al tipo de estudio realizado, enviando dichos datos a una variable matricial global además de a una matriz de STRING que posteriormente se enviará para guardar su contenido en el fichero XML, junto con los otros datos de las anteriores tareas.

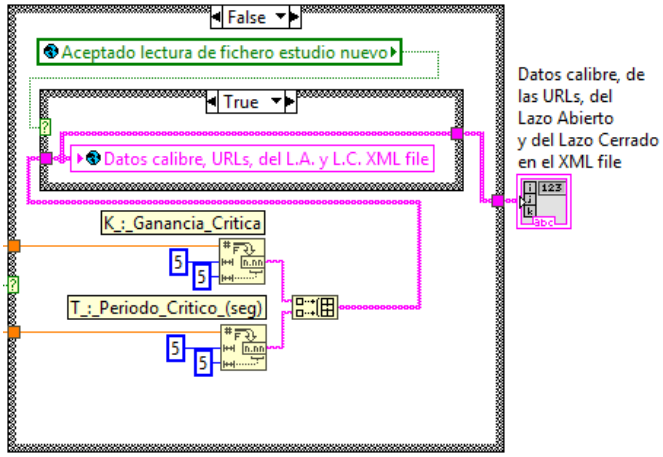


Figura 4.59. Código LabView de colocación de los datos para guardarlos en el fichero XML distinguiendo entre estudio nuevo o antiguo.

Otro de los aspectos que han de mencionar es el accionamiento de "aceptar" que aparece al final de esta subrutina junto con este "case", que no se aprecia en la imagen, pero que permite activar la última de las tareas del software; diseño del controlador PID. Con estos parámetros calculados o con los de la experimentación en lazo abierto, el usuario podrá acceder a la configuración de la sintonía de dicho controlador, por ello entiéndase que no es necesario tener ambas experimentaciones realizadas ya que con una de ambas bastaría para continuar con la siguiente tarea de control.

4.2.1.3.2.6. REGULACIÓN Y SINTONÍA PID:

En esta parte del programa se pretende diseñar el controlador PID para la automatización de la planta de proceso, en esta subrutina se diferenciarán varios puntos del código. A continuación, se verá el bloque (figura 4.60.):

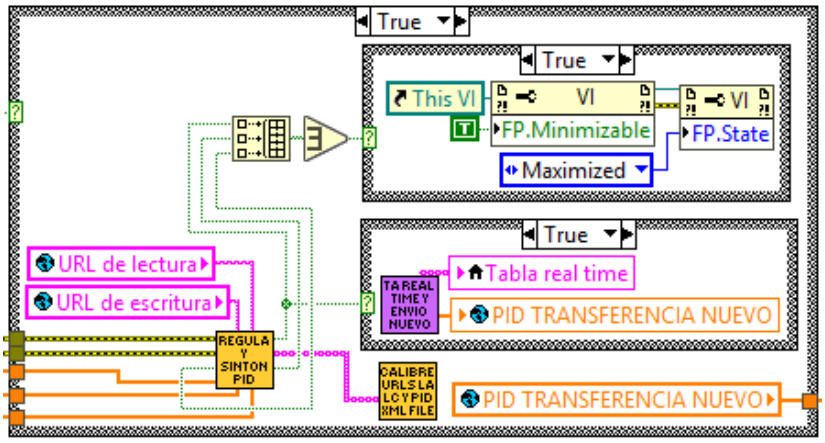


Figura 4.60. Código LabView diseño del controlador PID.

En la siguiente imagen se comenzará con el reinicio de todas las variables implicadas en la compleja operación distinguiéndose, entre las más comunes, las siguientes (Véase figura 4.61.):

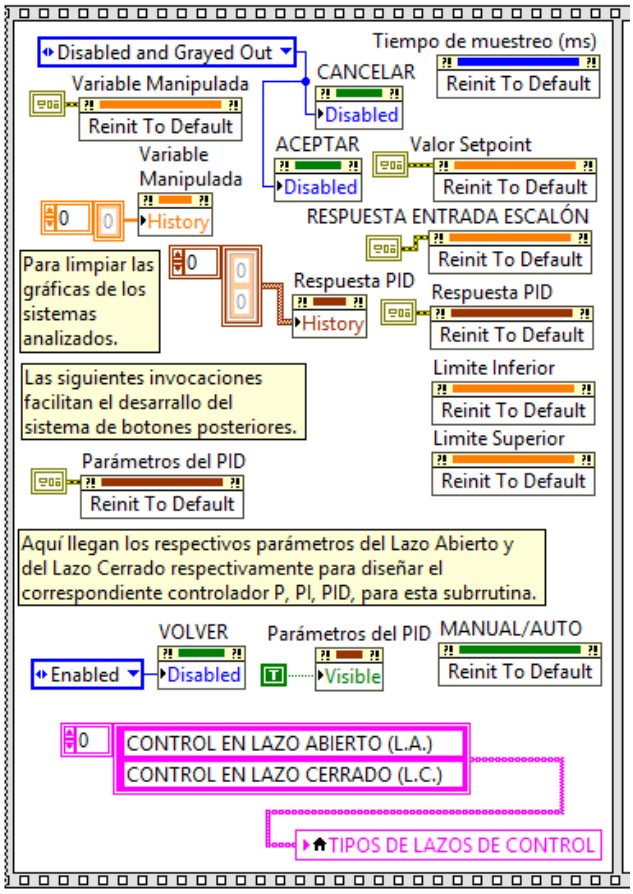


Figura 4.61. Código LabView reinicio variables controlador PID.

Como se argumentó en todas las tareas previas a ésta, programadas para una funcionalidad específica, aquí lo que se reinician son las variables:

- **Numérico:** Para poder usarlas en la configuración del controlador PID, para reiniciar las variables empleadas para graficar la respuesta que proporcione el sistema ante entradas escalón y por supuesto para el reinicio de la variable manipulada.
- **Booleana:** Para poder reiniciar la variable del accionamiento "configuración manual y automática" a manual por defecto. Además de reiniciar a los valores establecidos de habilitación de los botones de "volver", "aceptar" y "cancelar".
- **Nodo local de almacenaje:** Es un tipo de variable que aloja varios tipos y que en esta subrutina se guardarán de manera temporal los parámetros que se calcularon del controlador PID según qué tipo de sintonía se eligió para la automatización posterior.

Este último tipo de variable reiniciada tendrá bastante influencia en la construcción de este módulo programado que posteriormente se comentará con mayor detalle.

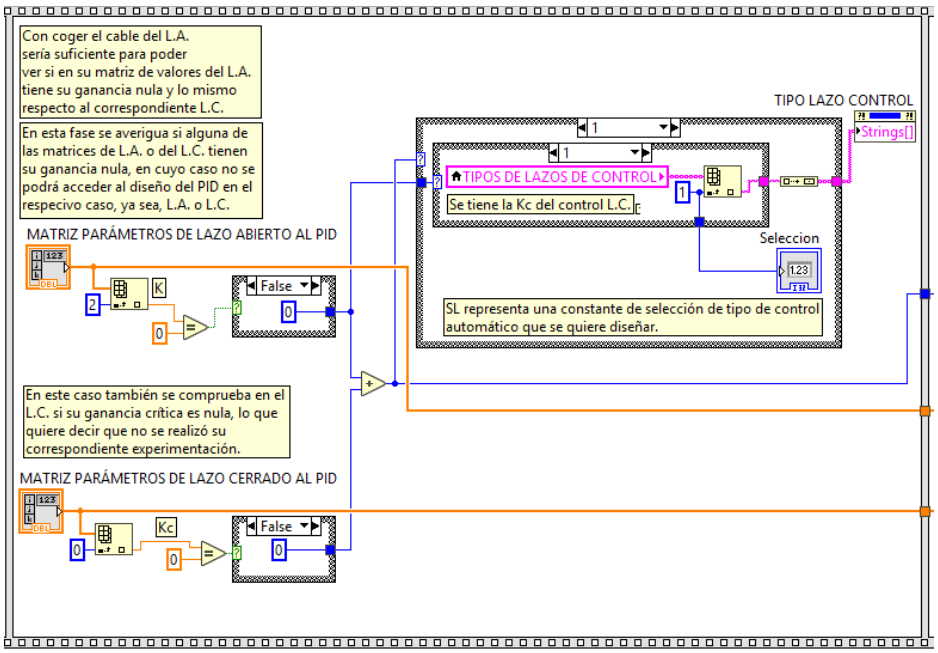


Figura 4.62. Código LabView elección automática del tipo de experimentación en lazo abierto o cerrado que se ha realizado previamente.

Esta segunda parte de la subrutina del diseño del controlador permite distinguir que tipo de experimentación se había realizado, es decir, si el usuario únicamente realizó la tarea de experimentación en lazo abierto, o si sólo fue el lazo cerrado o se realizaron ambas experimentaciones. Existe una cuarta opción en la que el usuario no ejecutó ninguna de las experimentaciones, en cuyo caso jamás podría tener acceso a esta tarea de sintonía del controlador PID más su accionamiento estaría deshabilitado y acceder, por parte del usuario, a esta subrutina sería imposible.

Para poder ejecutar esta decisión automática inteligente, se hace llegar en forma matricial de los respectivos parámetros de ambas experimentaciones en lazo abierto y cerrado con el fin de poder comprobar su contenido. Si la ganancia resulta ser nula quiere decir que no hay datos registrados que confirmen que la correspondiente experimentación debido a que no se puede tener una ganancia cero en ningún caso ya sea; lazo abierto o cerrado, el que se haya ejecutado previamente. En otro caso la experimentación se realizó completamente cuyos datos quedaron guardados en el fichero XML.

Posteriormente, las respectivas estructuras de tipo “case” generan, esta vez, un valor numérico (no booleano) “uno” o “cero” siendo para el primero de ellos

un valor asignado para el caso de que no se tenga constancia de que se hubiera hecho la experimentación correspondiente, mientras que el segundo de los valores representará el hecho contrario. El motivo de ello se debe a maximizar la facilidad con que se ejecutan las posteriores estructuras de tipo "case" anidadas.

Para los valores numéricos generados previamente para ambas experimentaciones, lo que se hace es implementar una operación de suma con el fin de que si el resultado sale "cero" supone que se tienen ambas experimentaciones hechas y el usuario podrá elegir de entre ambos qué tipos elegir para sintonizar el posterior controlador PID.

Sin embargo, si alguna de ambas experimentaciones genera un "uno" siendo el otro "cero", el programa diferenciará con las estructuras "case" anidadas qué tipo de experimentación fue realizada fijando en el menú de opciones, que verá el usuario, los tipos de sintonía que pertenecen a la experimentación ejecutada correctamente. Así pues, la suma de ambos números justo antes de los "case" anidados (figura 4.62.) será siempre "uno" entrando entonces en el primer "case" para comprobar qué experimentación fue la que habiendo sido realizada generó el valor numérico "uno" con el segundo de los "case".

Este último "case" coge el valor numérico de cualquiera de los "case" anteriores, donde desembocaron las matrices con el contenido numérico de las experimentaciones realizadas (en este caso se cogió de la experimentación en lazo abierto) y se utiliza para poder diferenciar cuál de ambas generó el "uno" para poder colocar en el menú de opciones las correspondientes sintonías disponibles.

Después de esta operación, el siguiente módulo adyacente lo que ejecutará será enviar los datos de ambas experimentaciones (aunque exista alguna de ambas que no contuviera dato alguno como se contempló anteriormente, en cuyo caso están vacías todas sus casillas) pudiendo elegir el usuario el tipo de sintonía que prefiera realizar; automática, basada en los métodos expuestos en el capítulo 3 de "fundamentos teóricos" o manual, en donde es él quien elige qué parámetros seleccionar en función de la respuesta que proporcione el sistema, de la planta de procesos, en cuestión. Como se puede ver en la siguiente imagen (figura 4.63.) los datos de ambas experimentaciones llegan a la estructura "do while" en cuyo interior se tiene primeramente, la selección, con una variable booleana, del tipo de sintonía que el usuario podrá elegir (estando en modo manual por defecto al iniciarse la subrutina) téngase en cuenta que el tipo de sintonía que se ejecutará dependerá exclusivamente del tipo de experimentación que se ha analizado que poseía datos en su interior para su posterior tratamiento en el diseño del controlador. El accionamiento que recibe la orden de habilitar qué sintonía ejecutar según qué

experimentación se realizó es la que se muestra en color azul en la zona superior de la imagen mencionada anteriormente, en donde, de manera automática, se inhabilitará la posibilidad de diseñar un controlador en el caso de que falte algún parámetro en concreto.

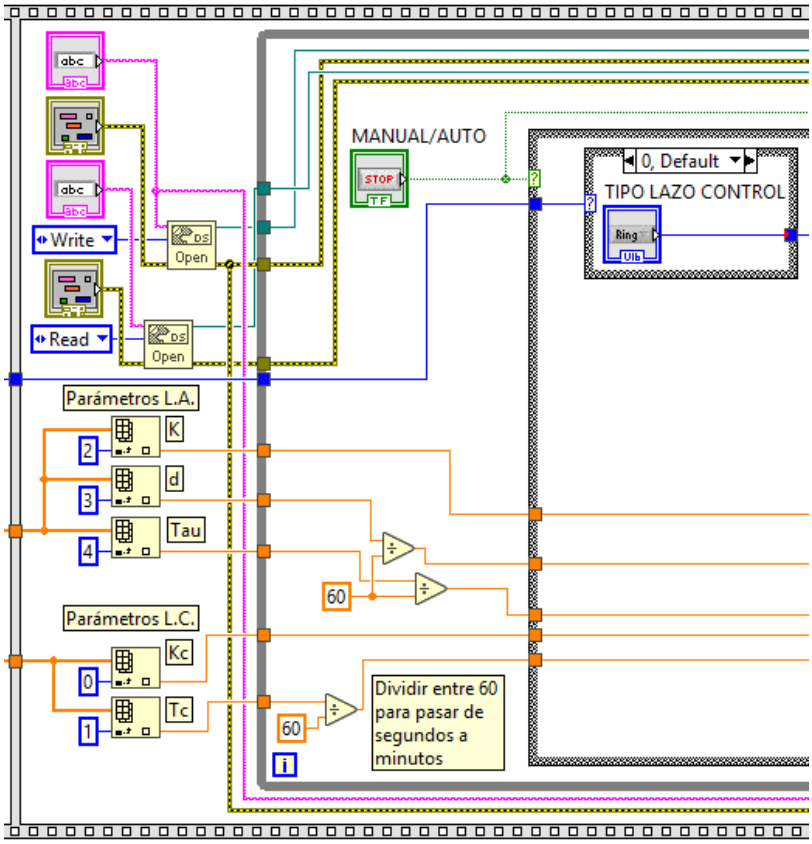


Figura 4.63. Código LabView llegada de los parámetros de ambas experimentaciones y selección del tipo de sintonía.

Otra de las cuestiones que se tiene que tener en cuenta en la imagen anterior es el cambio en las unidades de tiempo para los parámetros que hayan trabajado en segundos; retardo temporal (d) y constante de tiempo (tau) para la experimentación en lazo abierto y período crítico para la experimentación en lazo cerrado, ya que para las respectivas experimentaciones las unidades de tiempo con que se trabajaron son los segundos, mientras que en esta tarea de diseño de control del PID, los respectivos tiempos que se emplearán se tendrán que convertir a minutos, para poder calcular los respectivos tiempos integral y derivativo, que se calcularán en esta subrutina.

Una vez que se tienen todos los parámetros preparados para su posterior utilización en el diseño del controlador, el programa usará el siguiente módulo, diseñado para calcular los respectivos parámetros del controlador PID (Véase figura 4.64.). Entiéndase que el usuario eligió el modo automático para poder

realizar dicho cálculo, de manera que habiendo seleccionado esta opción podrá elegir el tipo de sintonía que se propone en el menú de opciones habilitado correspondientemente.

Si se elige un tipo de sintonía en concreto, el programa provocará la invisibilidad de las otras posibles sintonías que estén disponibles para centrar la atención y cálculo numérico de los parámetros del controlador que esté diseñado bajo ésta. Para contemplar esta técnica, se programó una única variable booleana que accionarán los nodos de visibilidad de todas las restantes sintonías con las que no se estén trabajando en un momento dado.

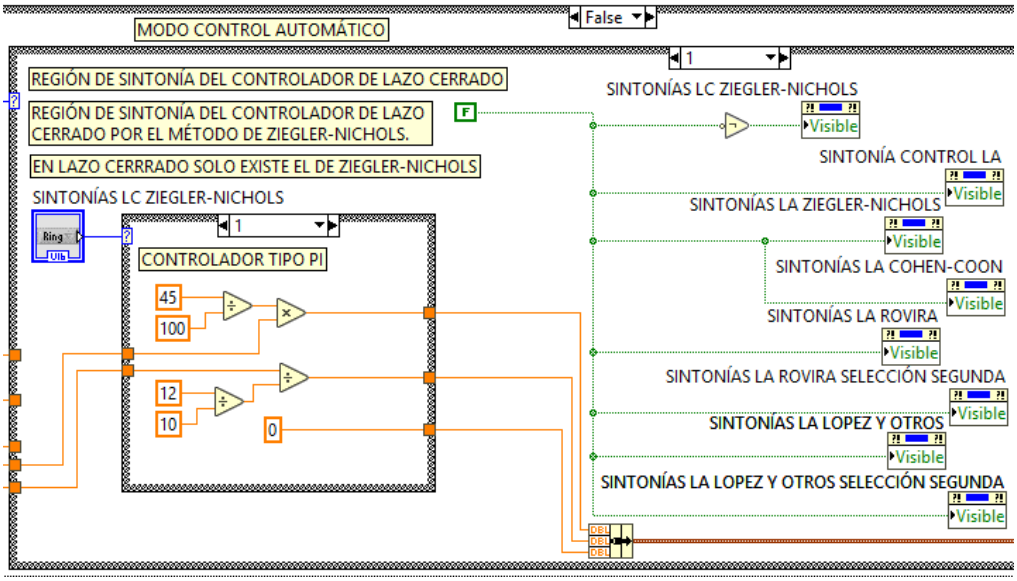


Figura 4.64. Código LabView sintonías de control, cálculo y funcionamiento.

Al calcular los parámetros del controlador PID, tres hilos de transmisión de datos; ganancia proporcional, tiempo integral y tiempo derivativo, serán enviados a un clúster en donde se juntará todo como una unidad para su posterior almacenamiento en el fichero XML, en el caso de que el usuario esté conforme con su decisión. Esta configuración será idéntica para todas las sintonías que se proponen ya sea viniendo de las experimentaciones en lazo abierto o cerrado. Para poder posibilitar la elección que se tome al final se programó empleando una estructura de “case” para cada sintonía propuesta a la que se le asignara un valor numérico proveniente de menú desplegable de opciones.

Llegados a este punto el programador ha de darse cuenta de un detalle esencial para considerar el módulo anterior del cálculo de todos los parámetros de la sintonía seleccionada. En la imagen mostrada a continuación (figura 4.65.) una variable booleana gobernará sobre la decisión que establezca el

usuario acerca del modo manual o automático del diseño del controlador PID, de manera que, si decide establecer un diseño en modo manual, los parámetros comenzarían teniendo por defecto los valores respectivos de: 1,000 para la ganancia proporcional, 0,001 para tiempo integral y 0,000 para el tiempo derivativo. Sin embargo, si decidiera cambiar el valor de esta variable, los parámetros calculados desembocarían en esta estructura "case" que enviará estos valores en dirección al controlador PID.

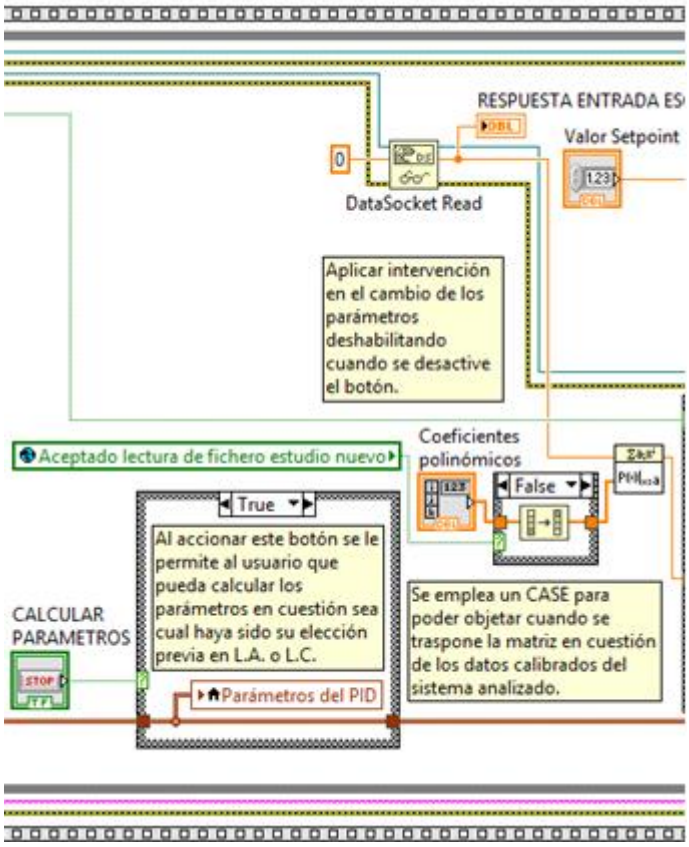


Figura 4.65. Código LabView del cálculo de los parámetros del controlador PID y decisión del tipo de estudio realizado.

En la imagen se puede ver, además, que los coeficientes polinómicos pertenecientes a la calibración anterior se han añadido en esta subrutina distinguiéndose entre estudio antiguo o nuevo al que pertenecieron sus valores calculados. Como se estudió en el programa de la experimentación en lazo cerrado (Véase figura 4.54.), los valores de la calibración se diferenciarán según como lleguen y debido a ello se tendrá que realizar una lectura de información desde el último dato hasta el primero de ellos en la matriz de datos calibrados según provengan de un estudio antiguo o nuevo. Todo ello de empezar a leer la matriz de datos desde la última posición, se debe a que los valores que se extraen del fichero XML en un estudio antiguo salen

posicionados en columna y LabView los lee incorrectamente de cara al uso de dichos valores. Si el estudio es nuevo, la matriz de valores siempre será en fila tal y como se incluyen en la interpolación del bloque que también recibe los datos de entrada. El motivo de que lleguen los valores de la calibración reside en que junto con los parámetros calculados del controlador PID, el usuario podrá ver el efecto causado por los valores numéricos establecidos en su diseño, pudiéndose observar mayor o menor optimización en la salida controlada.

Dentro de lo que supondrá el esquema del controlador PID se tendrá que tener en cuenta los nodos de habilitación para evitar que el usuario, encontrándose en modo manual, accione alguna sintonía propuesta ya sea proveniente de la experimentación en lazo abierto o cerrado. Dichos nodos de habilitación son los que se encuentran marcados en azul con el nombre de cada sintonía propuesta en letras mayúsculas (Véase figura 4.66.):

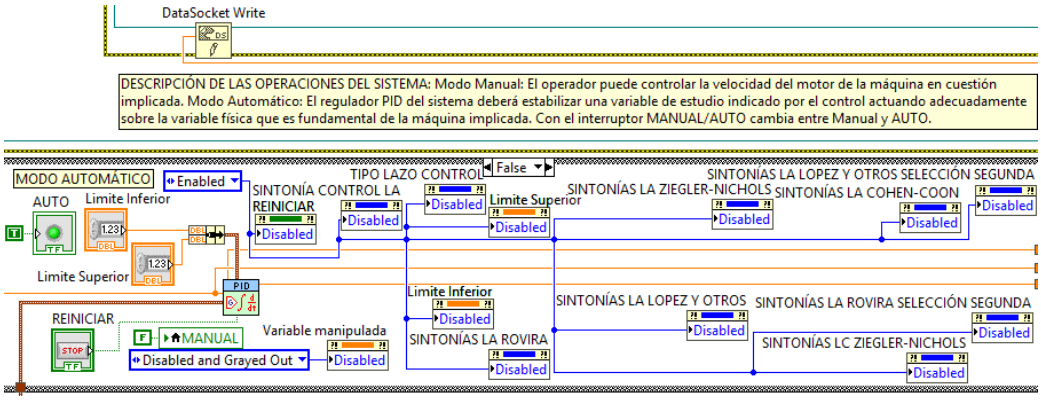


Figura 4.66. Código LabView de habilitación de los nodos de los tipos de sintonía según el modo de activación manual o automático.

Obsérvese que los parámetros que se habían calculado anteriormente desembocan en la parte inferior de la estructura "case" del modo de configuración del controlador ("hilo" granate) que llega hasta el bloque del controlador PID, el cual recibe también los datos de la previa calibración que se había hecho de la planta de procesos en cuestión y de los límites inferior y superior entre los cuales se tendrá graficada la respuesta que proporcione el mismo, de manera automática, tras la sintonización adecuada. El bloque en LabView del controlador PID recibe una señal booleana, que es un accionamiento que se le otorga al usuario para poder reiniciar el proceso de automatización de la planta de procesos que se pretende controlar mediante este software.

Una vez procesados aquellos datos de las experimentaciones y calculados los parámetros en cuestión, junto con los cambios de escalón impuestos, se podrá observar la salida en las diferentes gráficas que se le proporcionarán al usuario para que observe la evolución de la sintonización del controlador PID. Téngase en cuenta que cuando el usuario esté satisfecho de la configuración final del controlador, podrá detener la marcha de la subrutina provocando a parada del funcionamiento de la planta de procesos analizada y realizar un almacenaje posterior de los parámetros calculados del controlador automático convertidos en matriz de datos numéricos en vez del común "cluster" de datos usado anteriormente, hecho que facilita al programa acceder y guardar dichos parámetros en el correspondiente fichero XML (Véase figura 4.67.):

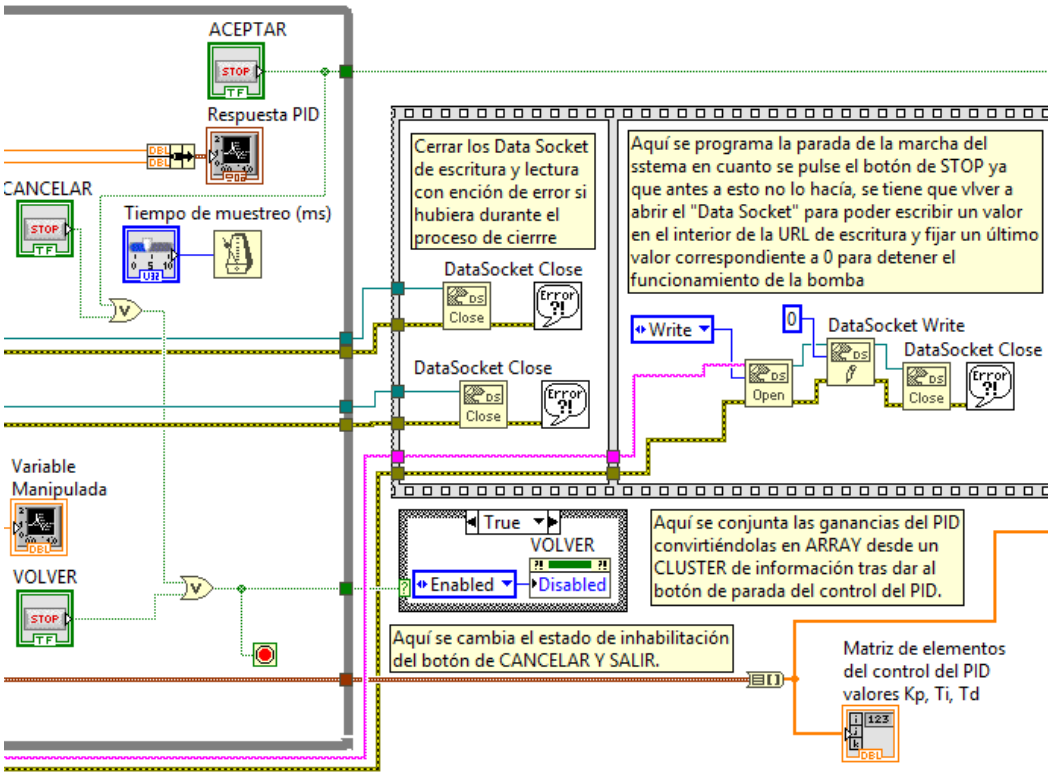


Figura 4.67. Código LabView de la respuesta graficada y almacenaje de los parámetros del controlador PID en el fichero XML.

Finalmente, y como se comentó en anteriores subrutinas, las mismas acaban con el común módulo de almacenaje de los datos generados durante la tarea en vigencia. La similitud de lo que se verá en la siguiente imagen (Ver figura 4.68.) con respecto a lo contemplado en las anteriores subrutinas queda destacada por la técnica empleada para guardar los datos en el fichero XML en forma de matriz de STRINGS que previamente se guardará en una variable global, aquella que corresponda con el estudio que se este ejecutando en esos instantes.

La decisión que se toma acerca del estudio que se haya realizado la lleva a cabo la variable booleana cuya función específica, que ya se había comentado, no era otra que enviar un valor lógico u otro según qué estudio se hizo, debido a que el fichero XML leído en cuestión, tendrá una antigüedad diferente.

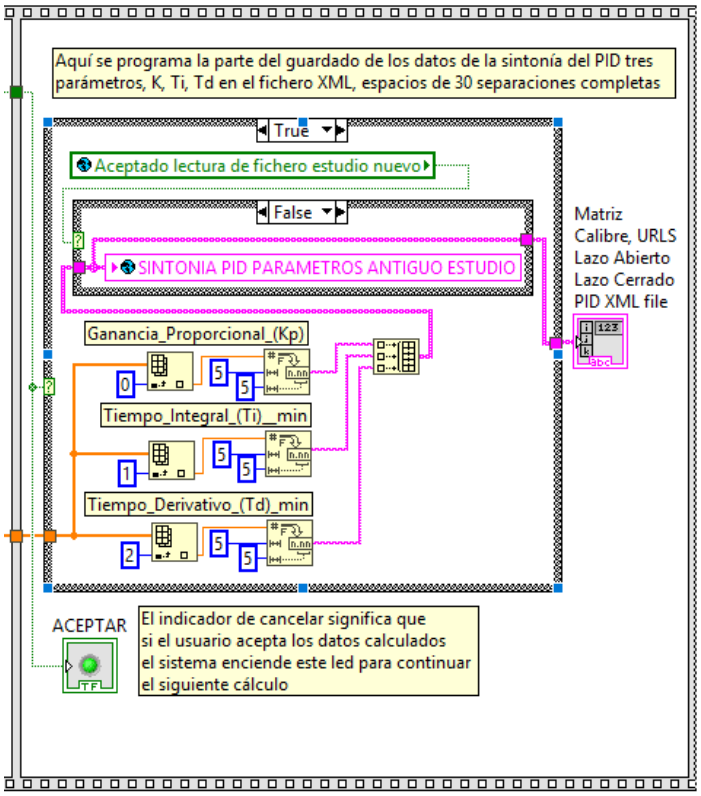


Figura 4.68. Código LabView del almacenaje de los parámetros del controlador PID en el fichero XML.

4.2.1.3.3. PROGRAMA SECUNDARIO ANTIGUO ESTUDIO:

En lo que corresponde el programa secundario, se comentarán los aspectos más importantes del software diseñado, teniendo en cuenta que las subrutinas que se mencionarán a continuación ya se han detallado en el apartado anterior, de manera que en éste únicamente se hará mención para puntualizar ciertas explicaciones que pudieran hacer referencia a tales subrutinas:

- Calibración de los datos.
- Experimentación en lazo abierto
- Experimentación en lazo cerrado.
- Regulación y sintonía PID.

Lo que si se diferencia de la subrutina del programa principal es lo referente a la carga del fichero cuyos datos se leerán para reanudar las tareas que

quedaron pendientes de ejecutar o simplemente reintentar la optimización de los resultados obtenidos de las tareas cuyos datos se han cargado correctamente. La carga del fichero XML que se introducirá previamente, será la primera de las tareas que se ejecutará en esta segunda parte del software, por tanto, debe ser programada lo más eficaz posible evitando que se cometan errores comunes de tipo; reconocimiento del tipo de fichero que se leerá, debiendo ser XML, carga del contenido del mismo y lectura de los datos para poder emplearlos en su momento adecuado evitando que se cree un error de asignación de los datos a una tarea que no corresponda. A continuación, se expondrán en detalle las configuraciones que se han de tener en cuenta para poder programar esta tarea tan fundamental en el funcionamiento de esta segunda parte del software.

4.2.1.3.3.1. INTRODUCIR FICHERO GUARDADO:

Cuando se procede a la carga de los datos de un fichero de tipo XML el programador tendrá en mente aspectos del código que desarrollará como; variable que apunte a un fichero XML que se halle en una dirección en el explorador de Windows, módulo que extraiga los datos del fichero en cuestión pudiendo asignar cada información extraída a la tarea correspondiente que pertenezca y por supuesto poder indicar al usuario en qué tarea dejó su última actualización antes de detener la ejecución del software.

Como siempre, al principio de cada subrutina, lo que se hace es poner en marcha cada una de las variables implicadas en su interior (téngase en cuenta que en la segunda parte del software referido a “continuar con el estudio antiguo” se ha reiniciado todas las variables en su programación junto con las variables globales que se empleen debidamente, este hecho es previo al bloque encargado de cargar los datos de un fichero XML). El motivo de este proceso de reinicio completo es, como se mencionó líneas más arriba, evitar que restos de datos antiguos obsoletos, interfieran en la nueva iteración que pudiera ejecutar algunas de las tareas comunes que se proponen.

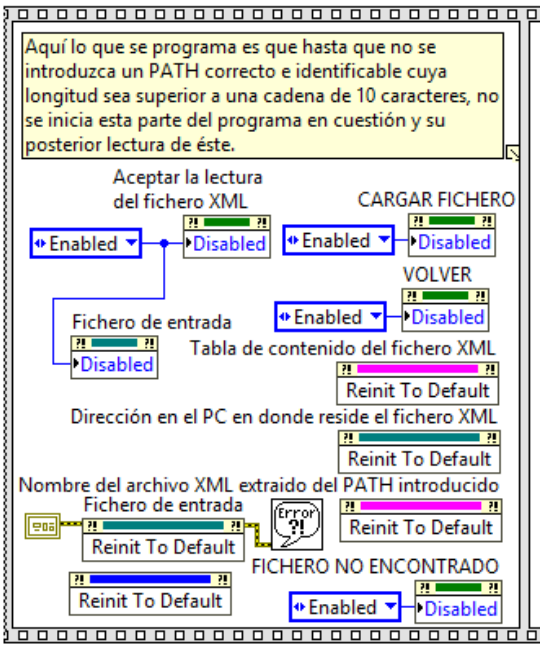


Figura 4.69. Código LabView del reinicio de las variables de la tarea de carga del fichero XML.

De entre las variables que se ponen en marcha, tras ejecutar la subrutina correspondiente con la carga de datos del fichero XML (Véase figura 4.69.), cabe mencionar la correspondiente a aquella que trata la dirección en el explorador de Windows de un ordenador común en donde se guardará el fichero en cuestión. Dicha variable será de tipo PATH en LabView. El resto de las variables tipo STRING permitirá informar al usuario de la información del fichero; nombre y contenido de este tras haberlo cargado. Y, por último, y siendo las variables más comunes en la programación del interfaz del usuario, las variables booleanas que controlan el común juego de accionamientos basado en "Volver" si no se desea continuar con la tarea ejecutada, "Cancelar" si habiendo ejecutado la totalidad de la tarea, no se deseen guardar los datos generados en ella y "Aceptar" en el caso contrario y que se puedan guardar en el posterior fichero XML.

Después de haber reiniciado todas las variables, se programará el código que pueda cargar el fichero XML, verificando primeramente su dirección en el explorador de Windows y posteriormente extraer toda la información respecto al estudio realizado que dicho fichero contiene. Para ello se solicitará al usuario que introduzca una dirección URL del directorio en donde sepa que se hallará el fichero XML. El software comprobará si la dirección es correcta y de serla que corresponda con el tipo de documento que puede ser leído por el programa para su posterior tratamiento (Ver figura 4.70. primera secuencia) En la primera de las secuencias se tiene todo lo que se comentó de manera programada

pudiéndose decidir si tras haber comprobado su veracidad se pretende cargar la información que el fichero posee o regresar al menú anterior para salir del segundo programa. En el caso que la dirección URL no sea reconocida o exista un fallo de identificación de esta, se habilitará un accionamiento que le permita al usuario regresar al menú anterior como opción de salida cuando comprobada la inexistencia de tal fichero se decida realizar uno nuevo o buscar alguno antiguo.

Cuando se haya comprobado que el fichero es correcto y sus datos se hayan reconocido, el usuario podrá decidir si continuar con las posteriores tareas que se le propongan tras dicha lectura o simplemente cancelar la operación dejando sin cargar ningún fichero y regresando al menú anterior sin estar alguna de las tareas habilitadas para su acceso al no haber leído documento alguno. Ello se puede visualizar en la segunda secuencia de la estructura de la imagen mostrada a continuación:

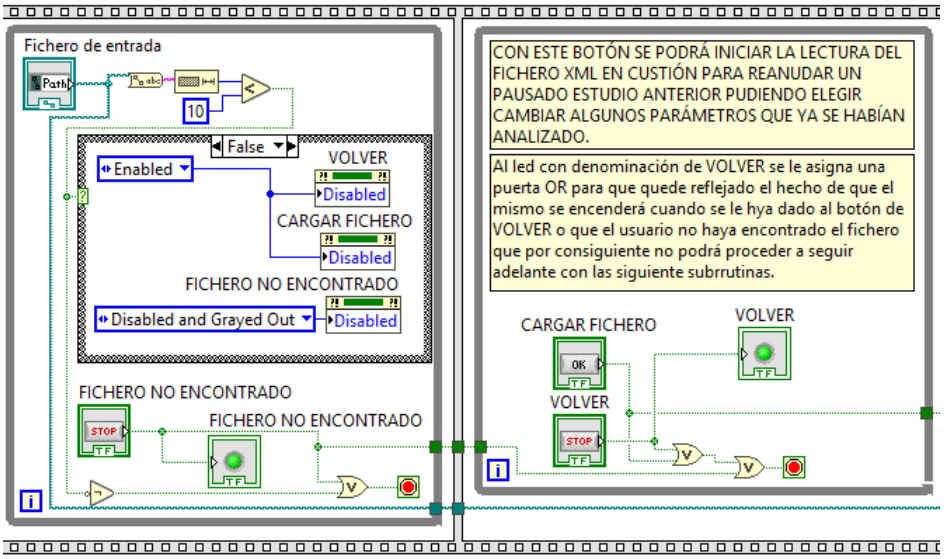


Figura 4.70. Código LabView registro del fichero XML cuya información se pretende cargar y aceptación o cancelación de dicha carga.

Seguidamente se programará la subrutina en donde introduciendo la URL que el usuario haya seleccionado previamente, se podrán extraer los datos que el fichero correspondiente poseía en su interior. El bloque en color verde con denominación “Abrir fichero XML a leer” se encarga de ejecutar todas las tareas correspondientes con el tratamiento del fichero para almacenarlas en posteriores variables globales habilitadas para poder funcionar en varias tareas sucesoras. Téngase en cuenta que el fichero contiene datos de cada una de las subrutinas de control de manera que para su eficiencia en el código se emplearán matrices que se rellenarán según cuántos datos se hayan obtenido

del documento. En la siguiente imagen se puede observar el código explicado (Véase figura 4.71.) en donde, además, cabe mencionar que la URL empleada también se guardará en una variable global por razones carga y almacenamiento al final de cada tarea que se ejecute obteniendo los resultados esperados.

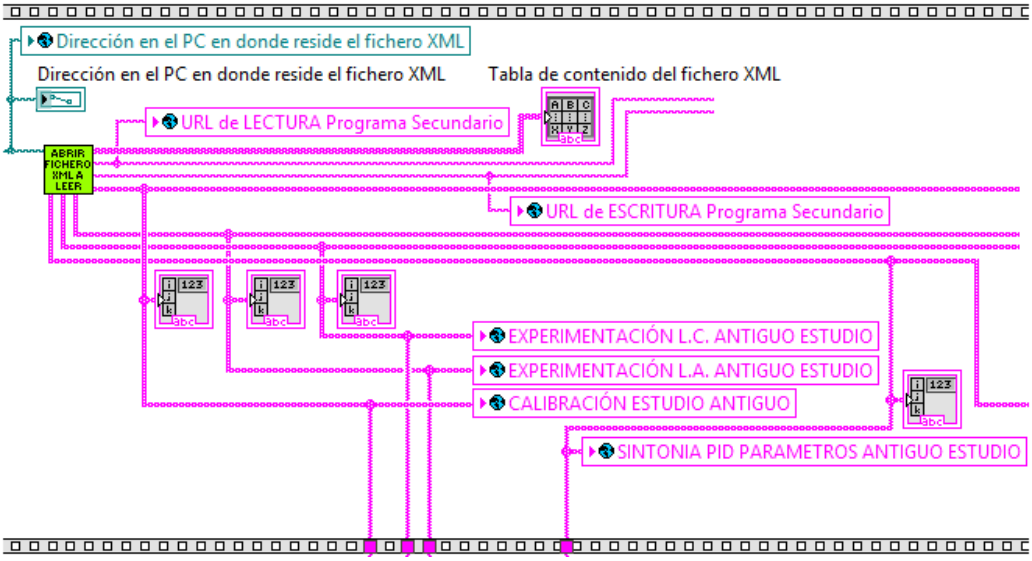


Figura 4.71. Código LabView introducción de la URL, extracción de los datos del interior del fichero XML y almacenaje en variables globales tipo STRING.

A continuación, se propuso verificar la cantidad de datos que se tenían y si eran reconocidos como información que se obtuvo en alguna ocasión en las tareas de las diferentes subrutinas. Para ello se emplea una estructura de tipo “for” en donde se comprueban cada una de las filas que contiene la matriz de STRINGS en referente a datos numéricos reconocidos para su posterior uso. Para poder conseguir programar este filtrado de datos correctos se realiza un conteo de la cadena de caracteres que existe en cada una de las filas de la matriz de datos de cada una de las tareas leídas del fichero XML. En el caso de que se genere un error, el fichero reemplazará el contenido existente no reconocido por una letra “E” de error que al ser contada como un único carácter que es inferior al establecido (dos caracteres) como filtro, el bucle “for” generará un valor booleano de “false” que se almacenará dinámicamente en el hilo de salida del bucle hasta terminar de haber comprobado la veracidad de los restantes datos analizados.

Posteriormente, el programa impondrá un bloque triangular representando una puerta lógica AND en donde el hilo que almacenaba los valores lógicos convergerán en esta puerta cuya salida se verá alterada únicamente por el hecho de haber encontrado un único valor de “false” en el análisis de todas las filas, en cuyo caso se procederá a desestimar la información que se obtuvo de

la tarea a la que corresponda tal fallo en su análisis de los datos comprobados, provocando que la salida que otorgue el programa será que el usuario no ejecutó aquella tarea y únicamente hizo correctamente las tareas precedentes. Como se puede ver en la imagen (figura 4.72.), para cada tarea que propone el software se ejecuta el análisis preliminar antes del reconocimiento de hasta que punto se guardaron correctamente los datos en el fichero XML concluyendo, de cara al usuario, en mostrarle qué subrutinas deberá ejecutar para terminar de diseñar el controlador PID que pretendió realizar antes de verse interrumpido e impedido de llevarlo a cabo.

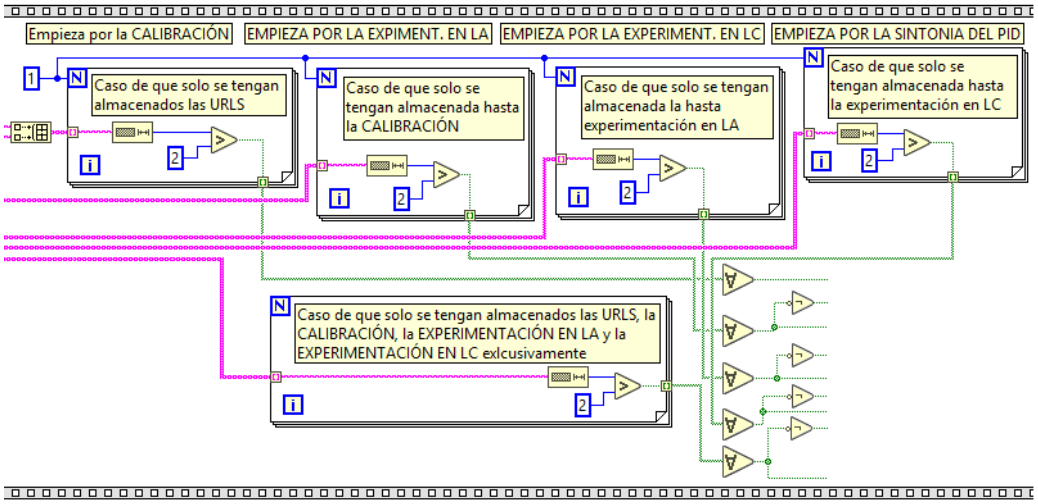


Figura 4.72. Código LabView de verificación de las tareas que han sido ejecutadas con datos numéricos obtenidos, guardados y aquí cargados.

Seguidamente, se programará empleando la lógica booleana para poder converger en el menú de opciones de carácter informativo (mostrado en la imagen de color azul en la zona superior derecha, Véase figura 4.73.) en donde se podrá mostrar al usuario aquellas tareas que se han hecho y de las que se han obtenido los valores que de manera correcta se han leído del fichero XML. Para poder llevar a cabo la comprensión del código que se ha diseñado en esta parte del software, se tiene que tener en cuenta que hay dos zonas de estricta señalización:

- Diseño del controlador PID a través de la experimentación en L.A. (Señalizado en la figura 4.73. zona centro: “PID con L.A.)
- Diseño del controlador PID a través de la experimentación en L.C. (Señalizado en la figura 4.73. zona centro: “PID con L.C.)
- Contabilización de tareas realizadas para definir punto de reanudación actual.

El motivo de que se fragmente la comprensión de esta manera reside en la forma en que se prende hacer entender al usuario qué camino dejó sin terminar y que si su interrupción fue justo durante el diseño del controlador PID sepa con certeza, tras cargar los datos del fichero XML, con qué tipo de experimentación estuvo ejecutando la sintonía del correspondiente PID. Aunque sea cierto que a posteriori, se podrá contemplar esto con mayor precisión, dado el panel de datos que visualizará cerciorándose de este detalle, la intención era que fuera consciente que la carga de datos del fichero fue correcta y éstos son los que generó en una anterior ocasión.

Las dos puertas lógicas que se encuentran en medio de sendos comentarios "PID con LC" y "PID con LA" tendrán la misión de ofrecer al usuario la información acerca de que el controlador que estuvo sintonizando empleó los parámetros de una concreta experimentación que bien podría haber sido de lazo abierto o lazo cerrado.

En la imagen se puede ver como este detalle de confirmar qué tipo de experimentación fue realizada para acceder al diseño del controlador depende de las tareas previas que se han ejecutado (debajo de la etiqueta "PID con LA"), de entre ellas ambas experimentaciones, así el usuario verá que en el caso de no haber accedido a la sintonía del PID puede cerciorarse de hasta qué tarea obtuvo los datos que el fichero XML cargó en esta subrutina de carga.

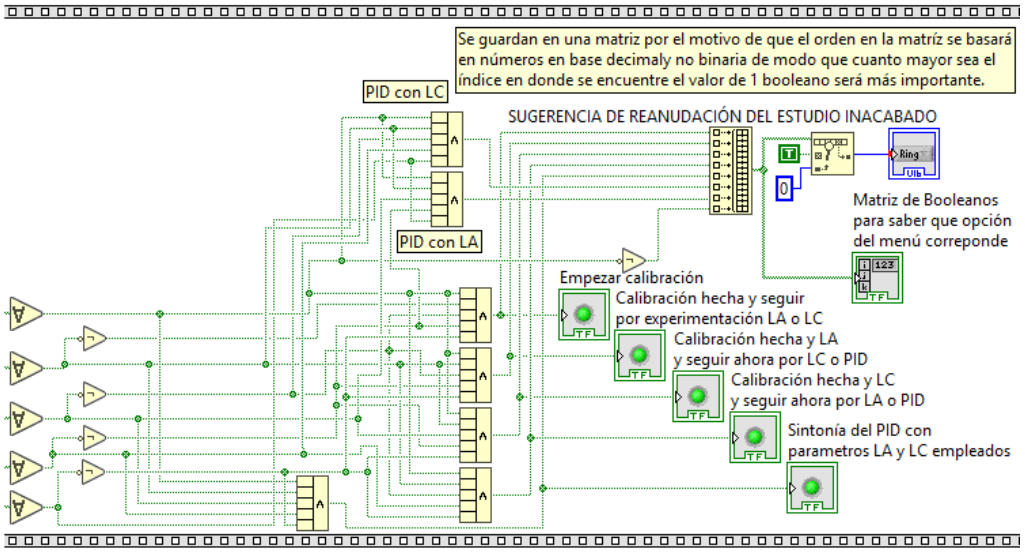


Figura 4.73. Código LabView generación del informe de tareas completadas según los datos leídos del fichero XML introducido para mostrar al usuario.

Solo cuando todos los valores lógicos hayan pasado por las puertas lógicas que se ven en el código se podrá acceder a la matriz de valores lógicos en donde

se podrá dictaminar qué tareas fueron ejecutadas y qué otras quedan pendientes de ser realizadas. La manera en que se decide por donde reanudar la ejecución del programa reside en la comparación de todas las tareas que hayan generado un valor lógico "true" a razón de su realización con aquellas que hayan generado un valor lógico "false" a razón de su no realización. Dado que para cada tarea se le hará llegar a una puerta AND, la comprobación directa de la propia y la negada de las demás (puerta NOT) para saber si una tarea fue realizada basta con que su propio valor lógico sea "true" más el valor de las demás que la suceden, si no fueron realizadas devolverán un valor "true" (cambiado al pasar por la puerta NOT).

Así en el caso de que se compruebe una tarea de las propuestas, y no quede reflejada como hecha, su valor lógico a "false", conjuntamente con las demás sucesoras dando el valor "true", el programa entenderá que no es la última tarea que ejecutó el usuario pudiendo guardar posteriormente los datos que obtuviera de ella, de manera que hará la misma comprobación con la tarea precedente para ver si ésta fue realizada (valor lógico "true") recibiendo de las sucesoras también "true".

En el caso de que haya sido realizada, la puerta AND generará un valor "true" en la matriz denominada "Matriz de booleanos para saber qué opción del menú corresponde" (Véase figura 4.73. zona superior derecha)

En dicha matriz sólo podrá haber un valor lógico "true" siendo las demás "false" permitiendo al programa distinguir con facilidad y justificadamente hasta qué tarea avanzó el usuario, ya que de la matriz booleana se buscará la posición del valor lógico "true" que extrayéndose su índice de posición en la matriz mencionada, se podrá enviar al desplegable de opciones (en color azul en cuyo interior cada opción tendrá asignado un número según el punto en que se dejó interrumpido el transcurso del programa ejecutado) el índice al que pertenecía la colocación del valor "true" que coincide con la frase en el desplegable de opciones informativas que aclara la tarea siguiente que podrá ejecutar y de qué tareas ya se obtuvieron sus datos específicos.

En el último de los módulos programados para poder finalizar con esta tarea de la carga de datos del fichero XML, se diseñó una serie de bloques encargados de la conversión de los datos extraídos del fichero de tipo STRING en los correspondientes de tipo numérico DBL. Pero téngase en cuenta que en el fichero se han incluido los correspondientes comentarios referentes a los datos numéricos que han sido tratados incluyéndose la tarea de la que provino, de manera que se deberá extraer el dato numérico sea cual sea el índice de posición en la matriz correspondiente a una tarea en concreto; comunicación OPC, calibración de datos, experimentación en lazo abierto, experimentación en lazo cerrado o sintonía y regulación PID.

El bloque que permite extraer de cada fila el dato numérico ejecuta un seguimiento para delimitar entre qué caracteres se encuentran los números que considerados de tipo STRING, posteriormente se cambiarán a DBL. En el caso de las URLS de lectura y escritura no hará falta buscar ningún valor numérico en concreto, sin embargo, al encontrarse situadas en una misma matriz, simplemente se extraerán y no se realizará conversión alguna en lo que corresponde al formato de tipo STRING del que procede. Por último, en lo referente al calibrado de datos, el primer dato que se guarda en su correspondiente matriz será el número referente al tipo de calibración que se empleó para la obtención de los resultados guardados. Dicha matriz se deberá fragmentar para no considerar este número y sí, los siguientes para poder convertirlos en formato DBL y trabajar con ellos en las tareas posteriores de experimentación y diseño del controlador (Véase figura 4.74.):

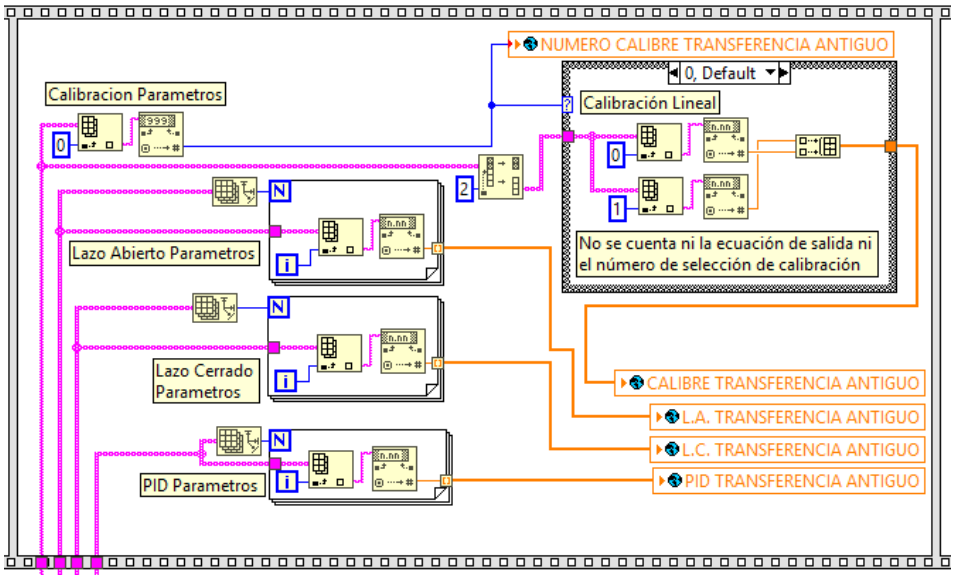


Figura 4.74. Código LabView conversión de datos tipo STRING a DBL numéricos para su utilización posterior.

Después de esta trama se tendrá otra que básicamente se centrará en la aceptación de la carga de los datos del fichero en cuestión. En cuyo caso la subrutina concluirá exponiendo en el panel del programa secundario todos los datos que se cargaron y que ya están disponibles para ser empleados en a posteriori.

4.2.1.3.3.2. NOTAS SOBRE SUBVIS REUTILIZADOS:

Dentro del software que se ha diseñado, existen varias subrutinas han sido reutilizados en dos zonas diferentes (estudio nuevo con un fichero vacío

esperando a ser usado y estudio antiguo con un fichero que ya contiene ciertos datos que se cargarán en el programa) de entre ellos se mencionarán los siguientes:

- **Calibración de datos:** esta subrutina será utilizada en ambas partes descritas del programa principal debido a que, en la primera parte, el usuario ejecutará la calibración de los datos que reciba de la planta de procesos por primera vez. Pero en el segundo de los casos el usuario ya tiene constancia de haber realizado una calibración, aunque no está satisfecho con los resultados que obtuvo de manera que procederá a ejecutar de nuevo esta subrutina para mejorar los resultados.

- **Experimentaciones en lazo abierto (L.A.) y cerrado (L.C.):** igual que la anterior subrutina del primer tratamiento de datos, en esta ocasión se considera una importante influencia en que el usuario reconsidere la opción de mejorar los parámetros obtenidos de cada una de las experimentaciones para optimizar el diseño del posterior controlador PID que se aplicará en la planta de procesos. En esta ocasión ambas experimentaciones también son subrutinas compartidas en ambos estudios por la misma razón de mejorar las pruebas realizadas.

- **Sintonización del controlador PID:** es una subrutina fundamental en el desarrollo del controlador debido a que es el encargado de sintonizarlo según qué experimentación se había realizado exigiendo al usuario que sea exacto en su elección ya que, tras conseguir los correspondientes parámetros, éstos deberán lograr de manera automática que controle una variable del proceso de la planta estudiada, al tratarse de un sistema de tipo SISO.

Ya se comentó que estas tres subrutinas eran copias idénticas en ambos estudios y que cambiar algún aspecto del interfaz o del software del interior implicaría que las modificaciones afectarían a ambos estudios. Ante ello es importante que el programador comprenda la sincronización que se ha de establecer para que se pueda acceder a estas cuatro subrutinas comentadas en función del estudio que se plantee realizar.

4.2.1.3.4. PROGRAMA ENVIAR POR CORREO O GUARDAR EN INTERNET:

Esta es una subrutina que no afecta en ningún instante a las restantes y podría decirse que es una sección ajena al diseño de controladores PID que son motivo principal del presente proyecto, sin embargo, es un código programado

para facilitar al usuario la decisión de poder guardar el informe de los datos en alguna zona concreta de internet abriéndose al mismo tiempo la carpeta en el explorador de Windows en donde se encuentra guardado el propio fichero XML. Como se puede ver en la siguiente imagen (Véase figura 4.75.) cada uno de los accionamientos booleanos porta una página web en donde el usuario podrá guardar o enviar el fichero mencionado, se encuentre este completado o no. Se proponen varias opciones de páginas web además del correspondiente botón de regreso al menú principal en el caso de que el usuario deseara no seguir con esta parte del programa. La estructura es sencilla más cualquier de las variables booleanas de la estructura de la siguiente imagen podrá terminar con el bucle "do while":

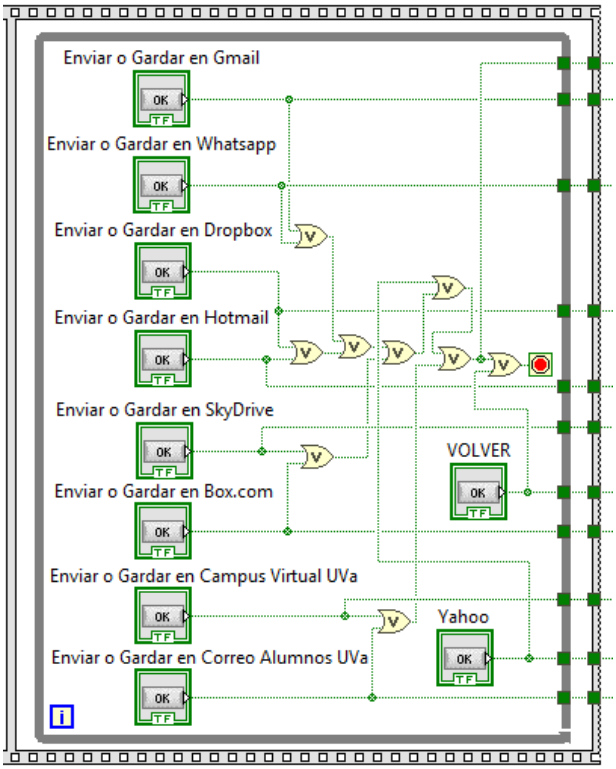


Figura 4.75. Código LabView selección de la página web en donde se almacenará o enviará el fichero XML.

Lo que se ha programado a continuación es la apertura del correspondiente directorio en donde se ha guardado el fichero XML con el que se ha trabajado, así cuando el usuario accione uno de los botones de una página web en concreto, se abrirá en segundo plano la ventana que abra su dirección en el ordenador. Ante lo mencionado se tiene que tener en cuenta un ligero detalle en el acceso a esta subrutina ya que ningún usuario podrá acceder a la misma sin antes haber creado un fichero XML (estudio nuevo) o cargado alguno que ya existía (estudio antiguo). Esta condición esta sujeta a que, de no imponerse,

el programa jamás podría saber a qué fichero se refiere al usuario cuando tiene la intención de almacenar sus progresos en internet o quiera enviárselo a alguien en particular. Otras de las cuestiones que cabe mencionar son las dos variables booleanas que se crearon al principio del programa principal con la finalidad de poder distinguir entre si el estudio que se esta ejecutando en un instante dado es nuevo o antiguo, evitando con ello que pudiera haber errores de direccionamiento del fichero XML en el explorador de Windows. Las estructuras de tipo "case" que se pueden ver en la ilustración que se adjunta a continuación (figura 4.76.) son idénticas salvo por la referencia a la URL correspondiente al nuevo o antiguo estudio, de manera que durante la ejecución del software únicamente podrá estar operativo uno de ambos bloques impidiendo que pudiera existir la interferencia recíproca en el instante en que se accione uno de ellos. El código parece fragmentado según lo que se puede ver en la imagen, pero va seguido para que pudiera caber con mejor detalle como apoyo en las presentes explicaciones argumentativas.

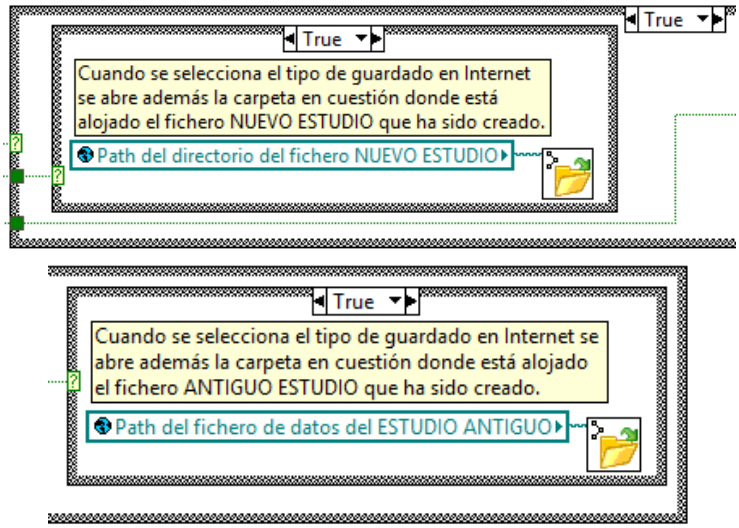


Figura 4.76. Código LabView de apertura del directorio del explorador de Windows en donde se aloja el fichero XML.

Por otra parte, en lo que se refiere a la llamada a la página web que ejecuta LabView se puede observar que lo realiza el bloque que aparece en la siguiente imagen (figura 4.77.), aquel que posee el icono de un folder amarillo sobre una alusión a URL en azul. El funcionamiento del bloque mencionado proporciona al usuario la dirección web directa que guardada en forma de STRING le llega tras ser ejecutada en un momento dado. Es importante que el programador entienda que la URL de la dirección web debe ser clara y concisa de cara a la apertura en internet de su contenido exacto sin que con ello conduzca a una ruta errada que no genere ningún lugar web esperado. Como el programador

ha podido ver en la imagen, se proponen diversas páginas web en donde el usuario podrá almacenar o enviar el fichero XML que haya editado en este software, todas las páginas web se copian y pegan en la variable de tipo STRING que se podrá actualizar o eliminar, siempre y cuando se haya cerrado la correspondiente página web que la respectiva URL sugiere o se haya modificado algún parámetro en el código de apertura de ésta. En la interfaz de usuario se proponen hasta nueve páginas web reconocidas en internet como zonas privadas de almacenamiento y envío, siendo una de ellas la página oficial de la universidad, presentada para alumnos o profesores que hayan utilizado este software.

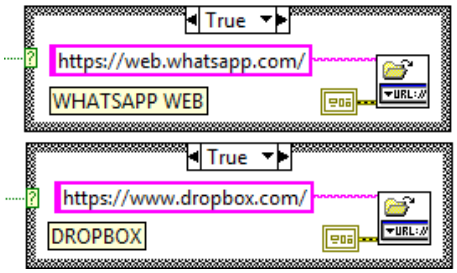


Figura 4.77. Código LabView de apertura de la página web en donde se almacenará o enviará el fichero XML.

Al igual que en todas las subrutinas anteriores, ésta posee de igual manera, una estructura “case” que podrá permitir al usuario elegir entre proseguir con el envío o almacenaje de un fichero XML o cancelar la operación saliendo de esta tarea y regresando al menú principal sin haber ejecutado ninguna clase de funcionalidad que se haya descrito en este apartado.

4.2.1.3.5. SUBVI DE ALMACENAMIENTO Y CARGA DE DATOS:

En esta sección apartada del conjunto de las demás, se argumentará con detalle aquel bloque que permita al usuario guardar los datos generados en el fichero XML y al programador comprender la dinámica de la que se está exponiendo. Como se comentó en el apartado del detalle de calibración de datos, aquí se argumentarán los principios fundamentales de los siguientes bloques:

- Bloque de almacenaje de datos generados en el fichero XML. Es una subrutina cuyo bloque es de color naranja, en el caso del programa de realización de un nuevo estudio y es una subrutina con un bloque de color verde para el caso de realización de un estudio antiguo.
- Bloque de actualización del panel de datos guardados en el fichero XML de las tareas ejecutadas en tiempo real. Es una subrutina cuyo

bloque está diseñado de un color morado para ambos estudios programados.

- Bloque de la gestión de carga o almacenamiento de los datos que se actualizarán en el fichero XML. Es una subrutina cuyo bloque es de color rojo claro.

Todos estos bloques que se han mencionado se expondrán a continuación con mayor detalle para poder comprender con eficacia el motivo de su diseño y su funcionalidad con el resto de las subrutinas del software. Para poder realizar las pertinentes consideraciones argumentativas respecto a estos tres bloques se utilizará el ejemplo de código programado para el calibrado de los datos de entrada:

4.2.1.3.5.1. BLOQUE DE ALMACENAJE DE DATOS:

Es un bloque que se encarga del almacenaje de los datos que se han obtenido en una tarea concreta. En la siguiente imagen (figura 5.51.) se puede observar que existen dos tipos, aunque ambos poseen las mismas funcionalidades más únicamente se distinguen en que el bloque superior (en color naranja) se emplea en la subrutina de la ejecución de un nuevo estudio, mientras que el inferior (en color verde) hace referencia a un antiguo estudio, en donde se cargarán los datos que habían sido guardados en un previo fichero XML.

Figura 4.78. Código LabView del bloque de almacenaje de datos en estudio nuevo y antiguo

En el interior de estos bloques se puede observar el código en LabView programado acerca del cometido que tienen en este proyecto. En la ilustración que se muestra a continuación (figura 4.79.) se puede ver el interior que poseen ambas subrutinas en referente al procesamiento del fichero XML en cuestión, con respecto a los datos que le llegan de la tarea que se concluyó previamente. La estructura que se halla en su interior es de tipo secuencial, en donde se alojan dos bloque idénticos pero con funcionalidades diferentes, esto es debido a que el diseño del bloque en cuestión se destinó para que se pudiera ejecutar la acción de carga de datos del fichero XML o el almacenamiento de éstos mismos, siendo controladas tales acciones mediante una variable booleana que dictaminará que acción se ejecutará en su interior (el cual se detallará más adelante) Entiéndase que la idea reside en que, para poder

guardar los datos generados en una tarea concreta, el programa cargará previamente la información referente a las URLS que se generaron al principio del todo; en la ejecución de la subrutina de la creación del fichero. Ello junto con la fecha y otros datos de interés, se guardarán dinámicamente para que cuando el usuario decida guardar sus progresos, éstos queden almacenados en las variables globales que se tienen en la imagen para poder utilizarlos en cualquiera de las subrutinas que tengan la autoridad de hacer uso de ellos. La cantidad de variables globales que aparecen en la imagen corresponden con las tareas que aún están pendientes de ser ejecutadas, no obstante, si la información se ha cargado del fichero directamente, no hará falta considerar una variable global que guarde el contenido de la misma.

Siempre que se deseen guardar los datos en un fichero XML, el procedimiento para lograr ello es realizar la conexión booleana en uno de sus terminales, siendo un "false" cuando se trate de cargar la información requerida. Pero si la entrada asignada al bloque fuera "true", el mismo cambiaría su forma de proceder habilitándose únicamente aquel en el que estuviera basado en el almacenaje de los datos generados de alguna de las tareas precedentes.

Aparte de todo ello se tiene en la trama inferior (figura 4.79) una tabla de datos "string" que no verá el usuario y en donde quedarán reflejados los datos cargados en tiempo real. El programador contemplará las operaciones de carga de los datos del fichero actualizado y los datos correspondientes a lo que portan las variables globales implicadas (en este caso existen tres variables globales que cargarán los datos en la tabla, véase figura 4.79):

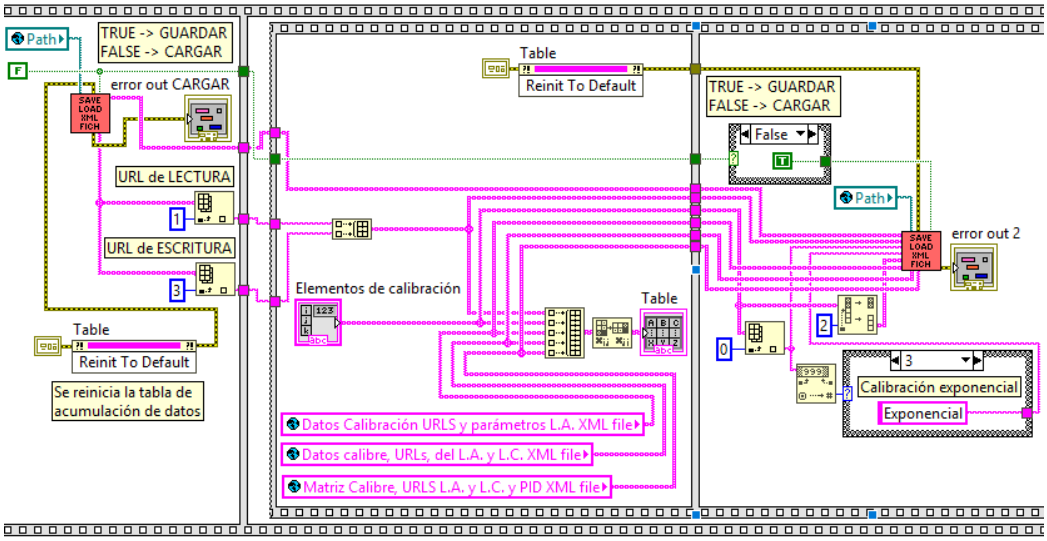


Figura 4.79. Código LabView del interior bloque almacenaje datos del estudio nuevo o antiguo.

4.2.1.3.5.2. BLOQUE DE ACTUALIZACIÓN DE DATOS:

Este bloque será fundamental en lo referente a la comprobación de los datos obtenidos en cada tarea que el usuario podrá visualizar en tiempo real en un panel que se le proporcionará para ver sus progresos. Dicho panel será rellenado por este bloque en cada iteración que se ejecute de cualquier tarea salvo cuando el usuario accione el botón de cancelar, momento en que no se actualizarán los datos por la insatisfacción de los resultados obtenidos que tampoco se guardarán en el fichero XML. En resumidas cuentas, lo que este bloque muestre por el panel habilitado para el usuario será lo mismo que se guardará en el fichero XML asignado para almacenar los datos generados. El icono que trate este código diseñado será el mostrado a continuación (figura 4.80.):

Figura 4.80. Código LabView del bloque de actualización de datos generados por las tareas de control en tiempo real.

El esquema que permitirá mostrar los datos generados es el mostrado a continuación (figura 4.81.) en donde se podrán ver todas las variables globales que se han creado para el estudio correspondiente (nuevo o antiguo según cual haya sido el seleccionado previamente). Todos los datos obtenidos estarán reconocidos por un título adjunto (en forma de variable STRING constante) que los agrupe según la tarea de la cual resultaron.

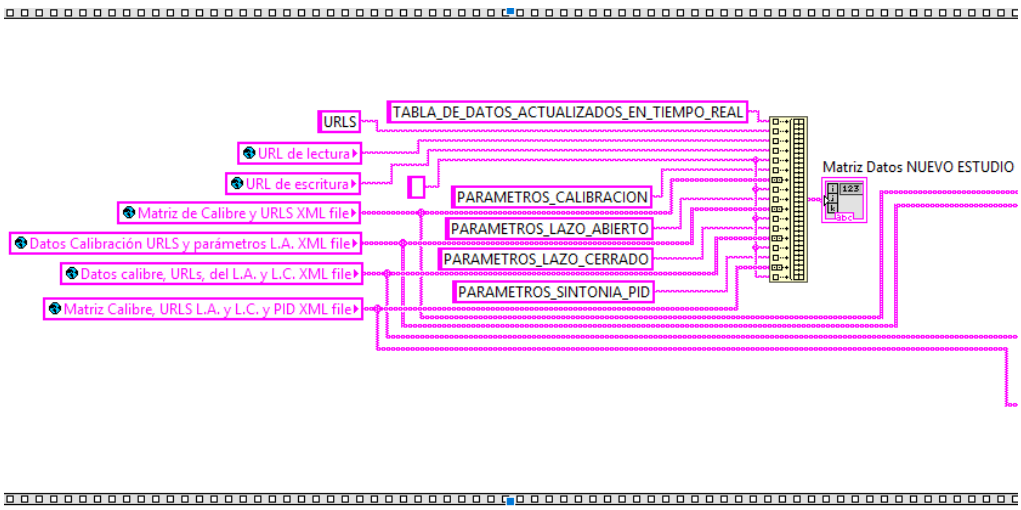


Figura 4.81. Código LabView del interior del bloque de actualización de datos generados por las tareas de control en tiempo real.

Para poder colocar los datos como si de un pequeño informe se tratara, se empleó un vector de datos de tipo STRING expandido en vertical para que el usuario vea cada uno de los datos según a qué tarea hayan pertenecido pudiendo tomar la decisión de continuar con las tareas cuyos datos aún son inexistentes o volver a calcular los datos de la última tarea ejecutada por disconformidad de los resultados anteriores.

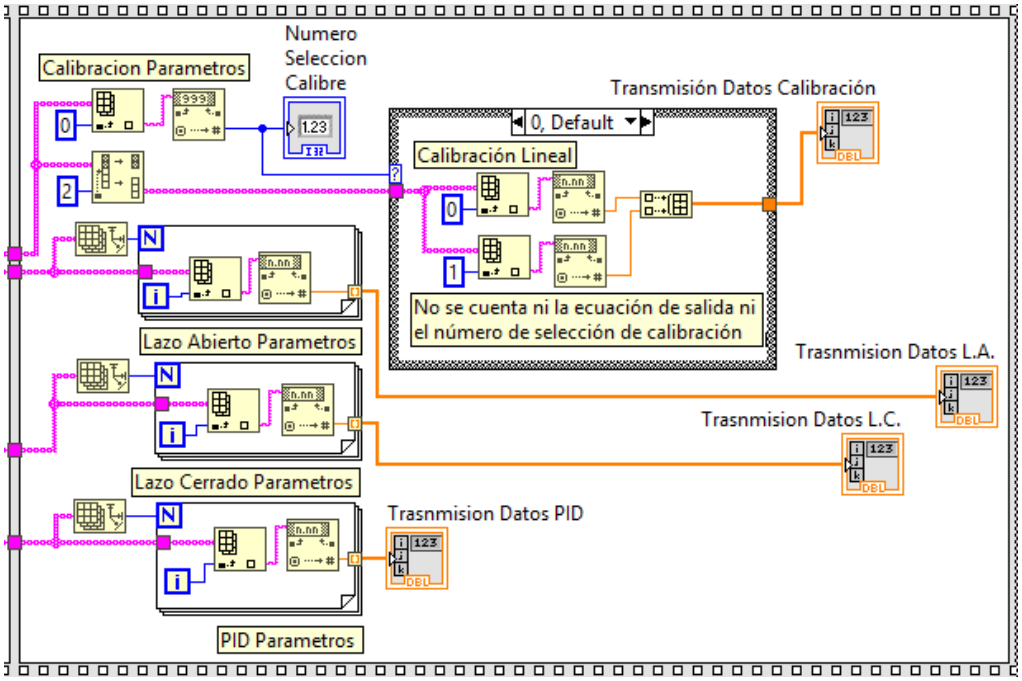


Figura 4.82. Código LabView del interior del bloque de actualización de datos generados en la zona de conversión de datos STRING a DBL

Seguidamente, en el mismo bloque, se procederá a realizar la conversión de los datos de tipo STRING a DBL, correspondiente a los numéricos con la finalidad de ser utilizados en posteriores tareas. Téngase en cuenta que al ser un bloque (un SubVI) se tuvo que incluir como conexión adicional, la correspondiente a los datos que en el panel informativo se expondrían ya que en donde desemboca dicha conexión es en el diagrama de programación del estudio al que se refiere (nuevo o antiguo) en donde con el uso de una matriz de STRING se verá cumplido el cometido de mostrarle al usuario los datos generados en tiempo real.

En lo que se refiere el exterior de este bloque, se mencionó la transferencia de los datos a una tabla (vector de datos tipo STRING) y aquí hay que mencionar que si se pretende que los datos se muestren en tiempo real, dicha tabla tendrá que tener carácter de variable local que siempre apuntará a su variable de la que depende en sí, esto es, todas las tareas que generen información relevante al estudio, sea nuevo o antiguo, tendrán en su diagrama de salida de este

bloque una variable local de tipo STRING, excepto la primera de las tareas que poseerá la misma variable pero sin ser local, la cual permitirá la existencia de las demás variables con dependencia en ésta. Esta técnica permitirá actualizar el panel de información que se modifique con cada conformidad en los resultados de las diversas tareas propuestas.

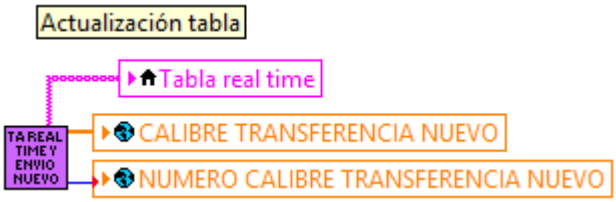


Figura 4.83. Código LabView de la transferencia de datos a una tabla de STRING mostrada al usuario como variable local.

4.2.1.3.5.3. BLOQUE DE GESTIÓN DE CARGA/GUARDA:

- **Código LabView del almacenaje de datos en el fichero XML:**

Este bloque es dual referente a su funcionalidad de poder guardar o cargar los datos de un fichero XML según el valor de una variable booleana. Hay que distinguir este bloque con aquellos argumentados líneas más arriba (uno para cada estudio), que poseen a éste en el diagrama de su funcionamiento debido a que, mientras los anteriores contenían a éste y parte del código encargado de gestionar qué variables globales guardarán qué datos de qué tareas realizadas, el presente bloque se encargará de, una vez gestionado lo anterior, organizar la interfaz del fichero XML en donde, por apartados, se irán guardando todos los datos recibidos o en su caso, cargarlos según cómo hayan sido estructurados, previamente, en la colocación interna del fichero.

Debido a la estructura del lenguaje marcado XML, éste facilitará el agrupamiento de los datos por marcas o apartados para que una vez completado, cualquier usuario pueda realizar una correcta lectura del fichero observando las ejecuciones que se realizaron con este software y aquellas en las que se necesite una mejora de los resultados obtenidos. El bloque en cuestión es el mostrado en la siguiente imagen (figura 4.84.)

Figura 4.84. Código LabView del bloque de gestión de carga/guarda de los datos generados en el fichero XML.

En cuyo interior se mostrarán los diferentes aspectos que posibilitaron el correcto funcionamiento descrito:

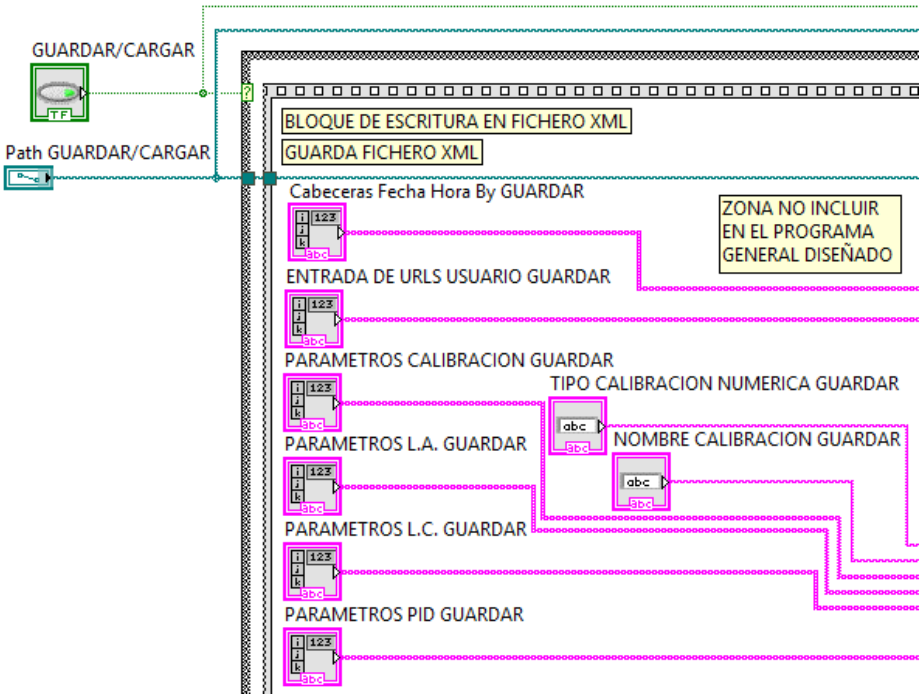


Figura 4.85. Código LabView del interior del bloque de gestión de carga/guarda zona de recepción de los datos y elección carga/almacena

Como se puede observar en la imagen anterior, (figura 4.85.) el interior del bloque comienza con la llegada de la URL en donde se encuentra el fichero XML creado en el explorador de Windows junto con la variable booleana encargada de decidir qué acción ejecutar de las dos que se proponen; carga de datos del fichero o almacenaje. La decisión únicamente se rige por el valor lógico 1; para guardar en el fichero, 0; para cargar datos del fichero. Dicho valor no lo seleccionará el usuario directamente, sino que estará definida por su decisión en lo que respecta a cargar datos del fichero XML (momento en que desea reanudar un estudio que se dejó pendiente de finalizar) o guardar los datos tras haber accionado el botón de "aceptar" en alguna de las tareas que se proponen en el software. Así, según cual haya sido su decisión el bloque ejecutará una subrutina u otra. En el caso del almacenaje de los datos, se puede ver en la anterior imagen las matrices de llegada de los datos, procedentes de sus correspondientes variables globales que no aparecen en este diagrama, pero si en el precedente argumentado. Tras la llegada de todos los datos con contenido definido el siguiente paso es la organización de los diferentes apartados que tendrá el fichero XML en donde se pretenderá guardar uno a uno los resultados obtenidos. El diagrama programado será el mostrado en la siguiente imagen (figura 4.86.):

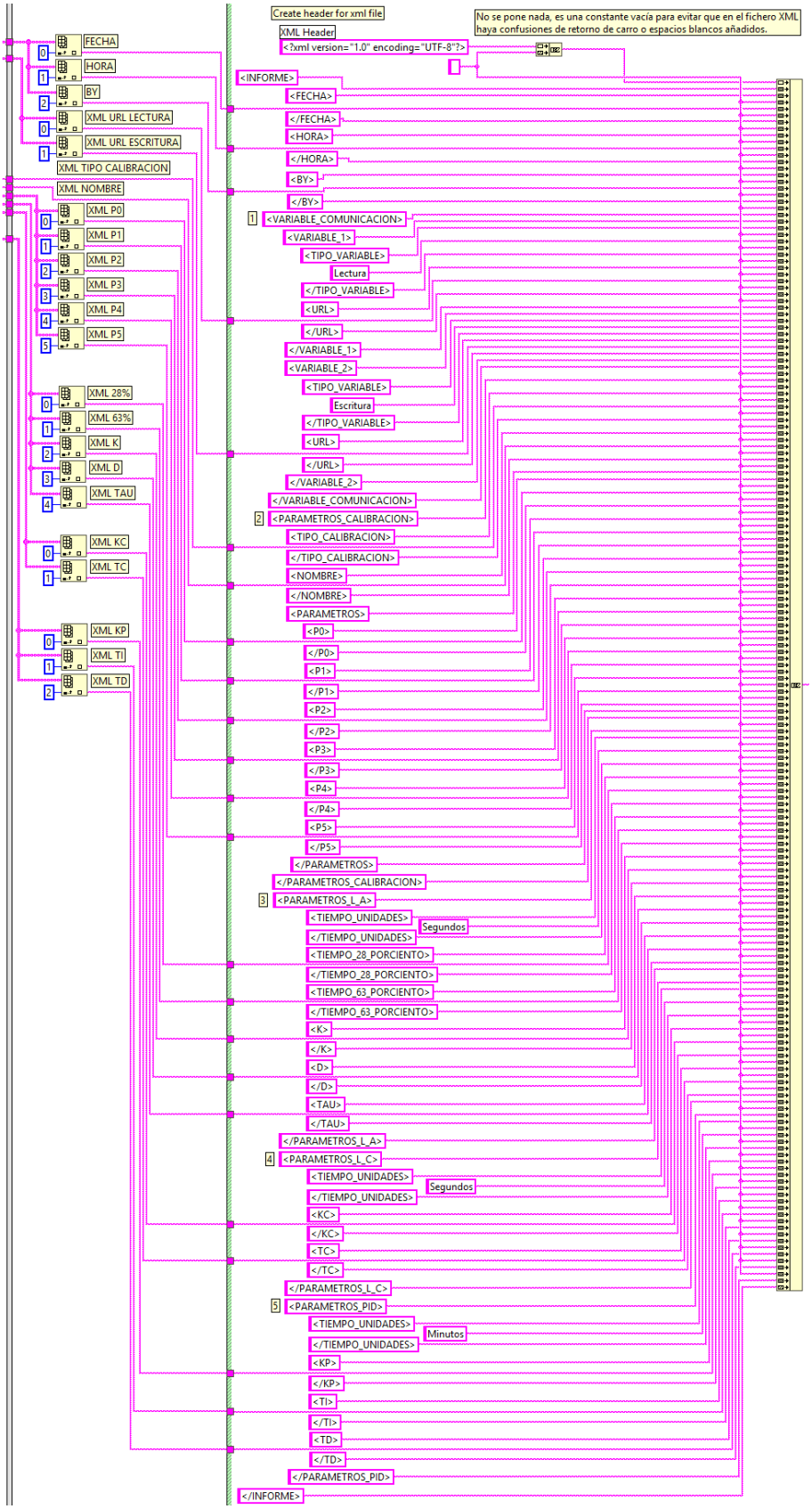


Figura 4.86. Código LabView organización datos fichero XML.

En donde se podrá ver la configuración que se ha elegido para poder facilitar la comprensión del programador y siendo la mejor opción para que se pueda concebir como estructura legible XML. El diagrama es bastante amplio dada la categoría de los datos que se irán guardando a medida que se vayan completando las tareas del programa.

Como puede observarse se han empleado los diversos bloques programados que ofrece la plataforma de LabView para la construcción del diagrama, entre ellos el más notorio es la que permite conglomerar aquella matriz de datos en forma de STRING que posteriormente se preparará para poder guardarlo en el fichero XML sin que surjan errores en la organización de los datos almacenados junto con los títulos adjuntos descriptivos.

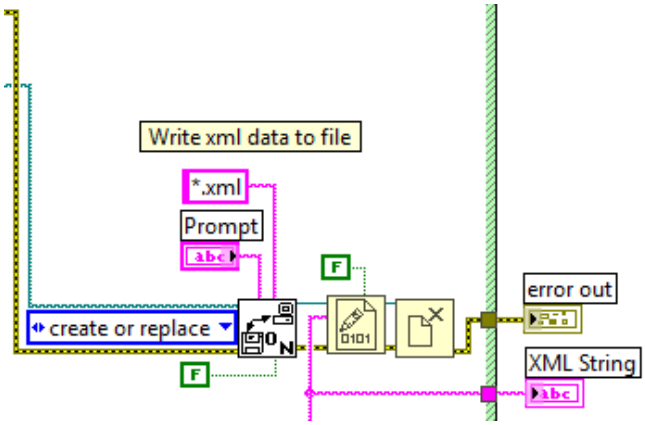


Figura 4.87. Código LabView de escritura en el fichero XML del diagrama de organización de los datos.

En la imagen anterior (figura 4.87.) se trabaja con el fichero XML en donde se le hará llegar el compendio de datos y epígrafes unidos en la matriz de STRING. Además de ello, se editará el tipo de fichero que se va a rellenar con los datos actualizados siendo ".xml" el formato que se le otorga para que sea legible cuando se haya terminado de editar concluyendo la correspondiente ejecución de alguno de los estudios analizados.

- **Código LabView de la carga de datos en el fichero XML:**

En lo que corresponde a la parte del código que carga los datos del fichero XML se han tenido en cuenta que el programa deberá leer en formato XML para conseguir extraer los datos que se pretenden utilizar a posteriori. De manera que para poder leer lo guardado se deberá diseñar una subrutina que lea por epígrafes o marcas que se hayan fijado en el fichero XML, así como puede verse

en la siguiente ilustración se han definido los diferentes apartados que se sugieren en el documento para poder organizar los datos a lo largo de éste:

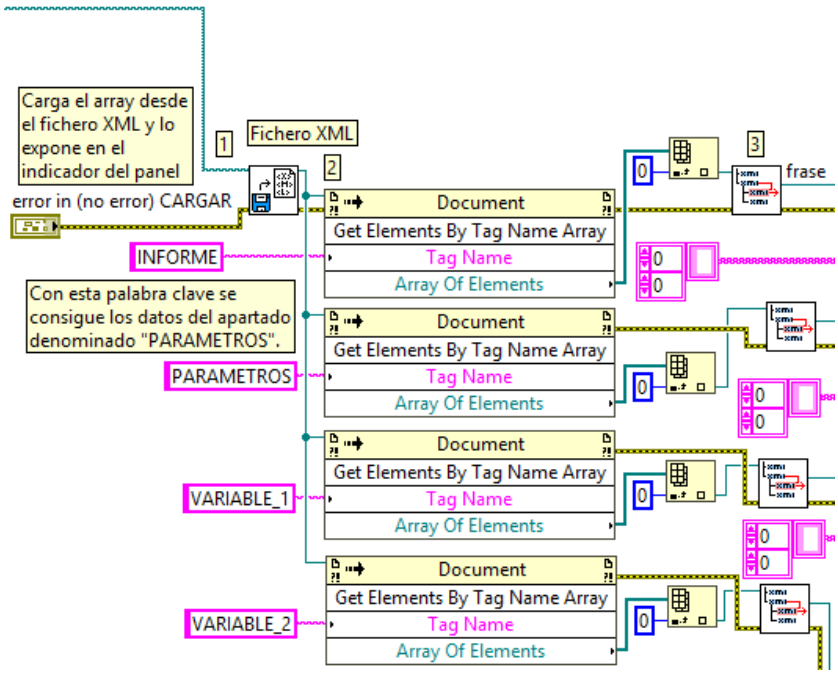


Figura 4.88. Código LabView de lectura de datos del fichero XML en función de las marcas asignadas en él.

Como se puede observar cuatro nodos cargarán los datos del fichero XML en referencia a los epígrafes y subepígrafes que pueda haber en el documento, en el cual sólo en lo que respecta a los datos de las URLS existe ese subapartado referente a asignación de la URL (lectura o escritura) y su correspondiente contenido. Sólo en lo mencionado anteriormente existe esa organización de subapartados más en el resto del fichero únicamente se expone el título o marca correspondiente a una tarea concreta y su respectivo contenido resultante. Como se puede ver en la imagen anterior (figura 4.88.) los bloques blancos que aparecen cerca del margen derecho ejecutan lo que se comentó líneas más arriba y lo que el nodo que les precede ordena de cara a la localización de un dato o varios de ellos a lo largo del fichero creado y actualizado.

El punto siguiente es la extracción de todos los datos que se hayan guardado en el fichero XML, teniendo en cuenta que la operación podrá ejemplificarse básicamente con un simple puntero que recorrerá un vector a lo largo de éste extrayendo en cada iteración lo que cada índice del mismo posea para mostrarlo o utilizarlo en posteriores operaciones, la forma de recorrer tal vector lo ejecutaría una estructura de tipo “for”. Volviendo al caso de este proyecto

con la lectura del fichero, lo que se recorrerá será un documento de texto con marcas que describen puntos donde se han guardado determinados datos y para recorrerlo se empleará un bucle de tipo “do while” finalizando éste cuando el fichero ya haya sido recorrido por un puntero que extrae los datos según lo recorre, cuyo puntero es el bloque de extracción para ficheros XML que en la imagen siguiente (figura 4.89.) aparece con fondo de color blanco y letras “ABC” en rojo entre dos marcas “<XML>” y con una flecha aludiendo la salida de un dato concreto.

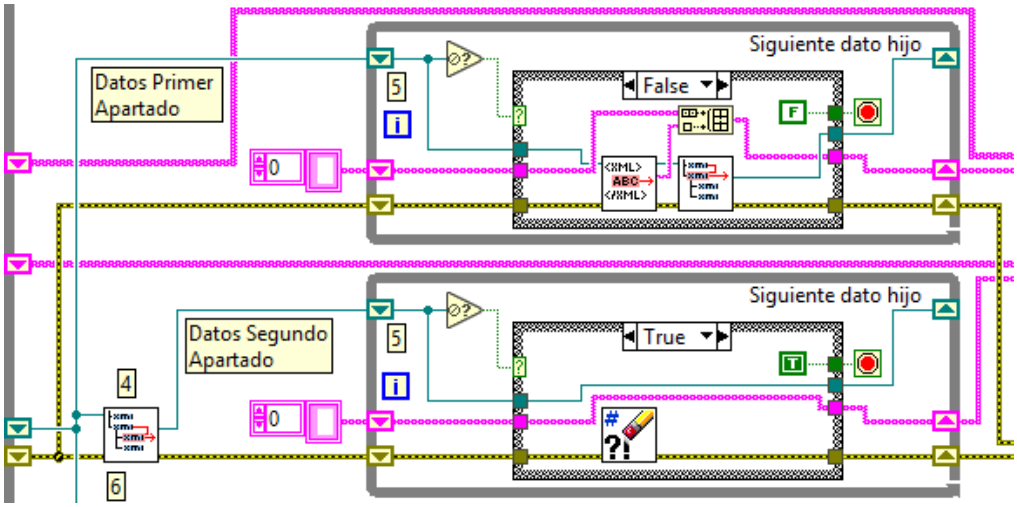


Figura 4.89. Código LabView extracción de los datos de cada marca descriptiva del fichero XML.

Únicamente, estas dos estructuras “do while” permiten ejecutar lo mencionado y están distribuidas como una unidad para cada marca alojada en el fichero XML, de manera que habrá tantas como epígrafes que se hayan establecido en el documento, todas ellas dentro de una estructura mayor también de tipo “do while” que se encarga de recorrer todo el fichero.

Cuando todos los datos hayan sido extraídos correctamente, el último proceso de esta subrutina será colocarlos en matrices de tipo STRING para su posterior conversión y utilización. Hay que contemplar el hecho de que se ha considerado el “hilo” amarillo en la extracción, encargado de gestionar los errores que se puedan producir durante la extracción de toda la información disponible en el documento. Se preverán fallos de tipo, impedimento de lectura por un puntero perdido o falta de sincronización entre varios procesos que pretenden leer del fichero al mismo tiempo que almacenarlo, de manera estas conexiones (Véase figura 4.90.) describirán el error o errores que hayan podido resultar de esta operación, permitiendo al programador localizar la falla con más eficacia que intentar recorrer todo el código diseñado en busca de la zona donde se produjo el fallo. Es importante que al emplear las matrices de tipo STRING se sepa cómo

ejecuta la extracción el programa de cara a evitar que, aun no habiendo errores de lectura, los datos se coloquen en matrices incorrectas, de manera que se intuirá primeramente una lectura convencional basada seguir un camino de izquierda a derecha y desde los datos colocados en la zona superior hasta la inferior, así como una persona que lee una página de un libro. La primera suposición resultó ser la que en realidad ejecutaba el programa de manera que se colocó de manera oportuna la salida de los datos tal y como se introdujeron en el fichero, en la zona programada para el almacenamiento.

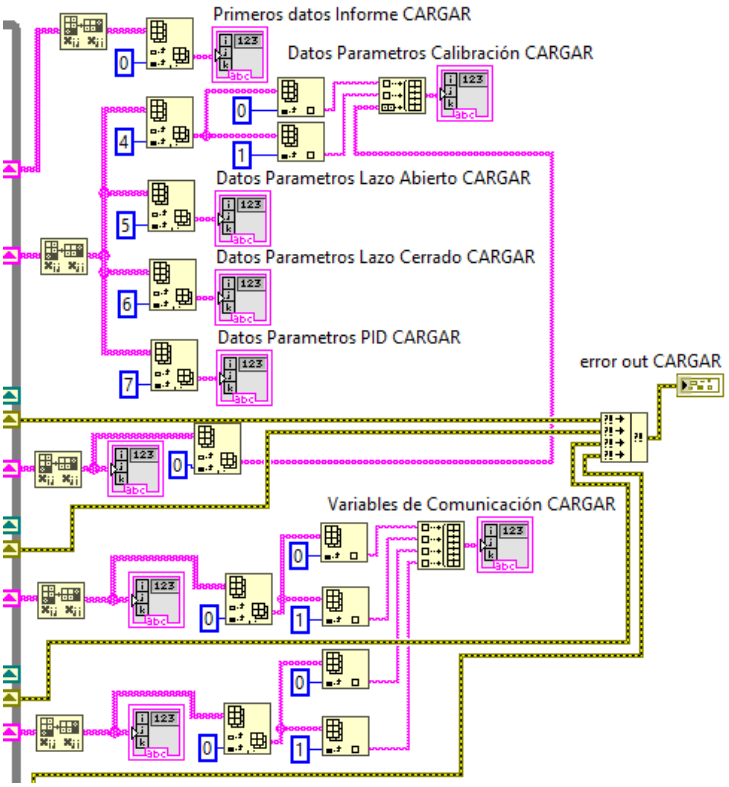


Figura 4.90. Código LabView colocación de los datos extraídos del fichero XML en matrices de tipo STRING para su posterior utilización.

En el caso de que no se hayan encontrado errores en esta parte del código tras su ejecución, el programa sale de la subrutina proporcionando los correspondientes datos cargados del fichero XML.

Con esto último queda finalizado el desarrollo y la arquitectura expuesta del código del programa de este proyecto. De manera que el lector ya estaría dispuesto a comprender la resolución de este proyecto con base en la programación visual de LabView junto con el manual de usuario estructurado anteriormente.

4.2.1.4. ESTRUCTURA DEL FICHERO XML:

Para poder comprender la estructura diseñada en lo que respecta al fichero XML que se ha implementado en este software, se mencionarán cada uno de los puntos que permiten al usuario guardar y/o cargar los datos registrados en este documento. Para poder guardar o cargar los datos del fichero XML se disponen en LabView de los siguientes bloques que fueron utilizados con el fin de almacenar los datos en el fichero XML (figura 4.91.):

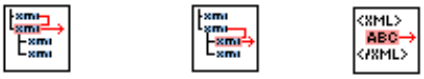


Figura 4.91. Edición del código del fichero XML

Como se puede observar, los dos primeros bloques de la izquierda generan un nuevo nodo de información relativa un nuevo apartado dentro del documento, mientras que el último de los bloques, el de más a la derecha permite extraer la información guardada en un apartado previamente creado. Dentro de lo diseñado se puede comenzar describiendo el inicio del documento de la siguiente manera (figura 4.92.):

```
<?xml version="1.0" encoding="UTF-8"?>
<INFORME>
    <FECHA>10_01_2018</FECHA>
    <HORA>12_19_46</HORA>
    <BY>Sfer4D_Corporation</BY>
```

Figura 4.92. Cabecera del fichero XML

Como se puede ver, se han incluido tres cabeceras previamente al registro de los datos de las posteriores tareas del software. De entre ellas son la fecha de creación del correspondiente documento junto con su hora. Además de ello se añadió de forma extraordinaria la denominación de una empresa ficticia para que el usuario viera la finalidad que podría tener a nivel industrial con su registro corporativo, además de lo aplicado en la universidad de cara a efectos académicos. Posteriormente se diseñará los nodos para guardar las

```
- <VARIABLES_COMUNICACION>
  - <VARIABLE_1>
      <TIPO_VARIABLE>Lectura</TIPO_VARIABLE>
      <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/ENTRADAS ANALOGICAS.In_Volts00</URL>
    </VARIABLE_1>
  - <VARIABLE_2>
      <TIPO_VARIABLE>Escritura</TIPO_VARIABLE>
      <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/SALIDAS ANALOGICAS.Out_%00</URL>
    </VARIABLE_2>
```

Figura 4.93. Registro de URLS de lectura y escritura en el fichero XML

URLS de lectura y escritura. Éstos se almacenarán creando una variable para cd una de ellas con el fin de que queden separadas como nodos diferentes en donde se podrían guardar varias URLS de lectura, así como de escritura, pero separadas según su naturaleza. Seguidamente el usuario podrá continuar guardando datos en el fichero XML, permitiéndosele contemplar como los datos posteriores quedan registrados en nuevos apartados. En el caso de la tarea siguiente, de calibración de los datos recibidos por el sensor de presión de la planta, se tendrá el siguiente código del documento actualizado:

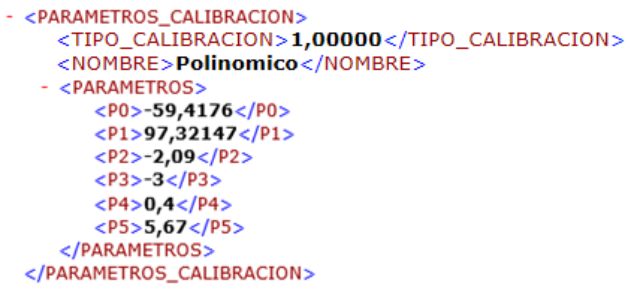


Figura 4.94. Registro de los parámetros de calibración en el fichero XML

En el ejemplo de la imagen anterior (figura 4.94.) el usuario ha guardado un tipo de calibrado polinomial de grado sexto, el motivo de ello es que es lo máximo que se puede calibrar de manera polinómica. En el fichero se distingue en primera instancia, el tipo de calibración, el nombre de la calibración y por último sus parámetros correspondientes a dicha ajuste lineal.

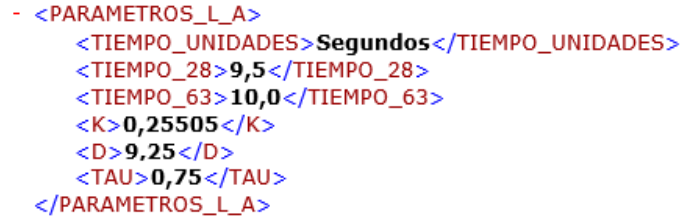


Figura 4.95. Registro de los parámetros de la experimentación en lazo abierto L.A. en el fichero XML

Para seguir introduciendo datos el programador ha debido de insertar nuevos bloques referentes a nuevas líneas de comando para poder inscribir en ellas el dato o datos que se hayan calculado en el programa referente a la tarea de experimentación en lazo cerrado, la cual será la siguiente de la cual se calcularán dos de los datos necesarios para el final diseño del controlador de tipo PID.

En el caso del presente proyecto, la vía óptima para poder guardar todos los datos que al final se verán en el fichero XML es prescindir de emplear los

bloques disponibles (se usarían dos por cada dato que se quieran introducir y en el caso del almacenaje de las URLS tres al considerar la variable 1 para lectura y variable 2 para escritura) y escribir el código del fichero XML según la naturaleza de los datos que se quieran guardar junto con los apartados que se quieran incluir. Para poder describir la estructura del fichero XML el código implica hacer uso de sucesivas constantes de tipo STRING para el código y los valores introducidos como variables de entrada, cuya salida única sería una matriz de STRINGS constante, conformando el fichero XML con líneas STRING.

Las ventajas de ello son evitar aglomerar varios bloques dependientes entre sí para generar subapartados en donde alojar, posteriormente, los datos extraídos de la ejecución del programa y poder modificar el fichero XML moviendo únicamente una fila de tipo STRING para añadir el apartado o dato que se quiera guardar (véase Figura 4.86. para facilitar la comprensión). De la otra forma en base a bloques, el programador deberá conocer perfectamente el funcionamiento de todos los bloques que trabajan con el fichero XML y poder "romper" la dependencia que entre ellos para incluir un nuevo apartado o nodo vacío donde se registraría un nuevo dato. En lo correspondiente a los datos obtenidos en la experimentación en lazo cerrado, el fichero muestra parte del siguiente código (figura 4.96.):

```
- <PARAMETROS_L_C>
    <TIEMPO_UNIDADES>Segundos</TIEMPO_UNIDADES>
    <KC>5,00000</KC>
    <TC>2,00000</TC>
  </PARAMETROS_L_C>
```

Figura 4.96. Registro de los parámetros de la experimentación en lazo cerrado L.C. en el fichero XML

Y por último en lo que respecta al diseño del controlador PID se tendrá una situación similar donde los parámetros generados en dicha tarea serán guardados en el fichero XML, en un nuevo apartado con el nombre de la tarea mencionada y con sus respectivos nodos vacíos con denominación para guardar en ellos los diversos datos respectivamente (véase figura 4.97.):

```
  - <PARAMETROS_PID>
      <TIEMPO_UNIDADES>Minutos</TIEMPO_UNIDADES>
      <KP>0,25</KP>
      <TI>0,01</TI>
      <TD>0,00</TD>
    </PARAMETROS_PID>
</INFORME>
```

Figura 4.97. Registro de los parámetros del controlador PID en el fichero XML

En lo que respecta la carga de los datos del fichero XML, el programador podrá ver que en este caso sí que han sido necesarios los bloques que se han expuesto al principio de este apartado (Véase Figura 4.91.) para acceder a los diversos puntos del código del fichero y por consiguiente los nodos de información de donde se extraerían los datos que se habían guardado en él, previamente. El programa recorrerá como en un vector de datos numéricos cada apartado para la extracción por apartados de la información almacenada en él.

Y con todo lo comentado, el lector podrá estar en disposición de comprender el funcionamiento completo del software, aquí presente ya sea en modo usuario; cuya interfaz carece de toda programación alguna que indique la configuración lógica estructurada de cada tarea individual y conjuntamente bajo una perfecta sincronización, y el modo programador en donde sí que se podrá contemplar todo lo codificado para levantar la arquitectura de todos los puntos de la totalidad del programa.

En el siguiente capítulo lo que se analizará, será la planta sobre la cual crece todo el proyecto junto con el compendio de elementos de hardware, además de los de comunicación y transmisión de datos, que han posibilitado la creación de este software.

Con todo ello descrito acerca del diagrama del código LabView del programador y la interfaz del usuario, se verá en el siguiente capítulo un ejemplo práctico referente a la planta de laboratorio que se utilizó para poder llevar a cabo el diseño del software del presente proyecto, en donde se detallarán los aspectos más importantes de cara a los dispositivos que tuvieron una implicación en lo correspondiente al funcionamiento del programa creado.

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

V – EJEMPLO PRÁCTICO

TRABAJO FIN DE GRADO – KEN VERA CHAN
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

INDICE DE CONTENIDOS:

5. EJEMPLO PRÁCTICO:

En este capítulo lo que se pretenderá será analizar la planta de pruebas sobre la cual se pudo construir la arquitectura y desarrollar el software de diseño de controladores PID. A lo largo de presente capítulo se expondrán todos los elementos que han formado parte del sistema cuyo funcionamiento básico se puede enunciar de la siguiente manera: "Control de la altura de agua de un depósito, con una bomba hidráulica, en función de la apertura de una válvula de evacuación de fluido situado en la base del recipiente" Para comprender el esquema al que la frase anterior se refería, se adjunta una imagen a continuación (figura 5.1.):

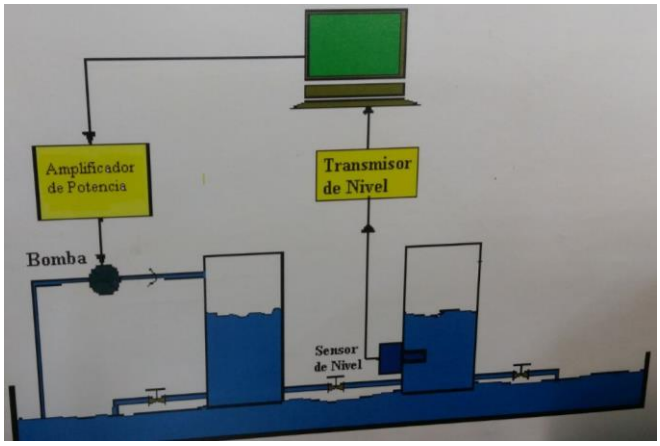


Figura 5.1. Descripción esquemática de la planta de procesos

Obsérvese que el nivel de agua que se pretende estudiar es la que se encuentra en el depósito de la derecha, según la imagen, ello es debido a que debido al cauce turbulento que la bomba provoca en el llenado del depósito de la izquierda, con sólo uno de ellos el sensor de presión no sería preciso en la medida que proporcionaría al ordenador debido al ruido que incluiría en las medidas.

En lo que respecta a la planta real, el usuario podrá contemplar las siguientes zonas del hardware que acompaña al proyecto (Ver figura 5.2.) en donde se ven, de izquierda a derecha, los dos depósitos verticales de agua, los tubos de flujo de agua que la bomba impulsa desde la realimentación con una toma de agua en un tubo inferior. La planta se encuentra instalada en una pequeña estantería blanca en cuyo nivel inferior se podrá ver el sensor de presión que generará los datos desde la medición física hasta la transmisión de dichos datos al ordenador que tiene a su izquierda. La bomba de agua se encuentra al lado del sensor de presión y tiene forma cilíndrica en posición horizontal. En

el nivel intermedio se puede ver el dispositivo de control de potencia y en el nivel superior se tiene el panel de control de la actividad del sensor, así como el juego de accionamientos que activan la planta de procesos:

Figura 5.2. Descripción real de la planta de procesos

A continuación, se expondrán cada uno de los elementos que intervienen en la imagen expuesta anteriormente en el funcionamiento del sistema.

5.1. DESCRIPCIÓN DE LA PLANTA DE PRUEBAS:

La planta de pruebas se encuentra en la residencia Alfonso VIII, calle doctor Mergelina, primera planta en el laboratorio de control de procesos. Actualmente, esta célula se utiliza para experimentos de control automático para las diversas asignaturas del área de ingeniería de sistemas y automática. A continuación, se podrán observar alguna de las ilustraciones de la planta en cuestión:

5.1.1. CONTROL DE POTENCIA Y CONVERTIDOR CA-CC:

Dispositivos que controlan y regulan la entrada de tensión que consumirá la bomba hidráulica en la planta de pruebas. En la figura 5.3. Se observa que el dispositivo de la izquierda representa un convertidor CA/CC que está alimentando al sensor de presión que debe estar entre 12 y 45 voltios en continua, en la planta se alimenta con 24 voltios. El dispositivo de la derecha alimenta a la bomba y del mismo modo que en el anterior, realiza la correspondiente conversión CA/CC proporcionando a la bomba una tensión nominal de 24 voltios. Como se puede ver en la siguiente imagen el dispositivo de control de potencia tendrá el siguiente aspecto exterior:

Figura 5.3. Control potencia planta

El convertidor de potencia es un componente electrónico que se ha diseñado para poder realizar la conversión que se ha comentado, en cuya imagen que se ofrece a continuación (figura 5.4.) se podrá ver que la circuitería ha sido diseñada por parte de algún departamento de la universidad que haber sido comprado en alguna tienda física o comprada por internet:

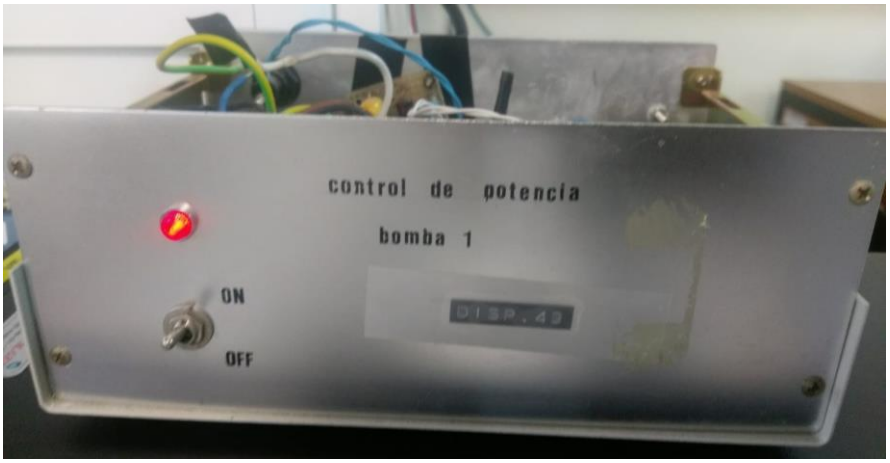


Figura 5.4. Vista en alzado del componente de control de potencia.

Al abrir la caja del dispositivo se pueden ver los cables y la serie de chapas metálicas que se conforman la misma, observándose el botón de encendido y el led de color rojo en frente que se iluminará cuando el dispositivo quede activo. Así puede verse en la siguiente imagen (figura 5.5.) que el circuito interior podría haberlo diseñado algún departamento de electrónica de la escuela de ingenierías industriales de la universidad de Valladolid (E.I.I. UVa).

Cabe comentar a título informativo que dentro de la caja de la imagen mostrada a continuación se pueden distinguir en color azul rectangular, condensadores de capacitancia media, los cilíndricos que se ven redondos con la perspectiva que se tiene en la imagen son también condensadores electrolíticos, mientras que los de color amarillo son condensadores cerámicos. Los elementos de color negro rectangular con pestañas doradas son amplificadores operacionales de diferente naturaleza, los elementos de tres patillas pegados en la pared superior de la imagen corresponden a transistores o amplificadores de corriente y el resto de los elementos alargados y con franjas de colores corresponden a resistencias de valores fijos según la gama de tonalidades de las bandas que las decoran.

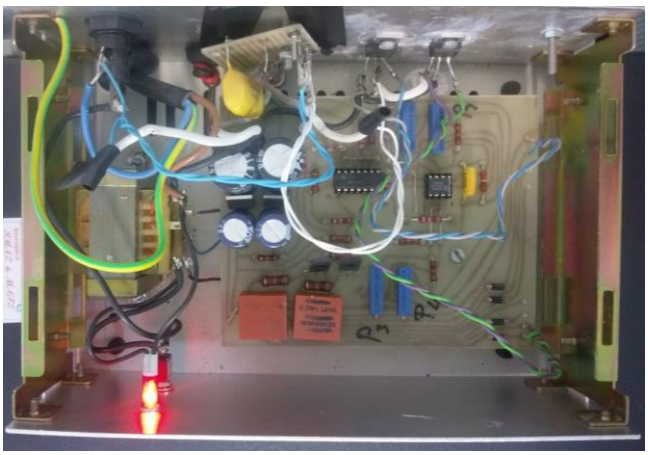


Figura 5.5. Vista interior del componente de control de potencia.

5.1.2. BOMBA HIDRÁULICA:

Es el componente que se encarga del movimiento del agua de cara a llenar los depósitos cilíndricos que se tienen a un lado pudiendo realizar las pruebas experimentales de control automático en la planta.

Figura 5.6. Bomba de agua de la planta

Bomba: FLOJET modelo RLF22202D 24v DC flujo de 3,8 litros/min y 2,4 bar de presión.

5.1.3. SENSOR DE PRESIÓN:

Este dispositivo es un transmisor de presión, colocado en la planta con el fin de poder conseguir indirectamente la altura de la superficie libre del fluido en el depósito (Véase cilindro vertical de la izquierda de la figura 5.11.), el funcionamiento de éste componente de cara a la medición de la mencionada altura se basa en el principio de la hidrostática, en donde el prisma de presiones en función de la altura ejerce una fuerza sobre la superficie circular de la base del depósito, que se traduce como presión, dato que se registra por éste sensor y se trasmite a un servidor para su posterior análisis (Véase figura 5.7):

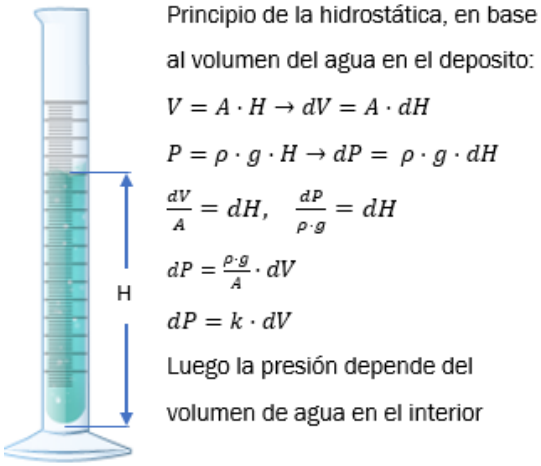


Figura 5.7. Principio de la hidrostática para la medición del sensor.

El funcionamiento del sensor de presión depende del tipo del diseño electrónico interno del transmisor que envía al ordenador la señal eléctrica de la altura del agua que tendría el depósito. En la siguiente ilustración se indican las partes de un transmisor común (figura 5.5.):

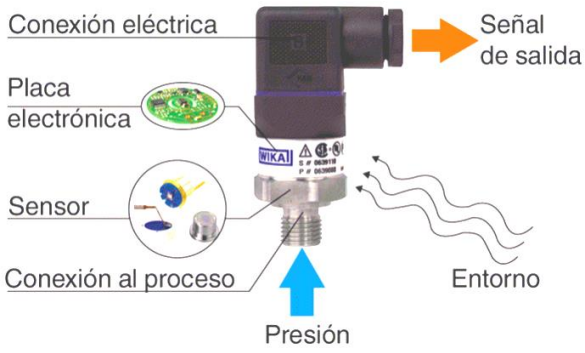


Figura 5.8. Elementos de un transmisor de presión.

Fuente: http://www.bloginstrumentacion.com/productos/como-funciona-un-transmisor-de-presion/

Para la medida de presión con transmisores de presión, o sensores de presión se requiere un sensor que mide el valor de presión o la variación de la misma y lo convierte en una señal eléctrica. La señal eléctrica indica el valor de presión recibida. Los cuatro principios más importantes son la medida con sensores resistivos, sensores piezoresistivos, sensores capacitivos y sensores piezoeléctricos. Se expondrá el funcionamiento del sensor de presión de tipo capacitivo, el cual es aquel que forma parte del hardware del proyecto en la planta de procesos.

Sensor de presión capacitivo:

Este principio está basado en la medición de la capacidad de un condensador que varía en función de la aproximación a la superficie activa. La capacidad de un condensador de dos placas puede expresarse por la siguiente ecuación:

$$C = \varepsilon \frac{A}{d}$$

C = capacidad condensador
ε = constante dieléctrica
A = área efectiva de las placas
d = distancia entre las placas

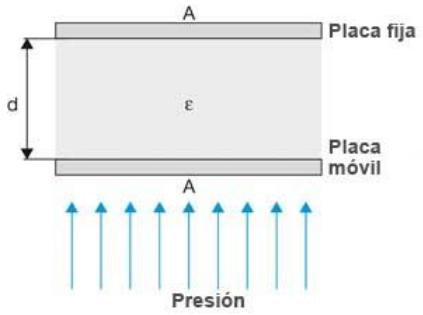


Figura 5.9. Funcionamiento de un transmisor de presión capacitivo.

Fuente https://www.wika.es/landingpage_pressure_sensor_es_es.WIKA

El principio de la medición capacitiva se realiza mediante un cuerpo base cuya membrana metálica, con recubrimiento metálico, constituye una de las placas del condensador. La deformación de la membrana, inducida por la presión, reduce la distancia entre las dos placas con el efecto de un aumento de la capacidad, manteniendo igual la superficie y la constante dieléctrica. Este sistema permite la medición de presión con elevada sensibilidad y por lo tanto la medición de rangos muy bajos hasta unos pocos milibares. Dado que la membrana permite una deformación máxima hasta apoyarse a la placa estática resulta una elevada seguridad contra sobrecarga. Las limitaciones practicas están determinadas por el material y las características de la membrana y las técnicas de unión y sellado.

En la planta del laboratorio analizada, el transmisor de presión es de tipo capacitivo. En la siguiente imagen se muestra el mismo (figura 5.10.), cuyo funcionamiento se describió líneas más arriba:

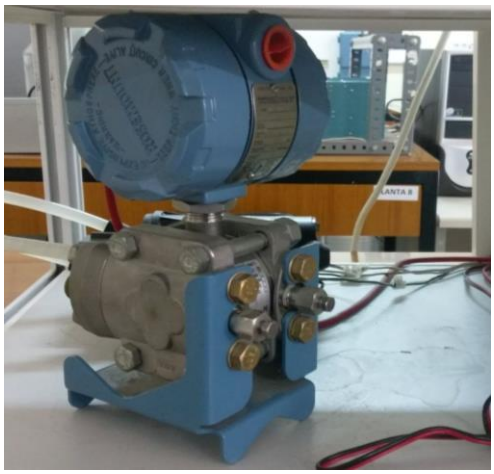


Figura 5.10. Sensor de presión de la planta

Sensor de presión: Sensor de presión inteligente modelo LD301MS alimentación de 12 a 45 V DC con señal de salida de 4,2 mA a dos hilos. Fluido del proceso Líquido, gas o vapor.

La descripción interna del sensor de presión es la que se muestra en la siguiente imagen (figura 5.11.):

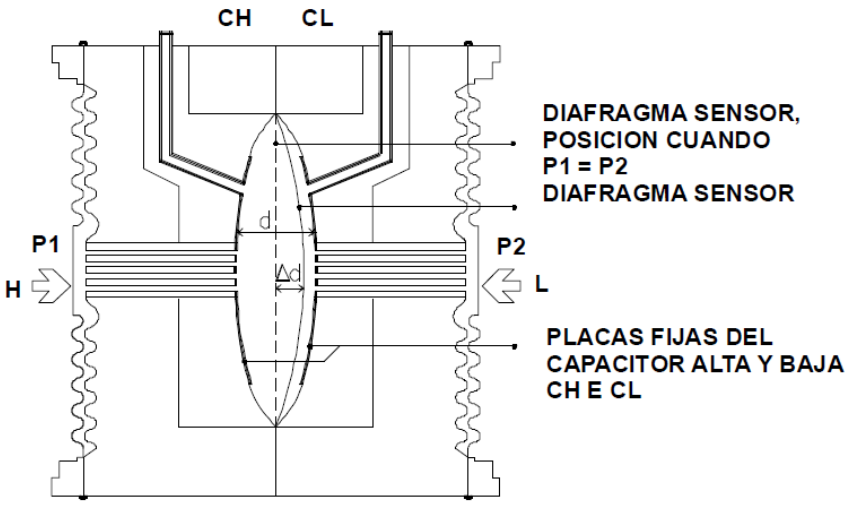


Figura 5.11. Sensor de presión, descripción interna.

Fuente: Manual sensor de presión modelo LD301MS

Cuyos datos representativos se exponen a continuación: P1 y P2 son las presiones aplicadas en las cámaras H y L.

CH: Corresponde con la capacitancia medida entre la placa fija en el lado P1 y el diafragma sensor.
CL: Corresponde con la capacitancia medida en la placa fija en el lado P2 y el diafragma sensor.
d: La distancia entre las placas fijas CH y CL.
Δd: supone la deflexión sufrida por el sensor de diafragma debida a la aplicación de presión diferencial DP=P1-P2.

Sabiendo que la capacitancia de un condensador de placas planas paralelas puede expresarse como una función del área de la placa A y de la distancia, d, que se separan como:

$$C = \frac{\in A}{d}$$

Donde, la ε corresponde con la constante dieléctrica del medio existente entre las placas del capacitor. Si se consideran CH y CL como las capacitancias de las placas planas y paralelas con áreas idénticas, se tendrá entonces:

$$CH = \frac{\in A}{(d/2)+\Delta d} \qquad \text{y} \qquad CL = \frac{\in A}{(d/2)-\Delta d}$$

Sin embargo, si la presión diferencial (ΔP) aplicado al elemento capacitivo no desvía el sensor del diafragma más allá de una distancia d/4, cabría suponer que ΔP es proporcional a Δd. Al desarrollar la expresión (CL-CH) /(CL+CH), se puede deducir que:

$$\Delta P = \frac{CL-CH}{CL+CH} = \frac{2\Delta d}{d}$$

Como la distancia (d) entre la placa fija CH y CL es constante, es posible concluir que la expresión (CL-CH) /(CL+CH) es proporcional a ΔP y, por consiguiente, a la presión diferencial a ser medida. Así es posible que resulte que la célula capacitiva es un sensor de presión formado por dos capacitores de capacitancias variables, según la presión diferencial aplicada.

5.1.4. DEPÓSITOS DE AGUA:

Son los depósitos que se irán llenando de agua a medida que se realicen las correspondientes experimentaciones. El depósito de la izquierda, en la imagen, es el que tiene adherido el sensor de presión mediante un cable de plástico por donde circulará el agua, el tubo de agua comunicará la misma presión hidrostática que posea el recipiente, aunque sea a través del tubo más apenas añade altura alguna al respecto. Con ello se recogerán los datos de la altura del agua en dicho depósito. Como se argumentó anteriormente, objetivo de tener el depósito de la derecha es introducir una dinámica un poco más compleja ya que habrá un cierto retardo entre la variación del caudal de entrada al depósito de la derecha y su transmisión por vasos comunicantes al de la izquierda. Es decir, hace el sistema un poco más difícil de controlar.

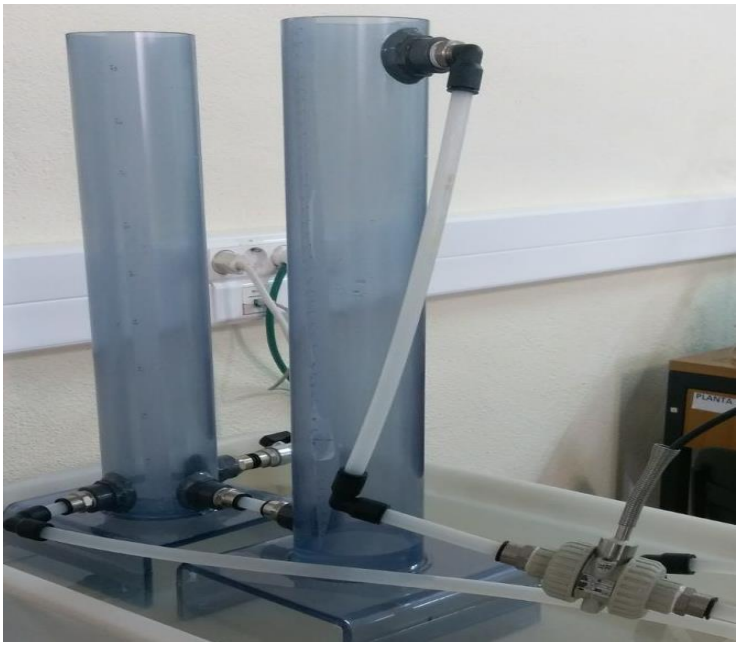


Figura 5.12. Depósitos de agua

Obsérvese en la imagen que el depósito de la derecha recibe el caudal que la bomba hidráulica proporciona del recipiente inferior donde reposa el fluido de trabajo.

5.1.5. CONTROL DE MANDO DE LA BOMBA Y SENSOR:

En esta sección se contemplarán los accionamientos eléctricos que posibilitan el encendido de la planta, así como los interruptores que ponen en marcha el convertidor de potencia y el transmisor de presión.

Figura 5.13. Accionamientos eléctricos y medidor de la planta.
Transmisor de variables analógicas: Modelo DMM4000, alimentación de 12-24 v DC, salida 2x4-20 mA, comunicación mediante puerto con cable de tipo RS-485 Modbus

En lo que respecta al accionamiento eléctrico, permite con bajo costo tomar medidas distribuidas en campo y transmitidos en miliamperios y en Modbus hacia sistemas de control como PC descargando a estos de las tareas de conversión de medida de conversión de analógica a digital (A/D) y a la inversa, conversión de digital a analógica (D/A)

5.2. TARJETA DE ADQUISICIÓN DE DATOS (DAQ):

El hardware DAQ actúa como la interfaz entre un ordenador y señales del mundo exterior. Funciona principalmente como un dispositivo que digitaliza señales analógicas entrantes para que una PC pueda interpretarlas.

Los tres componentes clave de un dispositivo DAQ usado para medir una señal son:

- El circuito de acondicionamiento de señales.
- El convertidor analógico-digital (ADC)
- Un bus de PC.

Varios dispositivos DAQ incluyen otras funciones para automatizar sistemas de medidas y procesos. Por ejemplo:

- Los convertidores digitales-analógicos (DAQs) envían señales analógicas
- Las líneas de E/S digital reciben y envían señales digitales.
- Los contadores/temporizadores cuentan y generan pulsos digitales.

En la siguiente ilustración se puede contemplar la DAQ que conecta la planta de proceso con el ordenador (figura 5.14.):

Figura 5.14. Tarjeta de Adquisición de Datos (DAQ) del laboratorio.

Tarjeta adquisición de datos (DAQ), modelo USB-1408-FS.PLUS, tensión máxima entrada 28V, impedancia de entrada 122 Kohm, resolución de 14 bits y salida de 0V a 5V

En el caso de la planta de pruebas para el desarrollo del presente proyecto se ha empleado una tarjeta de adquisición de datos basada en transferencia de datos proporcionados por un sensor en forma de señales eléctricas. El modo en que opera la tarjeta de adquisición de datos es la expuesta en la siguiente ilustración:

Figura 5.15. Funcionamiento de la tarjeta de adquisición de datos (DAQ)

Fuente: https://www.ni.com/data-acquisition/what-is/esa/

Como se mencionó, la DAQ trabaja con tres componentes, cuyas funcionalidades se detallarán a continuación:

- El circuito de acondicionamiento de señales:

Las señales de los sensores o del mundo exterior pueden ser ruidosas o demasiado peligrosas para medirse directamente. El circuito de acondicionamiento de señales manipula una señal de tal forma que la hace apropiada para entrada a un convertidor analógico-digital. Este circuito puede incluir amplificación, atenuación, filtrado y aislamiento. Algunos dispositivos DAQ incluyen acondicionamiento de señales integrado diseñado para medir tipos específicos de sensores.

- El convertidor analógico-digital (ADC):

Las señales analógicas de los sensores deben ser convertidas en digitales antes de ser manipuladas por el equipo digital como una PC. Un convertidor analógico-digital es un chip que proporciona una representación digital de una señal analógica en un instante de tiempo. En la práctica, las señales analógicas varían continuamente con el tiempo y un ADC realiza "muestras" periódicas de la señal a una razón predefinida, lo que se denomina en control, discretización de una señal analógica de entrada. Estas muestras son transferidas a una PC a través de un bus, donde la señal original es reconstruida desde las muestras en software.

- Un bus de PC:

Los dispositivos DAQ se conectan a un ordenador a través de una ranura o puerto. El bus de la PC sirve como la interfaz de comunicación entre el dispositivo DAQ y la PC para pasar instrucciones y datos medidos. Los dispositivos DAQ se ofrecen en los buses de PC más comunes, incluyendo USB, PCI, PCI Express y Ethernet.

Recientemente, los dispositivos DAQ han llegado a estar disponibles para redes de tipo 802.11 (las comúnmente conocidas como redes de tipo Wi-Fi) para comunicación inalámbrica. Hay varios tipos de buses y cada uno de ellos ofrece diferentes ventajas para diferentes tipos de aplicaciones.

- Servidor OPC:

En el presente proyecto se hace uso de un servidor OPC en donde se tienen registradas las correspondientes URLS que se emplearán en el cometido para el que fue diseñado el software, de manera que para ello se deberá encontrar aquella URL que permita mostrar como señal analógica lo que está leyendo un sensor, indistintamente de la naturaleza que posea, y otra que permita al usuario utilizarla como variable manipulada de cara a

permitírsele escribir un valor numérico para cambiar el comportamiento de la planta de proceso analizada.

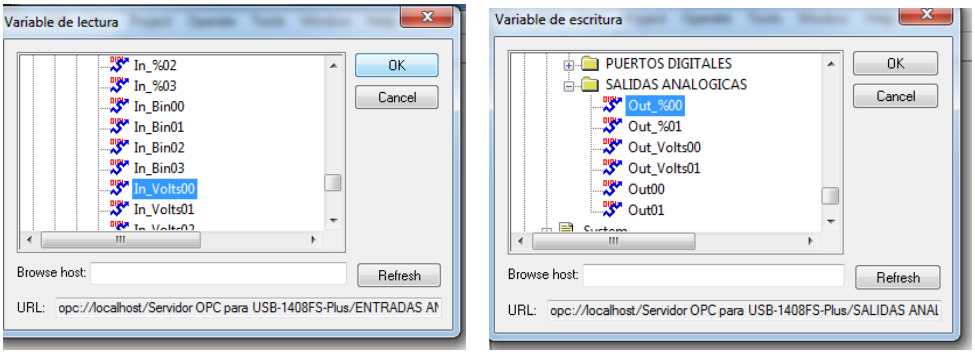


Figura 5.16. Variables lectura y escritura en las URLS del servidor OPC.

Cuando se conecta la tarjeta de adquisición de datos (DAQ) el servidor OPC posibilita conectar ésta con el host local del equipo proporcionando una sencilla interfaz al usuario con la misión de indicarle qué URLS están disponibles para un posterior estudio.

Las URLS se guardarán en diversas carpetas que al instalar el servidor en el ordenador (PC) podrá configurar los nombres para que se queden alojadas en ellas según la denominación asignada. En una posterior ocasión y tras fijar las direcciones de dichas URLS en el explorador de Windows, el usuario podrá seleccionar aquellas que le interese para poder realizar sus convenientes tareas de control.

5.3. CALIBRACIÓN DE DATOS DE TENSIÓN DE ENTRADA:

Cuando el ordenador recibe la información de la tarjeta de adquisición de datos, ésta llega en forma de tensión tras haberse procesado la variable física que el transmisor de medida le había proporcionado. En lo referente a la calibración de datos, es proceso que el software de tratamiento de la información obtenida por la DAQ, en este caso LabView, permitirá convertir los datos de tensión recibidos por el transmisor en datos de variable física, cualquiera que haya sido la que midió el transmisor. Dicho proceso sigue el diagrama ilustrativo siguiente para poder enfocar las ideas que se pretende con la calibración (Véase figura 5.17):

Figura 5.17. Esquema adquisición de datos y calibración

Como se puede observar, LabView puede diseñar programas capaces de tratar los datos de tensión que le llegan al ordenador y con ello realizar la calibración, como:

- Ecuación lineal.
- Ecuación polinómica.
- Ecuación exponencial.
- Ecuación logarítmica.

según qué sistema se trate y obtener la verdadera medida física que el transmisor captó en un cierto instante de tiempo. En el caso de una calibración basada en una ecuación lineal se trataría de una recta de regresión, que es la que en la planta de procesos del presente proyecto se ha empleado. Por ello, durante la ejecución del software del proyecto, tras establecer la comunicación mediante OPC, el usuario podrá calibrar la planta de procesos en la que esté ejecutándose el programa utilizando cualquiera de las posibles calibraciones comentadas líneas más arriba, siendo la lineal la más efectiva para conseguir una relación matemática simple de entrada de tensión y salida, la variable física medida que el transmisor proporcionó en forma de tensión durante un período de tiempo.

5.4. EJEMPLO INFORME PRÁCTICAS LABORATORIO EMPLEANDO ESTE SOFTWARE:

En el capítulo anterior se comentó cada uno de los aspectos acerca del uso del programa destinado al estudio de una planta de procesos en donde se pueda implementar un controlador PID. Debido a esto se va a incluir un ejemplo práctico con detalle para comprender la funcionalidad de cada uno de los puntos programados del software, en donde un usuario podrá contemplar todos los pasos que seguidamente se van a detallar.

Ante esto, se proporcionará una serie de ilustraciones de los pasos consecutivos para poder trabajar con eficiencia con este programa:

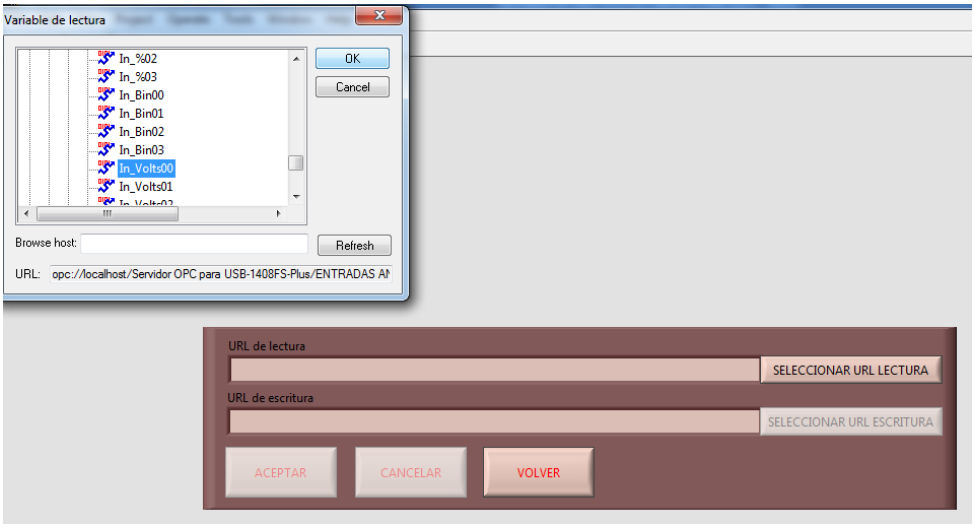


Figura 5.18. Interfaz usuario ejecución de la obtención URL de lectura.

Cuando se ejecuta la zona de obtención de las URLS, el usuario debe darse cuenta de que se abrirá una ventana con los servidores activos que se tendrán habilitados en el PC. Debe de tenerse en cuenta que al existir dos URL no puede caber confusión entre qué variable corresponde a una lectura de datos y qué variable corresponde con la de escritura (variable manipulada) más el programa informaría del error exigiendo de nuevo la petición de las mismas. En el caso de la imagen anterior (figura 5.18.) se puede ver que un usuario ha accionado el botón de introducción de la correspondiente URL de lectura; "In_Volts00" la cual corresponde con la obtención de todos los datos que se podrán recibir de la planta de proceso que se pretende controlar.

En la siguiente ilustración (figura 5.19.) se puede ver cómo se selecciona la URL de escritura, en el caso del ejemplo; "Out_%00", la cual se buscará y se introducirá tal cual y como se hizo con la variable de lectura.

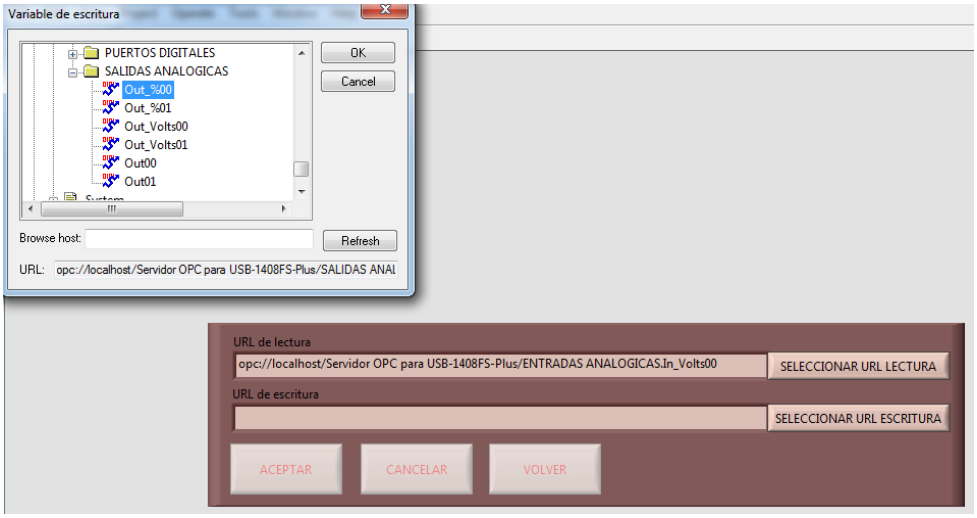


Figura 5.19. Interfaz usuario ejecución de la obtención URL de escritura.

La ventana emergente será idéntica a la que se propuso en la URL de lectura, existiendo un explorador de servidores activos. En el caso de que el usuario este conforme, la siguiente imagen mostrará estos datos guardados en el fichero XML y en la tabla "dinámica" de exposición de los datos registrados actuales (figura 5.19.) Tal y como se puede ver en la imagen las URLS se han quedado registradas en sendas celdas de una matriz de datos, hecho que se ejecutará a lo largo de todas las tareas que se van a realizar.

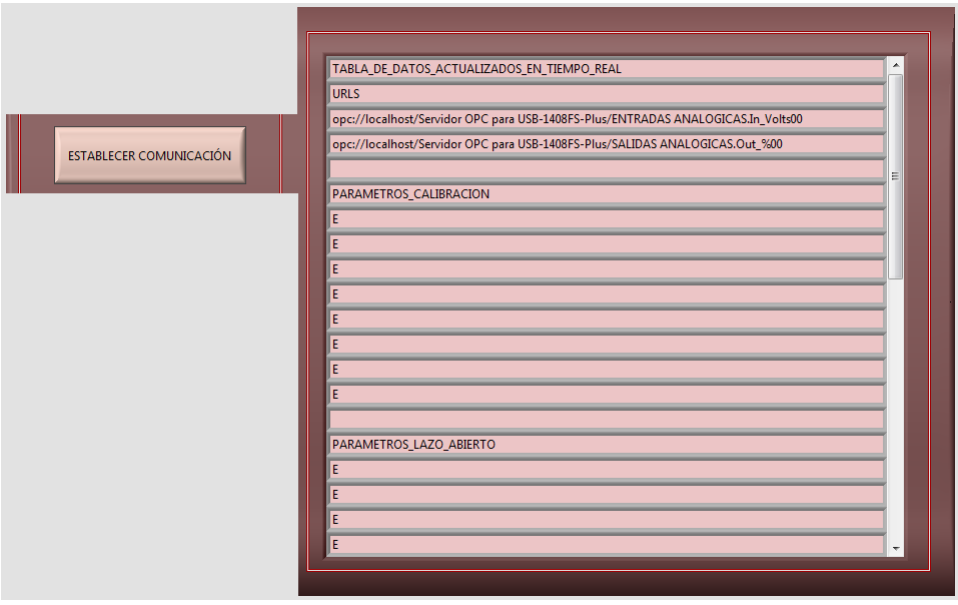


Figura 5.20. Interfaz usuario datos URLS registradas.

Posteriormente, y reanudando con la ejecución de las subrutinas, al seleccionar la calibración de los datos, como primera las tareas del cometido de este software, se tendrá lo siguiente (ver Figura 5.21.):

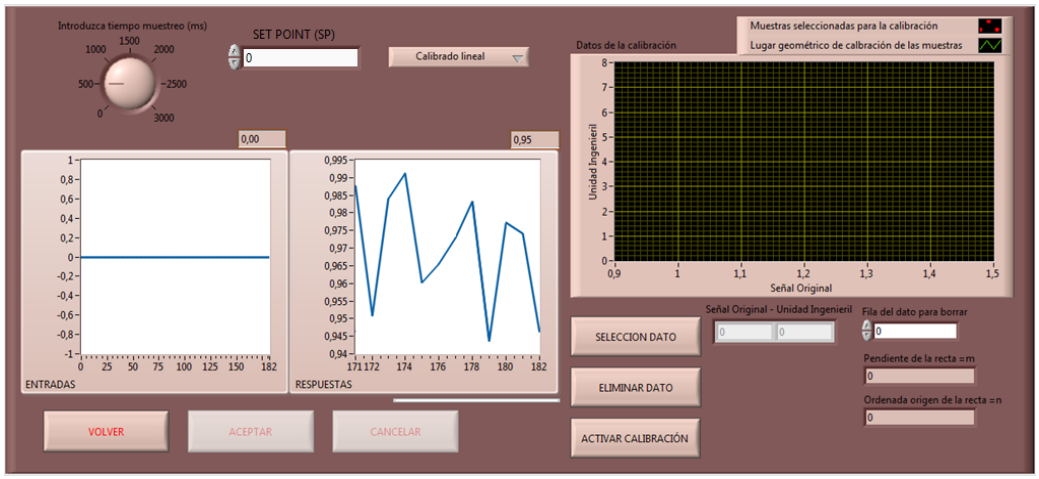


Figura 5.21. Interfaz usuario inicio ejecución de la tarea de calibración.

Los datos se irán recopilando en la segunda de las gráficas de fondo blanco oscilando dentro de un intervalo de datos debido al ruido de la señal percibida, hasta el instante en que el usuario introduzca una entrada escalón en la variable manipulada variando la magnitud de los datos que entrarían posteriormente. A medida que añada valores recibidos del sensor de presión, se podrá ajustar con mayor eficacia la interpolación lineal de los datos de cara a una aproximada calibración del sistema.

Obsérvese en la siguiente ilustración como se puede ver lo indicado líneas más arriba (Figura 5.22.), en donde un usuario ha introducido una serie de escalones reflejados en la primera de las gráficas de fondo blanco, contemplándose el efecto sobre la gráfica a su derecha.

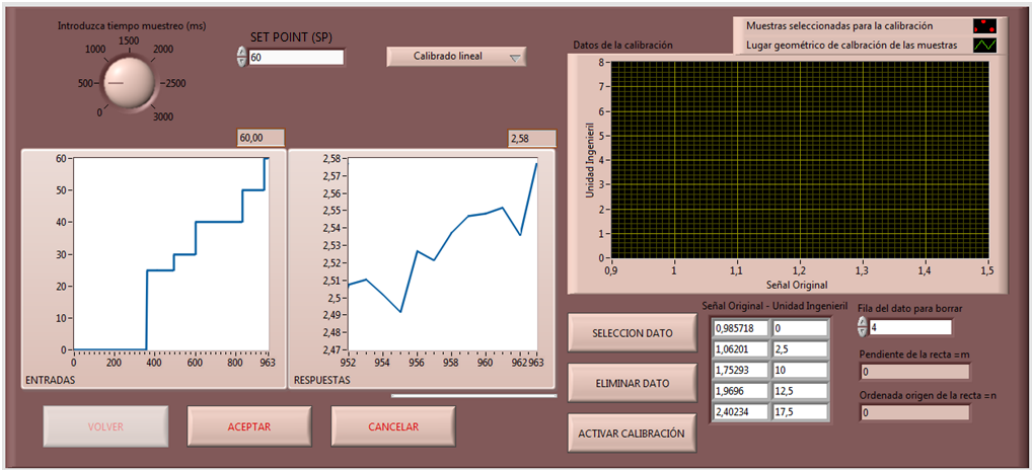


Figura 5.22. Interfaz usuario introducción de entradas escalón.

A medida que introduzca arbitrariamente escalones al sistema y ver como alcanza un punto estacionario de manera lenta, se podrá registrar el dato correspondiente a tal altura, guardándose en la matriz dinámica que se generará justo debajo de la gráfica de fondo oscura (ver figura 5.23.)

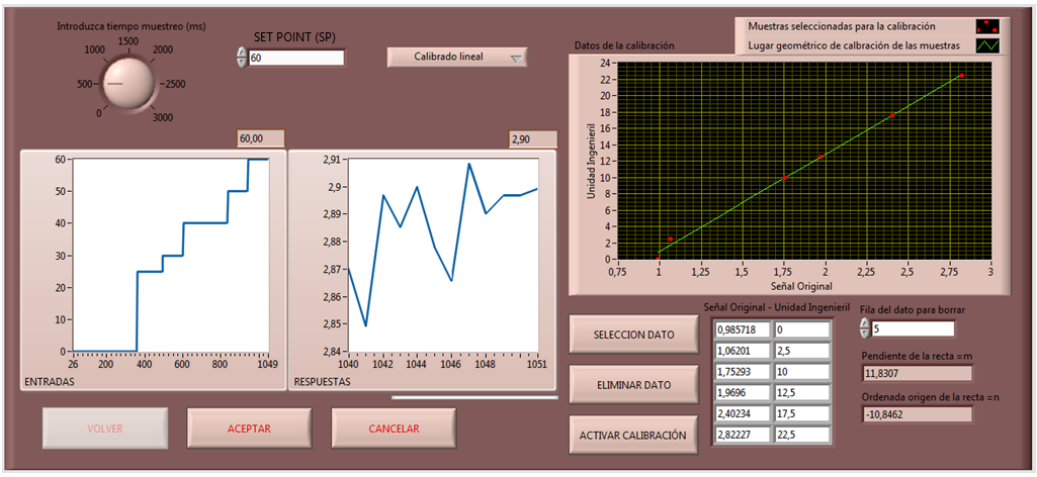


Figura 5.23. Obtención de la ecuación interpolada de la calibración.

Así pues, al ejecutar el calibrado de los datos, se obtendrá la curva generada en la gráfica de fondo negro en donde los puntos rojos representarán los datos almacenados mientras que la línea verde, sea recta o curva, corresponderá con la ecuación del calibrado del proceso. Seguidamente se obtendrán los parámetros de la ecuación que interpola los datos mostrados justo a la derecha de la matriz rellenada con los datos registrados de la planta analizada.

Si el usuario acepta el resultado que ha obtenido de la calibración, el programa abrirá el fichero XML para poder guardar estos datos obtenidos sea la calibración que sea, junto con un número adicional para poder identificar el tipo de calibrado que se ha utilizado en esta tarea. Este hecho del número incluido facilitará a las posteriores tareas, el reconocer el calibrado que se había ejecutado.

Si bien es cierto que las URLS que se generaron anteriormente, se guardaron en el fichero XML, éstas aparecerán en un primer apartado habilitado para ellas, identificando cuál corresponde con la de lectura y cuál con la de escritura (véase figura 5.24.) En este caso el almacenaje de los parámetros de la calibración se hará en forma matricial en donde se incluirá el dígito explicado, el nombre de la calibración y los parámetros de la misma, según la complejidad que posea. Dado que el fichero ha habilitado hasta seis coeficientes polinomiales (hasta grado quinto), en el caso de que sea interpolación lineal, quedarán algunos parámetros sin definir viéndose que no existen datos registrados en este punto dentro del fichero XML (P0,P1,...,P5).

```
<?xml version="1.0" encoding="UTF-8"?>
- <INFORME>
    <FECHA>08_06_2018</FECHA>
    <HORA>14_53_42</HORA>
    <BY>Sfer4D_Corporation</BY>
  - <VARIABLE_COMUNICACION>
    - <VARIABLE_1>
        <TIPO_VARIABLE>Lectura</TIPO_VARIABLE>
        <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/ENTRADAS ANALOGICAS.In_Volts00</URL>
      </VARIABLE_1>
    - <VARIABLE_2>
        <TIPO_VARIABLE>Escritura</TIPO_VARIABLE>
        <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/SALIDAS ANALOGICAS.Out_%00</URL>
      </VARIABLE_2>
    </VARIABLE_COMUNICACION>
  - <PARAMETROS_CALIBRACION>
      <TIPO_CALIBRACION>0,00000</TIPO_CALIBRACION>
      <NOMBRE>Lineal</NOMBRE>
    - <PARAMETROS>
        <P0>-10,84623</P0>
        <P1>11,83073</P1>
        <P2/>
        <P3/>
        <P4/>
        <P5/>
      </PARAMETROS>
    </PARAMETROS_CALIBRACION>
  - <PARAMETROS_L_A>
      <TIEMPO_UNIDADES>Segundos</TIEMPO_UNIDADES>
      <TIEMPO_28_PORCIENTO>E</TIEMPO_28_PORCIENTO>
      <TIEMPO_63_PORCIENTO>E</TIEMPO_63_PORCIENTO>
      <K>E</K>
      <D>E</D>
      <TAU>E</TAU>
    </PARAMETROS_L_A>
  - <PARAMETROS_L_C>
      <TIEMPO_UNIDADES>Segundos</TIEMPO_UNIDADES>
      <KC>E</KC>
      <TC>E</TC>
    </PARAMETROS_L_C>
  - <PARAMETROS_PID>
      <TIEMPO_UNIDADES>Minutos</TIEMPO_UNIDADES>
      <KP>E</KP>
      <TI>E</TI>
      <TD>E</TD>
    </PARAMETROS_PID>
</INFORME>
```

Figura 5.24. Obtención del fichero XML con los datos de la calibración.

Al igual que sucedía antes, todo lo que se guarde en el fichero XML se mostrará en la tabla de actualización en tiempo real adjunta al menú de opciones disponibles en el presente estudio (sea nuevo o antiguo)

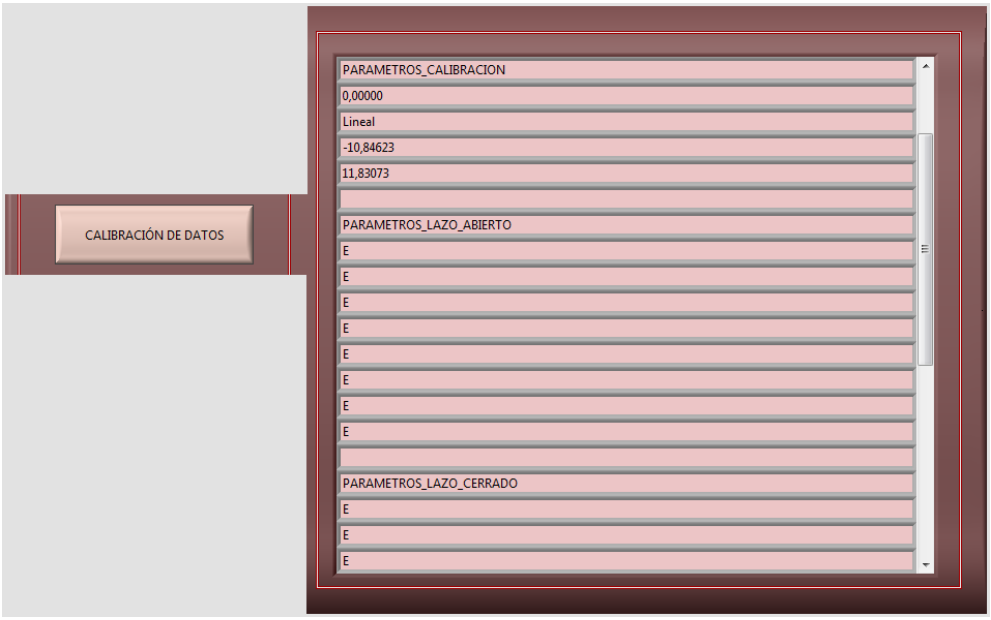


Figura 5.25. Actualización de los datos en la tabla tras la calibración.

Ahora que se tienen los parámetros de la calibración quedan habilitados las dos siguientes tareas basadas en la experimentación en lazo abierto (L.A.) y lazo cerrado (L.C.) en donde para poder seguir un orden de colocación de los accionamientos dentro del menú de opciones propuestos en la imagen anterior (figura 5.25.) se procederá con el inicio en la experimentación en lazo abierto.

Al adentrarse en la tarea mencionada, se podrá ver como el sistema comienza a recoger los datos leídos del sensor instalado en la planta del proceso, acumulándose en la segunda de las gráficas de la zona superior del panel de control propuesto (ver figura 5.26.):

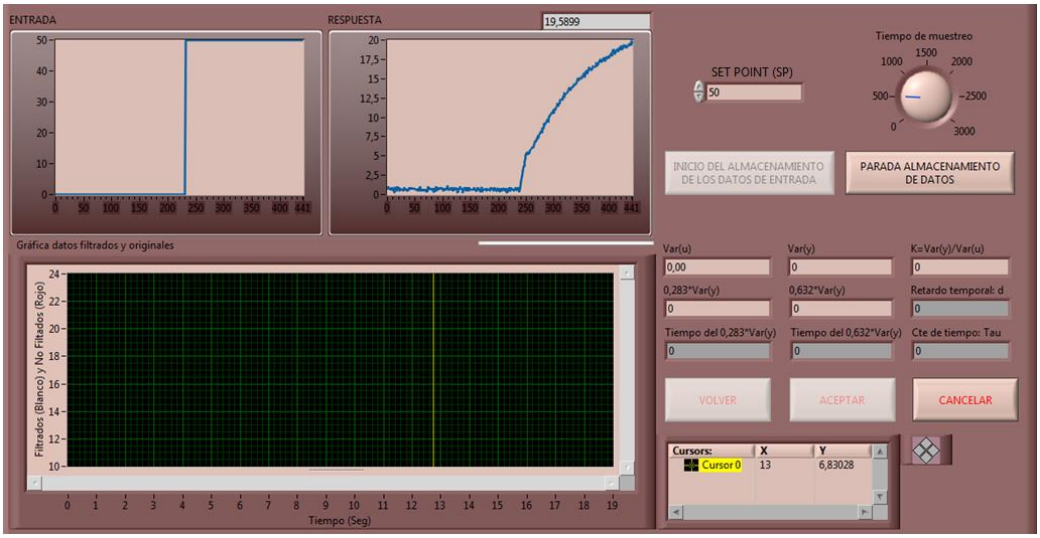


Figura 5.26. Almacenaje de los datos de la experimentación en lazo abierto.

Si se decide comenzar introduciendo una entrada escalón (reflejado en la gráfica de la izquierda) se podrá ver como en la de la derecha comenzará a generarse la curva de reacción resultante. En un momento dado, el usuario podrá accionar el botón de almacenaje de los datos calibrados recibidos (ver figura 5.26.), momento en que la experimentación habrá comenzado acumulándose los datos que compondrán la curva de reacción hasta que alcance un valor estacionario aproximado.

Cuando el sistema alcance dicho instante el usuario podrá detener el proceso generándose automáticamente la correspondiente curva con todos los parámetros que estudian la respuesta en lazo abierto (ver figura 5.27.); ganancia (K), constante de tiempo (Tau) y retardo (d) que se podrá visualizar en indicadores de la derecha junto con los tiempos correspondientes con el 28,3% y el 63,2% del total de la respuesta generada. Además de ello, el usuario podrá verificar la veracidad de los datos comprobando con un panel de control de cursor, las parejas de coordenadas que identificarán los porcentajes

mencionados anteriormente, así como el cálculo de los parámetros que se han conseguido en esta experimentación.

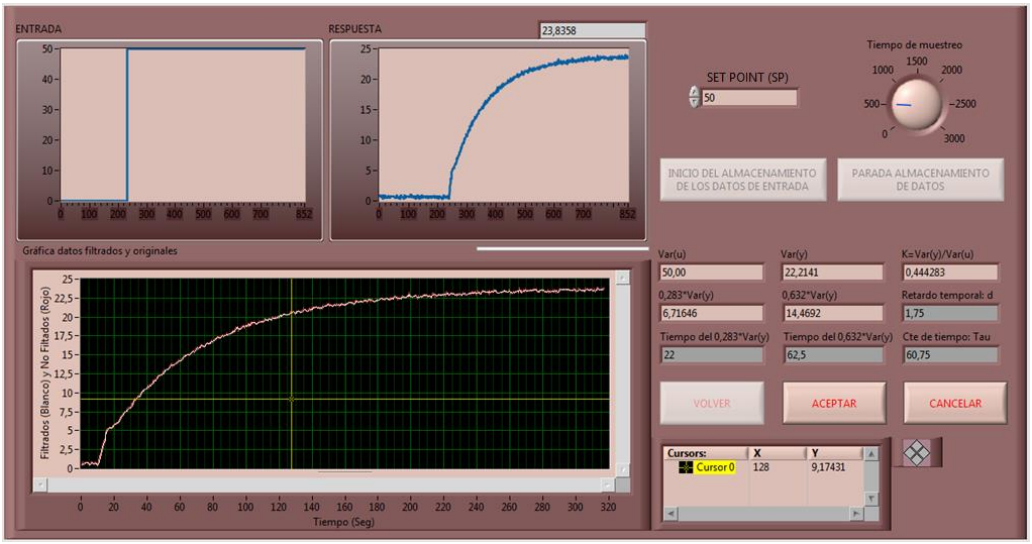


Figura 5.27. Generación curva reacción junto con los parámetros de la experimentación en lazo abierto.

Tras la detención se generará en la gráfica inferior de fondo oscuro, la acotación de datos generados por el almacenaje de los mismos aplicando la técnica de la media móvil para poder corregir posibles picos en la respuesta, así como otras complicaciones similares. Al accionar el botón de aceptar, el usuario podrá ver, una vez más, la tabla adjunta con los datos actualizados.

Figura 5.28. Actualización de los datos en la tabla tras la experimentación en lazo abierto (L.A.)

En ella se verá todos los datos recopilados de la finalizada tarea (figura 5.28.) pudiendo mencionarse que el usuario podrá acceder a la sintonía del PID sin necesidad de realizar la siguiente tarea de experimentación en lazo cerrado (L.C.)

En el fichero XML también quedará reflejado lo que se ha trabajado en dicha experimentación y como puede observarse, se ha incluido en otro apartado habilitado para ello (ver Figura 5.29.):

```
<?xml version="1.0" encoding="UTF-8"?>
- <INFORME>
    <FECHA>08_06_2018</FECHA>
    <HORA>14_53_42</HORA>
    <BY>Sfer4D_Corporation</BY>
  - <VARIABLE_COMUNICACION>
    - <VARIABLE_1>
        <TIPO_VARIABLE>Lectura</TIPO_VARIABLE>
        <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/ENTRADAS ANALOGICAS.In_Volts00</URL>
      </VARIABLE_1>
    - <VARIABLE_2>
        <TIPO_VARIABLE>Escritura</TIPO_VARIABLE>
        <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/SALIDAS ANALOGICAS.Out_%00</URL>
      </VARIABLE_2>
    </VARIABLE_COMUNICACION>
  - <PARAMETROS_CALIBRACION>
      <TIPO_CALIBRACION>0,00000</TIPO_CALIBRACION>
      <NOMBRE>Lineal</NOMBRE>
    - <PARAMETROS>
        <P0>-10,84623</P0>
        <P1>11,83073</P1>
        <P2/>
        <P3/>
        <P4/>
        <P5/>
      </PARAMETROS>
    </PARAMETROS_CALIBRACION>
  - <PARAMETROS_L_A>
      <TIEMPO_UNIDADES>Segundos</TIEMPO_UNIDADES>
      <TIEMPO_28_PORCIENTO>22,00000</TIEMPO_28_PORCIENTO>
      <TIEMPO_63_PORCIENTO>62,50000</TIEMPO_63_PORCIENTO>
      <K>0,44428</K>
      <D>1,75000</D>
      <TAU>60,75000</TAU>
    </PARAMETROS_L_A>
  - <PARAMETROS_L_C>
      <TIEMPO_UNIDADES>Segundos</TIEMPO_UNIDADES>
      <KC>E</KC>
      <TC>E</TC>
    </PARAMETROS_L_C>
  - <PARAMETROS_PID>
      <TIEMPO_UNIDADES>Minutos</TIEMPO_UNIDADES>
      <KP>E</KP>
      <TI>E</TI>
      <TD>E</TD>
    </PARAMETROS_PID>
</INFORME>
```

Figura 5.29. Obtención del fichero XML con los datos de la experimentación en lazo abierto (L.A.)

En el informe se podrá ver como se ha rellenado la parte correspondiente con la de la experimentación en lazo abierto.

Seguidamente, si se procede con la experimentación en lazo cerrado, se mostrará la siguiente ventana emergente en donde LabView ya está recogiendo los datos del sensor representándolos en la gráfica superior de fondo oscuro junto con la inferior que muestra la señal manipulada durante la ejecución. En

la gráfica superior se ve la señal en rojo del setpoint que se muestra a continuación en el panel del interfaz del usuario (ver figura 5.30.):

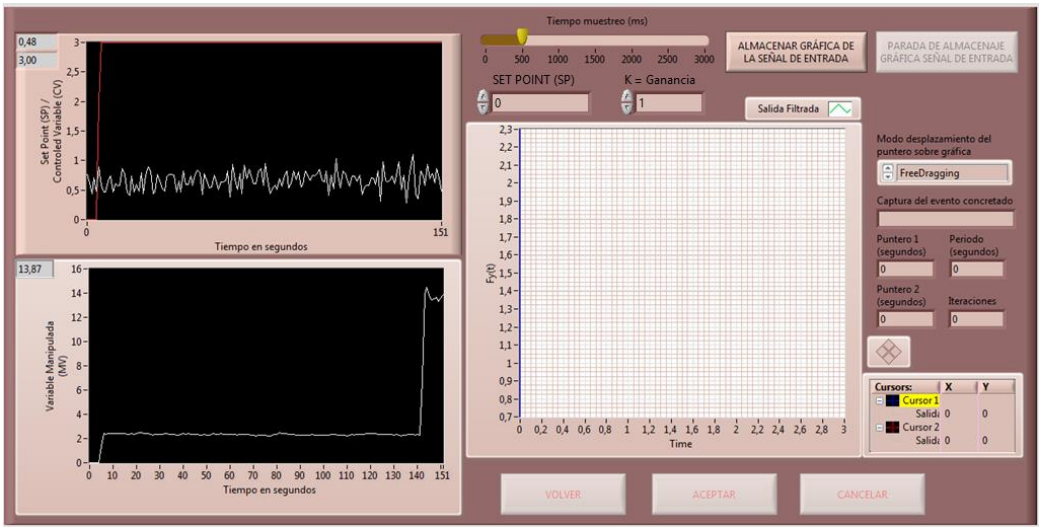


Figura 5.30. Inicio de la experimentación en lazo cerrado.

En la siguiente ilustración se puede contemplar cómo habiendo aumentado la ganancia a un valor crítico, se ha alcanzado un instante en que la respuesta se muestra oscilatoria. El usuario ha almacenado todos los datos que abordan el momento en que la ganancia alcanza su valor crítico por encima del cual el sistema proporciona una respuesta inestable. Cuando queda conforme con dicho valor, en la gráfica adjunta en la zona de la derecha podría sacar el valor del período crítico como se comentó en apartados anteriores, pudiendo realizar este hecho con los cursores que se proponen adjuntamente.

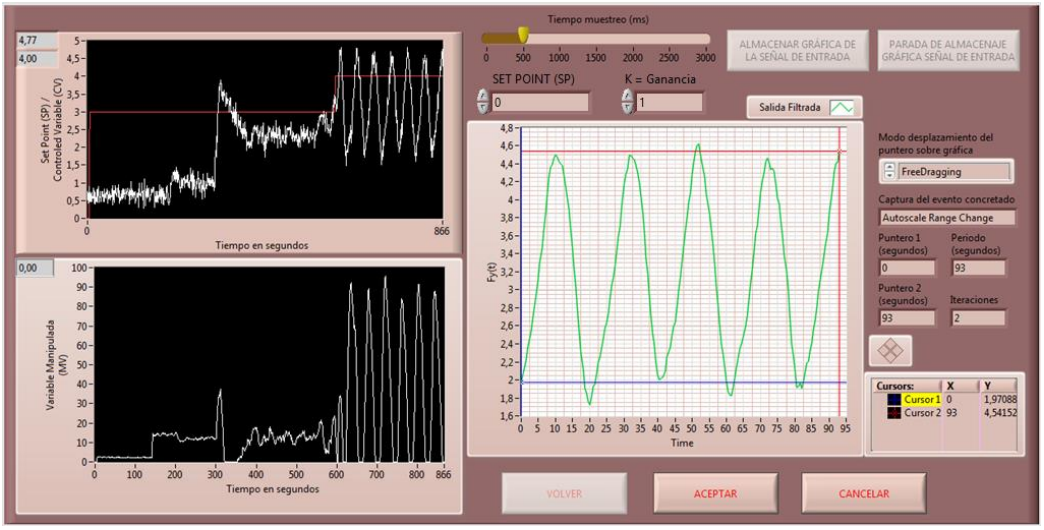


Figura 5.31. Obtención del período crítico tras finalizar el almacenamiento de los datos con su consecutiva obtención de la ganancia crítica.

Una vez conseguidos los dos parámetros de esta experimentación en lazo cerrado el usuario podrá observar las mismas acciones que visualizó en el caso de la experimentación en lazo abierto, calibración u obtención de las URLS del sistema; almacenaje de datos en el fichero XML y registro en la matriz dinámica de los mismos. Ello puede observarse en la siguiente imagen (véase figura 5.32.):

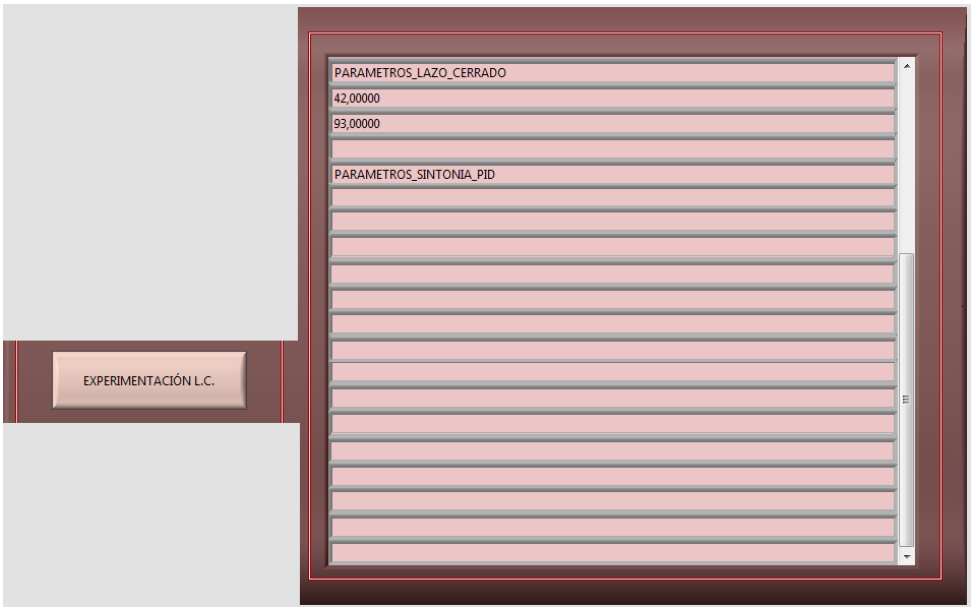


Figura 5.32. Actualización de los datos en la tabla tras la experimentación en lazo cerrado (L.C.)

Junto con el documento XML posterior, en donde también se reflejan los datos obtenidos de las tareas precedentes (figura 5.33.):

```
<?xml version="1.0" encoding="UTF-8"?>
- <INFORME>
    <FECHA>08_06_2018</FECHA>
    <HORA>14_53_42</HORA>
    <BY>Sfer4D_Corporation</BY>
  - <VARIABLE_COMUNICACION>
    - <VARIABLE_1>
        <TIPO_VARIABLE>Lectura</TIPO_VARIABLE>
        <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/ENTRADAS ANALOGICAS.In_Volts00</URL>
      </VARIABLE_1>
    - <VARIABLE_2>
        <TIPO_VARIABLE>Escritura</TIPO_VARIABLE>
        <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/SALIDAS ANALOGICAS.Out_%00</URL>
      </VARIABLE_2>
    </VARIABLE_COMUNICACION>
  - <PARAMETROS_CALIBRACION>
      <TIPO_CALIBRACION>0,00000</TIPO_CALIBRACION>
      <NOMBRE>Lineal</NOMBRE>
    - <PARAMETROS>
        <P0>-10,84623</P0>
        <P1>11,83073</P1>
        <P2/>
        <P3/>
        <P4/>
        <P5/>
      </PARAMETROS>
```

```
    </PARAMETROS_CALIBRACION>
  - <PARAMETROS_L_A>
      <TIEMPO_UNIDADES>Segundos</TIEMPO_UNIDADES>
      <TIEMPO_28_PORCIENTO>22,00000</TIEMPO_28_PORCIENTO>
      <TIEMPO_63_PORCIENTO>62,50000</TIEMPO_63_PORCIENTO>
      <K>0,44428</K>
      <D>1,75000</D>
      <TAU>60,75000</TAU>
    </PARAMETROS_L_A>
  - <PARAMETROS_L_C>
      <TIEMPO_UNIDADES>Segundos</TIEMPO_UNIDADES>
      <KC>42,00000</KC>
      <TC>93,00000</TC>
    </PARAMETROS_L_C>
  - <PARAMETROS_PID>
      <TIEMPO_UNIDADES>Minutos</TIEMPO_UNIDADES>
      <KP>E</KP>
      <TI>E</TI>
      <TD>E</TD>
    </PARAMETROS_PID>
  </INFORME>
```

Figura 5.33. Obtención del fichero XML con los datos de la experimentación en lazo cerrado (L.C.)

Por último, si el usuario está conforme con los datos que ha ido obteniendo, podrá acceder al bloque correspondiente al diseño del controlador PID cuya ventana emergente tras accionar su botón será la siguiente:

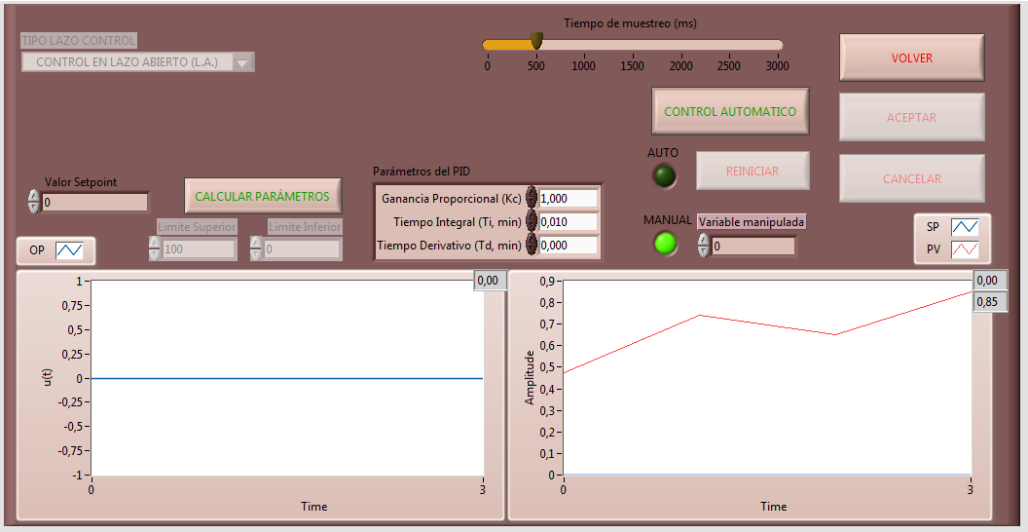


Figura 5.34. Interfaz usuario de la sintonía del controlador PID activado por defecto en modo manual.

Como en la imagen se menciona (Figura 5.34.) el programa iniciará el estudio con el modo manual activo, pudiendo, el usuario, cambiar al modo automático recogiendo los valores de las experimentaciones que ha realizado previamente, así como los datos entrantes del sensor de medida de presión ya calibrados para el posterior análisis que esta subrutina tiene por cometido:

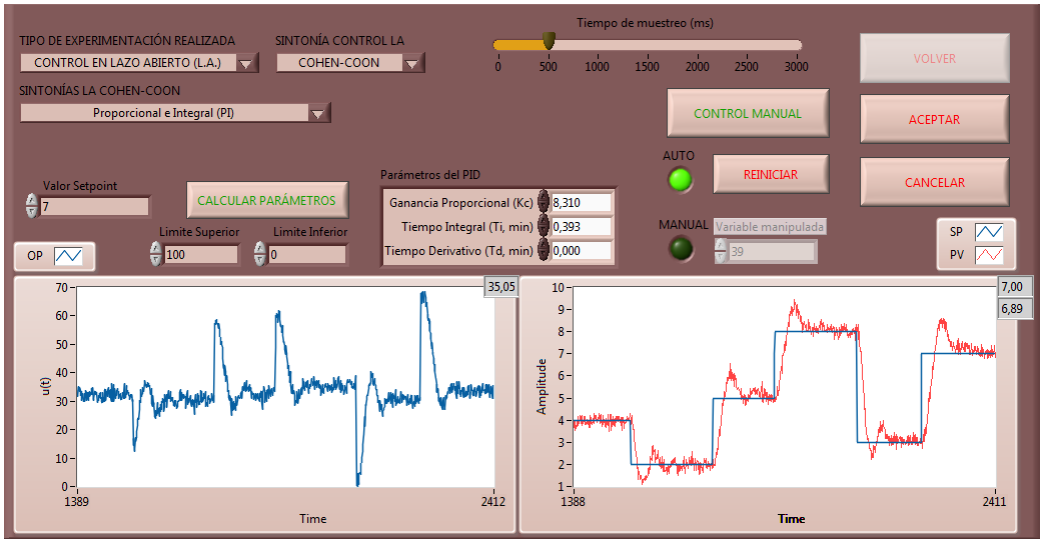


Figura 5.35. Generación de la respuesta controlada por el sintonizado PID.

El usuario podrá hacer varias iteraciones para poder sintonizar correctamente el controlador, resultando estas gráficas a lo largo del experimento y cuando haya decidido concluir con esta tarea, accionará el botón de aceptar, proporcionándose en la tabla actualizada, los datos que había calculado anteriormente y se verán a continuación:

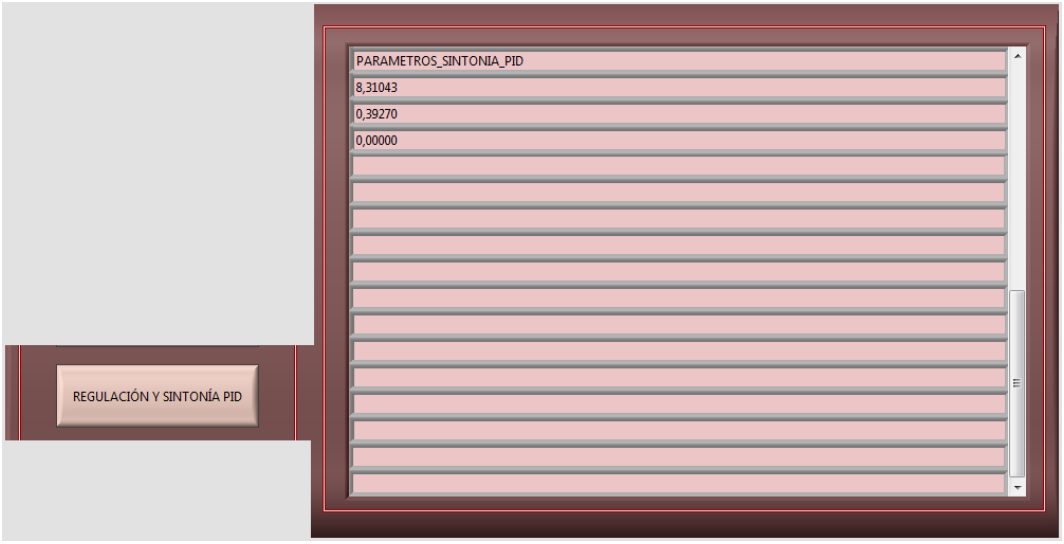


Figura 5.36. Actualización de los datos generados por el controlador PID.

Con los datos del fichero que serán los últimos datos que se incluirán en él:

```
<?xml version="1.0" encoding="UTF-8"?>
- <INFORME>
    <FECHA>08_06_2018</FECHA>
    <HORA>14_53_42</HORA>
    <BY>Sfer4D_Corporation</BY>
  - <VARIABLE_COMUNICACION>
    - <VARIABLE_1>
        <TIPO_VARIABLE>Lectura</TIPO_VARIABLE>
        <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/ENTRADAS ANALOGICAS.In_Volts00</URL>
      </VARIABLE_1>
    - <VARIABLE_2>
        <TIPO_VARIABLE>Escritura</TIPO_VARIABLE>
        <URL>opc://localhost/Servidor OPC para USB-1408FS-Plus/SALIDAS ANALOGICAS.Out_%00</URL>
      </VARIABLE_2>
    </VARIABLE_COMUNICACION>
  - <PARAMETROS_CALIBRACION>
      <TIPO_CALIBRACION>0,00000</TIPO_CALIBRACION>
      <NOMBRE>Lineal</NOMBRE>
    - <PARAMETROS>
        <P0>-10,84623</P0>
        <P1>11,83073</P1>
        <P2/>
        <P3/>
        <P4/>
        <P5/>
      </PARAMETROS>
    </PARAMETROS_CALIBRACION>
  - <PARAMETROS_L_A>
      <TIEMPO_UNIDADES>Segundos</TIEMPO_UNIDADES>
      <TIEMPO_28_PORCIENTO>22,00000</TIEMPO_28_PORCIENTO>
      <TIEMPO_63_PORCIENTO>62,50000</TIEMPO_63_PORCIENTO>
      <K>0,44428</K>
      <D>1,75000</D>
      <TAU>60,75000</TAU>
    </PARAMETROS_L_A>
  - <PARAMETROS_L_C>
      <TIEMPO_UNIDADES>Segundos</TIEMPO_UNIDADES>
      <KC>42,00000</KC>
      <TC>93,00000</TC>
    </PARAMETROS_L_C>
  - <PARAMETROS_PID>
      <TIEMPO_UNIDADES>Minutos</TIEMPO_UNIDADES>
      <KP>8,31043</KP>
      <TI>0,39270</TI>
      <TD>0,00000</TD>
    </PARAMETROS_PID>
</INFORME>
```

Figura 5.37. Obtención del fichero XML con los datos de la sintonía PID.

Una vez quede completado todo el fichero XML el usuario podrá modificarlo empleando la segunda parte del software en relación con la apertura del fichero introducido y lectura de los mismos para saber con qué información se podrá trabajar para continuar un estudio interrumpido o en el caso de quedar completado, modificar alguno de los datos obtenidos con los que no se tenga conformidad suficiente para concluir un posible estudio de diseño del controlador que automatice la planta analizada.

Si el usuario estuviera en disposición de modificar algún dato como se comentó líneas más arriba o en su defecto reanudar un estudio antiguo, podrá acceder a un panel similar al que se proporcionaba en el nuevo estudio en donde podrán verse los siguientes accionamientos (ver figura 5.38.):

Cabe mencionar que en el caso de que el usuario accione el botón de salir, volverá al menú principal pudiendo acceder a un estudio nuevo o uno antiguo del que había salido. Si no se ha registrado fichero alguno no podrá enviar ni almacenar ningún documento en internet.

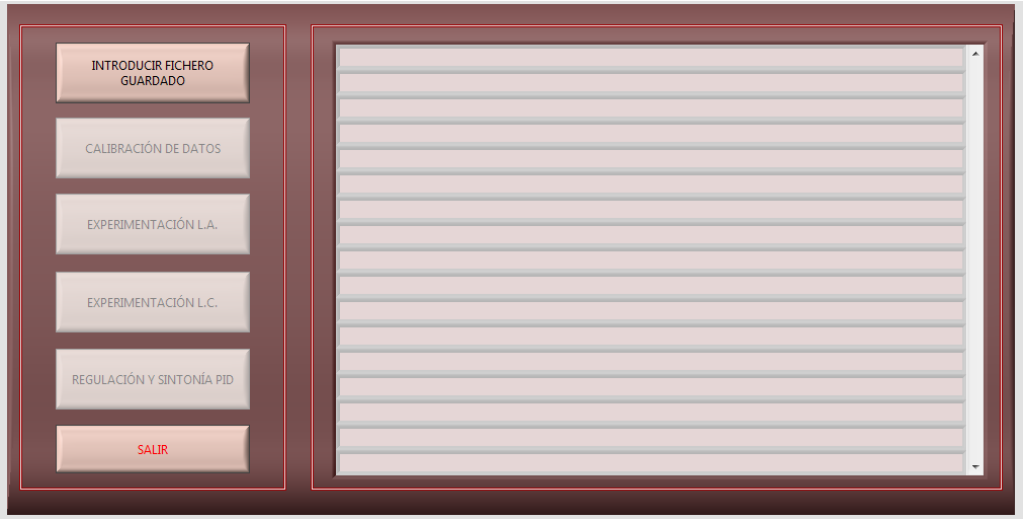


Figura 5.38. Panel de opciones de modificación de un antiguo estudio.

Ésta sería la interfaz propuesta para reanudar un estudio no concluido en donde se introducirá previamente un fichero XML que no se haya completado o se precise mejorar los resultados de un estudio anterior:

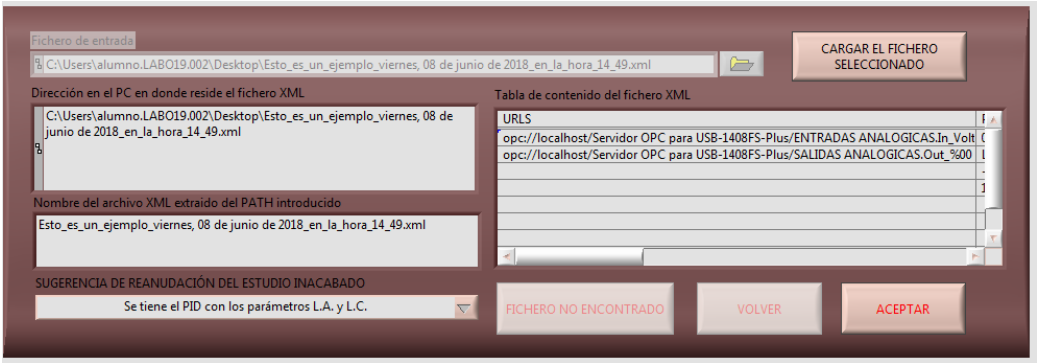


Figura 5.39. Panel de registro y lectura de los datos del fichero XML.

Si el usuario posee un fichero en cuestión (ver figura 5.39.) y lo introduce en la interfaz de este software, podrá ver como la subrutina lee el fichero y muestra su contenido en una tabla propuesta para que el usuario decida si proseguir con la mejora o reanudación de los datos antiguos de un fichero XML. Si decide aceptar los datos leídos y puestos en la tabla superior a la serie de botones de interacción, entonces el programa mostrará en la tabla adjunta (menú anterior Figura 5.38.) los datos cargados del fichero XML habilitando aquellas tareas que procedan según qué datos se tengan disponibles. En el ejemplo, el usuario ya había completado todas las tareas en un anterior estudio, de manera que todos los accionamientos han quedado habilitados, como puede observarse en la siguiente imagen (Figura 5.40.):

Figura 5.40. Panel de datos actualizados del fichero XML leído.

El resto de las tareas son idénticas a las expuestas anteriormente, de manera que, si el usuario decidiera salir de este menú y tomara la decisión de guardar el fichero podrá acceder al envío o almacenaje en internet. Los datos que se modifiquen o se obtengan de las tareas que se proponen tendrán la misma interfaz que lo que se expuso anteriormente, de manera que aquí quedaría concluida la parte del segundo estudio, en referencia a modificación de la información cargada del fichero XML.

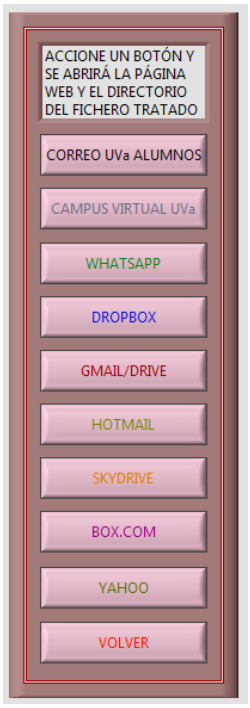


Figura 5.41. Panel de los accionamientos para enviar o guardar el fichero XML en la plataforma virtual habilitada de internet.

El usuario tendrá la posibilidad de poder acceder a una de las diversas plataformas virtuales que se ofrecen (ver figura 5.41.) con el fin argumentado en la última imagen expuesta.

En este ejemplo, el usuario (ver figura 5.42.) seleccionó la plataforma de envío y almacenamiento; "Gmail", en cuya imagen siguiente se puede ver la interfaz interactiva de dicha página para poder acceder a la propia cuenta con el fin de enviar a una segunda persona el informe XML de los datos generados por la planta de proceso o simplemente guadarlo en la nube; el correspondiente "Google Drive":

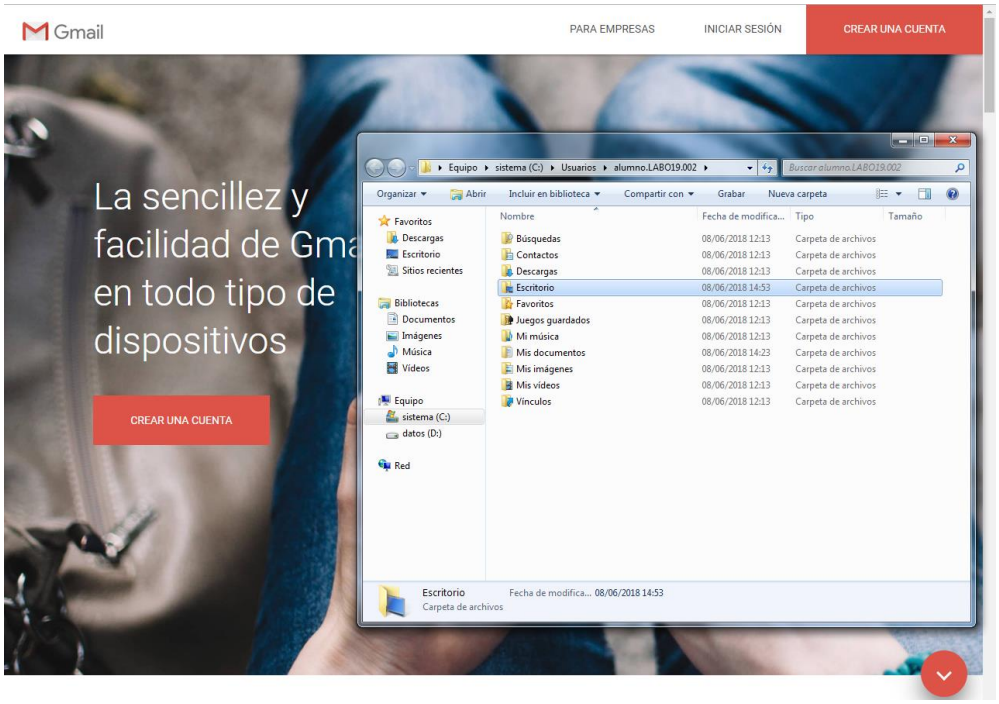


Figura 5.42. Página web generada tras elegir la plataforma señalada y carpeta cargada en donde reside el fichero XML guardado en Windows.

La página correspondiente con el directorio en donde se encunetra alojado el fichero XML se abre al mismo tiempo que la página web mencionada con el fin de poder sincronizar el movimiento del fichero del explorador de Windows a la zona virtual de internet.

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

VI – CONCLUSIONES Y LINEAS FUTURAS

TRABAJO FIN DE GRADO – KEN VERA CHAN
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

6. CONCLUSIONES Y LÍNEAS FUTURAS

En las conclusiones que se han podido extraer de este proyecto se pueden clasificar en las expuestas a continuación:

- El software base LabVIEW sobre la cual se levantan programas con una sólida arquitectura de control y simulación de procesos, permite a programadores diseñar aplicaciones que requieran pruebas, medidas y control con acceso rápido a la información proporcionada por los datos y hardware.

- En entornos reales, ofrece un enfoque de programación gráfica que ayuda a visualizar cada aspecto de la aplicación creada, incluyendo, como se ha comentado, configuración de hardware, datos de medidas y depuración.

  Esta visualización hace que sea más fácil integrar hardware de medidas de cualquier proveedor, representar una lógica compleja en el diagrama, desarrollar algoritmos de análisis de datos y diseñar interfaces de usuario personalizadas.

- En el presente proyecto de ingeniería, gracias al potencial del entorno de programación de LabVIEW se ha podido programar un sólido software de diseño de controladores PID en base a los fundamentos teóricos de sistemas control automático.

- La categoría de la comunicación mediante servidores OPC en la industria permitiendo establecer una conexión comunicativa estandarizada entre clientes y servidores OPC solucionando el clásico problema de los drivers propietarios. Ello permite intercambiar con bastante libertad y de manera sencilla la información entre dispositivos y aplicaciones de cualquier tipo, como por ejemplo soluciones de HMI (Human Machine Interface), planillas de cálculo, motores de base de datos, ERPs, entre otras.

Se proponen las siguientes líneas futuras de investigación para aquellos interesados en la teoría de diseño de sistemas de control de forma aplicativa:

  - Inclusión de controles predictivos, los cuales son otro tipo de controladores similares a los conocidos PID, pero en vez de basarse en la experiencia como lo hacen estos últimos, emplean un sistema predictivo de posibles respuestas que daría el sistema conociendo un

punto de referencia y estacionario, ya que el control predictivo integra control óptimo, control de procesos con tiempos muertos, procesos multivariables y utiliza las referencias futuras cuando están disponibles. Al utilizar una estrategia con horizonte de control finito permite la consideración de restricciones y procesos no lineales.

- Inclusión de otros métodos de sintonía. Como se ha visto, lo métodos de sintonía de controladores PID empleados en este proyecto son los más utilizados en sistemas que puedan modelizarse con un sistema de primer orden más retardo, sin embargo, existen otros métodos de sintonía que son ajustes empíricos, de entre ellos el método de sintonía del relé, diseñado por Aström y Hägglund Hägglund en 1984. Otros se basan en criterios integrales con cambio en la referencia, por ejemplo el método: Kaya y Scheib (1988) o el propio Aström y Hägglund Hägglund (1984), mencionado antes y que depende de lo obtenido cuantitativamente del método del relé, cuyo criterio es emplear el margen de fase y de ganancia para su sintonía.

- Incorporación directa de modelos en función de transferencia, los cuales puede ser de orden mayor, en donde el diseño de los controladores exige algoritmos más complejos debido a su correspondiente estructura representativa del sistema.

- Actualización para soporte OPC UA que es más moderna que la tecnología OPC empleada en este proyecto. Recuérdese que se empleaba un servidor OPC DA, de acceso de datos. A diferencia del empleado en el funcionamiento del presente software, el driver cliente OPC-UA proporciona un túnel de comunicaciones entre dos o más equipos, permitiendo que los datos sean transferidos con seguridad, sin necesidad de requerir una conexión VPN y de manera muy fiable.

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

VII – BIBLIOGRAFÍA

TRABAJO FIN DE GRADO – KEN VERA CHAN
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

7. BIBLIOGRAFÍA

[1] Área de programación y desarrollo – Manual XML
"XML ¿Qué es?"
Enlace proporcionado:
http://mundolinux.info/que-es-xml.htm

[2] Centro Integrado Politécnico ETI (2007);
"OPC: un estándar en las redes industriales y buses de campo"
Enlace proporcionado:
www.etitudela.com/entrenadorcomunicaciones/downloads/labviewintroduccionopcserver.pdf

[3] Martín-Romo, Miguel (2010): "Controladores PID y controladores PID modificados". Madrid 2010.
Editorial Pearson: 398-641

[4] Matrikon, Honeywell International Inc. (2018);
"¿Qué es un servidor OPC?"
Enlace proporcionado:
https://www.matrikonopc.es/opc-servidor/index.aspx

[5] Morilla García, Fernando (2005) Controladores PID.
Dpto. de Informática y Automática Escuela Técnica Superior de Ingenieros Informáticos,
UNED Madrid 2006.

[6] National Instruments – LabVIEW (2018)
"¿Por qué escoger una NI DAQ? - Construido para Durar, Diseñado para Ejecutar"
Enlace proporcionado:
http://www.ni.com/data-acquisition/what-is/esa/

[7] National Instruments – LabVIEW (2018)
"¿Qué es adquisición de datos?"
Enlace proporcionado:
http://www.ni.com/data-acquisition/what-is/esa/

[8] National Instruments – LabVIEW (2018)
"¿Qué es LabVIEW?"
Enlace proporcionado:
http://www.ni.com/es-es/shop/labview.html

[9] Ogata, Katsuhiko (1974) Ingeniería de control moderna. "Análisis y diseño de sistemas de control por el método de la respuesta en frecuencia".

[10] Pybonacci | Python y Ciencia (2017)
"Conceptos lazo abierto y lazo cerrado.
Enlace proporcionado:
https://www.pybonacci.org/2013/11/06/teoria-de-control-en-python-con-scipy-ii-control-pid/

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

VIII – INDICE DE FIGURAS

TRABAJO FIN DE GRADO – KEN VERA CHAN
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

8. INDICE DE FIGURAS

Software de laboratorio para el diseño, implementación y operación de controladores PIDs

IX – ANEXOS

TRABAJO FIN DE GRADO – KEN VERA CHAN
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

INDICE DE CONTENIDOS:

9. ANEXOS:

En lo referente al hardware que se ha empleado en esta planta de procesos, a continuación, se expondrán los datasheet de cada componente y su funcionalidad complementaria con el diseñado software.

9.1. ANEXO A: DATASHEET MAGNETOTÉRMICO 2 POLOS 20A K60N:

Referencia: A9K17220

Descripción:
Interruptor Automático
magnetotérmico
2 polos 20A K60N

Características:
Tensión de empleo 230/400V CA.

Curva de disparo: curva C.

Temperatura de referencia: 30ºC

Conexión: bornes de caja para cables rígidos de hasta 25 mm^2

Poder de corte: 6000ª

Ancho por polo: 2 pasos de 9 mm

Fabricante del producto: Merling Gerin

9.2. ANEXO B: DATASHEET BOMBA HIDRÁULICA FLOJET 02100-232:

FLOJET
a xylem brand

RLF122201D *12v dc*
RLF222201D *24v dc*
RLF122202D *12v dc*
RLF222202D *24v dc*
Compact Self-Priming Diaphragm Pump

- Built-in pressure switch automatically starts and stops pump instantaneously when discharge valve is opened and closed.
- Intermittent duty
- Self-priming capability up to 0.7m. Pump may be located above the liquid level.
- Maximum liquid temperature 43ºC (110ºF)
- Duplex diaphragm design allows extended dry running without damage.
- Built in thermal overload protection
- Low amp draw for battery powered applications

CONNECTIONS: 9.5mm (3/8") hose barb.

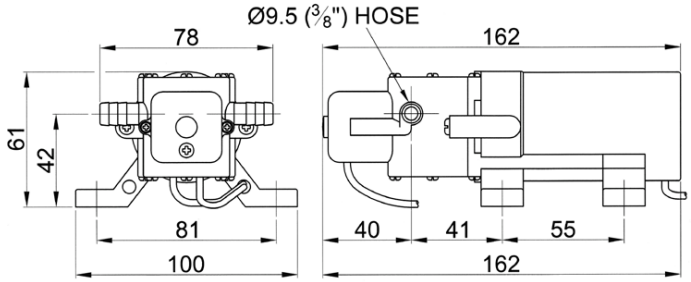
PRESSURE SWITCH SETTINGS:
Cuts in at 1.7 bar (25 psi)
Cuts out at 2.4 bar (35 psi)

MATERIALS OF CONSTRUCTION:

Pump head:	Glass-filled Polypropylene
Diaphragm:	Santoprene®
Valves:	***RLF122201D*** *12v dc* = Viton
	RLF222201D *24v dc* = Viton
	RLF122202D *12v dc* = EPDM
	RLF222202D *24v dc* = EPDM

OUTPUT:
Up to 3.8 litres/minute (0.8 gallons/minute).
Max recommended total head 25 metres

FUSE SIZE: 5 amp **WEIGHT:** 0.72 Kg



9.3. ANEXO C: DATASHEET ACCIONAMIENTO ELÉCTRICO HIDMM4000-ES-0838-I427-0-A4:

INSTRUCCIONES
DMM-4000

Transmisor 2 x 4-20 mA y RS-485 Modbus de variables analógicas para montaje en campo

Desin Instruments
desin@desin.com
www.desin.com

Se ruega leer estas instrucciones antes de manipular el aparato

CE 0838 I427-0

1. INTRODUCCIÓN

Esta **Hoja de Instrucciones** explica, en modo abreviado, el uso las partes comunes de los modelos genéricos y de las versiones específicas programadas en fábrica de la serie DMM según características definidas en su Hoja Técnica.

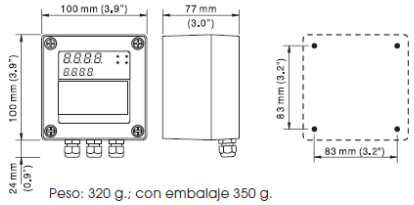
Las Instrucciones específicas de Utilización y Configuración de cada versión se suministran en Manual de Usuario aparte editados en pdf que pueden encontrarse en **www.desin.com** y que son actualizadas periódicamente.

MODELOS ESTÁNDAR

Transmisores 4-20 mA a 4 hilos, para captación, cálculo y transmisión de variables de campo o calculadas, en forma de señales estandard de 4-20 mA y digitalmente por RS-485. Apropiados para la introducción en PLC y PC de variables analógicas de TP, Pt100, mV, mA, Pulsos de RPM, Contadores, Humedad, Sensores químicos, Nivel, Presión, Caudal, Vibración, etc.

2. INSTALACIÓN

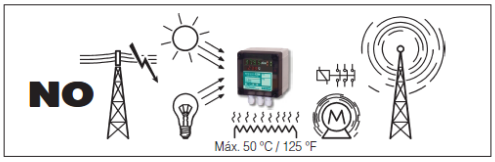
2.1 MEDIDAS MONTAJE Y ORIFICIOS EN PANEL



Peso: 320 g.; con embalaje 350 g.
Opcionalmente puede suministrarse soporte inoxidable con brida para conducciones de 2". Peso: añadir 120 g

2.2 PRECAUCIONES DE MONTAJE

- Impedir focos intensos de luz frontalmente.
- Montar lejos de vapores corrosivos, humedad, temperaturas mayores de 50 °C / 125 °F, vibraciones, etc.
- Alejar de focos de radiación electromagnéticos, radiofrecuencia, microondas, alta tensión, etc.



2.3 MONTAJE

MONTAJE EN PARED

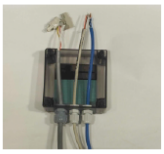
- Abrir el equipo quitando los 4 tornillos situados en las 4 esquinas del frontal
- Separar el frontal con cuidado
- Manejar sólo la caja vacía para marcar la fijaciones en la pared
- Pueden utilizarse 2 o 4 fijaciones dependiendo del tipo de pared
- Marcar en la pared a través de los agujeros los orificios del anclaje
- Taladrar la pared con broca de 5 mm en una profundidad mín de 20 mm
- Introducir tacos de 5 mm y fijar la caja con los tornillos que se adjuntan.

ACCESO A BORNES

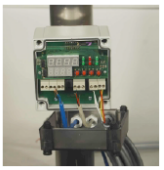
- Con el frontal retirado, los bornes de conexión quedan accesibles.
- Quitar los conectores desenchufables de los bornes del módulo marcándolos con rotulador para recordar su ubicación.
- Todos los modelos de esta serie disponen de 1 fila de 16 bornes para cable 1,5 mm de sección máximo.
- Pasar los cables por los prensaestopas del frontal dejando 20 cm. fuera para facilitar la conexión.

Atención: no apretar los prensaestopas hasta tener el frontal puesto de nuevo

- Conectar los cables a los conectores desenchufables según los diagramas de cada modelo.
- Acomodar los cables en el espacio vacío al lado de los bornes.
- Estirar los cables desde el exterior hasta tener el frontal situado de nuevo en su alojamiento.
- Cerrar el instrumento apretando los cuatro tornillos del frontal.
- Apretar los prensaestopas con el fin de estanqueizar el equipo.



MUY IMPORTANTE: Para que el instrumento cumpla la protección IP65, el frontal debe estar siempre completamente cerrado y los prensaestopas apretados sin que quede resquicio alguno.



MONTAJE EN TUBERÍA HASTA 2"

Bajo demanda se suministra una brida inox. para montaje en tuberías hasta 2" en horizontal o vertical. Se compone de una pletina de inox que ha de montarse detras del equipo fijándola con los 4 tornillos allen que se acompañan, y de una brida ajustable introducida en las ranuras que dispone.

- Con el frontal retirado, montar la pletina detrás, en los 4 orificios del equipo, poniendo los tornillos en el exterior y las tuercas de fijacion en el interior a través de los orificios del equipo. Apretar los tornillos con una llave allen.
- Situar la pletina en la posición de la tubería.
- Comprobar que la brida ajustable es del tamaño adecuado y pasarla alrededor de la tubería hasta su tornillo de sujeción.
- Apretar el tornillo de la brida hasta que el equipo quede bien sujeto.
- Hacer el conexionado como se ha indicado y cerrar el frontal.

3. CONEXIONADO

MUY IMPORTANTE: Estos aparatos cumplen CE y están protegidos contra sobretensiones. No obstante, para evitar que a pesar de todo, puedan ser afectados por parásitos de gran magnitud, es recomendable seguir las siguientes precauciones de conexionado.

3.1 PRECAUCIONES GENERALES

Antes de conectar la red, o las entradas y salidas, examinar bien los datos de la etiqueta de características y respetar las recomendaciones de instalación.

NOTA: Una instalación inadecuada, permitirá que le lleguen parásitos de la red que, dependiendo de su importancia, pueden llegar a bloquear el µP presentando **[Erro]**. Para su protección, incorporan un dispositivo de seguridad Watch-Dog que impide que el µP quede bloqueado, restableciendo su funcionamiento inmediatamente después del transitorio.

IMPORTANTE: Parásitos de gran intensidad pueden producir parpadeos en algunos dígitos del display. Esto no afecta al funcionamiento del instrumento, que continuará trabajando normalmente, volviendo a visualizar correctamente por si mismo o pulsando cualquier tecla del frontal.

3.2 PRECAUCIONES DE CONEXIONADO

Antes de conectar la red, o las entradas y salidas, examinar bien los datos de la etiqueta de características y respetar las recomendaciones de instalación.

NOTA: Una instalación inadecuada, permitirá que le lleguen parásitos de la red que, dependiendo de su importancia, pueden llegar a bloquear el µP presentando **[Erro]**. Para su protección, incorporan un dispositivo de seguridad Watch-Dog que impide que el µP quede bloqueado, restableciendo su funcionamiento inmediatamente después del transitorio.

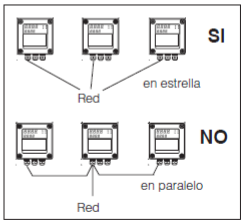
IMPORTANTE: Parásitos de gran intensidad pueden producir parpadeos en algunos dígitos del display. Esto no afecta al funcionamiento del instrumento, que continuará trabajando normalmente, normalizándose por si mismo.

CONEXIÓN A LA RED

La alimentación debe ser lo más directa posible desde la acometida general de la zona, con distribución en estrella, (evitar conectar desde otro dispositivo).

Evitar la alimentación de contactores, en la misma línea que los instrumentos.

En el caso de una red muy perturbada (debido a unidades de potencia, tiristores por ejemplo), alimentar la parte de instrumentación por medio de un transformador de aislamiento, con la pantalla unida a tierra.



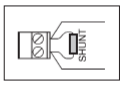
ENTRADAS ANALÓGICAS DE MEDIDA

Separar físicamente en todo el recorrido, las líneas de señal (mV, Pt 100, 4-20 mA) de las líneas de potencia o de mando de contactores, servomotores, actuadores, etc. (Utilizar bandejas o conducciones independientes).

- Para grandes longitudes de cable de señal, utilizar cables con hilos trenzados y apantallados.

IMPORTANTE: Las pantallas de los cables de señal deben estar unidas a tierra en un solo punto y en el lado de la recepción de la señal, es decir, en una toma de tierra cerca del instrumento receptor.

- **Entrada Termopar:** Usar cable de extensión o compensación del mismo **TP** HASTA LOS MISMOS BORNES DEL APARATO, observando su polaridad.
- **Entrada Pt100:** Usar cable de 3 hilos para compensar las influencias de las resistencias del cable de cobre (sección 1,5 mm² ó galga AWG 15).
- **Entrada mV:** Usar cable de cobre-cobre de 1,5 mm² o galga AWG 15 de sección. Respetar la polaridad.
- **Entrada mA:** Id. entrada mV añadiendo en paralelo con los bornes el shunt de 3,74 Ω que se adjunta.



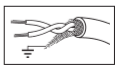
ENTRADAS LÓGICAS Y FRECUENCIA

- Si se utilizan como entrada de contactos de relé, es recomendable poner un condensador de 10 nF a 100 nF en paralelo para eliminar los rebotes.

ATENCIÓN: Este condensador ha de ser retirado cuando entren señales de transistor, Namur, etc. pues provocaría deformaciones del flanco y retardos.

SALIDAS ANALÓGICAS 4-20 mA

- Es recomendable utilizar cable trenzado apantallado, uniendo el blindaje a tierra como antes se ha explicado.

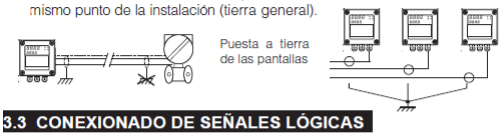


SALIDA DE 24 VDC DE ALIMENTACIÓN AUXILIAR

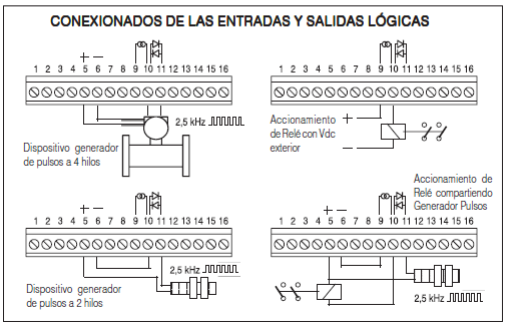
- La serie **DS-4000 y FMC-4000** dispone de una salida 24 Vdc de 40 mA máx. para alimentar dispositivos externos propios de las versiones específicas.

TOMA DE TIERRA

- Todas las mallas de los cables apantallados deben unirse en estrella en un mismo punto de la instalación (tierra general).



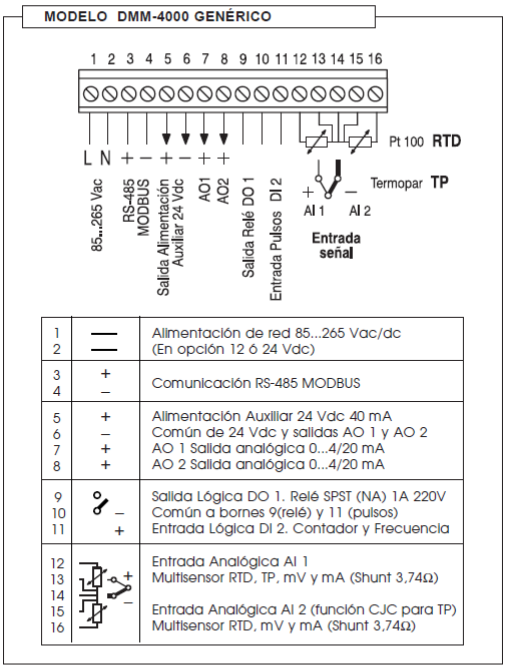
3.3 CONEXIONADO DE SEÑALES LÓGICAS



3.4 DIAGRAMA DE CONEXIONADO

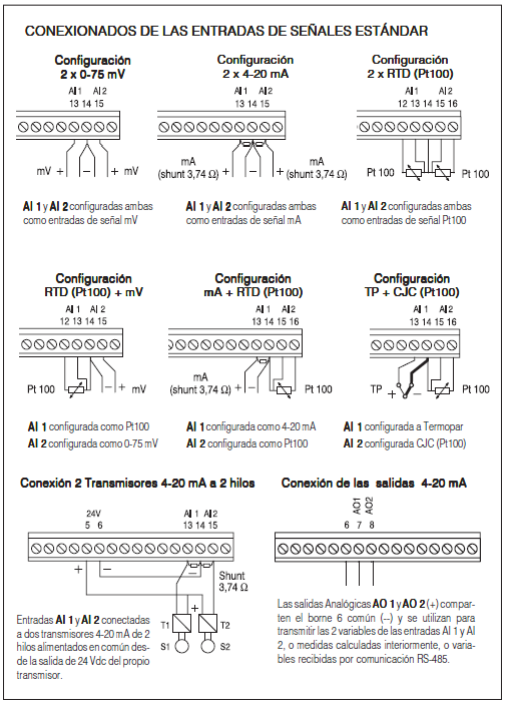
El esquema de conexionado, los datos técnicos de escala, entradas y salidas, vienen indicados por un adhesivo en la parte lateral de la caja.

MUY IMPORTANTE: Comprobar que la señal a medir y su rango son las mismas que se indican en la etiqueta adhesiva de características.



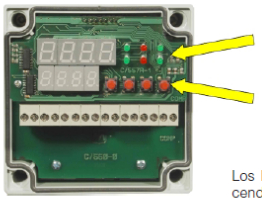
1 2	—	Alimentación de red 85...265 Vac/dc (En opción 12 ó 24 Vdc)
3 4	+ –	Comunicación RS-485 MODBUS
5 6 7 8	+ – + +	Alimentación Auxiliar 24 Vdc 40 mA Común de 24 Vdc y salidas AO 1 y AO 2 AO 1 Salida analógica 0...4/20 mA AO 2 Salida analógica 0...4/20 mA
9 10 11	– +	Salida Lógica DO 1. Relé SPST (NA) 1A 220V Común a bornes 9(relé) y 11 (pulsos) Entrada Lógica DI 2. Contador y Frecuencia
12 13 14 15 16	+ –	Entrada Analógica AI 1 Multisensor RTD, TP, mV y mA (Shunt 3,74Ω) Entrada Analógica AI 2 (función CJC para TP) Multisensor RTD, mV y mA (Shunt 3,74Ω)

3.5 CONEXIONADO DE SEÑALES ANALÓGICAS



4. DESCRIPCIÓN DEL TRANSMISOR

Estos equipos son Transmisores de campo, por cual, excepto en la puesta en marcha, no precisan ser manipulados habitualmente. No obstante, para facilitar ciertas tareas dispone de un display interno con un teclado (no accesible desde el exterior) para acceder manualmente a los diferentes parámetros de configuración en el caso de no poder hacerlo por RS-485.



Leds de estado (en el mismo orden)
L1 Y1 F1
L2 Y2 F2

Teclado interno (en el mismo orden)
↺ ⌃ ⟨ ↵

Los Led de estado, cuando están encendidos, muestran normalmente:

L1 = Pantalla 1: Medida principal
L2 = Pantalla 2: Medida auxiliar
Y1 = Relé de alarma activado
Y2 = Sólo en funciones espciales (ver Instrucciones Específicas)
F1 = Parpadea cuando se autocalibra (ver Calibración de Usuario)
F2 = Se utiliza según la función que se haya instalado en cada versión

NOTA: Estos leds, según aplicaciones, pueden presentar diferencias de función que en cada caso serán indicadas en las Instrucciones Específicas de la versión.

4.1 DESCRIPCIÓN DE LA RESOLUCIÓN DEL DISPLAY

Los transmisores **DMM-4000**, en la versión genérica o en sus versiones específicas presentan en el display las medidas de las entradas analógicas, o variables calculadas (p.e. Punto de Rocío, Caudal, Nivel, etc.). Estas medidas se presentan siempre con un rango entre -1999 y 9999, con posibilidad de definir hasta tres decimales.
Sin embargo, en su interior, el Transmisor realiza las conversiones A/D con resolución de 64000 puntos. Los calculos aritméticos los realiza en «coma flotante» ofreciendo una altísima precisión. Las acumulaciones, aunque sólo son visibles hasta 9999 puntos, alcanzan 2^{32} unidades en registros internos.

Si se desea acceder a estas medidas calculadas en alta precisión en el interior deben ser leidas por comunicación RS-485. (ver apartado de Comunicación)

4.2 DESCRIPCIÓN DEL INTERIOR DEL DISPLAY

Abriendo el transmisor se puede acceder al teclado del display con los pilotos Led, que están preprogramadas de fábrica para acceder a los datos específicos de la versión del **DMM-4000** suministrado. Las acciones posibles desde el teclado son la Parametrización, Configuración y Calibración de Usuario, etc.

DISPLAY SUPERIOR
Presenta el valor de la Variable Principal predefinida.
En MENÚ presenta el valor de los parámetros u otras opciones.

DISPLAY INFERIOR
Presenta el valor de la Variable Secundaria predefinida.
En MENÚ presenta el mnemónico del parámetro u otras opciones.

LEDS L1 L2 Indicadores de Pantalla
Indican el número de Pantalla presente en el display en ese momento. Normalmente el Transmisor presenta sólo los datos de la pantalla L1.

LEDS Y1 Y2 Indicadores de Alarmas
Indican el estado de las Alarmas (ver Manual Específico de cada Versión).

LEDS F1 y F2 Indicadores especiales
Indican el estado de funciones especiales (ver Manual Específico de cada Versión).

TECLADO INTERNO
El transmisor **DMM-4000** dispone de un teclado interno, accesible sólo retirando la cubierta del equipo, para configurar las funciones de cada versión:

↺ Tecla de **FUNCIÓN**
Permite desplazarse por todas las funciones habilitadas en el MENÚ.
Permite salir de un parámetro sin que el instrumento guarde el cambio realizado.

⌃ Tecla **INCREMENTO**
Permite modificar datos, incrementando el valor del dígito parpadeando.
En otras funciones cambia el estado de la opción, si ésta lo permite.

⟨ Tecla **DESPLAZAMIENTO**
En edición permite seleccionar el dígito a modificar (parpadeando), desplazándose hacia la izquierda uno a uno. También actúa como **Decrementación.**

↵ Tecla **VALIDACIÓN**
Sirve para entrar en una función o un parámetro del MENÚ.
Después de modificar un parámetro, guarda los cambios introducidos.

5. MODO DE EMPLEO

Estos transmisores disponen de dos pantallas de presentación de datos L1 y L2, de las que sólo la L1 está habilitada. En ciertas versiones específicas, pueden haber presentes en L2 otros datos preasignados de origen.

Para leerlos pulsar la tecla ⟨ encendiéndose el led L2 correspondiente.

5.1 VISUALIZACIÓN INICIAL DEL DISPLAY

Al alimentar el Transmisor, aparece **[Self] [tESt]** mientras se autocomprueba. Después muestra **[XXXX] [vErS]** que es el código de identificación del modelo. A continuación muestra **[XXXX] [rEV]** código de la versión del firmware. Después de estos datos, pasa mostrar los valores asignados al **Lazo 1**.

ÁRBOL DEL MENÚ PRINCIPAL
Presenta las funciones de la versión genérica o de las versiones específicas, como SP, Alarmas, Constantes Str, etc. El resto de funciones pueden variar con cada versión del **DMM-4000** (ver Manual Específico de cada Versión).
NOTA: El árbol es de desplazamiento descendente y cíclico.

5.2 MANEJO DEL MENÚ

Mediante este MENÚ se accede a la configuración y a otros SUBMENÚS.

ENTRADA EN LAS OPCIONES
Se utiliza la tecla ↺ pulsándola repetidamente hasta ver la opción deseada.
Para modificarla se utilizan las teclas ⌃, ⟨ y ↵ para validar los cambios.

SALIDA DE LAS OPCIONES
Para salir de cualquier opción pulsar repetidamente la tecla ↺

5.3 PRESELECCIÓN DE CONSIGNA PRINCIPAL [SP]

Sólo en caso de la versión disponga de esta función.

MODIFICACIÓN DE LAS CONSIGNAS SP
La serie **DMM-4000** dispone de dos modos de acceso a modificar **SP**:

Modo Directo: Pulsar ↺ y el display inferior **[SP]** empezará a parpadear. Aumentar o disminuir su valor con ⌃ o ⟨ y volver pulsando ↺ .

Modo por Desplazamiento: Pulsar ↺ dos veces y el display mostrará **[SP x]** (x = número de pantalla). Pulsar ↵ y el dígito de las unidades parpadeará.
Seleccionar dígito a dígito con ⟨ e incrementarlos de 0 al 9 con ⌃.
Una vez modificado validar con ↵. Salir pulsando ↺ o pasar a otra opción.
Si el valor introducido supera los límites predefinidos, no permite validarlo.

5.4 PRESELECCIÓN DE ALARMAS [AL1] y [AL2]

Sólo en caso de la versión disponga de esta función.
El modelo genérico dispone de una Alarma **AL 1** que normalmente está asignada al relé **Y1**. Algunas versiones disponen de otra alarma ***AL 2*** asociada al relé ***Y2***.

Modelo genérico: Alarma AL1 de Máximo independiente
Versiones específicas: Alarmas configuradas en función de cada versión

MODIFICACIÓN DE ALARMAS AL1 (y AL2 en opción)
Pulsar ↺ varias veces hasta que el display muestre **[AL x]** (x = nº Alarma). Pulsar ↵ y aparecerá **[SPAx]** y su valor. Otra vez ↵ y parpadeará el primer dígito. Seleccionar dígito a dígito con ⟨ e incrementarlos de 0 al 9 con ⌃.
Una vez modificado, validar con ↵. Salir pulsando ↺ o pasar a otra opción.
Si el valor introducido supera los límites predefinidos, no permite validarlo.

MODIFICACIÓN DE HISTÉRESIS HY x
Entrar en la **Alarma** a modificar **HY como se ha explicado en modificar Alarmas**. Después de **[SPAx]** pulsar ↺ hasta que ver **[HY x]** y modificar la histéresis.

5.5 PARÁMETROS CONFIGURABLES

Algunas versiones específicas del **DMM-4000** disponen de celdas para introducir datos para adaptar el funcionamiento del transmisor al entorno de aplicación (p.e. en las versiones de nivel permite introducir los datos del tanque, densidad y unidades para calcular el volumen, o en la versión de caudal permite introducir los datos de sección, unidades, tiempo base y constantes del fluido para calcular el caudal desde la medida de velocidad). Estos datos deben ser introducidos en las consignas auxiliares Str 1..2..3 y 4.

MODIFICACIÓN DE LAS CONSIGNAS AUXILIARES Str 1..2..3..4
Sólo en caso de la versión disponga de estas funciones.
Permiten ver y modificar los valores de las cuatro Consignas Auxiliares, introducidos por teclado como constantes de cálculo.

El margen máximo introducible por teclado es -1999 a 9999. Si se hace por comunicación RS-485 permite ±32000, en ambos modos con 1 a 3 decimales.

Estos parámetros están protegidos por una clave de acceso para evitar manipulaciones indebidas. Para modificar las consignas auxiliares Str 1, Str 2, Str 3 y Str 4 hay que entrar en el submenú de Congiguración como se indica en el apartado CONFIGURACIÓN. Una vez dentro de Configuración:

Pulsar ↺ varias veces hasta que el display muestre **Str 1, 2, 3 ó 4**
Pulsar ↵ y aparecerá **[Str X]** y su valor. Otra vez ↵ y parpadeará el primer dígito. Seleccionar dígito a dígito con ⟨ e incrementarlos de 0 al 9 con ⌃.
Una vez modificado, validar con ↵. Si el valor introducido supera los límites predefinidos, no permite validarlo. Salir pulsando ↺.

5.6 FILTROS DE AMORTIGUACIÓN (DAMPING)

Permiten amortiguar las oscilaciones y puntas que se producen en la medida.

Filtro de Amortiguación: Amortigua las oscilaciones de la medida.

Filtro de Picos: Elimina las picos de señal que pueda recibir el transmisor.

Ambos se encuentran entrando en el parámetro **P03** de **AI 1** y **AI 2**.

Amortiguar: con ⩔ y ⩓ el dígito de centenas [_0_ _] mín. y [_9_ _] máx.

Filtro Picos: con ⩔ y ⩓ el dígito de decenas [_ _0_] mín. y [_ _6_] máx.

6. SALIDAS DE SEÑAL ANALÓGICAS Y LÓGICAS

Salidas Analógicas: El DMM dispone de dos salidas 4-20 mA, salidas AO1 y AO2 que normalmente retransmiten las variables de las entradas AI 1 y AI 2. En algunas versiones transmiten variables calculadas (p.e. Caudal, Punto de Rocío, etc.). Los rangos se definen en los parámetros **P3** y **P4** de **AO1** y **AO2**.

Alarma: Visible y preseleccionable por teclado o por RS-485. Por teclado se muestra al pulsar la tecla 🔗, el display mostrará **AL 1** con el valor actual.

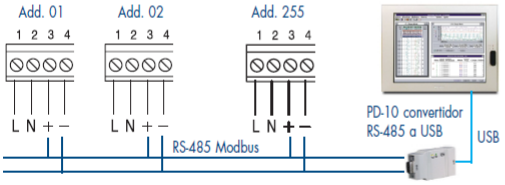
7. COMUNICACIONES

Todos los Caudalímetros series **DMM y RDT** disponen de **RS-485 Modbus** en modo RTU para comunicar los datos de campo. Ver Manual de Modbus.

7.1 COMUNICACIÓN RS-485 DEL DMM-4000

La serie **DMM-4000** dispone de un puerto para conexión por cable RS-485.

7.2 ESQUEMA DE CONEXIONADO PARA CABLE



7.3 PARÁMETROS DE COMUNICACIÓN

Para comunicar estos transmisores con una red de bus de campo deben tenerse en cuenta los siguientes parámetros:

Velocidad de transmisión (Baud-Rate): Seleccionable por teclado en menú Configuración entre 3 valores: 9600, 19200 y 38400 bauds. De fábrica se suministra a 38400 baud salvo que se haya especificado otra velocidad.

Dirección de dispositivo (Address): Seleccionable por teclado en menú de Configuración en hexadecimal, desde h1 a h255 (FF). De fábrica se suministra con dirección 0001 salvo que se haya especificado otra dirección.

Para más información ver Manual de la serie **DMM-4000**

Posiciones de memoria MODBUS de las variables y constantes más habituales:

Sólo Lectura		Lectura/Escritura	
Entrada AI 1	30020	Consigna de constante SP 1	40270
Entrada AI 2	30020	Consigna de constante SP 2	40277
Calculada Art 1	30031	Consigna de constante SP 3	40284
Calculada Art 2	30032	Consigna de constante Str 1	40054
Calculada Art 3	30033	Consigna de constante Str 2	40058
Calculada Art 4	30034	Consigna de constante Str 3	40062
Valor de Hold Hld	30028	Consigna de constante Str 4	40066
Acumulado Int	30024	Consigna de alarma AL 1	40138
Acumulado 32 bit	30025	Consigna de alarma AL 2	40144
Totalizada DI 2	30014	Consigna de alarma AL 3	40150
Total de 32 bit	30015	Consigna de alarma AL 4	40156

8. CALIBRACIÓN (CONSULTAR A SU DISTRIBUIDOR)

Función exclusiva del Técnico de Instrumentación. **NO ENTRAR** si no es preciso Permite reajustar el equipo o recuperar los ajustes de fábrica. Esta tarea puede realizarse desde el propio teclado o desde el PC. Ver **Manual de Calibración**.

IMPORTANTE: Algunas versiones **DMM-4000** disponen de una **Calibración de Usuario** independiente de la principal, que permite adaptar fácilmente la señal de cualquier sensor conectado al transmisor. Esta forma de calibración responde a la necesidad de calibrar periódicamente el transmisor en campo por medio de patrones físicos. P.e. las series VQ como pH, ORP, EC, O2, Cloro, Ozono, Turbidez o ESI, o versiones de Nivel, Volumen, Caudal, etc.

Pueden encontrarse dos versiones de esta **Calibración de Usuario** en función de la periodicidad u oportunidad de realizarla.

Así, para las series de Variables Químicas se utiliza la **Calibración por Buffers**, que se encuentra en la función ***[CAL] [GEN]***, y que dada su necesidad de realizarla cada poco tiempo (incluso diariamente), es de acceso inmediato para el Instrumentista de Campo. En este tipo de calibración, se utilizan dos soluciones patrón de valor conocido, una vez realizada, el transmisor reconoce esos valores como puntos de referencia.

Para las variables que precisen calibrarse sólo cuando se cambia el proceso, p.e. Nivel, Caudal, etc., se utiliza la **Calibración de dos Puntos**, que está protegida por clave, pero accesible al instrumentista.

En esta forma de calibración, se sustituye el valor de un patrón por el valor de una medición (referencia) que se esté realizando en ese instante en el mismo producto, por medio de otro instrumento con garantía de precisión.

La exactitud de estos tipos de Calibración siempre tendrán una incertidumbre de error superior a la de ambos instrumentos juntos.

Ambas formas precisan tener conocimientos de instrumentación ya que en caso de realizarla inadecuadamente puede dejar al transmisor inutilizable hasta que sea realizada correctamente. Es absolutamente recondable antes de abordar este tipo de calibraciones leerse las Instrucciones específicas de cada modelo.

9. MANTENIMIENTO (CONSULTAR A SU DISTRIBUIDOR)

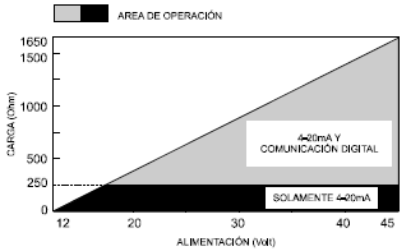
Estos equipos no precisan mantenimiento, debido a que todos sus parámetros de Configuración y Calibración son introducidos digitalmente, siendo posible acceder a ellos desde su teclado o con el software LoopWin que se suministra aparte. En cualquier forma, se recomienda que estas funciones sean realizadas por un Técnico Especializado con el fin de asegurar el buen funcionamiento del equipo.

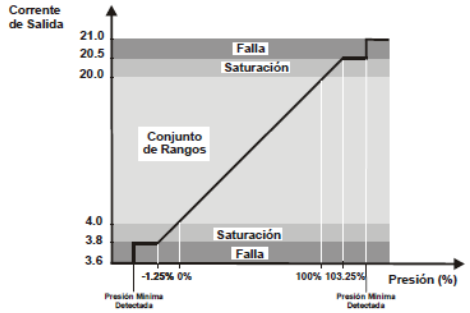
10. LOCALIZACIÓN DE AVERIAS

EFECTO	CAUSAS
Aparece en display undE	Señal de entrada menor que el límite bajo de escala. Línea de señal invertida.
Aparece en display oUEr	Señal de entrada mayor que el límite alto de escala. Línea de señal abierta.
Aparece en display Erro	Error de medida o línea desequilibrada. Comprobar continuidad o estado de sensor.
La medida indica siempre temperatura ambiente.	En aplicaciones con Termopar, la entrada o el sensor están cortocircuitados.
La medida indica un error por debajo de la temperatura real o la medida indica un error por encima de la temperatura real.	En aplicaciones con Termopar o RTD. Cable de línea inadecuado. Termopar no adecuado o señal no adecuada.
El display no se ilumina o luce débilmente.	Fallo de tensión de red o tensión baja. Exceso de calor dentro del cuadro.
En el display indica signo (-) y la señal es positiva.	Señal invertida en bornes de entrada o en el captador, transmisor o convertidor.
La lectura no corresponde a la medida prevista. Los errores son estables.	Señal no correcta, sensor equivocado, cable de línea inadecuado o mal polarizado. Captadores instalados incorrectamente o con vida agotada.
Los pilotos de los relés se apagan y se encienden, pero no hay salida.	Relés con contactos quemados, pistas fundidas, o abiertos, debido a que han sufrido un cortocircuito provocado por una defectuosa conexión.
Una salida relé o lógica asociada a una alarma no presenta el estado esperado.	Verificar la habilitación de las alarmas asociadas a la salida.
Existen unas pequeñas fluctuaciones en el display principal periódicamente.	Línea de entrada de señal muy parasitada (Actuaciones repetidas del sistema de reset de seguridad automático Watch-dog). Eliminar ruido en la línea.
En una entrada de 0..4-20 mA, la indicación del display principal pasa de mínimo a máximo con muy poca señal.	Rango mal configurado. Los valores mínimo y máximo están mal introducidos y no corresponden a la escala. Realizar de nuevo la configuración.

9.4. ANEXO D: DATASHEET SENSOR DE PRESIÓN LD301MS:

CARACTERÍSTICAS TÉCNICAS

Especificaciones Funcionales	
Fluido del Proceso	Liquido, gas o vapor.
Señal de Salida	4.20 mA a dos hilos, controlado de acuerdo a las especificaciones de NAMUR NE43 y con comunicación digital sobrepuesta (Protocolo HART). Observe la siguiente figura.
Alimentación	12 a 45 Vdc.
Indicador	Opcional de 4½ dígitos numéricos e indicador alfanumérico de 5 caracteres con indicador LCD.
Certificados de Area Peligrosa	Intrínsecamente seguro (FM, CSA, NEMKO, EXAM, CEPEL, NEPSI), a prueba de explosión (FM, CSA, NEMKO, CEPEL, NEPSI), a prueba en polvo (FM) y no inflamables (FM). Ver Apendice A.
Información de La Directiva Europea	**Authorized representative in European Community** Smar Gmbh-Rheingaustrasse 9-55545 Bad Kreuzanach **PED Directive (97/23/EC) – Pressure Equipment Directive** This product is in compliance with the directive and it was designed and manufactured in accordance with sound engineering practice using several standards from ANSI, ASTM, DIN and JIS. **EMC Directive (2004/108/EC) - Eletromagnetic Compatibility** The EMC test was performed according to IEC standard: IEC61326-1:2006, IEC61326-2-3:2006, IEC61000-6-4:2006, IEC61000-6-2:2005. For use in environment only. Keep the shield insulated at the instrument side, connecting the other one to the ground if necessary to use shielded cable. **ATEX Directive (94/9/EC) – Equipment and protective systems intended for use in potentially explosive atmospheres.** This product was certified according European Standards at NEMKO and EXAM (old DMT). The certified body for manufacturing quality assessment is EXAM (number 0158). **LVD Directive 2006/95/EC – Electrical Equipment designed for use within certain voltage limits** According the LVD directive Annex II the equipment under ATEX "Electrical equipment for use in an explosive atmosphere" directive are excluded from scope from this directive. The EC declarations of conformity for all applicable European directives for this product can be found at **www.smar.com.**
Ajustes de Cero y Span y Ajuste Local	No interactivo, vía comunicación digital. Jumper local ajustable en tres posiciones: simple, inhabilitado y completo.
Limitacion de Carga	

Especificaciones Funcionales	
Alarma de Falla	En caso de falla de sensor o del circuito, el auto-diagnóstico ajusta la salida a 3.6 o 21.0 mA, según la preferencia del usuario. 
Límites deTemperatura	Ambiente: -40 a 85 °C (-40 a 185 °F) Proceso: -40 a 100 °C (-40 a 212 °F) (Aceite Silicone) -40 a 85 °C (-40 a 185 °F) (Aceite Inerte Halocarbon) 0 a 85 °C (32 a 185 °F) (Aceite Inerte) -20 a 85 °C (-4 a 185 °F) (Aceite Inerte Krytox y Fomblim) -40 a 100 °C (-40 a 212 °F) (O'Ring Viton o Buna-N) -40 a 150 °C (-40 a 302 °F) (Modelo de Nivel) Almacen: -40 a 100 °C (-40 a 212 °F) Visor Digital: -20 a 80 °C (-4 a 176 °F) -40 a 85 °C (-40 a 185 °F) (Sin Daños)
Tiempo de Encendido	Funciona dentro de las especificaciones en menos de 5.0 segundos después de aplicarse la energía al transmisor.
Configuración	Por comunicación digital (Protocolo HART®) usando el software de configuración CONF401, DDCON (para Windows/Unix), o HPC301 y HPC401 (para Palms). Puede ser configurado usando la herramienta DD y FDT/DTM, y puede ser parcialmente configurado con el ajuste local. A fin de mantener la configuracion segura del equipo, el **LD301** tiene dos tipos de proteccion contra escritura en su memoria. Una es a traves del software y la otra un hardware seleccionando la clave del mecanismo con prioridad sobre el software.
Desplazamiento Volumétrico	Menos que 0.15 cm^3 (0.01 $pul.^3$)
Límites de Presión Alta y de Presión Estática	**De 3.45 kPa abs. (0.5 psia)* a:** 0.5 MPa (72.52 psi) para rango 0 8 MPa (1150 psi) para rango 1 16 MPa (2300 psi) para rangos 2, 3 & 4 32 MPa (4600 psi) para modelos H & A5 40 MPa (5800 psi) para modelos M5 52 MPa (7500 psi) para modelos M6 & A6 *excepto El modelo **LD301A** . Prueba de Presión de Brida: 60 MPa (8570 psi)

Clase de Presion ANSI B 16.5			
Clase	**150**	**300**	**600**
Temperatura	**Límite de Presion**		
-29 a 38 °C	1893 kPa (274,6 psi)	4962 kPa (719 psi)	9924 kPa (1439.4 psi)
93 °C	1618 kPa (234.7 psi)	4275 kPa (620 psi)	8551 kPa (1240.2 psi)
149 °C	1481 kPa (214.8 psi)	3864 kPa (560.4 psi)	7717 kPa (1119.3 psi)

Especificaciones Funcionales	

DIN EN 1092-1 / DIN 2501				
Material de Brida: Acero Inx 316L				
Temperatura	**- 10 a 50 °C**	**50 °C**	**100 °C**	**150 °C**
PN	**Límite de Presion**			
16	1230 kPa (178.4 psi)	1180 kPa (171.1 psi)	1020 kPa (148 psi)	930 kPa (135 psi)
40	3060 kPa (443.8 psi)	2960 kPa (429.3 psi)	2550 kPa (370 psi)	2310 kPa (335 psi)

Estas presiones no van a dañar el transmisor, pero puede ser necesaria una nueva calibración.

Límites de humedad	0 a 100% RH (Humedad relativa).
Ajuste de Amortiguación	Configurable por el usuario de 0 a 128 segundos (vía comunicación digital).

Especificaciones de Rendimiento	
Condiciones de referencia	Span a partir de cero, temperatura de 25°C (77°F), presión atmosférica, fuente de alimentación de 24 Vcc, fluido de llenado aceite de Silicona, aislamiento de diafragmas en 316L SST y ajustes digitales igual a valores inferior y superior del rango.
Precisión	**Para rango 0, Presión diferencial y manométrica, diafragma de 316L SST o hastelloy con fluido de silicona o halocarbón:** **0.2 URL ≤ span ≤ URL:** ± 0.1% de span **0.05 URL ≤ span < 0.2 URL:** ± [0.025+0.015 URL/span]% de span **Para rangos 1, 2, 3 ,4, 5 y 6, Presion differencial o manometrica, diafragma de 316L SST o hastelloy con fluido de silicona o halocarbon:** **0.1 URL ≤ span ≤ URL:** ± 0.075% de span **0.025 URL ≤ span < 0.1 URL:** ± [0.0375+0.00375.URL/span]% de span **0.0083 URL ≤ span < 0.025 URL:** ± [0.0015+0.00465.URL/span]% de span **Para rangos 2 a 6 y Presión absoluta. Para diafragma deTántalo o Monel. Para fluidos fluorables:** **0.1 URL ≤ span ≤ URL:** ± 0.1% de span **0.025 URL ≤ span < 0.1 URL:** ± 0.05[1+0.1 URL/span]% de span **0.0083 URL ≤ span < 0.025 URL:** ± [0.01+0.006 URL/span]% de span **Para rango 1 y Presión absoluta:** ± 0.2% de span **Para rangos 2, 3 o 4 y de nivel, diafragma de 316L SST con fluido silicona o halocarbón con presión máxima se pone la Brida de acuerdo a la clase de presión:** **0.1 URL ≤ span ≤ URL:** ± 0.075% de span **0.025 URL ≤ span < 0.1 URL:** ± [0.0375+0.00375.URL/span]% de span **0.0083 URL ≤ span < 0.025 URL:** ± [0.0015+0.00465.URL/span]% de span Efectos de linealidad, histéresis y repetibilidad están incluidos.
Estabilidad	**Para rangos 2, 3, 4, 5 and 6**: ± 0.15% de URL para 5 anos a 20 ºC cambios de temperatura y presion estatica de 7 MPa (1000 psi). **Para rangos 0 and 1:** ± 0.2% de URL por 12 meses a cambios de temperatura de 20 ºC y de presion estatica de 100 kPa (1bar). **Para modelos de nivel:** ± 0.2% de URL por 12 meses a cambios de temperatura 20 ºC.
Efecto de Temperatura	**Para rangos 2, 3, 4 y 5:** **0.2 URL ≤ span ≤ URL:** ± [0.02% URL + 0.06% span] à 20 ºC (68 ºF) **0.0085 URL ≤ span < 0.2 URL:** ± [0.023% URL + 0.045% span] à 20 ºC (68ºF) **Para rango 1:** **0.2 URL ≤ span ≤ URL:** ± [0.08% URL + 0.05% span] à 20 ºC (68 ºF) **0.025 URL ≤ span < 0.2 URL:** ± [0.06% URL + 0.15% span] à 20 ºC (68 ºF) **Para rango 0:** **0.2 URL ≤ span ≤ URL:** ± [0.15% URL + 0.05% span] à 20 ºC (68 ºF)

Especificaciones de Rendimiento	
	0.05 URL ≤ **span** < **0.2 URL:** ± [0.1% URL + 0.3% span] à 20 °C (68 °F) **Para modelos de nível:** 6 mmH_2O à 20 °C para 4" y DN100 17 mmH_2O à 20 °C para 3" y DN80 Consulte Smar para otras dimensiones de Brida y fluido de llenado.
Efecto de Presión Estática	**Error de Zero:** **Para rangos 2, 3, 4 and 5:** ± 0.033% of URL à 7MPa (1000 psi) **Para rango 1:** ± 0.05% da URL à 1.7 MPa (250 psi) **Para rango 0:** ± 0.1% da URL à 0.5 MPa (5 bar) **Para Nivel:** ± 0.1% da URL à 3.5 MPa (500 psi) El error de cero es un error sistemático que puede ser eliminado por calibración de la presión estática de operación. **Error de Span:** **Para rangos 2, 3, 4, 5 y 6:** corregibles a ± 0.2% de lectura por 7MPa (1000 psi) **Para rangos 1 y transmisores de nivel:** corregibles a ± 0.2% de lectura por 3.5 MPa (500 psi) **Para rango 0:** corregibles a ± 0.2% de lectura por 0.5 MPa (5 bar) (70 psi)
Efecto de Fuente de Alimentación	± 0.005% del span calibrado por volt.
Efecto de Posición de Montaje	Cambio cero de hasta 250 Pa (1 inH_2O) que puede ser calibrado. Ningún efecto span.
Efecto de Interferencia Electromagnética	Proyectado para atender las normas IEC61326-1:2006, IEC61326-2-3:2006, IEC61000-6-4:2006, IEC61000-6-2:2005.

Especificaciones Físicas		
Conexiones Eléctricas	1/2 - 14 NPT 3/4 - 14 NPT con adaptador en Acero Inox 316 para 1/2 - 14 NPT) 3/4 - 14 BSP con adaptador en Acero Inox 316 para 1/2 - 14 NPT) 1/2 - 14 BSP con adaptador en Acero Inox 316 para 1/2 - 14 NPT) M20 X 1.5 PG 13.5 DIN	**Note:** A prueba de explosión no aplica el adaptador, solo el transmisor.
Conexión de Proceso	1/4 - 18 NPT o 1/2 -14 NPT (con adaptador) Para modelos L, vea el código de pedidos.	
Partes húmedas	**Diafragmas Aisladores:** Acero Inox 316L, Hastelloy C276, Monel 400 o Tántalo **Válvulas de Drenaje/Ventilación:** Acero Inox 316, Hastelloy C276 o Monel 400 **Bridas:** Acero Carbono Bicromatizado, 316 SST-CF8M (ASTM - A351), Hastelloy C276 - CW-12MW, (ASTM - A494) o Monel 400 **Empaques (Para Bridas y Adaptadores):** En Buna N, VITON ™ o TEFLON ™. En Etileno-Propileno bajo consulta. El **LD301** está disponible en materiales conforme a NACE MR-01-75.	
Piezas no Húmedas	**Carcasa:** Aluminio y acabado con pintura de poliéster, y pintura epóxi en la carcasa en Acero Inox 316 - CF8M (ASTM - A351). Cumple con NEMA 4X/6P, IP66 o IP66W*, IP68 o IP68W* **El grado de protección de IP66/68W para 10m/24h es usado solamente para empaques. Para cualquier otra duda consulta Smar. IP66/68W fue probado por 200h y cumple con la norma NBR 8094 / ASTM B 117.* **Bridas Ciega (Para Modelos M y A):** Acero carbono bicromeado, cuando la Brida mojada sea hecha del mismo material, y el Acero Inox 316 para el modelo L y en los demás casos. **Material de la Brida de Nivel (LD301L):** Acero Inox 316 L, Acero Inox 304, Hastelloy C276 y Acero al Carbón Revestido **Fluido de Llenado:** Silicone, Inerte , Aceites Krytox, Halocarbon 4.2 o Fomblim	

Especificaciones Físicas	
	Empaque de las Tapas: Buna-N **Soporte de Montaje:** Acero al Carbon Bicromeado o Acero Inox 316 Accesorios (tornillos, tuerca etc)en Acero Carbono o Acero Inox 316 **Tornillos y Tuercas de la Brida:** Acero Carbono Bicromado, grado de resistencia 8, Acero Inoxidable 316, o Acero Carbono B7M (para aplicaciones NACE). **Placa de Identificación:** Acero Inox 316
Montaje	a - Con Brida montada para los modelos **LD301L**. b - Abrazadera de montaje opcional universal para superficie, o vertical / horizontal (DN 50) para tubo de 2" (opcional). c - Mediante la abrazadera en la válvula (opcional). d - Directamente en la tubería para combinaciones de Bridas en el caso de montaje opcional universal, o vertical/horizontal (DN50)
Pesos Aproximados	3.15 kg (7 lb) : todos los modelos, excepto los modelos L. 5.85 a 9.0 kg (13 lb. a 20 lb): modelos L según las bridas, la extensión y los materiales.
Características de Control (Opcional) PID	PID y TOT

Características técnicas de Alto rendimiento - CODE L1

La opción de Alto Rendimiento (código L1) esta disponible solo bajo las siguientes condiciones:

Aplicacion	Diferencial y Manometrica						
Rango	D2	-50	to	50 kPa	-200	to	200 inH_2O
	D3	-250	to	250 kPa	-36	to	36 psi
	D4	-2500	to	2500 kPa	-360	to	360 psi
	M2	-50	to	50 kPa	-200	to	200 inH_2O
	M3	-100	to	250 kPa	-14.5	to	36 psi
	M4	-100	to	500 kPa	-14.5	to	360 psi
Material de Diafragma	316L SST o Hastelloy C276						
Fluído de llenado	Silicone						

Especificaciones de Rendimiento	
Condiciones de referencia	Span iniciando en zero, temperatura de 25 °C (77 °F), Presión atmosférica, Alimentación de 24 Vdc, Fluido de Llenado Aceite de Silicon, Diafragma en 316L SST y ajustes digitales igual al valor inferior y superior del rango.
Precisión	**Para rango 2:** **0.2 URL ≤ span ≤ URL:** ± 0.04% de span **0.05 URL ≤ span < 0.2 URL:** ± [0.021667+0.003667URL/span]%de span **0.0085 URL ≤ span < 0.05 URL:** ± [0.0021+0.004645URL/span]% de span **Para rango 3 o 4:** **0.1 URL ≤ span ≤ URL:** ± 0.05% do span **0.05 URL ≤ span < 0.1 URL:** ± [0.005+0.0045URL/span]% do span **0.0085 URL ≤ span < 0.05 URL:** ± [0.0021+0.004645URL/span]% do span
Estabilidad	**Para rango 2:** ± 0.05% de URL por 6 meses **Para rango 3:** ± 0.075% de URL por 12 meses **Para rango 4:** ± 0.1% da URL por 24 meses ± 0.2% de URL por 12 anos, a cambios de temperatura de 20 °C y a 7 MPa (1000 psi) {70 bar} de Presión estática, en ambiente libre de Hidrogeno.
Efecto de Temperatura	**De -10 °C a 50 °C, protegido para radiacion solar:** **0.2 URL ≤ span ≤ URL:** ±[0.018% URL + 0.012% span] por 20 °C (68 °F) **0.0085 URL ≤ span < 0.2 URL:** ±[0.02% URL + 0.002% span] por 20 °C (68 °F)

Efecto de Presión Estática	**Error de Zero:** ± 0.025% URL para 7 MPa (1000 psi) El error de zero es un error sistemático que puede ser eliminado calibrando la presión estática de operación. **Span error:** Corregible a ± 0.2% de lectura por 7 MPa (1000 psi).

NOTA		
Hastelloy es una marca registrada de Cabot Corp. Monel es una marca registrada de International Nickel Co. Viton y Teflon son marcas registradas de E.I. Dupont de Nemours & Co.	Inert es una marca registrada de Hooker Chemical Corp. Halocarbon es una marca registrada de Halocarbon. HART® es una marca registrada de HART® communication Foundation.	Los Transmisores de Presión Smar están protegidos por la patente número 6,433,791 E.U.

9.5. ANEXO E: DATASHET USB-1408FS-PLUS DAQ:

Specifications

All specifications are subject to change without notice.
Typical for 25°C unless otherwise specified.
Specifications in *italic text* are guaranteed by design.

Analog input

Table 1. Analog input specifications

Parameter	Condition	Specification
A/D converter type		Successive approximation type
Input voltage range for linear operation	CHx to GND	Single-ended mode: ±10 V max Differential mode: –10 V min, +20 V max
Absolute maximum input voltage	*CHx to GND*	*±28 V max*
Input impedance		*122 kΩ*
Input current (Note 1)	Vin = +10 V	70 µA typ
	Vin = 0 V	–12 µA typ
	Vin = –10 V	–94 µA typ
Number of channels		8 single-ended or 4 differential; software-selectable
Input ranges	Single-ended	±10 V, G=2
	Differential	±20 V, G=1 ±10 V, G=2 ±5 V, G=4 ±4 V, G=5 ±2.5 V, G=8 ±2.0 V, G=10 ±1.25 V, G=16 ±1.0 V, G=20 Software-selectable
Throughput (Note 2)	Software paced	250 S/s typ, PC-dependent
	Hardware paced	0.014 S/s to 48 kS/s
Channel gain queue		Software selectable. 8 elements in SE mode, 4 elements in DIFF mode. One gain element per channel. Elements must be unique and listed in ascending order.
Resolution (Note 3)	Differential	14 bits, no missing codes
	Single-ended	13 bits
Integral linearity error		±2 LSB typ
Differential linearity error		±0.5 LSB typ
Absolute accuracy long term drift (Note 4)	±20 V range	±3 LSB typ (Δt = 1000 hrs)
	±4 V range	±6 LSB typ (Δt = 1000 hrs)
	±1 V range	±8 LSB typ (Δt = 1000 hrs)
Trigger source		External digital: TRIG_IN Software-selectable

Note 1: Input current is a function of applied voltage on the analog input channels. For a given input voltage, Vin, the input leakage is approximately equal to (8.181 * Vin – 12) µA.

Note 2: Maximum throughput when scanning is machine dependent.

Note 3: The ADS7871 converter only returns 13 bits (0 to 8,192 codes) in single-ended mode.

Note 4: Extrapolating the long term drift accuracy specifications will provide the approximate long term drift of the intermediate input ranges.

Accuracy

Table 2. Accuracy, differential mode

Range	Absolute Accuracy 25 °C (±mV)	Absolute Accuracy 0 °C to 50°C (±mV)
±20 V	10.98	49.08
±10 V	7.32	33.42
±5 V	3.66	20.76
±4 V	2.92	19.02
±2.5 V	1.83	14.97
±2 V	1.70	14.29
±1.25 V	1.21	12.18
±1 V	1.09	11.63

Table 3. Accuracy, single-ended mode

Range	Absolute Accuracy 25 °C (±mV)	Absolute Accuracy 0 °C to 50 °C (±mV)
±10 V	10.98	49.08

Noise performance

Table 4. Noise performance, differential mode

Range	Typical counts	Least significant bitroot mean square (LSBrms)
±20 V	8	1.21
±10 V	8	1.21
±5 V	9	1.36
±4 V	10	1.51
±2.5 V	12	1.81
±2 V	14	2.12
±1.25 V	18	2.72
±1 V	22	3.33

Table 5. Noise performance, single-ended mode

Range	Typical Counts	LSBrms
±10 V	8.0	1.21

Analog output

Table 6. Analog output specifications

Parameter	Condition	Specification
Resolution		12-bits, 1 in 4,096
Output range		*0 V to 5.0 V*
Number of channels		2
Throughput (Note 5)	Software paced	250 S/s single channel typ, PC dependent
	Hardware paced, per channel	50 kS/s max
Power on and reset voltage		0 V, ±20 mV typ; initializes to 000h code
Output drive	Each D/A OUT	5 mA, sourcing
Slew rate		0.8 V/ µs typ

Note 5: Maximum throughput when scanning is machine dependent.

Table 7. Analog output accuracy, all values are (±); accuracy tested at no load

Range	Accuracy (LSB)
0 V to 5.0 V	4.0 typ, 45.0 max

Table 8. Analog output accuracy components, all values are (±)

Range	% of FSR	Gain Error at FS (mV)	Offset (mV) (Note 6)	Accuracy at FS (mV)
0 V to 5.0 V	0.1 typ, 0.9 max	4.0 typ, 36.0 max	1.0 typ, 9.0 max	4.0 typ, 45.0 max

Note 6: Zero-scale offsets may result in a fixed zero-scale error producing a "dead-band" digital input code region. In this case, changes in digital input code at values less than 0x040 may not produce a corresponding change in the output voltage. The offset error is tested and specified at code 0x040.

Digital input/output

Table 9. Digital I/O specifications

Parameter	Specification
Digital type	CMOS
Number of I/O	16 (Port A0 through A7, Port B0 through B7)
Configuration	2 banks of 8. Port B is high current drive.
Pull up/pull-down configuration	All pins pulled up to 5V via 47 kΩ resistors (default). Change to pull-down using internal user-configurable jumpers.
Input high voltage threshold	2.0 V min
Input high voltage limit	5.5 V absolute max
Input low voltage threshold	0.8 V max
Input low voltage limit	–0.5 V absolute min 0 V recommended min
Output high voltage, Port A	4.4 V min (IOH = –20 µA) 3.84 V min (IOH = –6.0 mA)
Output low voltage, Port A	0.1 V max (IOL = 20 µA) 0.33 V max (IOL = 6.0 mA)
Output high voltage, Port B	4.4 V min (IOH = –50 µA) 3.76 V min (IOH = –24.0 mA)
Output low voltage, Port B	0.1 V max (IOH = 50 µA) 0.44 V max (IOH = 24.0 mA)
Power on and reset state	Input

External trigger

Table 10. Digital trigger specifications

Parameter	Specification
Trigger source (Note 7)	External digital; TRIG_IN terminal
Trigger mode	Edge sensitive; software-selectable for CMOS compatible rising or falling edge, high or low level.
Trigger latency	10 µs max
Trigger pulse width	1 µs min
Input type	Schmitt trigger, 47 kΩ pull-down to ground
Schmitt trigger hysteresis	1.01 V typ 0.6 V min 1.5 V max
Input high voltage threshold	2.43 V typ 1.9 V min 3.1V max
Input high voltage limit	5.5 V absolute max
Input low voltage threshold	1.42 V typ 1.0 V min 2.0 V max
Input low voltage limit	–0.5 V absolute min 0 V recommended min

External clock input/output

Table 11. External clock I/O specifications

Parameter	Condition	Specification
Terminal name		SYNC
Terminal type		Bidirectional
Direction (software-selectable)	Output	Outputs the internal A/D pacer clock. Active on rising edge.
	Input (default)	Receives A/D pacer clock from external source. Active on rising edge.
Input clock rate		48 kHz, max
Clock pulse width	Input mode	1 µs min
	Output mode	5 µs min
Input type		Schmitt trigger, 47 kΩ pull-down to ground
Schmitt trigger hysteresis		1.01 V typ 0.6 V min 1.5 V max
Input high voltage threshold		2.43 V typ 1.9 V min 3.1V max
Input high voltage limit		5.5 V absolute max
Input low voltage threshold		1.42 V typ 1.0 V min 2.0 V max
Input low voltage limit		–0.5 V absolute min 0 V recommended min
Output high voltage		4.4 V min (IOH = –50 µA) 3.80 V min (IOH = –8 mA)
Output low voltage		0.1 V max (IOL = 50 µA) 0.44 V max (IOL = 8 mA)

Counter

Table 12. Counter specifications

Parameter	Specification
Pin name	CTR
Counter type	Event counter
Number of channels	1
Input type	Schmitt trigger, 47 kΩ pull-down to ground, rising edge triggered
Input source	*CTR screw terminal*
Resolution	32 bits
Maximum input frequency	1 MHz
High pulse width	*500 ns min*
Low pulse width	*500 ns min*
Schmitt trigger hysteresis	1.01 V typ 0.6 V min 1.5 V max
Input high voltage threshold	2.43 V typ 1.9 V min 3.1V max
Input high voltage limit	5.5 V absolute max
Input low voltage threshold	1.42 V typ 1.0 V min 2.0 V max
Input low voltage limit	–0.5 V absolute min 0 V recommended min

Memory

Table 13. Memory specifications

Parameter	Specification
Non-volatile EEPROM	2,048 bytes (768 bytes calibration, 256 bytes user, 1,024 bytes DAQFlex)

Microcontroller

Table 14. Microcontroller specifications

Parameter	Specification
Type	*High performance 16-bit RISC microcontroller*

Power

Table 15. Power specifications

Parameter	Condition	Specification
Supply current	During USB enumeration	< 100 mA
	After USB enumeration, including DIO, AO, SYNC, and +VO output loading	< 500 mA
+VO power available	After USB enumeration	4.5 V min, 5.25 V max
+VO output current	After USB enumeration	100 mA max

General

Table 16. General specifications

Parameter	Specification
Device type	USB 2.0 full speed
Device compatibility	USB 1.1, USB 2.0

Environmental

Table 17. Environmental specifications

Parameter	Specification
Operating temperature range	0 °C to 70 °C
Storage temperature range	–40 °C to 70 °C
Humidity	0% to 90% non-condensing

Mechanical

Table 18. Mechanical specifications

Parameter	Specification
Dimensions (L × W × H)	79 × 82 × 27 mm (3.10 × 3.20 × 1.05 in.)
USB cable length	3 m (9.84 ft) max
User connection length	3 m (9.84 ft) max

Screw terminal connector

Table 19. Screw terminal specifications

Parameter	Specification
Connector type	Screw terminal
Wire gauge range	16 AWG to 30 AWG

Differential mode pinout

Table 20. 4-channel differential mode pinout

Pin	Signal name	Pin description	Pin	Signal name	Pin description
1	CH0 IN HI	Analog input 0+	21	Port A0	Port A bit 0
2	CH0 IN LO	Analog input 0–	22	Port A1	Port A bit 1
3	AGND	Analog ground	23	Port A2	Port A bit 2
4	CH1 IN HI	Analog input 1+	24	Port A3	Port A bit 3
5	CH1 IN LO	Analog input 1–	25	Port A4	Port A bit 4
6	AGND	Analog ground	26	Port A5	Port A bit 5
7	CH2 IN HI	Analog input 2+	27	Port A6	Port A bit 6
8	CH2 IN LO	Analog input 2–	28	Port A7	Port A bit 7
9	AGND	Analog ground	29	GND	Ground
10	CH3 IN HI	Analog input 3+	30	+VO	Power output
11	CH3 IN LO	Analog input 3–	31	GND	Ground
12	AGND	Analog ground	32	Port B0	Port B bit 0
13	D/A OUT 0	Analog output 0	33	Port B1	Port B bit 1
14	D/A OUT 1	Analog output 1	34	Port B2	Port B bit 2
15	AGND	Analog ground	35	Port B3	Port B bit 3
16	Reserved	Reserved for future use	36	Port B4	Port B bit 4
17	GND	Ground	37	Port B5	Port B bit 5
18	TRIG_IN	Trigger input	38	Port B6	Port B bit 6
19	SYNC	Synchronization I/O	39	Port B7	Port B bit 7
20	CTR	Counter input	40	GND	Ground

Single-ended mode pinout

Table 21. 8-channel single-ended mode pinout

Pin	Signal name	Pin description	Pin	Signal name	Pin description
1	CH0 IN	Analog input 0	21	Port A0	Port A bit 0
2	CH1 IN	Analog input 1	22	Port A1	Port A bit 1
3	AGND	Analog ground	23	Port A2	Port A bit 2
4	CH2 IN	Analog input 2	24	Port A3	Port A bit 3
5	CH3 IN	Analog input 3	25	Port A4	Port A bit 4
6	AGND	Analog ground	26	Port A5	Port A bit 5
7	CH4 IN	Analog input 4	27	Port A6	Port A bit 6
8	CH5 IN	Analog input 5	28	Port A7	Port A bit 7
9	AGND	Analog ground	29	GND	Ground
10	CH6 IN	Analog input 6	30	+VO	Power output
11	CH7 IN	Analog input 7	31	GND	Ground
12	AGND	Analog ground	32	Port B0	Port B bit 0
13	D/A OUT 0	Analog output 0	33	Port B1	Port B bit 1
14	D/A OUT 1	Analog output 1	34	Port B2	Port B bit 2
15	AGND	Analog ground	35	Port B3	Port B bit 3
16	Reserved	Reserved for future use	36	Port B4	Port B bit 4
17	GND	Ground	37	Port B5	Port B bit 5
18	TRIG_IN	Trigger input	38	Port B6	Port B bit 6
19	SYNC	Synchronization I/O	39	Port B7	Port B bit 7
20	CTR	Counter input	40	GND	Ground