

**UNIVERSIDAD DE OVIEDO**

**CENTRO INTERNACIONAL DE POSTGRADO**

**MÁSTER ERASMUS MUNDUS EN INGENIERÍA  
MECATRÓNICA (EU4M)**

**TRABAJO FIN DE MÁSTER**

**Implementación de un Sistema de Monitorización Industrial: caso de  
uso Máquina Eléctrica**

**JULIO 2018**

**Alumno**  
**Diego Aníbal Sandoval**  
**Núñez**

**Tutor académico**  
**Ph. D. María Jesús**  
**Lamela Rey**

**Tutor empresarial**  
**Ph. D. Javier Anduaga**

## **RESUMEN**

El ámbito de los sistemas de monitorización de la condición (Condition Monitoring Systems, CMS) ha vivido en los últimos años un período de novedosos desarrollos, los cuales permiten detectar el malfuncionamiento de un sistema o componente. Además, permiten en muchos casos localizar el origen del mismo y estimar su vida útil remanente en función de diferentes señales obtenidos de sensores ubicados dentro del sistema mecatrónico. Sin embargo, estas soluciones a menudo plantean desarrollos más ajustados a lo requerido por el cliente (sistemas ad-hoc). Por más que los proveedores de soluciones industriales realicen esfuerzos por adaptar sus soluciones a diferentes problemáticas, la necesidad de estandarizar estas soluciones dificulta su implantación por la industria. Es así como grandes fabricantes de sistemas de automatización (Siemens, Beckhoff, B&R, Omron) ofrecen soluciones generalistas, los cuales en múltiples ocasiones no poseen las prestaciones de otros métodos.

El presente proyecto fin de máster plantea la integración de técnicas avanzadas de monitorización de la condición en equipos de uso industrial. Para ello, emplea en primera instancia hardware disponible en la industria. Esto permite conocer más de cerca las soluciones disponibles en el mercado actual junto a su análisis de factibilidad técnica en la integración con técnicas más avanzadas de monitorización. Como caso práctico, se realiza el CMS de una máquina eléctrica.

## **PALABRAS CLAVE**

Sistemas de monitorización de la condición, análisis vibraciones, PLC.

## ÍNDICE GENERAL

<b>ÍNDICE DE FIGURAS.....</b>	<b>6</b>
<b>ÍNDICE DE TABLAS.....</b>	<b>8</b>
<b>1. INTRODUCCIÓN.....</b>	<b>9</b>
1.1. MOTIVACIÓN.....	10
1.2. CONTEXTO Y PLANTEAMIENTO DEL PROBLEMA.....	10
1.3. OBJETIVOS.....	11
1.3.1. <i>Objetivo general</i> .....	11
1.3.2. <i>Objetivos específicos</i> .....	11
<b>2. ESTADO DEL ARTE.....</b>	<b>13</b>
2.1. LA VIBRACIÓN Y SUS EFECTOS EN MÁQUINAS E INDUSTRIA.....	14
2.2. RODAMIENTOS Y SU ESTRUCTURA.....	16
2.3. VIBRACIÓN EN RODAMIENTOS E INDICADORES.....	18
2.4. BANDA Y TERCIO DE OCTAVA.....	22
2.5. SISTEMAS DE MONITORIZACIÓN DE LA CONDICIÓN.....	23
2.6. SISTEMAS DE MONITORIZACIÓN DE LA CONDICIÓN EN EL MERCADO.....	24
<b>3. DISEÑO DE SISTEMA.....</b>	<b>27</b>
3.1. ELEMENTOS DE HARDWARE.....	28
3.1.1. <i>PLC S7-1212C</i> .....	29
3.1.2. <i>SM1281</i> .....	29
3.1.3. <i>HMI TP700 Comfort Panels</i> .....	31
3.1.4. <i>Sensor S01</i> .....	32
3.1.5. <i>Motor eléctrico</i> .....	32
3.2. ELEMENTOS DE SOFTWARE.....	33
3.2.1. <i>Totally Integrated Automation Portal (TIA Portal)</i> .....	33
3.2.2. <i>STEP7</i> .....	36
3.2.2.1. Lenguaje SCL.....	37
3.2.2.1.1. Bloque de Organización.....	38
3.2.2.1.1.1. Funciones.....	39
3.2.2.1.1.2. Bloque de función.....	39
3.2.2.1.1.3. Bloques de Datos.....	40
3.2.2.1.2. Diseño de programa y modulación.....	40
3.2.3. <i>WinCC</i> .....	44
3.2.3.1. Windows CE.....	45
3.2.3.2. Visual Basic Script.....	46
3.2.3.3. Diseño de interfaz.....	47
<b>4. INSTALACIÓN Y PUESTA EN MARCHA DEL SISTEMA.....</b>	<b>49</b>
4.1. ELEMENTOS DE HARDWARE.....	49

4.1.1.	<i>PLC S7/1212C</i> .....	49
4.1.2.	<i>SM1281</i> .....	49
4.1.3.	<i>HMI TP700 Comfort Panels</i> .....	51
4.2.	ELEMENTOS DE SOFTWARE.....	51
4.2.1.	<i>TIA Portal</i> .....	51
4.2.2.	<i>STEP7</i> .....	52
4.2.2.1.	Funcionamiento módulo SM 1281 .....	52
4.2.3.	<i>WinCC</i> .....	55
<b>5.</b>	<b>DESARROLLO DE SOLUCION IMPLEMENTADA</b> .....	<b>57</b>
5.1.	OBTENCIÓN DATOS DE SEÑAL DE VIBRACIÓN.....	58
5.1.1.	<i>Generar archivo en SM 1281 usando el PLC</i> .....	58
5.1.2.	<i>Descarga de archivo</i> .....	59
5.1.2.1.	Alternativas comunicación WebDAV.....	60
5.1.2.2.	Alternativas comunicación FTP.....	61
5.1.3.	<i>Extracción de datos raw desde archivo WAV</i> .....	62
5.2.	PROCESAMIENTO .....	64
5.2.1.	<i>Lenguaje de programación para PLC</i> .....	64
5.2.1.1.	Librería OSCAT.....	65
5.2.2.	<i>Código MATLAB</i> .....	66
5.2.2.1.	Programa en MATLAB .....	66
5.2.2.2.	Análisis funciones en MATLAB.....	67
5.2.3.	<i>código en Visual Basic Script</i> .....	68
5.3.	INTERFAZ HMI .....	69
<b>6.</b>	<b>COMPROBACIÓN Y VALIDACIÓN DE SISTEMA</b> .....	<b>75</b>
6.1.	COMPARACIÓN CÓDIGOS MATLAB, SCL Y VBS.....	75
6.2.	VALIDACIÓN DE SISTEMA EN MAQUETA DE ASCENSOR.....	76
<b>7.</b>	<b>CONCLUSIONES</b> .....	<b>78</b>
7.1.	CONCLUSIONES GENERALES.....	78
7.2.	CONCLUSIONES ESPECÍFICAS .....	78
7.2.1.	<i>Conclusiones sobre PLC</i> .....	78
7.2.2.	<i>Conclusiones sobre SM1281</i> .....	79
7.2.3.	<i>Conclusiones sobre HMI P700 Comfort</i> .....	79
7.2.4.	<i>Conclusiones sobre TIA Portal</i> .....	79
7.2.5.	<i>Líneas Futuras</i> .....	80
<b>8.</b>	<b>BIBLIOGRAFÍA</b> .....	<b>81</b>
<b>9.</b>	<b>ANEXO</b> .....	<b>87</b>
	ANEXO I: PROGRAMAS DESARROLLADOS.....	87
	Función PLC: MAIN.....	87
	<i>Función PLC: Calc_dB</i> .....	90
	<i>Interfaz HMI: Grafica</i> .....	91
	<i>Interfaz HMI: Grafica2</i> .....	95
	<i>Interfaz HMI: Instrucciones</i> .....	99
	<i>VBScript: generar_10</i> .....	101

<i>VBScript: convertir_20</i> .....	103
<i>VBScript: calculodB_30</i> .....	106
<i>Código MATLAB: octband</i> .....	108
<i>Código MATLAB: octfilt</i> .....	109
<i>Código MATLAB: linears</i> .....	109
<i>Código MATLAB: exponencial</i> .....	109
<i>Código MATALB: filtrito_3500</i> .....	110
ANEXO II: PRESUPUESTO .....	111

## ÍNDICE DE FIGURAS

Figura 2.1: Esquema de medición de vibración directamente sobre equipo. Fuente: Fluke. ....	14
Figura 2.2: Captura de la señal de vibración de un motor eléctrico y como las distintas vibraciones son sumadas en una única señal de vibración. Fuente: Elaboración propia. ....	16
Figura 2.3: Esquema general de un rodamiento y las partes que lo componen. Fuente: "Fatiga De Ejes, Soportes y Ajustes", Julio Vergara Aimone ,2017 PUC. ..	17
Figura 2.4: Datos estadísticos de daños en rodamientos a nivel mundial. Fuente: Averías de los rodamientos, FAG, 2002.....	17
Figura 2.5: Daño por defecto de montaje. Fuente: <a href="http://www.blogmecanicos.com">http://www.blogmecanicos.com</a> . 18	18
Figura 2.6: Relación entre la amplitud de una señal y su frecuencia. Fuente: CMS1200, Operating instructions, Siemens, 2017. ....	18
Figura 2.7: Distintas métricas de un rodamiento que afectan a las componentes espectrales de vibración. Fuente: página web SM1281, Siemens, 2018. ....	20
Figura 2.8: Aspecto físico del dispositivo Dynamix 1444 Integrated Condition Monitoring. Fuente: Allen-Bradley, 2017.....	24
Figura 2.9: TC3 Scope View Professional en ejecución. Fuente: Beckhoff, 2017.....	25
Figura 2.10: CMS 2000 VIB de Siemens. Fuente: Siemens, 2017. ....	26
Figura 3.1: Esquema general conexionado elementos. Fuente: elaboración propia. 27	27
Figura 3.2: Disposición física final de los elementos del sistema en laboratorio.....	28
Figura 3.3: Aspecto real SIMATIC s7-1200. Fuente: Siemens, 2017.....	29
Figura 3.4: Aspecto real SM1281. Fuente: Siemens, 2017. ....	30
Figura 3.5: Aspecto página web del SM1281. Fuente: Siemens, 2017.....	30
Figura 3.6: Aspecto página web del SM1281 para la configuración de rodamientos a monitorizar. Fuente: Siemens, 2017.....	31
Figura 3.7: Aspecto real HMI TP700 Comfort Panels. Fuente: Siemens, 2017.....	32
Figura 3.8: Sensor S01. Fuente: Siemens, 2017. ....	32
Figura 3.9: Esquema conjunto motor en maqueta ascensor. Fuente: "Construcción y puesta a punto de un banco de ensayos de ascensor a escala", Jon Galdos, Escuela Politécnica Superior de Mondragon Unibersitate, 2014.....	33
Figura 3.10: Pantalla principal de TIA Portal. Fuente: TIA Portal, Siemens, 2017. 34	34
Figura 3.11: Distribución configuraciones en interfaz de TIA Portal. Fuente: TIA Portal, Siemens, 2017.....	35
Figura 3.12: Árbol de proyecto y de opciones dentro del PLC en TIA Portal. a) destacado el PLC dentro del proyecto creado. b) destacado las secciones de programación en rojo y variables a utilizar en negro. c) detalle de estructura de programación y variables de PLC. Fuente: TIA Portal, Siemens, 2017. ....	36
Figura 3.13: Aspecto visual TIA Portal al programar en lenguaje LADDER y SCL. a) detalle de programación en LADDER. b) detalle interfaz programación en SCL. Fuente: TIA Portal, Siemens, 2017. ....	37
Figura 3.14: Estructura de funcionamiento del software del dispositivo PLC al momento de inicializar. Fuente: TIA Portal, Siemens, 2017.....	38

Figura 3.15: Tipos de bloques presentes para la programación en SCL. a) representación para los bloques de organización. b) representación de los bloques de función. c) representación para los bloques función. d) representación de los bloques de datos. Fuente: TIA Portal, Siemens, 2017..... 39

Figura 3.16: Estructura de anidamiento de los distintos bloques funcionales en un programa SCL. Fuente: TIA Portal, Siemens, 2017. .... 41

Figura 3.17: Interfaz gráfica para programar en lenguaje SCL dentro de TIA Portal. Fuente: TIA Portal, Siemens, 2017. .... 41

Figura 3.18: Interfaz de código escrito en SCL bajo el entorno de programación de TIA Portal. a) espacio de programación y sus diferentes partes. b) mensajes contextuales entregados por TIA Portal Fuente: TIA Portal, Siemens, 2017. .... 42

Figura 3.19: Árbol de proyecto y detalle de importación de código desde fuentes externas. a) opción de importación dentro del elemento PLC en el árbol de proyecto. b) pasos para generar código desde archivo externo en TIA Portal. Fuente: TIA Portal, Siemens, 2017. .... 43

Figura 3.20: Pantalla de TIA Portal para programación de dispositivos HMI. Fuente: TIA Portal, Siemens, 2017. .... 44

Figura 3.21: Fotografía de uno de los primeros pocket PC del mercado con Windows CE, Casio Cassiopeia A-11. Fuente: <https://www.flickr.com/photos/johndecember/2264105808>, 1997..... 45

Figura 3.22: Interfaz de TIA Portal para la programación en VBS. Fuente: TIA Portal, Siemens 2018. .... 46

Figura 3.23: Árbol de proyecto y de opciones dentro del HMI en TIA Portal. a) elemento HMI dentro del árbol de proyecto. b) detalle de opciones utilizadas mayormente para programar la interfaz en el HMI. c) detalle de opciones utilizadas para programar interfaz en HMI. Fuente: TIA Portal, Siemens, 2017. .... 47

Figura 3.24: Pantalla principal de TIA Portal. Fuente: TIA Portal, Siemens, 2017. 48

Figura 4.1: Esquema de los firmware presentes en el SM1281. Fuente: Industry Support, Siemens, 2018. .... 50

Figura 4.2: Representación bloque “SM1281\_Module” dentro de programación LADDER de PLC. Fuente: SIPLUS CMS1200 Operating instructions, Siemens 2017. .... 53

Figura 4.3: Representación bloque “SM1281\_Channel” dentro de programación LADDER de PLC. Fuente: SIPLUS CMS1200 Operating instructions, Siemens 2017. .... 53

Figura 4.4: Esquema del ordenamiento de los distintos modos de funcionamiento del SM1281. Marcado con “1” los modos por requerimiento de usuario. Fuente: SIPLUS CMS1200 Operating instructions, Siemens 2017..... 54

Figura 5.1: Esquema obtención archivo desde SM 1281 hasta HMI TP700. Fuente: elaboración propia. .... 57

Figura 5.2: Esquema ubicando wininet.dll dentro de modelo OSI. Fuente: “WinInet and the OSI Model”[68], Microsoft 2004. .... 61

Figura 5.3: Esquema de ordenamiento de bytes en un archivo WAV. Fuente: elaboración propia. .... 62

Figura 5.4: Esquema de los primeros 53 bytes en un archivo WAV. Fuente: elaboración propia. .... 63

Figura 5.5: Esquema de ordenamiento de bytes en un archivo WAV. Fuente: elaboración propia. ....	64
Figura 5.6: Ejemplos de los lenguajes de programación para PLC disponibles en TIA Portal. a) lenguaje de escalera. b) lenguaje de control estructurado. c) lenguaje de diagrama de bloque de función. d) lenguaje de lista de declaraciones e) lenguaje de control secuencial. Fuente: “Automating with SIMATIC S7-300 inside TIA Portal”, Hans Berger 2014.....	65
Figura 5.7: Esquema de ordenamiento de los programas desarrollados en MATLAB. Fuente: elaboración propia. ....	66
Figura 5.8: Esquema de ordenamiento de bytes en un archivo WAV. Fuente: elaboración propia. ....	69
Figura 5.9: Esquema Imagen principal de programa. Fuente: elaboración propia..	70
Figura 5.10: Ventana emergente con instrucciones para obtener archivo WAV. Fuente: elaboración propia. ....	71
Figura 5.11: Ubicación icono de Internet Explorer en escritorio de Windows CE. Fuente: elaboración propia. ....	71
Figura 5.12: Ingreso de usuario y clave en el navegador web. Fuente: elaboración propia. ....	72
Figura 5.13: Directorio de archivos WAV. Fuente: elaboración propia.....	73
Figura 5.14: Ejecución de descarga de un archivo WAV. Fuente: elaboración propia. ....	73
Figura 5.15: Interfaz de valores obtenidos por el cálculo de banda de Octava. Fuente: elaboración propia. ....	74
Figura 6.1: Disposición física de sensores sobre motor en maqueta ascensor. a) fotografía del motor junto con la caja de cambios montado en maqueta. Marcado en rojo los sensores instalados b) detalle de los sensores montados. A la izquierda sensor de la maqueta, a la derecha sensor S01. Fuente: elaboración propia.....	77
Figura 6.2: Señales de vibración de motor para viaje de bajada de cabina en maqueta de ascensor. a) señal obtenida con sensor PCB. b) señal obtenida con sensor S01. Fuente: elaboración propia.....	77

## ÍNDICE DE TABLAS

Tabla 2.1: Fallos típicos en máquinas rotatorias. ....	21
Tabla 2.2: Bandas de Octavas con su respectivas frecuencias. ....	22
Tabla 2.3: Tercio de banda de Octavas con su respectivas frecuencias para la banda -1. ....	22
Tabla 4.1: Tabla resumen versiones de software necesarios para programar un SM1281 en función de su firmware. Fuente: Siemens, 2018.....	50
Tabla 4.2: Resumen versiones elementos de hardware y firmware. ....	51
Tabla 5.1: Funciones de MATLAB utilizadas por cada función creada. ....	67
Tabla 5.2: Funciones matemáticas disponibles en Visual Basic Script. ....	68
Tabla 6.1: Ganancia en dB de señal a 159.2 Hz en banda de Octava de 125 Hz en a una velocidad de muestreo de 3.500 Hz. ....	76



## 1. INTRODUCCIÓN

La presente memoria describe el desarrollo realizado para el trabajo final del Máster Erasmus Mundus en Ingeniería Mecatrónica EU4M, titulado “Implementación de un Sistema de Monitorización Industrial: caso de uso Máquina Eléctrica” en el centro de investigación vasco IK4-Ikerlan. Para ello, se empieza con la descripción de la motivación para la realización de este trabajo final de máster (TFM), seguido del contexto y planteamiento del problema. Posteriormente será particularizado el objetivo general y los respectivos objetivos específicos necesarios para cumplir el desarrollo del presente trabajo. En conjunto, se plantea el problema que motiva el desarrollo de este trabajo, exponiéndose el interés de la empresa Orona. La empresa Orona se dedica a la fabricación, comercialización y mantenimiento de soluciones de transporte vertical, colaborando en sus desarrollos con Ikerlan. Actualmente existe en Ikerlan un Banco de Ensayos que simula un ascensor, donde se ensayan y ponen a punto diferentes tipos de algoritmos CMS. La complejidad de estos algoritmos precisa de una capacidad de cálculo propio de dispositivos tipo PC. El problema empieza al implementar estos algoritmos en un ambiente industrial, donde los PC no poseen la misma robustez que los PLC. Por ello, esta empresa se encuentra interesada en explorar las posibilidades de ejecución de estos algoritmos sobre elementos de menores prestaciones, pero mayor robustez. Por tanto, en este TFM se exploran las capacidades de los autómatas para incorporar algoritmos matemáticos que actualmente se ejecutan bajo el software MATLAB.

Seguidamente, en el capítulo dos se expone la teoría necesaria para entender los sistemas de monitorización de la condición (CMS por sus siglas en inglés), el análisis de las vibraciones mecánicas de los elementos presentes en la industria y como esta información es analizada. Posteriormente se explican los tipos de análisis existentes para obtener información relevante sobre el estado de los elementos y/o máquinas utilizando su señal de vibración. Una vez desglosadas las distintas analíticas existentes, se procederá a exponer las soluciones que ofrecen las tres empresas con mayor presencia en la industria vasca sobre este tipo de analítica, con su respectivo hardware y software.

En el capítulo tres se explica los elementos utilizados en el sistema desarrollado. Se individualizan los elementos de hardware y software utilizados, destacando los lenguajes de programación involucrados en el proceso de desarrollo de la solución final. Posteriormente, se expone la conexión de los elementos para obtener un sistema funcional.

El capítulo cuatro explica de qué forma se realiza la instalación y puesta en marcha de los dispositivos descritos en el capítulo tres, comentando los problemas enfrentados y las soluciones que aparecieron en el desarrollo del trabajo.

El capítulo cinco describe los caminos buscados para cumplir con el objetivo general, exponiendo cuales fueron descartados y los motivos para ello. Se expone la obtención de los datos de señal de vibración desde su generación hasta su

disposición para procesar por los algoritmos. Luego se describe de qué forma serán procesados estos datos.

El capítulo seis expone los resultados obtenidos, comparando los distintos algoritmos y contrastando resultados. Asimismo se muestra el proceso de validación del sistema implementado para cumplir el objetivo general.

Finalmente, las conclusiones generales y específicas del presente TFM son escritas en el capítulo cinco, terminando con la bibliografía y el Anexo.

### ***1.1. Motivación***

La monitorización de la condición es un proceso que está siendo valorado en los últimos años por las empresas de producción, al ser una herramienta que permite conocer el estado de las máquinas utilizadas en sus procesos, detectando con antelación la aparición de diferentes tipos de problemas o incluso de sus mismos productos en las instalaciones del cliente. El conocimiento del estado actual de las máquinas y su funcionamiento en un futuro próximo permite prever cuando los equipos utilizados comenzarán a presentar daños, pudiéndose traducir en millonarias pérdidas. Por otra parte, el correcto funcionamiento de estas máquinas es determinante para que los productos fabricados cumplan los estándares de calidad.

Las soluciones presentes en el mercado actual de las grandes casas de automatización permiten realizar una limitada analítica mediante las capacidades implementadas en sus propios sistemas, lo que limita a las empresas en el desarrollo de algoritmos que se adecuen a la detección de problemas específicos. Disponiendo de estos dispositivos (PLC, etc.) con limitadas capacidades, la pregunta a responder es la posibilidad de utilizarlos para realizar analíticas más evolucionadas que las propuestas por los fabricantes de estos equipos. Esta es la motivación del presente TFM.

### ***1.2. Contexto y planteamiento del problema***

El tema del presente TFM proviene de las necesidades solicitadas por la empresa Orona al centro de investigación IK4-Ikerlan, lugar donde se realiza este TFM. El grupo Orona es referente internacional en la fabricación, comercialización y mantenimiento de ascensores. Ikerlan es un centro tecnológico enfocado en la transferencia de conocimiento y aportación de valor competitivo a las empresas. Dentro del desarrollo de los productos de Orona, se busca incluir tecnología que permita conocer en todo momento el estado actual de sus maquinarias y elementos estructurales presente en sus ascensores. Por ello, la necesidad de usar CMS en sus instalaciones.

Aunque las grandes firmas de automatización disponen dentro de sus servicios dispositivos y programas para la realización de CMS, sus soluciones abarcan la mayor cantidad de situaciones presentes en el mercado, dificultando el desarrollo de

soluciones para escenarios más específicos, como la requerida por Orona en sus ascensores. En este escenario, se plantea entonces el estudio de las capacidades de estos dispositivos con el objetivo de saber si es posible elaborar herramientas informáticas que permitan obtener otros indicadores más específicos.

Por ello, se planea conocer las soluciones propuestas por Siemens por ser la marca que habitualmente utiliza la empresa Orona. Sobre estos dispositivos, se pretende documentar las capacidades que presentan, a fin de valorar la posibilidad de integrar un algoritmo propio. Como análisis de caso se ha elegido la composición de un programa creado en MATLAB que calcula la ganancia en dB de una banda de octava de una señal de vibración. Dado que MATLAB utiliza algoritmos matemáticos que se ejecutan sobre un computador de escritorio, el desafío consiste en adaptar este programa a los recursos menores de cómputo que dispone un PLC.

### ***1.3. Objetivos***

A continuación serán presentados el objetivo general del presente TFM y los correspondientes objetivos específicos para cumplirlo.

#### ***1.3.1. OBJETIVO GENERAL***

El presente trabajo final de máster (TFM) busca como objetivo general analizar la factibilidad técnica de integrar técnicas avanzadas de monitorización de la condición de elementos o máquinas de la industria en dispositivos robustos de uso habitual en ambientes industriales que habitualmente disponen de capacidades de cálculo menores a los disponibles en un ordenador de escritorio.

#### ***1.3.2. OBJETIVOS ESPECÍFICOS***

Los objetivos específicos necesarios para avanzar hasta el objetivo general enunciado previamente son:

- Aprendizaje de las capacidades hardware de dispositivos de la marca Siemens para la adquisición de datos de vibración y monitorización de estado de la condición.
- Conocimiento de la programación de los dispositivos de la marca Siemens para su correcta comunicación y posterior uso.
- Comprensión de las ventajas y limitaciones en el uso del lenguaje de programación SCL (Structured Control Language), según norma IEC 61131-3 para autómatas de Siemens.

- Configuración del autómata PLC S7-1212C y el módulo de monitorización de la condición SIPLUS CMS SM1281 de Siemens, para la obtención de datos de vibración de una máquina eléctrica.
- Configuración del autómata PLC S7-1212C y el HMI TP700 Comfort Panels de Siemens para desplegar información y gráficas de datos.
- Elaborar un programa en el lenguaje de programación VBScript para el manejo de archivos a nivel de byte.
- Analizar las ventajas y desventajas de la herramienta Simulink© PLC Coder de MATLAB.
- Comparación de resultados obtenidos con algoritmo en MATLAB, SCL y VBS.
- Validación de algoritmos desarrollados con datos reales de máquina eléctrica.

## 2. ESTADO DEL ARTE

Para entender el estado de la condición de las máquinas, es necesario saber las fuentes de información disponibles. Dentro de los métodos posibles para obtener información sobre las maquinas y dispositivos, el uso del análisis de emisiones ultrasónicas se fundamenta en que ciertos fenómenos van acompañados de emisiones acústicas por encima de las frecuencias del rango audible [1]. Los detectores de ultrasonido pueden ser utilizados para la detección de fugas de fluidos en conductos, verificación de purgadores de vapor, inspección mecánica de rodamientos, reductoras, inspecciones eléctricas en armarios eléctricos, transformadores, aisladores y largo etcétera.

El analizar los lubricantes utilizados en las máquinas permite determinar el deterioro del lubricante, la entrada de contaminantes y la presencia de partículas de desgaste [2]. Actualmente existen equipos de taller para el análisis de aceite que permiten montar un laboratorio pequeño rápidamente para el estudio de aceites en la planta industrial, lo que permite obtener resultados inmediatos, reduciendo costes de análisis por muestra. Los equipos de taller para el análisis rápido pueden medir, entre otros indicadores: índice de detracción química, constante dieléctrica, contenido en agua, índice de desgaste férrico (para conocer el desgaste en rodamientos y engranajes), indicador de partículas no férricas (verificación de entradas contaminantes), viscosidad, entre otros.

Determinar temperaturas a distancia sin necesidad de contacto físico mediante la captación infrarroja del espectro electromagnético es el concepto básico de las cámaras infrarrojas [3]. Esta técnica se ha ido extendiendo durante más de 20 años desde aplicaciones médicas o militares hasta el ámbito industrial. En la mayoría de situaciones se aplica a la inspección de armarios eléctricos, aunque también hay muchas otras aplicaciones: inspección de motores eléctricos para buscar calentamientos localizados por fallos en el estator, calentamiento de cojinetes y rodamientos debido a malas lubricaciones o daños en las pistas de rodadura entre otras muchas otras aplicaciones. Otras técnicas a mencionar pueden ser: análisis de las trazas de presión de cilindro, descargas parciales para máquinas eléctricas, técnicas para el análisis de corriente y tensión de alimentación en motores eléctricos de AC.

A día de hoy, lo que se utiliza en gran medida y a lo largo de la presente memoria es el fenómeno de la vibración. La vibración es un movimiento oscilatorio y reiterado de un sólido, como una máquina o estructura, alrededor de una posición de referencia[4]. Generalmente, no es perceptible visualmente pero si al tacto. Las vibraciones en una máquina pueden originarse producto de fuerzas que actúan de forma interna o externa. Las vibraciones producto de fuerzas internas pueden ser provocadas por situaciones como desbalance o desalineación en sus ejes, rodamiento en mal estado, etc. Las vibraciones producto de las fuerzas externas son aquellas que provienen del ambiente donde se encuentra la máquina.

Para medir el fenómeno de la vibración en un objeto se utilizan transductores, los cuales deben encontrarse rígidamente unidos a la superficie de cuerpo a medir. De

esta forma se trasmite la vibración hacia el elemento sensor, produciéndose la medición, como se aprecia en la figura 2.1.



Figura 2.1: Esquema de medición de vibración directamente sobre equipo. Fuente: Fluke.

## **2.1. La vibración y sus efectos en Máquinas e Industria**

Cuando existen elevadas vibraciones en el funcionamiento de una máquina, están pueden estar causado por variadas situaciones, en las que se pueden mencionar:

- Errores de fabricación: Producto de materiales no adecuados, baja calidad de los componentes, uso de componentes inadecuados, rotor no centrado, el desbalance, entre otros.
- Errores de montaje: Una de las principales causas de las vibraciones en las máquinas se debe a errores al momento del montaje. Estas situaciones pueden ser un mal alineamiento, soltura, distorsión, entre otros.
- Defectos estructurales y de materiales: El desgaste normal, daño estructural y abusos en el uso de la máquina que pueden modificar su función, causando en principio vibraciones.
- Lubricación: Los engranajes y rodamientos desarrollan defectos metálicos localizados después de años, originado por carga y descarga o la presencia de materiales extraños y principalmente por una lubricación deficiente. La lubricación deficiente genera ruido y un desgaste acelerado de los rodamientos. Tales defectos causan vibraciones impulsivas que indican deficiencia del rodamiento.
- Desbalance: El desbalance del motor ocasiona altas vibraciones, repercutiendo directamente en el desempeño del motor.

Definido lo que son las vibraciones, es propicio entender el resultado de las excesivas vibraciones en una máquina. En esta situación la máquina en cuestión presenta un movimiento mayor al estipulado por diseño, lo cual deriva en el rozamiento de sus distintos elementos, los cuales puede dañar juntas selladas y causar daño incipiente en los rodamientos. Los efectos de las vibraciones pueden ser resumidos en los siguientes tipos[5]:

- Daño por fatiga: La fatiga se da a altas frecuencias, es decir, un gran número de ciclos en el tiempo. Asimismo, cuando las fuerzas son suficientemente grandes, el daño puede ocurrir como resultado de la fatiga de baja frecuencia.
- Pérdida de la calidad del proceso: Los procesos puede ser afectados negativamente por la excesiva vibración, como puede ser el caso de una imprenta; el maquinado y revestimiento de los papeles se ve afectado con la presencia de vibración, no presentando el producto final la calidad requerida.
- Salud del personal: Las vibraciones pueden afectar a la salud del personal que trabaja cerca de la máquina, tanto por el ruido que pueden generar como por la propia vibración si hay contacto físico entre el operario y la vibración.

Dados los casos expuestos con anterioridad, es posible ver la importancia de la monitorización de la condición de los equipos en la industria, ya que permite evitar pérdidas económicas derivadas de las paradas no programadas de planta. Asimismo, la necesidad de repuestos en almacén para acortar los tiempos de parada no programada aumentan los costos al tener que contar con los elementos de repuesto inmovilizados en el almacén, incrementando los costos de la parada no programada. Se puede evitar estas paradas no programadas analizando las vibraciones de la máquina, que en su sencilla consiste en captar los datos de las vibraciones por medio de un equipo analizador de vibraciones. Este equipo obtendrá la señal de vibración directamente de un transductor conectado a la máquina, siendo luego esta señal procesada por el software incorporado por el mismo dispositivo que adquiere la señal. De esta forma, se obtiene el espectro de las vibraciones en el dominio de la frecuencia. La interpretación correcta de esta información podrá determinar el estado actual de los elementos que componen la máquina y el estado general de la vida útil de la máquina.

Dentro de las máquinas usadas en la industria, las máquinas rotativas son de especial importancia dado el amplio rango de procesos en los que son utilizadas[6], [7]. Las máquinas rotativas producen vibraciones por variadas fuentes. La mayoría de estas se encuentran relacionadas con la velocidad de giro, siendo múltiplos o submúltiplos de ésta [8]–[10]. Aún así, también pueden aparecer otras frecuencias que no están relacionadas a esta velocidad. Cada fuente produce su propia frecuencia única, entregando un patrón de frecuencia determinado[11], [12].

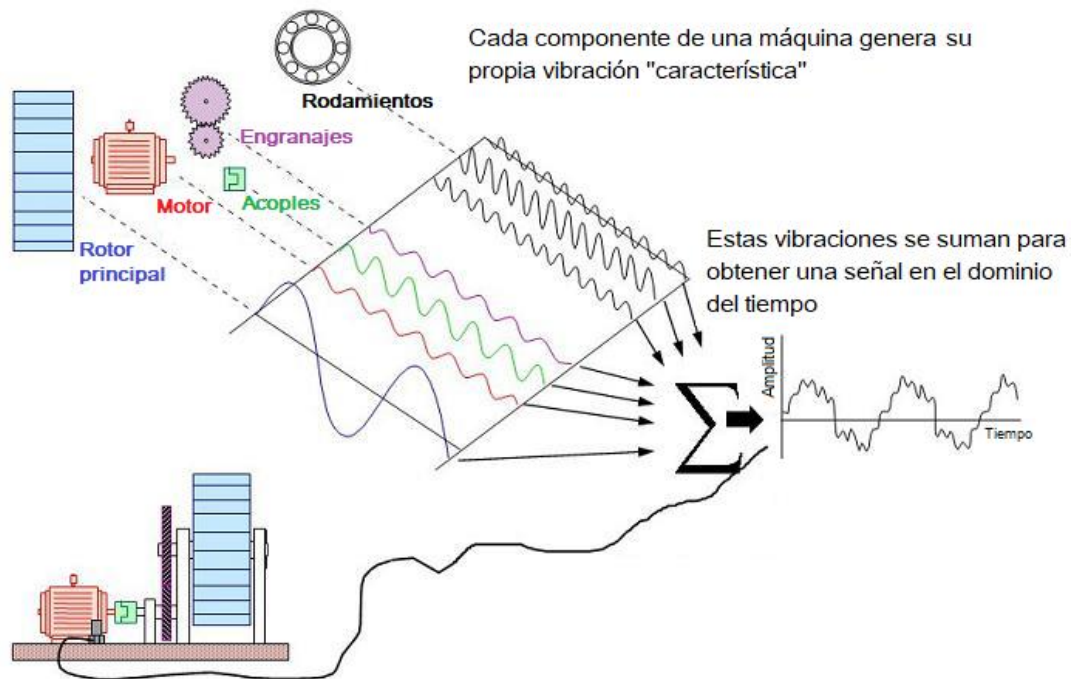


Figura 2.2: Captura de la señal de vibración de un motor eléctrico y como las distintas vibraciones son sumadas en una única señal de vibración. Fuente: Elaboración propia.

Cuando se miden las vibraciones de una máquina, el sensor o transductor lee la combinación de todas ellas. La Figura 2.2 expone gráficamente como se combinan una gran variedad de fuentes de vibración. Cada una de estas vibraciones posee una componente (frecuencia) fundamental y múltiplos de ésta. Por ello, la vibración a la velocidad de giro se indica como “1X” componente fundamental, “2X” se refiere a la velocidad de giro por dos veces la fundamental, “3X” se refiere a la velocidad de giro por 3 veces la fundamental, etc. Todas estas señales son combinadas y leídas a través del tiempo por el sensor a una velocidad de muestreo adecuada para realizar luego los análisis respectivos.

## 2.2. Rodamientos y su estructura

Dentro del análisis de los elementos presentes en las máquinas rotatorias, son de especial importancia el análisis de los rodamientos. Los rodamientos son elementos mecánicos que emplean pequeños elementos rodantes, cuyo objetivo es disminuir la fricción entre las piezas conectadas a un eje y al eje mismo[13]. Esto es posible dado que la resistencia de fricción por rodadura es menor que la resistencia de fricción por deslizamiento[14]. La figura 2.3 muestra las distintas partes que componen un rodamiento.



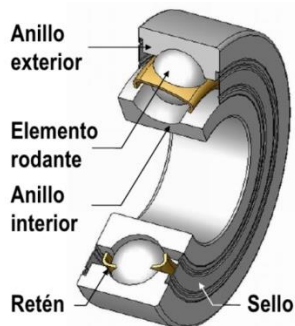


Figura 2.3: Esquema general de un rodamiento y las partes que lo componen. Fuente: "Fatiga De Ejes, Soportes y Ajustes", Julio Vergara Aimone ,2017 PUC.

La importancia que tienen los rodamientos en la industria con el pasar de los años se ha vuelto tan alta, que un daño catastrófico en un rodamiento provoca una pérdida en la producción por el tiempo de paradas necesario para su arreglo, junto a los gastos de mantenimiento y compra de repuestos. Por ello, la detección del estado en que se encuentran los rodamientos de una máquina y previendo los daños incipientes que pueden presentarse evita estos gastos. Según datos estadísticos [15], es posible afirmar que el porcentaje de rodamientos dañados en el lugar de su fabricación es muy bajo, debido a que la mayoría de empresas que fabrican rodamientos tienen un estricto sistema de control de calidad. Por tanto, la mayor fuente de fallo de los rodamientos reside en su uso indebido. La figura 2.4 muestra los datos estadísticos tomados de diferentes marcas en el centro técnico de NSK, empresa japonesa que fabrica rodamientos a nivel mundial.

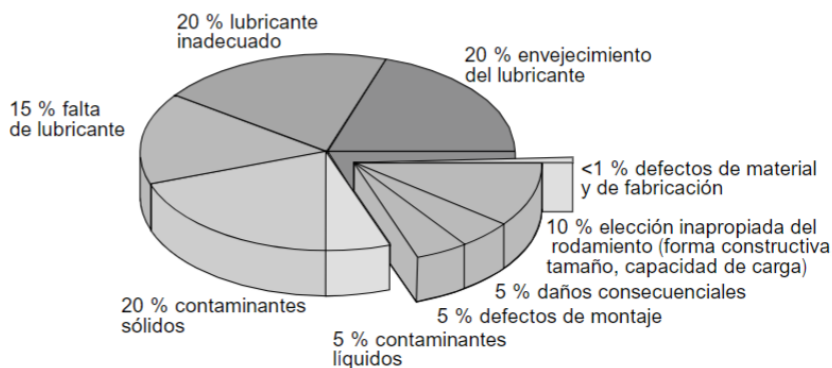


Figura 2.4: Datos estadísticos de daños en rodamientos a nivel mundial. Fuente: Averías de los rodamientos, FAG, 2002.

De la figura anterior, se extrae que existen 2 grandes fuentes de daños en los rodamientos comerciales: contaminación y problemas de lubricación, existiendo además otro conjunto que provoca error de menor influencia. Entre las causas de fallo en el rodamiento por contaminación, tenemos desde contaminaciones en estado líquido por presencia de agua en el lubricante del rodamiento a contaminaciones sólidas, como partículas metálicas que son arrastradas por el lubricante en el interior. Existen problemas de lubricación cuando la grasa lubricante es demasiado

dura para la aplicación que se requiere, o también cuando existe deficiencia de lubricante, haciendo que los rodamientos disminuyan su vida útil. En el último grupo se aúnan las anomalías que afectan al correcto funcionamiento del rodamiento. Entre estos está el mal montaje, maltrato del rodamiento, mal almacenamiento y mal ajuste del rodamiento en su alojamiento. En la Figura 2.5 podemos ver un caso de daño por defecto de montaje, donde el rodamiento se encuentra apretado en el eje, por lo cual las bolas desgastan la pista interna del rodamiento.



Figura 2.5: Daño por defecto de montaje. Fuente: <http://www.blogmecanicos.com>.

### 2.3. Vibración en rodamientos e indicadores

Para comprender como se relacionan las variables involucradas en el análisis de vibraciones, la figura 2.6 muestra como las amplitudes de las variables de vibración (desplazamiento, velocidad y aceleración) cambian a medida que aumenta la frecuencia[16]. Este diagrama proporciona información sobre las frecuencias hasta las cuales la medición y evaluación de una determinada variable de vibración puede proporcionar datos significativos.

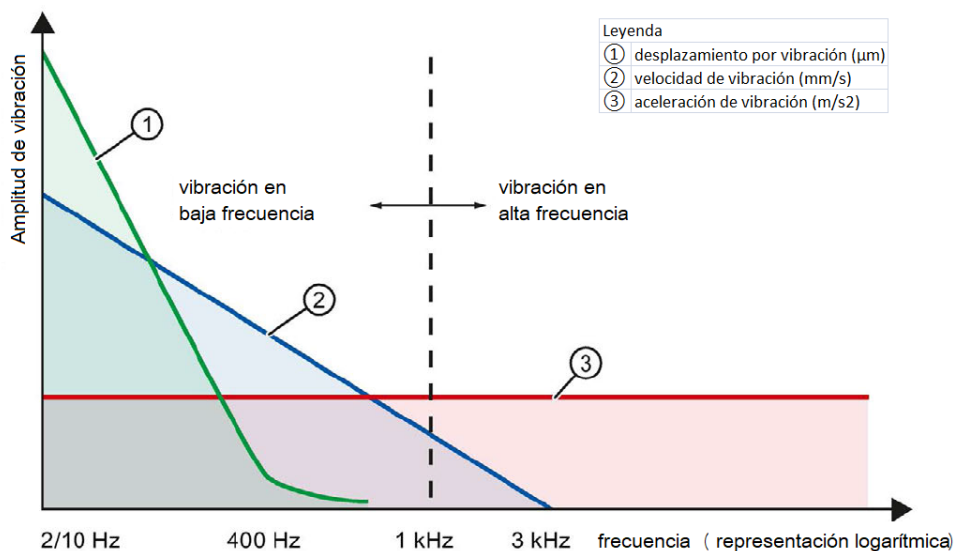


Figura 2.6: Relación entre la amplitud de una señal y su frecuencia. Fuente: CMS1200, Operating instructions, Siemens, 2017.

Como puede apreciarse en la figura anterior, la información proporcionada por las vibraciones de desplazamiento se ubican generalmente entre los 1 y 400 Hz. Generalmente estas vibraciones se producen en los ejes. Por otra parte, las vibraciones generadas por las velocidades se producen usualmente en los cerramientos, comprendiéndose en el rango de los 2-10 Hz a 1k Hz. Finalmente, las vibraciones generadas por las aceleraciones pueden encontrarse entre los 2 a 20k Hz, correspondiendo a la caja de engranajes y al ruido de estructura.

Habitualmente, la condición de las máquinas es monitorizada adquiriendo valores característicos, con los cuales se evalúa la condición de vibración general de la máquina. Las tendencias de estas variables indican si la condición está empeorando, es decir, si aparece un daño incipiente. Variables típicas calculadas típicamente son  $v_{RMS}$  (valor RMS de la velocidad) y  $a_{RMS}$  (valor RMS de la aceleración). Además de estos valores, también se considera el valor  $K(t)$  según la norma VDI 3832 [17], el cual permite evaluar el estado de rodamientos. Este indicador se encuentra formado por los valores pico y efectivo de la aceleración de vibración en el rango de frecuencia de 1k a 10k Hz. El filtro de paso alto de 1k Hz garantiza que ruidos estructurales, como por ejemplo fuerzas de desequilibrio, no se superpongan a la señal de rodamiento. El valor  $K(t)$  se expresa como:

$$K(t) = \frac{a_{rms}(0) \cdot a_{peak}(0)}{a_{rms}(t) \cdot a_{peak}(t)} \quad (2.1)$$

Donde:

- $a_{rms}(0)$  es el valor efectivo en el momento temporal de inicio de uso o de referencia.
- $a_{peak}(0)$  corresponde al valor máximo (valor peak) en el momento temporal de inicio de uso o de referencia.
- $a_{rms}(t)$  es el valor efectivo actual.
- $a_{peak}(t)$  el valor máximo actual (valor peak).

Si  $K(t)$  tiene un valor mayor a 0,5, significa que la condición del rodamiento esta en un buen estado. Si se encuentra entre 0,5 y 0,2, ya hay una influencia que incrementa el deterioro. En el rango 0,2-0,002 el proceso de deterioro ya se encuentra en un nivel avanzado, y para valores menores a 0,002 el deterioro es efectivo. Por tanto, el valor se reduce a medida que el daño aumenta. Si definimos DKW como el recíproco de  $K(t)$ , tendremos un valor que aumenta a medida que el daño se incrementa, siendo un indicador que refleja proporcionalmente la misma información.

El desarrollo de la condición de la máquina puede ser comprobado comparando las medidas actuales con medidas históricas o comparando con valores guías o datos de fabricante. Con este análisis de tendencia, el empeoramiento de las condiciones puede ser detectado con suficiente antelación y las medidas apropiadas pueden ser planeadas e implementadas.

Tanto el valor DKW como los valores  $a_{RMS}$  y  $v_{RMS}$  no son siempre suficientes para detectar las posibles averías o desperfectos que tengan las máquinas. Por eso, se suele realizar otros tipos de análisis para obtener mayor información. En esa línea, muchos tipos de daños son reconocibles en el espectro de frecuencia, por su patrón espectral típico. Los espectros de la velocidad y aceleración de la vibración junto al espectro de la envolvente son utilizados para este fin. La tabla 2.1 indica los fallos típicos que pueden ser detectados de esta forma, con su correspondiente indicador.

La señal de vibración emitida por un rodamiento puede contener componentes espectrales que están relacionados con la geometría del rodamiento, el número de elementos rodantes, la velocidad de rotación, la lubricación del defecto y el tipo de carga aplicada. Es de gran importancia en la industria moderna la detección de estas vibraciones para detectar defectos en los rodamientos. Para un ojo experimentado, el reconocer daños con solo observar el gráfico de amplitud versus frecuencia es posible, ya que los daños se caracterizan por tener formas muy propias. La frecuencia a la que se produce la máxima amplitud da una idea del elemento defectuoso del rodamiento. Las frecuencias de paso son generadas cuando las bolas o rodillos pasan por algún defecto en las pistas de rodadura. La frecuencia generada en la pista externa es aproximadamente igual al 40 % del producto del número de bolas por la frecuencia de rotación, la frecuencia generada en la pista interna se aproxima al 60 % del producto anterior. Se debe al 40 % de las bolas pasan por un defecto en la pista externa durante cada revolución y un 60 % de ellas lo hace por la pista interna en cada revolución [13].

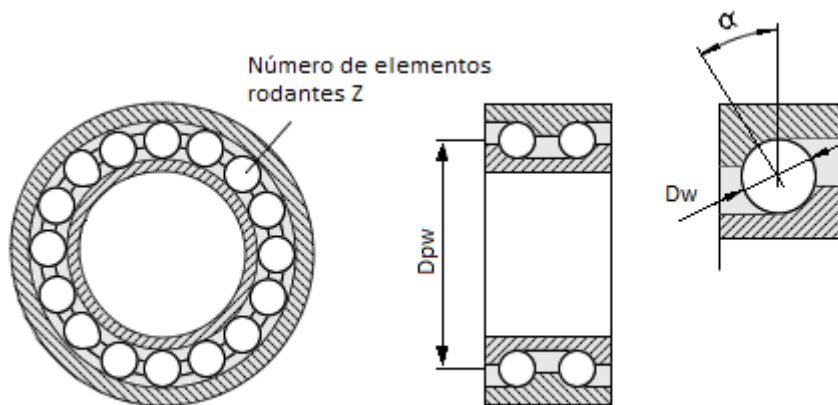


Figura 2.7: Distintas métricas de un rodamiento que afectan a las componentes espectrales de vibración. Fuente: página web SM1281, Siemens, 2018.

Tabla 2.1: Fallos típicos en máquinas rotatorias.

Tipo de Fallo	Medición en rango de tiempo	Espectro de análisis de Frecuencia		
		velocidad de vibración	aceleración de vibración	curva de la envolvente
desbalance	RMS	frecuencia rotación simple $f_n$	-	-
desalineamiento	RMS	frecuencia rotación simple $f_n$	-	-
defectos de acoples	RMS	frecuencia de rotación doble $f_n$	-	-
defectos de montaje	RMS	frecuencia de rotación simple doble y triple $f_n$	-	-
frecuencia de paso de aspas	RMS	$f_{SP} \leq 1\text{ k Hz}$	$f_{SP} > 1\text{ k Hz}$	-
defecto de mallado	-	$f_Z \leq 1\text{ kHz}$	$f_Z > 1\text{ kHz}$	-
defecto de cinta	RMS	$f_R \leq 1\text{ kHz}$	$f_R > 1\text{ kHz}$	-
resonancia	RMS	frecuencia de rotación igual a la frecuencia de rotación $f_n$	-	-
desgaste cojinete	DKW	-	$3\text{ kHz} \leq f_{LE} \leq 10\text{ kHz}$	-
frecuencia de daño de cojinete	DKW	-	-	dependiente geometría rodamiento
daño estator eléctrico	RMS	doble línea en frecuencia $f_{line}$	-	-
daño rotor eléctrico	RMS	$f_{bar} \leq 1\text{ kHz}$	$f_{bar} > 1\text{ kHz}$	-
rotura barra del rotor	RMS	línea doble en frecuencia $f_{line}$ y modulación con frecuencia de deslizamiento $f_{slip}$	-	-

## 2.4. *Banda y Tercio de Octava*

Dentro del análisis de frecuencias, no siempre es necesario realizar un análisis de todo el espectro de frecuencia, sino que en ocasiones es suficiente analizar un cierto rango de valores. El análisis de un rango de frecuencias para obtener los valores necesarios permite reducir la cantidad de cálculo necesario, lo que se traduce en un ahorro de tiempo (junto con sus respectivas compensaciones económicas).

Las bandas de octava se denominan a los intervalos de conjuntos de frecuencias agrupados de valores fijos y conocidos, con el fin de analizar una región específica del espectro de frecuencias [18]. Estos rangos presentan la característica de que la frecuencia límite inferior es la mitad de la frecuencia del límite superior. Se caracteriza cada banda con su valor central, siendo por ende la media geométrica de ambos límites. La tabla 2.2 presenta bandas con sus respectivos límites.

Tabla 2.2: Bandas de Octavas con su respectivas frecuencias.

Numero de Banda	Frecuencia límite inferior [Hz]	Frecuencia Nominal [Hz]	Frecuencia límite Superior [Hz]
-1	11	16	22
0	22	31.5	44
1	44	63	88
2	88	125	176
3	176	250	352
4	352	500	704
5	704	1k	1408
6	1408	2k	2816
7	2816	4k	5632
8	5632	8k	11264
9	11264	16k	22528

Cada una de estas bandas pueden ser divididas en otras 3 bandas de rango más reducido llamadas tercios de octava [19]. Esto permite ahorrar más procesamiento en frecuencias que no son de interés. Para definir estas bandas, la frecuencia de corte superior es la frecuencia inferior multiplicado por la raíz cúbica de 2. En la tabla 2.3 se muestra los tercios de banda para la banda -1.

Tabla 2.3: Tercio de banda de Octavas con su respectivas frecuencias para la banda -1.

Frecuencia límite bajo [Hz]	Frecuencia central [Hz]	Frecuencia límite superior [Hz]
11.2	12.5	14.1
14.1	16	17.8
17.8	20	22.4

## ***2.5. Sistemas de Monitorización de la Condición***

Los sistemas de monitorización de la condición (CMS por sus siglas en inglés) son aquellos procesos que adquieren de forma continua variables de funcionamiento de una máquina (como puede ser la temperatura, las vibraciones, etc.) con el fin de identificar posibles cambios en su desempeño [20]. Estos cambios pueden ser un signo de posibles daños o deterioros tempranos en distintos sistemas, por lo que convierte estos sistemas o métodos en el componente de mayor importancia dentro del mantenimiento predictivo (entendido como el conjunto de técnicas diseñadas para ayudar a determinar la condición de servicio de un equipo con el objetivo de indicar cuándo se debe dar el mantenimiento para que resulte óptimo desde el punto de vista de los recursos económicos y personal [21]). El uso de estas técnicas permite planificar distintos mantenimientos o revisiones sobre las máquinas utilizadas en la industria de forma más eficiente, ayudando a prevenir posibles fallos en los tiempos de funcionamiento y evitando paradas de emergencia. También permiten prever daños que truncarían el ciclo de vida normal de la máquina.

Existen diversos fabricantes dentro del mundo de la automatización y cada uno de ellos enfoca sus CMS desde su propio punto de vista.

Las técnicas de CMS son utilizadas en todo tipo de equipos y máquinas, destacando aquellos que poseen partes rotativas, como bombas, motores eléctricos, motores de combustión interna, prensas, etc. La clave para realizar un buen proceso de CMS reside en:

- Saber qué parámetros son convenientes de controlar.
- Conocer cómo interpretar dichos parámetros.
- Combinar todos los datos obtenidos para conseguir un correcto uso.

Los métodos CMS pueden ser aplicados con distintos grados de intensidad en un proceso. Si un proceso es muy crítico (proceso continuo), se precisa una monitorización más exhaustiva de las condiciones de la máquina para evitar su parada por fallo. Para un proceso menos importante es posible realizar una monitorización de las condiciones más espaciada en el tiempo, ya que el fallo no influirá en la misma medida en el desempeño global de la instalación. A esto se conoce como grado de criticidad.

El CMS basa su funcionamiento en la información extraída de los sensores y sistemas de sensorización, enfocándose de las siguientes formas [22]:

- Sensorización directa: se miden de forma directa las variables de interés, por lo que se necesita sensorizar la instalación misma. La ventaja principal radica en que la medición se realiza en el lugar del fenómeno, lo cual permite tener un grado de confianza sobre el dato del sensor. Las desventajas principales son los costes de los sensores utilizados, fallos en sensores, dificultades de instalación, entre otros.
- Sensorización indirecta: la medición de las variables físicas se realiza sin una intrusión en la instalación a diferencia del anterior caso. Esto es de especial interés en instalaciones peligrosas.

- Sensorización basado en sensores virtuales “*virtual sensing*”: parte de información disponible de otras medidas y modelos físicos para estimar las variables de interés [23].

En cuanto a la forma en que se utiliza la información disponible, podemos diferenciar los siguientes enfoques:

- Análisis de grandes volúmenes de datos “*data driven*”.
- Análisis de modelos físico.
- Modelos híbridos.

En el análisis de grandes volúmenes de datos se emplean modelos de aprendizaje estadísticos, para lo que se necesita un conjunto de datos experimentales con que producir un modelo. Ejemplo de este tipo son las redes neuronales y la clusterización [24]. Para el análisis de modelos físicos se requiere disponer de un modelo físico-matemático que describa el comportamiento del sistema. Existen técnicas que se basan en filtros adaptativos y observadores, los cuales estiman parámetros de interés utilizando las medidas y el conocimiento sobre el modelo del sistema [25], [26]. Finalmente, en los modelos híbridos se utilizan métodos basados en la combinación de los dos anteriores.

## **2.6. *Sistemas de Monitorización de la Condición en el Mercado***

Existe una gran cantidad de soluciones y fabricantes distintos que proporcionan sistemas de monitorización de la condición en la industria. A fin de tener una visión del conjunto de los mismos se han analizado las soluciones implementadas por tres empresas cuyos equipos tienen mayor presencia en la industria del País Vasco: Allen – Bradley, Siemens y Beckhoff.

Dentro del CMS, Allen-Bradley se centra en ofrecer soluciones y protecciones en tiempo real. Además, busca como objetivo que sus sistemas de adquisición de datos (junto a su software) sean capaces de manejar bruscos en las condiciones de funcionamiento de la máquina.



Figura 2.8: Aspecto físico del dispositivo Dynamix 1444 Integrated Condition Monitoring. Fuente: Allen-Bradley, 2017.



El sistema que ofrece este fabricante para realizar CMS es el Dynamix 1444 Integrated Condition Monitoring [27], el cual integra protección para la máquina en un sistema de control estándar. Al ser utilizado en ambiente industrial, posee un sistema robusto capaz de soportar hasta 70°C, descargas eléctricas estandarizadas y vibraciones, siendo posible también su uso en áreas de riesgo. El software que proporciona Allen-Bradley para CSM se denomina el Emonitor [28] y permite la vista de datos en tiempo.

Beckhoff proporciona un toolbox muy completo de hardware y de software para el pesaje, medición de espectros de potencia, medida de potencia, tensión y corrientes, presión, temperatura, vibración, entre otras. De todas estas variables, Beckhoff da especial importancia al control de vibraciones, teniendo a disposición un módulo de 2 canales analógicos [29]. Como apoyo a sus productos comerciales posee además el software TC3 Database Server, el cual permite interactuar con distintos tipos de bases de datos sin mayor inconveniente. Dentro de software enfocado al CMS, Beckhoff presenta un paquete dentro de su plataforma TC3 Scope View [30], el cual pretende ser tan poli funcional como un osciloscopio.

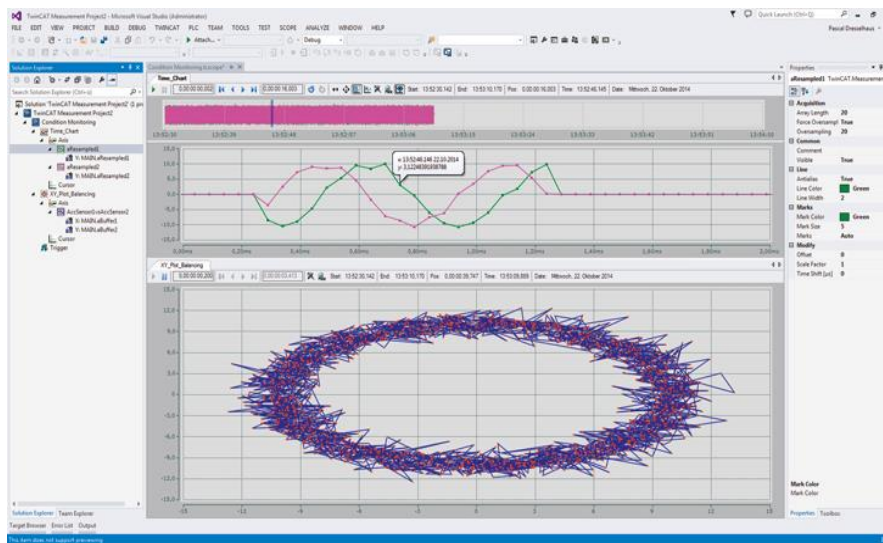


Figura 2.9: TC3 Scope View Professional en ejecución. Fuente: Beckhoff, 2017.

Por parte de Siemens, la línea que cubre las necesidades de CMS para la industria es llamada SIPLUS CMS [31]. Para ello combina sistemas y entornos de programación de sus equipos con MindSphere – Siemens Cloud for Industry [32]. El MindSphere es un ecosistema de trabajo basado en una plataforma abierta en el cual los usuarios y desarrolladores pueden utilizar para desarrollar, extender y operar con aplicaciones en la nube. Esta solución proporciona la base para el desarrollo de aplicaciones y servicios basados tanto en los datos de la empresa como de terceros, en campos como el mantenimiento predictivo, gestión de datos de energía u optimización de recursos. Es una herramienta que posibilita múltiples servicios diseñados ad-hoc para las empresas manufactureras que quieren

emprender el camino hacia la Industria 4.0. El SIPLUS CMS1200 está basado en SIMATIC S7-1200, un autómata programable que apoya el uso del módulo CMS1200 que permite corroborar continuamente el estado de los componentes mecánicos. Este sistema hace un aporte crucial para conseguir la disponibilidad y funcionamiento de una planta de forma continua. El presente TFM plantea el desarrollo de herramientas propias de CMS utilizando los equipos propuestos por Siemens.



Figura 2.10: CMS 2000 VIB de Siemens. Fuente: Siemens, 2017.

### 3. DISEÑO DE SISTEMA

En capítulos anteriores fue descrito el planteamiento del problema para contextualizar el TFM y el estado del arte. A continuación se describen los elementos utilizados en el sistema desarrollado.

En la figura 3.1 se ve el diseño del sistema, el cual es capaz de captar los datos de vibración de máquina eléctrica, tratarlos y extraer de ellos las características necesarias para monitorizar la condición de la misma.

El elemento principal de este sistema, capaz de gestionar el flujo de información en el conjunto del sistema es el PLC S7-1200 (en adelante llamado PLC), el cual tiene anexo el módulo SM1281 (llamado en adelante SM), diseñado para captar señales y procesarlas para CMS junto al PLC. El SM tiene conectado directamente un sensor de aceleración S01 que obtiene la señal de vibración. Este sensor de aceleración está colocado sobre un motor eléctrico accionado con un convertidor de frecuencia. Este motor eléctrico se encuentra montado dentro de una maqueta a escala de un ascensor tipo M34 de suspensión 1:1.

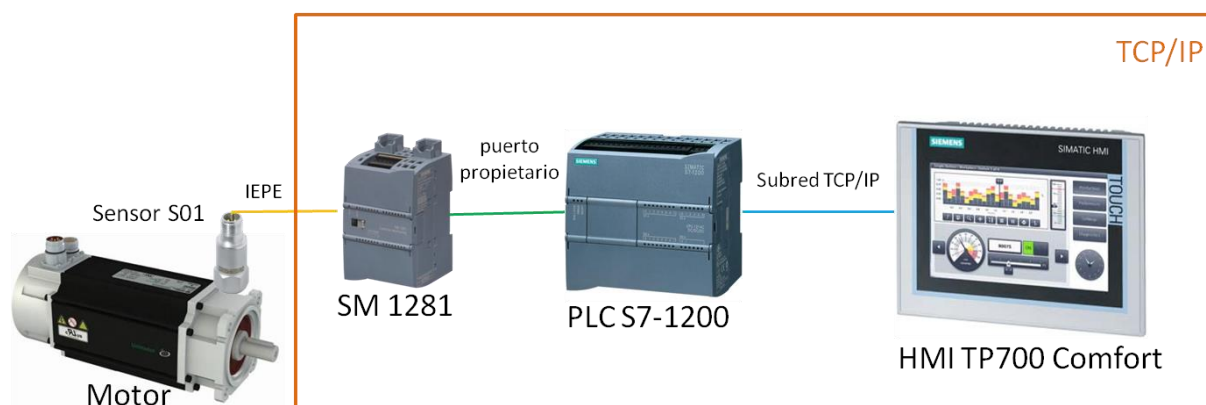


Figura 3.1: Esquema general conexionado elementos. Fuente: elaboración propia.

El sensor S01 adquiere las vibraciones presentes en el motor, se conecta al módulo de adquisición de señales SM [33] mediante estándar IEPE (electrónica integrada piezo-eléctrico por sus siglas en inglés) El SM se conecta a su vez al PLC S7-1200, mediante un bus de comunicaciones propietario de Siemens [34]. El PLC entrega los parámetros de configuración al SM 1281 por este bus propietario y obtiene los datos procesados por el SM. El PLC tiene la posibilidad de comunicar estos datos vía TCP/IP al dispositivo HMI P700 Comfort (llamado de ahora en adelante HMI). El HMI es una pantalla táctil que permite ver y modificar valores mediante una interfaz gráfica, prescindiendo de un computador portátil o de escritorio para observar los valores en tiempo real.

### 3.1. *Elementos de Hardware*

Para explicar los elementos utilizados en este sistema, se empieza con la descripción de los elementos físicos. Luego, serán descritos los entornos de programación utilizados para el PLC y el HMI; STEP7 y WinCC, ambos incluidos en el software propietario de Siemens TIA Portal. Los elementos físicos utilizados son los siguientes:

- PLC S7-1212C AC/DC/Rly.
- SIPLUS SM1281.
- Sensor S01.
- HMI TP-700.
- Motor eléctrico.

La disposición física de estos elementos (a excepción del motor eléctrico) se aprecia en la figura 3.2, donde se ve el panel con el conexionado de los distintos dispositivos Siemens.

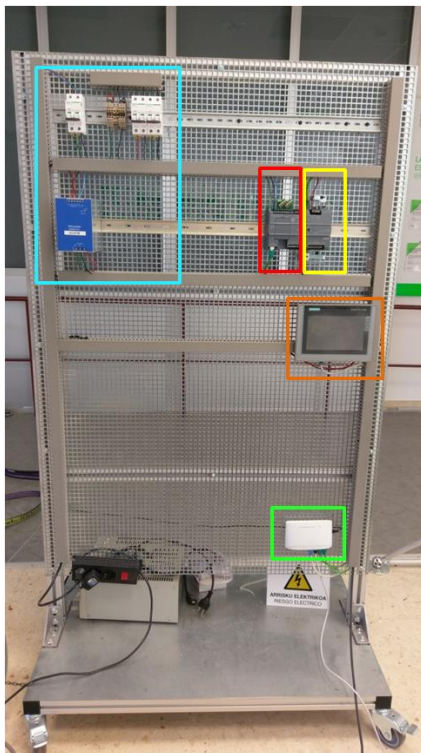


Figura 3.2: Disposición física final de los elementos del sistema en laboratorio.

En la figura anterior se encuentra destacado en rojo el PLC conectado físicamente al SM, destacado en amarillo. En naranja se encuentra la pantalla HMI. Todos estos elementos se encuentran conectados vía TCP/IP utilizando un Switch Ethernet destacado en verde a la red local de la empresa Ikerlan. Los elementos destacados en celeste pertenecen al conexionado eléctrico encargado de alimentar con  $220 V_{ac}$  para el PLC y  $24V_{dc}$  para el SM y el HMI.

### 3.1.1. PLC S7-1212C

El PLC utilizado corresponde a la familia S7-1200, la cual tiene como modelo de CPU 1212C AC/DC/Rly [35]. Dispone de 8 entradas digitales de 24V, 6 salidas digitales de relé y 2 entradas analógicas. La tensión de alimentación admisible del dispositivo puede variar entre 85 y 264V<sub>ac</sub> con un rango de frecuencia de 47 a 63 Hz. Tiene una memoria de programas de 2 Mb y de datos de 75 Kb. Dispone de un puerto PROFINET para la comunicación y programación utilizando el software STEP 7 Basic incluido en TIA Portal, pudiendo usar los lenguajes estandarizados de escalera (LAD), diagrama de bloque de función (FBD) y lenguaje de control estructurado (SCL). Su configuración permite además anexar distintos módulos que amplían sus capacidades de comunicación y de ampliar el número de entradas/salidas digitales y analógicas (con un máximo de 3 bloques los cuales se conectan a su izquierda). El puerto con el cual se comunican estos módulos con el PLC es propietario. No dispone de botones físicos para poner en marcha o inicio, realizándose estas funciones a través del software TIA Portal y un elemento externo (PC). Así mismo, posee integrado una página web con la que es posible monitorizar el estado del PLC, el valor de las variables de su programa y el estado de los módulos que se encuentran conectados.



Figura 3.3: Aspecto real SIMATIC s7-1200. Fuente: Siemens, 2017.

### 3.1.2. SM1281

El SIPLUS SM 1281 es un módulo de extensión para la línea de PLC S7-1200 que permite la monitorización del estado de la condición [16]. Cuenta con 4 canales analógicos que permiten captar las señales de sensores, mas una entrada digital para medir velocidad de giro, útil en la monitorización de motores eléctricos.

El software incorporado permite realizar estadísticas básicas de CMS, tales como valores RMS de aceleración y velocidad, análisis espectral y almacenamiento de la señal de vibración. También puede almacenar en memoria los datos adquiridos para su posterior descarga vía servidor web o comunicación utilizando protocolo FTP. La configuración de este dispositivo es posible con un S7-1200. Por ello, su programación es realizable utilizando el software STEP 7 presente en TIA Portal,

siendo además necesario plasmar la configuración específica dentro del programa en el PLC que esté conectado a este módulo.



Figura 3.4: Aspecto real SM1281. Fuente: Siemens, 2017.

El SM1281 dispone de conectividad TCP/IP, lo cual permite obtener los datos del sensor u observar las estadísticas calculadas de forma remota. La conexión TCP/IP también permite acceder a su página web integrada, la cual se aprecia en la figura 3.5.

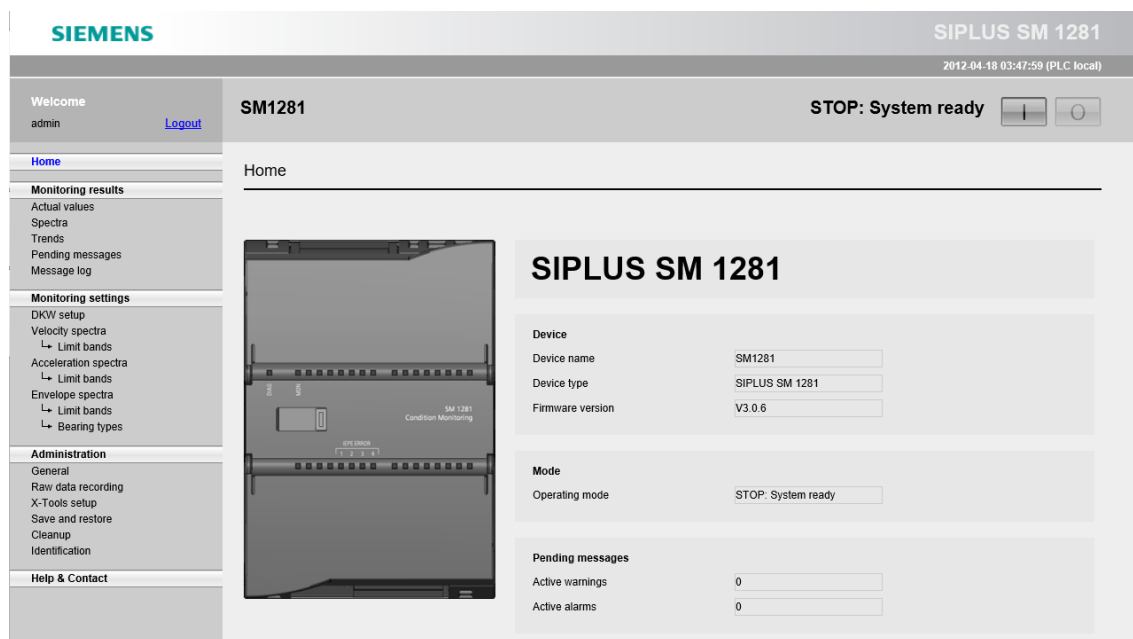


Figura 3.5: Aspecto página web del SM1281. Fuente: Siemens, 2017.

La figura anterior muestra un menú a la izquierda que permite configurar los distintos parámetros:  $a_{RMS}$ ,  $v_{RMS}$ , DKW. La configuración de los distintos sensores y bandas admisibles para el cálculo de indicadores se puede realizar a través de esta página, siendo posible incorporar las características de los rodamientos que se requiere analizar, tal como aparece en la figura 3.6.

Bearing type: pequeño

Reload New... Open... Save Save as... Delete...

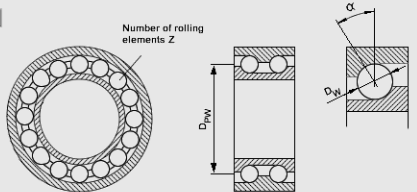
Define bearing type via

Contact angle  $\alpha$   °

Pitch circle diameter  $D_{pw}$   mm

Ball diameter  $D_w$   mm

Number of rolling elements Z



Calculate fault frequencies for this bearing type

Speed for fault frequencies  rpm

Ball passing frequency outer race  Hz

Ball passing frequency inner race  Hz

Ball spin frequency  Hz

Fundamental train frequency  Hz

Figura 3.6: Aspecto página web del SM1281 para la configuración de rodamientos a monitorizar. Fuente: Siemens, 2017.

### 3.1.3. *HMI TP700 COMFORT PANELS*

El HMI TP700 Comfort Panel es un dispositivo con panel táctil de 7 pulgadas, diseñada para su uso en ambiente industrial [36]. Este dispositivo cuenta con una construcción robusta que cumple con los estándares IP65, conectividad PROFINET, PROFIBUS, USB 2.0 tipo A y Mini-B, mas una ranura para SIMATIC HMI Memory Cards. Se alimenta directamente de 24V<sub>dc</sub>. Las interfaces son programadas con el software HMI WinCC, el cual permite realizar las distintas configuraciones y menús personalizados. En su interior dispone de una versión personalizada por Siemens de Windows CE 6.0 R3 [37]. Dentro del mismo sistema operativo, contiene aplicaciones que permite abrir archivos de tipo PDF, xls/xlsx, doc e incluso reproducción de archivos multimedia (archivos avi no comprimido y mp3). Dependiendo de su configuración, incluso permite navegar por internet gracias al navegador web integrado. Siemens integra a este dispositivo la tecnología Sm@rtServer, que permite el acceso a este dispositivo con las mismas funcionalidades que una conexión a escritorio remoto. También permite la creación de servidores OPC UA [38] para la interacción transparente con autómatas de Siemens y distintas marcas. Finalmente, permite servicios web vía SOAP.



Figura 3.7: Aspecto real HMI TP700 Comfort Panels. Fuente: Siemens, 2017.

#### 3.1.4. SENSOR S01

Este sensor cuenta con un piezoeléctrico de cuarzo con electrónica de procesamiento digital integrada. Tiene una sensibilidad de 100mV/g. Su tensión BIAS [39] se encuentra entre los 10 hasta los 14 VDC, siendo alimentado bajo el mismo estándar IEPE, con un consumo máximo de 10mA. Gracias a su grado de protección IP65 y su cuerpo de acero inoxidable, es posible someterlo a ambientes industriales exigentes. Dependiendo del tipo de anclaje que se realice a la máquina a medir, el sensor puede variar su rango sensible de medición [16]. Para este trabajo, se realiza la unión con cera química similar a la cera de abeja, permitiendo un rango de sensibilidad hasta los 18kHz.



Figura 3.8: Sensor S01. Fuente: Siemens, 2017.

#### 3.1.5. MOTOR ELÉCTRICO

El motor eléctrico utilizado se encuentra montado dentro de una maqueta a escala de un ascensor tipo M34 de suspensión 1:1, con tecnología sin cuarto de máquinas (MRL por sus siglas en inglés). Este motor es del tipo Unimotor fm de la marca Control Techniques [40], teniendo acoplada una reductora 1:16. En la figura 3.9 se aprecia el conjunto motor.



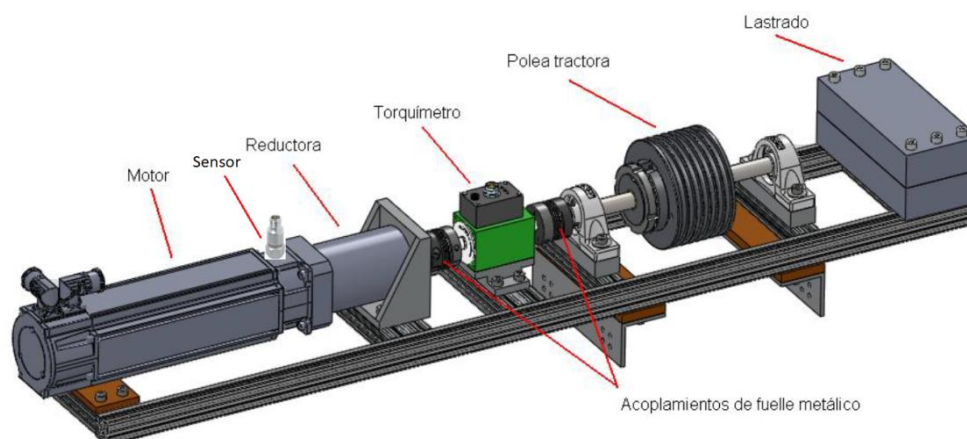


Figura 3.9: Esquema conjunto motor en maqueta ascensor. Fuente: “Construcción y puesta a punto de un banco de ensayos de ascensor a escala”, Jon Galdos, Escuela Politécnica Superior de Mondragon Unibersitatea, 2014.

## 3.2. *Elementos de Software*

### 3.2.1. *TOTALLY INTEGRATED AUTOMATION PORTAL (TIA PORTAL)*

La programación de los distintos elementos se encuentra centralizado bajo el software Totally Integrated Automation Portal, o simplemente TIA Portal [41]. Este software permite uniformar la programación de los autómatas y periféricos de Siemens.

Una versión de este software propietario viene incluida al adquirir los autómatas de Siemens. La versión de software es función del año de compra del elemento a programar. La compatibilidad con proyectos programados en versiones antiguas de este software se mantiene hasta la versión 13 [42]. Dado los distintos esquemas de diseño dentro de la plataforma, desde la versión 14 deja de existir compatibilidad, adjuntando la versión 13 en paralelo a la versión 14 para importar los proyectos a la nueva versión.

Cada versión de TIA Portal tiene paquetes de actualización (llamados *Updates*) y paquetes de servicio [43] (llamados *Service Packs*). Dependiendo de las versiones de *Update* y *Service pack* instalado, será posible la compatibilidad con versiones de firmware de distintos dispositivos. Para este TFM se utiliza la versión TIA Portal versión 14 SP1 upd 5. En la figura 3.10 se muestra la pantalla principal en el momento de ejecutar la aplicación. Esta pantalla o vista se suele denominar vista de “Portal”.

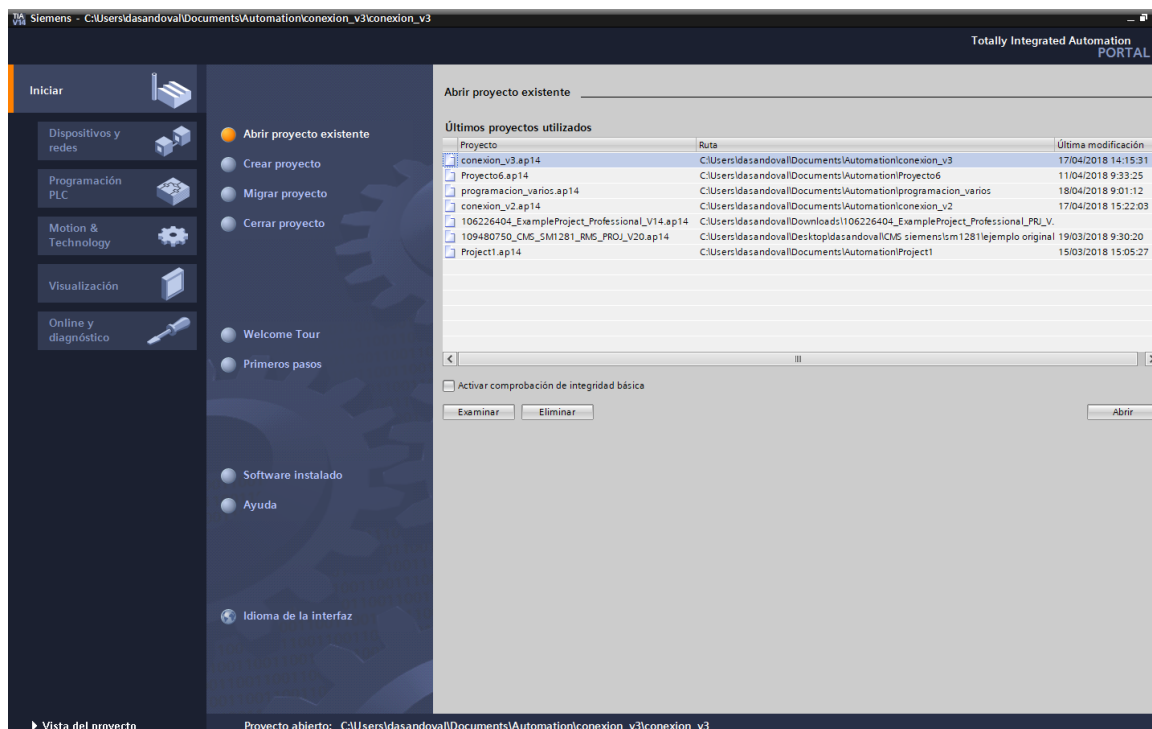


Figura 3.10: Pantalla principal de TIA Portal. Fuente: TIA Portal, Siemens, 2017.

La idea de crear TIA Portal es programar centralizadamente la red de dispositivos Siemens interconectados en el sistema de automatización. Esto permite facilitar el trabajo de programación y disminuir las horas hombre en el desarrollo del software del sistema. Para agregar dispositivos al proyecto se utiliza su “catálogo de hardware”. Este catálogo viene con un número de dispositivos configurados por defecto, permitiendo agregar nuevos dispositivos a los ya existentes. Para ello se utilizan los paquetes llamados HSP [44] (Hardware Support Packages), descargables en la página oficial de soporte de Siemens. Como los paquetes HSP permiten solo agregar los dispositivos al proyecto, es necesario instalar los programas que permitan compatibilizar la programación del PLC junto al módulo a utilizar. Para ello, se disponen de librerías con configuraciones y bloques programados para la integración al código creado en el PLC, llamadas librerías [45]. Estas librerías dependen directamente de la versión de firmware que tenga cada dispositivo dentro del, por lo que resulta obligado corroborar la compatibilidad entre la versión de TIA Portal con los paquetes HSP y las librerías. La interfaz principal donde se realiza las configuraciones y programación de los equipos aparece en la figura 3.11.

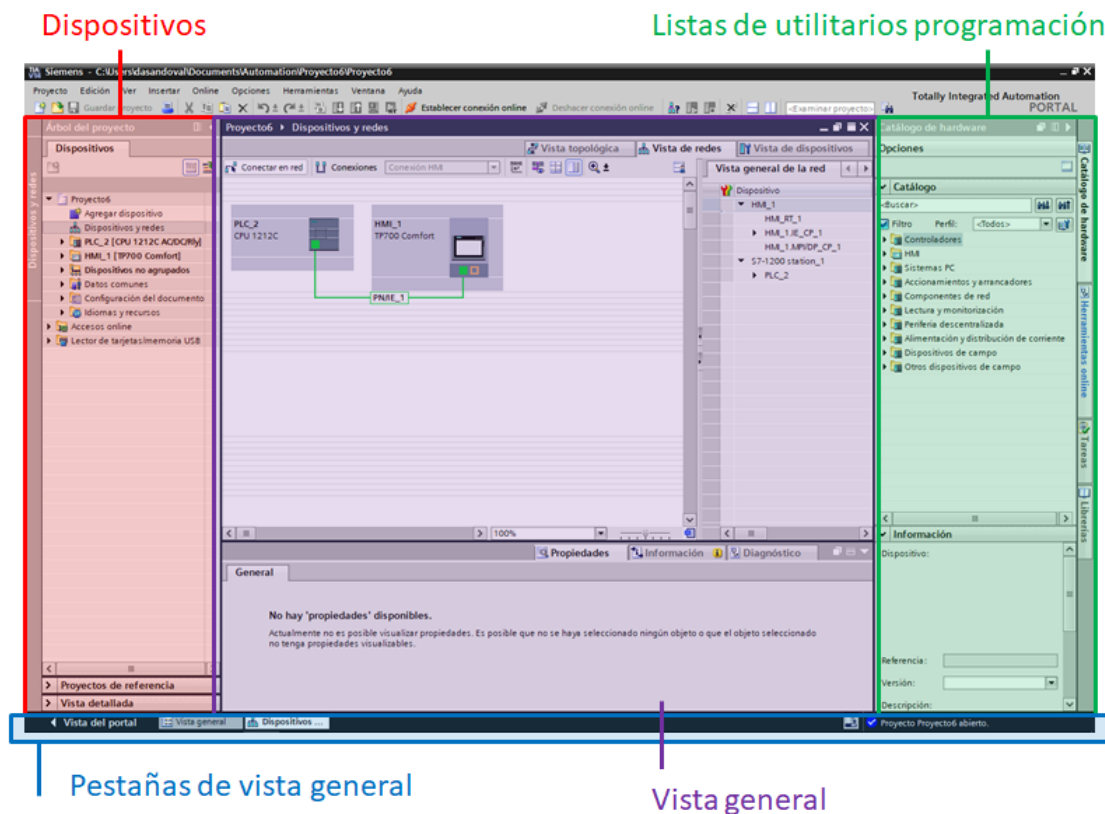


Figura 3.11: Distribución configuraciones en interfaz de TIA Portal. Fuente: TIA Portal, Siemens, 2017.

En la figura anterior, el área destacada en rojo comprende el árbol del proyecto. En él, se muestran los elementos presentes dentro del proyecto; autómatas, HMI, configuración de documentos, idioma, etc. Es posible modificar el árbol de proyecto mediante el asistente de configuración al ejecutar el programa TIA Portal, en el tipo de vista portal.

El área destacada en púrpura muestra el área de trabajo principal de la interfaz. En esta se realiza:

- La configuración de los dispositivos dentro del mismo proyecto creado en TIA Portal.
- La configuración general del proyecto.
- La programación de las líneas de código necesarias dentro de los autómatas y módulos anexos.
- La creación de las interfaces gráficas para los HMI.

En la figura 3.11, se observa la vista general dividida en área de configuración de conexión (arriba) y un área con las propiedades, información y diagnóstico de sistema (abajo).

El área destacada en verde presenta las herramientas contextuales de trabajo disponibles al momento de utilizar la vista general. Las distintas opciones presentes en esta área dependen directamente de la vista en la cual se encuentra trabajando, cambiando cuando se encuentra escribiendo un programa, configurando la conexión

de distintos dispositivos, realizando una interfaz gráfica para un HMI o la monitorización de los dispositivos conectados. Al momento de programar, se muestran las funciones disponibles. Si el PLC se encuentra conectado al programa, las herramientas de inicio, parada y reinicialización.

Finalmente, en el área destacada en azul se ordenan las distintas vistas de configuración y programación. Es así como se posibilita la configuración en paralelo tanto de cada dispositivo dentro del proyecto, como los diversos programas, funciones, base de datos, interfaces gráficas, comunicaciones entre dispositivos, entre otros.

TIA Portal permite personalizar las dimensiones de cada ventana integrada a la interfaz y las aéreas que se pueden ver al mismo tiempo, facilitando la programación.

### 3.2.2. *STEP7*

La configuración y programación de los autómatas y los módulos de expansión, se realiza utilizando el software STEP7 [46] dentro de TIA Portal. Anterior a TIA Portal, STEP7 era un software independiente, el cual fue modificado para ser integrado dentro de TIA Portal. La figura 3.12 muestra el área donde aparecen las opciones para la programación del PLC.

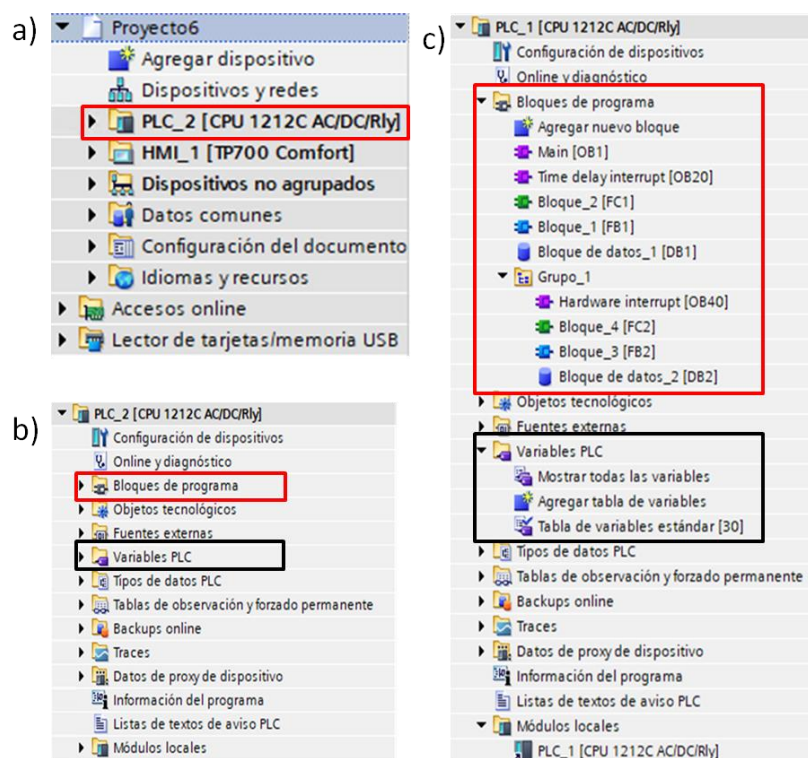


Figura 3.12: Árbol de proyecto y de opciones dentro del PLC en TIA Portal. a) destacado el PLC dentro del proyecto creado. b) destacado las secciones de programación en rojo y variables a utilizar en negro. c) detalle de estructura de programación y variables de PLC. Fuente: TIA Portal, Siemens, 2017.

En la figura 3.12.a se destaca la sección donde se encuentra el PLC dentro del árbol de proyecto. La figura 4.13.b tiene dos secciones destacadas en rojo y negro. En rojo, la escritura de código dentro del autómatas. En negro, las tablas de variables del PLC, donde se declaran los nombres de las entradas y salidas físicas del dispositivo, que puede verse en más detalle en la figura 4.13.c. La filosofía de programación de los distintos dispositivos cumple con la norma IEC 61131-3 [47]. Esta norma fue el primer paso en la estandarización de los autómatas programables y sus periféricos, incluyendo los lenguajes de programación a utilizar. Con IEC 61131-3, se establecen las especificaciones de la sintaxis y semántica de un lenguaje de programación, incluyendo el modelo de software y la estructura del lenguaje.

### 3.2.2.1. LENGUAJE SCL

En la programación del S7-1200 se utiliza principalmente los lenguajes de escalera (LADDER) [48] y de control estructurado (SCL por sus siglas en inglés) [49]. Gran parte de la programación del PLC en el sistema se realiza en SCL por ser un lenguaje de alto nivel, facilitando el desarrollo de programas numéricos complejos. En la figura 3.13, el aspecto visual de ambos lenguajes dentro de TIA Portal.

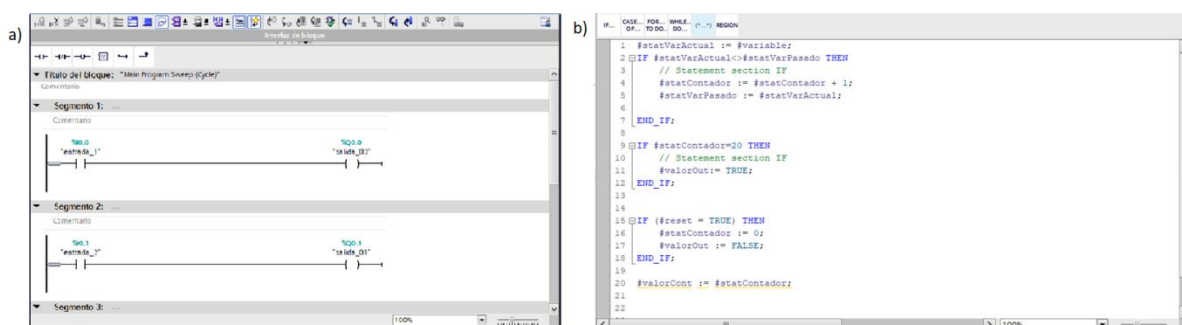


Figura 3.13: Aspecto visual TIA Portal al programar en lenguaje LADDER y SCL. a) detalle de programación en LADDER. b) detalle interfaz programación en SCL. Fuente: TIA Portal, Siemens, 2017.

Al energizar el PLC, éste se inicializa verificando las distintas partes que lo componen dentro de su sistema operativo. Posteriormente, inicializa el programa de usuario. El programa de usuario es el código escrito que será ejecutado en el PLC. En la figura 3.14 se aprecia gráficamente lo expuesto.

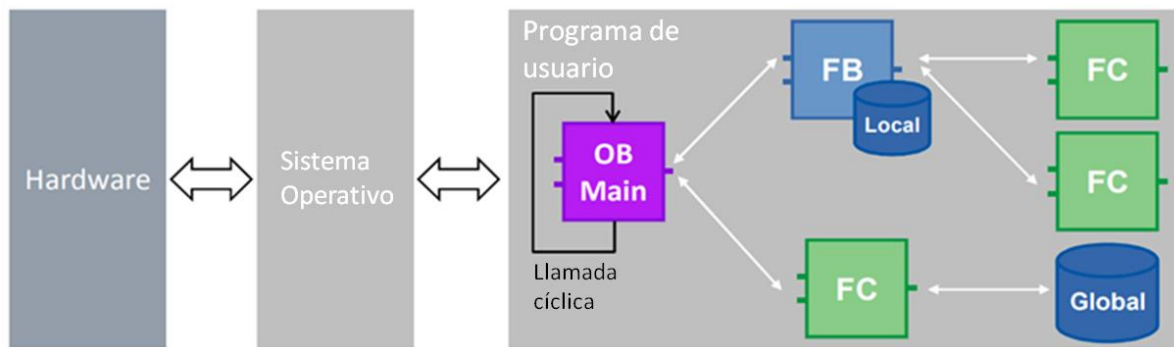


Figura 3.14: Estructura de funcionamiento del software del dispositivo PLC al momento de inicializar. Fuente: TIA Portal, Siemens, 2017.

El entorno de programación entregado por Siemens para sus autómatas responde al estándar IEC 61131-3, en el cual podemos encontrar figuras e instancias que representan porciones de código. Este enfoque permite manejar la programación con una marcada habilitación en la reutilización de código. Por ello, al crear el programa de usuario para las tareas de automatización, las instrucciones del programa se insertan en bloques lógicos. La filosofía de programación dentro del STEP7 incluye la diferenciación de distintos bloques de programa como “procesos”; bloques de organización (OB), bloques de función (FB), funciones (FC) y bloque de datos (DB) [50].

### 3.2.2.1.1. BLOQUE DE ORGANIZACIÓN

Un bloque de organización (OB) es una porción de código ejecutado al momento de un evento específico en la CPU, realizando la interrupción de la ejecución del programa de usuario. Estos OB permiten estructurar el programa, sirviendo de interfaz entre el sistema operativo y el programa de usuario. Dentro de los distintos OB existentes, hay OB que tienen eventos de arranque y comportamiento de arranque predefinidos. El OB de ciclo contiene el programa principal y existen también eventos al obtener distintos tipos de error. Al momento de realizar la programación de un PLC en TIA Portal, es posible incluir más de un OB de ciclo en el programa de usuario. Para ello, los OB de ciclo se ejecutan en el nivel de prioridad más bajo, pudiendo ser interrumpidos por todos los demás tipos de eventos. El OB de arranque no interrumpe el OB de ciclo, ya que la CPU ejecuta el OB de arranque antes de pasar al estado operativo RUN. Tras finalizar el procesamiento de los OB de ciclo, la CPU vuelve a ejecutarlos inmediatamente. Esta ejecución cíclica es el tipo de procesamiento "normal" que se utiliza para los controladores lógicos programables. Es habitual encontrarse con programas en los cuales el programa de usuario está contenido íntegramente en un solo OB de ciclo. La creación de otros tipos de OB para ejecutar funciones específicas es posible, tales como procesamiento de alarmas y tratamiento de errores. Para realizar su ejecución, estos OB interrumpen el funcionamiento de la ejecución de los OB de ciclo. Cuando ocurre un evento, la CPU interrumpe la ejecución del programa de

usuario y llama el OB configurado para procesar el evento que sucede en el momento. Una vez finalizada la ejecución del OB para una alarma especificada, la CPU reanuda la ejecución del programa de usuario en el punto de interrupción. La figura 3.15 muestra los íconos utilizados por TIA Portal para diferenciar los bloques comentados.

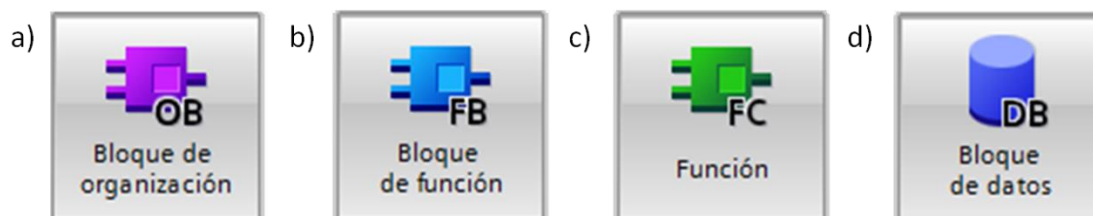


Figura 3.15: Tipos de bloques presentes para la programación en SCL. a) representación para los bloques de organización. b) representación de los bloques de función. c) representación para los bloques función. d) representación de los bloques de datos. Fuente: TIA Portal, Siemens, 2017.

#### ***3.2.2.1.1.1. FUNCIONES***

Junto a las OB, existen las funciones (FC) como bloque de código que realiza una operación específica en un conjunto de valores de entrada. Las FC almacenan los resultados de esta operación en posiciones de memoria. Por ello, las FC son utilizadas para ejecutar operaciones estándar y reutilizables (como cálculos matemáticos) o funciones tecnológicas (como el control individual que se apoyan de lógica de bits). Al ser un bloque aislado, permite su reutilización numerosas veces dentro del programa de usuario. Esto facilita la programación de tareas que se repiten con frecuencia sin tener la necesidad de rescribir el código múltiples veces. Una FC no tiene asignado o asociado algún bloque de datos asociado (DB). Las FC usan la memoria volátil del PLC para alojar los datos temporales utilizados, por lo que no se almacenan para su posterior uso. Si se requiere esta necesidad, es preciso asignar el valor de salida a una posición de memoria global, por ejemplo el área de marcas o un DB global. Es posible realizar con mayor flexibilidad un programa modular al llamar bloques de función específicos que ejecutan tareas específicas. Como se realiza la subdivisión del programa principal en varios módulos, la tarea de automatización compleja se divide en tareas más pequeñas.

#### ***3.2.2.1.1.2. BLOQUE DE FUNCIÓN***

Similar a los FC son los bloques de función (FB), los cuales se diferencian principalmente en el uso de un bloque de datos (DB) llamado de “instancia” para sus parámetros y datos estáticos. El DB instancia ofrece un bloque de memoria asociado al FB, almacenando datos una vez que haya finalizado el FB. Es posible asociar distintos DB de instancia a diferentes llamadas del FB. Los DB de instancia permiten utilizar un FB genérico para controlar varios dispositivos. El programa se

estructura de manera que un bloque lógico llame un FB y un DB instancia. La CPU ejecuta luego el código del FB y almacena los parámetros del bloque y los datos locales estáticos en el DB instancia. Cuando finaliza la ejecución del FB, la CPU regresa al bloque lógico que ha llamado el FB. El DB instancia conserva los valores de esa instancia del FB. Estos valores están disponibles para las llamadas posteriores al bloque de función, bien sea en el mismo u otros ciclos.

### ***3.2.2.1.1.3. BLOQUES DE DATOS***

Los bloques de datos (DB) son instancias que se crean en el programa de usuario. Su única función es la de almacenar datos. Existen principalmente 2 tipos de DB; global y de instancia. Todos los bloques del programa de usuario pueden acceder a los datos en un DB global. En cambio, un DB instancia almacena los datos de un FB específico. Los datos almacenados en un DB no se borran cuando finaliza la ejecución del bloque lógico asociado. La estructura de los datos en un DB instancia refleja los parámetros (Input, Output e InOut) y los datos estáticos del FB (La memoria temporal del FB no se almacena en el DB instancia).

### ***3.2.2.1.2. DISEÑO DE PROGRAMA Y MODULACIÓN***

Diseñando FB y FC que ejecuten tareas genéricas, se crean bloques lógicos modulares. El programa se estructura de manera que otros bloques lógicos llamen estos bloques modulares. El bloque que efectúa la llamada transfiere los parámetros específicos del dispositivo al bloque llamado. Cuando un bloque lógico llama otro bloque lógico, la CPU ejecuta el código del programa en el bloque llamado. Una vez finalizada la ejecución del bloque llamado, la CPU reanuda la ejecución del bloque que ha efectuado la llamada. El procesamiento continúa con la ejecución de la instrucción siguiente a la llamada de bloque.

Las llamadas de bloque pueden anidarse para crear una estructura modular. En la figura 3.16, la profundidad de anidamiento es de orden 3: El OB de ciclo de programa más 3 niveles de llamadas de bloques lógicos.



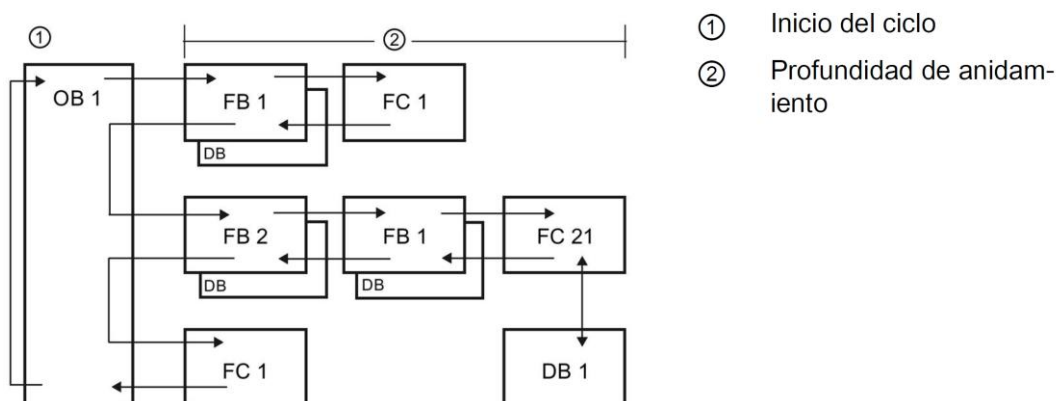


Figura 3.16: Estructura de anidamiento de los distintos bloques funcionales en un programa SCL. Fuente: TIA Portal, Siemens, 2017.

STEP 7 cuenta con un editor de programas en lenguaje SCL, que incluye:

- Sección de variables de programa para definir los parámetros del bloque lógico.
- Área de código para la codificación del programa.
- Apartado de instrucciones, tareas y librerías permisibles de utilizar en SCL soportadas por la CPU.

Estas secciones están en la figura 3.17 marcadas en rojo, púrpura y verde respectivamente.

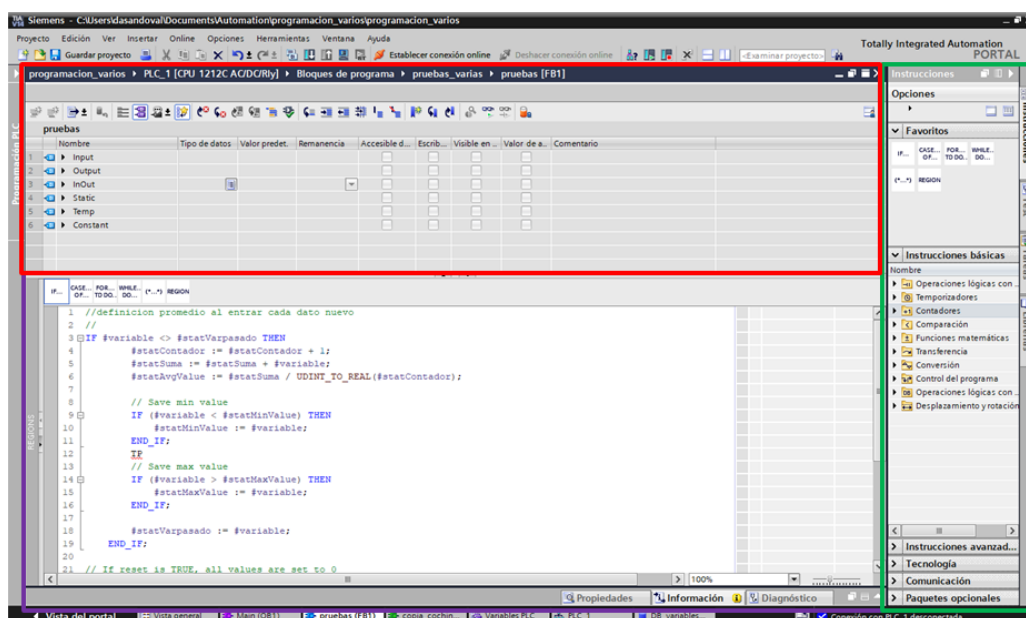


Figura 3.17: Interfaz gráfica para programar en lenguaje SCL dentro de TIA Portal. Fuente: TIA Portal, Siemens, 2017.

El código SCL se ingresa directamente en la sección de código. El editor incluye botones para las construcciones y comentarios de código habituales. Para obtener instrucciones más complejas, basta con arrastrar las instrucciones SCL del árbol de instrucciones y soltarlas en el programa.

Es posible emplear cualquier software editor de texto para crear un programa SCL, importando luego el archivo creado con el código generado a STEP 7. En la sección de interfaz del bloque lógico SCL se pueden declarar:

- Input, Output, InOut y Ret\_Val: estos parámetros definen las variables de entrada, las variables de salida y el valor de retorno del bloque lógico. El nombre de la variable introducida en este punto se emplea de forma local durante la ejecución del bloque lógico. Normalmente, no se emplea el nombre de variables globales en la tabla de variables.
- Static: el bloque lógico utiliza variables estáticas para almacenar resultados intermedios estáticos en el bloque de datos de instancia. El bloque retiene datos estáticos hasta que se sobrescriben, lo cual puede ocurrir después de varios ciclos. Los nombres de los bloques, también se almacenan en los datos locales estáticos. Solo disponible para FB.
- Temp: estos parámetros son variables temporales que se emplean durante la ejecución del bloque lógico.
- Constant: son valores constantes con nombre para el bloque lógico.

Si se llama el bloque lógico SCL desde otro bloque, los parámetros del bloque lógico SCL aparecen como entradas o salidas.

La programación en SCL permite utilizar tanto variables declaradas en el mismo bloque de función, como variables almacenadas en la tabla de variables de PLC e incluso variables que se encuentren en DB dentro del PLC. Al momento de escribir el programa, se despliega un menú contextual con sugerencias sobre las instrucciones de los bucles de programa. Incluso es posible arrastrar el cursor del ratón sobre una variable utilizada en el programa y el editor muestra un mensaje contextual el tipo de la variable apuntada. La figura 3.18 resume lo descrito.

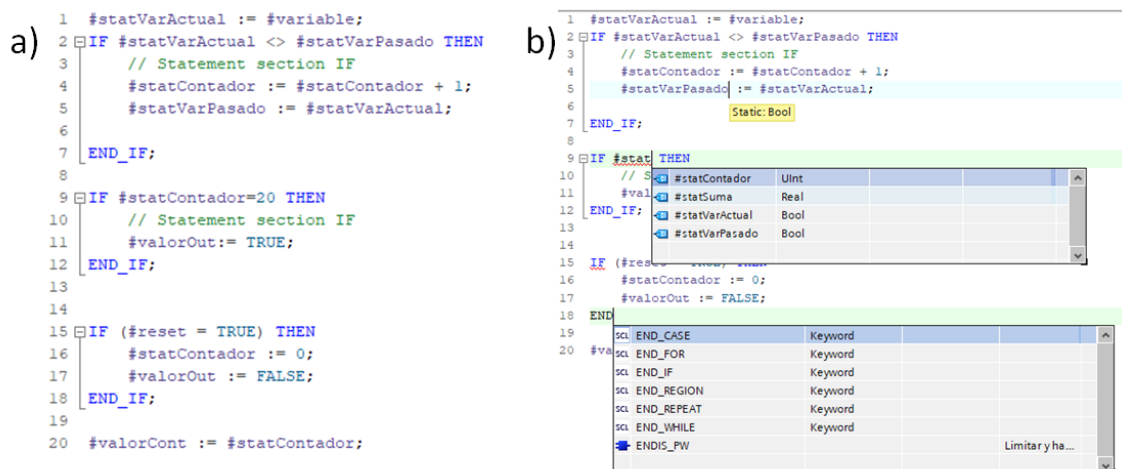


Figura 3.18: Interfaz de código escrito en SCL bajo el entorno de programación de TIA Portal. a) espacio de programación y sus diferentes partes. b) mensajes contextuales entregados por TIA Portal Fuente: TIA Portal, Siemens, 2017.

El código expuesto en la figura 3.18 muestra una rutina que cuenta las veces que cambia de estado una entrada lógica del PLC (designada como “variable”). Al momento de realizarse veinte cambios de estado, lleva a 1 el estado de una salida lógica del PLC (nombrada como “valorOut”). Es posible llevar a cero el contador de permutación de la variable de entrada más el valor de la salida lógica con una segunda entrada (denominada “reset”). Esto demuestra la posibilidad de controlar entradas y salidas del PLC con SCL.

Además de realizar la programación directamente desde el editor de lenguaje SCL de TIA Portal, es posible importar código generado desde otros programas para ser utilizados en el proyecto. Para ello, es necesario acceder a la opción “Fuentes externas” dentro de las opciones disponibles del autómatas dentro del árbol de proyecto. La figura 3.19 muestra esta opción.

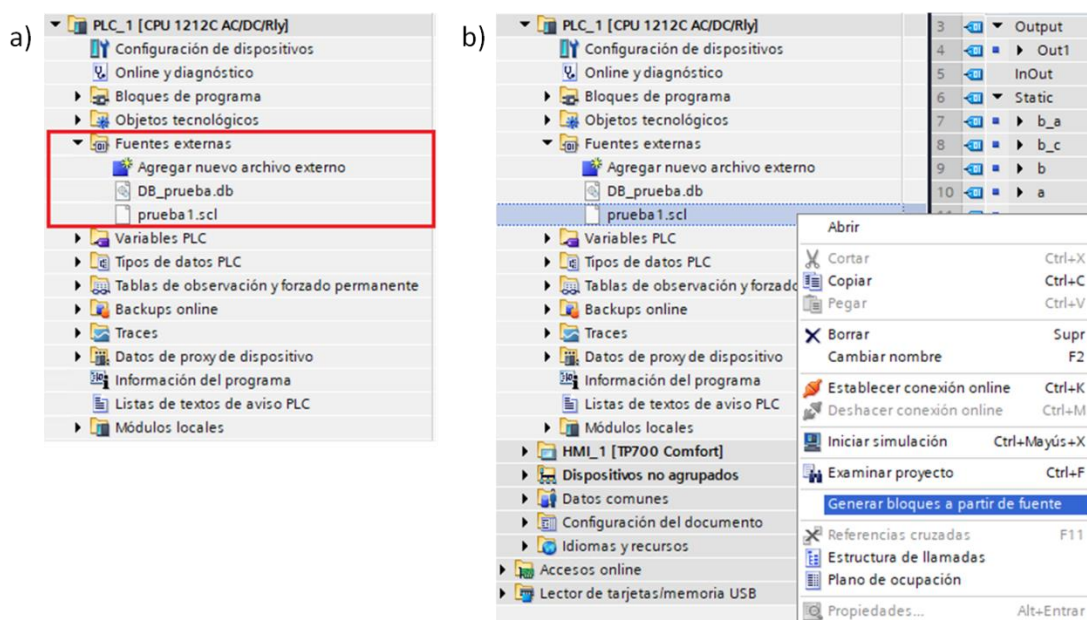


Figura 3.19: Árbol de proyecto y detalle de importación de código desde fuentes externas. a) opción de importación dentro del elemento PLC en el árbol de proyecto. b) pasos para generar código desde archivo externo en TIA Portal. Fuente: TIA Portal, Siemens, 2017.

De la figura 3.19.a, se observa que existen 2 archivos importados con distinto formato. El archivo prueba1.scl contiene código generado para crear un FB. El archivo DB\_prueba.db contiene código para crear un DB con valores ya establecidos. Es posible realizar la exportación a código compatible en distintas plataformas que respondan al estándar IEC 61131-3 es posible desde TIA Portal, como es posible apreciar en la figura 3.19.b. Incluso se permite leer estos archivos con editores de texto simple.

Siemens recomienda formas de programación que permiten estructurar correctamente el código y estandarizar la forma en que son creados los bloques, permitiendo que sean compartidos luego estos bloques dentro de la comunidad que

trabaja con dispositivos Siemens. Dentro de estas recomendaciones, se pueden mencionar:

- Uso de DB optimizados frente al uso de bloques DB no optimizados. Esto permite no solo un ahorro en espacio en memoria, sino que también aumenta la velocidad de cómputo del PLC.
- Uso de variables del tipo VARIANT frente al uso del tipo ANY. Su mayor ventaja es el ahorro de espacio en memoria y agregar una prueba de integridad al tipo de dato apuntado.
- Direccionamiento simbólico frente al uso de direccionamiento absoluto. Facilita la interpretación de código y ahorrar memoria utilizada.
- Uso de librerías. Esto permite reutilizar en múltiples proyectos bloques de funciones creados en proyectos anteriores, aumentando la eficiencia en la creación de proyectos en TIA Portal.

### 3.2.3. WINCC

TIA Portal incluye el software para programar interfaces gráficas en los HMI, llamado WinCC [51]. Este software permite vincular variables internas del HMI hacia la interfaz y hacia el PLC, junto con variables desde el PLC.

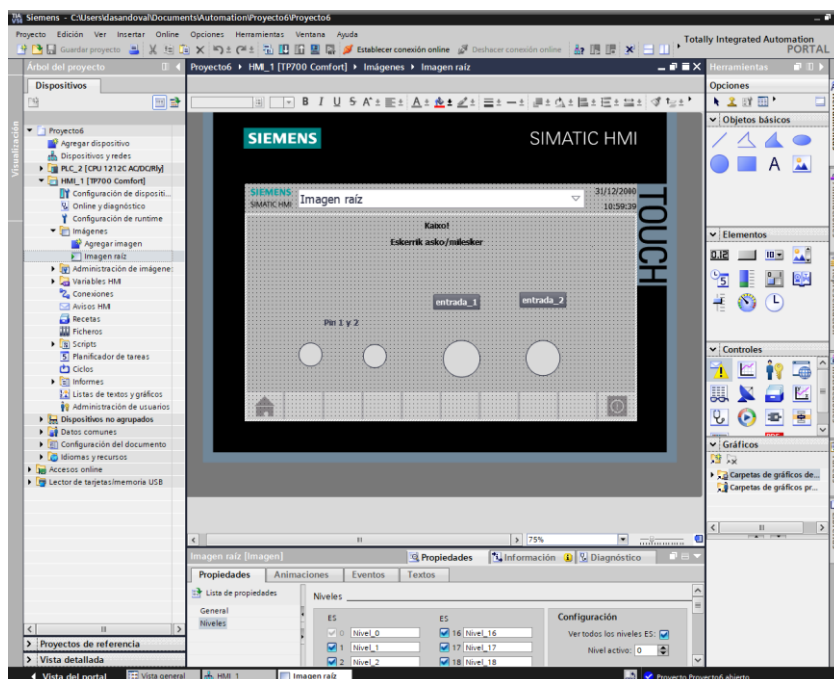


Figura 3.20: Pantalla de TIA Portal para programación de dispositivos HMI. Fuente: TIA Portal, Siemens, 2017.

Al igual que STEP7, WinCC existía como un programa independiente, el cual fue modificado en su interfaz para ser utilizado en el entorno de trabajo de TIA Portal, como se aprecia en la figura 3.20. Es así como existe en el área de trabajo (centro) la

vista de la interfaz y los distintos tipos de elementos que pueden ser agregados (derecha), como figuras geométricas, botones, indicadores de valores, controles de comunicación y media. También hay elementos que integran gráficas de datos más el árbol de proyecto donde se encuentran las distintas propiedades a configurar (izquierda). El HMI utilizado en este TFM permite realizar programas que se ejecuten en conjunto con la interfaz (scripts) creados en el lenguaje de programación Visual Basic script [52] (conocido normalmente como VBScript o VBS).

### 3.2.3.1. WINDOWS CE

Para entender mejor el funcionamiento del HMI, es necesario comprender como funciona la versión de Windows que utiliza. Esta versión se llama Windows Embedded CE o simplemente Windows CE, en su versión 6 Revisión 3[53]. Windows CE es un sistema operativo modular desarrollado por Microsoft enfocado en la optimización del uso de recursos, con el objetivo de ser ejecutado en dispositivos con recursos de hardware limitados. Para ello, la versión del *kernel* de Windows CE es modificado para tener un sistema operativo “cerrado”, lo que se traduce que solo puede ser modificado antes de ser compilado y grabado en la memoria del dispositivo donde será ejecutado. Windows CE cumple con la denominación de sistema operativo de ejecución en tiempo real, presentando latencia de interrupciones determinística. Utilizando el sistema de prioridades, presenta 256 niveles, siendo fundamental el uso de *thread* o “hilos” para la ejecución de los programas. Para su distribución, Microsoft ofrece porciones de código con variadas funcionalidades, las cuales van siendo agregadas por los desarrolladores dependiendo del hardware a realizar. Esto permite tener un sistema operativo creado específicamente para el hardware a utilizar (de ahí su nombre de sistema operativo modular). Sin embargo, un número considerado de componentes fundamentales de Windows CE solo son distribuidos bajo su formato binario, siendo por ende imposibles de modificar. Estos componentes no necesitan adaptación para el ambiente de hardware a utilizar, siendo independiente del hardware.



Figura 3.21: Fotografía de uno de los primeros pocket PC del mercado con Windows CE, Casio Cassiopeia A-11. Fuente: <https://www.flickr.com/photos/johndecember/2264105808>, 1997.

### 3.2.3.2. VISUAL BASIC SCRIPT

VBS es un lenguaje de programación interpretado instaurado por Microsoft el año 1996, que fue recibido con gran respaldo por parte de los administradores de sistemas Windows como herramienta de automatización, porque entrega mayor flexibilidad y actuación dentro del sistema en comparación al lenguaje *batch* desarrollado a finales de la década de los 70 para MS-DOS [54]. Al ser un lenguaje interpretado, no presenta herramientas *debugger*, siendo invocado por motores de ejecución en Windows o dentro de páginas web. Su última actualización fue realizada el año 2002 dado que fue reemplazado por el lenguaje ASP.NET para el desarrollo web y su uso es compatible con versiones modernas de Windows utilizando el interprete vbscript.dll. Este lenguaje de programación permite manejar archivos, modificar, realizar *streaming* de texto, manejar archivos binarios y memoria I/O utilizando la clase "ADODB.Stream" [55]. Permite a su vez realizar la construcción de arreglos de datos. También permite programar el acceso a bases de datos y manipulación de archivos XML y navegación en internet a través de los objetos interpretes XMLHTTP y ServerXMLHTTP en su versión 6.0. Por último, es necesario mencionar su funcionalidad a través de tecnologías ActiveX. Esta funcionalidad permite precisamente el uso de las interfaces creadas en el HMI por el usuario e interactuar con los distintos dispositivos conectados a través de la subred Ethernet del proyecto.

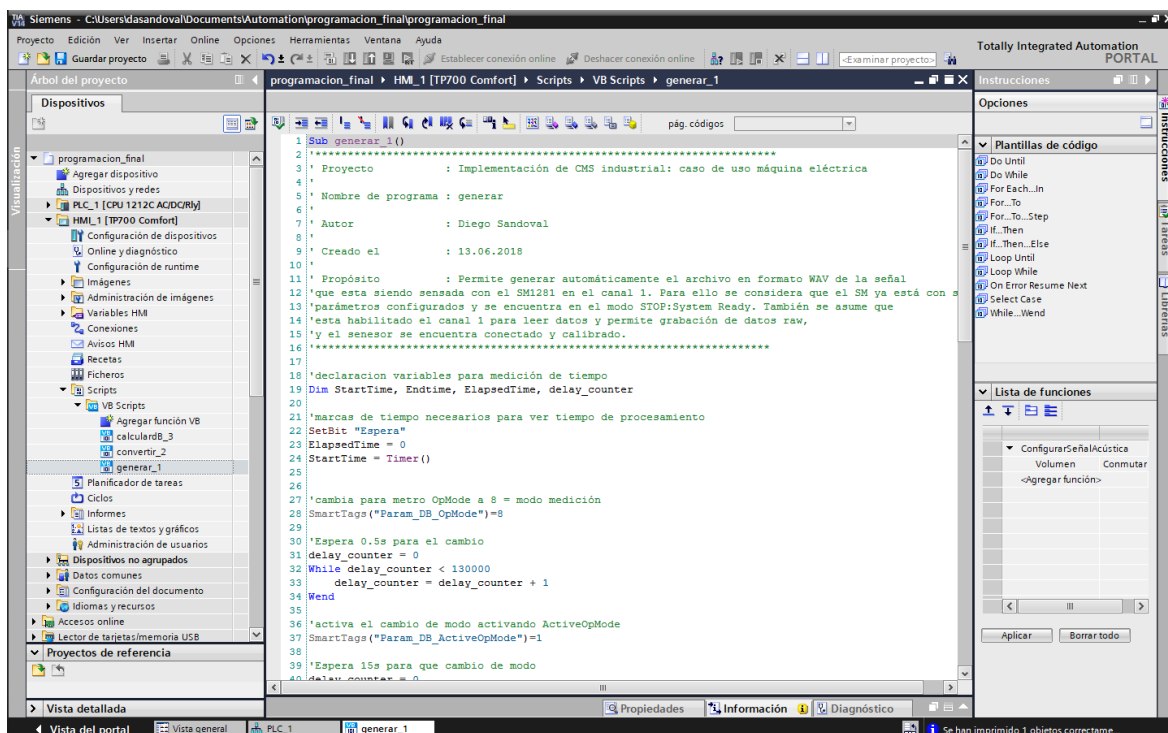


Figura 3.22: Interfaz de TIA Portal para la programación en VBS. Fuente: TIA Portal, Siemens 2018.

Dentro del entorno de programación permitido por Siemens en su HMI, este lenguaje de programación presenta diferencias considerables a sus otras versiones, como lo son Visual Basic y Visual Basic para aplicaciones. Como fue comentado, el OS utilizado no presenta la totalidad de herramientas estándar de un sistema operativo de escritorio. Esto incide en las librerías o “DLL” disponibles para la programación. Además de las limitaciones del OS, existen limitaciones impuestas por Siemens. Ejemplo de ello es la imposibilidad de usar la librería “ADODB.Stream” dentro de la programación en VBS [56]. Tampoco es posible utilizar algunas propiedades de ActiveX, lo cual no se encuentra documentado por parte de Siemens. El soporte de Siemens para VBS solo es posible en su versión de pago. Aún en este escenario, la programación bajo este lenguaje permite realizar diferentes tareas como lectura/escritura de archivos a nivel de byte, manipular variables que se encuentran dentro de un DB alojado en el PLC, funciones matemáticas entre otros.

### 3.2.3.3. DISEÑO DE INTERFAZ

Para crear una interfaz en los HMI de Siemens, el programa WinCC presenta el sistema de plantillas [57], donde cada pantalla visible se denomina Imagen. Las plantillas entregan por defecto un diseño de pantalla al cual puede ir agregándose distintos elementos. Estos elementos se encuentran por defecto al momento de agregar una nueva Imagen. Los elementos activos de la Interfaz pueden ser cuadro de valores, botones, imágenes, gráficos, dibujos geométricos, etc. A medida que se agregan imágenes al proyecto, estos deben respetar una jerarquía definida por el usuario. Junto con las Imágenes creadas por plantilla, es posible de realizar la programación de imágenes emergentes (conocidos también como *pop ups*). La figura 3.23 muestra marcado en rojo y negro donde se encuentran dentro del árbol de proyecto de la interfaz de TIA Portal.

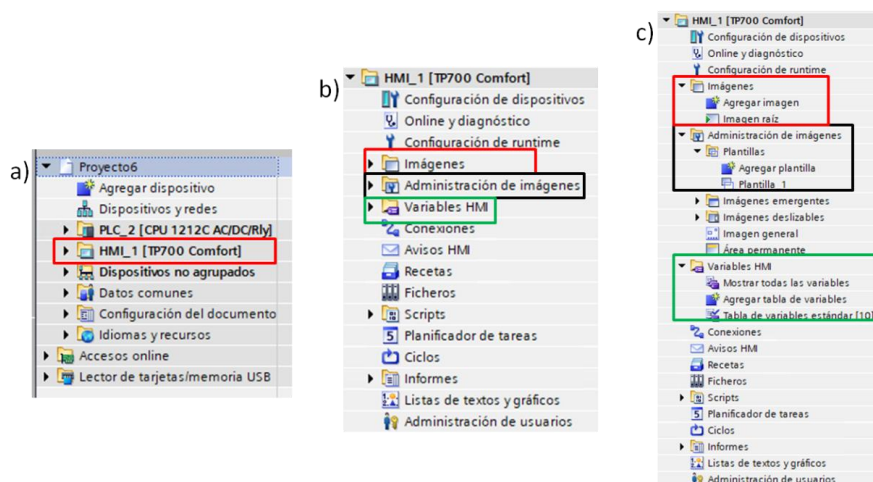


Figura 3.23: Árbol de proyecto y de opciones dentro del HMI en TIA Portal. a) elemento HMI dentro del árbol de proyecto. b) detalle de opciones utilizadas mayormente para programar la interfaz en el HMI. c) detalle de opciones utilizadas para programar interfaz en HMI. Fuente: TIA Portal, Siemens, 2017.

De la figura 3.23 se puede apreciar destacado en verde la sección “Variables HMI”. En esta sección es posible agregar variables para ser utilizadas por la interfaz, las cuales pueden ser internas al funcionamiento del HMI o estar enlazadas a otras variables presentes en la programación de un PLC. Es por ello, que puede suceder ambas situaciones; trabajar con variables internas del PLC y variables internas de la interfaz.

Dentro de las herramientas ofrecidas para la creación de interfaces, existen 3 grandes grupos de opciones. Los objetos básicos son aquellos que permiten dibujar figuras geométricas, escribir texto y pegar imágenes almacenados en memoria. Los elementos permiten la interacción de la interfaz con variables alojadas internamente o externamente. Estos elementos pueden ser botones, campos que despliegan variables, listas de elementos, interruptores e indicadores de valores. Los controles permiten acciones más avanzadas a las permitidas con los elementos. Ejemplo de ello es el visor de gráficos, visor de funciones, visor de PDF y video, e inclusive navegador por una página web. En general todos los elementos, controles y objetos básicos cuentan con propiedades comunes. La figura 3.24 muestra como ejemplo la configuración de un botón.

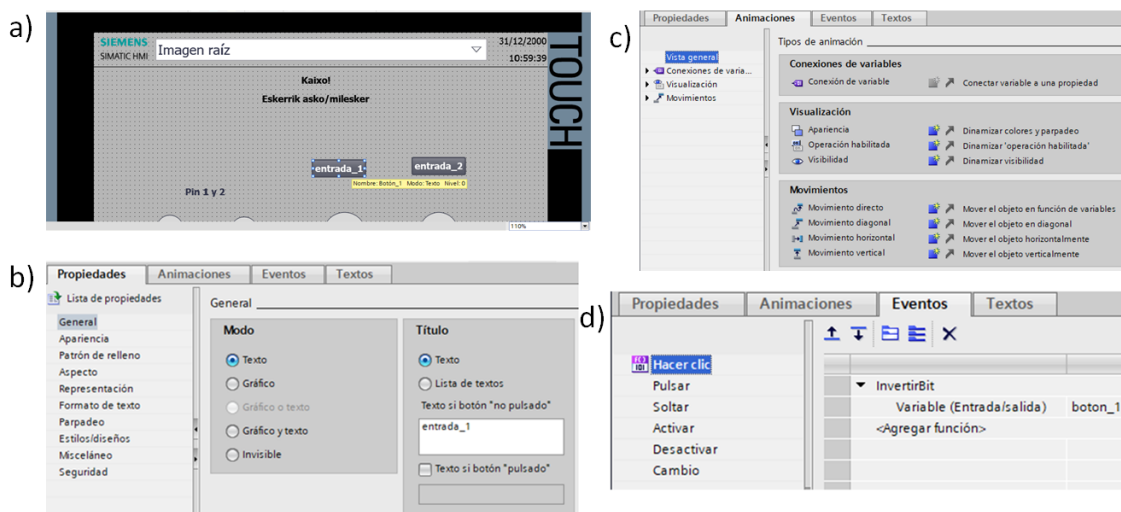


Figura 3.24: Pantalla principal de TIA Portal. Fuente: TIA Portal, Siemens, 2017.



## 4. INSTALACIÓN Y PUESTA EN MARCHA DEL SISTEMA

Los elementos descritos en el capítulo anterior responden a las exigencias de la empresa Orona sobre la marca a utilizar, siendo facilitados por Ikerlan para realizar el trabajo. En este capítulo será descrito el proceso de puesta en marcha del sistema, donde se describen las dificultades encontradas en el proceso de puesta en marcha del sistema.

Las principales fuentes de información utilizadas en un principio fueron los manuales y la ayuda encontrada en los foros sobre autómatas en general y de la página oficial de Siemens [58], [59].

### 4.1. *Elementos de hardware*

#### 4.1.1. *PLC S7/1212C*

El PLC fue el elemento que menos problemas dio al momento de utilizar el entorno de programación de Siemens. El desconocimiento del número IP configurada para entablar la comunicación con TIA Portal generó problemas en un comienzo, siendo superado tras varias pruebas. También fue necesario entender cómo funcionaba su programación, la cual se realizó mediante el software TIA Portal. Dada la complejidad del software TIA Portal, fueron necesarias un par de semanas para programar el PLC utilizando los lenguaje LADDER y SCL.

#### 4.1.2. *SM1281*

El SM1281 fue el dispositivo que mas trabajo dio para integrarlo en el sistema. Mientras llegaba el dispositivo a las instalaciones de Ikerlan, se estudió manual de usuario. Coincidiendo con su llegada a Ikerlan, se produjo el lanzamiento de una nueva versión del firmware del SM 1281, pasando de la versión 2.5 a la versión 3.0. Esto produjo una desincronización entre los manuales y ejemplos subidos a la página de soporte de Siemens, basados en la versión 2.5. Tras revisar las versiones de los paquetes de software necesarios para realizar la programación utilizando TIA Portal como aparece en la tabla 4.1, se concluyó que se carecía de la información necesaria para el correcto funcionamiento del SM, haciendo necesario apoyo del *Product Manager* de España.

Tabla 4.1: Tabla resumen versiones de software necesarios para programar un SM1281 en función de su firmware. Fuente: Siemens, 2018.

Módulo	Versión de firmware de modulo	Librería requerida en TIA Portal	Software de configuración STEP 7 (TIA Portal)
SM 1281	V1.x	LSM 1281 V1.x	a partir de V13, Update 9 y HSP 0113
SM 1282	V2.x	LSM 1281 V2.1	a partir de V14 y HSP0113
SM 1283	V3.x	LSM 1281 V3.0	V14 SP1 y HSP 0113

El apoyo desde Madrid del *Product Manager* en la programación se limitó en un inicio a verificar el correcto conexionado de los diferentes dispositivos, siguiendo con asistir la correcta actualización del firmware del SM. El SM1281 tiene 2 firmware instalados: MAP y CMS [60]. El firmware MAP puede actualizarse utilizando TIA Portal, proceso llevado a cabo con éxito.

El firmware CMS se actualiza ingresando a la página web dentro del dispositivo, siendo necesario conocer el numero IP del dispositivo. Este proceso no es posible de llevar a cabo, porque no se conoce el numero IP del dispositivo. Junto con la asistencia del la *Product Manager*, transcurrida un par de semanas no fue posible obtener el numero IP del dispositivo. Se realiza la devolución al distribuidor por efecto de garantía, llegando un nuevo dispositivo transcurrida tres semanas. A la llegada del nuevo dispositivo, se realizan las pruebas pertinentes para su funcionamiento utilizando el firmware 3.0.6, volviendo a los mismos problemas obtenidos con anterioridad.

El problema en el software generado radicaba en la omisión de un código que no aparecía referenciado en el manual. Este código apareció en el programa ejemplo enviado por el *Product Manager*.

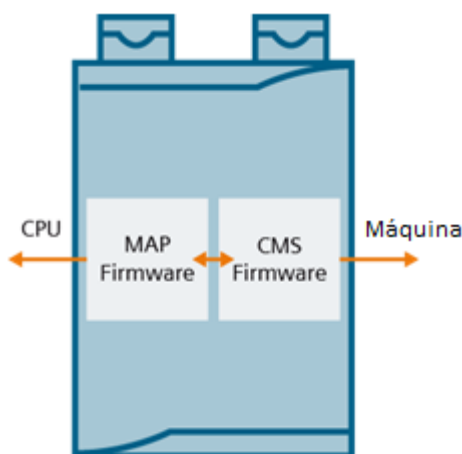


Figura 4.1: Esquema de los firmware presentes en el SM1281. Fuente: Industry Support, Siemens, 2018.

### 4.1.3. *HMI TP700 COMFORT PANELS*

Este dispositivo, al igual que el PLC, no generó problemas en su integración con el PLC utilizando TIA Portal. Sin embargo, en un momento dado dejó de funcionar totalmente. Tras ponerse en contacto con el servicio técnico se devolvió el HMI, ya que se encontraba en garantía. Un par de semanas llegó un nuevo HMI. La información entregada por el distribuidor indica que el problema era conocido, situándose el problema en un daño de un fusible interno. Esta situación coincide con la devolución del equipo SM, existiendo un tiempo en que solo estaba operativo el PLC. El HMI de reemplazo no presenta problemas posteriormente.

## 4.2. *Elementos de software*

### 4.2.1. *TIA PORTAL*

A lo largo del capítulo 3 se menciona multitud de software con sus versiones, firmware y librerías. En la tabla 4.2 se resumen las distintas versiones de hardware y firmware (cuando corresponda) de los elementos utilizados.

Tabla 4.2: Resumen versiones elementos de hardware y firmware.

Nombre	Referencia	Hardware	Firmware
S7-1200 1212C AC/DC/Rly	6ES7 212-1BE40-0XB0	v5	v4.2.1
SM1281	1P 6AT8007-1AA10-0AA0	FS04	v3
HMI TP700 Comfort Panels	6AV2 124-0GC01-0AX0	FS24	v14.0.1.0
TIA Portal- SIMATIC	STEP 7 Basic/WinCC Comfort	---	v14 SP1 Upd 5
TIA Portal-Catálogo	HSP0113	---	v3
TIA Portal-Librería Global	LSM1281	---	v3.5.5

Se desprende de la tabla anterior que TIA Portal requiere de varios paquetes adicionales para programar el SM. Al momento de instalar el software que viene en los CDs junto al PLC, solo contenía el software STEP7 Basic, necesario para programar el dispositivo. Esto no permite programar con WinCC, que debe ser instalado con los CDs que vienen junto al HMI. Se debía instalar los paquetes HSP0113 [61] y LSM1281 [62] antes de programar directamente el SM. Aunque TIA Portal unifica las interfaces y la programación en una sola pantalla, es responsabilidad del usuario la instalación de todo el software necesario para la programación.

Para utilizar el PLC, se necesita una actualización de firmware compatible con la versión de TIA Portal. Misma situación para escribir el programa que maneja el SM desde el PLC. Para el caso del HMI, se instala una versión de firmware menor a la que venía por defecto en el dispositivo por su incompatibilidad. Para tener compatibilidad con el firmware del SM, se instala el paquete HSP0113. Como requisito a esta instalación, era necesario instalar el paquete de actualización Upd 5 a la versión 14 SP1. Una vez instalada esta actualización, fue posible integrar el paquete adicional al catálogo. Posteriormente se agregó al proyecto el SM con la versión de firmware 3.0 [63]. Se instala luego un segundo paquete de software, consistente en la librería que contiene las funciones escritas en SCL dentro de la programación del PLC (según versión del firmware del SM). Una vez instalado este último paquete de software, es posible empezar a programar el PLC en conjunto con el SM 1281.

#### **4.2.2. STEP7**

La programación en STEP7 está enfocada al uso de los autómatas (PLC) y sus distintos módulos anexos. En la programación del PLC no existieron mayores problemas, a diferencia de lo sucedido con el SM.

La programación del PLC se limita a la comunicación entre PC y PLC, monitorización en línea del estado del PLC, revisión en línea de sus variables y estados en las entradas y salidas del dispositivos. Lo que toma más tiempo es aprender a utilizar las opciones disponibles para programar en lenguaje SCL.

##### **4.2.2.1. FUNCIONAMIENTO MÓDULO SM 1281**

Como se menciona en el punto 4.1.2, la configuración del SM consumió una gran cantidad de tiempo y esfuerzo, por lo que la mayor parte del desarrollo del proyecto fue realizado teóricamente, al no disponer de un SM operativo. Finalmente, cuando se dispone del SM operativo, los bloques función de la librería LSM1281 (escritas en SCL) se agregan al programa principal del PLC. El primer bloque a parametrizar se llama “SM1281\_Module”, y su configuración permite establecer las siguientes propiedades:

- Número IP.
- Velocidad de giro de la máquina a sensar (si aplica).
- Transferencia de parámetros de canal y módulo al SM1281.
- Petición de grabación de datos *raw* y estado instantáneo de indicadores.
- Ejecución de transmisión de datos *raw*.
- Restauración de valores de parámetros del módulo.
- Entrega de estado e información de errores.
- Selección de modo de operación del SM 1281.
- Cambio de prioridad de control entre PLC y pagina web de SM1281.

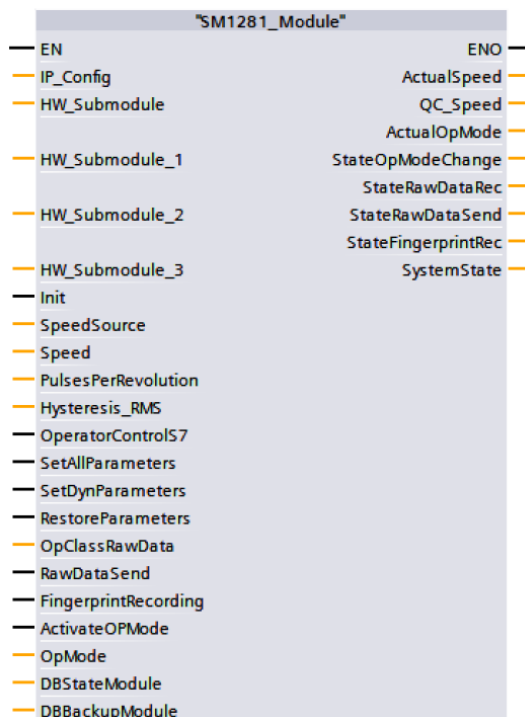


Figura 4.2: Representación bloque "SM1281\_Module" dentro de programación LADDER de PLC. Fuente: SIPLUS CMS1200 Operating instructions, Siemens 2017.

El segundo bloque de configuración permite parametrizar los canales a utilizar. En este sistema se utiliza el canal 1. El bloque "SM1281\_Channel" permite configurar parámetros específicos de canal y requerimiento de datos *raw* del canal en específico. Este bloque también entrega de estado e información de errores del canal. Si se utiliza más de un canal, es necesario agregar tantos bloques "SM1281\_Channel" como canales a usar.

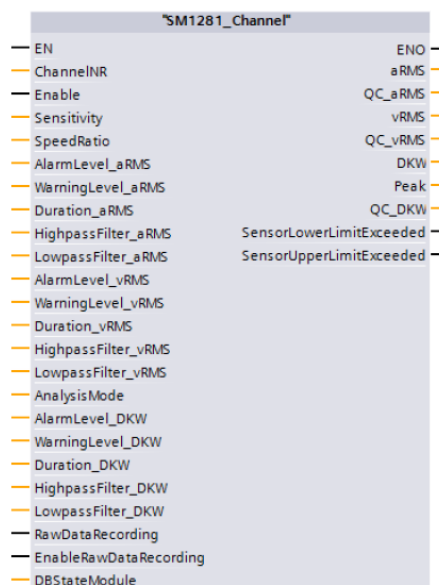


Figura 4.3: Representación bloque "SM1281\_Channel" dentro de programación LADDER de PLC. Fuente: SIPLUS CMS1200 Operating instructions, Siemens 2017.

El SM 1281 dispone de varios modos de operación. Al momento de energizar el dispositivo, este inicializa el modo “Startup”. Este modo toma aproximadamente entre uno y 3 minutos. Este modo comprueba que este todo en orden dentro del dispositivo para poder funcionar. Luego de superar este modo, el SM cambia al modo “IDLE: Wait for S7-1200”. En este modo el SM espera la comunicación con el bus hacia el PLC, estableciendo finalmente la conexión. Durante esta etapa aun no es posible de disponer de su página web. Una vez que es posible de comunicarse con el PLC, cambia al estado “STOP: Configuration” que permite empezar la configuración del mismo. En este modo se ejecutan los parámetros transmitidos por última vez por el PLC. Luego cambia al modo “STOP: System ready”, encontrándose el módulo listo para ser utilizado. En este modo es posible realizar las siguientes acciones:

- Borrar, exportar y restaurar datos de configuración.
- Reiniciar el dispositivo.
- Asignar parámetros de configuración de monitorización en el espectro de las señales captadas.
- Reinicio del dispositivo a sus valores de fábrica.
- Tareas administrativas pertinentes.

El ordenamiento de los modos de funcionamiento puede ser apreciado en la figura 4.4.

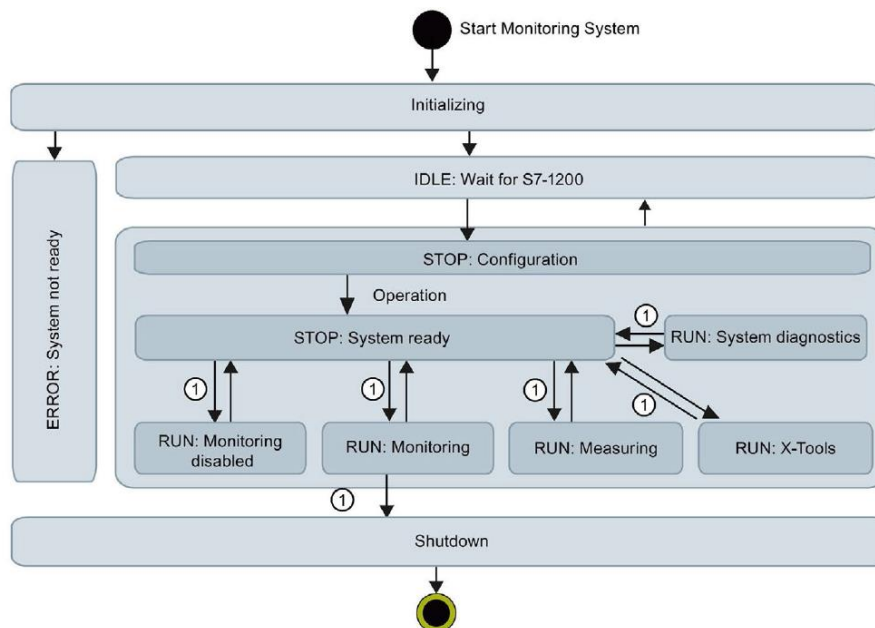


Figura 4.4: Esquema del ordenamiento de los distintos modos de funcionamiento del SM1281. Marcado con “1” los modos por requerimiento de usuario. Fuente: SIPLUS CMS1200 Operating instructions, Siemens 2017.

### 4.2.3. *WINCC*

La programación de la pantalla estaba presupuestada solo para la realización de una interfaz que muestra resultados numéricos, junto a la posibilidad de desplegar datos en un gráfico. Esta idea evoluciona a medida que se conocen mejor las capacidades de los distintos elementos del sistema. Finalmente se utiliza esta pantalla para el procesamiento de la señal de vibración.

El lenguaje de programación utilizado en este dispositivo es Visual Basic Script. Este lenguaje está integrado dentro de este entorno de programación para asistir en pequeñas rutinas dentro de las tareas necesarias para desplegar información en el HMI. Por ello, opciones avanzadas como debugger y disponibilidad de objetos para programación avanzada son bloqueados por el software de Siemens. La asistencia técnica de Siemens para este lenguaje de programación es de pago, haciendo complicado trabajar a un nivel avanzado sobre este lenguaje.





## 5. DESARROLLO DE SOLUCION IMPLEMENTADA

En el capítulo anterior se describió la instalación y puesta en marcha de distintos elementos utilizados en el sistema. En el presente capítulo se describe la solución implementada para conseguir el objetivo general del trabajo, posibilitar la incorporación de algoritmos avanzados de tratamiento de señal desarrollados por un tercero en los dispositivos Siemens, los cuales no están incluidos en las capacidades del software por defecto.

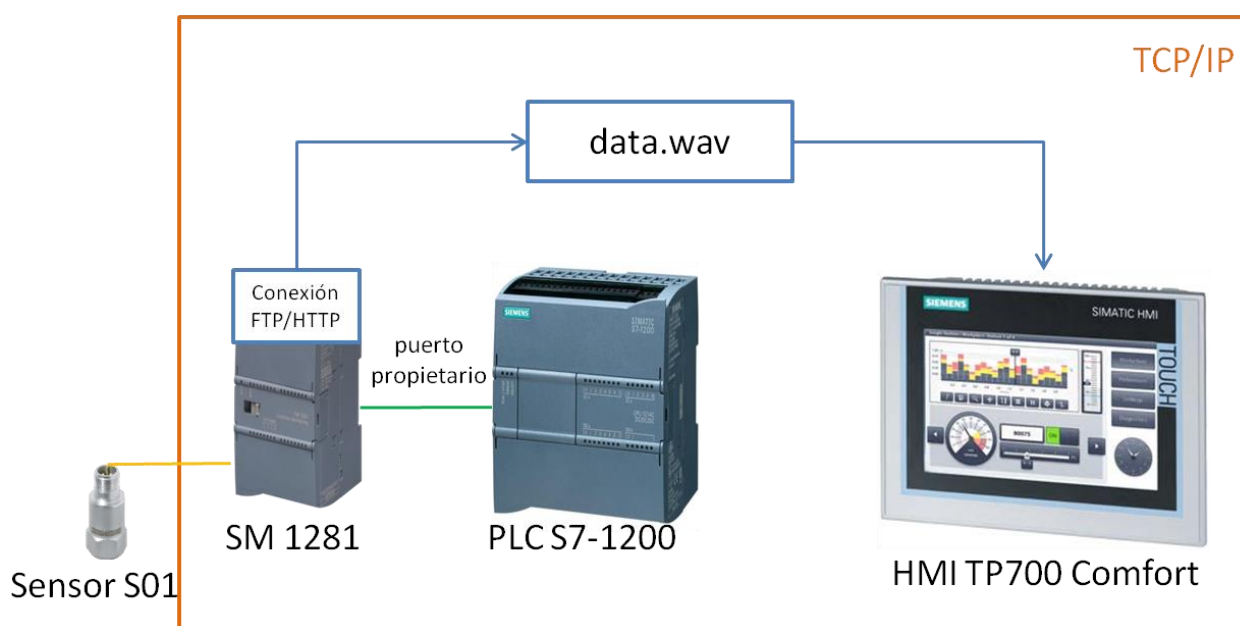


Figura 5.1: Esquema obtención archivo desde SM 1281 hasta HMI TP700. Fuente: elaboración propia.

En la figura 5.1 puede verse un esquema de esta solución, donde la comunicación se realiza utilizando la red TCP/IP para obtener la información captada por el sensor hasta exponer el valor en el HMI.

La señal de vibración es recogida a una frecuencia de 46.875 Hz, y el tiempo de ciclo del PLC es de 250 ms. Por tanto, el PLC no es capaz de obtener directamente los valores de la señal de vibración a medida que se generan, necesitando obtener estos valores (desfasados en tiempo) en un archivo. El SM puede enviar datos estadísticos de la vibración al PLC a través del puerto propietario; aRMS y vRMS. La única opción es la generación, por parte del SM, de un fichero que contenga todos los datos captados por el SM provenientes del acelerómetro. Una vez generado este archivo es necesario enviarlo al dispositivo capaz de abrirlo y tratarlo. Para ello, se deben analizar dos puntos: que opciones de comunicación dispone el SM y que dispositivo es capaz de comunicarse con el SM para obtener el archivo. La investigación para solucionar esta problemática será descrita a continuación.

### **5.1. Obtención datos de señal de vibración**

El dispositivo SM permite generar un archivo con los valores obtenidos por el sensor S01 de los últimos 10 segundos al momento de solicitar la grabación de los datos. Para generar este archivo, existen varias opciones:

- Violación de valor límite, configurado en análisis CMS integrado en SM 1281.
- Comandado por usuario al momento de utilizar la pagina web del dispositivo
- Gestionado por el programa configurado en el S7-1200.

De estas tres opciones, la violación de valor límite configurado no es necesario porque no se busca monitorizar el valor de un parámetro en especial, sino demostrar la posibilidad de incluir un algoritmo avanzado utilizando los datos obtenidos desde este dispositivo. Por otro lado, comandar por usuario al momento de utilizar la página web del dispositivo supone la intervención de una persona cada vez que se desea el análisis de los datos, además de incorporar un computador al sistema para ingresar a la página web del dispositivo. Ambas opciones se tratan de evitar, siendo descartada esta opción. La opción viable para la elaboración de este sistema, es escribir un programa que permita la creación del archivo usando el PLC. Aunque este sistema suponga la intervención de una persona para iniciar la adquisición de datos, solo será al hacer clic a un botón dentro de la interfaz HMI, pero posible de ser automatizado.

#### **5.1.1. GENERAR ARCHIVO EN SM 1281 USANDO EL PLC**

Para la creación del archivo dentro de la memoria del SM, se utiliza el PLC para enviar comandos al SM. Estos comandos cambian de estado el SM como se describe en el punto 4.2.2.1, automatizándose por un programa creado en VBS. Esto permite que la generación del archivo sea posible con trigger, que a efectos de este proyecto es sustituido por un botón en la pantalla del HMI. El programa que genera el archivo se llama “generar\_10”. Su código fuente se encuentra en el Anexo I. La secuencia de comandos generada por el programa es el siguiente:

- Cambia el modo de operación a “RUN: Measuring”.
- Activa el cambio de modo con el bit de activación de modo.
- Envía el bit de petición de grabación de datos *raw*.
- Espera 15 segundos para que sean grabados los datos de los últimos 10 segundos adquiridos desde el momento que se envía la petición al SM.
- Se cambia el modo del SM a “STOP: System Ready”.

Entre cada comando se espera el tiempo necesario para que reaccione el módulo, utilizando aproximadamente 40 segundos desde el momento que se ejecuta el programa hasta la creación del archivo.

Una vez creado el archivo es necesario descargarlo para ser utilizado por los dispositivos del sistema. Esto será tratado en el siguiente apartado.

### **5.1.2. DESCARGA DE ARCHIVO**

El SM permite ser conectado a una red TCP/IP consintiendo comunicarse mediante varios protocolos de comunicación que utilizan esta plataforma. La conexión por TCP/IP habilita la descarga del archivo creado con el método descrito en el apartado anterior. Las conexiones posibles son los siguientes protocolos: FTP y WebDAV.

El protocolo WebDAV [64] es un set de extensiones de comunicación HTTP junto con codificación XML, el cual permite a los usuarios editar y manejar archivos de forma colaborativa. Su característica principal es de bloquear archivos al momento de ser usados, evitando así su reescritura por parte de otros usuarios.

El protocolo FTP [65] es un protocolo de red IP simple, que permite a los usuarios transferir archivos entre computadores al momento de establecer la conexión. Su arquitectura está basada en el modelo cliente/servidor. No permite el bloqueo de archivo, permitiendo modificación simultánea de dos usuarios al mismo archivo. Aparte de estas opciones, es posible de descargar del dispositivo SM a través de su página web integrada.

Por otro lado, junto al análisis de las opciones que dispone el SM para descargar el archivo, hay que considerar las posibilidades de conexión utilizando solo el PLC y el HMI, buscando evitar incluir el uso de un computador. Por un lado, el PLC dispone de funciones escritas en SCL que permiten la comunicación FTP [66]. Esta comunicación obliga a escribir un programa que permita descargar el archivo a la memoria del PLC. Como se describió anteriormente, una de las limitaciones que presenta el PLC es la memoria disponible, teniendo solo 75kb de memoria disponible. La comunicación FTP utilizando el PLC como cliente está diseñada para intercambiar un máximo de información de 8 kb por mensaje [67], mientras que el archivo generado por el SM tiene aproximadamente 2Mb. Por este motivo no es posible utilizar el PLC.

En contraste, el HMI dispone de memoria y un lenguaje de programación que permite el manejo de archivos a nivel de fichero. El HMI dispone de memoria interna suficiente y permite usar una memoria USB para el manejo de archivos. Además, Windows CE dispone de un navegador de internet llamado “Internet Explorer”, que permite acceder a la página del SM desde el HMI directamente, pudiendo descargar el archivo desde el módulo SM hasta la memoria del HMI para su posterior procesamiento.

Por tanto, a fin de evitar el uso de dispositivos externos al sistema para su funcionamiento, se buscaron alternativas para automatizar el proceso de descargar el archivo desde el SM al HMI. Las opciones se agrupan en la tecnología a utilizar: FTP o WebDAV.

### 5.1.2.1. ALTERNATIVAS COMUNICACIÓN WEBDAV

Para realizar la conexión hacia el SM utilizando el protocolo WebDAV, una alternativa es utilizar un software cliente que permita el acceso al SM. Este software debe ser escrito para Windows CE por tener una estructura diferente a la edición de Windows de escritorio. Originalmente Windows CE soporta procesadores SH3 y MIPS, por lo que la gran mayoría de software encontrado estaba creado para soportar estos procesadores. El procesador del HMI es un procesador AMD de 32 bits, por lo que no fue posible encontrar un programa que cumpliera con las características nombradas anteriormente.

Una segunda opción fue crear un archivo en VBScript para ejecutar comandos *batch* que permita ejecutar en consola el comando *net use*. El comando *net use* permite conectarse a un recurso compartido a través de la red local. En teoría se puede acceder a recursos compartidos con este comando, pero luego fue encontrado que este recurso (carpeta) debe ser creado solo con el asistente de Windows. Las carpetas compartidas por WebDAV no fueron posibles de encontrar utilizando este método, siendo necesario buscar otra opción.

La implementación de WebDAV se basa en las instrucciones HTTP y extensión XML, por lo que se planteó la opción de utilizar un programa escrito en VBScript que permitiera utilizar el objeto XMLHTTP dentro del lenguaje VBS, realizando de esta forma la comunicación. El software de Siemens dentro del HMI tiene bloqueado el acceso a este objeto, quedando descartada esta opción.

Para buscar una opción válida, fue necesario profundizar en los métodos que utiliza Windows como sistema operativo para comunicarse a través de la red. Esto permitió encontrar que todas las comunicaciones a través de la web utilizan la biblioteca de enlace dinámico *wininet.dll*. Esta librería contiene funciones utilizadas para comunicarse a través de Internet, incluyendo HTTP y FTP. Para su uso, es necesario direccionar este DLL dentro del programa escrito en VBS, para luego importar cada función a utilizar. Esta vía, si bien es válido en principio, tiene una alta complejidad, descartándose su realización.

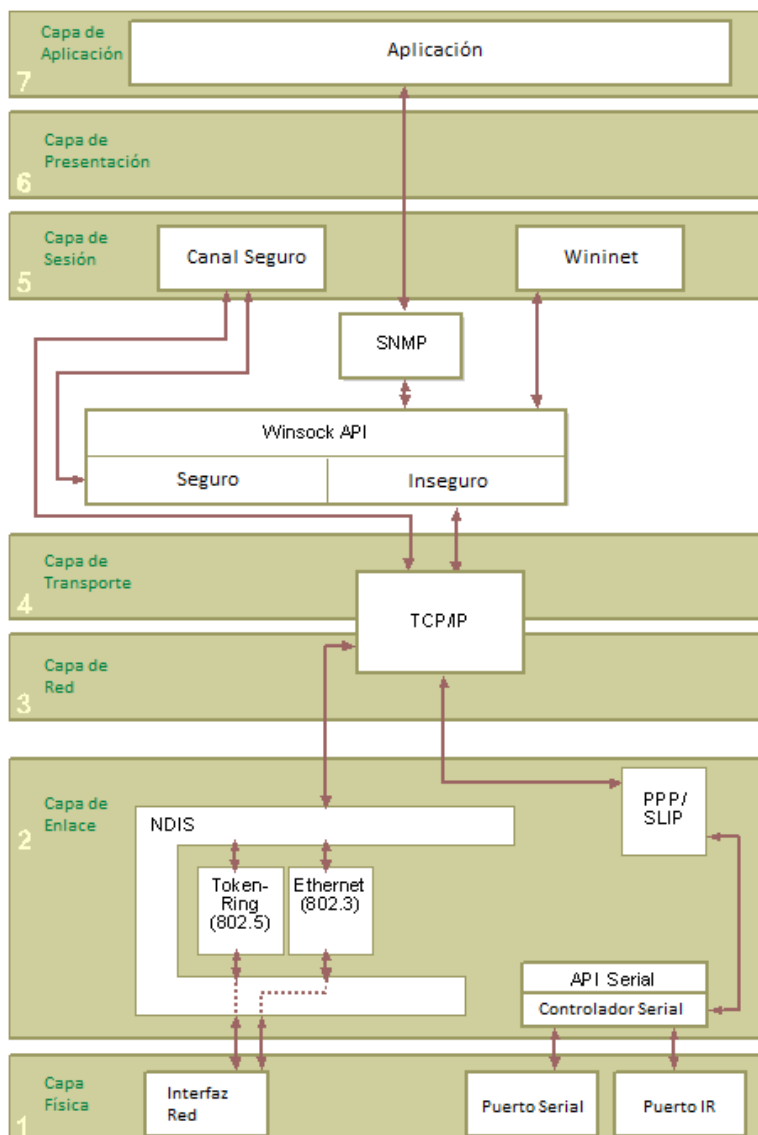


Figura 5.2: Esquema ubicando wininet.dll dentro de modelo OSI. Fuente: “WinInet and the OSI Model”[68], Microsoft 2004.

### 5.1.2.2. ALTERNATIVAS COMUNICACIÓN FTP

Al igual que para la comunicación WebDAV, existe la posibilidad de usar un cliente FTP. Al igual que el apartado anterior, no se encontró un cliente gratuito que pudiera funcionar con el procesador del HMI (procesador AMD de 32 bits), teniendo por ello que desechar esta opción.

Una segunda opción sería la creación de un programa que utilice *wininet.dll*. Situación similar ocurre en este caso, donde se debe direccionar a este DLL y luego importar las funciones necesarias para su uso.

Analizando las diferentes alternativas de comunicación para evitar que una persona extraiga manualmente el archivo utilizando el navegador de internet instalado dentro del HMI, no queda otra opción que utilizar asistencia humana mas el uso de

un ratón USB. El motivo de utilizar un ratón USB es hacer clic derecho al nombre del archivo a descargar al momento de ingresar a la página del FTP del SM, desplegando las opciones disponibles que no se acceden con un clic normal, como aparece en la figura 5.13. La pantalla táctil del HMI utilizado no permite este clic derecho, obligando el uso de un ratón USB para descargar el archivo.

### 5.1.3. EXTRACCIÓN DE DATOS RAW DESDE ARCHIVO WAV

Una vez que el archivo generado por el SM se encuentra en la memoria del HMI, se puede extraer la información de la señal de vibración. Luego de contactar con servicio técnico de Siemens, se confirma que el archivo tiene el formato WAV, como los archivos de audio que son posibles de encontrar en un computador de escritorio. Teniendo como nombre formal WAVE (popularmente conocido por su extensión como WAV y por el cual es citado en el resto del documento), este formato es un subconjunto de la especificación RIFF de Microsoft para el almacenamiento de archivos multimedia [69]. Un archivo RIFF comienza con un encabezado seguido de una secuencia de fragmentos de datos. Un archivo WAV es un archivo RIFF con un solo fragmento "WAVE", constituido de dos subunidades: un fragmento "fmt", que especifica el formato de datos y un fragmento "de datos" que contiene los datos de muestra. La figura 5.3 ilustra la estructura de un archivo WAV.

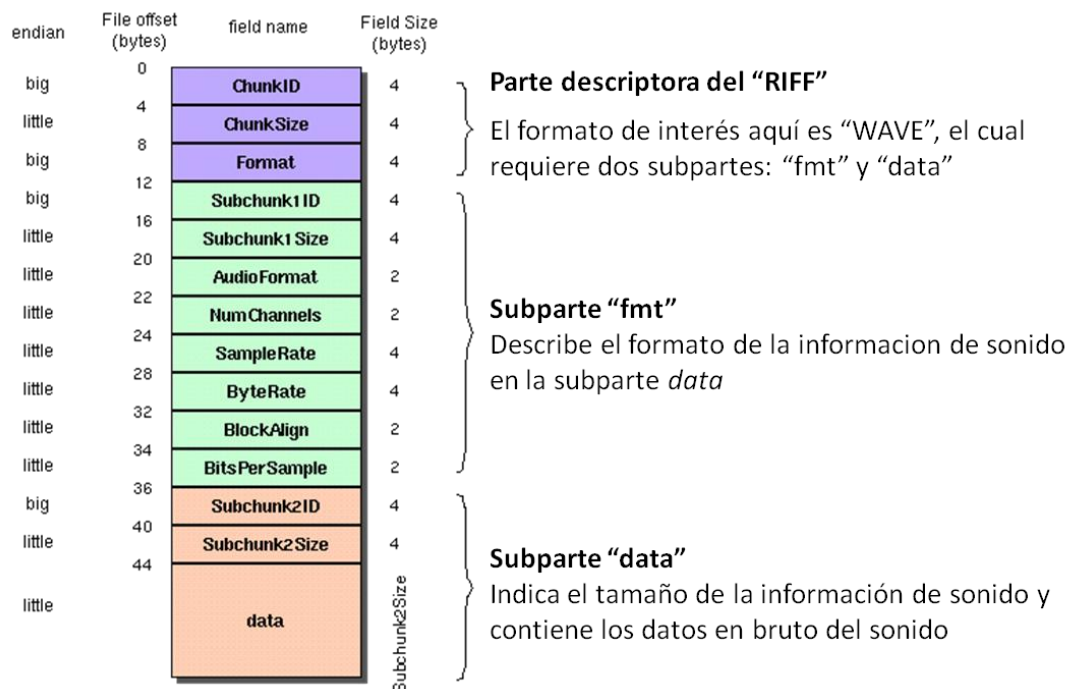


Figura 5.3: Esquema de ordenamiento de bytes en un archivo WAV. Fuente: elaboración propia.

Dentro de la subparte "data" se encuentran los valores de cada muestra, compuestos de 2 bytes (16 bits en total) por muestra. Estos 2 bytes son el complemento a 2 de enteros con signo, con un rango de valores entre -32.768 a 32.767 [70]. Cada

muestra aparece en el archivo utilizando el orden de almacenamiento Little endian; el byte de menor valor se registra primero. A modo de ejemplo, la figura 5.4 muestra un esquema donde se aprecia los primeros 53 bytes de un archivo tipo WAV. Por tanto, la cantidad de bytes total a utilizar será el doble de la cantidad de muestras efectivas de la señal.

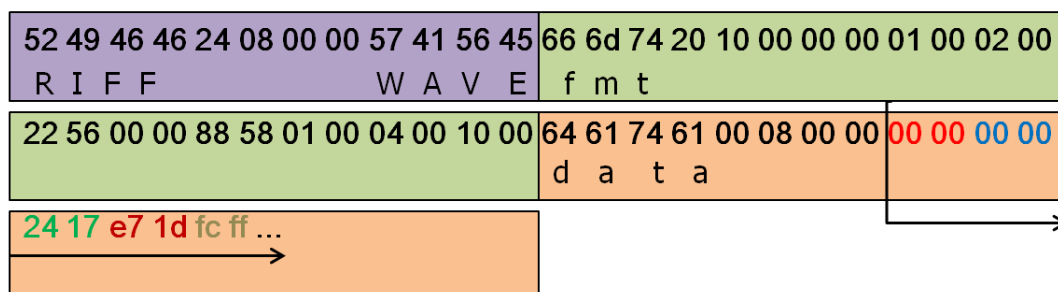


Figura 5.4: Esquema de los primeros 53 bytes en un archivo WAV. Fuente: elaboración propia.

Los datos de un archivo WAV se sitúan desde el byte 45 hasta el final del archivo, los cuales corresponderían a los datos reales de la señal de vibración. El archivo que se obtiene del SM contiene las señales de velocidad de rotación de la máquina (escrita en la configuración del canal al cual se encuentra conectado el sensor al SM) en revoluciones por minuto (valor constante) y la señal de vibración. Además, no es posible de disminuir esta frecuencia dentro de la configuración del SM a menos de 46.870 muestras por segundo. Para ver el archivo a nivel de byte se utilizaron los software WinHex [71] y Notepad++ con el plug-in HEX-Editor [72]. Se observó que el formato de la cabecera del archivo WAV cambia con respecto al estándar, comenzando la información de la señal en el byte 589. Una explicación es que esté diseñado para su uso con el software CMS X-Tools [73]. Por ello, los datos *raw* no inician en el byte 45, sino que empiezan en el byte 589. El archivo WAV proveniente del módulo SM tiene dos canales, un canal con la señal y otro con el número de revoluciones del equipo. Esto hace que cada muestra tenga 4 bytes (2 bytes por canal). Para el sistema solo se utiliza un canal, omitiendo la extracción del segundo canal. Por tanto, para cada 4 bytes se utilizan 2. Es reordenar estos bytes y convertirlos a un valor decimal. Para convertir a decimal, hay que comprobar si es un número positivo o negativo para luego aplicar la operación binaria complemento a 2 de los bytes, obteniendo finalmente el número que corresponde. Esta lógica se resume en la figura 5.5.

	byte1	byte2		byte1	byte2
Dato1:	e7	1d		Dato2:	fc ff
	1d	e7	Invertir bytes		ff fc
	1d ≥ 80?	NO	Comprobar signo		ff ≥ 80? SI
	(1d e7) →	7655	Convertir a decimal		~(ff fc)+1 → 5
	7655/32767		Valor por unidad		5/32767
	0.233619		Valor final		-0.000152593

Figura 5.5: Esquema de ordenamiento de bytes en un archivo WAV. Fuente: elaboración propia.

Para realizar el algoritmo descrito en la figura 5.5 fue necesario realizar una serie de pasos en el programa de VBS. Primero, se obtienen todos los valores byte a utilizar y se escriben en un archivo de extensión TXT. Este archivo contiene los valores bytes del archivo WAV original como caracteres ASCII, los cuales son nuevamente leídos. Para cada par de bytes leído, estos son convertidos en valores enteros, ejecutándose luego el esquema de la figura anterior como valores decimales, siendo estos valores guardados en un nuevo archivo TXT. Este proceso se realiza en el programa “convertir\_20” que se encuentra en el anexo I.

Comparando el archivo WAV original con el archivo TXT final generado con los valores decimales, el tamaño del archivo aumenta 5,44 veces en comparación al tamaño del archivo WAV. Esto se explica porque cada número en el archivo WAV corresponde solo a 2 bytes, pero en el archivo TXT cada dígito (y el símbolo punto) dentro de un valor corresponde a un byte, haciendo que de 2 bytes pasen a 6 o 12 bytes.

## 5.2. Procesamiento

El apartado anterior explica el proceso de obtención de los datos de la señal de vibración, ya listos para ser procesados. En este apartado se van a exponer las posibilidades existentes para el procesamiento.

### 5.2.1. Lenguaje de Programación para PLC

Si se desea realizar el procesamiento en el PLC, es necesario saber que lenguajes de programación se pueden utilizar. En un principio, TIA Portal permite usar cinco lenguajes:

- Escalera (LAD/LD/KOP): lenguaje básico para programación de PLC basado en lógica de escalera. La operatoria de señales binarias están representadas por arreglos de contactos en serie y/o paralelo. Funciones complejas como operaciones aritméticas, son representadas por cajas que son dispuestas en el programa como un contacto más.
- Diagrama de bloques de función (FBD/FUP): basado en sistemas de circuitos electrónicos. Las operaciones binarias son implementadas por bloques.



- Lenguaje de control estructurado (SCL): lenguaje enfocado a las programación de algoritmos complejos o para tareas que utilizan manejo de área de datos.
- Lista de declaraciones (STL/AW/IL): el programa se desarrolla a través de secuencias de declaraciones. Cada declaración contiene una especificación de que se debe hacer.
- Control secuencial (GRAPH): lenguaje estructurado como un control secuencial en donde prevalece una secuencia de acciones. Se permiten pasos individuales y ramificaciones.

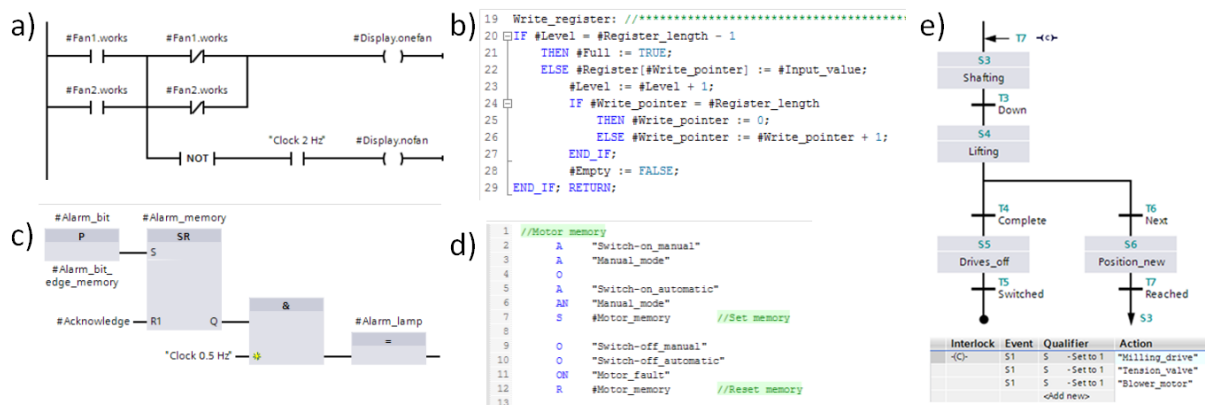


Figura 5.6: Ejemplos de los lenguajes de programación para PLC disponibles en TIA Portal. a) lenguaje de escalera. b) lenguaje de control estructurado. c) lenguaje de diagrama de bloque de función. d) lenguaje de lista de declaraciones e) lenguaje de control secuencial. Fuente: “Automating with SIMATIC S7-300 inside TIA Portal”, Hans Berger 2014.

Los lenguajes STL y GRAPH no son soportados por la línea S7-1200. Por ello, quedan LAD, FBD y SCL. Se elige el lenguaje SCL por ser el lenguaje de programación que permite escribir algoritmos más complejos.

### 5.2.1.1. LIBRERÍA OSCAT

Tras analizar las capacidades del de programación SCL, se concluyó que no está preparado para manejar matrices ni números complejos. Buscando soluciones alternativas, se encontró una librería llamada OSCAT [74]. La librería OSCAT está escrita siguiendo la norma IEC61131-3, permitiendo su uso en distintos autómatas incluyendo Siemens. Çesta librería dispone de dunciones que pueden manejar números complejos. Esta alternativa llevaría a reescribir todas las funciones utilizadas en MATLAB desde 0, exigiendo una amplia dedicación. Como la versión de STEP7 (versión basic) en TIA Portal no permite el uso de propiedades de PLC necesarias para el funcionamiento de la librería, no ha sido posible utilizarlo como opción.

Considerando las capacidades del PLC, se observa que su memoria no es suficiente para el manejo de una cantidad superior a 3.500 valores (memoria de 75 kb). Dado que la señal de vibración se encuentra a una velocidad de muestreo de 46.875 Hz, no es posible tratar todos los datos para realizar el análisis.

Este problema puede ser sorteado utilizando un PLC de una línea mayor, como son los PLC de la línea S7-1500. Esta línea puede tener desde 150 kb de memoria hasta 24 Mb, siendo posible procesar toda la señal obtenida del SM. Sin embargo, al no ser posible disponer de otro PLC con mejores prestaciones dentro de este proyecto, se analizaron otras alternativas.

### 5.2.2. CÓDIGO MATLAB

Una alternativa para procesar los datos obtenidos por el SM1281 es portar el código desde MATLAB a SCL. MATLAB dispone de una herramienta que permite convertir modelos desarrollados en Simulink© a lenguaje SCL, llamado Simulink© PLC Coder [75]. Esta conversión tiene las mismas limitaciones que la conversión del programa en MATLAB a C++ [76]. El paso del lenguaje C++ a SCL también presenta limitaciones, siendo necesario comprender cómo funciona el código original. MATLAB dispone de una página que especifica las limitaciones de esta conversión, pudiéndose mencionar:

- No es posible el uso de números complejos.
- No permite referencias a otros modelos.
- Soporte limitado para funciones matemáticas.
- Parámetros y señales de tamaño variable no soportado.
- Objetos y bloques del sistema de MATLAB.
- En los bloques de funciones de MATLAB solo son soportados funciones de MATLAB estándar.
- Funciones de Toolbox que no son soportados.

#### 5.2.2.1. PROGRAMA EN MATLAB

La estructura de los programas desarrollados en MATLAB son expuestos en la figura 5.7.

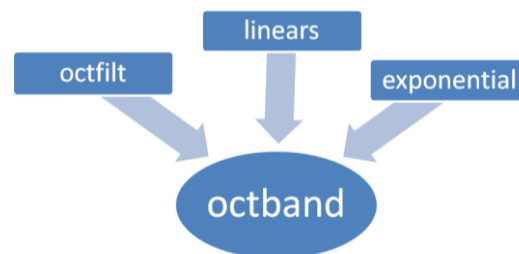


Figura 5.7: Esquema de ordenamiento de los programas desarrollados en MATLAB. Fuente: elaboración propia.

La figura anterior muestra como se relacionan las funciones creadas. La función principal “octband” utiliza las funciones “octfilt”, “linears” y “exponential” para

realizar el cálculo de la banda especificada en sus argumentos. Los programas pueden ser revisados en el Anexo I. Estas funciones a su vez utilizan funciones de MATLAB. Las funciones son ordenadas y expuestas en la tabla 5.1.

Tabla 5.1: Funciones de MATLAB utilizadas por cada función creada.

Función	octband	octfilt	linears	Exponential
Funciones utilizadas	zeros	Sqrt	Length	Length
	Find	butter	Sqrt	Sqrt
	Round		Log10	Log10
	Resample		Max	Max
	Strcmpi			
	Filter			

El programa desarrollado en MATLAB permite el cálculo de varias bandas de octava al mismo tiempo de una señal especificada. Para este trabajo será desarrollado el cálculo de una sola banda.

El software MATLAB dispone de multitud de herramientas de análisis de datos y permite cálculos de complejidad matemática tal que utiliza variables complejas y operaciones con matrices. El reto consiste en adaptar el código escrito en este entorno de programación a un lenguaje con capacidades mucho más reducidas como lo es el lenguaje SCL, donde no posee por diseño el manejo de números complejos y operaciones con matrices.

### 5.2.2.2. ANÁLISIS FUNCIONES EN MATLAB

Para una correcta conversión del código en MATLAB a lenguaje SCL es necesario entender en detalle los cálculos de las subfunciones utilizadas.

De todas las funciones utilizadas MATLAB, *butter* tiene una mayor complejidad de cálculo. Esta función entrega los coeficientes de la función de transferencia de un filtro digital de orden N para una frecuencia de muestreo especificada. Para realizar este cálculo, utiliza números complejos y matrices para la obtención de valores propios de polinomios, ambos tipos de cálculos no posibles en SCL.

El módulo encargado de realizar la conversión (llamado Simulink PLC Coder) tiene como exigencia para *butter* que los valores de entrada deben ser constantes reales. Al cumplir con esta exigencia, el código generado en SCL contiene solamente 2 vectores de valores reales, siendo estos los coeficientes mencionados. Estos valores son utilizados por la función *filter* para realizar el cálculo de la banda de octava. Se deduce que los valores entregados por la función *butter* serán siempre los mismos, ya que cada banda esta previamente establecida. Entonces, el programa puede reformularse como una función de transferencia, el cual va cambiando según la banda a filtrar y la velocidad de muestreo de la señal. Esto simplifica las condiciones de trabajo para el cálculo para una sola banda de octava. Si es posible

obtener los valores para una banda de octava, reemplazando correctamente los coeficientes será posible calcular para cualquier banda requerida.

Por tanto la programación se enfoca a la realización del código necesario para adaptar la función de transferencia en código SCL, creándose el bloque de función Calc\_dB. La herramienta PLC Coder genera un archivo de extensión SCL, importado al proyecto donde se realiza la programación del PLC. En el Anexo I es posible revisar el código generado.

### 5.2.3. *CÓDIGO EN VISUAL BASIC SCRIPT*

Utilizando el lenguaje SCL y dadas las limitaciones de hardware del PLC, se traslada el procesamiento al HMI. Para ello, fue necesario analizar la traducción del código MATLAB a VBS. Es posible traducir el código creado en SCL a VBS porque existen funciones matemáticas suficientes en VBS. La tabla 5.2 muestra las funciones matemáticas disponibles en VBS.

Tabla 5.2: Funciones matemáticas disponibles en Visual Basic Script.

Función	Descripción
Abs	Devuelve el valor absoluto de un número especificado
Atn	Devuelve el arcotangente de un número especificado
Cos	Devuelve el coseno de un número especificado (ángulo)
Exp	Devuelve la constante "e" elevado a una potencia
Hex	Devuelve el valor hexadecimal de un número especificado
Int/Fix	Devuelve la parte entera de un número especificado
Log	Devuelve el logaritmo natural de un número especificado
Oct	Devuelve el valor octal de un número especificado
Rnd	Devuelve un número aleatorio menor que 1 pero mayor o igual a 0
Sgn	Devuelve un entero que indica el signo de un número especificado
Sin	Devuelve el seno de un número especificado (ángulo)
Sqr	Devuelve la raíz cuadrada de un número especificado
Tan	Devuelve la tangente de un número especificado (ángulo)

De la tabla anterior se puede ver la biblioteca de funciones disponibles para trabajar en VBS. Para el cálculo de la banda de octava, la aritmética posible en VBS tiene capacidades similares a las del PLC. Esto permite que el dispositivo sea una alternativa al procesamiento de la señal de vibración. Por ello, se tiene finalmente un código en MATLAB que es traducido al lenguaje SCL y VBS. La comparación de estos algoritmos y su posterior evaluación se realiza en el capítulo seis.

### 5.3. *Interfaz HMI*

A continuación será expuesta la interfaz gráfica elaborada para realizar todos los pasos descritos conjuntamente.

Primero, es necesario ejecutar al programa previamente cargado en el HMI haciendo clic en el botón “Start” dentro de la ventana Start Center V14.0.1.3, como indica la figura 5.8.

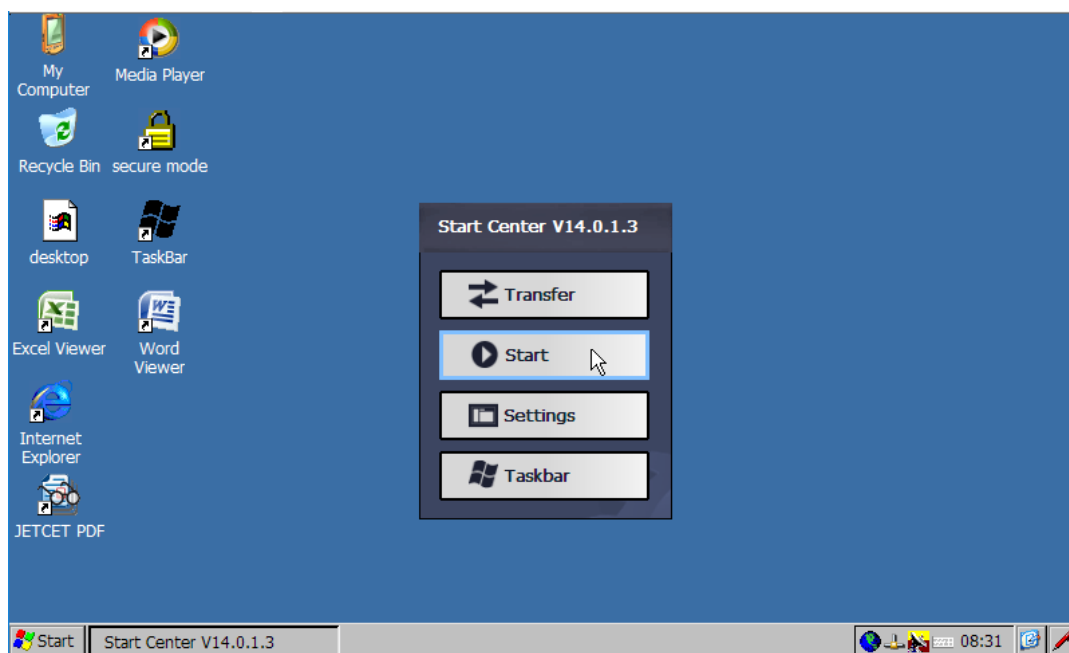


Figura 5.8: Esquema de ordenamiento de bytes en un archivo WAV. Fuente: elaboración propia.

La primera pantalla que aparecerá al ejecutar el botón Start será la pantalla principal, tal como aparece en la figura 5.9.

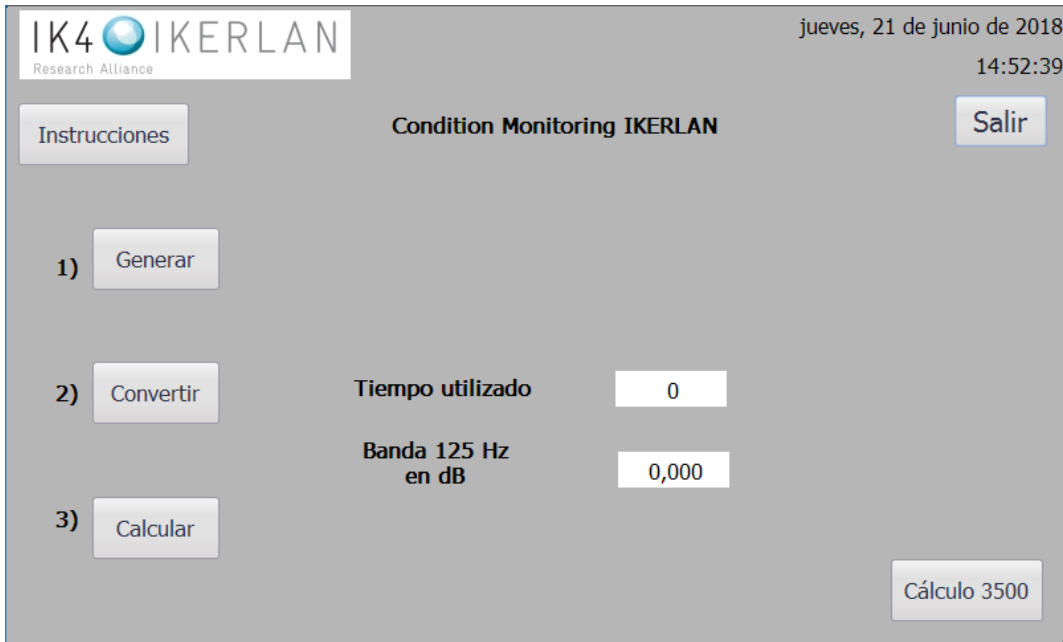


Figura 5.9: Esquema Imagen principal de programa. Fuente: elaboración propia.

En esta versión de la interfaz se realiza paso por paso la conversión del archivo hasta obtener el valor de ganancia en la banda de octava de 125 Hz. Este proceso sería posible de automatizar con solo el uso de un botón (salvo el descargar el archivo del SM). El primer paso dentro del programa es generar el archivo WAV dentro del SM. Para esta labor se debe hacer clic en el botón “Generar”. Mientras se realiza la rutina, aparecerá un mensaje “Procesando” en pantalla, lo cual indica que se debe esperar para seguir al siguiente paso. Una vez se haya finalizado la ejecución de este paso, se debe copiar el archivo WAV a la memoria del HMI.

La interfaz contiene un botón para mostrar las instrucciones que el usuario debe seguir para copiar el archivo WAV a la memoria local del HMI. Al hacer clic al este botón, aparece la lista de instrucciones como en la figura 5.10.

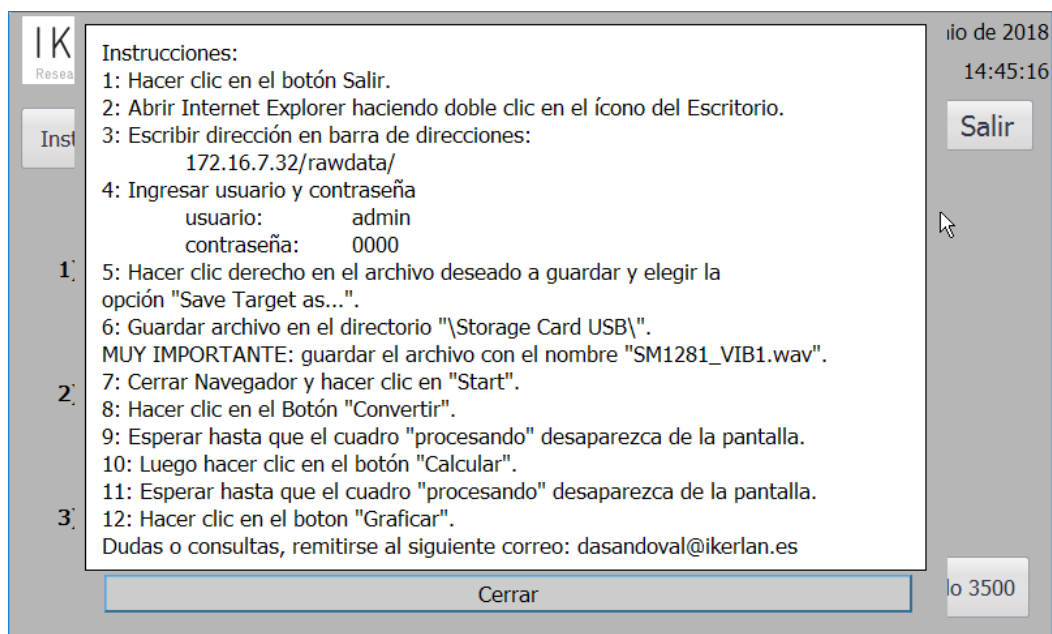


Figura 5.10: Ventana emergente con instrucciones para obtener archivo WAV. Fuente: elaboración propia.

Como aparece en las instrucciones, es necesario hacer clic en el botón cerrar y luego en el botón Salir. Hecho esto, se realiza la apertura del navegador de internet “Internet Explorer”, como muestra la figura 5.11. En esta figura se muestra donde se ubica el ícono del navegador web dentro del escritorio de Windows CE.

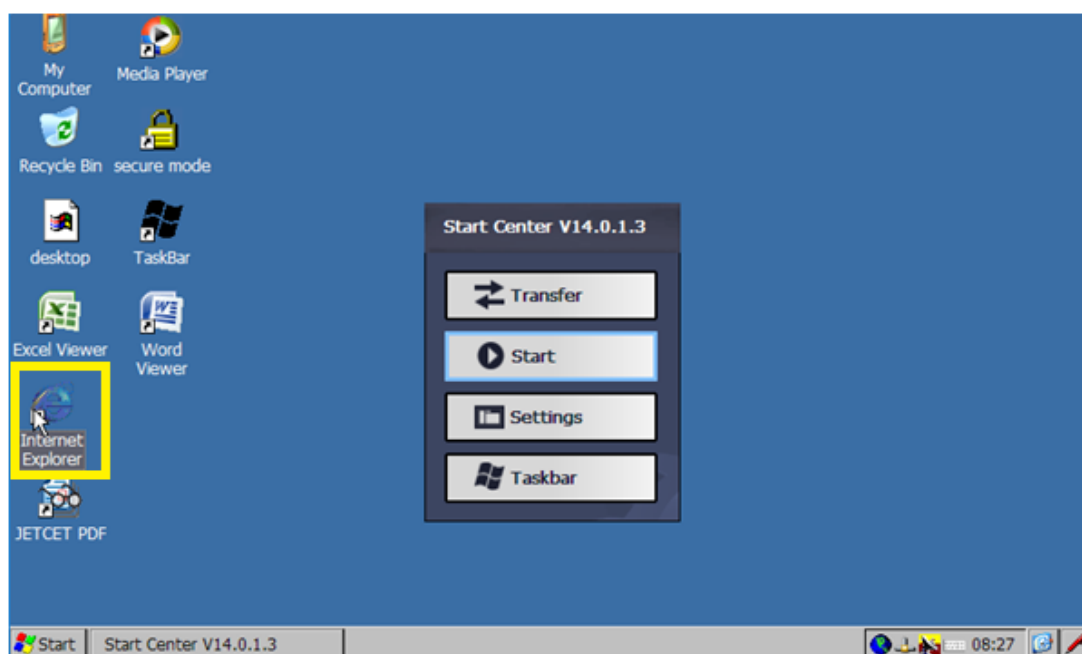


Figura 5.11: Ubicación icono de Internet Explorer en escritorio de Windows CE. Fuente: elaboración propia.

Una vez en el navegador web, será necesario introducir la dirección 172.16.7.32/rawdata/. Esto permitirá acceder directamente al directorio donde se

encuentran almacenados los archivos generados por el SM1281. Será necesario introducir el nombre de usuario y la clave para entrar a esta página (servidor FTP), los cuales por defecto son Admin y 0000. La figura 5.12 muestra lo descrito.

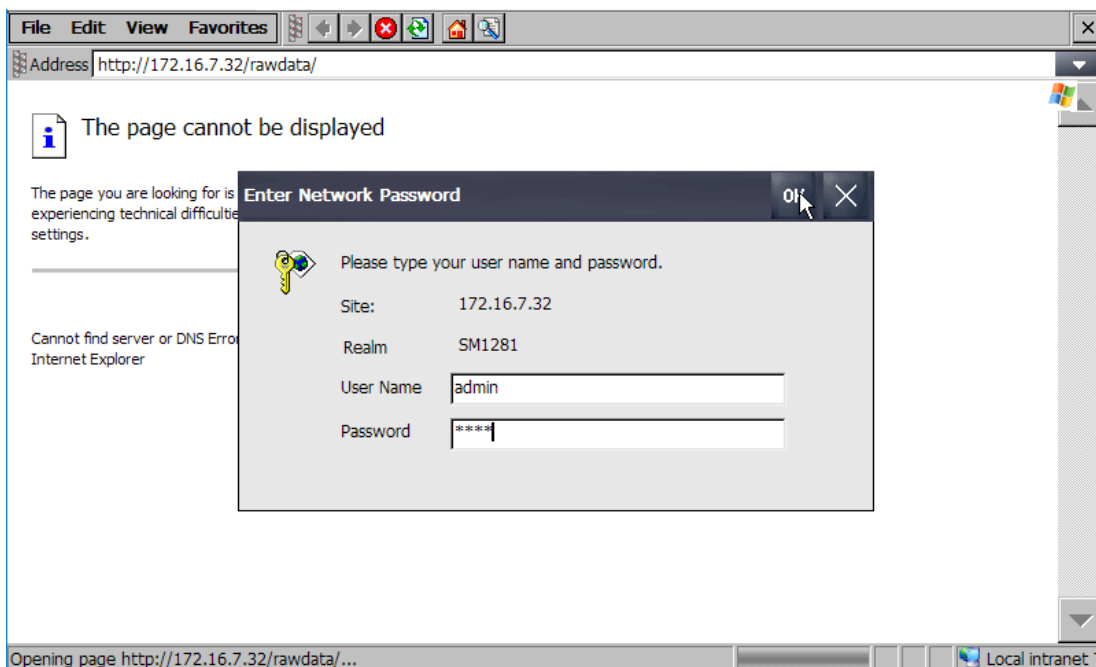


Figura 5.12: Ingreso de usuario y clave en el navegador web. Fuente: elaboración propia.

Tras introducir el usuario al FTP del SM, será necesario guardar el archivo haciendo clic derecho encima del nombre del archivo. Para hacer clic derecho, es obligatorio el uso de un ratón USB, el cual puede ser conectado en uno de los puertos habilitados en el HMI. La figura 5.13 muestra este paso.



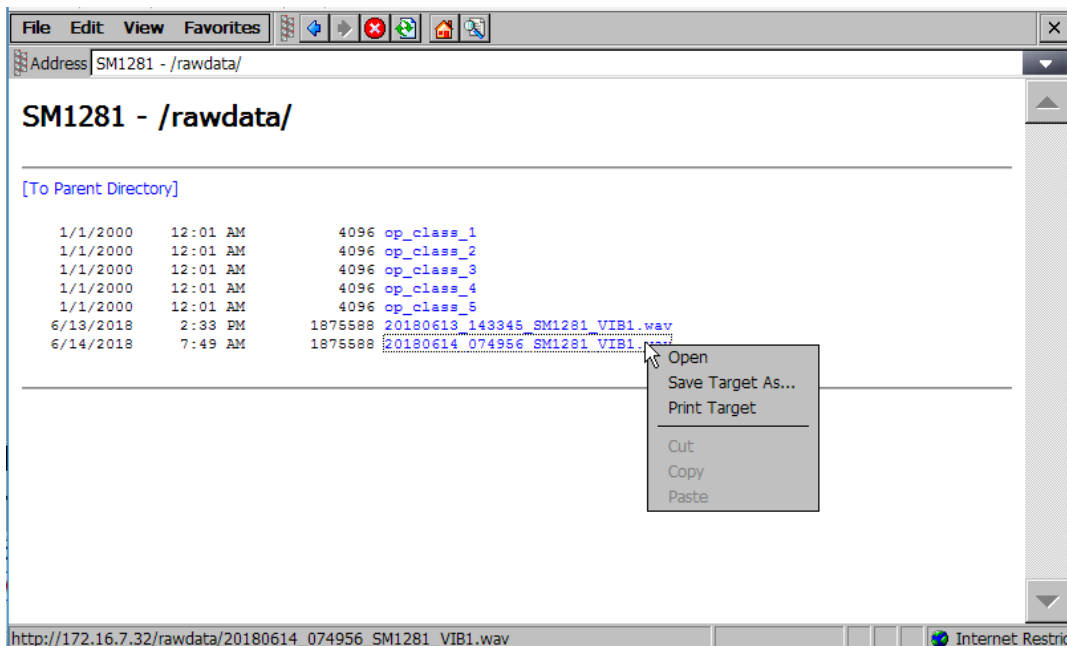


Figura 5.13: Directorio de archivos WAV. Fuente: elaboración propia.

Un detalle sumamente importante al momento de copiar el archivo a la memoria del HMI es cambiar de nombre el archivo a SM1281\_VB1.wav. Destacado en gris se muestra en la figura 5.14 la sección donde esto es posible. Si este cambio de nombre no se efectúa no será posible seguir con el programa.

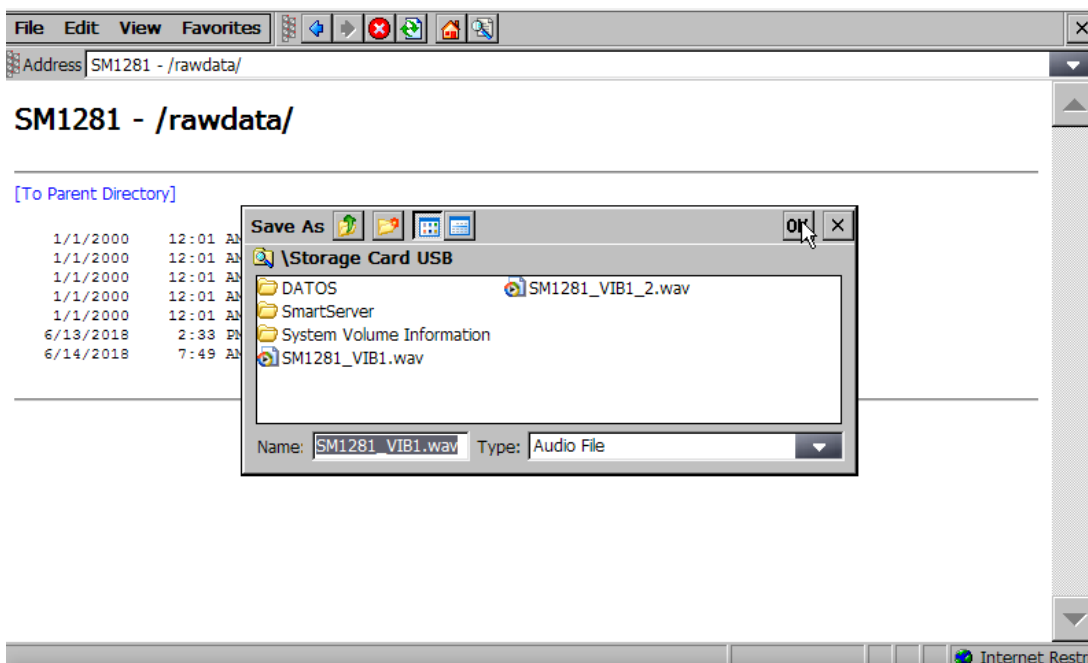


Figura 5.14: Ejecución de descarga de un archivo WAV. Fuente: elaboración propia.

Una vez guardado el archivo, basta con cerrar el navegador y volver a la interfaz. Teniendo el archivo WAV en memoria, es posible hacer clic al botón “Convertir”. Esto permitirá extraer los valores contenidos en el archivo WAV para luego ser

procesados. Por último, al hacer clic al botón Calcular, se realizara el cálculo matemático que obtiene el nivel de la señal en la banda de los 125 Hz.

De forma particular, al hacer clic al botón “Cálculo 3500” aparece el apartado donde se calcula lo descrito en el sub-apartado 5.2.2, mostrando los valores y tiempo utilizados, como aparece en la figura 5.15.

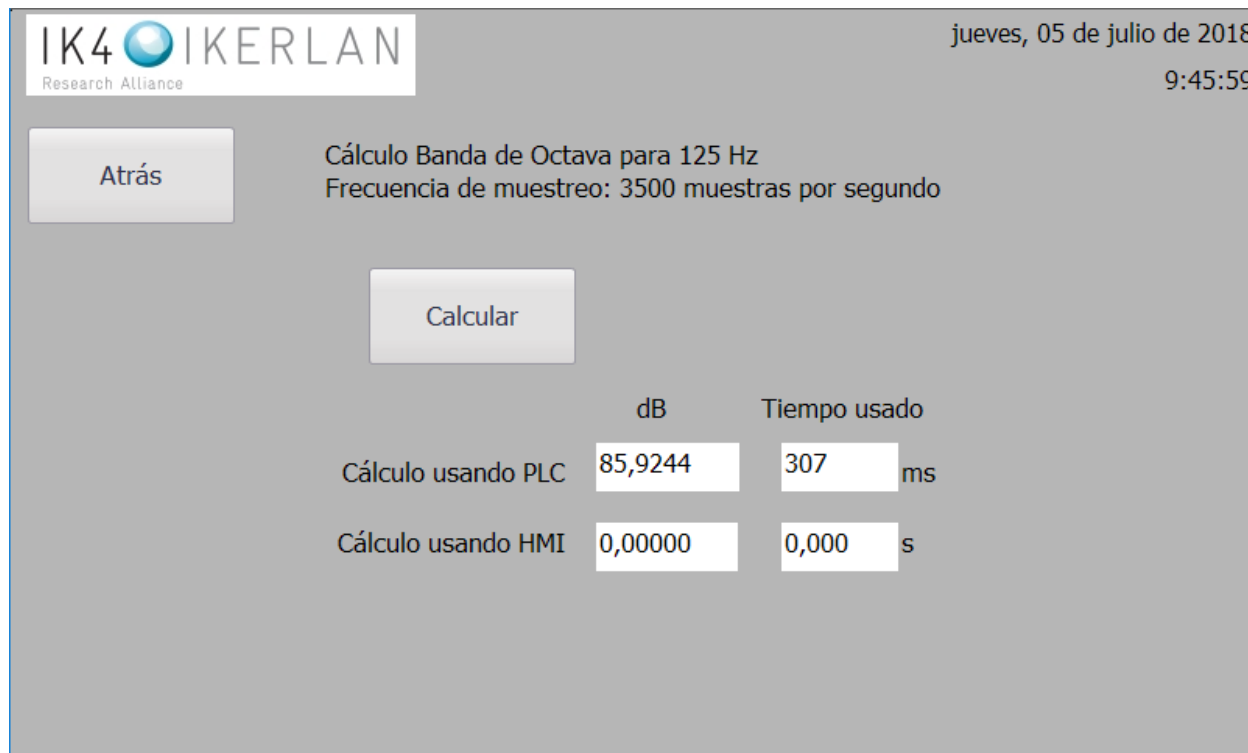


Figura 5.15: Interfaz de valores obtenidos por el cálculo de banda de Octava. Fuente: elaboración propia.

## 6. COMPROBACIÓN Y VALIDACIÓN DE SISTEMA

La idea fundamental del TFM fue evaluar e implementar algoritmos complejos de tratamiento de señal en dispositivos robustos. Como caso práctico se plantea la implementación de un cálculo de bandas de Octavas de una señal de vibración de un motor. El sistema desarrollado cuenta con un HMI que permite comandar y ver resultados. Como se expuso en el capítulo cinco, se pudo adaptar el código escrito en MATLAB en lenguaje SCL para ser ejecutado en el PLC, siendo además traducido al lenguaje VBS para usar el HMI como elemento de computo. En este apartado se muestra la validación de la solución, obteniendo valores reales del motor, procesándolos y comparándolos con un sistema de referencia.

### 6.1. Comparación códigos MATLAB, SCL y VBS

El escenario actual presenta un código creado en MATLAB traducido con una herramienta a lenguaje SCL, traducido luego manualmente a lenguaje VBS. A continuación se comparan estos tres códigos y se analiza su funcionamiento bajo los distintos dispositivos. Por un lado, el código ejecutado en MATLAB necesita la mayor cantidad de recursos de hardware que entrega un computador de escritorio. A continuación está la situación del código en SCL, el cual será ejecutado en el hardware del PLC, lo cual supone una disminución significativa de recursos. Por último, el código en VBS se ejecuta en el HMI, el cual no está diseñado en un principio para este tipo de tareas.

Dado que el lenguaje en SCL presenta la limitación de memoria, será utilizada una señal generada por un calibrador de acelerómetros. Este calibrador es el 394C06 de la marca PCB Piezotronics [77]. Este calibrador genera una señal de vibración de 1g de aceleración a 159.2 Hz. Por este motivo que al procesar la señal, será calculada la banda de octava a 125 Hz.

La señal obtenida directamente del SM está a una frecuencia de muestreo de 46.870 Hz. Por tanto, el tiempo que comprende 3.500 muestras de esta señal es aproximadamente 75ms, tiempo insuficiente para que sea representativo de la señal en general. La solución implementada fue remuestrear la misma señal a 3.500 Hz, permitiendo así tener 1 segundo de muestras. La generación de esta señal a la frecuencia deseada fue realizada con la función *resample* de MATLAB. La ganancia calculada utilizando los distintos lenguajes y el tiempo utilizado para su cálculo son presentados en la tabla 6.1.

Tabla 6.1: Ganancia en dB de señal a 159.2 Hz en banda de Octava de 125 Hz en a una velocidad de muestreo de 3.500 Hz.

	MATLAB	SCL	VBScript
Ganancia (dB)	85,7932	85,9244	85,7944
Tiempo de cálculo (ms)	0,091	307	2.531

De la tabla anterior se observa la gran diferencia de tiempo utilizado para el cálculo. Por un lado, el programa escrito en MATLAB utiliza menos de un milisegundo en obtener el valor, mientras que el código escrito en VBS tarda aproximadamente dos segundos y medio. Esta diferencia de tiempo puede ser explicado de que el HMI necesita abrir el archivo que contiene los valores, leerlo e importar a memoria RAM. Este procedimiento supone un uso considerable de tiempo y recursos. En contraste, el PLC tiene los valores de la señal almacenados directamente en un bloque DB, que permite realizar el cálculo. El PLC toma 307 milisegundos en realizar el cálculo, el doble de tiempo que acostumbra a utilizar para un ciclo de PLC. Tomando como referencia el valor de ganancia obtenido con el código escrito en MATLAB, el valor obtenido por el PLC presenta mayor diferencia con respecto al valor obtenido por VBS, presentándose diferencias entre 0.1312 y 0.0012 dB respectivamente.

## 6.2. Validación de Sistema en Maqueta de Ascensor

Como última etapa se procede a validar el sistema dentro de la maqueta de ascensor, donde se encuentra el motor que genera las vibraciones que se desea medir. La maqueta se configuró para que la cabina se moviera hacia arriba y hacia abajo con la dinámica de un ascensor de 12 metros de alto, lo cual corresponde aproximadamente a un edificio de 4 pisos. Los datos utilizados corresponden al viaje de la cabina hacia abajo. El sistema con el cual se comparan los resultados es un sistema de adquisición de datos y señales de la marca Nationa Instrument, el cual se componen de tarjetas de adquisición de señales introducidas en el Chasis NI CompactDAQ de 8 ranuras modelo NI cDAQ-9178 [78]. Este sistema permite la conexión de un sensor marca PCB Piezotronics de tres ejes, modelo 356A17[79]. Este sensor es del tipo cerámico con una sensibilidad nominal de 500 mV/g (10 veces más sensible al S01), que adquiere la señal de vibración a una frecuencia de muestreo de 51,2k Hz. Esta señal es enviada a la tarjeta NI 9234 [80] dentro del chasis. Este chasis entrega las señales captadas en la maqueta en u computador, siendo luego recuperado la señal descrita para comparar. La figura 4.16.b muestra cómo están colocados uno al otro los sensores.

Ambos sistemas de adquisición presentan una frecuencia de muestreo distinto. Para el sensor S01 la frecuencia de muestreo es 46.875 Hz, mientras que para el sensor PCB 51,2k Hz. La comprobación consiste en obtener la vibración obtenida por cada sensor y compararlas utilizando para ello MATLAB, obteniendo el valor en la banda de octava de 125 Hz.

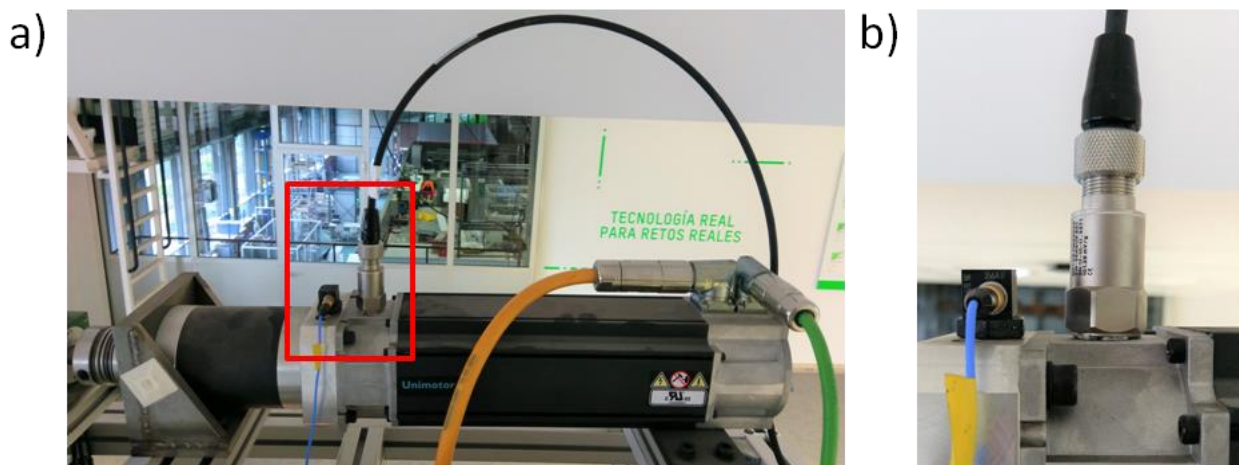


Figura 6.1: Disposición física de sensores sobre motor en maqueta ascensor. a) fotografía del motor junto con la caja de cambios montado en maqueta. Marcado en rojo los sensores instalados b) detalle de los sensores montados. A la izquierda sensor de la maqueta, a la derecha sensor S01. Fuente: elaboración propia.

La figura 6.2 muestra las señales obtenidas con cada sensor. La figura 6.2.a corresponde a la señal del S01 y la figura 6.2.b la del sensor PCB. Se observa a simple vista que la señal del sensor PCB presenta una mayor cantidad de ruido en comparación a la señal del S01.

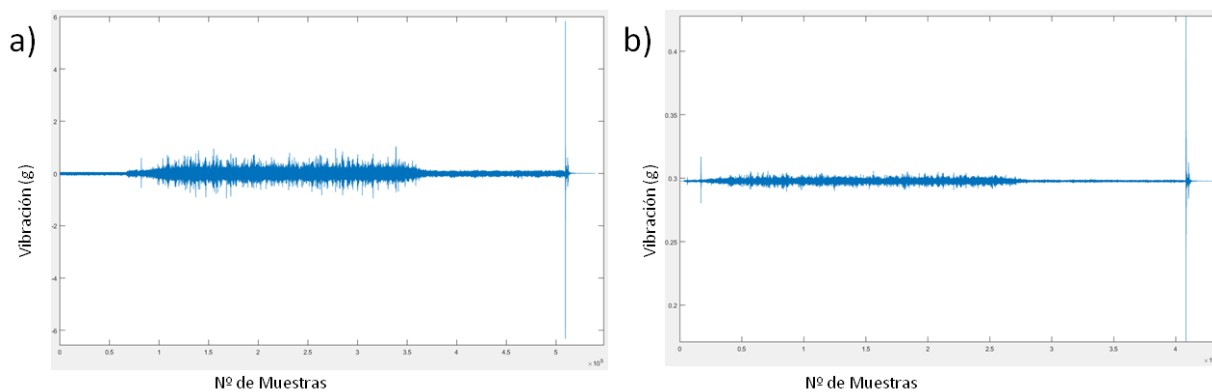


Figura 6.2: Señales de vibración de motor para viaje de bajada de cabina en maqueta de ascensor. a) señal obtenida con sensor PCB. b) señal obtenida con sensor S01. Fuente: elaboración propia.

Asimismo, se observa que la señal obtenida por el S01 se encuentra retrasada en comparación con la señal adquirida por el PCB.

Para comparar el algoritmo utilizado, se procesan estas señales para obtener su valor de ganancia en dB en la banda de 125 Hz utilizando MATLAB, por la cantidad de muestras. Para la señal del PCB, se obtiene una ganancia de 64,578. Para la señal del sensor S01, se obtiene 64,251. Con esto se valida los datos obtenidos por el sistema en comparación a lo obtenido con el sensor de la maqueta de ascensor.

## 7. CONCLUSIONES

### 7.1. Conclusiones generales

El objetivo general del proyecto fue buscar la forma de incorporar algoritmos de tratamiento de señal complejos para monitorización de la condición en dispositivos robustos como autómatas lógicos programables (PLC). Como caso práctico, se plantea la implementación de un cálculo de bandas de Octavas de una señal de vibración de un motor montado en una maqueta de ascensor, escrito en lenguaje MATLAB utilizando para ello las capacidades de un PLC y hardware asociado.

En este TFM se describe como se logra implementar con éxito el caso práctico descrito, combinando las capacidades de varios dispositivos de Siemens que tienen como denominador común sus precios competitivos (PLC, HMI, SM).

La solución implementada utiliza las capacidades de programación y cálculo del PLC y de la pantalla HMI. Las capacidades de esta pantalla para ejecutar el algoritmo fueron descubiertas en el transcurso del trabajo, haciendo posible el uso de VBS para ejecutar los algoritmos que inicialmente se pensaba incorporar solamente al PLC, pero sin presentar las mismas limitaciones de memoria.

Esta solución de monitorización de la condición ha sido validada utilizando la señal de vibración de un motor, el cual se encuentra montado en la maqueta de ascensor. Se compara la rapidez y precisión de los resultados frente a un sistema basado en PC y hardware de adquisición de NI.

Por tanto, la solución desarrollada permite la implementación de algoritmos complejos en dispositivos del tipo PLC, con las ventajas que estos aportan de robustez para su uso industrial y precio.

### 7.2. Conclusiones específicas

Para la realización del sistema se contó con equipos de precios competitivos dentro de las líneas de productos de automatización de Siemens. Con excepción de la pantalla HMI (que está en la gama de productos siguiente a la básica) tanto el PLC como el SM son los equipos más competitivos que dispone Siemens para automatización y CMS. Esto supuso limitaciones en las capacidades de hardware y software, a las que fue necesario buscar soluciones en el transcurso del TFM para cumplir con el objetivo general.

#### 7.2.1. CONCLUSIONES SOBRE PLC

En lo que respecta al PLC, las capacidades de hardware fueron las de un dispositivo de gama de entrada de Siemens. Estas capacidades influenciaron al desarrollo del

sistema, porque la memoria disponible repercute directamente en el número de datos a procesar, limitándose a 3.500. Este número de datos es insuficiente para realizar el análisis de Octavas de la señal de vibración. Sin embargo, se ha realizado la adaptación del programa en MATLAB considerando el límite de memoria del PLC escrito en SCL.

### **7.2.2. CONCLUSIONES SOBRE SM1281**

El equipo SM presenta interesantes características para su funcionamiento como CMS a un precio muy atractivo para el mercado actual. Sin embargo, la dificultad en su programación obliga contactar al servicio técnico de Siemens, donde la documentación no se encuentra completa y actualizada. La puesta en marcha de este dispositivo hizo uso de un gran número de horas, incluyendo el contactar con servicio técnico de Siemens y la devolución del producto. Uno de los motivos de esta dificultad fue la ausencia de ejemplos de programación actualizados. Asimismo, la conectividad del dispositivo por red Ethernet presenta protocolos con más de 20 años de antigüedad, reemplazados hoy en día por otros más eficientes. Como consecuencia de todo ello, se tuvieron que desarrollar múltiples soluciones para hacer funcionar el sistema.

### **7.2.3. CONCLUSIONES SOBRE HMI P700 COMFORT**

El uso del dispositivo HMI estaba proyectado al inicio del trabajo para mostrar datos numéricos y gráficos. A medida que avanzó el proyecto, se encuentra la posibilidad de programar utilizando el lenguaje VBS, dada las limitaciones del PLC (límites de memoria y conectividad). A pesar de que el software de Siemens evita el acceso a varios recursos y librerías en VBS, es posible aprovechar las capacidades presentes en el HMI para lograr descargar el archivo, obtener la información contenida y luego procesar esta información. Esto permite utilizar el dispositivo para tareas que no estaban proyectadas en un comienzo del proyecto, pero que han sido claves para concluir satisfactoriamente este trabajo.

### **7.2.4. CONCLUSIONES SOBRE TIA PORTAL**

El mayor reto para el uso de TIA Portal fue coordinar las distintas versiones de software; firmware, hardware, librerías y catálogo de dispositivos. Esto supuso limitaciones dentro de TIA Portal, por ser la versión “Basic” de STEP7 o la versión “Comfort” de WinCC. Esto repercute en las opciones para el desarrollo de una solución al problema del cálculo de Octavas. La versión de STEP7 permite la programación en SCL sin mayores inconvenientes, pero resulta más fácil escribir el código utilizando otros editores de texto, como Notepad++. Para WinCC, la versión

del software utilizado es robusta y salvo detalles en el momento de programar en VBS, resulta bastante intuitivo su uso.

#### ***7.2.5. LÍNEAS FUTURAS***

Una de las principales líneas futuras sería el reemplazo del PLC utilizado por uno de gama superior. Esto permitiría tener más memoria y evitaría el límite de datos posibles para analizar. También sería posible utilizar las opciones disponibles dentro de TIA Portal con el software STEP7 por la versión de software.

El uso del SM podría facilitarse con la opción de disminuir la frecuencia de muestreo de los datos *raw*, junto con una mejora en la conectividad de este dispositivo en redes Ethernet con protocolos mas actualizados. El uso de módulos de gama alta que reemplacen a este dispositivo permitiría sortear estos problemas.

En el comienzo del proyecto y dada la cantidad de información disponible sobre los dispositivos, no se sabía que algoritmo sería adaptado para su funcionamiento en el PLC. Las dificultades ya explicadas hizo que el alcance del caso práctico se focalizara en el desarrollo del cálculo de bandas de Octava. La solución desarrollada abre camino a la incorporación de los algoritmos actualmente existentes.

El reemplazo del dispositivo HMI por uno de gama alta permitiría evitar la asistencia de una persona para que funcione el sistema, automatizándolo completamente. También sería posible el uso del lenguaje C, habilitando mayores funcionalidades para automatizar el proceso de obtención de datos.



## 8. BIBLIOGRAFÍA

- [1] J. D. N. Cheeke, *Fundamentals and applications of ultrasonic waves*. CRC Press, 2012.
- [2] P. Etxaniz, “monitorización online,” pp. 72–73.
- [3] Fluke, “How infrared cameras work.” [Online]. Available: <http://en-us.fluke.com/training/training-library/measurements/how-infrared-cameras-work.html>. [Accessed: 22-Jun-2018].
- [4] AZIMA DLI, “¿Que es Vibración?” [Online]. Available: <http://azimadli.com/vibman-spanish/queesvibracin.htm>. [Accessed: 22-Jun-2018].
- [5] W. B. A. . M. C. Z. . B. A. SPARC (Organization) and Universidad Tecnológica de Pereira., *Scientia et technica.*, vol. XVI, no. 45. Universidad Tecnológica de Pereira, 1995.
- [6] G. Arce, M. D. Campbell, M. Fisher, and R. Turner, “Global Standards for Rotating Machinery: Navigating Worldwide Industry Requirements for Electric Motors,” *IEEE Ind. Appl. Mag.*, vol. 23, no. 1, pp. 58–69, Jan. 2017.
- [7] Universidad de Vigo, “Máquinas rotatorias.” [Online]. Available: [http://quintans.webs.uvigo.es/recursos/Web\\_electromagnetismo/dispositivos\\_maquinasrotatorias.htm](http://quintans.webs.uvigo.es/recursos/Web_electromagnetismo/dispositivos_maquinasrotatorias.htm). [Accessed: 22-Jun-2018].
- [8] I.-S. Jang *et al.*, “Method for Analyzing Vibrations Due to Electromagnetic Force in Electric Motors,” *IEEE Trans. Magn.*, vol. 50, no. 2, pp. 297–300, Feb. 2014.
- [9] Hyeon-Jae Shin, Jang-Young Choi, Han-Wook Cho, and Seok-Myeong Jang, “Analysis on electromagnetic vibration source permanent magnet synchronous motor for compressor of electric vehicles,” in *2012 IEEE Vehicle Power and Propulsion Conference*, 2012, pp. 200–203.
- [10] S. Gopinath, “Study on electric motor mass unbalance based on vibration monitoring analysis technique,” in *2010 International Conference on Mechanical and Electrical Technology*, 2010, pp. 539–542.
- [11] H.-S. Ko and K.-J. Kim, “Characterization of Noise and Vibration Sources in Interior Permanent-Magnet Brushless DC Motors,” *IEEE Trans. Magn.*, vol. 40, no. 6, pp. 3482–3489, Nov. 2004.
- [12] M. Xu and R. D. Marangoni, “Vibration Analysis Of A Motor-Flexible Coupling-Rotor System Subject To Misalignment And Unbalance, Part I: Theoretical Model And Analysis,” *J. Sound Vib.*, vol. 176, no. 5, pp. 663–679, Oct. 1994.
- [13] M. J. (Michael J. Neale and Society of Automotive Engineers., *Bearings*. Society of Automotive Engineers, 1993.
- [14] T. A. Harris and M. N. Kotzalas, *Rolling bearing analysis*. CRC/Taylor & Francis, 2007.
- [15] F. A. G. Sales and E. Gmbh, “Averías de los rodamientos Reconocimiento de daños e inspección,” vol. WL 82 102/, pp. 1–75, 2002.
- [16] J. F. Suri, “Condition Monitoring System,” pp. 5–6.

- [17] The Association of German Engineers, “VDI 3832.” [Online]. Available: [http://www.vdi.eu/guidelines/vdi\\_3832-koerperschallmessungen\\_zur\\_zustandsbeurteilung\\_von\\_waelzlagern\\_in\\_maschinen\\_und\\_anlagen/](http://www.vdi.eu/guidelines/vdi_3832-koerperschallmessungen_zur_zustandsbeurteilung_von_waelzlagern_in_maschinen_und_anlagen/). [Accessed: 22-Jun-2018].
- [18] Dr. Apple L. S. Chan, “Spectrum and Octave Band.” [Online]. Available: <http://personal.cityu.edu.hk/~bsapplec/spectrum.htm>. [Accessed: 22-Jun-2018].
- [19] Castle Group, “What are 1/1 and 1/3 Octave Bands and why are they used?” [Online]. Available: <https://www.castlegroup.co.uk/guidance/octave-bands/>. [Accessed: 22-Jun-2018].
- [20] B. K. N. Rao, *Handbook of condition monitoring*. Elsevier Advanced Technology, 1996.
- [21] Wikipedia, “Mantenimiento predictivo - Wikipedia, la enciclopedia libre.” [Online]. Available: [https://es.wikipedia.org/wiki/Mantenimiento\\_predictivo](https://es.wikipedia.org/wiki/Mantenimiento_predictivo). [Accessed: 22-Jun-2018].
- [22] H. Chelladurai, V. K. Jain, and N. S. Vyas, “Development of a cutting tool condition monitoring system for high speed turning operation by vibration and strain analysis,” *Int. J. Adv. Manuf. Technol.*, vol. 37, no. 5–6, pp. 471–485, May 2008.
- [23] M. A. Salvador Carlos, “Fundamentos y Aplicaciones de los Sensores Virtuales,” 2008.
- [24] Parexel Informatics, “Understanding data-driven monitoring,” 2014. [Online]. Available: [https://www.parexel.com/files/7614/1624/0879/PRXLBH\\_DDM\\_Infographic\\_Nov-14.pdf](https://www.parexel.com/files/7614/1624/0879/PRXLBH_DDM_Infographic_Nov-14.pdf). [Accessed: 22-Jun-2018].
- [25] NASA, “Hybrid Modeling Improves Health and Performance Monitoring.” [Online]. Available: [https://spinoff.nasa.gov/Spinoff2007/ct\\_3.html](https://spinoff.nasa.gov/Spinoff2007/ct_3.html). [Accessed: 22-Jun-2018].
- [26] D. Azagra and T. Hay, “A case for physical models,” *Struct. Eng.*, vol. 90, no. 9, pp. 14–20, 2012.
- [27] Allen-Bradley, “Dynamix 1444.” [Online]. Available: <https://ab.rockwellautomation.com/Condition-Monitoring>. [Accessed: 22-Jun-2018].
- [28] Allen-Bradley, “Emonitor Condition Monitoring Software.” [Online]. Available: <https://ab.rockwellautomation.com/Condition-Monitoring/9309-Emonitor-Software>. [Accessed: 22-Jun-2018].
- [29] infoPLC, “Condition Monitoring con PC-based Control de Beckhoff (En) - infoPLC.” [Online]. Available: <http://www.infopl.net/descargas/18-beckhoff/2119-condition-monitoring-pc-based-control-beckhoff>. [Accessed: 22-Jun-2018].
- [30] Beckhoff, “Practicas y ejemplos de programación con TwinCAT - infoPLC.” [Online]. Available: <http://www.infopl.net/descargas/18-beckhoff/1914-practicas-y-ejemplos-de-programación-con-twincat>. [Accessed: 26-Jun-2018].
- [31] Siemens AG, “SIPLUS CMS - Products for specific requirements - Siemens Global Website.” [Online]. Available:

- <https://www.siemens.com/global/en/home/products/automation/products-for-specific-requirements/siplus-cms.html>. [Accessed: 22-Jun-2018].
- [32] Siemens AG, “MindSphere – Siemens Cloud for Industry - Digitalization in Machine Building - Industry - Home - Siemens Global Website.” [Online]. Available: <https://www.siemens.com/customer-magazine/en/home/industry/digitalization-in-machine-building/mindsphere-siemens-cloud-for-industry.html>. [Accessed: 22-Jun-2018].
- [33] MMF, “IEPE Standard.” [Online]. Available: [https://www.mmf.de/iepe\\_standard.htm](https://www.mmf.de/iepe_standard.htm). [Accessed: 22-Jun-2018].
- [34] Siemens AG, “Manual S7-1200,” 2016.
- [35] Siemens AG, “SIMATIC S7 - 1200 - El Futuro de la Industria - Siemens.”
- [36] C. Panels, *SIMATIC HMI*. 2012.
- [37] EMAC Inc, *Windows CE overview*. .
- [38] *OPC Foundation and MTConnect Institute Announce a Memorandum of Understanding*. OPC Foundation, 2010.
- [39] Electronics tutorials, “Transistor Biasing and the Biasing of Transistors.” [Online]. Available: <https://www.electronics-tutorials.ws/amplifier/transistor-biasing.html>. [Accessed: 22-Jun-2018].
- [40] Control Techniques, “Unimotor fm | Control Techniques.” [Online]. Available: <http://acim.nidec.com/es-es/drives/control-techniques/products/servo-drives/ac-servo-motors/unimotor-fm>. [Accessed: 22-Jun-2018].
- [41] Siemens AG, “Siemens TIA Portal - El Futuro de la Industria - Siemens.”
- [42] Siemens AG, “How do you convert STEP 7 Professional / Basic projects in TIA Portal.” [Online]. Available: <https://support.industry.siemens.com/cs/document/109476392/how-do-you-convert-step-7-professional-basic-projects-in-tia-portal-which-you-have-created-with-earlier-versions-of-tia-portal?dti=0&lc=en-BR>. [Accessed: 26-Jun-2018].
- [43] Siemens AG, “How should you install the TIA Portal V14 SP1 when TIA Portal V14 is already inst... - ID: 109747881 - Industry Support Siemens.” [Online]. Available: <https://support.industry.siemens.com/cs/document/109747881/how-should-you-install-the-tia-portal-v14-sp1-when-tia-portal-v14-is-already-installed-on-your-computer?dti=0&lc=en-CR>. [Accessed: 26-Jun-2018].
- [44] Siemens AG, “How do you update the hardware catalog in the TIA Portal? - ID: 54163658 - Industry Support Siemens.” [Online]. Available: <https://support.industry.siemens.com/cs/document/54163658/how-do-you-update-the-hardware-catalog-in-the-tia-portal?dti=0&lc=en-AF>. [Accessed: 26-Jun-2018].
- [45] Siemens AG, “Libraries in the TIA Portal.” [Online]. Available: <https://support.industry.siemens.com/cs/document/109738702/libraries-in-the-tia-portal?dti=0&lc=en-BO>. [Accessed: 26-Jun-2018].
- [46] Siemens AG, “SIMATIC STEP 7 Professional (TIA Portal) - TIA Portal - Siemens.”

- [47] IEC, “IEC 61131-3:2013.” [Online]. Available: <https://webstore.iec.ch/publication/4552>. [Accessed: 26-Jun-2018].
- [48] PLCS.net, “Your Personal PLC Tutor Site - Basic Instructions.” [Online]. Available: <http://www.plcs.net/chapters/basic9.htm>. [Accessed: 26-Jun-2018].
- [49] David Berno, “Efficient SCL Development in TIA Portal V14 | DMC, Inc.” [Online]. Available: <https://www.dmcinfo.com/latest-thinking/blog/id/9540/efficient-scl-development-in-tia-portal-v14>. [Accessed: 26-Jun-2018].
- [50] Siemens AG, “SIMATIC Programming with STEP 7,” 2010.
- [51] Siemens AG, “SIMATIC WinCC - HMI Software.” [Online]. Available: <https://w3.siemens.com/mcms/human-machine-interface/en/visualization-software/scada/simatic-wincc/Pages/default.aspx>. [Accessed: 26-Jun-2018].
- [52] Microsoft, “VBScript.” [Online]. Available: <https://msdn.microsoft.com/en-us/library/t0aew7h6.aspx>. [Accessed: 28-Jun-2018].
- [53] Chris Tilley, “The History of Microsoft Windows CE - Index & Humble Beginnings - HPC Factor.” [Online]. Available: <https://www.hpcfactor.com/support/windowsce/>. [Accessed: 28-Jun-2018].
- [54] L. P. Petrusha Ron, Childs Matt, “VBScript in a Nutshell, 2nd Edition [Book].” [Online]. Available: <https://www.safaribooksonline.com/library/view/vbscript-in-a/0596004885/ch01.html>. [Accessed: 28-Jun-2018].
- [55] V. J. Milener Gene, Guyer Craig, Rabeler Carl, “Stream Object (ADO) | Microsoft Docs.” [Online]. Available: <https://docs.microsoft.com/en-us/sql/ado/reference/ado-api/stream-object-ado?view=sql-server-2017>. [Accessed: 28-Jun-2018].
- [56] Siemens AG, “SIMATIC HMI WinCC V7.4 SP1 WinCC: Scripting (VBS, ANSI-C, VBA),” pp. 1–2672, 2017.
- [57] Siemens AG, “WinCC WinCC Advanced V14 SP1 - Referencia para programación,” pp. 1–1366, 2017.
- [58] Siemens AG, “Forum - Industry Support - Siemens.” [Online]. Available: <https://support.industry.siemens.com/tf/ww/en/threads/252/?page=0&pageSize=10>. [Accessed: 02-Jul-2018].
- [59] PLCS.net, “Your Personal PLC Tutor - Learn PLC Programming Here. FREE.” [Online]. Available: <http://www.plcs.net/>. [Accessed: 02-Jul-2018].
- [60] Siemens AG, “Firmware Update SM 1281.” [Online]. Available: <https://support.industry.siemens.com/cs/document/109755629/firmware-update-siplus-cms1200-sm-1281?dti=0&lc=en-WW>. [Accessed: 02-Jul-2018].
- [61] Siemens, “HSP SM1281.” [Online]. Available: [https://support.industry.siemens.com/cs/document/72341852/support-packages-for-the-hardware-catalog-in-the-tia-portal-\(hsp\)?dti=0&lc=en-WW](https://support.industry.siemens.com/cs/document/72341852/support-packages-for-the-hardware-catalog-in-the-tia-portal-(hsp)?dti=0&lc=en-WW). [Accessed: 26-Jun-2018].
- [62] Siemens, “Library SM 1281.” [Online]. Available: <https://support.industry.siemens.com/cs/document/109482016/library-siplus-cms1200-sm-1281?dti=0&lc=en-WW>. [Accessed: 26-Jun-2018].

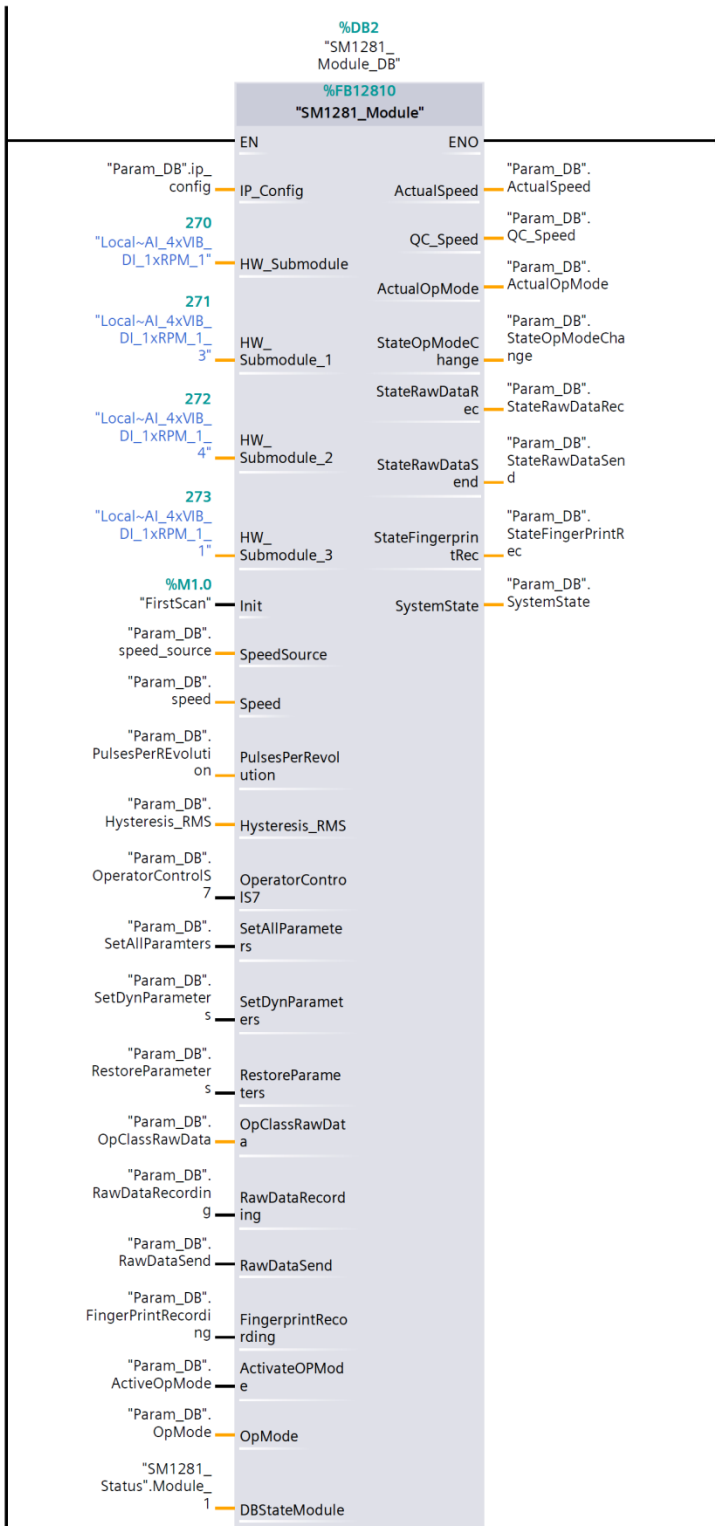
- [63] Siemens, “Firmware Update SIPLUS CMS1200 SM 1281 - ID: 109755629 - Industry Support Siemens.” [Online]. Available: <https://support.industry.siemens.com/cs/document/109755629/firmware-update-siplus-cms1200-sm-1281?dti=0&lc=en-WW>. [Accessed: 26-Jun-2018].
- [64] IETF Trust, “HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV),” *“Internet Official Protocol Standards” (STD 1)*, 2007. .
- [65] J. Postel and J. Reynolds, “File Transfer Protocol (FTP).”
- [66] Siemens AG, “FTP Client Communication with S7-1200/1500 - ID: 81367009 - Industry Support Siemens.” [Online]. Available: <https://support.industry.siemens.com/cs/document/81367009/ftp-client-communication-with-s7-1200-1500?dti=0&lc=en-WW>. [Accessed: 26-Jun-2018].
- [67] S. S-, S. S-, and F. T. P. Server, “FTP Data Exchange between FTP Server and SIMATIC S7-1200 / S7-1500.”
- [68] Microsoft, “WinInet and the OSI Model (Windows CE 3.0).” [Online]. Available: <https://msdn.microsoft.com/en-us/library/ms903396.aspx>. [Accessed: 04-Jul-2018].
- [69] Microsoft, “Extensible Wave-Format Descriptors | Microsoft Docs,” 2017. [Online]. Available: <https://docs.microsoft.com/en-us/windows-hardware/drivers/audio/extensible-wave-format-descriptors>. [Accessed: 26-Jun-2018].
- [70] Prof. Peter Kabal, “Wave File Specifications,” 2017. [Online]. Available: <http://www-mmsp.ece.mcgill.ca/Documents/AudioFormats/WAVE/WAVE.html>. [Accessed: 26-Jun-2018].
- [71] X-Ways Software Technology AG, “WinHex: Hex Editor & Disk Editor, Computer Forensics & Data Recovery Software.” [Online]. Available: <https://www.x-ways.net/winhex/>. [Accessed: 04-Jul-2018].
- [72] jenslorenz, “Notepad++ Plugins download | SourceForge.net.” [Online]. Available: <https://sourceforge.net/projects/npp-plugins/?source=navbar>. [Accessed: 04-Jul-2018].
- [73] Siemens AG, “Ordering Data Overview CMS X-Tools.” [Online]. Available: <https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/9AE4160-1AD00>. [Accessed: 28-Jun-2018].
- [74] Tobias Mühlbauer, “OSCAT - Neuigkeiten.” [Online]. Available: <http://www.oscat.de/>. [Accessed: 26-Jun-2018].
- [75] MATLAB, “Simulink PLC Coder - MATLAB & Simulink.” [Online]. Available: <https://es.mathworks.com/products/sl-plc-coder.html>. [Accessed: 26-Jun-2018].
- [76] MATLAB, “Functions and Objects Supported for C/C++ Code Generation - Category List - MATLAB & Simulink - MathWorks España.” [Online]. Available: <https://es.mathworks.com/help/simulink/ug/functions-supported-for-code-generation-categorical-list.html>. [Accessed: 04-Jul-2018].
- [77] PCB Piezotronics, “PCB Model 394C06.” [Online]. Available: <http://www.pcb.com/Products/model/394C06>. [Accessed: 22-Jun-2018].

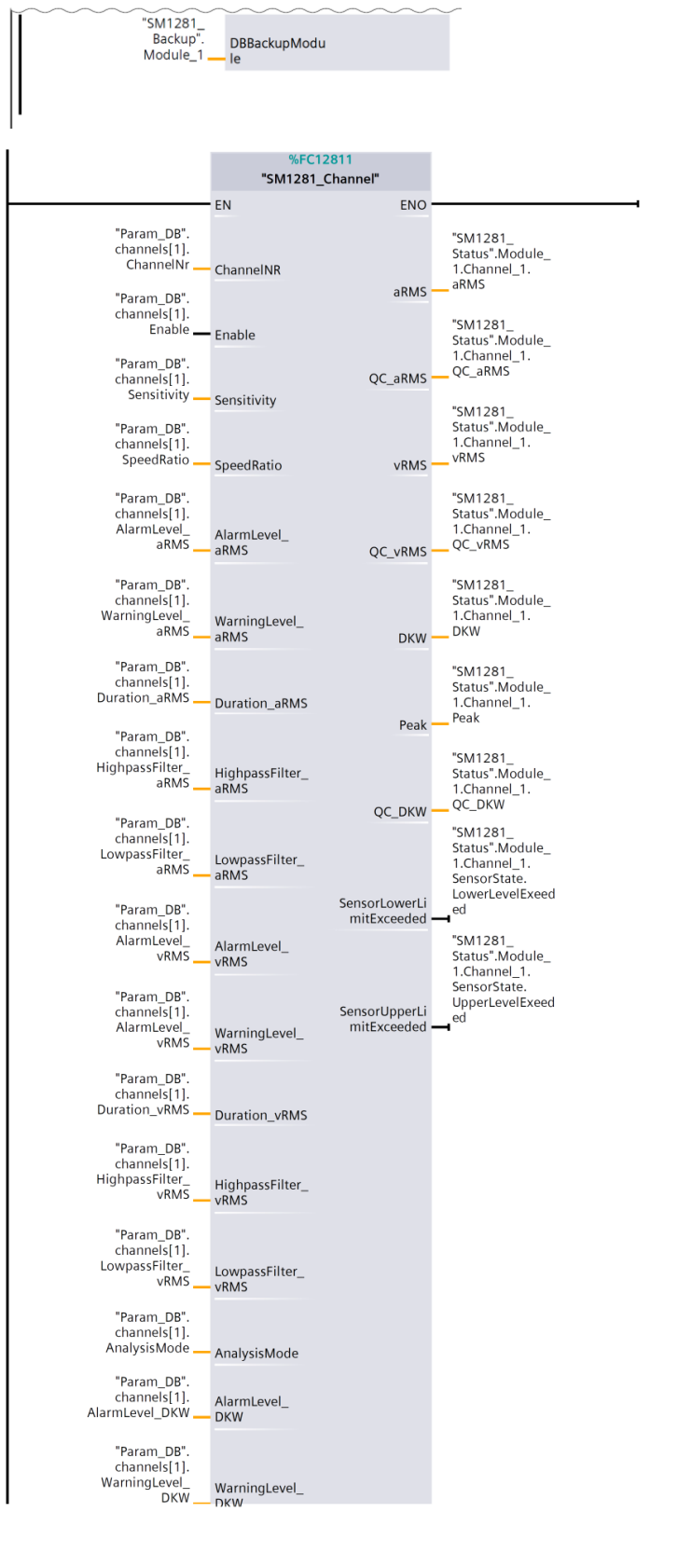
- [78] NI, “cDAQ-9178 - National Instruments.” [Online]. Available: <http://www.ni.com/es-es/support/model.cdaq-9178.html>. [Accessed: 06-Jul-2018].
- [79] P. C. B. Piezotronics, “Model 352C04 General purpose , ceramic shear ICP ® accel ., 10 mV / g , 0 . 5 to 10k Hz , 10-32 Installation and Operating Manual,” 2007.
- [80] NI, “NI 9234 Datasheet.” [Online]. Available: <http://www.ni.com/nisearch/app/main/p/ap/tech/lang/es/pg/1/sn/ssnav:spc/aq/pmdmid:122186/>. [Accessed: 06-Jul-2018].

## 9. ANEXO

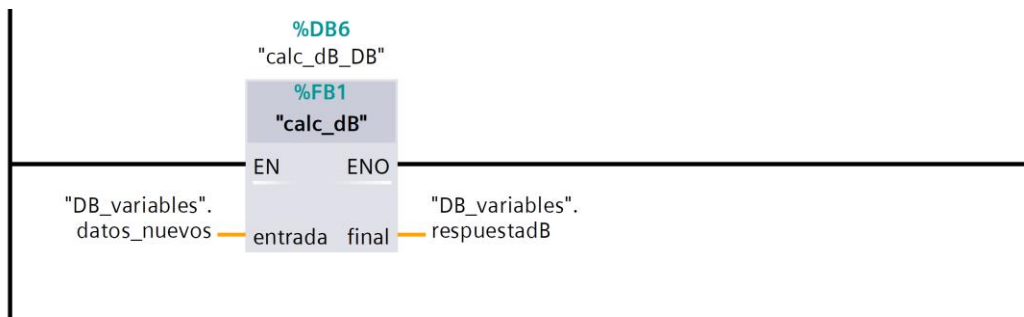
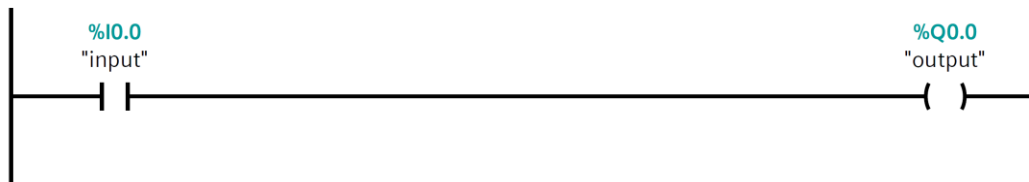
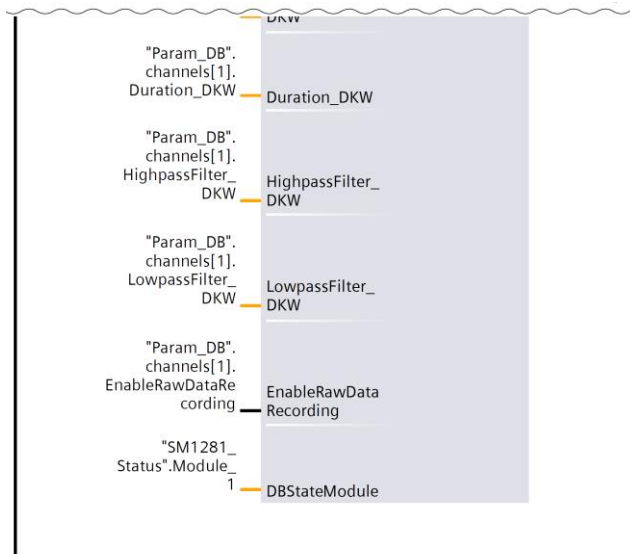
### Anexo I: Programas desarrollados

#### FUNCIÓN PLC: MAIN









***FUNCIÓN PLC: CALC\_DB***

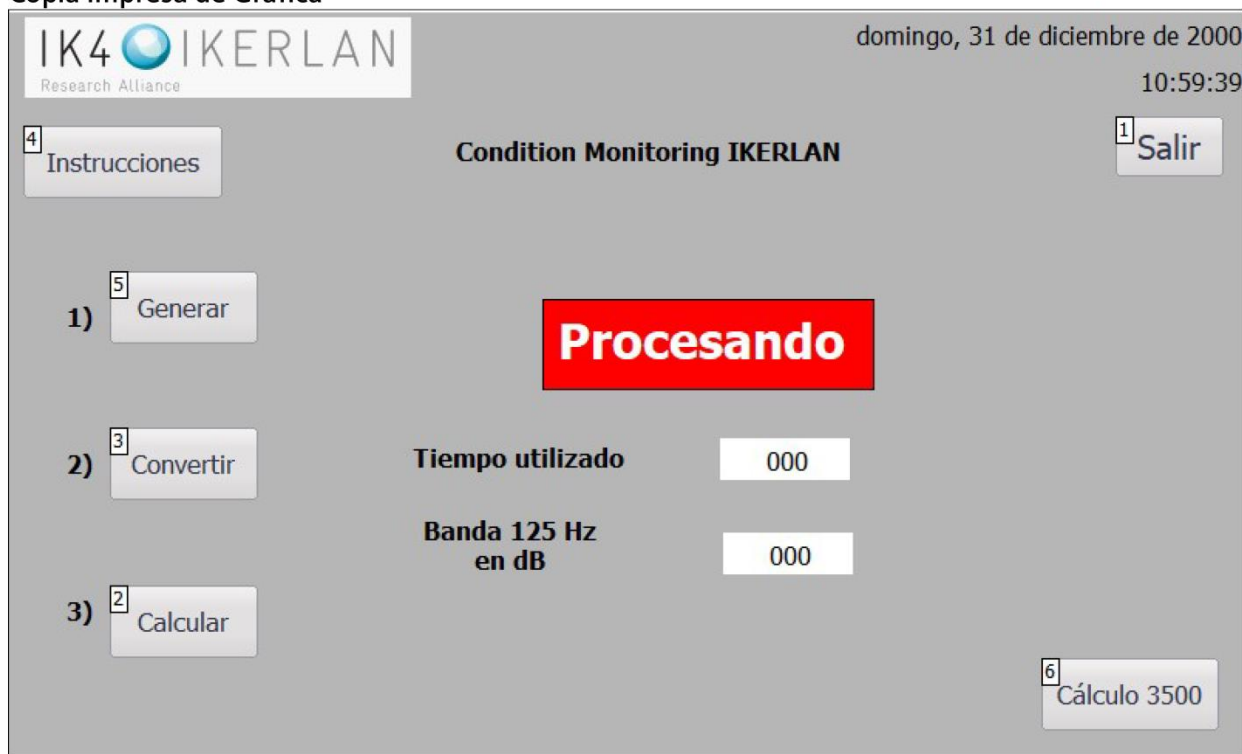
```

0001 (* MATLAB Function 'MATLAB Function': '<S1>:1' *)
0002 (* '<S1>:1:3' a=[1 -5.540193938690162 12.928277688192221 -16.261736159518428
11.628149435302925 -4.482221650813031 0.727832871740883]; *)
0003 (* '<S1>:1:4' b=[4.287301831741014e-04 0 -0.001286190549522 0
0.001286190549522 0 -4.287301831741014e-04]; *)
0004 (* '<S1>:1:7' final=filter(b,a,entrada); *)
0005 (* MATLAB Function: '<Root>/MATLAB Function' incorporates:
0006 * Inport: '<Root>/entrada' *)
0007 FOR #k := 0 TO 3499 DO
0008 #rtb_final[#k] := 0.0;
0009 END_FOR;
0010 FOR #k := 0 TO 3499 DO
0011 IF (3499 - #k) < 7 THEN
0012 #naxpy := 3500 - #k;
0013 ELSE
0014 #naxpy := 7;
0015 END_IF;
0016 #j := 0;
0017 WHILE (#j + 1) <= #naxpy DO
0018 #rtb_final[#k + #j] := #rtb_final[#k + #j] + (#entrada[#k] * #b[#j]);
0019 #j := #j + 1;
0020 END_WHILE;
0021 IF (3499 - #k) < 6 THEN
0022 #naxpy := 3499 - #k;
0023 ELSE
0024 #naxpy := 6;
0025 END_IF;
0026 #as := - #rtb_final[#k];
0027 #j := 1;
0028 WHILE #j <= #naxpy DO
0029 #rtb_final[#k + #j] := #rtb_final[#k + #j] + (#as * #b_c[#j]);
0030 #j := #j + 1;
0031 END_WHILE;
0032 END_FOR;
0033 (* End of MATLAB Function: '<Root>/MATLAB Function' *)
0034 (* Outport: '<Root>/final' *)
0035 (*FOR k := 0 TO 3499 DO
0036 final[k] := rtb_final[k];
0037 END_FOR;*)
0038 (* End of Outport: '<Root>/final' *)
0039 (*programacion del dB*)
0040 #X := 0.0;
0041 FOR #indice_i := 0 TO 3499 DO
0042 #X := #X + SQR(#rtb_final[#indice_i]);
0043 END_FOR;
0044
0045 #final := MAX(IN1 := 0.0, IN2 := (LN(SQRT(#X / 3499.0) / 1.0E-6) / LN(10)) * 20.0);

```

**INTERFAZ HMI: GRAFICA**

Copia impresa de Grafica



Nombre	Grafica	Color de fondo	182; 182; 182
Color Cuadrícula	0; 0; 0	Número	1
Plantilla	Plantilla_1	Tooltip	

**Dinamizaciones\Evento**

Nombre de evento	Creada
------------------	--------

**Lista de funciones\DefinirVariable**

Variable	Número_imagen_variable	Valor	1
----------	------------------------	-------	---

**HmiScreenItemData\_2**

Tipo	Campo de texto	Nombre	HmiScreenItemData_2
Posición X	245	Posición Y	17
Ancho	337	Altura	25
Nivel	0 - Nivel_0	Fuente	Tahoma, 16px, style=Bold
Texto	Condition Monitoring IKERLAN		

**Salir de runtime\_1**

Tipo	Botón	Nombre	Salir de runtime_1
Posición X	716	Posición Y	7
Ancho	70	Altura	39
Modo	Texto	Texto OFF	Salir
Texto ON	Salir de runtime		

**Dinamizaciones\Evento**

Nombre de evento	Soltar
------------------	--------

**Lista de funciones\PararRuntime**

Modo	Runtime
------	---------

**Salir de runtime\_3**

Tipo	Botón	Nombre	Salir de runtime_3
Posición X	65	Posición Y	310
Ancho	96	Altura	47
Modo	Texto	Texto OFF	Calcular
Texto ON	Salir de runtime		

**Dinamizaciones\Evento**

Nombre de evento	Hacer clic
------------------	------------

**Lista de funciones\calculardB\_3**

**Rectangle\_1**

Tipo	Rectángulo	Nombre	Rectangle_1
Posición X	345	Posición Y	125
Ancho	215	Altura	59
Nivel	0 - Nivel_0	Color de fondo	255; 0; 0
Color Borde	0; 0; 0		

**Dinamizaciones\Visibilidad**

Variable - Ciclo	Espera -	Tipo de datos	Bit
Define el bit que debe monitorizarse.	0	Define la visibilidad en función de los valores de proceso seleccionados.	Visible

**Text field\_3**

Tipo	Campo de texto	Nombre	Text field_3
Posición X	353	Posición Y	130
Ancho	200	Altura	43
Nivel	0 - Nivel_0	Fuente	Tahoma, 32px, style=Bold
Texto	Procesando		

**Dinamizaciones\Visibilidad**

Variable - Ciclo	Espera -	Tipo de datos	Bit
Define el bit que debe monitorizarse.	0	Define la visibilidad en función de los valores de proceso seleccionados.	Visible

**Salir de runtime\_4**

Tipo	Botón	Nombre	Salir de runtime_4
Posición X	65	Posición Y	208
Ancho	96	Altura	47
Modo	Texto	Texto OFF	Convertir
Texto ON	Salir de runtime		

**Dinamizaciones\Evento**

Nombre de evento	Hacer clic
------------------	------------

**Lista de funciones\convertir\_2**

**I/O field\_1**

Tipo	Campo ES	Nombre	I/O field_1
Posición X	460	Posición Y	215

Ancho	84	Altura	27
Nivel	0 - Nivel_0	Modo	Salida
Fuente	Tahoma, 17px		
<b>Dinamizaciones\Conexión de variable</b>			
Nombre de la propiedad	Valor de proceso	Variable	TiempoTranscurrido

#### HmiScreenItemData\_1

Tipo	Campo de texto	Nombre	HmiScreenItemData_1
Posición X	254	Posición Y	206
Ancho	151	Altura	44
Nivel	0 - Nivel_0	Fuente	Tahoma, 16px, style=Bold
Texto	Tiempo utilizado		

#### Salir de runtime\_2

Tipo	Botón	Nombre	Salir de runtime_2
Posición X	9	Posición Y	13
Ancho	129	Altura	47
Modo	Texto	Texto OFF	Instrucciones
Texto ON	Salir de runtime		

#### Dinamizaciones\Evento

Nombre de evento	Hacer clic
------------------	------------

#### Lista de funciones\MostrarImagenEmergente

Nombre de la imagen	instrucciones	Coordenada X	50
Coordenada Y	0	Modo de visualización	Conmutar
Animación	Off	Velocidad de la animación	Medio

#### Salir de runtime\_5

Tipo	Botón	Nombre	Salir de runtime_5
Posición X	65	Posición Y	107
Ancho	96	Altura	47
Modo	Texto	Texto OFF	Generar
Texto ON	Salir de runtime		

#### Dinamizaciones\Evento

Nombre de evento	Hacer clic
------------------	------------

#### Lista de funciones\generar\_1

#### I/O field\_2

Tipo	Campo ES	Nombre	I/O field_2
Posición X	462	Posición Y	276
Ancho	84	Altura	27
Nivel	0 - Nivel_0	Modo	Salida
Fuente	Tahoma, 17px		
<b>Dinamizaciones\Conexión de variable</b>			
Nombre de la propiedad	Valor de proceso	Variable	TiempoTranscurrido

#### HmiScreenItemData\_3

Tipo	Campo de texto	Nombre	HmiScreenItemData_3
Posición X	245	Posición Y	262

Ancho	158	Altura	44
Nivel	0 - Nivel_0	Fuente	Tahoma, 16px, style=Bold
Texto	Banda 125 Hz en dB		

#### HmiScreenItemData\_4

Tipo	Campo de texto	Nombre	HmiScreenItemData_4
Posición X	24	Posición Y	104
Ancho	43	Altura	256
Nivel	0 - Nivel_0	Fuente	Tahoma, 16px, style=Bold
Texto	1) 2) 3)		

#### Salir de runtime\_6

Tipo	Botón	Nombre	Salir de runtime_6
Posición X	668	Posición Y	357
Ancho	115	Altura	47
Modo	Texto	Texto OFF	Cálculo 3500
Texto ON	Salir de runtime		

#### Dinamizaciones\Evento

Nombre de evento	Hacer clic
------------------	------------

#### Lista de funciones\ActivarImagen

Nombre de imagen	Grafica2	Número de objeto	0
------------------	----------	------------------	---

#### Copia impresa de Área permanente

	domingo, 31 de diciembre de 2000
	10:59:39

Nombre	Área permanente	Color de fondo	182; 182; 182
Color Cuadrícula	0; 0; 0	Altura	60
Nivel activo	0		

#### HmiScreenItemData

Tipo	Campo de fecha y hora	Nombre	HmiScreenItemData
Posición X	695	Posición Y	30
Ancho	105	Altura	30
Modo	Salida	Fuente	Tahoma, 16px

#### HmiScreenItemData\_1

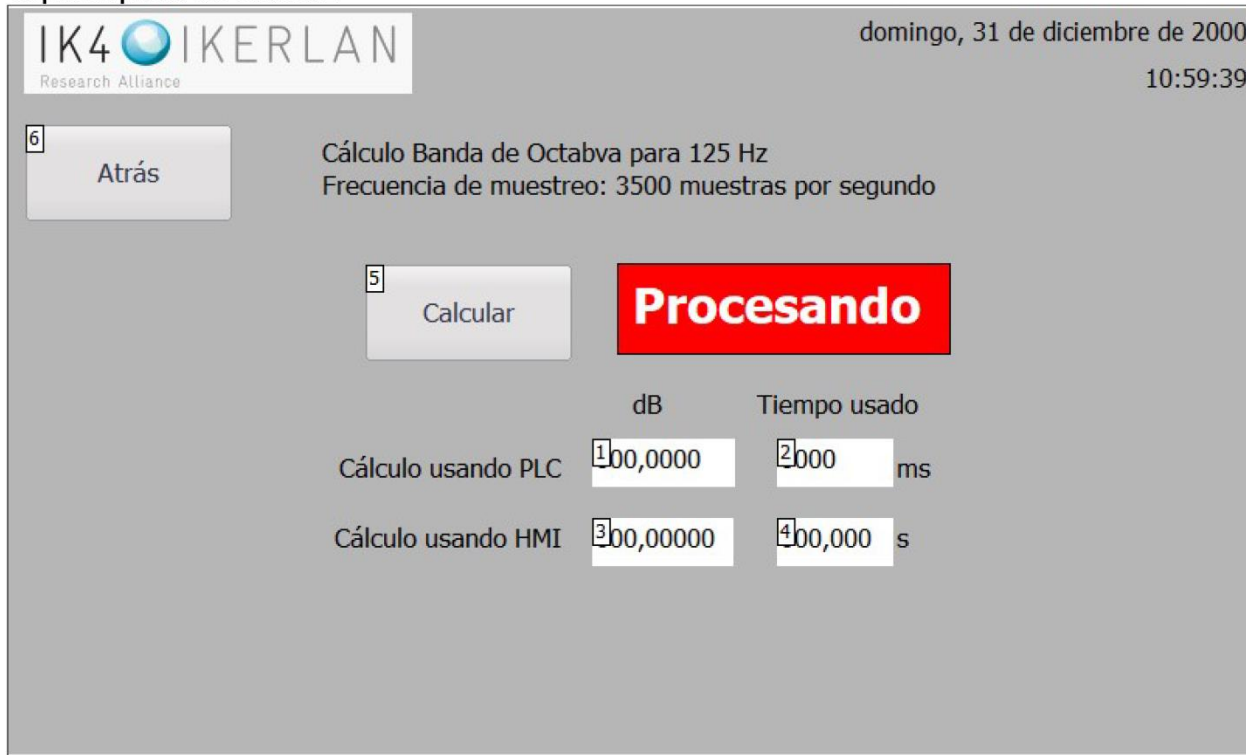
Tipo	Campo de fecha y hora	Nombre	HmiScreenItemData_1
Posición X	530	Posición Y	0
Ancho	270	Altura	30
Modo	Salida	Fuente	Tahoma, 16px

#### Graphic view\_1

Tipo	Visor de gráficos	Nombre	Graphic view_1
Posición X	10	Posición Y	3
Ancho	250	Altura	52
Nivel	0 - Nivel_0	Gráfico	ikerlan_1
Adaptar gráfico a tamaño objeto	Extender imagen		

***INTERFAZ HMI: GRAFICA2***

Copia impresa de Grafica2



Nombre	Grafica2	Color de fondo	182; 182; 182
Color Cuadrícula	0; 0; 0	Número	2
Plantilla		Tooltip	

**Campo ES\_1**

Tipo	Campo ES	Nombre	Campo ES_1
Posición X	376	Posición Y	217
Ancho	92	Altura	31
Nivel	0 - Nivel_0	Modo	Entrada/salida
Fuente	Tahoma, 17px		

**Dinamizaciones\Conexión de variable**

Nombre de la propiedad	Valor de proceso	Variable	DB_variables_respuestaDB
------------------------	------------------	----------	--------------------------

**Campo ES\_2**

Tipo	Campo ES	Nombre	Campo ES_2
Posición X	495	Posición Y	217
Ancho	75	Altura	31
Nivel	0 - Nivel_0	Modo	Entrada/salida
Fuente	Tahoma, 17px		

**Dinamizaciones\Conexión de variable**

Nombre de la propiedad	Valor de proceso	Variable	DB_variables_tiempo_PLC
------------------------	------------------	----------	-------------------------

**Campo ES\_3**

Tipo	Campo ES	Nombre	Campo ES_3
------	----------	--------	------------

Posición X	376	Posición Y	268
Ancho	91	Altura	31
Nivel	0 - Nivel_0	Modo	Entrada/salida
Fuente	Tahoma, 17px		
<b>Dinamizaciones\Conexión de variable</b>			
Nombre de la propiedad	Valor de proceso	Variable	valordB

#### Campo ES\_4

Tipo	Campo ES	Nombre	Campo ES_4
Posición X	495	Posición Y	268
Ancho	75	Altura	31
Nivel	0 - Nivel_0	Modo	Entrada/salida
Fuente	Tahoma, 17px		
<b>Dinamizaciones\Conexión de variable</b>			
Nombre de la propiedad	Valor de proceso	Variable	TiempoTranscurrido

#### Campo de texto\_1

Tipo	Campo de texto	Nombre	Campo de texto_1
Posición X	211	Posición Y	223
Ancho	147	Altura	25
Nivel	0 - Nivel_0	Fuente	Tahoma, 17px
Texto	Cálculo usando PLC		

#### Campo de texto\_2

Tipo	Campo de texto	Nombre	Campo de texto_2
Posición X	570	Posición Y	223
Ancho	26	Altura	25
Nivel	0 - Nivel_0	Fuente	Tahoma, 17px
Texto	ms		

#### Campo de texto\_3

Tipo	Campo de texto	Nombre	Campo de texto_3
Posición X	400	Posición Y	182
Ancho	23	Altura	25
Nivel	0 - Nivel_0	Fuente	Tahoma, 17px
Texto	dB		

#### Campo de texto\_4

Tipo	Campo de texto	Nombre	Campo de texto_4
Posición X	480	Posición Y	182
Ancho	108	Altura	25
Nivel	0 - Nivel_0	Fuente	Tahoma, 17px
Texto	Tiempo usado		

#### Campo de texto\_5

Tipo	Campo de texto	Nombre	Campo de texto_5
Posición X	208	Posición Y	268
Ancho	150	Altura	25
Nivel	0 - Nivel_0	Fuente	Tahoma, 17px
Texto	Cálculo usando HMI		



**Campo de texto\_6**

Tipo	Campo de texto	Nombre	Campo de texto_6
Posición X	570	Posición Y	268
Ancho	12	Altura	25
Nivel	0 - Nivel_0	Fuente	Tahoma, 17px
Texto	s		

**Rectangle\_1**

Tipo	Rectángulo	Nombre	Rectangle_1
Posición X	392	Posición Y	104
Ancho	215	Altura	59
Nivel	0 - Nivel_0	Color de fondo	255; 0; 0
Color Borde	0; 0; 0		
<b>Dinamizaciones\Visibilidad</b>			
Variable - Ciclo	Espera -	Tipo de datos	Bit
Define el bit que debe monitorizarse.	0	Define la visibilidad en función de los valores de proceso seleccionados.	Visible

**Text field\_3**

Tipo	Campo de texto	Nombre	Text field_3
Posición X	400	Posición Y	109
Ancho	200	Altura	43
Nivel	0 - Nivel_0	Fuente	Tahoma, 32px, style=Bold
Texto	Procesando		
<b>Dinamizaciones\Visibilidad</b>			
Variable - Ciclo	Espera -	Tipo de datos	Bit
Define el bit que debe monitorizarse.	0	Define la visibilidad en función de los valores de proceso seleccionados.	Visible

**Campo de texto\_7**

Tipo	Campo de texto	Nombre	Campo de texto_7
Posición X	200	Posición Y	20
Ancho	399	Altura	46
Nivel	0 - Nivel_0	Fuente	Tahoma, 17px
Texto	Cálculo Banda de Octabva para 125 Hz Frecuencia de muestreo: 3500 muestras por segundo		

**Salir de runtime\_5**

Tipo	Botón	Nombre	Salir de runtime_5
Posición X	230	Posición Y	105
Ancho	133	Altura	62
Modo	Texto	Texto OFF	Calcular
Texto ON	Salir de runtime		
<b>Dinamizaciones\Evento</b>			
Nombre de evento	Hacer clic		

**Lista de funciones\calculardB\_3**

**Salir de runtime\_1**

Tipo	Botón	Nombre	Salir de runtime_1
Posición X	11	Posición Y	15
Ancho	133	Altura	62
Modo	Texto	Texto OFF	Atrás
Texto ON	Salir de runtime		

**Dinamizaciones\Evento**

Nombre de evento	Hacer clic
------------------	------------

**Lista de funciones\ActivarImagen**

Nombre de imagen	Grafica	Número de objeto	0
------------------	---------	------------------	---

**Copia impresa de Área permanente**

	domingo, 31 de diciembre de 2000
	10:59:39

Nombre	Área permanente	Color de fondo	182; 182; 182
Color Cuadrícula	0; 0; 0	Altura	60
Nivel activo	0		

**HmiScreenItemData**

Tipo	Campo de fecha y hora	Nombre	HmiScreenItemData
Posición X	695	Posición Y	30
Ancho	105	Altura	30
Modo	Salida	Fuente	Tahoma, 16px

**HmiScreenItemData\_1**

Tipo	Campo de fecha y hora	Nombre	HmiScreenItemData_1
Posición X	530	Posición Y	0
Ancho	270	Altura	30
Modo	Salida	Fuente	Tahoma, 16px

**Graphic view\_1**

Tipo	Visor de gráficos	Nombre	Graphic view_1
Posición X	10	Posición Y	3
Ancho	250	Altura	52
Nivel	0 - Nivel_0	Gráfico	ikerlan_1
Adaptar gráfico a tamaño objeto	Extender imagen		

***INTERFAZ HMI: INSTRUCCIONES***

Copia impresa de instrucciones

Instrucciones:

- 1: Hacer clic en el botón Salir.
- 2: Abrir Internet Explorer haciendo doble clic en el ícono del Escritorio.
- 3: Escribir dirección en barra de direcciones:  
172.16.7.32/rawdata/
- 4: Ingresar usuario y contraseña  
 usuario: admin  
 contraseña: 0000
- 5: Hacer clic derecho en el archivo deseado a guardar y elegir la opción "Save Target as...".
- 6: Guardar archivo en el directorio "\\Storage Card USB\".

**MUY IMPORTANTE:** guardar el archivo con el nombre "SM1281\_VIB1.wav".

- 7: Cerrar Navegador y hacer clic en "Start".
- 8: Hacer clic en el Botón "Convertir".
- 9: Esperar hasta que el cuadro "procesando" desaparezca de la pantalla.
- 10: Luego hacer clic en el botón "Calcular".
- 11: Esperar hasta que el cuadro "procesando" desaparezca de la pantalla.
- 12: Hacer clic en el boton "Graficar".

Dudas o consultas, remitirse al siguiente correo: [dasandoval@ikerlan.es](mailto:dasandoval@ikerlan.es)

1 Cerrar

Nombre	instrucciones	Ancho	670
Altura	470	Nivel activo	0

**Rectángulo\_1**

Tipo	Rectángulo	Nombre	Rectángulo_1
Posición X	8	Posición Y	9
Ancho	646	Altura	419
Nivel	0 - Nivel_0	Color de fondo	255; 255; 255
Color Borde	0; 0; 0		

**Campo de texto\_1**

Tipo	Campo de texto	Nombre	Campo de texto_1
Posición X	19	Posición Y	18
Ancho	561	Altura	403
Nivel	0 - Nivel_0	Fuente	Tahoma, 17px

<b>Texto</b>	<p>Instrucciones:</p> <p>1: Hacer clic en el botón Salir.</p> <p>2: Abrir Internet Explorer haciendo doble clic en el ícono del Escritorio.</p> <p>3: Escribir dirección en barra de direcciones: 172.16.7.32/rawdata/</p> <p>4: Ingresar usuario y contraseña usuario: admin contraseña: 0000</p> <p>5: Hacer clic derecho en el archivo deseado a guardar y elegir la opción "Save Target as...".</p> <p>6: Guardar archivo en el directorio "Storage Card USB".</p> <p>MUY IMPORTANTE: guardar el archivo con el nombre "SM1281_VIB1.wav".</p> <p>7: Cerrar Navegador y hacer clic en "Start".</p> <p>8: Hacer clic en el Botón "Convertir".</p> <p>9: Esperar hasta que el cuadro "procesando" desaparezca de la pantalla.</p> <p>10: Luego hacer clic en el botón "Calcular".</p> <p>11: Esperar hasta que el cuadro "procesando" desaparezca de la pantalla.</p> <p>12: Hacer clic en el botón "Graficar".</p> <p>Dudas o consultas, remitirse al siguiente correo: dasandoval@ikerlan.es</p>
--------------	---

#### Botón\_1

<b>Tipo</b>	Botón	<b>Nombre</b>	Botón_1
<b>Posición X</b>	23	<b>Posición Y</b>	431
<b>Ancho</b>	620	<b>Altura</b>	29
<b>Modo</b>	Texto	<b>Texto OFF</b>	Cerrar
<b>Texto ON</b>	Text		

#### Dinamizaciones\Evento

<b>Nombre de evento</b>	Hacer clic
-------------------------	------------

#### Lista de funciones\MostrarImagenEmergente

<b>Nombre de la imagen</b>	instrucciones	<b>Coordenada X</b>	0
<b>Coordenada Y</b>	0	<b>Modo de visualización</b>	Off
<b>Animación</b>	Off	<b>Velocidad de la animación</b>	Medio

**VBSSCRIPT: GENERAR\_10**

```
1 Sub generar_1()
2 '*****
3 ' Proyecto : Implementación de CMS industrial: caso de uso máquina eléctrica
4 '
5 ' Nombre de programa : generar
6 '
7 ' Autor : Diego Sandoval
8 '
9 ' Creado el : 13.06.2018
10 '
11 ' Propósito : Permite generar automáticamente el archivo en formato WAV de
12 ' la señal que esta siendo sensada con el SM1281 en el canal 1. Para ello se considera
13 ' que el SM ya está con sus parámetros configurados y se encuentra en el modo
14 ' STOP: System Ready. También se asume que esta habilitado el canal 1 para leer datos
15 ' y permite grabación de datos raw, el sensor se encuentra conectado y calibrado.
16 '*****
17
18 'declaracion variables para medición de tiempo
19 Dim StartTime, Endtime, ElapsedTime, delay_counter
20
21 'marcas de tiempo necesarios para ver tiempo de procesamiento
22 SetBit "Espera"
23 ElapsedTime = 0
24 StartTime = Timer()
25
26 'cambia para metro OpMode a 8 = modo medición
27 SmartTags("Param_DB_OpMode")=8
28
29 'Espera 0.5s para el cambio
30 delay_counter = 0
31 While delay_counter < 130000
32 delay_counter = delay_counter + 1
33 Wend
34
35 'activa el cambio de modo activando ActiveOpMode
36 SmartTags("Param_DB_ActiveOpMode")=1
37
38 'Espera 15s para que cambio de modo
39 delay_counter = 0
40 While delay_counter < 3900000
41 delay_counter = delay_counter + 1
42 Wend
43
44 'desactiva el bit ActiveOpMode
45 'SmartTags("Param_DB_ActiveOpMode")=0
46
47 'Espera 3s para que se realice la instruccion
48 delay_counter = 0
49 While delay_counter < 130000
50 delay_counter = delay_counter + 1
51 Wend
52
53 'indica que se grabe el archivo de datos raw
54 SmartTags("Param_DB_RawDataRecording")=1
55
56 'Espera 15 segundos para que grabe los datos
```

```
57 delay_counter = 0
58 While delay_counter < 3900000
59 delay_counter = delay_counter + 1
60 Wend
61
62 'resetea los bits de grabacion de datos
63 SmartTags("Param_DB_RawDataRecording")=0
64
65 'Espera 0.5s para el cambio
66 delay_counter = 0
67 While delay_counter < 130000
68 delay_counter = delay_counter + 1
69 Wend
70
71 'cambia el modo a 6= STOP:System Ready
72 SmartTags("Param_DB_ActiveOpMode")=0
73 SmartTags("Param_DB_OpMode")=6
74
75 'Espera 0.5s para el cambio
76 delay_counter = 0
77 While delay_counter < 130000
78 delay_counter = delay_counter + 1
79 Wend
80
81 'activa el cambio de estado del SM
82 SmartTags("Param_DB_ActiveOpMode")=1
83
84 'Espera 5s para que cambio de modo
85 delay_counter = 0
86 While delay_counter < 1300000
87 delay_counter = delay_counter + 1
88 Wend
89
90 'desactiva el bit
91 SmartTags("Param_DB_ActiveOpMode")=0
92 'Espera 0.5s para el cambio
93 delay_counter = 0
94 While delay_counter < 130000
95 delay_counter = delay_counter + 1
96 Wend
97
98 'entrega el tiempo total utilizado
99 Endtime = Timer()
100 ElapsedTime = Endtime-StartTime
101 ResetBit "Espera"
102 SmartTags("TiempoTranscurrido")= ElapsedTime
103 End Sub
```

### *VBSCRIPT: CONVERTIR 20*

```
1 Sub convertir_2()
2
3 *****
4 ' Proyecto : lectura datos WAV a valores reales
5 '
6 ' Nombre de programa : convertir
7 '
8 ' Autor : Diego Sandoval
9 '
10 ' Creado el : 01.06.2018
11 '
12 ' Propósito : Lectura de los datos de la señal contenida en un archivo tipo
13 'wav leyendo cada byte para luego ser reordenados y tranformados a un valor real
14 'decimal del tipo REAL. Estos valores seran guardados en un nuevo archivo de
15 'extension txt para su posterior uso.
16 *****
17
18 'declaracion de variables
19 'declaracion objetos
20 'declaracion ruta archivo
21 'declaracion apertura de archivo
22 'variables auxiliares
23 'variables para calcular valor
24 'medicion tiempo de cálculo
25
26
27 Dim archivo1, archivo2, archivo3, archivo4
28 Dim path1, path2, path3
29 Dim mode1, mode2, mode3
30 Dim S1, binario, i
31 Dim data1, data2, valor1, valor2, valor3, signo
32 Dim StartTime, Endtime, ElapsedTime
33
34 'señal visual de que esta corriendo el programa
35 SetBit "Espera"
36 ElapsedTime = 0
37 'iniciar contador de tiempo de ejecucion
38 StartTime = Timer()
39
40 'iniciar modo de apertura de archivo
41 mode1=32 '32 = binario
42 mode2=8 '8 = tipo Append
43 mode3=1 '1 = solo lectura
44
45 'indicar donde se encuentran guardados los archivos
46 path1 = "\Storage Card USB\SM1281_VIB1_2.wav"
47 path2 = "\Storage Card USB\dato_byte.txt"
48 path3 = "\Storage Card USB\dato_decimal.txt"
49
50 On Error Resume Next
51
52 'crear objetos para abrir los archivos
53 Set archivo1= CreateObject("FileCtl.File")
54 Set archivo2= CreateObject("FileCtl.File")
```

```
55
56 'se abren los archivos
57 archivo1.open path1,mode1
58 archivo2.open path2,mode2
59
60 'hasta aca solo hay errores si no esta presente el archivo WAV de origen
61 'o no se encuentre presente el dispositivo USB
62 'programacion extraccion byte desde archivo origen
63
64 'puesto que la lectura es a nivel de byte, leemos hasta la posicion del
65 'byte 589 para empezar la lectura de los datos en bruto
66 For i = 1 To 588
67 binario=archivo1.InputB(1)
68 Next
69
70 i=1
71 'bucle para leer los bytes requeridos
72
73 'mientras no estemos al final del archivo
74 While archivo1.EOF = False
75 For i=1 To 4
76 'obtiene el byte como un arreglo de 1 byte
77 binario=archivo1.InputB(1)
78 'se convierte el byte a string
79 binario = CStr(binario)
80 'se convierte a hexagesimal para enviar como char al archivo2
81 S1 = Hex(AscB(MidB(binario, 1, 1)))
82 If i < 3 Then
83 'se imprime una linea con el byte impreso como char
84 archivo2.LinePrint S1
85 End If
86 Next
87 Wend
88
89 'una vez copiados los valores desde el byte 45 hasta el final
90 'del archivo se cierran los archivos
91 archivo1.Close 'se cierra archivo1
92 archivo2.Close 'se cierra archivo2
93 Set archivo1 = Nothing 'se desenlazan los objetos archivos
94 Set archivo2 = Nothing 'se desenlazan los objetos archivos
95
96
97 'desde aca empieza el programa para convertir los 2 bytes en un
98 'valor real crea objetos para leer archivo donde se guardaron
99 'valores hexadecimal se leen datos y convierten a valores numericos
100
101 'se crean los objetos para convertir los bytes en valores reales
102 Set archivo3= CreateObject("FileCtl.File")
103 Set archivo4= CreateObject("FileCtl.File")
104
105 'se abren los archivos
106 archivo3.open path2,mode3
107 archivo4.open path3,mode2
108
109 'signo del valor a extraer (puede ser un valor entre -1 a +1)
110 signo=1
111 'bucle que extrae datos
112
```



```
113 'mientras no lleguemos al final del archivo, se extraen dato linea por linea
114 While archivo3.EOF = False
115 'se obtiene el dato por linea
116 data1=archivo3.LineInputString
117 'se agrega &H para formatear luego como valor hexadecimal
118 data1= "&H"&data1
119 'se convierte valor de base hexadecimal a entero base decimal
120 valor1=CInt(data1)
121
122 'se obtiene el bit siguiente y se hace el mismo proceso (wav con resolucion 16 bits)
123 data2=archivo1.LineInputString
124 data2= "&H"&data2
125 valor2=CInt(data2)
126
127 'comprobacion valor negativo o positivo para complemento a 2
128 'si el bit del signo es 1 entonces es negativo
129 'se invierten los valores en binario del numero expresado en binario
130 'se invierten los valores en binario del numero y se agrega un 1 en binario operando
131 'en decimal se pone negativo el signo al ser negativo el numero
132 'de no ser negativo, el signo es positivo
133
134
135 If valor2>=128 Then
136 valor2=255-valor2
137 valor1=256-valor1
138 signo=-1
139 Else
140 signo=1
141 End If
142
143 'conversion 2 bytes separados a un solo numero
144
145 'se suman ambos bytes en un valor entero de 2 bytes operando en decimal
146 valor3=signo*((valor2*256 + valor1)/32767)
147 'se fija el valor del tipo single para ser exportado como numero decimal
148 valor3=CSng(valor3)
149 'se escribe en archivo 4
150 archivo4.LinePrint(valor3)
151 Wend
152
153 archivo3.Close 'se cierra archivo3
154 archivo4.Close 'se cierra archivo4
155 Set archivo3 = Nothing 'se desentazan los objetos archivos
156 Set archivo4 = Nothing 'se desentazan los objetos archivos
157
158 'calculo tiempo transcurrido en realizar calculo
159 Endtime = Timer()
160 ElapsedTime = Endtime-StartTime
161 SmartTags("TiempoTranscurrido")= ElapsedTime 'mostrar tiempo transcurrido en segundos
162 ResetBit "Espera" 'desaparece señal visual de espera
163
164 End Sub
```

***VBS SCRIPT: CALCULODB 30***

```

1 Sub calculodB()
2
3 *****
4 ' Proyecto : Implementación de CMS industrial: caso de uso máquina eléctrica
5 '
6 ' Nombre de programa : calculodB
7 '
8 ' Autor : Diego Sandoval
9 '
10 ' Creado el : 18.06.2018
11 '
12 ' Propósito : Procesamiento de los datos de señal a valores segun filtro
13 'de banda de octava a 125 Hz para una frecuencia de muestreo de 3500 Hz. Programa
14 'portado de código generado por MATLAB para ser utilizado en lenguaje SCL para PLC
15 'utilizando TIA Portal.
16 *****
17
18 'inicialización de las variables
19 Dim archivo_origen
20 Dim origen
21 Dim modo_input
22 Dim data, delimiter
23 Dim splitdata
24 Dim i, valor(3499)
25 Dim StartTime, Endtime, ElapsedTime
26
27 'variables para algoritmo del filtro
28 Dim k,rtb_final(3499),naxpy,j,indice_i,X,a_s,calc, b_a, b_c
29 'los valores fueron calculados para una frecuencia de muestreo de 3500 Hz por el
30 'limite de memoria impuesto por el PLC.
31 'estos valores deben ser calculados segun la banda y la frecuencia de muestreo a
32 'utilizar.
33 b_a =
34 Array(0.00042873018317410143,0.0,-0.001286190549522,0.0,0.001286190549522,0.0,-0.00042
35 873018317410143)
36 b_c = Array(1.0,-5.5401939386901624,12.928277688192221,-16.261736159518428,
37 11.628149435302925,-4.482221650813031,0.727832871740883)
38
39
40 'iniciar modo de apertura de archivo
41 modo_input=1 '1 = Input
42 origen ="\Storage Card USB\dato_db.txt" 'ubicacion de archivo dentro del hmi, en el
43 USB conectado
44 delimiter=";"
45
46 'indicar que se inicia el cálculo y medición de tiempo
47 SetBit "Espera"
48 StartTime = Timer()
49
50 On Error Resume Next
51
52 'crear objeto para abrir archivo con datos y abrirlo
53 Set archivo_origen = CreateObject("FileCtl.File")
54 archivo_origen.open origen,modo_input
55
56 'se leen los datos de archivo de origen y son guardados en el vector valor()
57 For i = 0 To UBound(valor)

```

```
52 data=archivo_origen.LineInputString
53 splitdata=Split(data,delimiter)
54 'por como funciona VBS, es necesario multiplicar para que los valores sean
    decimales
55 'y no enteros como entrega VBS al momento de extraer los valores del archivo.
56 'El motivo de esta situación es desconocido.
57 'Se crea este Case para agregar las posiciones necesarias
58 Select Case Len(splitdata(0))
59 Case 6
60 valor(i) = splitdata(0)*1.0E-4
61 Case 7
62 valor(i) = splitdata(0)*1.0E-5
63 Case 8
64 valor(i) = splitdata(0)*1.0E-6
65 Case 9
66 valor(i) = splitdata(0)*1.0E-7
67 Case Else
68 valor(i) = splitdata(0)*1.0E-7
69 End Select
70 Next
71
72 'algoritmo para filtro banda de octava @3500 muestras por segundo
73 'en teoria es posible usar este mismo algoritmo cambiando solo los vectores
74 'b_a y b_c para la banda y muestreo correspondiente
75 For k = 0 To 3499
76 rtb_final(k) = 0.0
77 Next
78 For k = 0 To 3499
79 If (3499 - k) < 7 Then
80 naxpy = 3500 - k
81 Else
82 naxpy = 7
83 End If
84 j = 0
85 Do While (j + 1) <= naxpy
86 rtb_final(k + j) = rtb_final(k + j) + (valor(k) * b_a(j))
87 j = j + 1
88 Loop
89 If (3499 - k) < 6 Then
90 naxpy = 3499 - k
91 Else
92 naxpy = 6
93 End If
94 a_s = -rtb_final(k)
95 j = 1
96 Do While j <= naxpy
97 rtb_final(k + j) = rtb_final(k + j) + (a_s * b_c(j))
98 j = j + 1
99 Loop
100 Next
101 X = 0.0
102 For indice_i = 0 To 3499
103 X = X + ((rtb_final(indice_i))*(rtb_final(indice_i)))
104 Next
105
106 calc=(Log(Sqr(X/3499.0) / 1.0E-6) / Log(10)) * 20.0
107
108 SmartTags("valor_dB") = calc
```

```

109
110 'cerrar archivo
111 archivo_origen.Close
112 Set archivo_origen = Nothing
113 'mostrar tiempo de computo
114 ResetBit "Espera"
115 Endtime = Timer()
116 ElapsedTime = Endtime-StartTime
117 SmartTags("TiempoTranscurrido")= ElapsedTime
118 End Sub

```

### CÓDIGO MATLAB: OCTBAND

```

function [octband f]=octband(P,P0,fs,fmin,fmax,averaging,tr)
%
% P= signal;
% P0= 1e-6;
% fs= 46875;
% fmin= 65;
% fmax= 200
% averaging='linear';
% tr=0;

fc=[16 31.5 63 125 250 500 1000 2000 4000 8000 16000]';

fcpositions = (fc >= fmin) & (fc <= fmax);
f = fc(fcpositions);
octband=zeros((length(f)),1);
x=P;
if f(1) < fs/100
    midx=(find(f < fs/100, 1, 'last' ));
    nfs=round(fs/20);
    D=round(fs/nfs); % Decimation factor
    nfs=fs/D;
    %nx=resample(x,1,D); %señal resampleada
    for n=1:midx
        [b,a] = octfilt(fs,f(n));
        y=filter(b,a,x); %sin resampling
        %[b,a] = octfilt(nfs,f(n));
        %y=filter(b,a,nx); %sin resampling
        if strcmpi(averaging,'linear')
            octband(n)=linears(y,P0);
        else
            octband(n)=exponential(y,P0,tr,1/nfs);
        end
    end
    for n=midx+1:length(f)
        [b,a] = octfilt(fs,f(n));
        y=filter(b,a,x);
        if strcmpi(averaging,'linear')
            octband(n)=linears(y,P0);
        else
            octband(n)=exponential(y,P0,tr,1/fs);
        end
    end
end
else

```

```
for n=1:length(f)
    [b,a] = octfilt(fs,f(n));
    y=filter(b,a,x);
    if strcmp(averaging,'linear')
        octband(n)=linears(y,P0);
    else
        octband(n)=exponential(y,P0,tr,1/fs);
    end
end
end
```

### **CÓDIGO MATLAB: OCTFILT**

```
function [b,a] = octfilt(fs,f0)
%OCTFILT octave-band filter
% [b,a] = octfilt(fs,f0)
% fs sampling frequency
% f0 center frequency

flo = f0/sqrt(2);
fhi = f0*sqrt(2);
fny = fs/2;
if fhi>fny
    fhi=fny-1;
end
[b,a] = butter(3,[flo/fny fhi/fny]);
```

### **CÓDIGO MATLAB: LINEARS**

```
function [Lp]=linears(P,P0)
X=0;
r=length(P);
parfor i=1:r
    X=X+(P(i))^2;
end
Nr=sqrt(X/r);
Lp=20*log10(Nr/P0);
Lp=max(0,Lp);
```

### **CÓDIGO MATLAB: EXPONENCIAL**

```
function [Lp]=exponential(P,P0,tr,dt)
N=length(P);
Nr=0;
for i=1:N
    Nr=Nr*(1-2*dt/tr)+2*dt/tr*((P(i))).^2;
end
Lp=20*log10(sqrt(Nr)/P0);
Lp=max(0,Lp);
```

***CÓDIGO MATLAB: FILTRITO 3500***

```
function final=filtrito_3500(entrada)

%fs=46875;
P0=1e-6;
%D=20;
%nfs=fs/D;
%redimensionado=resample(nx,1,D);

a=[1 -5.540193938690162 12.928277688192221 -16.261736159518428
11.628149435302925 -4.482221650813031 0.727832871740883];
b=[4.287301831741014e-04 0 -0.001286190549522 0 0.001286190549522 0 -
4.287301831741014e-04];

y=filter(b,a,entrada);

X=0;
r=length(y);
for i=1:r
    X=X+(y(i))^2;
end
Nr=sqrt(X/r);
final=20*(log(Nr/P0)/log(10));
final=max(0,final);
```

***Anexo II: Presupuesto***

<b>MÁSTER DE INGENIERÍA MECATRÓNICA</b>		<b>TRABAJO CAPITULO EQUIPO</b>		<b>Pag. MEDICIONES Y PRESUPUESTO</b>		
<b>Nº ORDEN</b>	<b>CONCEPTOS</b>	<b>Nº UNIDADES</b>	<b>FABRICANTE</b>	<b>TIPO</b>	<b>PRECIO UNITARIO MATERIAL</b>	<b>TOTAL</b>
	SIPLUS CMS1200 SM 1281 Condition Monitoring	1	Siemens	6AT8007-1AA10-0AA0	1303.4	1.303,4
	Juego de estribos apantallados para SIPLUS CMS1200 SM 1281	1	Siemens	6AT8007-1AA20-0AA0	65.17	65.17
	Sensor VIB S01 captador de vibraciones, rango de frecuencia 0,5Hz	1	Siemens	6AT8002-4AB00	377.72	377.72
	Cable-MIL-1000, cable de conexión para conectar sensores VIB, longitud 10,0m	1	Siemens	6AT8002-4AC10	266	266
	PLC S7 1200 1212C AC/DC/Ryl	1	Siemens	6ES7 212-1HE31-0XB0	260	260
	Protección magnetotérmica Acti9 iC60N C16A	3	Schneider	A9F04202	50	150
	Fuente 24VDC DRB480-24-1	1	TDK-Lambda	DRB480241	220.2	220,2
	Switch D-link DGS-1008D	1	D-link	DGS-1008D	25	25
	Hora Hombre	960	--	---	25	24.000
<b>Total</b>						<b>24.2667,49</b>

