BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE INGENIERÍA

COLEGIO DE INGENIERÍA MECÁNICA Y ELÉCTRICA

“Monitoreo estructural de las vibraciones mecánicas mediante PLC S7-1200 y Arduino“

TESIS

QUE PARA OBTENER EL TITULO DE:

LICENCIADO EN INGENIERÍA MECÁNICA Y ELÉCTRICA

PRESENTA:

Víctor Israel Gómez Romero

ASESOR:

Dr. FILIBERTO CANDIA GARCÍA

COASESOR

Mtro. Víctor Galindo López

Puebla, Pue. Agosto 2020

Acta resolutiva de impresión de tesis

BUAP

Oficio D-SA 0374/2020

C. VÍCTOR ISRAEL GÓMEZ ROMERO
PASANTE DE LA CARRERA DE INGENIERÍA
MECÁNICA Y ELÉCTRICA
P r e s e n t e.

En atención al Tema de Tesis que puso Usted a consideración de la Coordinación de Área y de esta Secretaría Académica en coordinación con la Dirección de ésta Facultad de Ingeniería, dentro del marco de Titulación por Examen Profesional, como medio de Titulación se dio revisión y se ha autorizado el tema denominado:

"MONITOREO ESTRUCTURAL DE LAS VIBRACIONES MECÁNICAS MEDIANTE PLC S7 – 1200 Y ARDUINO".

Por lo anterior hacemos de su conocimiento que se asigna como asesor al Dr. Filiberto Candia García y como Co Asesor al Mtro. Víctor Galindo López.

Sin más por el momento, le envío la seguridad de mi consideración más distinguida.

A t e n t a m e n t e
"Pensar bien, para vivir mejor"
H. Puebla de Z. a 07 de febrero de 2020

M. en I. Fernando Daniel Lazcano Hernández
Director

M'FDLH/M'JAJT/BARV
C.c.p. Interesado
C.c.p. Archivo

Facultad de Ingeniería
Blvd. Valsequillo y Av. San Claudio s/n, edif. ING 4, Col. San Manuel, Ciudad Universitaria, Puebla, Pue. C.P. 72570
01 (222) 229 55 00 Ext. 7610

M. en I. Fernando Daniel Lazcano Hernández
Director de la Facultad de Ingeniería
Benemérita Universidad Autónoma de Puebla
P r e s e n t e.

El que suscribe: Dr. Filiberto Candia García, Asesor del proyecto:

"MONITOREO ESTRUCTURAL DE LAS VIBRACIONES MECÁNICAS MEDIANTE PLC S7 – 1200 Y ARDUINO".

Presentada por el C. Víctor Israel Gómez Romero, pasante del Colegio de Ingeniería Mecánica y Eléctrica, y en atención al oficio No. D – SA 0374/2020 con fecha de emisión 07 de febrero de 2020, me permito informar a Usted que después de haber revisado cuidadosamente el contenido temático, metodología, redacción y ortografía de la tesina correspondiente, no tengo inconveniente en autorizar la impresión de la misma.

Sin otro particular, le reitero la seguridad de mi más atenta y distinguida consideración.

A t e n t a m e n t e
"Pensar bien, para vivir mejor"
H. Puebla de Z. a 05 de agosto de 2020

Dr. Filiberto Candia García
Asesor

D'FCG/BARV
C.c.p. Interesado
C.c.p. Archivo

Agradecimientos

A Dios.

A mis padres y hermano por el apoyo incondicional que me han otorgado, su fe en mí y sobre todo por el sacrificio que han hecho para que esté aquí.

A mi esposa por todo, hoy y siempre.

A mis hijos por ser mi motor.

A mis profesores y compañeros

Contenido

Resumen

El monitoreo de la salud estructural en el ámbito industrial y sobre todo en el automotriz es un factor determinante e indicativo de una buena calidad del producto y la minimización de fallas en el mismo. La prevención de deterioro de un sistema estructural mediante la medición de las vibraciones es una técnica que permite la predicción de fallas y su rápida atención antes de colapso estructural. Siendo el objetivo contar con un prototipo que permita en las Instituciones Educativas (básica, media superior, superior y posgrado), y el objetivo específico fomentar el diagnóstico por monitoreo de la salud estructural en modelos de 1 gdl (grado de libertad). La metodología es teórica-experimental, con base en el método científico-deductivo, los resultados se obtienen por manipulación directa en un prototipo funcional. El desarrollo comprende el diseño de la estructura de 1 gdl y su fabricación, la concepción de una interfaz electrónica-gráfica en Simulink, que se consideran software analítico de alto desempeño, y se complementa con el sensor MPU6050 que es el medio para la adquisición de las señales de vibración que muestran el comportamiento de la estructura. Se han realizado hasta el momento simulaciones y pruebas del comportamiento de la estructura, donde los resultados han sido comparables y congruentes con la teoría. Esto es un mayor esfuerzo de Von Mises, cuando existe un mayor apriete de las conexiones tornillo, debido a que se origina un incremento del esfuerzo cortante en el plano de mayor rigidez. Se ha concluido que enseñar mediante un prototipo físico y por simulación, el comportamiento de estructuras de 1 gdl ante vibraciones aleatorias, resulta en la reducción de la incertidumbre de la salud estructural de las estructuras de 1 gdl.

Introducción

El presente desarrollo tecnológico, consiste en el diseño de un dispositivo mecatrónico que monitorea en tiempo real el estado de las vibraciones de una estructura destinada a soportar máquinas generadoras de movimientos armónicos (aerogeneradores, bombas, compresores, ventiladores, turbinas, entre otras) que afectan la rigidez de la estructura que ha sido ensamblada por medio de remaches o tornillos. Esta tipología estructural se ha determinado por ser una recurrente solución para la instalación de máquinas/equipos dentro de las instalaciones/procesos industriales de producción o fabricación, debido a que es una incorporación al proyecto original (ya sea para incrementar la eficiencia o eficacia o para compensar errores en el proyecto original). Asimismo, es una práctica usual en las empresas ensambladoras (industria automotriz, aeronáutica, etc.) o de logística (FEDEX, UPS, DHL) cuando se instalan transportadores (bandas, rodillos, etc.) para desplazar los materiales o productos aun no terminados.

Por ello es requerido de manera indispensable que las estructuras destinadas a mantener las máquinas/equipos que producen movimientos armónicos, sean rígidas, ya que de lo contrario se pueden provocar valores peligrosos de variación de rigidez (pérdida) y generar inclusive condiciones de resonancia (Martínez, Pino, Terán, & Arteaga, 2017) (Martínez, Pino, Terán, & Arteaga, Vibraciones Mecánicas I, 2017) (Pope, 2000).

Ante esta problemática y considerando que los proyectos de las empresas petroquímicas, energías alternativas, ensambladoras o de logística son inversiones de miles de millones de dólares, se justifica el desarrollo de productos o dispositivos que sean capaces de monitorear en tiempo real el estado de las vibraciones en cada elemento del proceso determinado crítico o vital. Asimismo, el conocer el estado de esfuerzo (que deteriore la rigidez) que puede alcanzar una máquina/equipo por un evento aleatorio como los sismos es una condición deseable, pues permitirá el diseño de estructuras de soporte (figura 1), que se consideran como masa no estructural, pues no adicionan rigidez a la estructura principal.

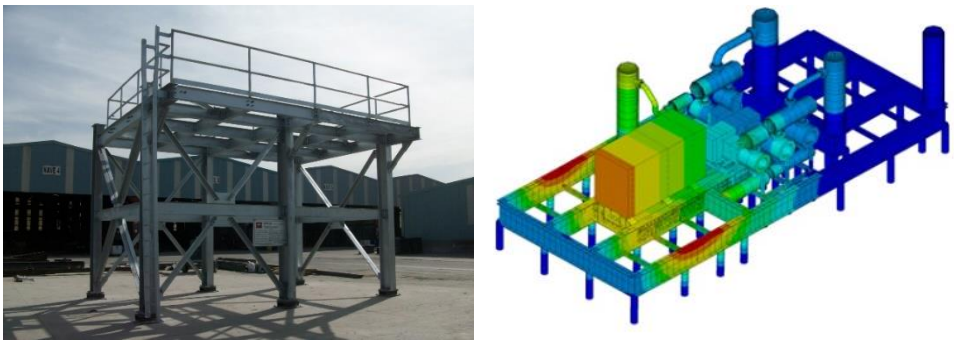


Figura 1. Estructuras de soporte para máquina/equipo rotatorio.

Los resultados a obtener serán un dispositivo capaz de monitorear condiciones de falla en máquinas/equipos, para determinar la frecuencia de vibración, periodo del movimiento y sus valores máximos periódicos, a través de gráficas en función del tiempo y frecuencia.

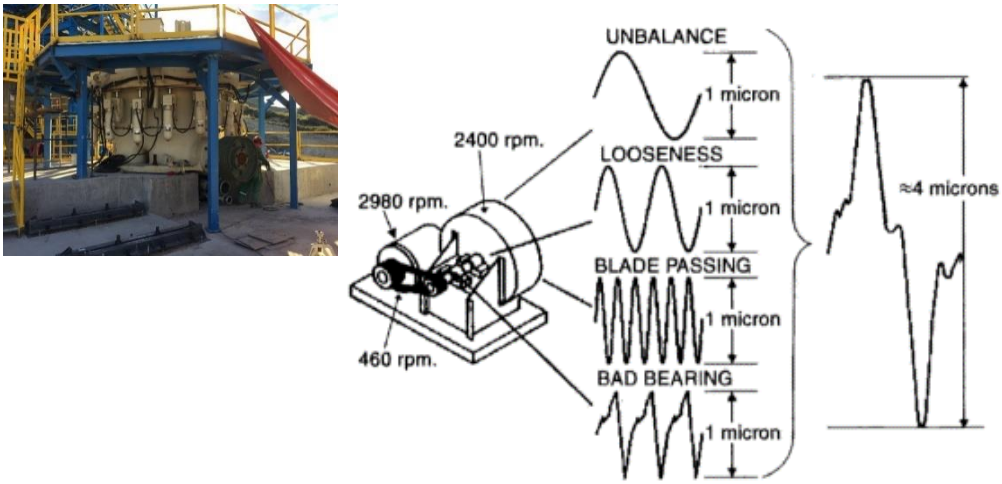


Figura 2. Dispositivo generador de movimientos armónicos de baja frecuencia.

Se concluye que el monitorear el comportamiento de la rigidez de una estructura ante el efecto y consecuencias de las cargas dinámicas, es una solución confiable y económica que protege los bienes materiales de mayor relevancia en una empresa.

Capítulo I.- Protocolo de Investigación

Descripción de la situación

Cuando una máquina/equipo rotativo es soportado por una estructura rectangular de acero que ha sido fabricada con remaches o tornillos como medio de sujeción en lugar de soldadura o maquinado en una solo pieza. La estructura debido a la carga armónica permanente tiende a perder rigidez (figura 3) por el aflojamiento de los medios de sujeción, los cuales al alcanzar niveles de resonancia tienden a generar fisuras y falla estructural. Sobre todo, en dispositivos como las turbinas o aerogeneradores que estas destinados a la producción de energía eléctrica.

Figura 3. Ejemplos pérdida de rigidez en acoplamiento / sujeción.

Las instalaciones de naves industriales (figura 4) de las industrias petroquímicas, ensambladoras o de logística, generalmente son estructuras con cimentación local que no considera la adición o incorporación de elementos rotatorios (figura 5), los cuales tienden a influenciar el Layout de la producción.

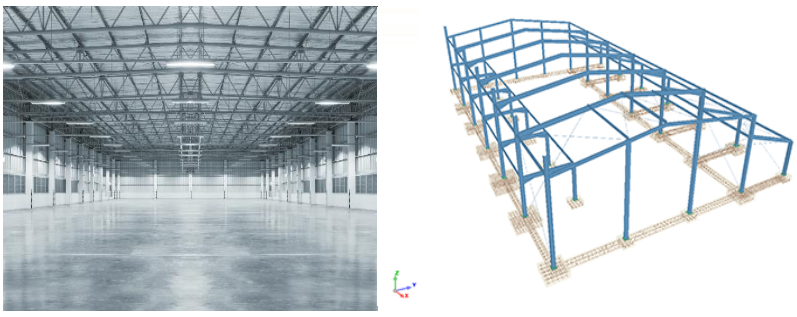


Figura 4. Tipología de nave industrial.

Figura 5. Tipología de las estructuras que soportan máquinas/equipos rotatorios.

Justificación

Diseñar un concepto integral (prototipo) que permita mostrar las técnicas de monitoreo de la salud estructural de las estructuras de 1 gdl, permite contribuir a la formación profesional de los estudiantes de ingeniería Mecánica y Eléctrica/Civiles de las IES, mediante la implementación de software de alto nivel, para llevar a cabo el análisis de la respuesta dinámica de la estructura, es posible que las representaciones de falla o comportamiento no adecuado de las estructuras sean identificados y valorados en función de los registros de sus frecuencias naturales.

Objetivos

Objetivo general

El objetivo por lo tanto es el desarrollo de un sistema de monitoreo de bajo costo basado en el uso de PLC S7-1200 y Arduino. Con la finalidad de incorporar un sistema de control que permita la desconexión de los equipos de manera segura dentro del proceso y emita las correspondientes alarmas de seguridad.

Objetivos específicos del proyecto

- Fabricar / Desarrollar la interface de adquisición de señales con sensor MPU6050, Arduino UNO y el Arduino Support Package.
- Identificar de manera gráfica el valor máximo en amplitud y frecuencia de condiciones de riesgo operativas.

Hipótesis

Si se cuenta con un prototipo que permita modificar el estado operativo de los elementos de las conexiones y la geometría de una estructura de acero de 1 gdl. Entonces es posible monitorear la salud estructural a partir de la frecuencia de operación y la amplitud de las señales de trabajo no estables o preestablecidas.

Variables

El estado de operativo de los elementos de conexión por sujeción y la variación de la geometría, se determinan como la variable independiente. La frecuencia natural (ω_n) de operación del sistema, los esfuerzos (σ_r)y las deformaciones (δ_n) de las conexiones de la estructura, se consideran como la actuación de la estructura, cuando es sometida a condiciones diferentes de esfuerzo para las cuales fue diseñada originalmente. Por lo tanto, se consideran como la variable dependiente.

Capítulo II.- Marco de Referencia

Estudio técnico-económico

La relación costo beneficio es favorable, debido a que el dispositivo es relativamente económico con respecto al costo de un equipo rotatorio industrial, que puede verse afectado y destruido por la pérdida de rigidez de la estructura que lo soporta o por entrar en resonancia.

La falta de monitoreo conlleva a situaciones donde el daño se hace evidente una vez que la falla es crítica y/o demasiado costosa de reparar, implementar sistemas de monitoreo de salud estructural representa una inversión sumamente rentable, permitiendo reducir costos altamente significativos de mantenimiento, pérdidas de producción por reparaciones, y accidentes que puedan afectar la integridad física de personas que se encuentren cerca de las estructuras al detectar problemas en su etapa más temprana, así como asegurar un tiempo de vida útil de los bienes materiales.

Sensor MPU-6050

El MPU-60X0[1] tiene integrado un giroscopio MEMS (Sistemas Microelectromecánicos) de 3 ejes, un acelerómetro de 3 ejes MEMS, y un procesador digital de movimiento (DMP™) motor acelerador de hardware con un puerto I2C auxiliar que se conecta a las interfaces de sensores digitales de terceras partes tales como magnetómetros. Cuando se conecta a un magnetómetro de 3 ejes, el MPU-60X0 entrega una salida completa de 9 ejes MotionFusion para su primario I2C o puerto SPI (SPI está disponible sólo en MPU-6000)».

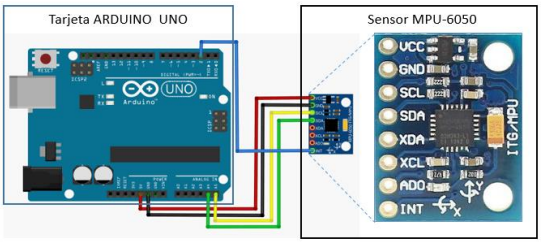


Figura 7. Sensor MPU-6050

El MPU-60X0 combina la aceleración y el movimiento de rotación más la información de rumbo en un único flujo de datos para la aplicación. Esta integración de la tecnología

[1] https://www.diarioelectronicohoy.com/blog/sensor-mpu6050

MotionProcessing™ presenta un diseño más compacto y tiene ventajas de costos inherentes en comparación con soluciones discretas de giroscopio más acelerómetro. El MPU-60X0 también está diseñado para interactuar con múltiples sensores digitales no inerciales, tales como sensores de presión, en su bus I2C auxiliar maestro. El MPU-60X0 es un procesador de movimiento de segunda generación y es la huella compatible con la familia MPU-30X0.

Así pues, el sensor MPU-6050 es una pequeña pieza tecnológica de procesamiento de movimiento. El cual mediante la combinación de un MEMS giroscopio de 3 ejes y un MEMS acelerómetro de 3 ejes en la misma pastilla de silicio junto con un DMP™ (Movimiento Digital Processor™), es capaz de procesar los algoritmos de movimientos complejos de 9 ejes (MotionFusion™) en una placa.

El MPU-6050 elimina los problemas de alineación del eje transversal que puede arrastrarse hacia arriba en porciones discretas. Las piezas integran el algoritmo MotionFusion para 9 ejes 'pueden incluso acceder a magnetómetros externos u otros sensores a través de un bus I2C auxiliar maestro, permitiendo reunir un conjunto completo de dispositivos sensores de datos, sin la intervención del procesador del sistema. El MPU-6050 es un 6 DOF (grados de libertad = Degrees of Freedom) o un sensor IMU de seis ejes, lo que significa que da seis valores de salida.

El sensor MPU-6050 es muy preciso, ya que contiene una conversión hardware de 16 bits de A/D por cada canal, para la digitalización de las salidas del acelerómetro. Para ello capta los canales x, y y z al mismo tiempo. Como se ha comentado, el sensor utiliza el I2C-bus para interconectar con el Arduino. Aunque algunas versiones (como la mostrada) llevan un regulador que permite conectarla a 5V.

Arduino

Arduino[2] es una plataforma electrónica de código abierto basada en hardware y software fáciles de usar. Las placas Arduino pueden leer entradas (luz en un sensor, un dedo en un botón o un mensaje de Twitter) y convertirla en una salida: activar un motor, encender un LED y publicar algo en línea. Puede decirle a su tarjeta qué debe hacer enviando un conjunto de instrucciones al microcontrolador de la tarjeta. Para hacerlo, utiliza el lenguaje de

[2] https://www.arduino.cc/en/Guide/Introduction

programación Arduino (basado en Wiring) y el software Arduino (IDE), basado en el procesamiento.

Figura 8. Placa Arduino uno.

Arduino nació en el Instituto de Diseño de Interacción Ivrea como una herramienta fácil para la creación rápida de prototipos, dirigida a estudiantes sin experiencia en electrónica y programación. Tan pronto como llegó a una comunidad más amplia, la placa Arduino comenzó a cambiar para adaptarse a las nuevas necesidades y desafíos, diferenciando su oferta de tablas simples de 8 bits a productos para aplicaciones IoT, impresión portátil, impresión 3D y entornos integrados. Todas las placas Arduino son completamente de código abierto, lo que permite a los usuarios construirlas de forma independiente y, eventualmente, adaptarlas a sus necesidades particulares. El software también es de código abierto y está creciendo a través de las contribuciones de los usuarios de todo el mundo.

Gracias a su experiencia de usuario simple y accesible, Arduino se ha utilizado en miles de proyectos y aplicaciones diferentes. El software Arduino es fácil de usar para principiantes, pero lo suficientemente flexible para usuarios avanzados. Se ejecuta en Mac, Windows y Linux. Los profesores y los estudiantes lo utilizan para construir instrumentos científicos de bajo costo, para probar los principios de química y física, o para comenzar con la programación y la robótica.

Hay muchos otros microcontroladores y plataformas de microcontroladores disponibles para la computación física. Parallax Basic Stamp, BX-24 de Netmedia, Phidgets, Handyboard de MIT y muchos otros ofrecen una funcionalidad similar. Todas estas herramientas toman los detalles desordenados de la programación del microcontrolador y lo envuelven en un paquete fácil de usar. Arduino también simplifica el proceso de trabajo con microcontroladores, pero

ofrece algunas ventajas para profesores, estudiantes y aficionados interesados sobre otros sistemas:

- Barato
- Multiplataforma
- Entorno de programación simple y claro
- Software de código abierto y extensible
- Fuente abierta y hardware extensible

PLC S7-1200

El PLC *Simatic S7-1200*[3] (figura 9) ofrece a los profesionales de la instalación un amplio abanico de características técnicas entre las cuales cabe destacar las siguientes:

Figura 9. S7-1200 Starter Kit

- Alta capacidad de procesamiento. Cálculo de 64 bits.
- Interfaz Ethernet / PROFINET integrado.
- Entradas analógicas integradas.
- Bloques de función para control de ejes conforme a PLCopen.

Programación mediante la herramienta de software STEP 7 Basic v15 para la configuración y programación no sólo del S7-1200, sino de manera integrada los paneles de la gama Simatic

[3] https://w5.siemens.com/spain/web/es/industry/automatizacion/simatic/controladores_modulares/controlador_basico_s71200/pages/s7-1200.aspx

Basic Panels. El sistema S7-1200 desarrollado viene equipado con cinco modelos diferentes de CPU (CPU 1211C, CPU 1212C, CPU 1214C, CPU 1215C y CPU 1217C) que se podrán expandir a las necesidades y requerimientos de las máquinas.

Signal Board: Puede añadirse en la parte frontal de cualquiera de las CPUs de manera que se pueden expandir fácilmente las señales digitales y analógicas sin afectar al tamaño físico del controlador.

Módulos de señal: A la derecha de la CPU (a excepción de la CPU1211C) pueden colocarse los módulos de ampliación de E/S digitales y analógicos. La CPU 1212C está capacitada para aceptar hasta dos módulos, la CPU 1214C, CPU 1215C y CPU 1217C hasta un total de ocho módulos de señal.

Módulos de comunicación: Todas las CPUs Simatic S7-1200 pueden equiparse hasta con tres módulos de comunicación los cuales se colocan a la izquierda del controlador, lo que permite una comunicación sin discontinuidades. Esto módulos son:

- PROFIBUS Maestro/esclavo
- Comunicación GPRS
- AS-i y más sistemas Fieldbus

Procesamiento de señales Matlab y Simulink

El procesamiento de señales es esencial para una amplia gama de aplicaciones, desde la ciencia de datos hasta los sistemas integrados en tiempo real. MATLAB ® y Simulink ® productos hacen que sea fácil de usar técnicas de procesamiento de señales para explorar y analizar datos de series temporales, y proporcionan un flujo de trabajo unificado para el desarrollo de sistemas integrados y aplicaciones de transmisión.

Con los productos de procesamiento de señal de MATLAB y Simulink, es posible:

- Adquirir medir y analizar señales de muchas fuentes.
- Diseñar algoritmos de transmisión para dispositivos de audio, sensores inteligentes, instrumentación y IoT.
- Realizar prototipos, pruebe e implemente algoritmos DSP en PC, procesadores integrados, SoC y FPGA.

Los productos de procesamiento de señales y MATLAB ayudan a analizar las señales de una variedad de fuentes de datos. Puede adquirir, medir, transformar, filtrar y visualizar señales sin ser un experto en teoría de procesamiento de señales. Puede aplicar herramientas de procesamiento de señal para:

- Preprocesar y filtrar las señales antes del análisis.
- Explore y extraiga funciones para análisis de datos y aplicaciones de aprendizaje automático.
- Analizar tendencias y descubrir patrones en señales.
- Visualice y mida las características de tiempo y frecuencia de las señales.

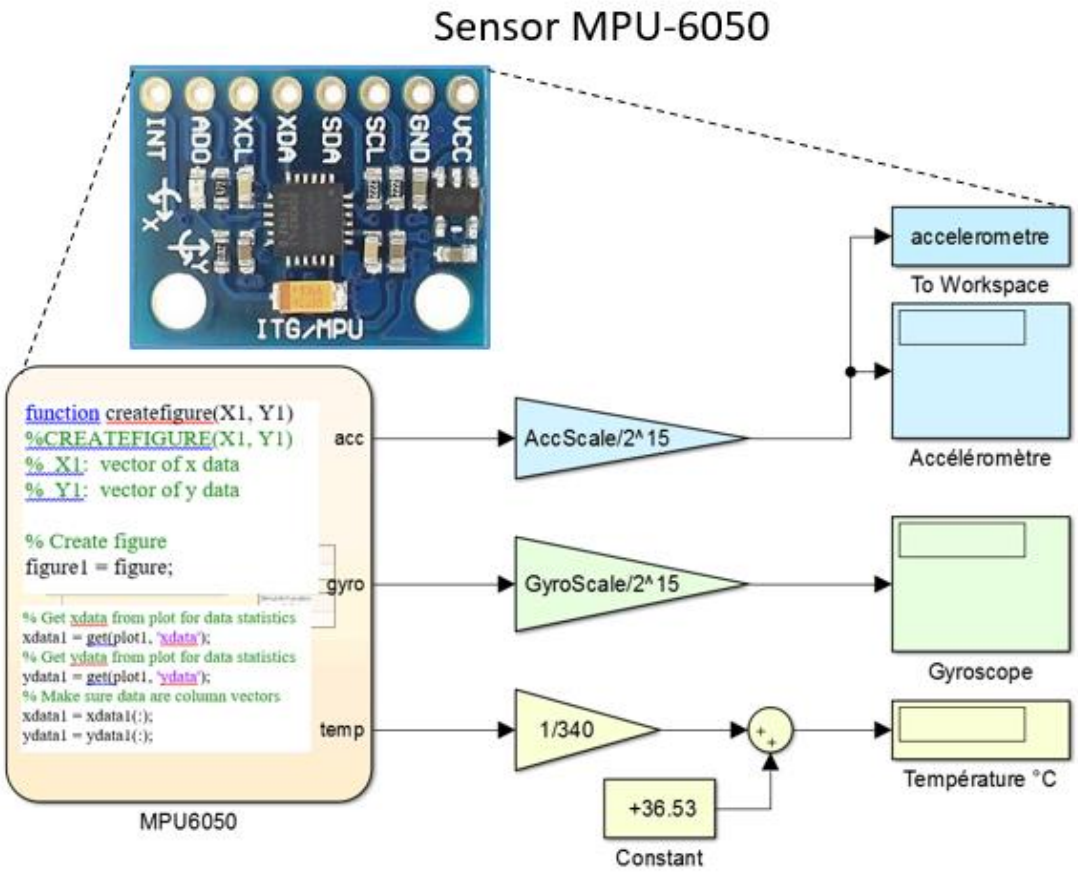


Figura 10. Procesamiento de señales de un sensor de vibración.

Impacto ambiental

Considerando que los equipos rotatorios industriales se destinan a procesos de la industria petroquímica, química y energías alternativas, el daño que puede causar al ecosistema por la

falla de los mismos es alta; el cual puede ir desde derrames, daño a estructuras cercanas, explosiones, entre otros. Sobre todo en los aerogeneradores que se incorporan de manera arquitectónica con las fachadas de las construcciones (Meléndez, 2017).

Por ello, el contar con un sistema de monitoreo representa una manera de maximizar el rendimiento de las instalaciones al corregir problemas causados por la pérdida de rigidez en las estructuras, que pueden dar pauta a un incidente en el que sustancias contaminantes (aceites, gases derivados de explosiones, solventes, etc.) sean arrojados al ambiente, afectando al medio.

Capítulo III.- Marco Metodológico

La metodología es experimental-pragmática, apoyada para su desarrollo en softwares de simulación por elemento finito (NX Siemens) y simulación analítica (Matlab/Simulink), en los cuales se realizarán la simulación de comportamientos límite y posibles consecuencias. Para posteriormente integrar tecnologías de hardware y software, que contribuyan al control de la vibración.

El desarrollo se divide en las siguientes etapas:

1. Concepción de la idea
2. Determinación de las condiciones de riesgo
3. Selección de equipos materiales y dispositivos
4. Desarrollo de planos mecánicos
5. Desarrollo de planos eléctricos
6. Programación del Arduino
7. Programación del PLC
8. Integración de mediante de los dispositivos mediante una ampliación como Matlab
9. Monitoreo de prueba y asociación de fallas con eventualidades
10. Instalación y puesta en marcha

Investigación y métodos de solución

La investigación es experimental y el método de solución es el científico-deductivo-propositivo. El estudio de referencia involucra la definición de conceptos como vibración y resonancia (Martínez, Pino, Terán, & Arteaga, Vibraciones Mecánicas I, 2017), (Martínez,

Pino, Terán, & Arteaga, 2017). La organización del trabajo a desarrollar se estructura en el diagrama de bloques mostrado en la figura 6.

Productos Altech

Entre los productos Altech[4] a utilizar se encuentran:

- Relevadores de control
- Contactores
- Botones Pulsadores
- Gabinetes
- Clemas
- Rieles DIN

Y la incorporación de elementos no considerados durante la ejecución del dispositivo, como relevadores de estado sólido, fuentes de poder o módulos de interfaces.

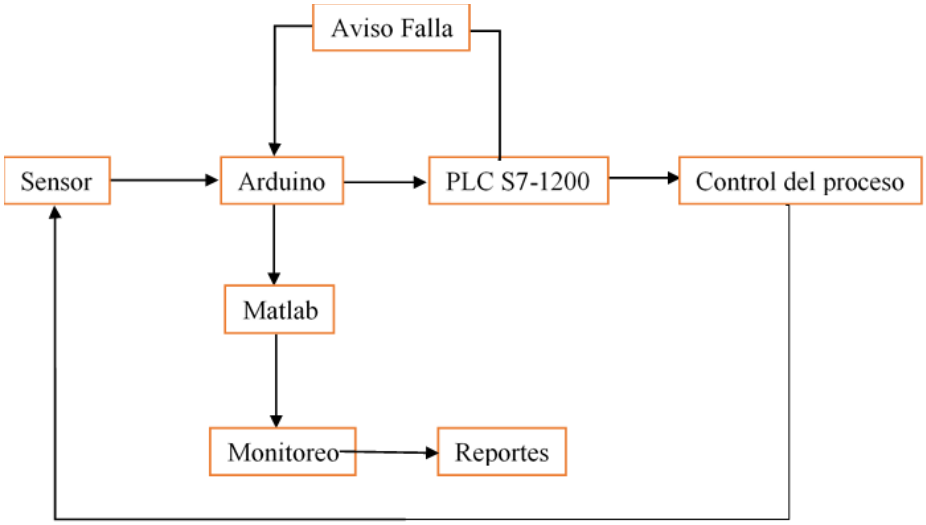


Figura 6. Diagrama de bloques del dispositivo a desarrollar.

[4] http://ecommerce.altechmexico.com/ecommerce/

Capítulo IV.- Desarrollo tecnológico

Diagramas eléctricos

- En proceso, la figura muestra el esquema de conexión de entre tecnologías.

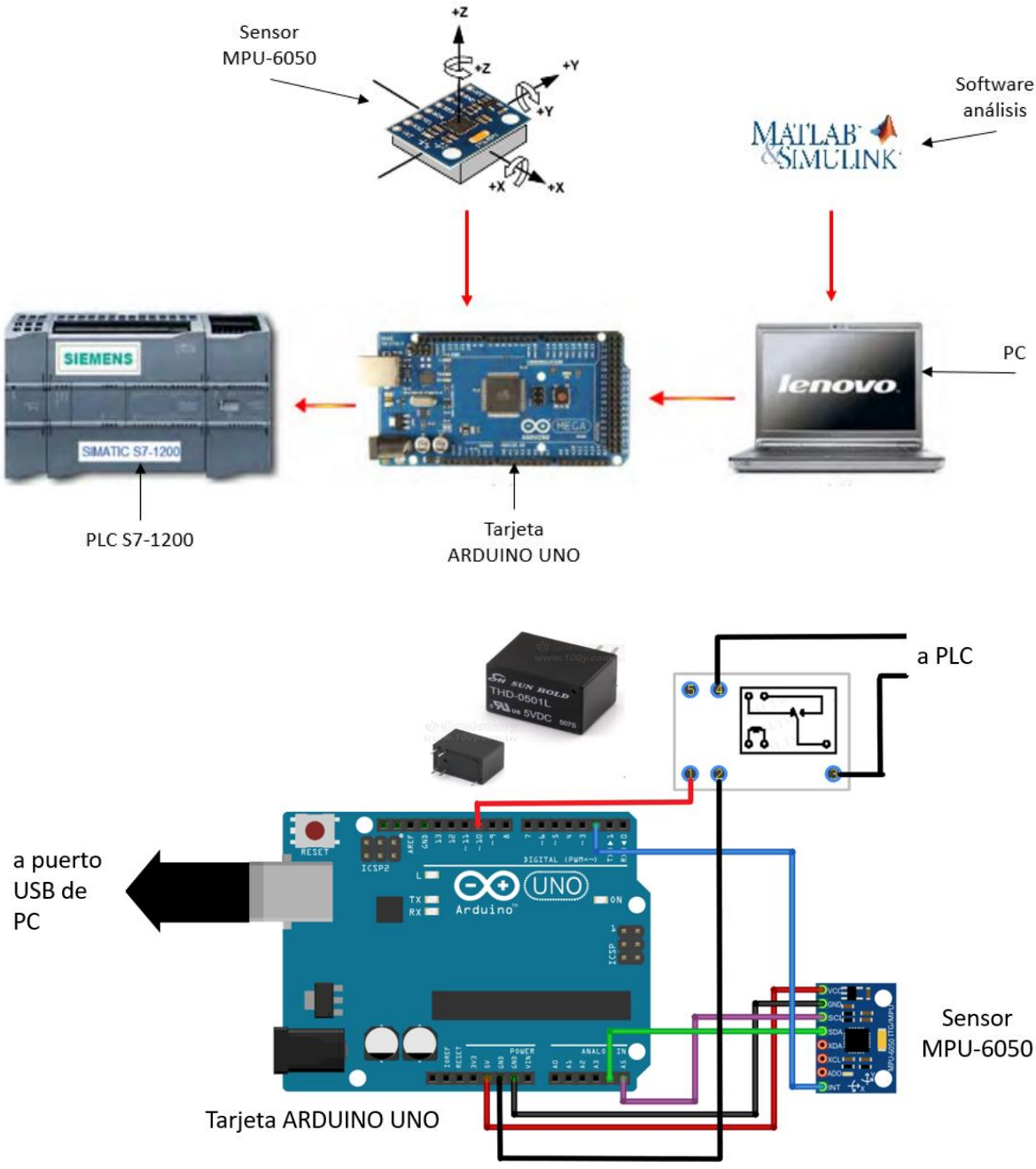


Figura 11.- Aproximación al esquema eléctrico.

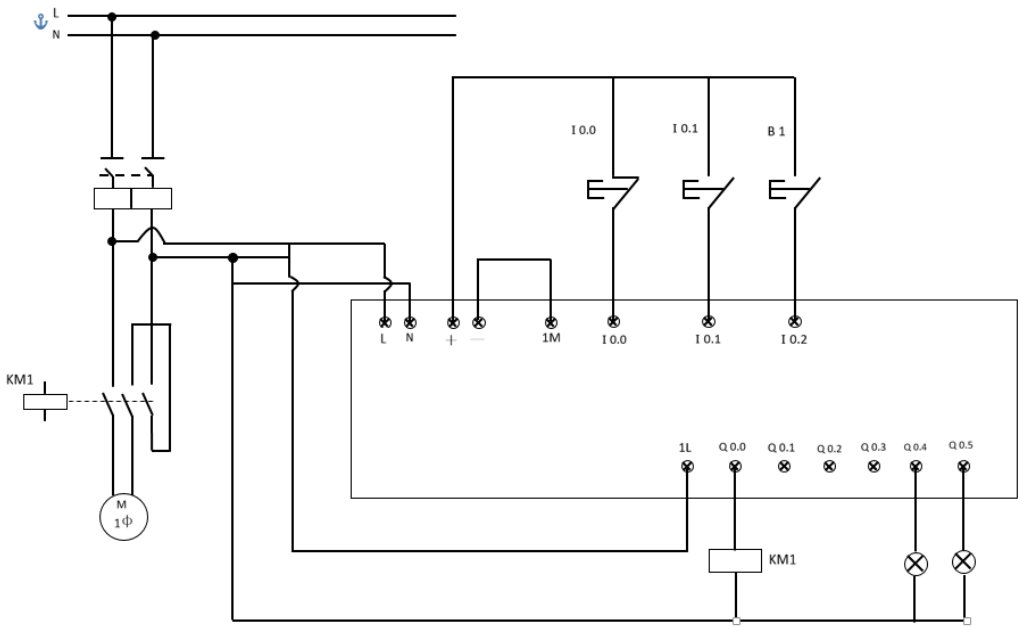


Figura 12. Diagrama de conexión de PLC S7-1200.

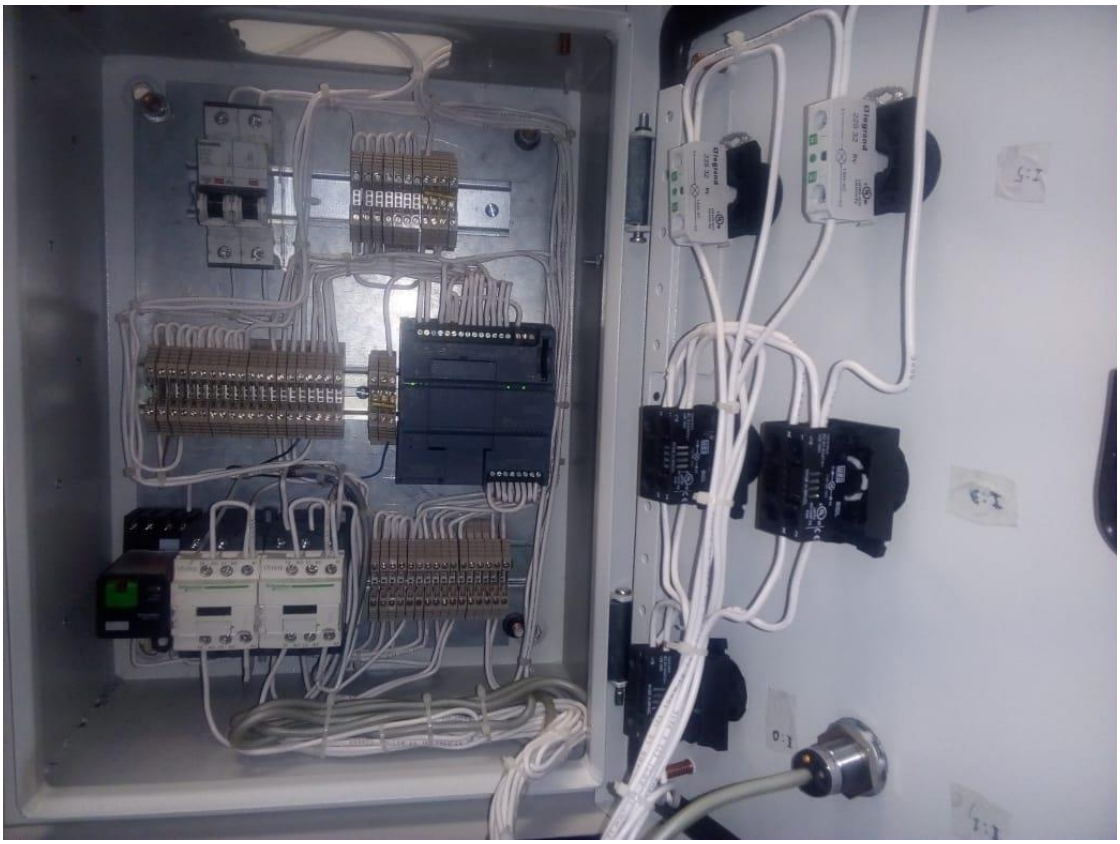


Figura 13. Representación conexiones de PLC en gabinete.

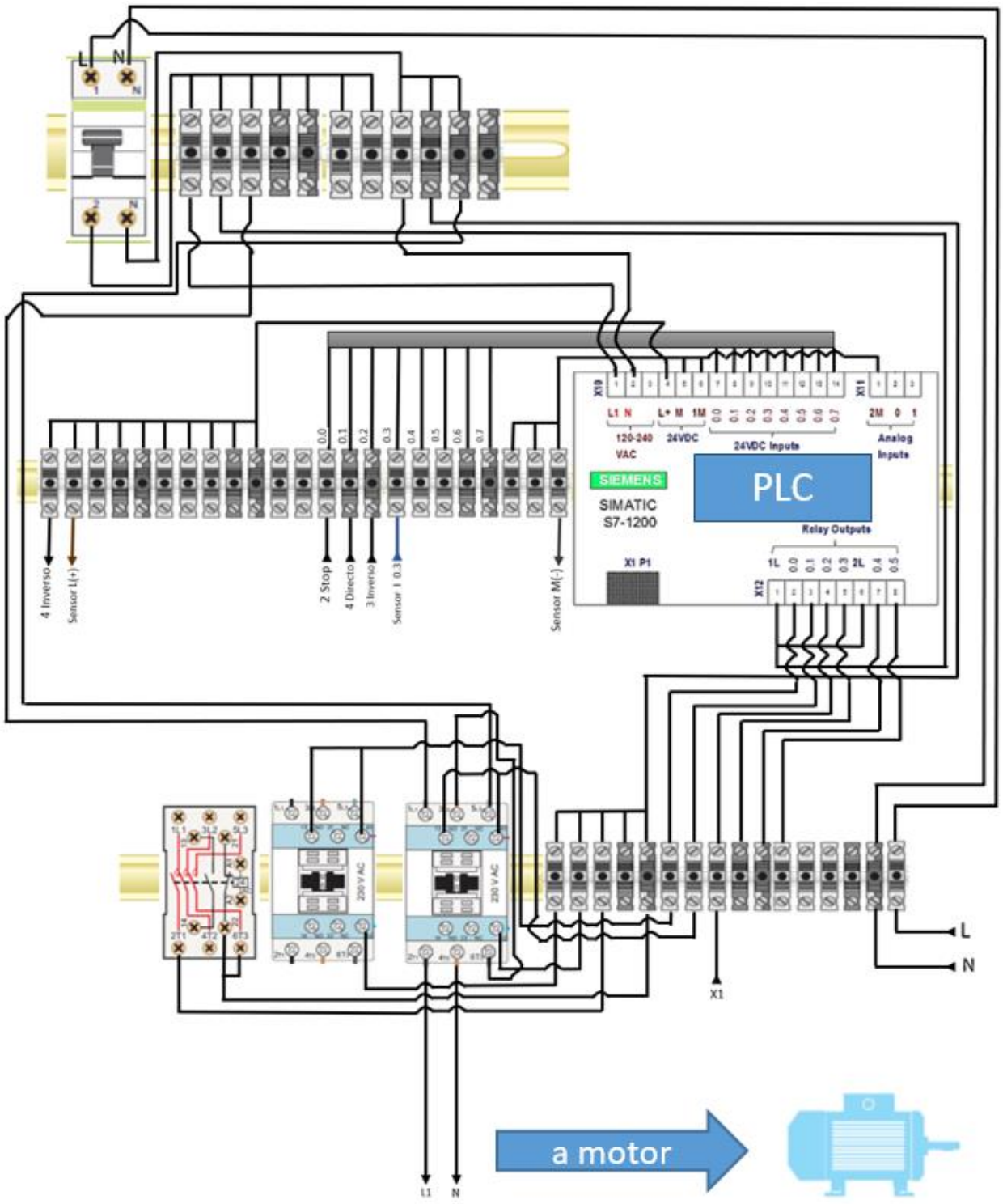


Figura 14. Representación de conexión de PLC y carga de la estructura.

Programación del PLC

El PLC de SIMATIC S7-1200 cuenta con memoria de trabajo 50 KB; Fuente de alimentación de 120 / 240VCA con DI8 x 24VDC SINK / SOURCE, relé DQ6 x y AI2 a bordo; 4 contadores de alta velocidad (ampliables con placa de señal digital) y 4 salidas de pulso a bordo; la placa de señal expande las E / S a bordo; hasta 3 módulos de comunicación para comunicación en serie; hasta 2 módulos de señal para expansión de E / S; 0.04 ms / 1000 instrucciones; Interfaz PROFINET para programación, HMI y comunicación PLC a PLC

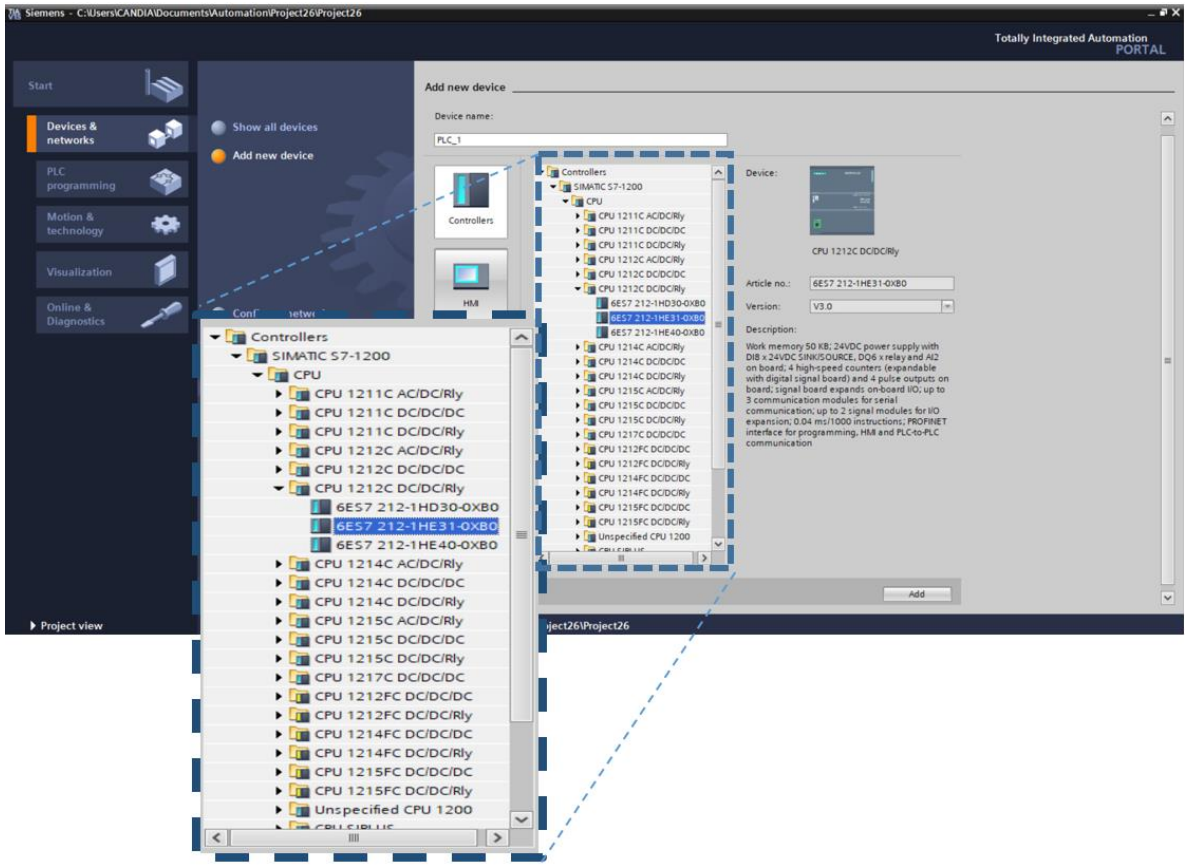


Figura 15. Interfaz de inicio para seleccionar el módulo PLC a programar.

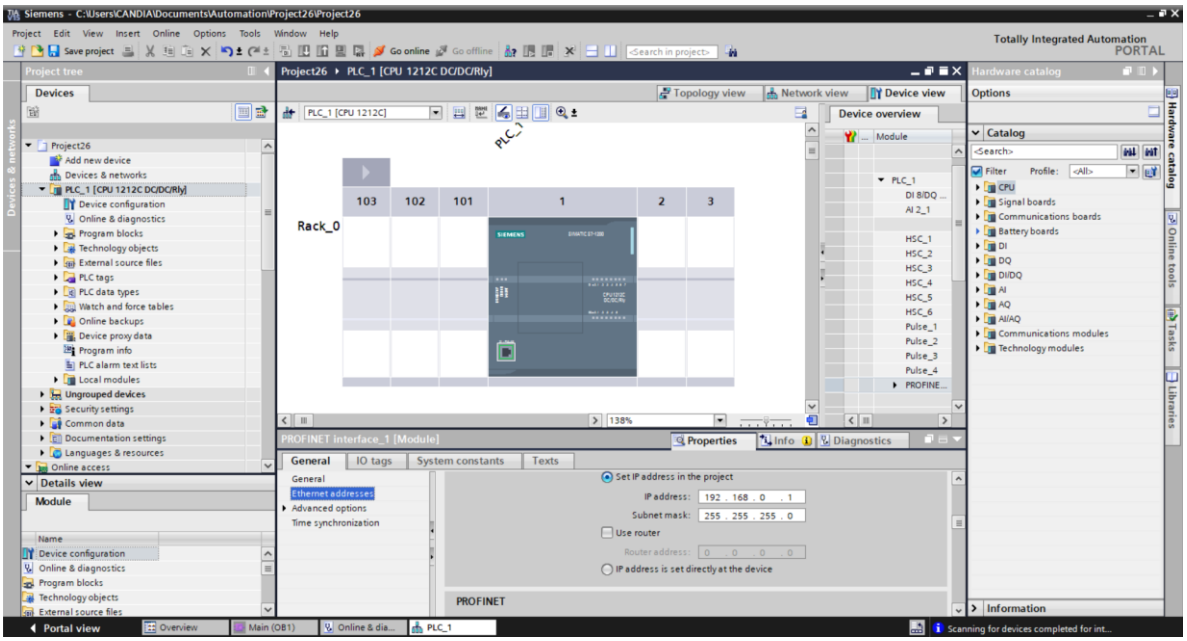


Figura 16. Configuración de módulo PLC SIMATIC S7-1200.

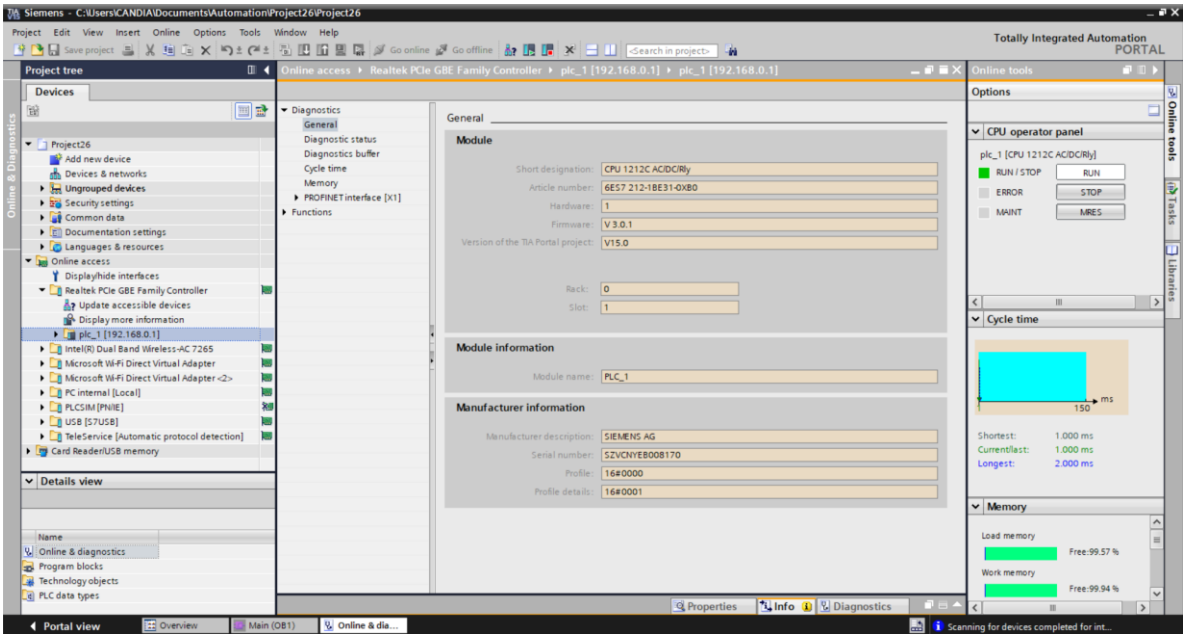


Figura 17. Información general de PLC SIMATIC S7-1200.

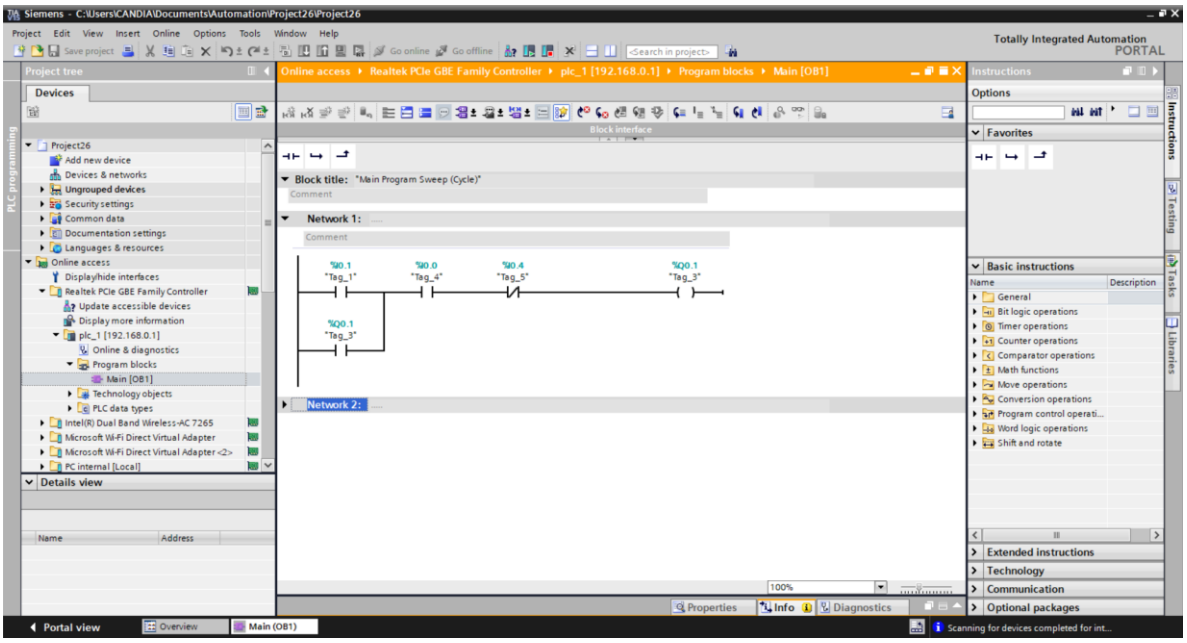


Figura 18. Programación de PLC SIMATIC S7-1200

Programación de placa Arduino UNO en MATLAB y Simulink

Para tener control en caso de detectarse una condición insegura de trabajo, así como para recolectar muestras de aceleración para monitorear el estado de la estructura, se realizó un script que automatizara las tareas de inicio de la simulación, recolección de muestras del acelerómetro MPU-6050, análisis de las muestras por la transformada rápida de Fourier (FFT), determinar los picos de amplitud máxima en G y mm/s^2 y la frecuencia en Hz donde se presentan dichas amplitudes.

Se utilizaron 2 scripts, el primero tiene la función de realizar las gráficas en función de los valores resultantes de la transformada rápida de Fourier (FFT) e indicar los valores máximos y promedio de amplitud en G y mm/s^2, el segundo se encarga del inicio de la simulación, finalización de la misma, realizar el procesamiento de las señales mediante FFT, obtener los valores máximos de desplazamiento en mm detectados en cada eje (x, y, z) y determinar las variables a graficar por el primer script.

Script 1

```
function createfigure(X1, Y1)
%CREATEFIGURE(X1, Y1)
%  X1:  vector of x data
%  Y1:  vector of y data

% Create figure
figure1 = figure;

% Create axes
axes1 = axes('Parent',figure1);
hold(axes1,'on');

% Create plot
plot1 = plot(X1,Y1,'DisplayName','data1');

% Get xdata from plot for data statistics
xdata1 = get(plot1, 'xdata');
% Get ydata from plot for data statistics
ydata1 = get(plot1, 'ydata');
% Make sure data are column vectors
xdata1 = xdata1(:);
ydata1 = ydata1(:);


% Get axes ylim
axXLim1 = get(axes1,'xlim');

% Find the max
ymax1 = max(ydata1);
% Get coordinates for the max line
maxValue1 = [ymax1 ymax1];
% Plot the max
statLine1 = plot(axXLim1,maxValue1,'DisplayName','   y max','Tag','max y',...
    'Parent',axes1,...
    'LineStyle','-.',...
    'Color',[0 0 1]);

% Set new line in proper position
setLineOrder(axes1,statLine1,plot1);

% Find the mean
ymean1 = mean(ydata1);
% Get coordinates for the mean line
meanValue1 = [ymean1 ymean1];
% Plot the mean
statLine2 = plot(axXLim1,meanValue1,'DisplayName','   y mean',...
```

```
    'Tag','mean y',...
    'Parent',axes1,...
    'LineStyle','-.',...
    'Color',[0 0.5 0]);

% Set new line in proper position
setLineOrder(axes1,statLine2,plot1);

% Create ylabel
ylabel('Accel (g)');

% Create xlabel
xlabel('Frequency (Hz)');

% Create title
title('eje x');

% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 60]);
% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes1,[0 0.35]);
box(axes1,'on');
grid(axes1,'on');
% Create legend
legend(axes1,'show');

%-------------------------------------------------------------%
function setLineOrder(axesh1, newLine1, associatedLine1)
%SETLINEORDER(AXESH1,NEWLINE1,ASSOCIATEDLINE1)
%  Set line order
%  AXESH1:  axes
%  NEWLINE1:  new line
%  ASSOCIATEDLINE1:  associated line

% Get the axes children
hChildren = get(axesh1,'Children');
% Remove the new line
hChildren(hChildren==newLine1) = [];
% Get the index to the associatedLine
lineIndex = find(hChildren==associatedLine1);
% Reorder lines so the new line appears with associated data
hNewChildren = [hChildren(1:lineIndex-1);newLine1;hChildren(lineIndex:end)];
% Set the children:
set(axesh1,'Children',hNewChildren);
```

Script 2

```
%Inicio de simulación en simulink
```

```
%Especifique tiempo de simulación en segundos
set_param('sensor_yz','Simulationcommand','start')
set_param('sensor_yz','Simulationcommand','stop','StopTime','30')

%Establecer un tiempo de pausa del script mayor al del tiempo de la
%simulación, por ejemplo si se establecen 20 segundos de simulación,
%ponemos 35 segundos de pausa.
pause(35)

%Gráfica de recolección de señales del sensor sin implementación de fft
figure()
plot(acxyz.signals.values)
title('Lecturas de sensor MPU-6050 sin fft')
xlabel('tiempo (ms)')
ylabel('Aceleración (G)')
grid on

x=acxyz.signals.values
t=acxyz.time(:,1) %Establecer el tiempo de la muestra para fft
Fs=1/(t(2)-t(1)) %obtener la frecuencia de muestreo para f  1   ft
[N,m]=size(acxyz.signals.values)
Fs=1/(t(2)-t(1)) %obtener la frecuencia de muestreo para fft
fprintf('%12.0f data points\n',N)
freq = 0:Fs/length(x):Fs/2; %frequency array for FFT
xdft=fft(x)
xdft = 1/length(x).*xdft; %Normalize
xdft(2:end-1) = 2*xdft(2:end-1);


%Gráfica de fft para eje x
ejx=acxyz.signals.values(:,1)
tx=acxyz.time(:,1) %Establecer el tiempo de la muestra para fft
Fsx=1/(tx(2)-tx(1)) %obtener la frecuencia de muestreo para fft
[N,m]=size(acxyz.signals.values(:,1))
fprintf('%12.0f data points\n',N)
freqx = 0:Fsx/length(ejx):Fsx/2; %frequency array for FFT
ejxdft=fft(ejx)
ejxdft = 1/length(ejx).*ejxdft; %Normalize
ejxdft(2:end-1) = 2*ejxdft(2:end-1);

figure()
grafft(freqx,abs(ejxdft(1:floor(N/2)+1)))
xlabel('Frequency (Hz)');
ylabel('Accel (g)');
title('eje x');

grid on
```

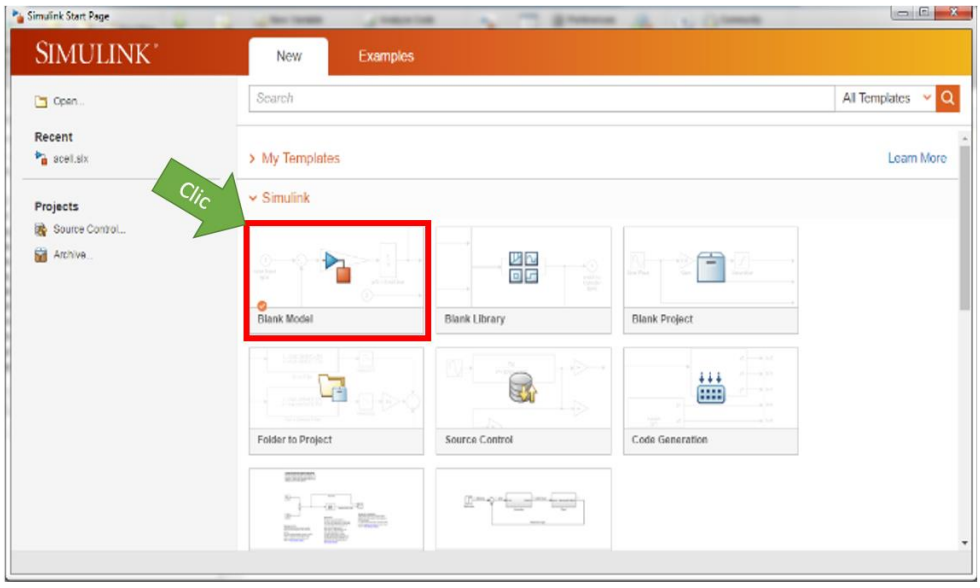
```
%Gráfica de fft para eje Y
ejy=acxyz.signals.values(:,2)
ty=acxyz.time(:,1) %Establecer el tiempo de la muestra para fft
Fsy=1/(ty(2)-ty(1)) %obtener la frecuencia de muestreo para fft
[N,m]=size(acxyz.signals.values(:,1))
fprintf('%12.0f data points\n',N)
freqy = 0:Fsy/length(ejy):Fsy/2; %frequency array for FFT
ejydft=fft(ejy)
ejydft = 1/length(ejy).*ejydft; %Normalize
ejydft(2:end-1) = 2*ejydft(2:end-1);
figure()
grafft(freqy,abs(ejydft(1:floor(N/2)+1)))
xlabel('Frequency (Hz)');
ylabel('Accel (g)');
title('eje y')
grid on

%Gráfica de fft para eje z
ejz=acxyz.signals.values(:,3)
tz=acxyz.time(:,1) %Establecer el tiempo de la muestra para fft
Fsz=1/(tz(2)-tz(1)) %obtener la frecuencia de muestreo para fft
[N,m]=size(acxyz.signals.values(:,1))
fprintf('%12.0f data points\n',N)
freqz = 0:Fsz/length(ejz):Fsz/2; %frequency array for FFT
ejzdft=fft(ejz);
ejzdft = 1/length(ejz).*ejzdft; %Normalize
ejzdft(2:end-1) = 2*ejzdft(2:end-1);

figure()
grafft(freqz,abs(ejzdft(1:floor(N/2)+1)))
xlabel('Frequency (Hz)');
ylabel('Accel (g)');
title('eje z')
grid on

%-----------------------------------------------------------%
%conversión de valores de G a mm
convgmm=(acxyz.signals.values)*9810
%Gráfica de recolección de señales del sensor sin implementación de fft (en
%mm/s^2)

figure()
plot(convgmm)
title('Lecturas de sensor MPU-6050 sin fft')
xlabel('tiempo (ms)')
ylabel('Aceleración (mm/s^2)')
```

```
grid on

xmm=convgmm
t=acxyz.time(:,1) %Establecer el tiempo de la muestra para fft
Fs=1/(t(2)-t(1)) %obtener la frecuencia de muestreo para f  1   ft
[N,m]=size(acxyz.signals.values)
Fs=1/(t(2)-t(1)) %obtener la frecuencia de muestreo para fft
fprintf('%12.0f data points\n',N)
freq = 0:Fs/length(xmm):Fs/2; %frequency array for FFT
xmmdft=fft(xmm)
xmmdft = 1/length(xmm).*xmmdft; %Normalize
xmmdft(2:end-1) = 2*xmmdft(2:end-1);


%Gráfica de fft para eje x (en mm/s^2)
ejxmm=convgmm(:,1)
txmm=acxyz.time(:,1) %Establecer el tiempo de la muestra para fft
Fsxmm=1/(txmm(2)-txmm(1)) %obtener la frecuencia de muestreo para fft
[N,m]=size(acxyz.signals.values(:,1))
fprintf('%12.0f data points\n',N)
freqxmm = 0:Fsxmm/length(ejxmm):Fsxmm/2; %frequency array for FFT
ejxdftmm=fft(ejxmm)
ejxdftmm = 1/length(ejxmm).*ejxdftmm; %Normalize
ejxdftmm(2:end-1) = 2*ejxdftmm(2:end-1);

figure()
grafft(freqxmm,abs(ejxdftmm(1:floor(N/2)+1)))
xlabel('Frequency (Hz)');
ylabel('Accel (mm/s^2)');
title('eje x');

grid on

%Gráfica de fft para eje Y (en mm/s^2)
ejymm=convgmm(:,2)
tymm=acxyz.time(:,1) %Establecer el tiempo de la muestra para fft
Fsymm=1/(tymm(2)-tymm(1)) %obtener la frecuencia de muestreo para fft
[N,m]=size(acxyz.signals.values(:,1))
fprintf('%12.0f data points\n',N)
freqymm = 0:Fsymm/length(ejymm):Fsymm/2; %frequency array for FFT
ejydftmm=fft(ejymm)
ejydftmm = 1/length(ejymm).*ejydftmm; %Normalize
ejydftmm(2:end-1) = 2*ejydftmm(2:end-1);

figure()
grafft(freqymm,abs(ejydftmm(1:floor(N/2)+1)))
xlabel('Frequency (Hz)');
```

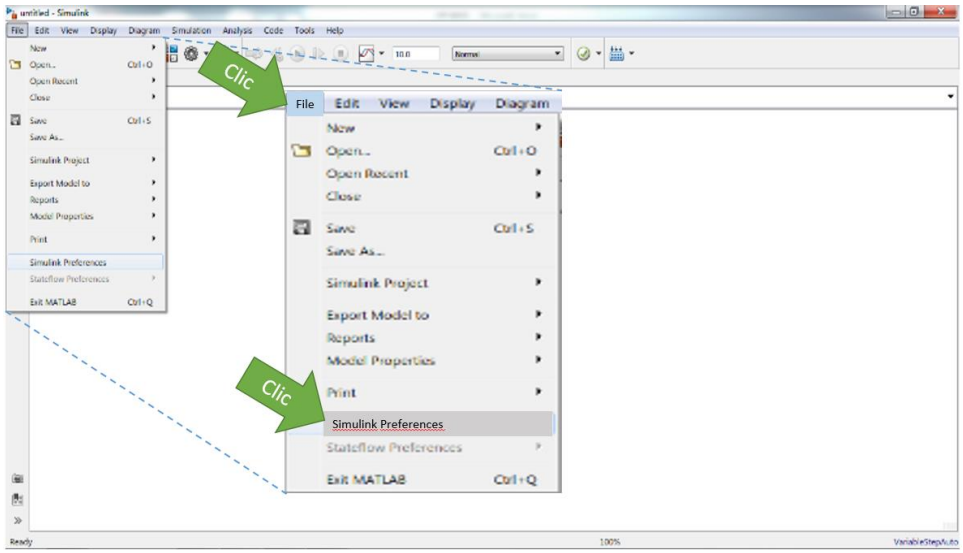
```
ylabel('Accel (mm/s^2)');
title('eje y')
grid on

%Gráfica de fft para eje z (en %mm/s^2)
ejzmm=convgmm(:,3)
tzmm=acxyz.time(:,1) %Establecer el tiempo de la muestra para fft
Fszmm=1/(tzmm(2)-tzmm(1)) %obtener la frecuencia de muestreo para fft
[N,m]=size(acxyz.signals.values(:,1))
fprintf('%12.0f data points\n',N)
freqzmm = 0:Fszmm/length(ejzmm):Fszmm/2; %frequency array for FFT
ejzdftmm=fft(ejzmm);
ejzdftmm = 1/length(ejzmm).*ejzdftmm; %Normalize
ejzdftmm(2:end-1) = 2*ejzdftmm(2:end-1);

figure()
grafft(freqzmm,abs(ejzdftmm(1:floor(N/2)+1)))
xlabel('Frequency (Hz)');
ylabel('Accel (mm/s^2)');
title('eje z')
grid on

%integrales para conocer el desplazamiento en mm
% los valores de desplazamiento se guardan en las variables del workspace
% de MATLAB, de x, y y z en las variables smmx, smmy y smmz
% respectivamente.

%obtener máximos de aceleración
mxmm=max(convgmm)
%declarar función para realizar la integral y conocer el desplazamiento en
%x en mm
fun1=@(tm) (mxmm(1,1)*tm)
%Declarar la variable con la que se guardará el desplazamiento en x en mm
smmx=integral(fun1,0,0.01)
%declarar función para realizar la integral y conocer el desplazamiento en
%y en mm
fun2=@(tm) (mxmm(1,2)*tm)
%Declarar la variable con la que se guardará el desplazamiento en y en mm
smmy=integral(fun2,0,0.01)
%declarar función para realizar la integral y conocer el desplazamiento en
%z en mm
fun3=@(tm) (mxmm(1,3)*tm)
%Declarar la variable con la que se guardará el desplazamiento en z en mm
smmz=integral(fun3,0,0.01)
```

Con los scripts listos para realizar el procesamiento de las señales, se desarrolla el modelo en Simulink para establecer la comunicación entre el sensor MPU6050, la placa Arduino UNO, y la interfaz para el procesamiento de las señales, el cual consta de las siguientes operaciones para su creación:

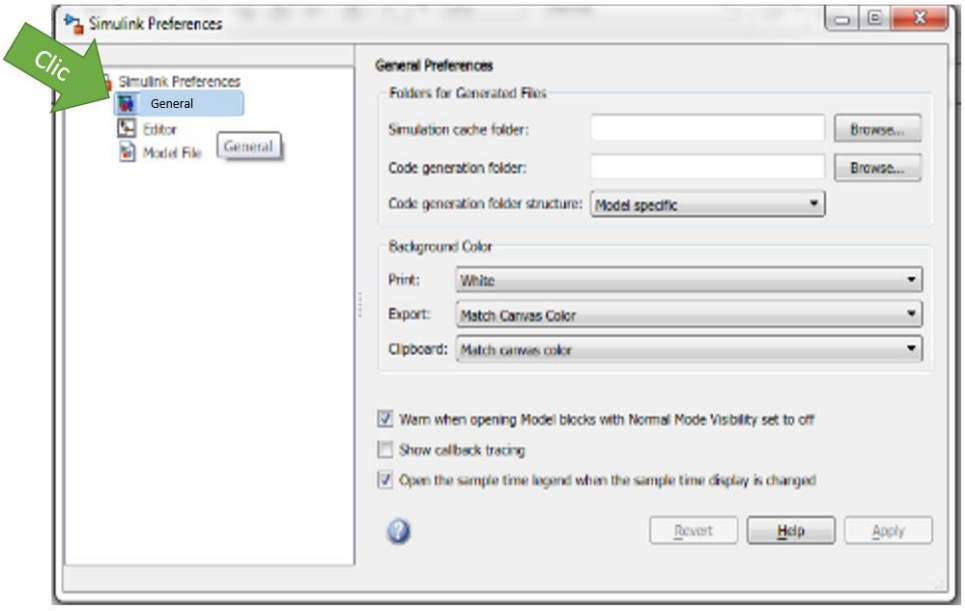
1.En la pantalla principal de Simulink hacer click en Blank Model



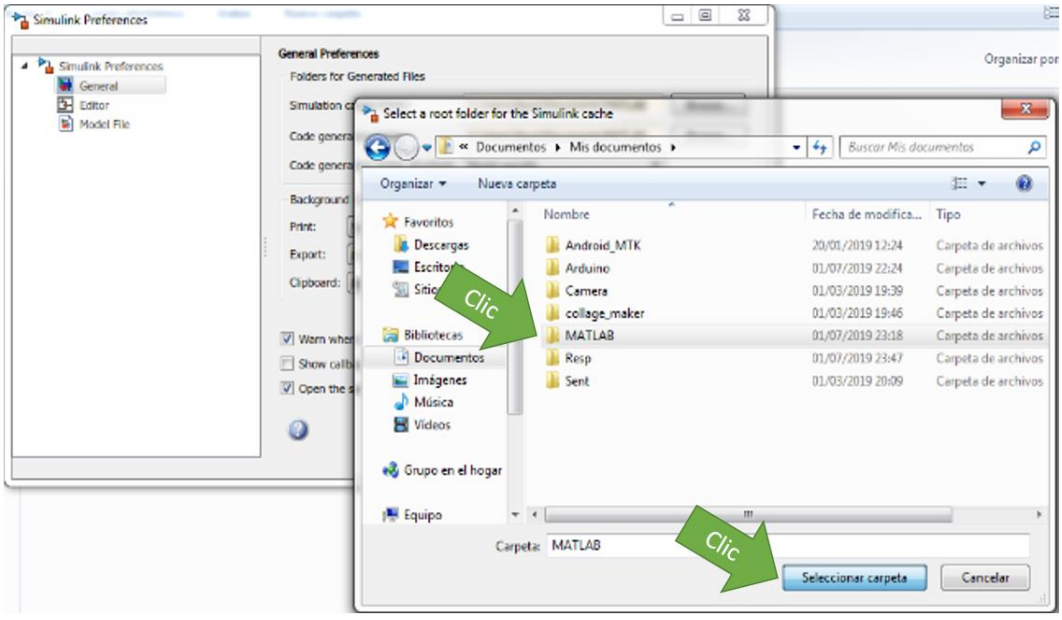
2. Una vez abierto Simulink, dar clic en la pestaña File>Simulink Preferences, para configurar dos carpetas en las que se guarden temporalmente los archivos necesarios para las simulaciones y evitar errores al intentar compilar el programa.



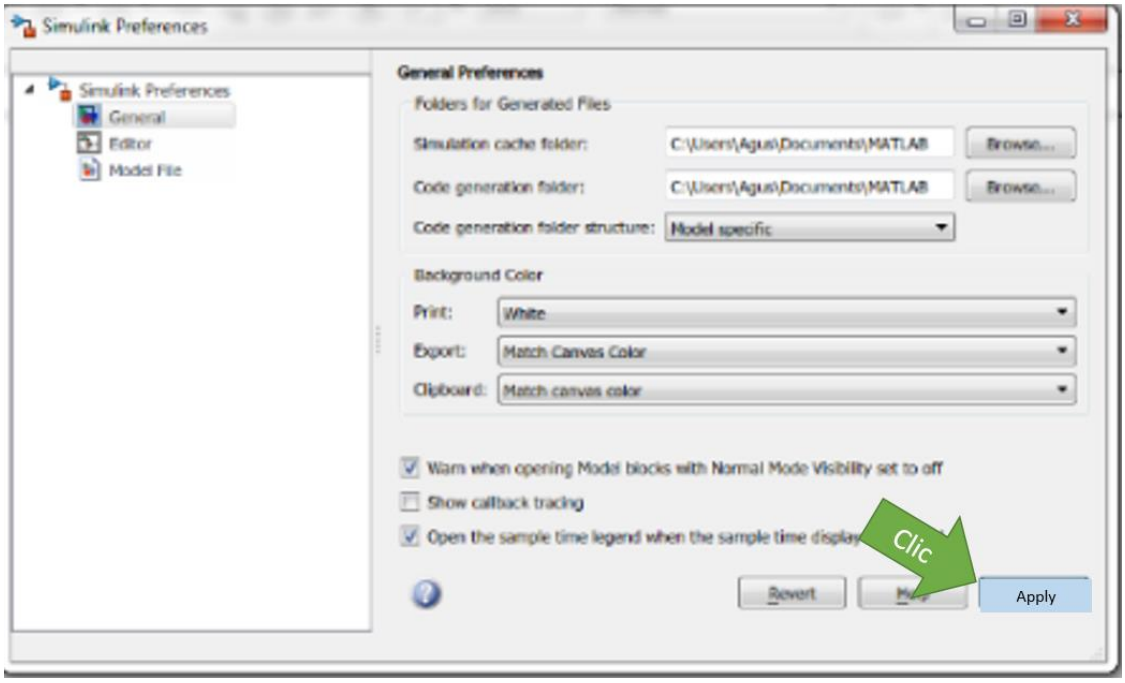
3. En la ventana que se abre dar clic en General.



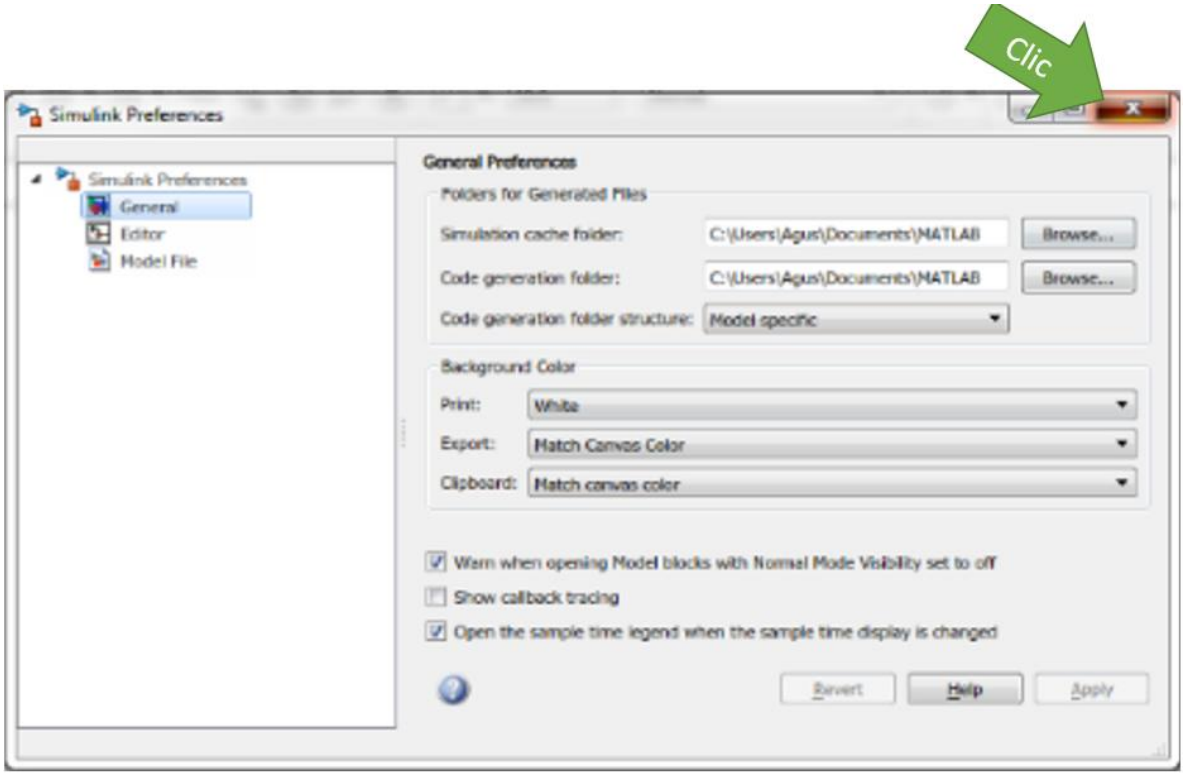
4. En General Preferences, cambiar los apartados “Simulation cache folder” y “Code generation folder”, especificando una carpeta existente que no se encuentre directamente en el disco duro o en el directorio raíz de MATLAB, puede seleccionar una carpeta existente en “Mis documentos” o crear una nueva.



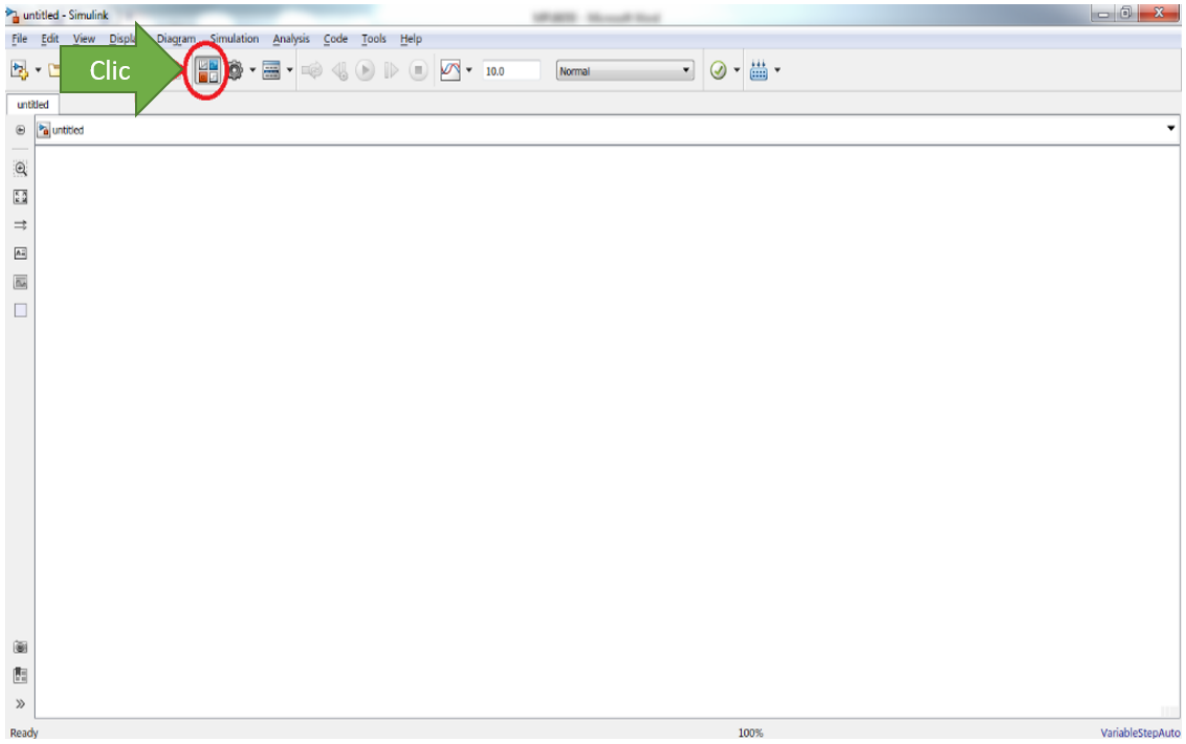
5. Dar click en Apply



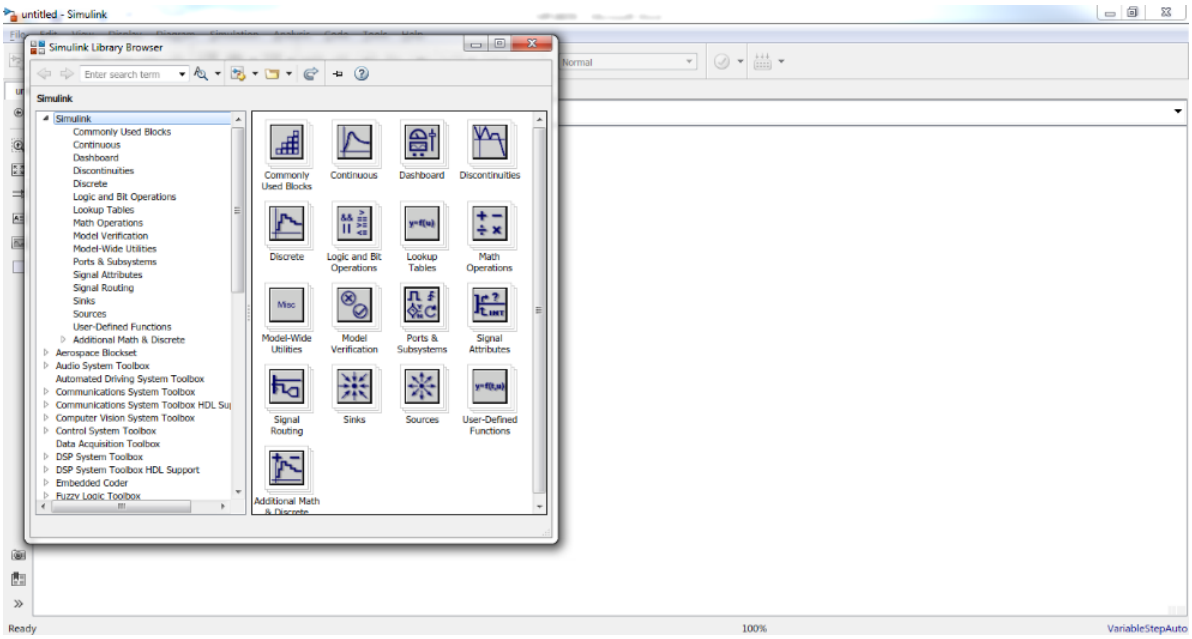
6. Cerrar la ventana Simulink Preferences.



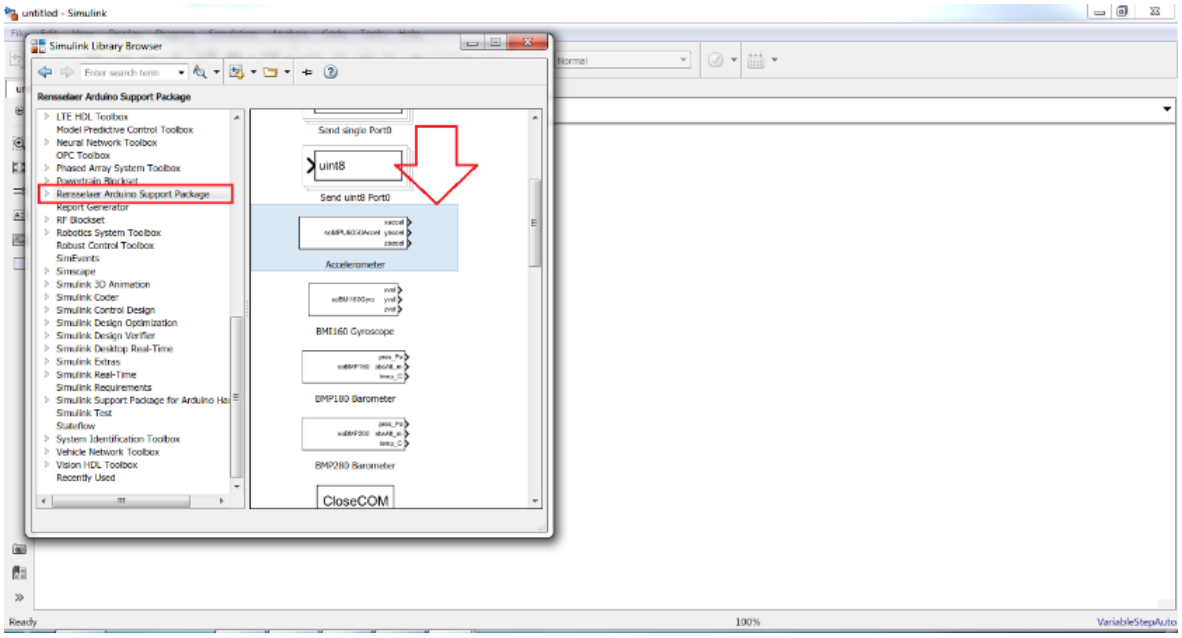
7. Para armar el modelo del programa, necesario para trabajar con el sensor, abrir la librería de Simulink



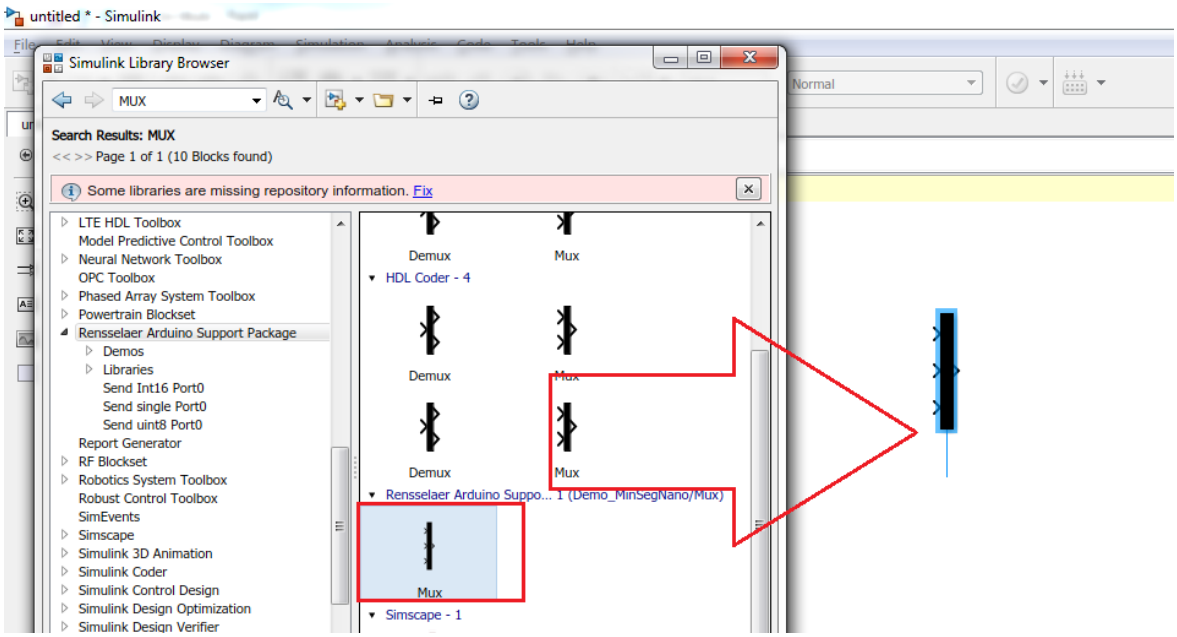
8. Se abre la siguiente ventana



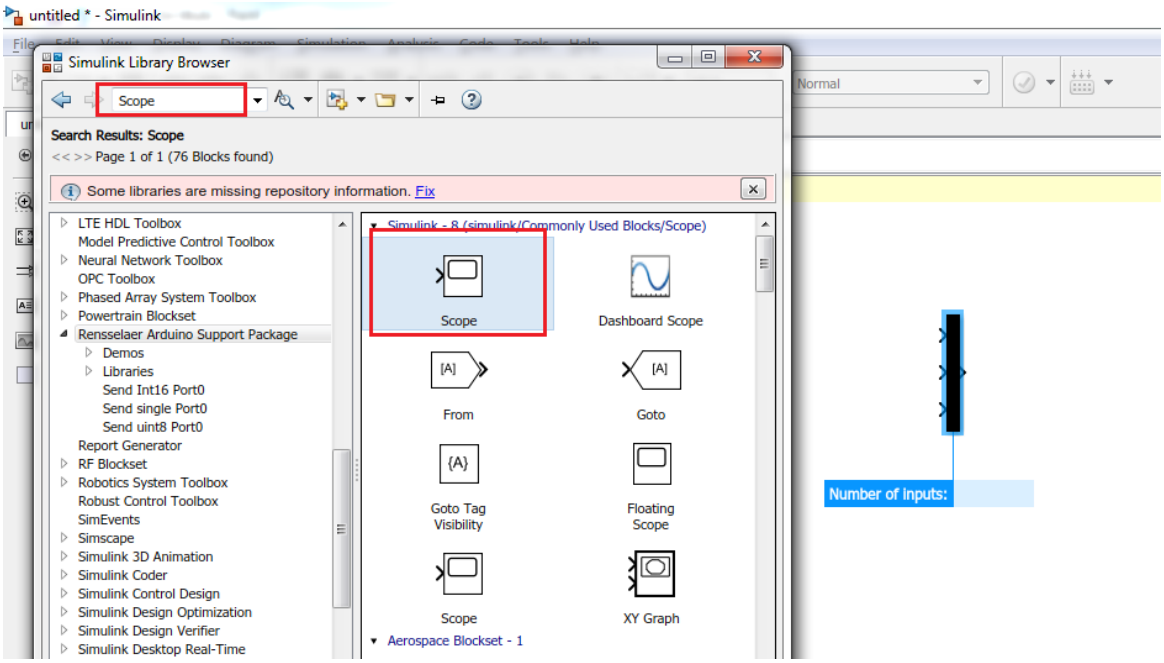
9. Buscar el apartado Rensselaer Arduino Support Package y seleccionar el bloque del acelerómetro correspondiente al sensor MPU6050 y arrastrar al entorno de trabajo de Simulink.



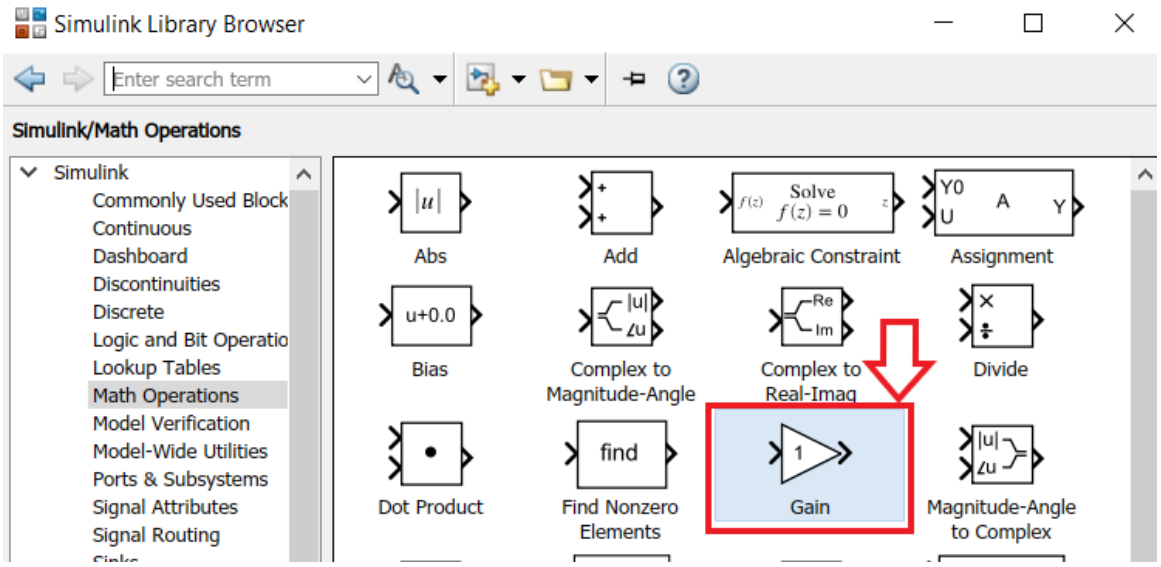
10. Buscar el bloque MUX dentro de la librería de simulink, con tres entradas y una salida para conectar los 3 ejes del acelerómetro y posteriormente enviar los valores que detecte el sensor al entorno de trabajo de MATLAB. Arrastrar el bloque MUX al entorno de trabajo de Simulink.



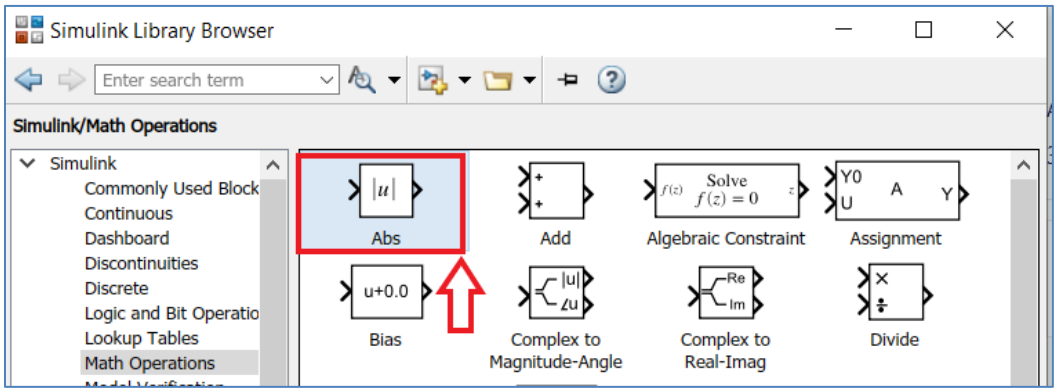
11. Para visualizar en tiempo real las lecturas del sensor MPU-6050 buscar el bloque Scope y arrastrarlo al entorno de trabajo de Simulink.

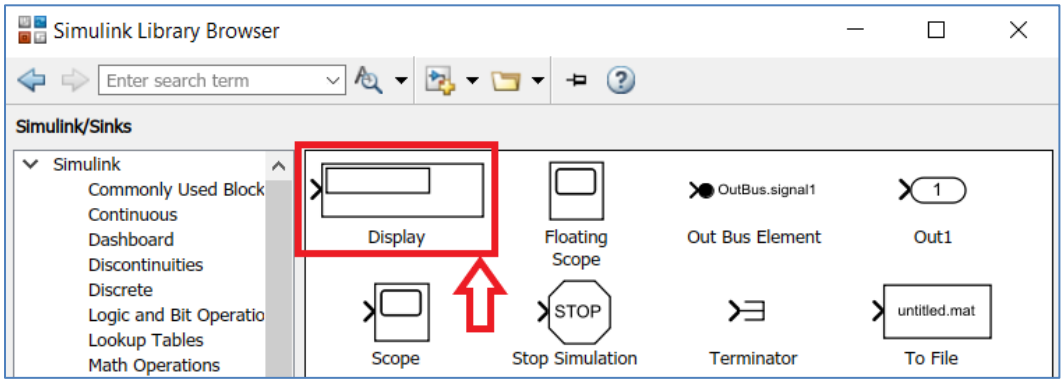


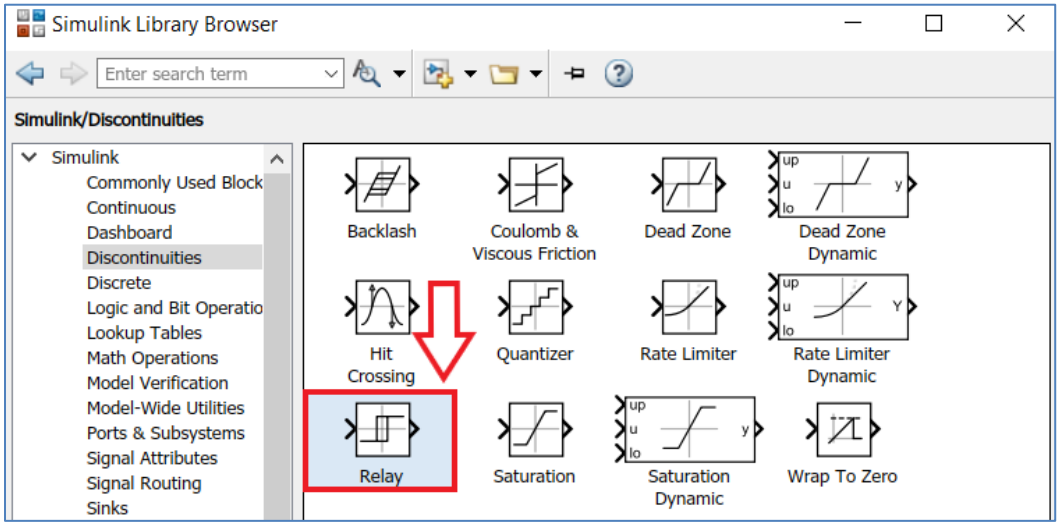
12. Buscar el bloque “Gain” para realizar la conversión de valores RAW captados por el sensor a valores en “G” para medir la aceleración.

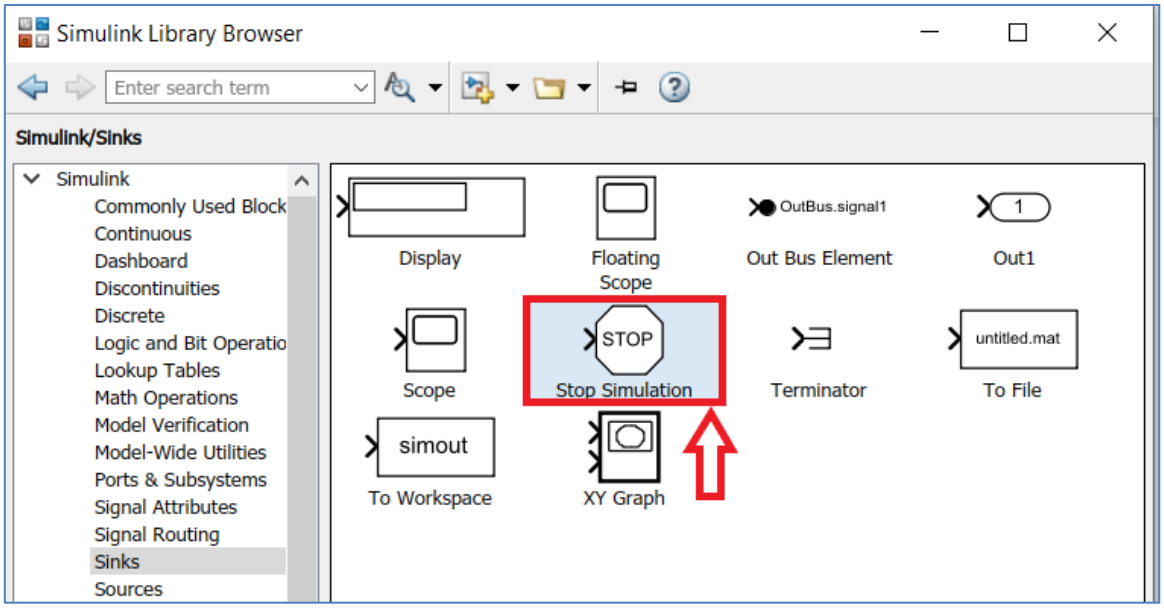


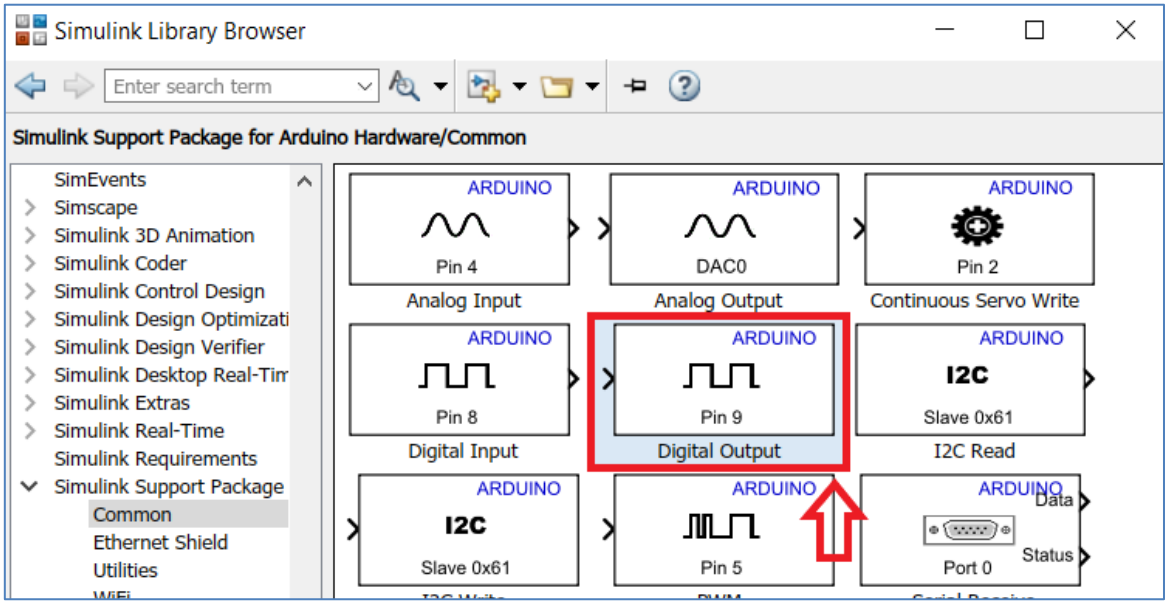
13. Para realizar el apagado del motor cuando se detecte un valor pico de aceleración por el sensor, se requiere de los bloques “Relay”, “Abs”, “Display”, “Stop simulation”, “Digital Output” y “Display”, este último permite visualizar en el espacio de trabajo de Simulink cuando el valor pico es detectado y se envía un pulso a la placa Arduino para que el motor se detenga.



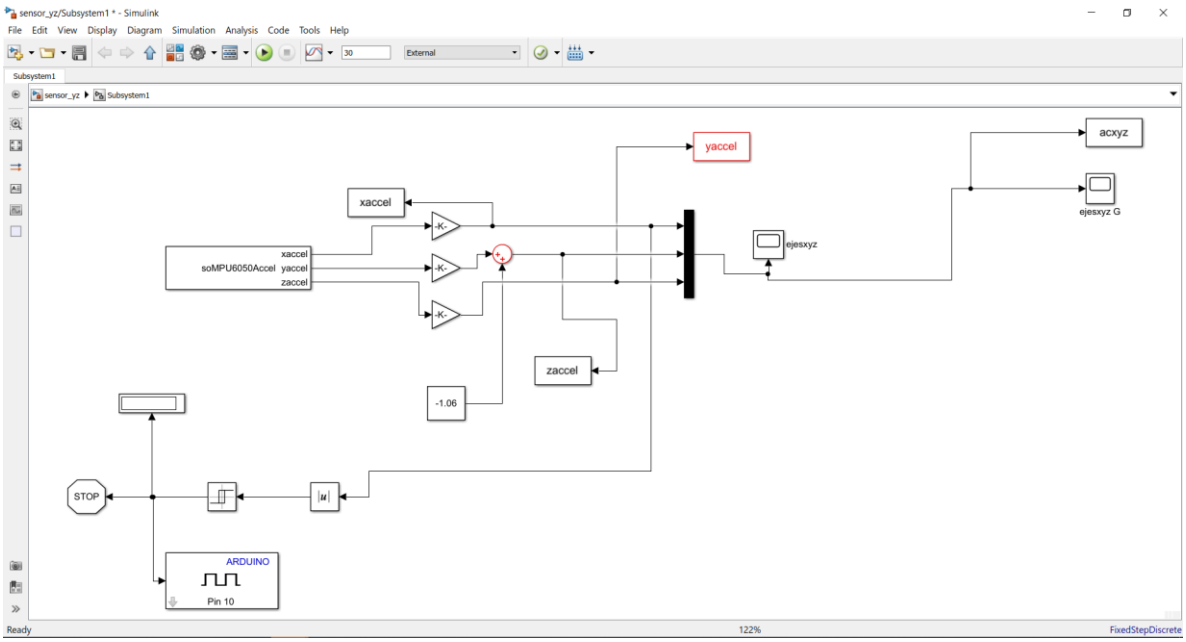





Simulink Library Browser
Enter search term
Simulink/Sinks
Simulink
Commonly Used Block
Continuous
Dashboard
Discontinuities
Discrete
Logic and Bit Operatio
Lookup Tables
Math Operations
Model Verification
Model-Wide Utilities
Ports & Subsystems
Signal Attributes
Signal Routing
Sinks
Sources
Display
Floating Scope
OutBus.signal1
Out Bus Element
1
Out1
Scope
STOP
Stop Simulation
Terminator
untitled.mat
To File
simout
To Workspace
XY Graph

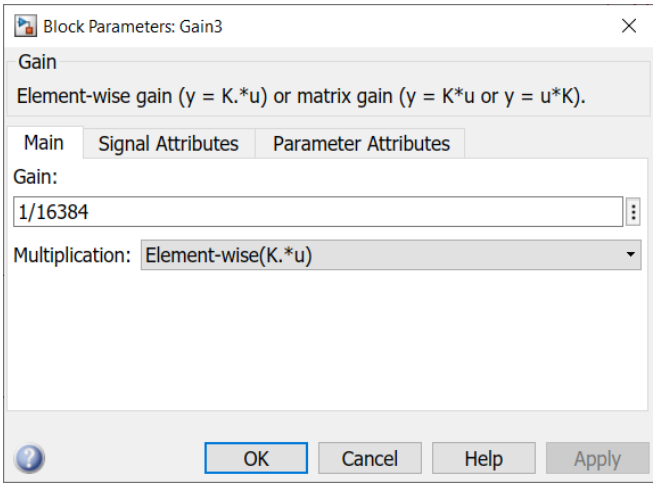

Simulink Library Browser
Enter search term
Simulink Support Package for Arduino Hardware/Common
SimEvents
Simscape
Simulink 3D Animation
Simulink Coder
Simulink Control Design
Simulink Design Optimizati
Simulink Design Verifier
Simulink Desktop Real-Tim
Simulink Extras
Simulink Real-Time
Simulink Requirements
Simulink Support Package
Common
Ethernet Shield
Utilities
ARDUINO
Pin 4
Analog Input
DAC0
Analog Output
Pin 2
Continuous Servo Write
Pin 8
Digital Input
Pin 9
Digital Output
I2C
Slave 0x61
I2C Read
Slave 0x61
Pin 5
Port 0
Data
Status

14. Conectar los elementos del modelo para poder realizar la simulación. Posteriormente se configura el modelo para que se simule utilizando la placa Arduino y el sensor MPU6050.

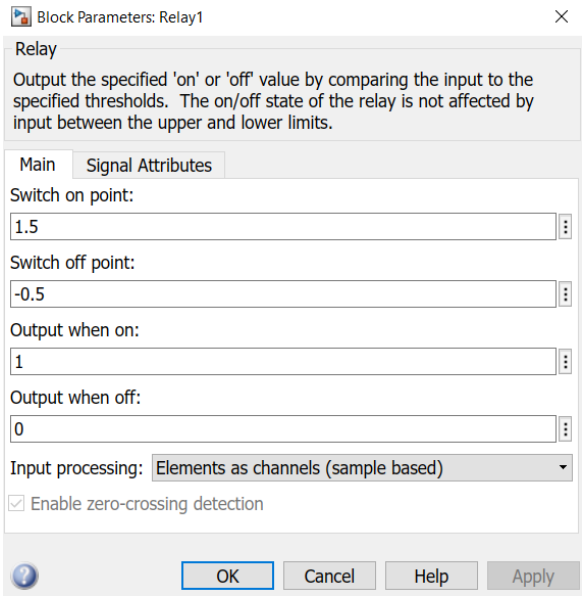


15. Los valores a configurar son las variables a las que se enviarán los datos captados por el sensor, las operaciones para apagar el motor cuando se detecte un pico y la conversión de datos RAW a G para medir la aceleración.

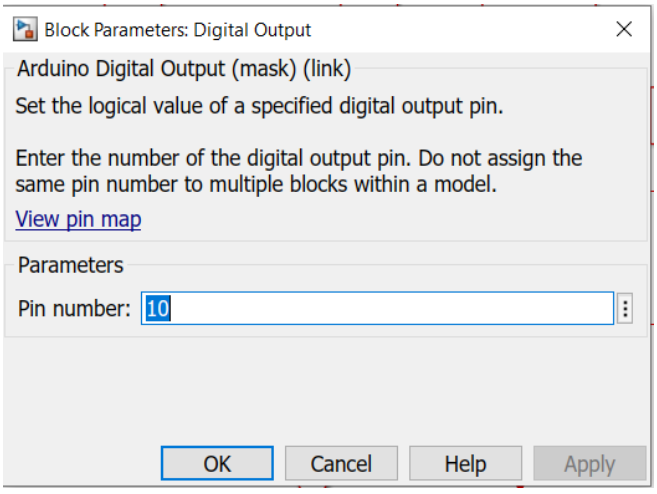
    Para configurar los valores se da doble click en los bloques y se establecen los parámetros requeridos para la simulación. Sólo se configuran los valores mostrados en las capturas, los demás se dejan a como están por defecto.



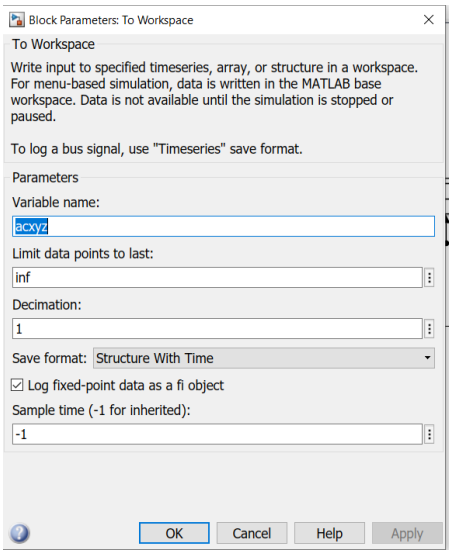
La operación indicada es para realizar la conversión de datos RAW a G para la aceleración.



Este bloque es el que dará la condición para enviar la señal al relevador THD-0501L para apagar el motor. Se establece el intervalo de funcionamiento de acuerdo a los valores de operación normal del motor, para que cuando el valor detectado por el sensor sea mayor a estos, se detenga la simulación y el motor (El rango establecido para el funcionamiento del sensor es de ±2 G).

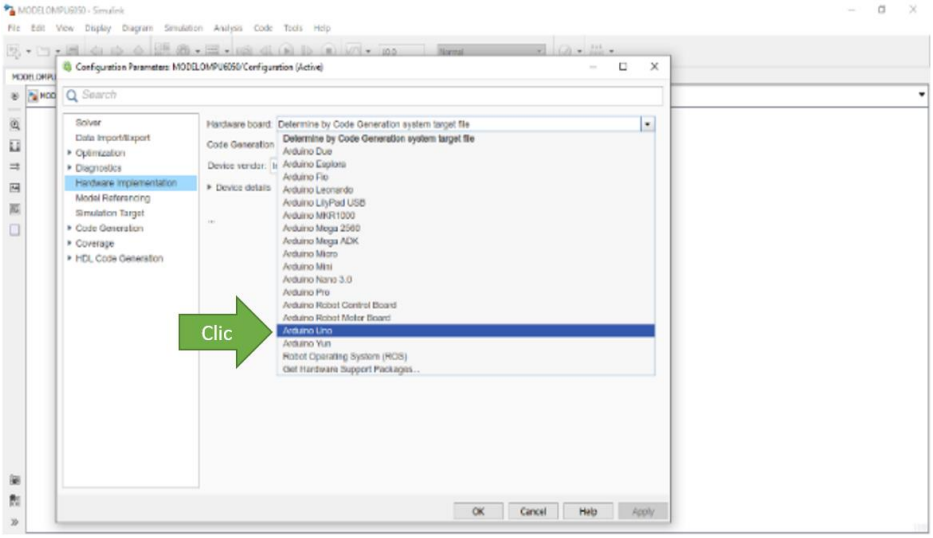


Establecer el pin donde se enviará el pulso para apagar el sensor, en este caso se elige el 10 pero es posible utilizar cualquier otro de los que estén disponibles para dicha operación, los cuales se pueden ver en la opción “View pin map”.

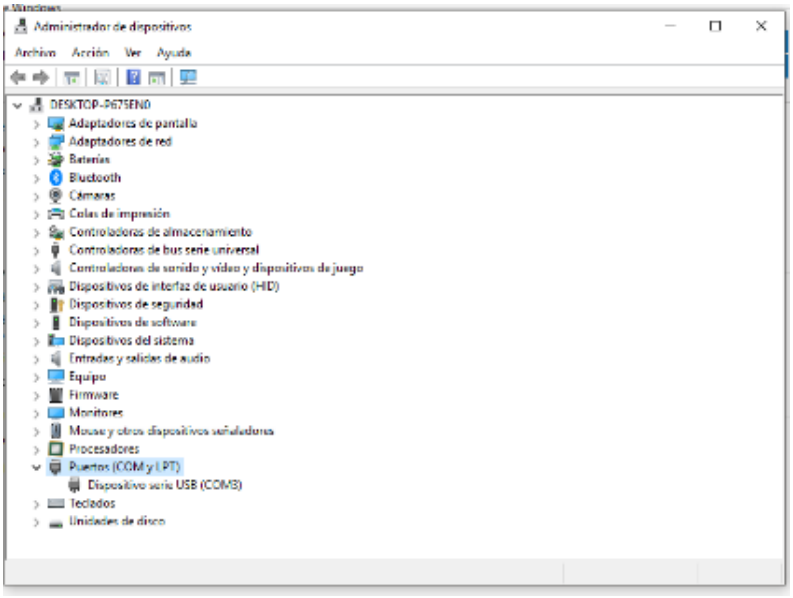


Los bloques “To Workspace” se configuran para enviar los datos recopilados por el sensor a MATLAB, el nombre de variable se establece por la que el usuario desee, y los demás ajustes se especifican como están en la imagen.

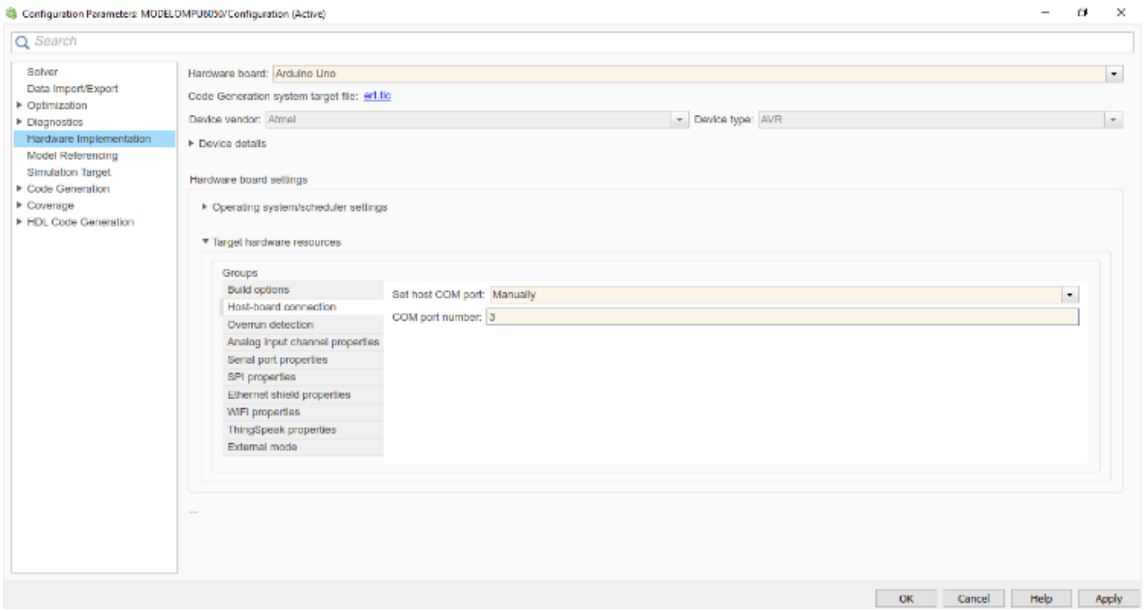
16. Para vincular el Arduino (en nuestro caso Arduino UNO), en la barra de herramientas de SIMULINK se elige la opción Simulación>Hardware Implementation > Hardware Board > Arduino UNO



17. En algunos equipos el reconocimiento de la placa Arduino no se da de forma automática, de modo que debemos configurarla. Para esto, seleccionamos Este equipo > Click derecho > Propiedades > Administrador de Dispositivos > Puertos (COM y LPT). En este caso, el COM correspondiente es el 3.



18. De este modo, continuar con la configuración. Simulación > Hardware Implementation > Target Hardware Resources > Host-board connection > set host COM port: > Manually > COM port number > Apply > Ok.



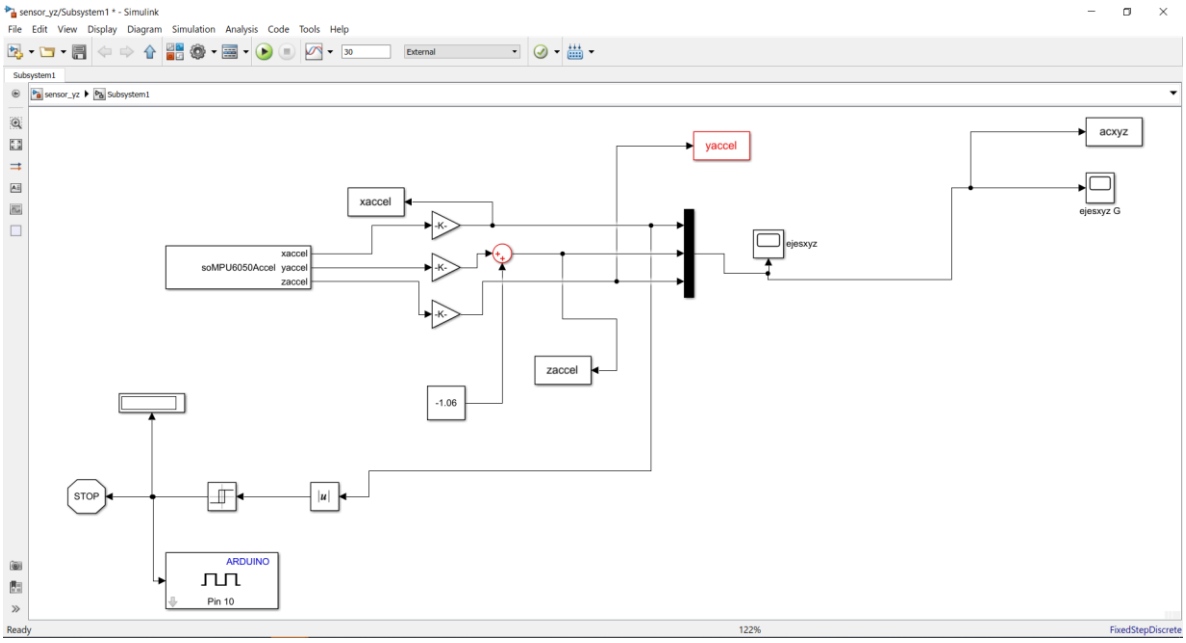


Figura 19. Interfaz en Simulink para recolección de muestras de aceleración por el sensor MPU6050.

Es importante que los bloques “To Workspace” se especifiquen como se encuentra en la figura 19, para evitar errores en la ejecución de los scripts.

Una vez terminada la interfaz, se puede inicializar la simulación directamente desde Simulink para comprobar su funcionamiento o desde el script que contiene las instrucciones para realizar todas las operaciones deseadas, y comprobar la ejecución correcta de la programación.

Planos mecánicos

La estructura metálica fue ensamblada a partir de elementos estructurales fabricados de ángulo y solera comerciales. Material en el cual se ha considerado un E= 200 GPa y un módulo de Poisson de 0.26.

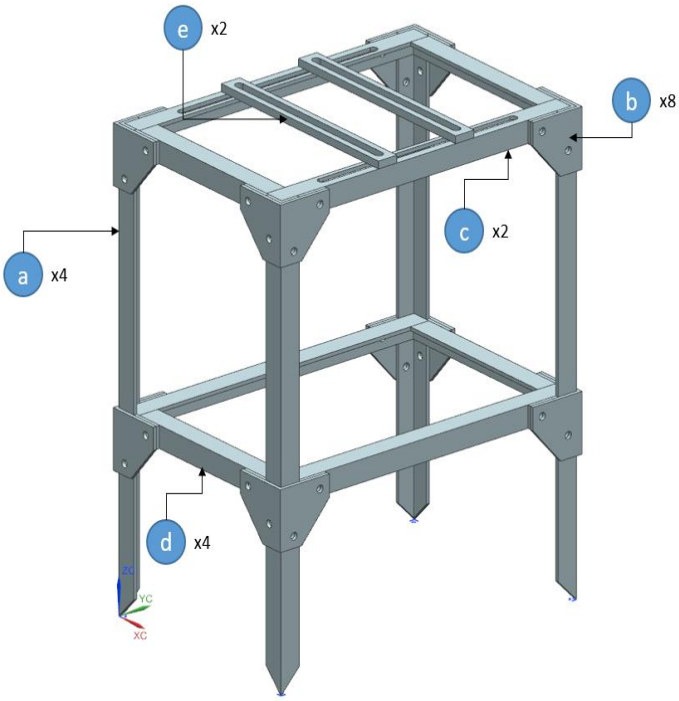


Figura 19. Vista isométrica de la estructura propuesta de 1 GDL.

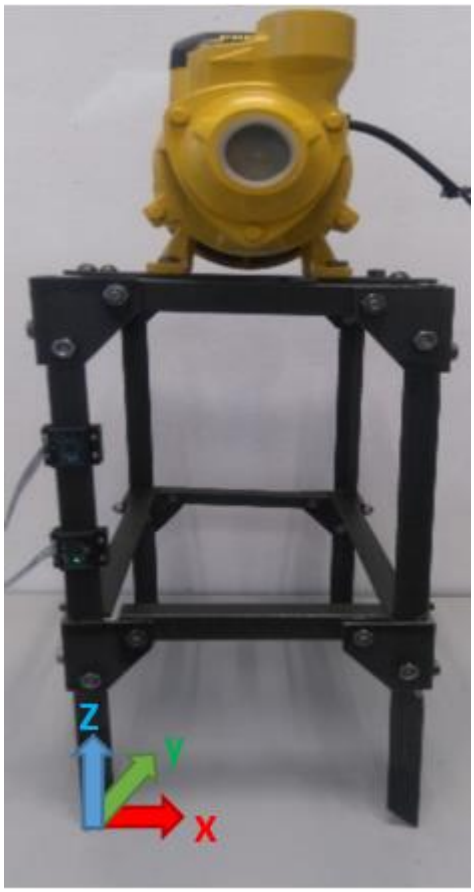


Figura 20. Prototipo construido para monitorear vibraciones.

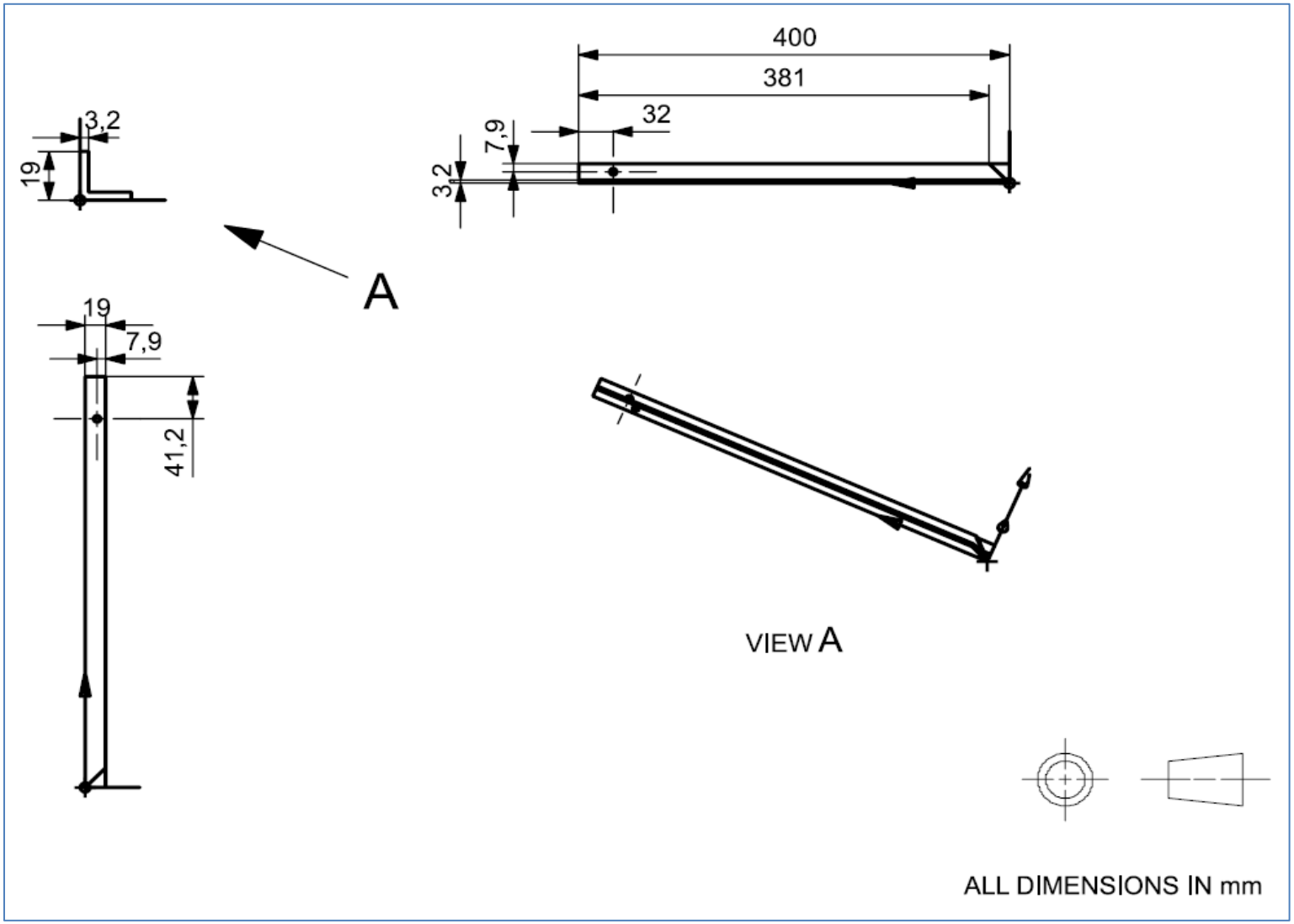


Figura 21. Plano de solera (elemento “a”) utilizada en estructura a monitorear

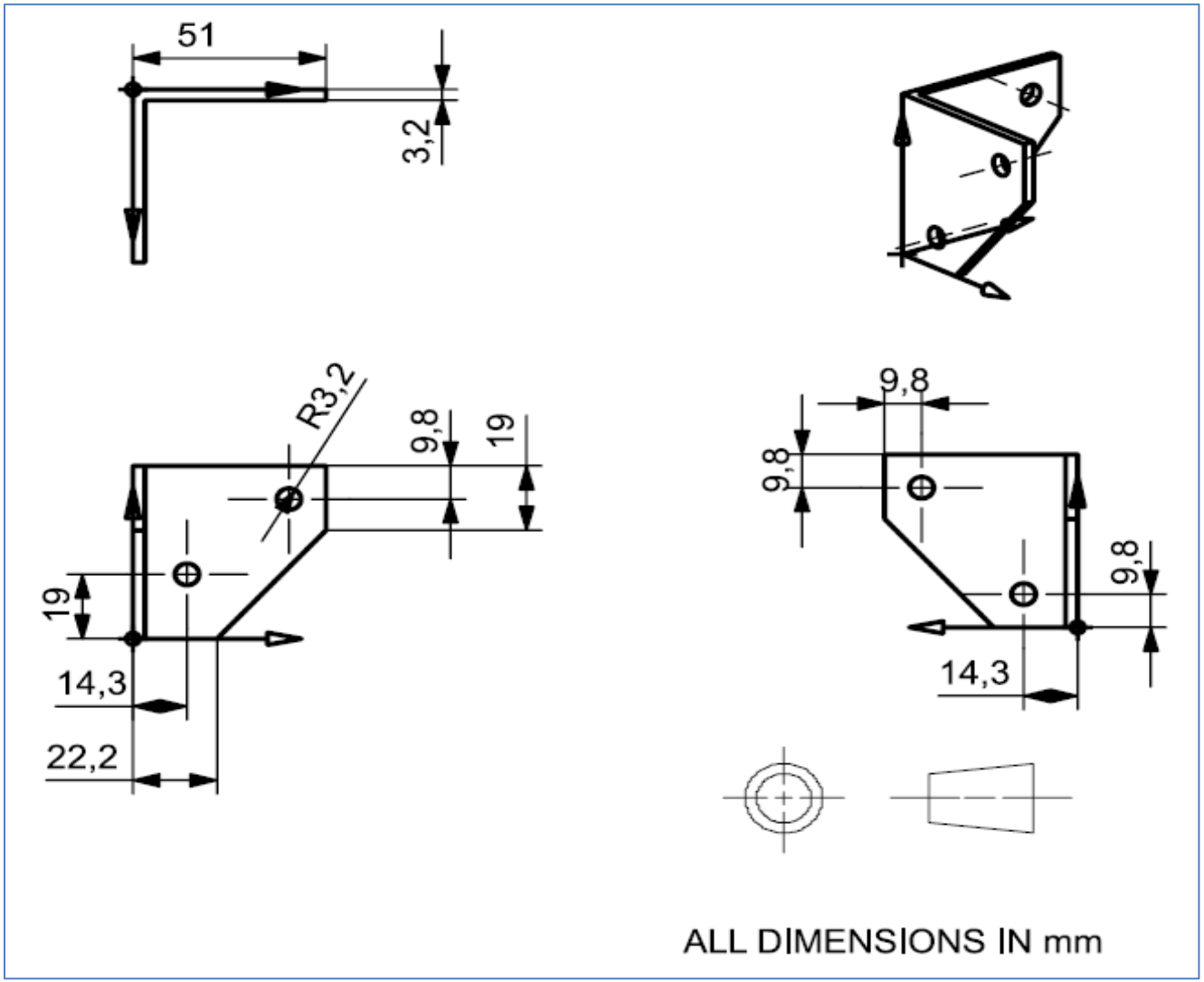


Figura 22. Plano (elementos “b”) para conexiones de estructura a monitorear.

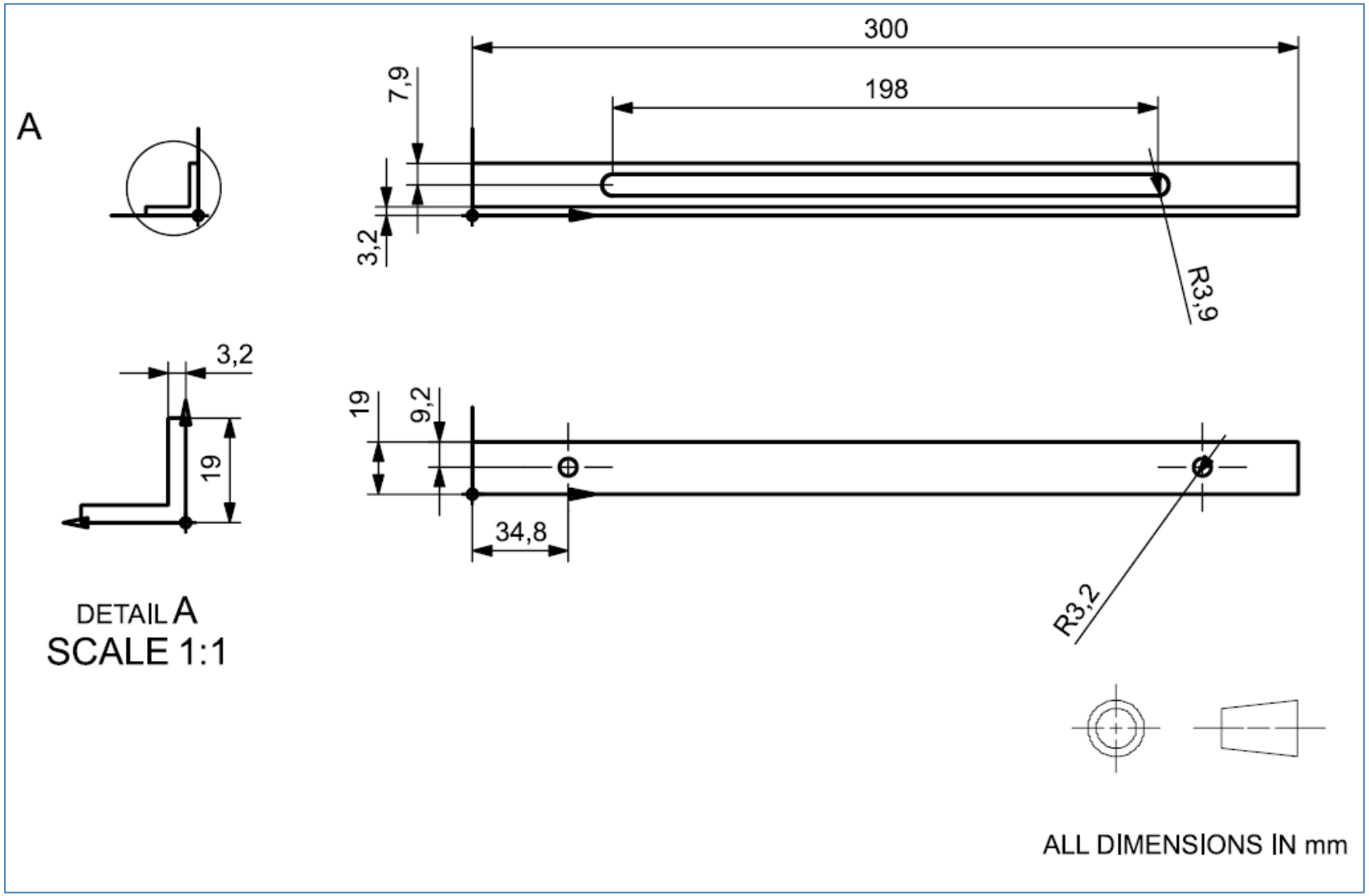


Figura 23. Plano de ranuras (elemento “c”) para sujeción de soleras de carga en estructura a monitorear.

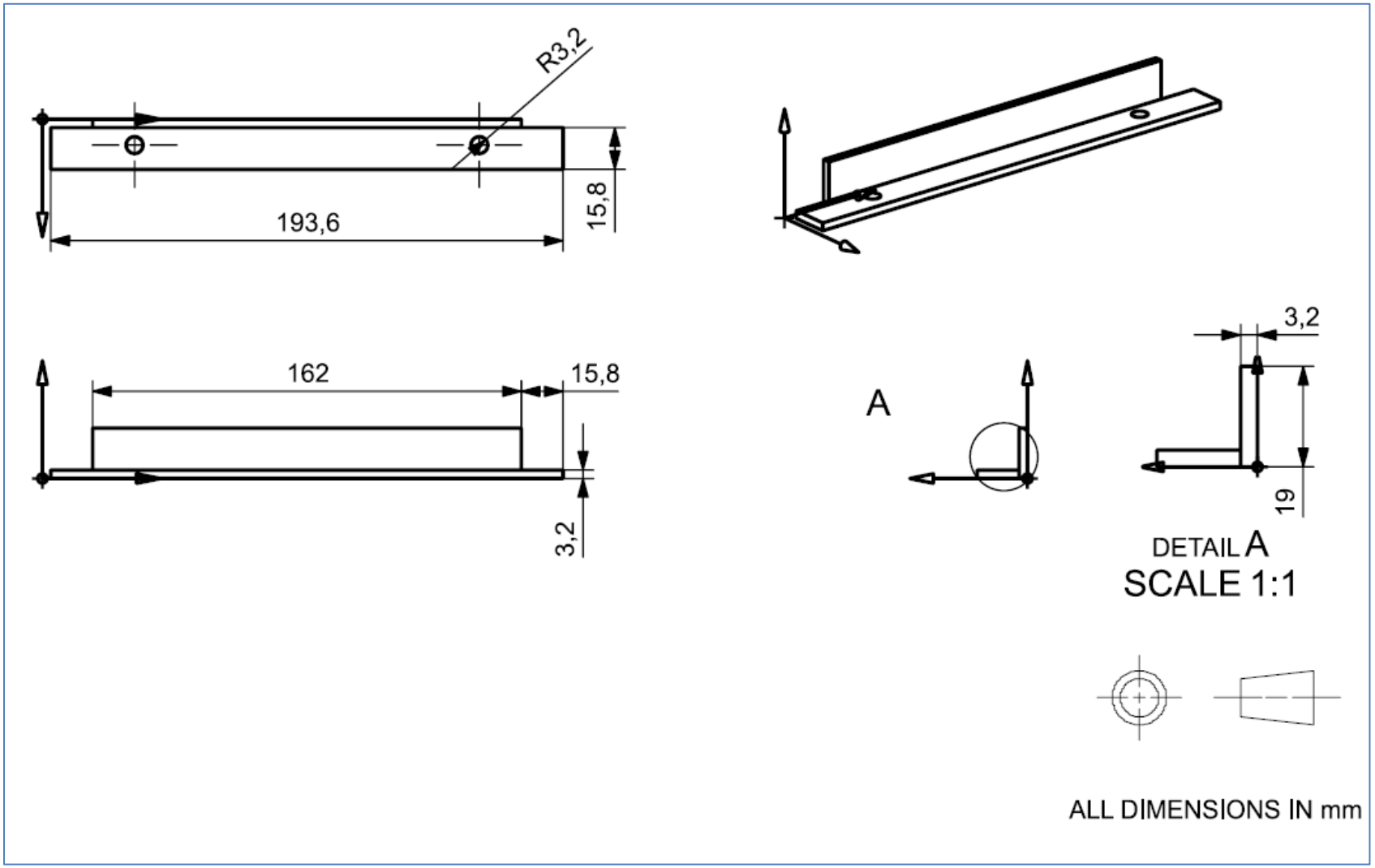


Figura 24. Plano de soleras (elemento “d”) para soporte de estructura a monitorear.

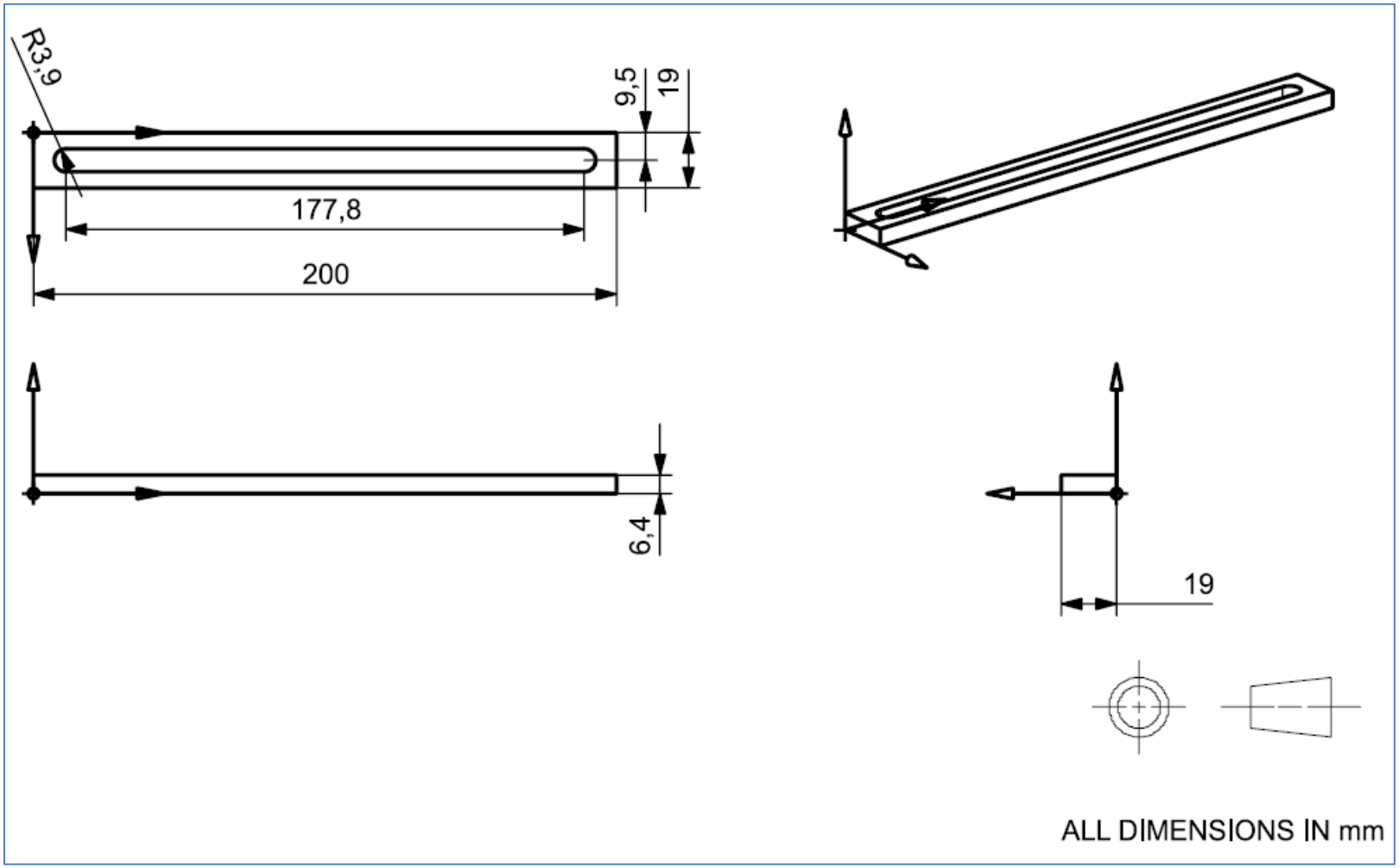


Figura 25. Plano de soleras (elemento “e”) con ranura para soporte y desplazamiento de carga en estructura a monitorear.

Capitulo V.- Resultados

Evaluación y pruebas

Parámetros para las ejecuciones de simulación en Simulink para el prototipo de estructura de 1 gdl

- Tiempo de simulación: 30 segundos.
- Inicio de simulación: Con el motor en operación**.
- Características de la carga
- RPM de la carga: 3450
- 120 V
- 60 Hz
- 5 A
- 3.7 Kg
- 0.5 HP
- La localización de la carga varía por cada prueba realizada.
- El eje z se normalizó a cero para las simulaciones, si se desea conocer las magnitudes sin normalizar, se debe desconectar el bloque “Constant” para realizar las ejecuciones de la simulación sin normalizar z a cero, o en su defecto, sumar a los valores en los registros enviados al Workspace de MATLAB el valor establecido en la constante (1.06)
- Para propósitos del reporte de pruebas, no se mostrarán todas las gráficas en el reporte por cuestión de espacio y resolución de imágenes, sin embargo, se especifican los resultados de las simulaciones realizadas en la tabla.

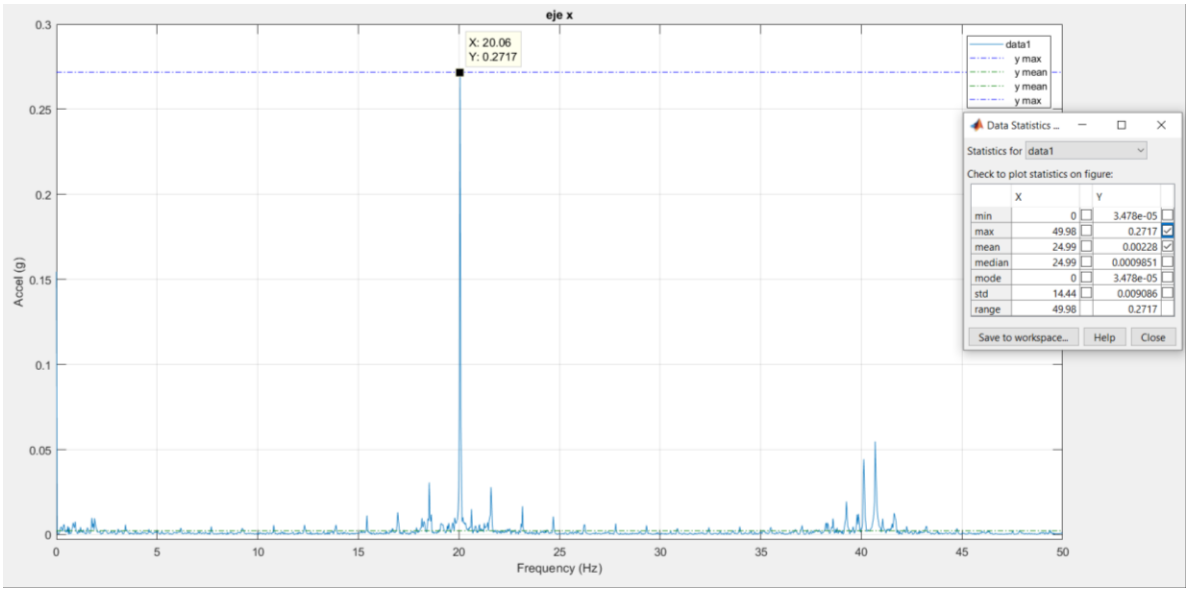


Figura 26. Espectro obtenido por la transformada rápida de Fourier (FFT) de acuerdo a las mediciones realizadas por el sensor MPU-6050 en el eje x.

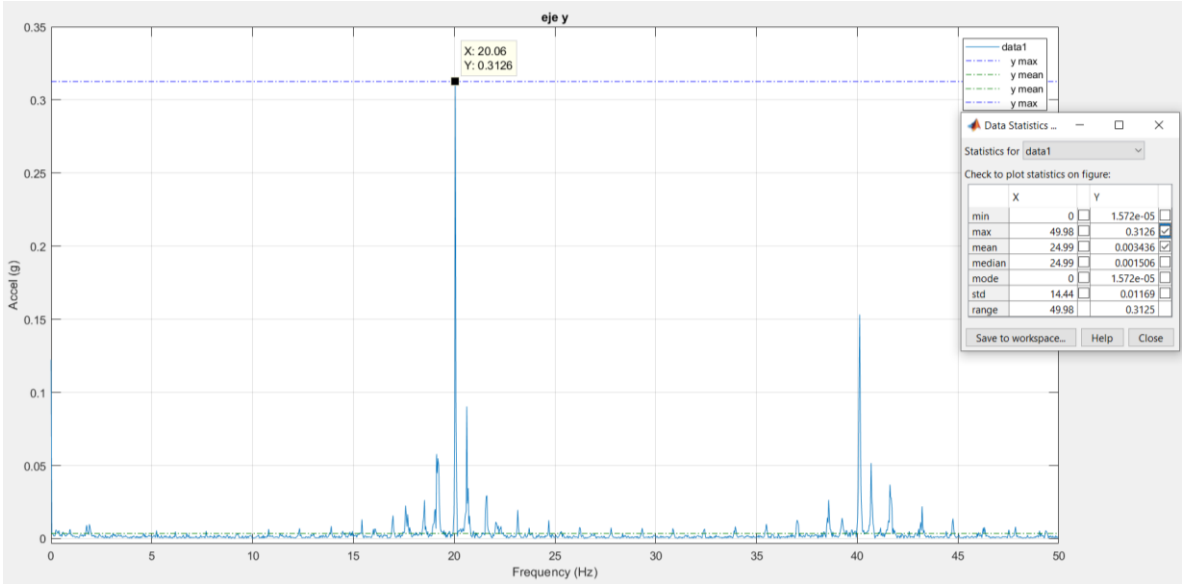


Figura 27. Espectro obtenido por la transformada rápida de Fourier (FFT) de acuerdo a las mediciones realizadas por el sensor MPU-6050 en el eje y.

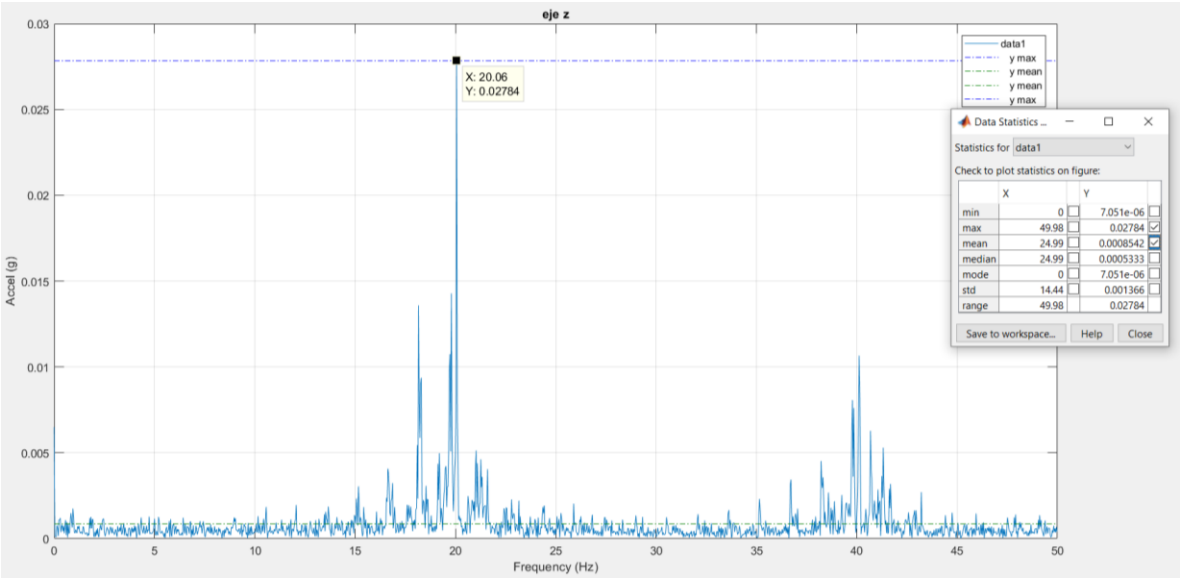


Figura 28. Espectro obtenido por la transformada rápida de Fourier (FFT) de acuerdo a las mediciones realizadas por el sensor MPU-6050 en el eje z.

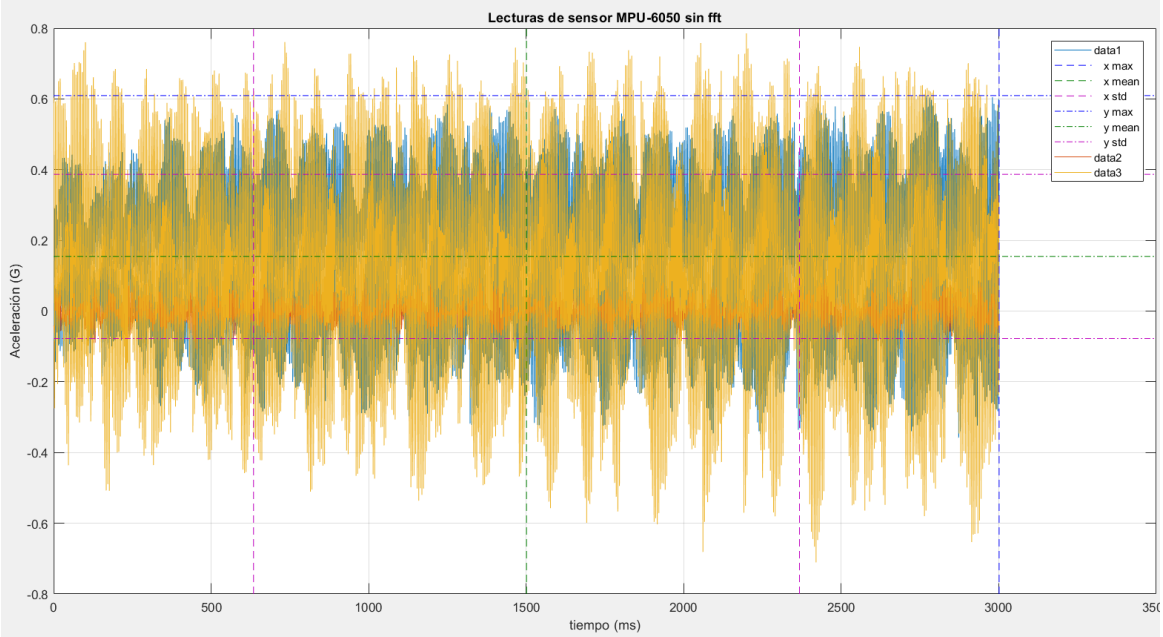


Figura 29. Acelerograma con las mediciones realizadas por el sensor MPU-6050 en los ejes x, y y z, sin implementación de transformada rápida de Fourier (FFT).

Localización de la carga estructura	Torque conexiones estructura	Valor promedio amplitud eje x (G)	Valor promedio amplitud eje y (G)	Valor promedio amplitud eje z (G)	Valor máximo amplitud eje x		Valor máximo amplitud eje y		Valor máximo amplitud eje z	
					G	Hz	G	Hz	G	Hz
Frontal-izquierda	0	0.002765	0.001571	0.000594	0.5442	19.99	0.07055	40.72	0.04975	19.99
Frontal-derecha	0	0.002837	0.001274	0.0006024	0.4759	19.99	0.04448	40.75	0.04229	19.99
Frontal-centro	0	0.004086	0.001105	.0007738	0.04115	20.07	0.0333	40.75	0.05125	20.07
Centro	0	0.002433	0.001753	0.001349	0.363	19.96	0.06861	40.75	0.0635	40.09
Posterior-izquierda	0	0.00228	0.003436	0.0008542	0.2717	20.06	0.3126	20.06	0.02784	20.06
Posterior-derecha	0	0.001262	0.00117	0.0004689	0.07643	20.03	0.037	20.03	0.006053	38.89
Posterior-centro	0	0.001572	0.001315	0.001086	0.1533	20.03	0.1042	20.03	0.02834	0.6664

Tabla 1. Resultados de mediciones realizadas por el sensor MPU-6050.

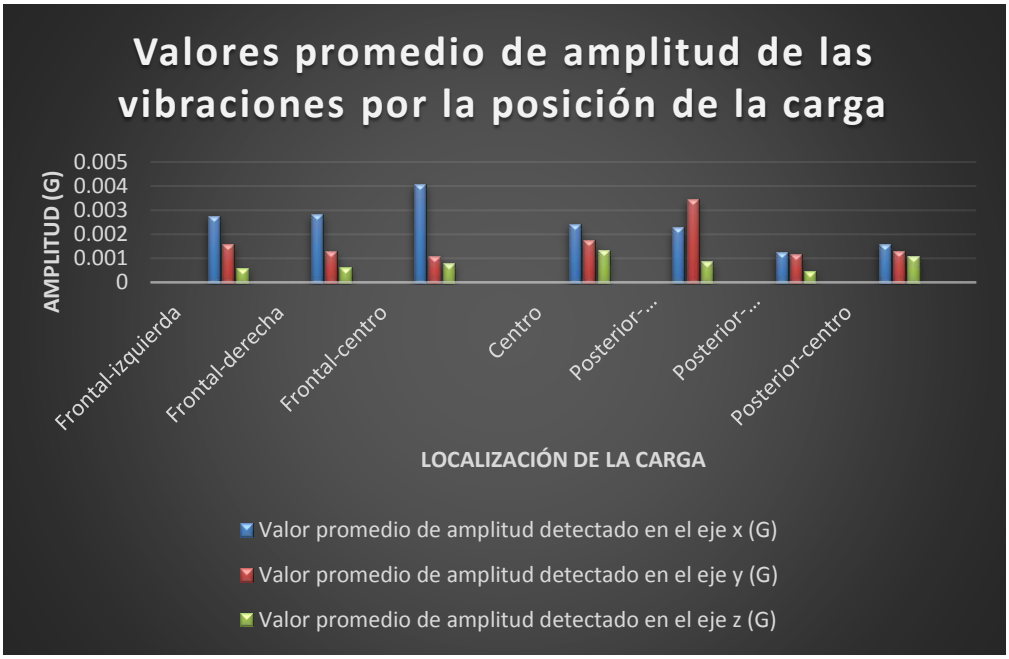


Figura 30. Valores promedio de amplitud de las vibraciones por la posición de la carga, sin precarga.

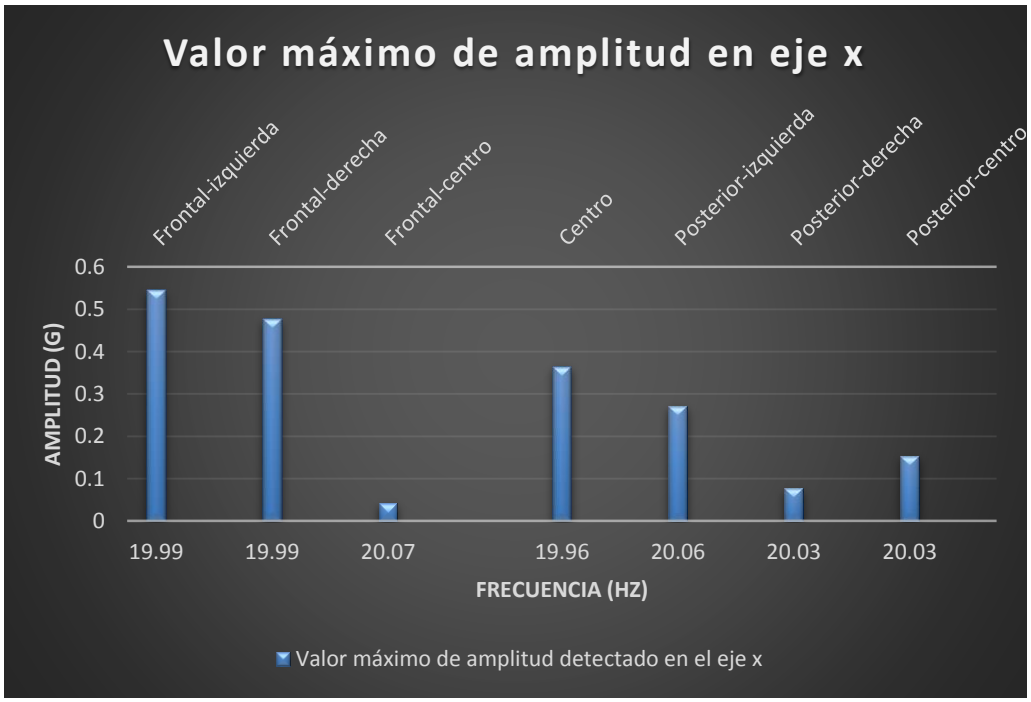


Figura 31. Máxima amplitud en el eje X, sin precarga.

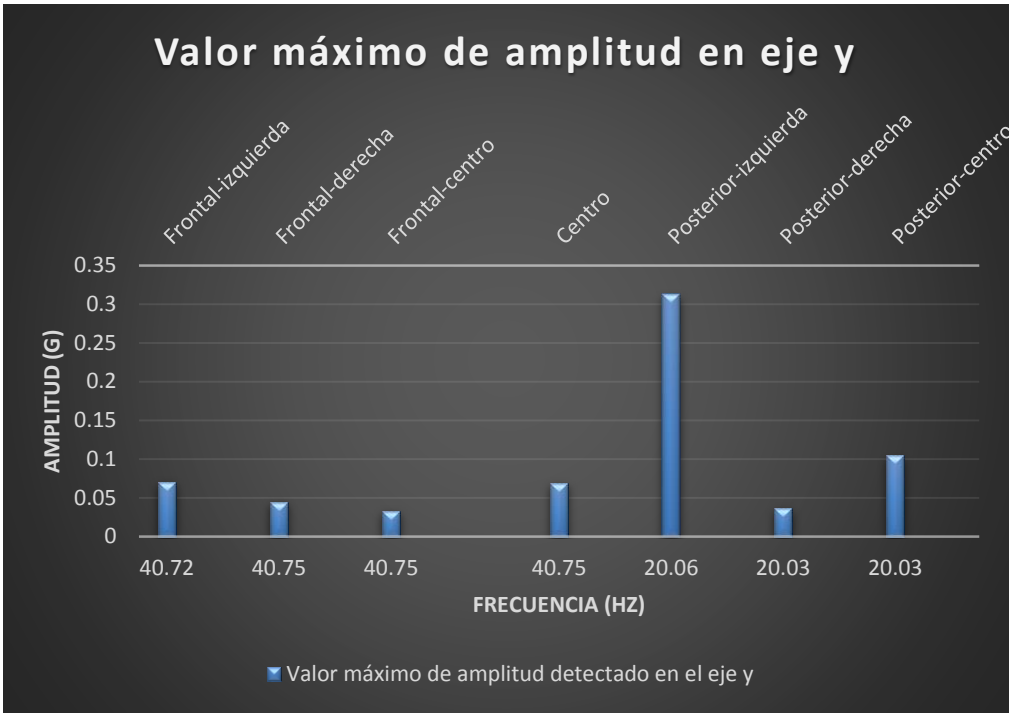


Figura 32. Máxima amplitud en el eje Y, sin precarga.

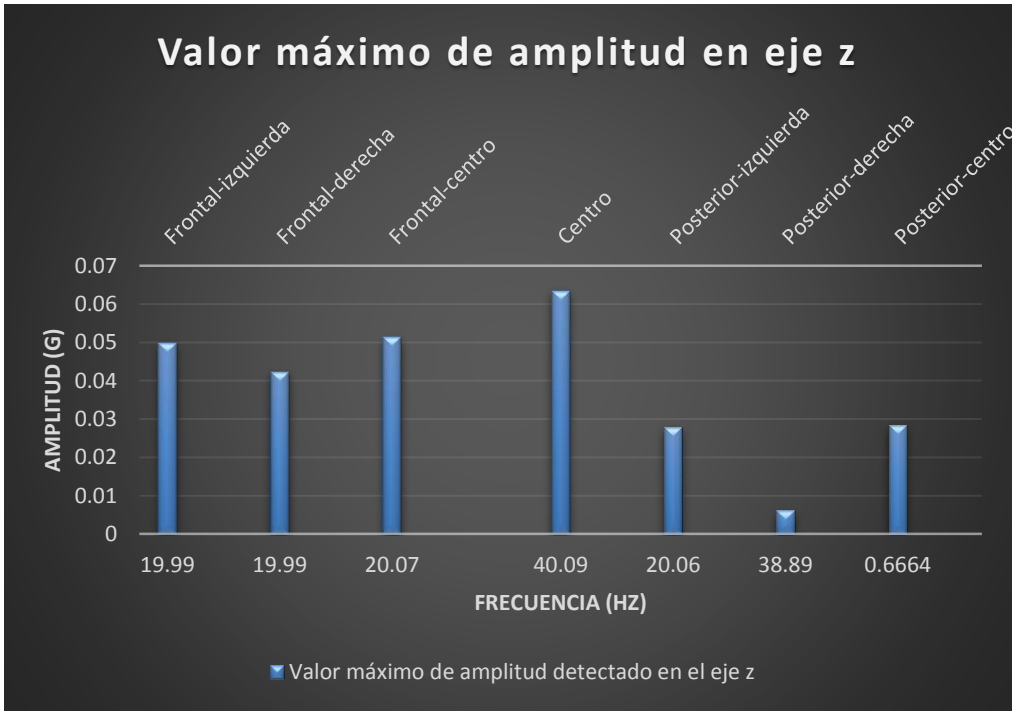


Figura 33. Máxima amplitud en el eje Z, sin precarga.

Los valores de aceleración detectados por el sensor MPU-6050 varían de acuerdo con la posición que tenga la carga en la estructura, así como del torque que tengan las conexiones.

Localización de la carga en la estructura	Torque conexiones de la estructura	Valor promedio amplitud eje x (G)	Valor promedio amplitud eje y (G)	Valor promedio amplitud eje z (G)	Valor máximo de amplitud eje x		Valor máximo de amplitud eje y		Valor máximo de amplitud eje z	
					G	Hz	G	Hz	G	Hz
Frontal-Izquierda	23.03 kg-cm	0.005578	0.003749	0.001974	0.291	40.7	0.1446	20.07	0.04496	20.07
Frontal-derecha	23.03 kg-cm	0.002769	0.001676	0.001154	0.2258	20.03	0.1707	40.69	0.02582	40.09
Frontal-centro	23.03 kg-cm	0.004675	0.002549	0.00235	0.4907	20.03	0.09626	20.03	0.05102	20.03
Centro	23.03 kg-cm	0.00412	0.00335	0.001497	0.3116	19.99	0.1647	20.73	0.06316	19.99
Posterior-derecha	23.03 kg-cm	0.004933	0.005282	0.006117	0.2754	19.93	0.3372	19.93	0.6496	19.96
Posterior-izquierda	23.03 kg-cm	0.002822	0.002487	0.002377	0.1255	40.72	0.1248	40.72	0.02552	20.66
Posterior-centro	23.03 kg-cm	0.004508	0.006862	0.008587	0.256	40.69	0.1546	19.89	0.08914	2.233

Tabla 2. Resultados de mediciones realizadas por el sensor MPU-6050 cambiando posición del acelerómetro en la estructura.

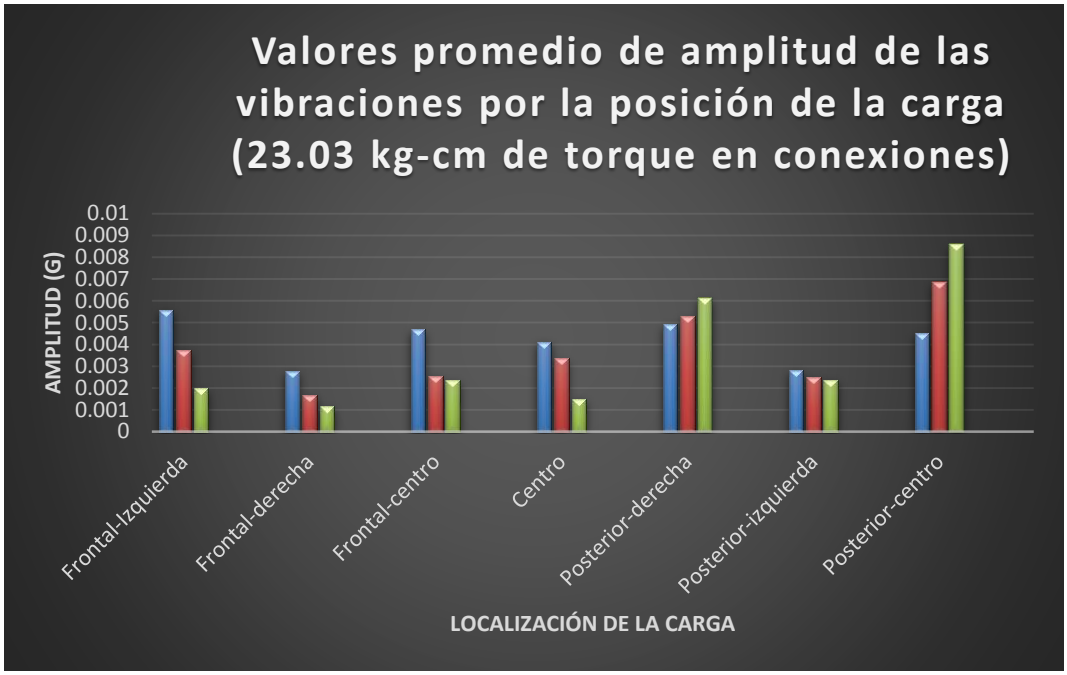


Figura 34. Valores promedio de amplitud de las vibraciones por la posición de la carga, con precarga.

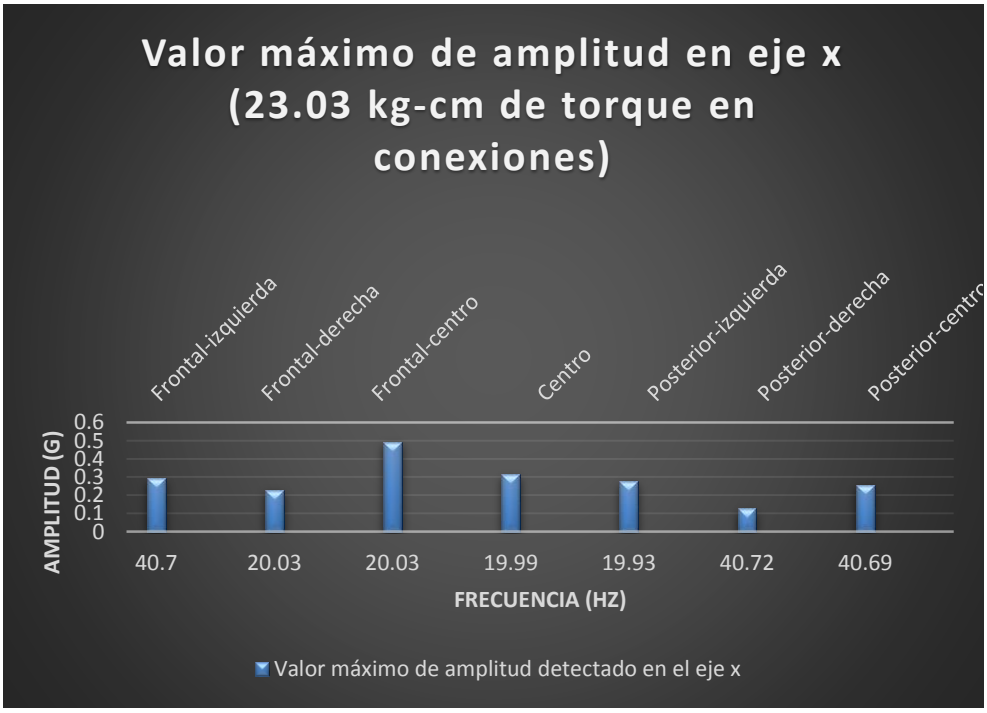


Figura 35. Máxima amplitud en el eje X, con precarga.

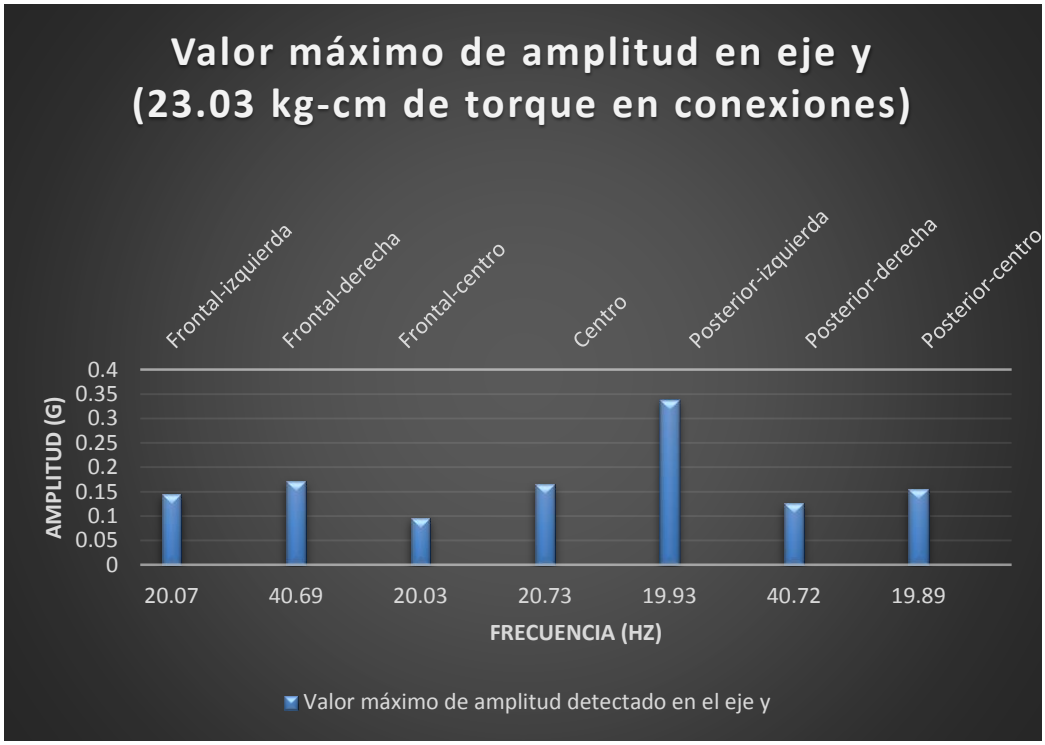


Figura 36. Máxima amplitud en el eje y, con precarga.

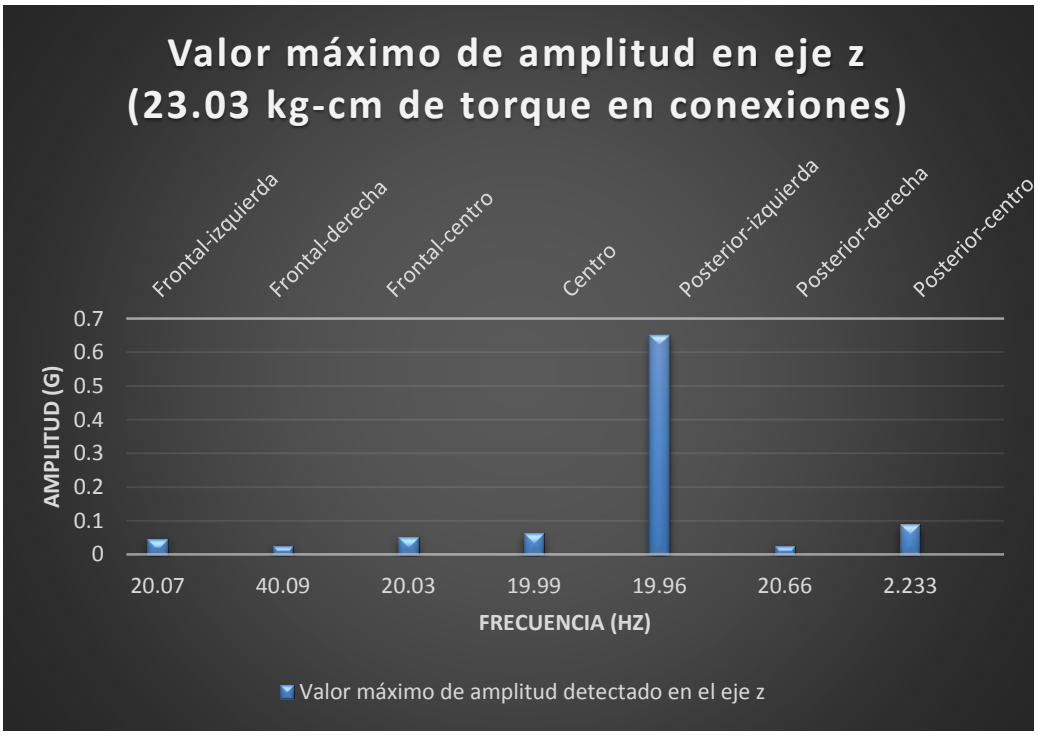


Figura 37. Máxima amplitud en el eje z, con precarga.

Conclusiones

Se concluye de manera anticipada que este dispositivo (figura 11,12, 13 y 14), aporta un nivel de seguridad apropiado, el cual permite predecir daños significativos a los equipos y mantener la continuidad de los procesos de producción, mediante desarrollos tecnológicos a bajo costo (Ocampo, 2013).

Es de suma relevancia el generar avances científico-tecnológicos referidos al campo del monitoreo de salud estructural, debido a su impacto en el sector de seguridad industrial y protección civil. Conocer el estado en que se encuentra la estructura y detectar condiciones de operación inseguras representa una ventaja para poder anticipar y/o minimizar las consecuencias de las fallas, reduciendo probabilidades de accidentes donde se vea involucrada la seguridad de las personas que se encuentran cerca del área donde se encuentra instalada la estructura y también los costos de mantenimiento o reparación a causa de la pérdida de rigidez al detectar problemas en sus primeras etapas, evitando que se magnifiquen.

La construcción de prototipos de bajo costo que permitan cumplir de manera satisfactoria con el monitoreo de salud estructural otorga la posibilidad de ser implementados de forma rápida en donde se requiera asegurar el estado de la rigidez en una estructura y en consecuencia, incrementar la confiabilidad en la operación normal y maximizar el tiempo de vida útil del sistema donde sea aplicado.

Debido a la naturaleza de la operación de las estructuras, las vibraciones pueden cambiar de acuerdo con el propósito que deban cumplir dichas estructuras (turbinas eólicas, torres de distribución eléctrica, puentes, represas, edificios, túneles, almacenes de combustibles, ductos, integridad perimetral, etc), las condiciones para el sistema de monitoreo pueden cambiar en las condiciones de amplitudes de aceleración y frecuencia permisibles. El sistema propuesto permite adecuar los parámetros necesarios para ajustarse a los requerimientos del usuario, estableciendo un rango de operación específico donde evitar falsos positivos de condiciones inseguras de operación.

Bibliografía

Bernal, C. F., & Cortés, N. D. (2016). Simulación de un banco de pruebas para análisis de vibraciones. Bogotá, Colombia: Universidad Distrital Francisco José de Caldas. Recuperado el 05 de 06 de 2019, de http://repository.udistrital.edu.co/bitstream/11349/6143/1/CalderonBernalFabioAndres2017.pdf

Mandado, P. E., Marcos, A. J., Fernández, S. C., & Armesto, Q. J. (2009). *Automatas programables y sistemas de automatización.* México D.F.: Alfaomega.

Martínez, B. R., Pino, T. J., Terán, H. H., & Arteaga, L. O. (2017). *Vibraciones Mecánicas I.* Sangolquí, Ecuador: Universidad de las Fuerzas Armadas ESPE. Obtenido de http://repositorio.espe.edu.ec/xmlui/handle/21000/13747

Martínez, B. R., Pino, T. J., Terán, H. H., & Arteaga, L. O. (2017). *Vibraciones Mecánicas II.* Sangolquí, Ecuador: Universidad de las Fuerzas Armadas ESPE. Obtenido de http://repositorio.espe.edu.ec/xmlui/handle/21000/13753

Meléndez, G. S. (2017). *Fachadas con energía renovable.* México D.F.: Trillas.

Ocampo, J. O. (2013). SISTEMA DE BAJO COSTO PARA MONITOREO DE DISTURBIOS DE VIBRACIÓN EN MÁQUINAS ROTATORIAS. *Innovare,* 14-30.

Pope, E. J. (2000). *Soluciones prácticas para el ingeniero mecánico.* Mexico D.F.: Mc. GrawHill.

ANEXO 1.- Manual de Operación

Monitoreo estructural de las vibraciones mecánicas mediante PLC S7-1200 y Arduino

Se compone de las siguientes secciones, las cuales se desarrollan en función del avance y resultado de las pruebas realizadas.

1. Generalidades

   El sistema de monitoreo de vibraciones desarrollado tiene una interfaz sencilla, y una vez calibrado por el usuario de acuerdo a sus necesidades, la operación se simplifica para realizar los análisis necesarios, resultado de las mediciones obtenidas por el sensor MPU-6050.

   La confiabilidad de las mediciones estará influenciada por la operación correcta del sistema de monitoreo. Es necesario tomar en cuenta las condiciones ambientales en donde se implementará, así como de evitar cualquier condición que pueda causar una vibración alta por causas independientes a la estructura o a la carga y que no sea de relevancia para el monitoreo, tales como golpes accidentales o una sujeción incorrecta del sensor.

2. Manejo seguro

   El sistema desarrollado para el monitoreo de vibraciones consta de los siguientes elementos:

   Placa Arduino UNO

   Sensor MPU-6050

   Relevador SUNHOLD THD-0501L

   MATLAB R2017b o posterior (versiones anteriores provocan fallos al instalar librerías)

   Simulink

   Librerías para reconocimiento y control de placa Arduino UNO y sensor MPU6050:

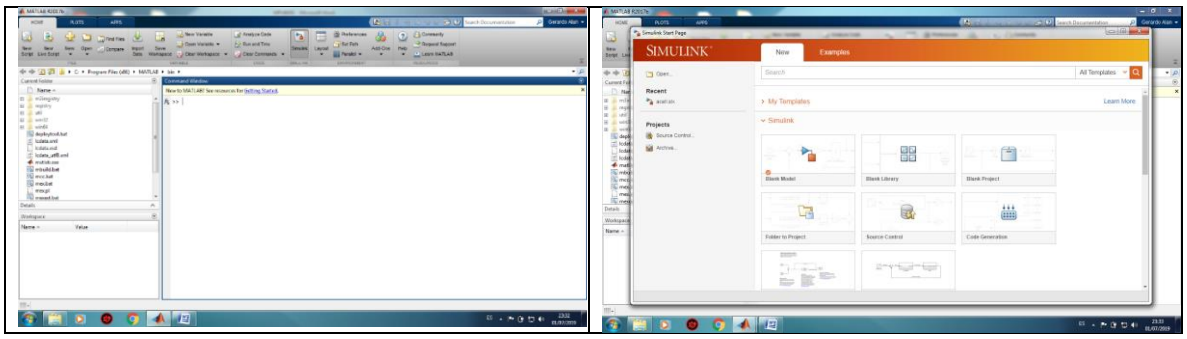
   - MATLAB Support Package for Arduino Hardware
   - Simulink Support Package for Arduino Hardware (RASPLib)
   - Rensselaer Arduino Support Package Library (RASPLib)

Asegurarse de instalar todas las librerías o de lo contrario el reconocimiento del sensor o de la placa podría fallar.

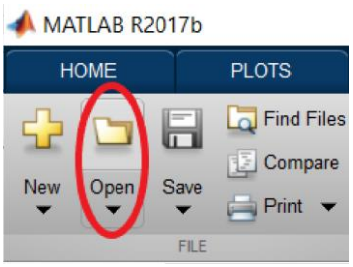
Verificar las conexiones establecidas entre el PLC y el relevador de control para el apagado de la carga en caso de detectar condiciones de operación inseguras antes de energizar el sistema, se deben seguir los esquemas eléctricos para evitar problemas con las conexiones y/o evitar descargas eléctricas. Operar las conexiones del motor al PLC cuando el equipo se encuentra apagado.
Verificar que las conexiones entre la placa, el sensor y la computadora funcionen adecuadamente para evitar fallas en los registros obtenidos en la simulación.

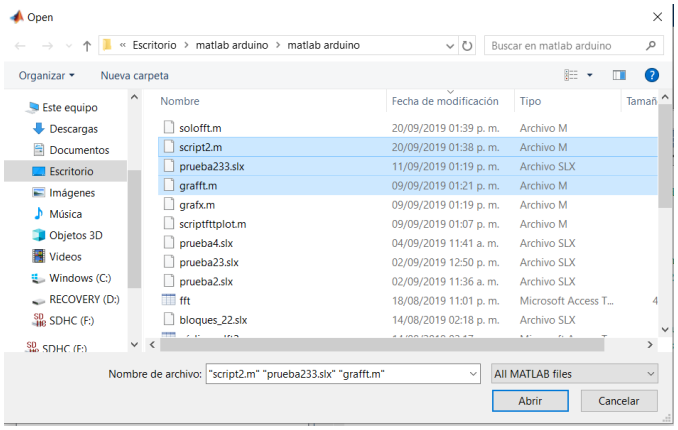
3. Energización.
   Se energiza el dispositivo de monitoreo en paralelo con el equipo a supervisar.
   El dispositivo de monitoreo requiere de tiempo para que la interfaz desarrollada en Simulink cargue el programa en la placa Arduino y esta reciba las señales captadas por el sensor.
   Para evitar fallas en la simulación se contempla que antes de que el dispositivo de monitoreo inicie su operación, la carga debe estar en funcionamiento, debido a que por la sensibilidad del dispositivo, al arranque de la carga podría detectar un pico alto de aceleración y detectarlo como un "falso positivo de condición insegura", o en su defecto, configurar la tolerancia de amplitudes de aceleración, realizando varias pruebas para determinar un rango de valores promedio entre los valores normales de vibración cuando se encuentra activa la carga.

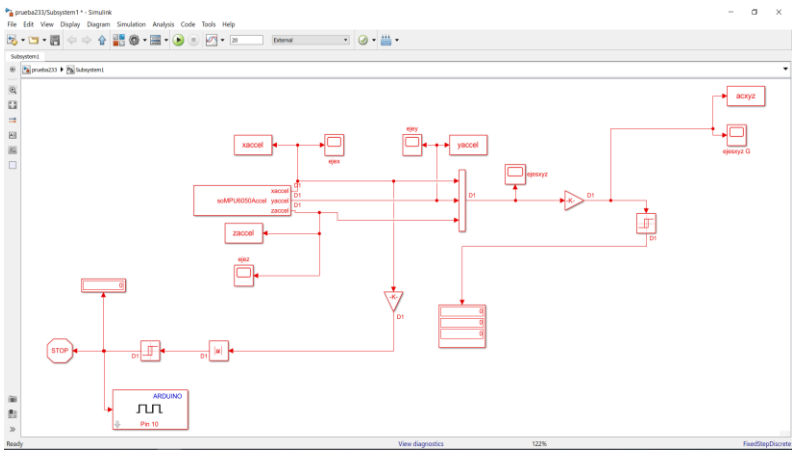
4. Puesta en marcha.
   a. Arranca el dispositivo de monitoreo.
      1. Abrir MATLAB R2017b y Simulink

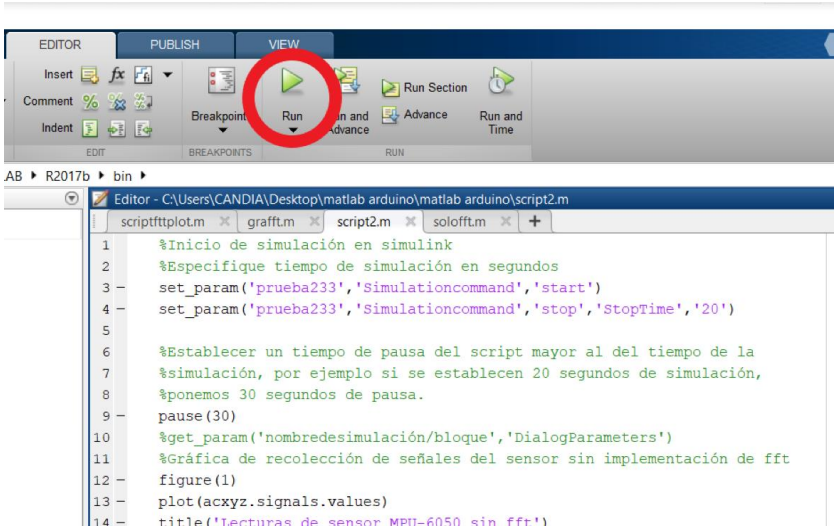


2. Cargar scripts (xxxxxx.m, xxxxxx.m, xxxxxx.m) e interfaz gráfica desarrollada en Simulink (xxxxxxxx.slx)

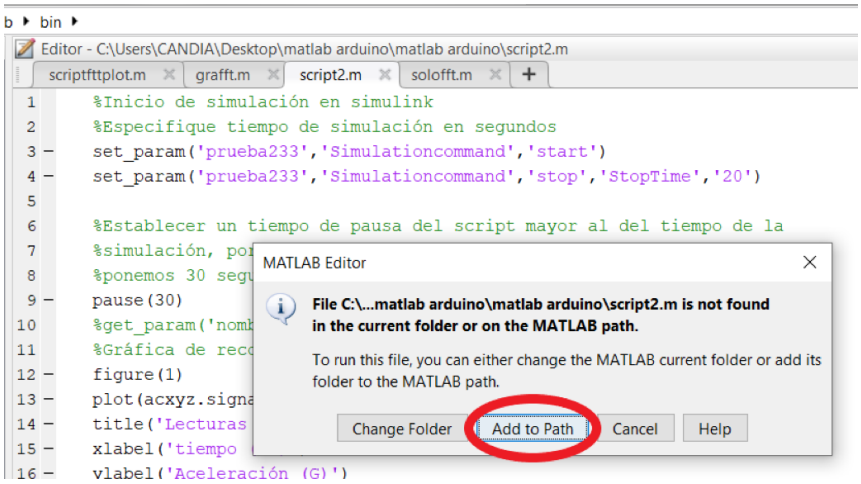








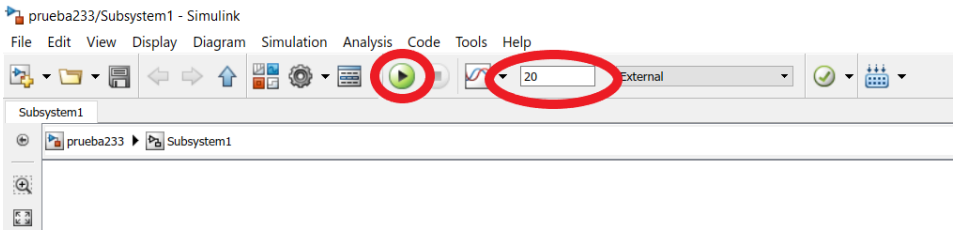
Cuando se inicia por primera vez el script, aparece la siguiente ventana:



Dar click en “Add to Path” y comenzará la ejecución del script.

Al finalizar la ejecución del script se abrirán automáticamente las ventanas con las gráficas correspondientes para cada eje

3. Cargar el programa a la placa una vez para verificar que las conexiones estén realizadas correctamente.

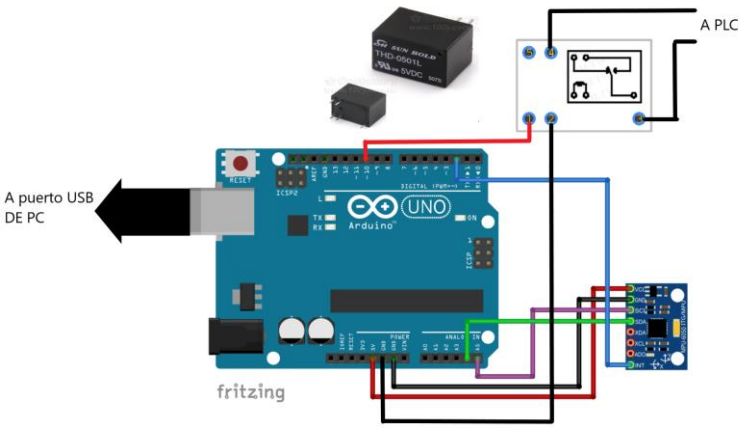


- Para iniciar la simulación desde Simulink, dar clic en el botón que se muestra en la imagen, con el botón se inicia la simulación y el tiempo se establece por el usuario en el recuadro señalado, o se deja como "inf" para una simulación sin tiempo de paro.

4. Detener la simulación, o establecer el tiempo de la ejecución en la interfaz de Simulink para que se detenga automáticamente.

b. Arranca el equipo en supervisión.

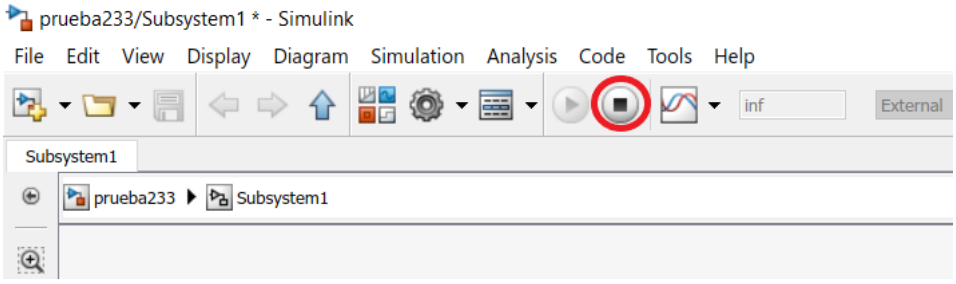
1. Verificar las conexiones entre el motor y el PLC.



- Las terminales 1 y 2 del relevador corresponden a la bobina del relevador, van conectadas hacia La placa Arduino UNO, en el pin seleccionado para la salida del pulso (pin 10) y el pin GND. 3 es el común, 5 y 4 son los contactos normalmente cerrado y abierto respectivamente.

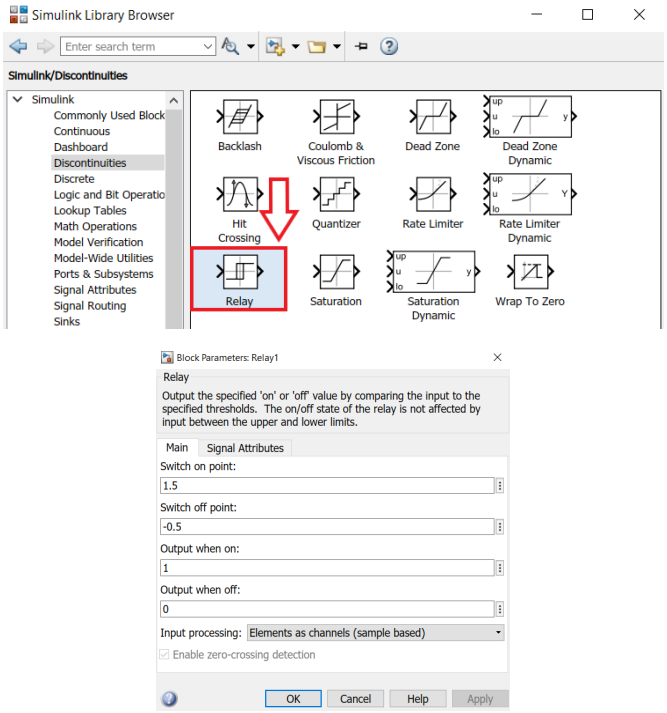
2. Inicializar el equipo antes de que la simulación se ejecute.

3. Ejecutar la simulación, utilizando el script xxxxx.m, para realizar la operación de la simulación y mostrar las gráficas resultantes de las lecturas de aceleración y la aplicación de la FFT a dichas muestras.
4. Esperar a que se recolecten las mediciones establecidas por el usuario y apagar el equipo a supervisar (si es requerido, para ahorrar energía) cuando la simulación termine.



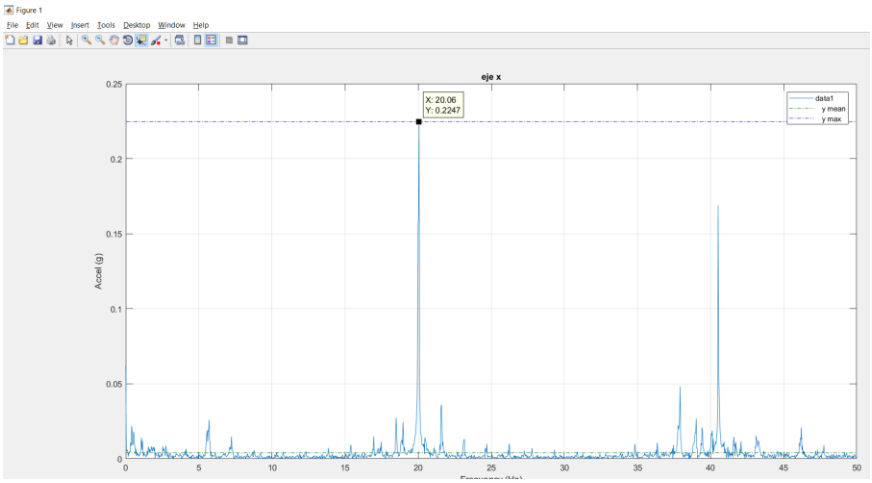
5. Configuración de parámetros. Se establece la frecuencia natural de la estructura como referencia.

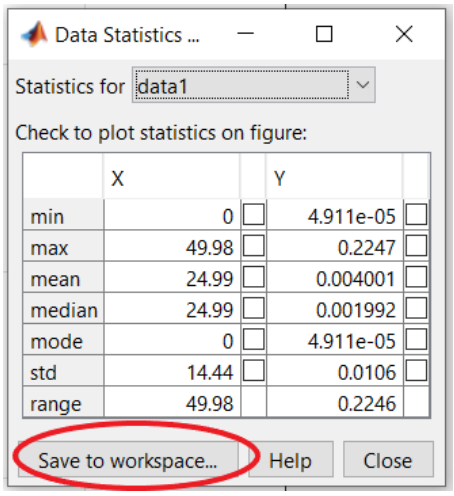
   Realizar varias pruebas para establecer un rango de valores de aceleración, y establecer un límite para que cuando sea rebasado, la interfaz desactive la carga automáticamente.



6. Tabla de referencia de daño, por amplitud y frecuencia.

   Al finalizar la simulación, se guardan los registros en variables que pueden manejarse en el entorno de trabajo de MATLAB, las gráficas mostradas automáticamente por el script contienen la información de los valores máximos de amplitud (en G y mm/s^2) y promedio de las vibraciones, así como la frecuencia en las que estas se detectaron.





El formato para anotar los valores obtenidos se muestra en las siguientes tablas:

Localización de la carga en la estructura	Torque conexiones de la estructura	Valor promedio amplitud eje x (G)	Valor promedio amplitud eje y (G)	Valor promedio amplitud eje z (G)	Valor máximo de amplitud eje x		Valor máximo de amplitud eje y		Valor máximo de amplitud eje z	
					G	Hz	G	Hz	G	Hz

Localización de la carga en la estructura	Torque conexiones de la estructura	Valor promedio amplitud eje x (G)	Valor promedio amplitud eje y (G)	Valor promedio amplitud eje z (G)	Valor máximo de amplitud eje x		Valor máximo de amplitud eje y		Valor máximo de amplitud eje z	
					G	Hz	G	Hz	G	Hz