Universidad de Valladolid

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería en Electrónica Industrial y Automática

Medida de la fuerza centrípeta mediante sendos sensores analógico y digital. Análisis de datos y aplicaciones

Autor:

De la Fuente Quintanilla, Carlos

Tutor:

Páramo Vela, José Ricardo
Departamento de Física Aplicada

Valladolid, junio 2018.

Quiero agradecer a todas las personas que han ayudado a la elaboración de este Trabajo Fin de Grado, en especial a mi familia, amigos, al tutor José Ricardo Páramo Vela y al técnico de Laboratorio Rafael Viñas García. Con su estimable apoyo, ha sido posible la realización de este TFG.

Dímelo y lo olvidaré; muéstramelo y lo recordaré; involúcrame y aprenderé.
Confucio.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS Y GRÁFICAS

RESUMEN

Este Proyecto realiza una automatización parcial de un equipo de laboratorio. En concreto, el equipo sirve para el estudio de la fuerza centrípeta. El proceso se lleva a cabo mediante dos sensores, uno analógico y otro digital. El sensor analógico mide la fuerza de las masas del equipo. El digital indica la velocidad angular. Además, es necesario una fuente de alimentación que suministre electricidad al motor, para el giro de la plataforma del equipo. La automatización consiste en la adquisición de datos de los sensores y su presentación. Para ello, se utiliza una tarjeta de adquisición de datos y un programa informático: LabVIEW. Este software de programación gráfica permite el diseño de una aplicación para efectuar las mediciones variando tres parámetros: masa, radio y velocidad angular (omega). Esta aplicación se ejecuta de un modo sencillo e intuitivo para el usuario, facilitando el estudio y el análisis de los datos.

Palabras clave: fuerza centrípeta, automatización, sensor, adquisición de datos, LabVIEW.

ABSTRACT

This Project develops a partial automation of a laboratory equipment. In particular, the equipment is useful for the centripetal force analysis. The process is carried out by two sensors, one is analog and the other one is digital. The analog sensor measures the force of the equipment masses. The digital one indicates the angular velocity. A power source is also needed to provide electricity to the rotational motor, so that the rotating platform of the equipment spins. Automation involves the gathering of data from the sensors and its presentation. For this purpose, it is used a data acquisition card and a visual programming language (LabVIEW). This software makes possible the design of an application to carry out the measurements by varying one of these three parameters: mass, radio and angular velocity (omega). The application is run in a simple and intuitive way for users, in order to allow for the study and analysis of the data.

Keywords: Centripetal force, automation, sensor, data acquisition, LabVIEW.

CAPÍTULO 1.
INTRODUCCIÓN

1.1. Antecedentes

El control automático ha sido clave en el desarrollo de la ciencia y de la ingeniería. Es un factor esencial en tecnología espacial, sistemas robóticos y procesos industriales. Resulta indispensable en el control numérico de las máquinas del sector de la industria, así como en otros procesos, como el control de presión, temperatura y velocidad, entre otras magnitudes.

Los avances en la teoría del automatismo otorgan buenos medios para conseguir el funcionamiento óptimo de sistemas dinámicos y de mejora de la productividad. La automatización es fundamental para el conocimiento de científicos e ingenieros.

De hecho, los desarrollos de las últimas décadas en la teoría de control moderna se centran en el campo del control óptimo de sistemas, tanto determinísticos como estocásticos. También incluye otros sistemas complejos, como adaptación y aprendizaje. Las aplicaciones recientes, además de la ingeniería, engloban a la biología, biomedicina, economía y diversos aspectos socioeconómicos.

Además, la automatización industrial es posible gracias a la unión de varias tecnologías: instrumentación, sensores, sistemas de comunicación, robótica y telemetría, entre otras. Se encuentra en muchos sectores de la economía y abarca, además de la maquinaria y la fabricación de productos, la gestión de procesos, el manejo de información, la mejora de cualquier proceso (desde su instalación, mantenimiento y diseño) e incluso su comercialización.

La instrumentación electrónica mide las variables de la materia en sus diferentes estados: gaseoso, líquido y sólido. Los sensores muestran qué sucede en el proceso, dónde se encuentra en un momento determinado y da la señal para el siguiente paso. Los sistemas de comunicación enlazan los datos del sistema. También es importante destacar la neumática y los motores, que se encargan del movimiento.

Por otra parte, la computadora se ha convertido en el elemento indispensable de estos sistemas de control. Gracias a ella, los datos obtenidos por sensores se analizan para controlar y tener una información

precisa y exacta de qué ocurre en cada momento al sistema. La informática ha contribuido enormemente a la automatización de procesos. Con una plataforma adecuada, que conjugue hardware y software de modo productivo, se aumenta la eficiencia y eficacia de distintos procesos, con la mejora de la actividad humana ante problemas complejos y dinámicos.

1.2. Objetivos

Este Proyecto versa sobre la automatización parcial de un equipo de laboratorio, de la medición de magnitudes a través de sensores y de la adquisición, tratamiento y análisis de datos. Para lograr esta meta, se dispone de los materiales comprados por la empresa estadounidense Pasco y del software LabVIEW *(Laboratory Virtual Instrument Engineering Workbench)*, un potente programa informático para desarrollar sistemas de medidas y de control.

Todos los materiales fueron aportados por el departamento de Física Aplicada, tanto el material de laboratorio como el material informático. Parte de la elaboración del Proyecto se realizó en el laboratorio del departamento, en la Escuela de Ingenierías Industriales de la Universidad de Valladolid, sede Francisco Mendizábal.

El equipo en concreto es el sistema de mecánica rotacional de Pasco ME-8950, con sus correspondientes accesorios. Este equipo permite estudiar la fuerza centrípeta. Las dos magnitudes a medir son la velocidad y la fuerza, gracias a sendos sensores de la misma compañía.

A través de una tarjeta de adquisición de datos de National Instruments, estos se envían a un ordenador, donde un programa realizado con LabVIEW se encarga de gestionarlos.

Todos estos materiales permiten el estudio de sistemas de control, la automatización parcial de un sistema y la observación de los datos obtenidos. Pero además de estos propósitos, también se ha realizado este Proyecto con el fin de ayudar al alumno en la realización de esta práctica de laboratorio, así como conocer los aspectos básicos de la automatización y el manejo de un software de lenguaje de programación gráfica de instrumentación virtual.

1.3. Estructura del Proyecto

Esta memoria está estructurada en los siguientes capítulos:

- **Capítulo 1:** Se explica brevemente la introducción y los objetivos, así como una visión global del Proyecto.

- **Capítulo 2:** Se describe la parte teórica de este Trabajo Fin de Grado (TFG). A través del análisis de la cinemática y la dinámica, pasando por la segunda ley de Newton, se deduce la fuerza centrípeta.

- **Capítulo 3:** Se detallan los materiales utilizados en este Proyecto, como el equipo de laboratorio, el hardware y el software.

- **Capítulo 4:** Aquí se expone el montaje del equipo paso a paso.

- **Capítulo 5:** Se explica cómo es el proceso de la adquisición de datos mediante una tarjeta, así como el software necesario. Además, se especifica el proceso de calibración de los sensores de fuerza y de movimiento.

- **Capítulo 6:** Se narra el diseño en LabVIEW de la aplicación, tanto el diagrama de bloques como el panel de control. También se comenta el funcionamiento de la aplicación con ejemplos.

- **Capítulo 7:** Contiene las conclusiones de la realización de este TFG.

- **Bibliografía:** Se presentan los recursos bibliográficos y las fuentes de Internet utilizados para el desarrollo de la memoria.

CAPÍTULO 2.
FUNDAMENTO TEÓRICO

2.1. Cinemática

2.1.1. Movimiento y trayectoria

Para estudiar el movimiento de un cuerpo hay que determinar, en primer lugar, su posición en el espacio en función del tiempo, para lo cual se necesita un sistema de referencia. Un cuerpo se encuentra en movimiento respecto de dicho sistema si su posición respecto a él cambia a lo largo del tiempo. El movimiento es un concepto relativo, de manera que resulta distinto para observadores que refieren la posición del cuerpo a sistemas de referencia distintos.

ECUACIÓN VECTORIAL DEL MOVIMIENTO

El movimiento de una partícula *P* se determina si se conoce la posición que ocupa la partícula en cualquier instante. Se consigue mediante el vector de posición trazado desde el origen de un sistema de referencia *XYZ* a la posición de la partícula. Las componentes *x*, *y*, *z* son las coordenadas del punto donde se encuentra la partícula. Si esta se mueve sobre la curva *C* su vector de posición varía con el tiempo:

$$\vec{r}(t) = x(t)\vec{i} + y(t)\vec{j} + z(t)\vec{k} \qquad [1]$$

La expresión [1] determina el movimiento de una partícula y se llama ecuación vectorial del movimiento.

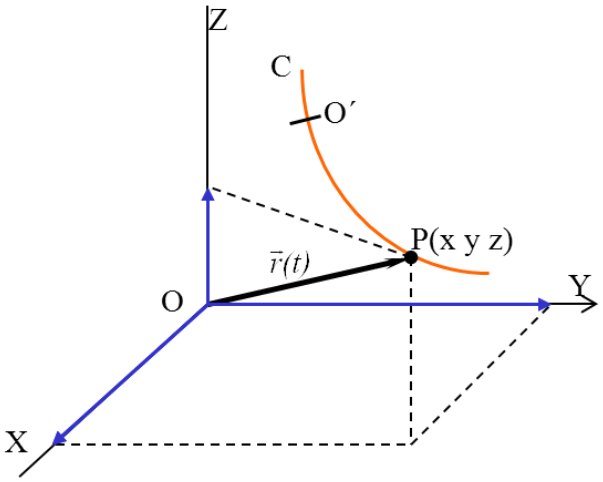


Figura 2.1. Posición de una partícula respecto a un sistema de referencia.

VECTOR DESPLAZAMIENTO

El **vector desplazamiento** entre dos instantes de tiempo es el incremento que experimenta el vector de posición de la partícula entre dichos instantes. Este vector se expresa así:

$$\Delta \vec{r}_{t_1}^{t_2} = \vec{r}(t_2) - \vec{r}(t_1) \qquad [2]$$

El espacio recorrido entre dos instantes es la distancia, medida sobre la trayectoria, que ha recorrido la partícula entre esos dos instantes.

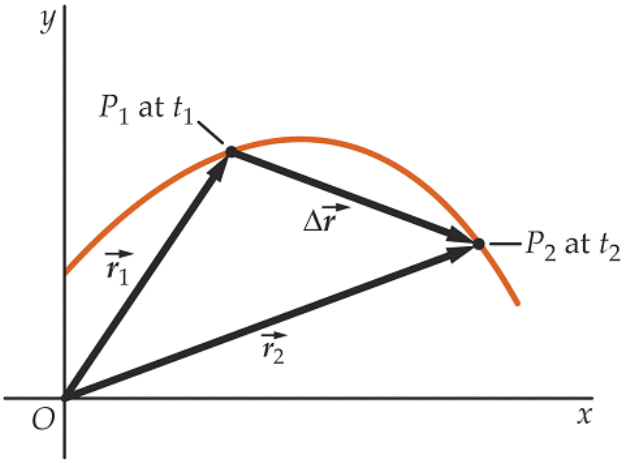


Figura 2.2. Espacio recorrido de una partícula.

2.1.2. Velocidad y aceleración

VECTOR VELOCIDAD

Si durante un intervalo de tiempo Δt la partícula realiza un desplazamiento $\Delta \vec{r}$, se define la **velocidad media** en ese intervalo de tiempo como:

$$\vec{v}_m = \frac{\Delta \vec{r}}{\Delta t} \qquad [3]$$

Desde el punto de vista físico, la expresión [3] carece de interés. Sin embargo, es importante conocer el significado físico del límite al que tiende la expresión [3] cuando el intervalo de tiempo considerado es tan pequeño que los dos puntos tienden a confundirse. Este límite define la **velocidad instantánea**.

$$\vec{v} = \lim_{\Delta t \to 0} \frac{\Delta \vec{r}}{\Delta t} = \frac{d\vec{r}}{dt} \qquad [4]$$

Las componentes cartesianas del vector velocidad son las derivadas respecto del tiempo de las respectivas componentes del vector de posición. La dirección es tangente a la trayectoria en la posición en la que se encuentra la partícula y el sentido es el del movimiento. El módulo del vector velocidad se llama **celeridad** y mide la rapidez con la que la partícula recorre la trayectoria.

$$\vec{v} = \frac{d\,x}{d\,t}\vec{i} + \frac{d\,y}{d\,t}\vec{j} + \frac{d\,z}{d\,t}\vec{k} \qquad [5]$$

$$|\vec{v}| = \sqrt{\left(\frac{d\,x}{d\,t}\right)^2 + \left(\frac{d\,y}{d\,t}\right)^2 + \left(\frac{d\,z}{d\,t}\right)^2} \qquad [6]$$

VECTOR ACELERACIÓN

En cada punto de la trayectoria se define un vector velocidad, que, en general, cambia tanto en módulo como en dirección. Cambia en módulo cuando la celeridad no es constante y en dirección cuando la trayectoria no es rectilínea. La magnitud que mide la variación del vector velocidad con el tiempo se denomina vector aceleración. Se define la **aceleración media** en el intervalo Δt, en el que el cambio en la velocidad es $\Delta\vec{v}$, como:

$$\vec{a}_m = \frac{\Delta\vec{v}}{\Delta t} \qquad [7]$$

La aceleración instantánea determina el límite al que tiende la expresión [7] cuando dos puntos tienden a confundirse sobre la trayectoria.

$$\vec{a} = \lim_{\Delta t \to 0} \frac{\Delta\vec{v}}{\Delta t} = \frac{d\,\vec{v}}{d\,t} = \frac{d^2\,\vec{r}}{d\,t^2} \qquad [8]$$

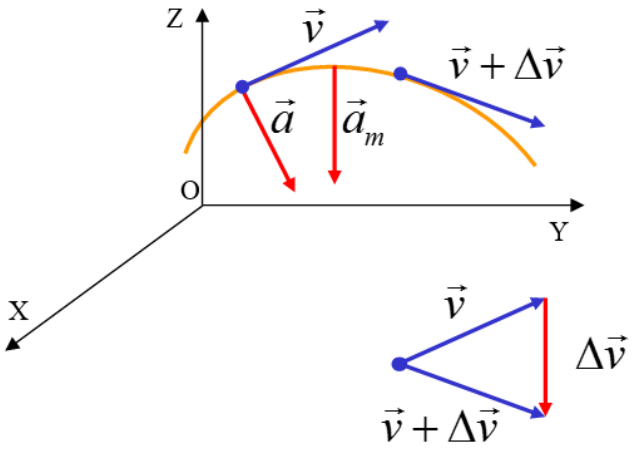


Figura 2.3. Sentido de los vectores velocidad y aceleración.

Como el vector $d\vec{v}$ se dirige hacia la parte cóncava de la trayectoria, el vector aceleración apunta siempre hacia el interior de la trayectoria, como se indica en la figura 2.3.

Las componentes cartesianas de la aceleración son:

$$\vec{a}=\frac{d\,v_x}{d\,t}\vec{i}+\frac{d\,v_y}{d\,t}\vec{j}+\frac{d\,v_z}{d\,t}\vec{k}=\frac{d^2\,x}{d\,t^2}\vec{i}+\frac{d^2\,y}{d\,t^2}\vec{j}+\frac{d^2\,z}{d\,t^2}\vec{k} \qquad [9]$$

COMPONENTES INTRÍNSECAS DE LA ACELERACIÓN

Todo vector puede expresarse como el producto de su módulo por el vector unitario en su dirección. De este modo, se escribe el vector velocidad de la siguiente forma:

$$\vec{v}=|\vec{v}|\,\vec{u}_v \qquad [10]$$

Si se deriva la expresión [10], se consigue la aceleración manifestada como suma de dos términos de esta manera:

$$\vec{a}=\frac{d\,\vec{v}}{d\,t}=\frac{d\,|\vec{v}|}{d\,t}\vec{u}_v+|\vec{v}|\frac{d\,\vec{u}_v}{d\,t} \qquad [11]$$

El primer término de [11] es tangente a la trayectoria por ser paralelo a $\vec{u}_v$ y el segundo término es normal a la misma (la derivada de un vector de módulo constante es perpendicular al vector que se deriva). Por esto, el vector aceleración puede descomponerse en una componente *tangente* y otra *normal* a la trayectoria. Estas componentes se llaman *intrínsecas* por estar

referidas a un sistema de referencia intrínseco al movimiento, el formado por la dirección tangente y la dirección normal a la trayectoria en cada punto.

El escalar de la componente tangencial de la aceleración es igual a $d|\vec{v}|/dt$, como se puede comprobar en [11]. Para calcular el escalar de la componente normal de la aceleración, se expresa el último término de [11] en función de $\vec{u}_n$, vector unitario normal a la trayectoria y dirigido hacia la parte cóncava de la misma.

En esta demostración se considera una trayectoria plana y recorrida en el sentido positivo. Los resultados son válidos para el caso general.

$$\vec{u}_v = \cos\phi\,\vec{i} + \operatorname{sen}\phi\,\vec{j} \qquad [12]$$

$$\vec{u}_n = \cos\left(\phi+\frac{\pi}{2}\right)\vec{i} + \operatorname{sen}\left(\phi+\frac{\pi}{2}\right)\vec{j} = -\operatorname{sen}\phi\,\vec{i} + \cos\phi\,\vec{j} \qquad [13]$$

$$\frac{d\vec{u}_v}{dt} = -\operatorname{sen}\phi\frac{d\phi}{dt}\vec{i} + \cos\phi\frac{d\phi}{dt}\vec{j} = \vec{u}_n\frac{d\phi}{dt} \qquad [14]$$

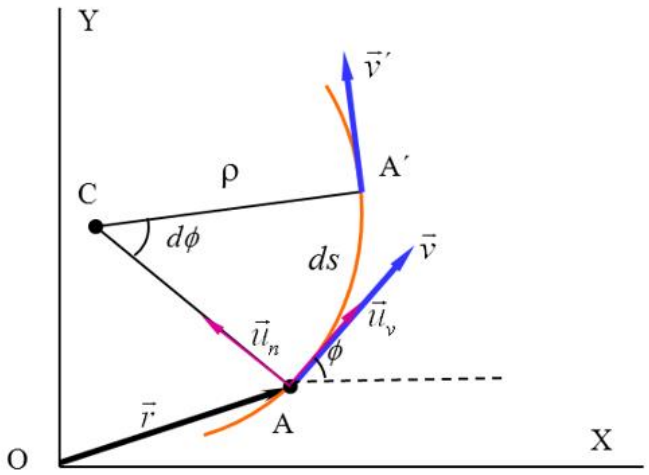


Figura 2.4. Representación de los vectores normal y tangencial.

Las normales a la trayectoria en los puntos *A* y *A'* se cortan en el centro de curvatura, *C*, siendo la distancia desde este punto a la curva el radio de curvatura, ρ. Si se considera que $ds = \rho\,d\phi$, entonces:

$$\frac{d\phi}{dt} = \frac{d\phi}{ds}\frac{ds}{dt} = \frac{|\vec{v}|}{\rho} \qquad [15]$$

Se deduce que:

$$\vec{a} = \frac{d|\vec{v}|}{dt}\vec{u}_v + \frac{|\vec{v}|^2}{\rho}\vec{u}_n = a_t\,\vec{u}_v + a_n\,\vec{u}_n \qquad [16]$$

El escalar de la aceleración tangencial, a_t, puede ser positivo (movimiento acelerado) o negativo (movimiento retardado), mientras que el escalar de la aceleración normal, a_n, es siempre positivo, porque la aceleración apunta hacia la parte cóncava de la trayectoria y el vector $\vec{u}_n$ se define señalando hacia esta parte.

La componente tangencial advierte de la variación en el módulo de la velocidad y la componente normal de la variación en la dirección de la velocidad, informando sobre la mayor o menor curvatura de la trayectoria.

2.1.3. Movimiento circular

En este caso la trayectoria es una circunferencia, por lo que el radio de curvatura es constante e igual al radio de la circunferencia. La aceleración normal es distinta de cero (excepto en aquellos puntos en los que sea cero la velocidad). La aceleración tangencial determina el tipo de movimiento.

$$\rho = cte = R \rightarrow a_n \neq 0 \begin{cases} a_t = 0 \rightarrow circular\ uniforme \\ a_t = cte \rightarrow circular\ uniformemente\ variado\ (acelerado\ si\ a_t > 0\ y\ retardado\ si\ a_t < 0) \\ a_t \neq 0 \rightarrow circular\ variado \end{cases}$$

Este movimiento se describe mediante el vector de posición. Si se considera la trayectoria en un plano paralelo al *XY* y centro de circunferencia en el eje *Z*, es de la siguiente forma:

$$\vec{r} = R\cos\theta\,\vec{i} + R\,\mathrm{sen}\,\theta\,\vec{j} + OO'\vec{k} \qquad [17]$$

Con $\theta = \theta(t)$ la **ecuación angular del movimiento**.

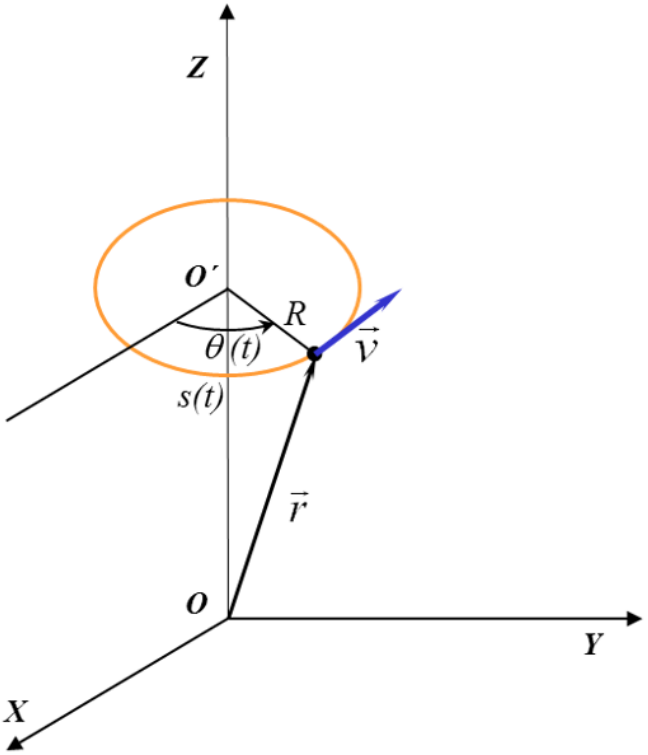


Figura 2.5. Componentes del movimiento circular de una partícula.

Si se conoce la trayectoria (el radio de la circunferencia), el movimiento de la partícula puede estudiarse a partir de su posición sobre la trayectoria. Esta posición puede estar determinada por $s = s(t)$ o por $\theta = \theta(t)$, figura 2.5. Estas expresiones deben estar relacionadas, porque determinan la misma posición en cada instante. La relación resulta de la definición de radián. Siempre que θ se exprese en radianes, se cumple:

$$s = \theta \cdot R \qquad [18]$$

A partir de la ecuación escalar del movimiento, se define la velocidad angular escalar y la aceleración angular escalar, llamadas generalmente **velocidad angular** y **aceleración angular**, respectivamente, de la siguiente manera:

$$\omega = \frac{d\theta}{dt} \qquad [19]$$

$$\alpha = \frac{d\omega}{dt} \qquad [20]$$

La relación con las correspondientes magnitudes lineales se obtiene a partir de [18].

$$v = \omega \cdot R \qquad [21]$$

$$a = \alpha \cdot R \qquad [22]$$

La descripción del movimiento con magnitudes angulares (θ, ω, α) es análoga a la descripción con magnitudes escalares (s, v, a), por lo que es necesario comparar los signos de ω y α para conocer si el movimiento es acelerado o retardado.

MAGNITUDES ANGULARES COMO VECTORES.
RELACIÓN CON LAS MAGNITUDES LINEALES

El vector velocidad angular se define de la siguiente manera: Su módulo es el valor absoluto de la velocidad angular puntualizado en [19]. Su dirección es perpendicular al plano de la trayectoria y su sentido coincide con el de avance de un sacacorchos que gira en el sentido del movimiento de la partícula. El vector velocidad angular se expresa de la siguiente forma:

$$\vec{\omega} = |\vec{\omega}|\, \vec{u}_{\omega} \qquad [23]$$

Si se considera que la dirección de $\vec{\omega}$ es constante, su derivada respecto al tiempo tiene su misma dirección.

$$\vec{\alpha} = \frac{d\,\vec{\omega}}{d\,t} = \frac{d\,|\vec{\omega}|}{d\,t}\vec{u}_{\omega} \qquad [24]$$

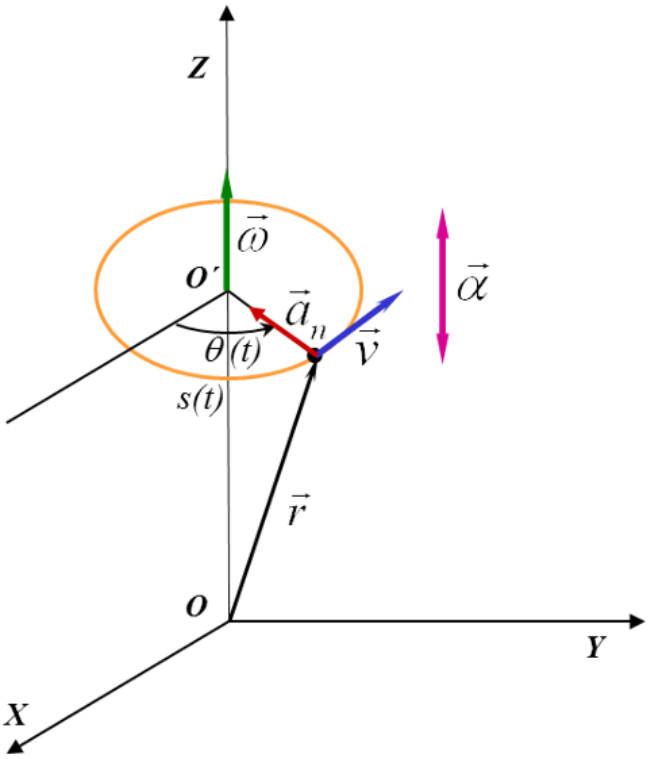


Figura 2.6. Representación de los vectores velocidad angular y aceleración angular.

La expresión [24] representa el **vector aceleración angular**, $\vec{\alpha}$. Su módulo es el valor absoluto de la aceleración angular definida en [20]. Su dirección perpendicular al plano de la trayectoria y el sentido serán el mismo (o el contrario) de la velocidad angular si $|\vec{\omega}|$ aumenta (o disminuye) con el tiempo.

Teniendo en cuenta las direcciones y sentidos de los vectores $\vec{r}$, $\vec{v}$ y $\vec{\omega}$ indicados en la figura 2.6, así como la relación entre sus valores numéricos expresada en la ecuación [21], se comprueba la siguiente relación:

$$\vec{v} = \vec{\omega} \wedge \vec{r} \qquad [25]$$

Para encontrar la relación entre aceleraciones se deriva respecto del tiempo la expresión [25].

$$\frac{d\vec{v}}{dt} = \frac{d\vec{\omega}}{dt} \wedge \vec{r} + \vec{\omega} \wedge \frac{d\vec{r}}{dt} \qquad [26]$$

$$\vec{a} = \vec{\alpha} \wedge \vec{r} + \vec{\omega} \wedge \vec{v} \qquad [27]$$

Analizando cada uno de los términos que aparecen en la expresión [27], se corrobora que coinciden con las componentes intrínsecas de la aceleración.

$$\vec{a}_t = \vec{\alpha} \wedge \vec{r} \qquad [28]$$

$$\vec{a}_n = \vec{\omega} \wedge \vec{v} \qquad [29]$$

2.1.3.1. Movimiento circular uniforme

En la situación en la que la aceleración tangencial sea cero $(a_t = 0)$, el movimiento a lo largo de una circunferencia o a lo largo de una trayectoria circular se denomina **movimiento circular uniforme**. El movimiento de una partícula es acelerado incluso cuando el módulo de la velocidad es constante. En la figura 2.7 se analizan los vectores posición y velocidad de una partícula que se mueve sobre una circunferencia a velocidad constante. El ángulo $\Delta\theta$ entre $\vec{v}(t)$ y $\vec{v}(t+\Delta t)$ es el mismo que entre $\vec{r}(t)$ y $\vec{r}(t+\Delta t)$, ya que los vectores posición y velocidad deben girar el mismo ángulo para conservar su perpendicularidad mutua. Con los dos vectores velocidad y con $\Delta\vec{v}$ se forma un primer triángulo isósceles. El segundo triángulo isósceles se forma con los dos vectores posición y con $\Delta\vec{r}$.

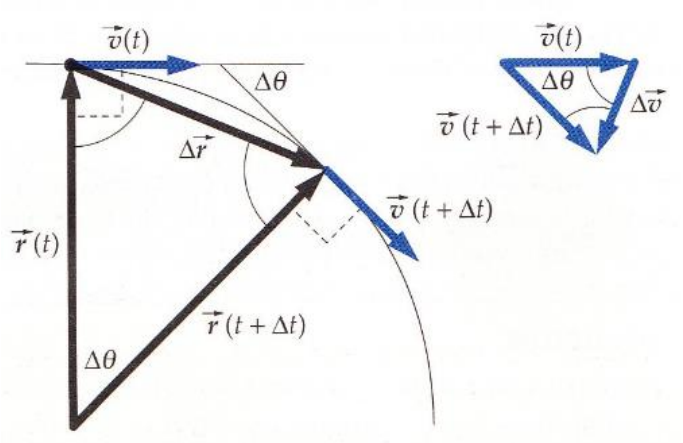


Figura 2.7. Vectores posición y velocidad de una partícula.

Para encontrar la dirección del vector aceleración, se examina el triángulo formado por los dos vectores velocidad y $\Delta\vec{v}$. La suma de los ángulos de un triángulo es 180º y los ángulos de la base de cualquier triángulo isósceles son iguales. En el límite en que Δt es muy pequeño, $\Delta\theta$ también es muy pequeño, por lo que en este límite los dos ángulos de la base se aproximan a 90º. Esto significa que $\Delta\vec{v}$ es perpendicular a la velocidad. Si $\Delta\vec{v}$ se dibuja en la posición de la partícula, señala en la dirección centrípeta.

Los dos triángulos son semejantes y las correspondientes longitudes de figuras geométricamente similares son proporcionales. Entonces,

$$\frac{|\Delta \vec{v}|}{|\Delta \vec{r}|} = \frac{v}{r} \qquad [30]$$

Multiplicando ambos lados por $|\Delta \vec{r}| / \Delta t$, se obtiene:

$$\frac{|\Delta \vec{v}|}{\Delta t} = \frac{v}{r} \frac{|\Delta \vec{r}|}{\Delta t} \qquad [31]$$

En el límite cuando Δt es muy pequeño, el término $|\Delta \vec{r}| / \Delta t$ se parece cada vez más al módulo de la aceleración instantánea a, y el término $|\Delta \vec{r}| / \Delta t$ se parece a v, el módulo de la velocidad instantánea. Entonces, en el límite cuando $\Delta t \to 0$, la ecuación [31] se transforma en $a = v^2 / R$. El vector aceleración está en la dirección centrípeta, así $a_c = a$, donde a_c es la componente del vector aceleración en la dirección centrípeta.

$$a_c = \frac{v^2}{R} \qquad [32]$$

2.2. Dinámica

2.2.1. 2ª ley de Newton o principio fundamental de la Dinámica

La velocidad de un cuerpo se modifica en el caso de que exista una interacción entre el cuerpo y el medio que lo rodea. La ley que rige esta situación fue establecida por Newton y puede enunciarse así: “En un sistema inercial, la resultante de las fuerzas que en un instante cualquiera actúan sobre una partícula es igual a la rapidez con que en ese instante varía el producto de la masa por la velocidad de la partícula”.

$$\vec{F} = \frac{d(m\vec{v})}{dt} \qquad [33]$$

Con $\vec{F}$ la resultante de las fuerzas que actúan sobre la partícula.

Si además la masa no varía con el tiempo (como ocurre en los sistemas de masa variable), la ecuación [33] se convierte en:

$$\vec{F} = m\frac{d\vec{v}}{dt} = m\vec{a} \qquad [34]$$

2.2.2. Fuerza centrípeta

Según la segunda ley de Newton en su forma vectorial $\sum \vec{F} = m\vec{a}$, la aceleración y la fuerza neta han de tener la misma dirección. En el caso de un movimiento circular con rapidez constante, la fuerza neta debe señalar hacia el centro del círculo.

Si una partícula se mueve con velocidad v a lo largo de una trayectoria curva con un radio de curvatura R , dicha partícula tiene una componente de la aceleración normal o centrípeta $a_c = v^2 / R$ en la dirección hacia el centro de curvatura.

En un movimiento circular uniforme,

$$\rho = cte = R \begin{cases} a_t = 0 \\ a_n = a_c = \dfrac{v^2}{R} \end{cases}$$

Por consiguiente, la componente de la fuerza tangente a la trayectoria, o fuerza tangencial, es:

$$F_t = M\, a_t = \frac{d\left|\vec{v}\right|}{dt} = 0 \qquad [35]$$

Y la componente de la fuerza perpendicular a la trayectoria, llamada fuerza normal o centrípeta, es:

$$F_n = M\, a_n = M\,\frac{v^2}{R} \qquad [36]$$

La fuerza tangencial es la responsable del cambio del módulo de la velocidad, mientras que la fuerza centrípeta o normal es la responsable del cambio de la dirección de la velocidad. Cuando la fuerza tangencial es cero, no hay aceleración tangencial y lo que se tiene es un movimiento curvilíneo uniforme. Si además la fuerza centrípeta es constante, el movimiento es

circular. Cuando esta fuerza es cero, no hay aceleración normal y el movimiento es rectilíneo.

En el movimiento circular, $v = \omega \cdot R$, de modo que la fuerza centrípeta se puede escribir también como:

$$F_n = M\, a_n = M\, \frac{v^2}{R} = M\, R\, \omega^2 \qquad [37]$$

El radio de la circunferencia es R. En el movimiento circular uniforme, la única fuerza es F_n. La fuerza centrípeta es el nombre que se le da a la componente de la fuerza resultante en la dirección normal a la trayectoria. De la misma manera que la fuerza neta, la fuerza centrípeta, como fuerza resultante en la dirección normal, no aparece en el diagrama de fuerzas. Sólo deben aparecer las fuerzas reales.

CAPÍTULO 3.
EQUIPO, HARDWARE Y SOFTWARE

3.1. Material de laboratorio

3.1.1. Sistema mecánico rotacional ME-8950 de Pasco

Este sistema ofrece diversos experimentos para analizar la fuerza centrípeta, momento angular y movimiento rotacional. Un set de accesorios le otorga una herramienta ideal para estudiar el par de torsión, la fricción, la levitación magnética y la ley de Faraday.

Figura 3.1. Componentes del sistema ME-8950.

Características:

El sistema perite montar sensores para medir datos de la posición angular, velocidad y aceleración. También se puede acoplar un motor para que la plataforma gire con una determinada velocidad constante. Además, tiene las siguientes singularidades.

- Base de hierro fundido de cuatro kilogramos para la estabilidad.
- Rodamientos de bolas.
- Eje de acero inoxidable de precisión.

Especificaciones:

- Base: De hierro fundido de cuatro kilogramos, con la posibilidad de nivelación.
- Plataforma giratoria: Extrusión de aluminio, 24 centímetros de longitud útil (26 centímetros en total).
- Eje: De acero inoxidable, 1,25 centímetros de diámetro, 18 centímetros aproximadamente de longitud, *moleteado* (superficie labrada con estrías cruzadas).

- Disco: PVC duro, rodamiento de bolas central, 25,4 centímetros, 1500 gramos.
- Anillo: De acero, 12,7 centímetros de diámetro exterior, 1420 gramos.

Asimismo, el equipo incluye los siguientes materiales:
- Plataforma de rotación (ME-8951).
- Accesorio de fuerza centrípeta (ME-8952).
- Accesorio de inercia rotacional (ME-8953).
- Manual de instrucciones.

Para este TFG se utiliza la plataforma de rotación y parte de los accesorios de fuerza centrípeta. Aquellos que son de inercia rotacional no se necesita.

3.1.2. Plataforma de rotación ME-8951 de Pasco

La base estable y los rodamientos de la plataforma proporcionan el sistema rotacional para la ejecución de experimentos generales. Por esta razón, la plataforma es el componente principal del equipo. Si se combina con accesorios de movimiento rotacional, el sistema es adecuado para realizar experimentos sobre el par de torsión, la fricción, la levitación magnética y la ley de Faraday.

Figura 3.2. Plataforma.

Incluye:
- Plataforma de aluminio para el montaje de experimentos de rotación.
- Dos masas cuadradas de 300 gramos. No se utilizan para este Proyecto.
- Base sólida de hierro fundido de cuatro kilogramos.
- Barra de apoyo.

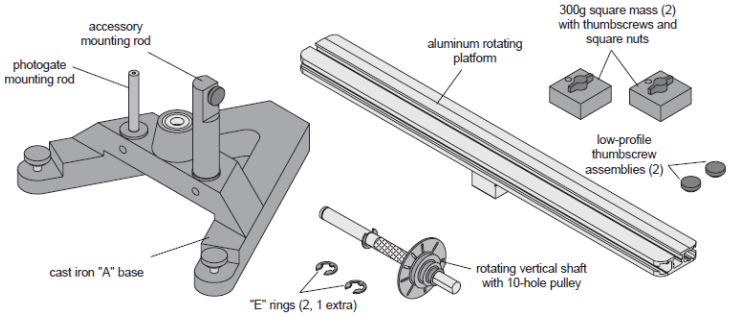


Figura 3.3. Elementos de la plataforma.

3.1.3. Motor rotacional ME-8955 de Pasco

Es un accesorio para el sistema, que sirve para girar la plataforma, a velocidad constante. El motor requiere una fuente de alimentación de 15 voltios DC como máximo, que puede ser variable.

Figura 3.4. Motor rotacional.

Incluye:

- Motor.
- Polea de tres pasos.
- Correa de transmisión.

Especificaciones:

- 15 voltios máximo.
- 0,2 amperios mínimo.
- 1,5 amperios máximo.
- Rango de velocidad del eje: 10 – 600 rpm.

3.1.4. Adaptador rotacional base “A” CI-6690 de Pasco

Este adaptador permite el montaje del sensor de movimiento rotatorio a la plataforma giratoria. Una vez montado, el sensor mide la rotación de cualquier experimento, siempre que se use un hardware y software adecuados. Una revolución del eje vertical se corresponde con una revolución del sensor de movimiento, dando hasta 4000 datos por revolución.

Figura 3.5. Adaptador.

3.1.5. Sensor de movimiento rotatorio CI-6538 de Pasco

Este sensor es uno de los dispositivos de medición posición/movimiento más versátiles para los alumnos que utilizan las prácticas de laboratorio. Puede medir la posición lineal con una resolución de 0,055 milímetros o un movimiento rotatorio con una resolución de 0,25°. También es bidireccional, indicando la dirección del movimiento.

El eje de 6,35 milímetros, con doble cojinete de bolas, se extiende desde ambas caras del sensor y proporciona un gran soporte para los experimentos del equipo. Su abrazadera, que puede ser fijada en las tres caras del sensor, permite montarlo en cualquier orientación.

El núcleo del sensor es un codificador óptico. Por esto, es una tecnología digital y no hay deriva o error acumulativo. El cero siempre indica el mismo punto. La resolución, 1° o 0,25°, es detectada por software.

Figura 3.6. Sensor de movimiento.

Especificaciones:

- Polea de tres pasos: 10, 29 y 48 milímetros de diámetro.
- Dimensiones del sensor: 10 cm x 5 cm x 3,75 cm, con eje de 6,35 mm de diámetro.
- Resolución:
    - 1°/0,087 mm.
    - 0,25°/0,022 mm (seleccionable por software).
- Resolución de rotación: -
- Velocidad máxima de rotación: 13 rev/segundo a 1° de resolución (360 puntos/revolución); 3,25 rev/segundo a 0,25° de resolución (1440 puntos/resolución).
- Codificador (generador de pulsos) óptico: Bidireccional, indica la dirección del movimiento.

3.1.6. Sensor de fuerza CI-6746 de Pasco

Es ideal para realizar mediciones de fuerza en prácticas de laboratorio. Los orificios para los dedos lo convierten en un sensor práctico para uso manual. Además, puede montarse en accesorios de Pasco, como por ejemplo el carro dinámico. Es perfecto para medir la fuerza en equipos de un eje.

Accesorios:

- Accesorio de parachoques.
- Accesorio de gancho.
- Soporte tornillo de fijación.
- Varilla ajustable.

No se usan el accesorio de parachoques, el accesorio de gancho ni la varilla ajustable en este Proyecto.

Figura 3.7. Sensor de fuerza.

Especificaciones:

- Rango: ±50 N.
- Resolución: 0,03 N.
- Frecuencia máxima de muestreo: Depende de la interfaz.
- Protección de sobrecarga de fuerza: Hasta 75 N sin daños.
- Deflexión del haz: 0,28 milímetros.
- Opciones de montaje:
    - En carros Pasco.
    - En varillas estándar de 12,7 milímetros.
- Tensión de salida:
    - +8 V para +50 N (compresión).
    - −8 V para −50 N (tracción).
- Ruido de salida: ±2 mV.
- Tasa máxima de cambio en la fuerza: 30 N/ms.
- Límite de ancho de banda: 2 kHz (filtro de paso bajo interno).

3.1.7. Accesorio de fuerza centrípeta ME-8089 de Pasco

Este producto proporciona los materiales para el cálculo de la velocidad angular de una masa durante el giro de la plataforma.

Características:

- Mediciones basadas en computadora.
- Variación de las masas. El peso a medir puede ser cambiado añadiendo masas al *portamasas*. El radio está determinado por la posición vertical del sensor de fuerza. Aumenta o disminuye el radio al bajar o subir el sensor, respectivamente.

Figura 3.8. Accesorios para calcular la velocidad.

Incluye:

- Bloque pivote-polea.
- Cable de conexión con rodamiento giratorio.

- Dos *portamasas*.
- Cuatro masas de 50 gramos.
- Dos masas de 100 gramos.

Una ranura a lo largo de la longitud en la plataforma de rotación sirve para mover y ajustar los dos *portamasas*. Uno de estos *portamasas*, que se mueve libremente por la ranura, está unido al cable conectado al sensor de fuerza. El cable pasa por debajo de la polea para que no se enrede al girar el brazo de la plataforma. El otro *portamasas*, fijo, se coloca a la misma distancia del eje de rotación que la del *portamasas* libre. De este modo se equilibran los brazos de la plataforma cuando gira.

3.1.8. Adaptador analógico ELVIS CI-6718 de Pasco

Este adaptador permite que los sensores analógicos Pasco *ScienceWorkshop* puedan utilizarse para la interfaz del paquete informático NI ELVIS (*Educational Laboratory Virtual Instrumentation Suite)* de LabVIEW. El paquete combina perfectamente instrumentación y adquisición de datos.

Para la recogida de datos, el adaptador se encaja a una placa *protoboard* y el cable del sensor se conecta al adaptador. Una vez que todos los datos han sido adquiridos, LabVIEW los muestra y analiza según un determinado programa. Este sistema ofrece un buen procedimiento para que los estudiantes aprendan las técnicas de procesado de datos analógicos.

Figura 3.9. Adaptador para un sensor analógico.

3.1.9. Adaptador digital ELVIS CI-6719 de Pasco

Este adaptador posibilita que los sensores digitales Pasco *ScienceWorkshop* se usen para la interfaz del paquete informático NI ELVIS de LabVIEW.

Al igual que en el adaptador analógico, se encaja a una placa *protoboard* y los cables del sensor se conectan al adaptador para adquirir datos. LabVIEW

los analiza según un programa. Es ideal para comprender las técnicas de procesado de datos digitales.

Figura 3.10. Adaptador para un sensor digital.

Especificaciones:

- Puertos: Dos, tipo *ScienceWorkshop*.

3.1.10. Tarjeta de adquisición de datos NI USB-6210

Esta tarjeta es un módulo de adquisición de datos (DAQ) multifunción de la serie M, alimentado por bus y optimizado para una precisión superior a velocidades de muestreo altas. Dispone de 16 entradas analógicas, velocidad de muestreo de un solo canal de 250 kS/s, cuatro líneas de entrada digital, cuatro líneas de salida digital, cuatro rangos de entrada programable (±0,2 V a ±10 V) por canal, disparo digital y dos contadores/temporizadores. Su diseño es perfecto para aplicaciones móviles o con restricciones de espacio. La instalación *plug-and-play* minimiza el tiempo de configuración y montaje.

Software controlador

El controlador *NI-DAQmx* y el software de servicios de medida de LabVIEW ofrecen interfaces de programación y configuración fáciles de usar. Por ejemplo, el *DAQ Assistant* de LabVIEW ayuda a reducir el tiempo de desarrollo. Hay que destacar que los dispositivos de la serie M no son compatibles son el software controlador *Traditional NI-DAQ (Legacy)*.

Software de aplicación

Cada dispositivo de adquisición de datos de la serie M incluye una copia de NI LabVIEW *SignalExpress* para que se adquieran, analicen y representen los datos rápidamente sin programar. Estos dispositivos son compatibles son las siguientes versiones (o posteriores) de software de NI: LabVIEW 7.1, LabWindows/CVI 7.x y Measurement Studio 7.x. Los dispositivos DAQ de la

Serie M también son compatibles con Visual Studio .NET, C/C++ y Visual Basic 6.0.

Figura 3.11. Tarjeta NI USB-6210.

Especificaciones globales:

General

- Familia de productos: DAQ Multifunción.
- Tipo de medida: Voltaje.
- *Form factor*: USB.
- Sistema operativo: Linux, Mac OS, Windows.
- Compatible con RoHS: Sí.
- Tipo de aislamiento: Ninguno.

Entrada analógica

- Canales de un solo terminal: 16.
- Canales diferenciales: 8.
- Resolución de entrada analógica: 16 bits.
- Rango de voltaje máximo:
  - Rango: De –10 V a +10 V.
  - Precisión: 2,69 mV.
  - Sensibilidad: 91,6 µV.
- Rango de voltaje mínimo:
  - Rango: De –200 mV a +200 mV.
  - Precisión: 0,088 mV.
  - Sensibilidad: 4,8 µV.
- Número de rangos: 4.
- Muestreo simultáneo: No.

- Memoria interna: 4095 muestras.

Salida analógica
- Número de canales: 0.

E/S digital
- Canales bidireccionales: 0.
- Canales sólo de entrada: 4.
- Canales sólo de salida: 4.
- Temporización: Software.
- Niveles lógicos: TTL.
- Filtros de entrada programables: No.
- Soporte de estados de encendido programables: Sí.

Entrada digital
- Tipo de entrada: *Sinking sourcing*.
- Rango de voltaje máximo: De 0 V a 5,25 V.

Salida digital
- Tipo de salida: *Sinking sourcing*.
- Capacidad de corriente simple: 16 mA.
- Capacidad de corriente total: 50 mA.
- Rango de voltaje máximo: De 0 V a 3,8 V.

Contadores/Temporizadores
- Temporizador *watchdog*: No.
- Contadores: 2.
- Operaciones a *buffer*: Sí.
- *Debouncing/Glitch Removal*: Sí.
- Frecuencia máxima de la fuente: 80 MHz.
- Generación de pulso: Sí.
- Tamaño: 32 bits.
- Estabilidad de tiempo: 50 ppm.
- Niveles lógicos: TTL.

Temporización/Disparo/Sincronización
- Disparo: Digital.

Especificaciones Físicas
- Longitud: 16,9 cm.
- Ancho: 9,4 cm.
- Altura: 3,1 cm.

- Conector de E/S: Terminales de tornillo.
- Potencia USB: Energizado por bus.

3.1.11. Fuente de alimentación E3646A de Agilent

Este aparato compacto y pequeño ofrece un bajo nivel de ruido de salida. Sirve para aplicaciones de uso general.

Figura 3.12. Fuente de alimentación.

Especificaciones:

- Dos posiciones de salida (0 a 40 ºC).
  - Rango 1: 0 a 8 V / 3 A.
  - Rango 2: 0 a 20 V / 1,5 A.

- Exactitud de programa (25 ºC ± 5 ºC), ± (% output + offset).
  - Voltaje: < 0,05% + 10 mV (< 0,1% + 25 mV para salida 2 de E3646/47/48/49A).
  - Intensidad: < 0,2% + 10 mA.

- Ondulación y ruido 20 Hz a 20 MHz.
  - Voltaje modo normal: < 5 mVpp / 0,5 mVrms.
  - Intensidad modo normal: < 4 mArms.

- Exactitud de lectura (25 ºC ± 5 °C), ± (% output + offset).
  - Voltaje: < 0,05% + 5 mV (< 0,1% + 25 mV para salida 2 de E3646/47/48/49A).
  - Intensidad: < 0,15% + 5 mA (< 0,15% + 10 mA para salida 2 de E3646/47/48/49A).

3.1.12. Cable de conexión GPIB/USB 82357B de Agilent

Esta interfaz proporciona una conexión directa desde un puerto USB de la computadora al instrumento GPIB. No requiere tarjetas ni fuentes de alimentación externas.

Figura 3.13. Cable GPIB/USB.

Especificaciones:

- Interfaz *plug and play*.
- Interfaz USB 2.0 (compatible con USB 1.1) e interfaz IEEE-488 (se une hasta con 14 instrumentos GPIB).
- Alta velocidad de transferencia, 1,15 MB/s.
- Puede comprobar la respuesta de hasta 8 dispositivos a la vez.
- Librerías estándar de Keysight, de Agilent.

3.1.13. Otros componentes

Batería de 12 V de Yuasa

Se necesitan dos baterías iguales para la alimentación del sensor de fuerza.

Especificaciones:

- Tipo de carcasa: Ind AGM.
- Tensión: 12 V.
- Capacidad a velocidad de 20 horas (Ah): 1,2.
- Rendimiento de arranque en frío (amperios) EN1: NA.
- Tensión nominal: 12 V.
- Capacidad Ah: 1,2.
- Disposición de las celdas: 3.

- Terminal: A.
- Velocidad de carga recomendada (amperios): 0,1.
- Longitud (mm): 97.
- Anchura (mm): 48.
- Altura (mm): 54,5.
- Altura total (mm): 54,5.
- Peso (kg): 0,58.

Figura 3.14. Batería de 12 voltios.

Cable USB 2.0
Tipo A (macho) a conexión Tipo B (macho).

Figura 3.15. Cable USB.

Placa de pruebas (*protoboard*)
Para las conexiones de los cables.

Figura 3.16. Placa *protoboard*.

Nivel de plástico de Ratio
Con base magnética y tres burbujas. Su largura es de 21 centímetros.

Figura 3.17. Nivel.

Bases, barras y nueces para barras
Para el sensor de fuerza.

Figura 3.18. Una base, barras y una nuez.

3.2. Hardware

Estos son los requisitos mínimos que debe tener una computadora (PC, portátil, *netbook*...) para la ejecución correcta de LabVIEW en un sistema operativo Windows.

	Windows	
	Run-Time Engine	**Entorno de desarrollo**
Procesador	Pentium III/Celeron de 866 MHz (o equivalente) o posterior (32 bits) Pentium 4 G1 (o equivalente) o posterior (64 bits)	Pentium 4M (o equivalente) o posterior (32 bits) Pentium 4 G1 (o equivalente) o posterior (64 bits)
RAM	256 MB	1 GB
Resolución de pantalla	1024 x 768 píxeles	1024 x 768 píxeles
SO	Windows 10/8.1/8/7 SP1 (32 y 64 bits) Windows Server 2012 R2 (64 bits) Windows Server 2008 R2 SP1 (64 bits)	Windows 10/8.1/8/7 SP1 (32 y 64 bits) Windows Server 2012 R2 (64 bits) Windows Server 2008 R2 SP1 (64 bits)
Espacio en disco	620 MB	5 GB (Incluye controladores predeterminados del DVD de Controladores de Dispositivos de NI)

Tabla 3.1. Especificaciones mínimas de una computadora.

3.3. Software

Este Proyecto usa la versión 17 de LabVIEW, un programa basado en un entorno de desarrollo integrado y diseñado específicamente para el sector científico y de la ingeniería. El lenguaje de programación gráfica (G) es nativo de este programa. Utiliza un modelo de flujo de datos en lugar de líneas secuenciales de código de texto. Esto permite escribir código funcional usando un diseño visual, de líneas y bloques.

Además, es necesario la instalación del controlador NI-DAQmx. Se utiliza la versión 17.0.0. Este módulo cubre los requisitos de los dispositivos de National Instruments para adquisición de datos. Maximiza el rendimiento de la computadora y del dispositivo de medida. Es de alto rendimiento. Incluye medidas adicionales como *Measurement & Automation Explorer* (MAX), generación de código *DAQ Assistant* y el software de medida basado en configuración *LabVIEW Signal Express*. Este software brinda:

- Una sola interfaz de programación para programar entradas analógicas, salidas analógicas, E/S digital y contadores en cientos de dispositivos de hardware DAQ multifunción.

- Los mismos VI (*Virtual Instrument*) y funciones en NI LabVIEW, NI LabWindows/CVI, Visual Basic, Visual Studio .NET y C/C++.
- *NI Measurement & Automation Explorer*, *DAQ Assistant* y software LabVIEW *SignalExpress LE* para ahorrar tiempo de configuración, desarrollo y registro de datos.

También se deben instalar las librerías y controladores de la fuente de alimentación para la comunicación con la computadora (*Benchvue* software). El enlace de descarga está en la bibliografía. Asimismo, se deben instalar los drivers de la fuente, ya que contienen los bloques para trabajar en LabVIEW (epígrafe *6.2.5. Control de la fuente de alimentación*).

CAPÍTULO 4.
MONTAJE DEL EQUIPO

PLATAFORMA DE ROTACIÓN ME-8951

1. Insertar el extremo cilíndrico del eje en los cojinetes de la cara superior de la base de hierro en forma de “A”. Asegurar el eje en el lugar mediante la colocación del anillo “E” en la ranura de la parte inferior del eje.
2. Montar la plataforma al eje y apretar el tornillo contra el lado plano de la forma “D” en el eje.

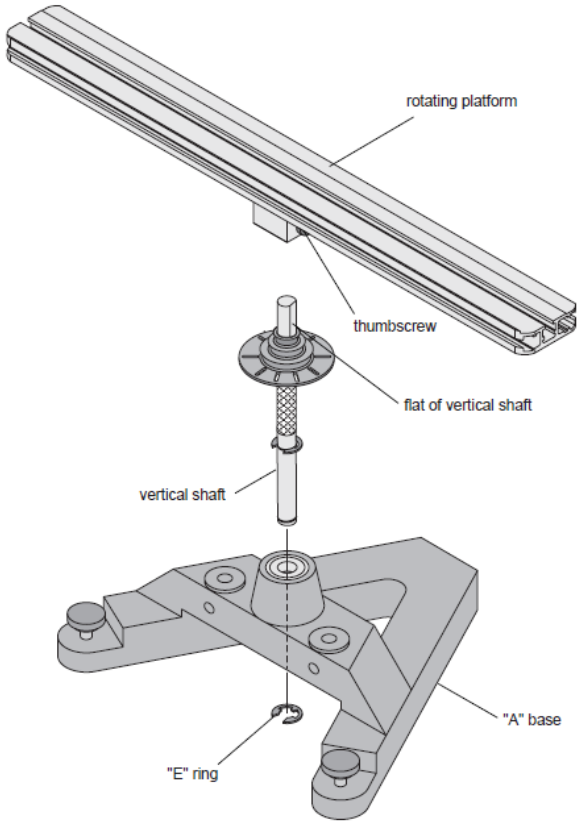


Figura 4.1. Ensamblaje del eje vertical en la base y en la plataforma.

Nivelación de la base

Para que el equipo funcione correctamente, la base debe estar perfectamente nivelada, puesto que si no lo está, los resultados pueden verse afectados y ser erróneos. Para nivelar, se ejecutan los siguientes pasos:

1. Sitúe una masa cuadrada de 300 gramos al final de la plataforma y apriete el tornillo para que no se deslice.

2. Ajuste el tornillo de nivelación de uno de los pies de la base hasta el final de la plataforma cuando la masa está alineada sobre el tornillo del otro pie.

3. Gire la plataforma 90 grados para que se sitúe de forma paralela a un lado de la base “A” y ajuste el otro tornillo hasta que la plataforma se mantenga en esta posición. Use un nivel para comprobar la posición horizontal correcta.

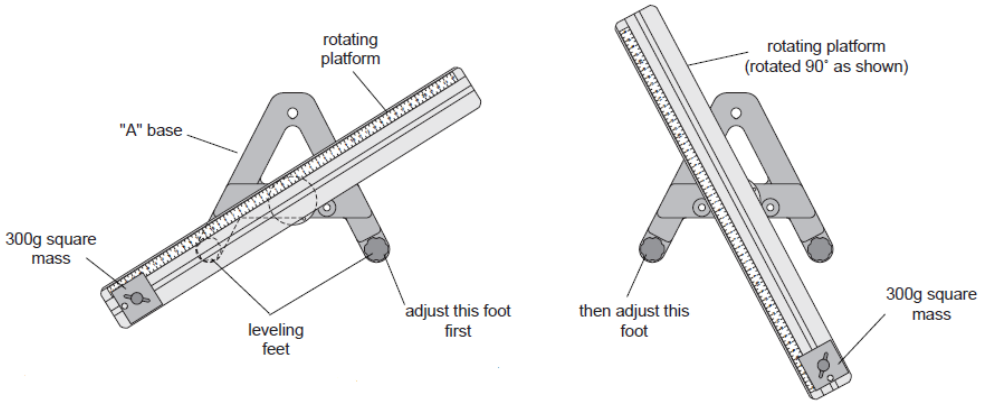


Figura 4.2. Nivelación de la base.

MOTOR ROTACIONAL ME-8955

1. Encajar el motor en uno de los agujeros situados sobre la base de la plataforma “A”. La orientación del motor se distingue en la figura 4.3.

2. Ajuste la correa de transmisión alrededor del eje vertical de la plataforma giratoria y en una de las tres poleas del motor.

3. Gire el motor en su orificio de montaje hasta que la correa esté tensa. Apriete el tornillo de montaje. Tenga en cuenta la orientación del motor.

4. Conecte el motor a una fuente de alimentación de corriente continua.

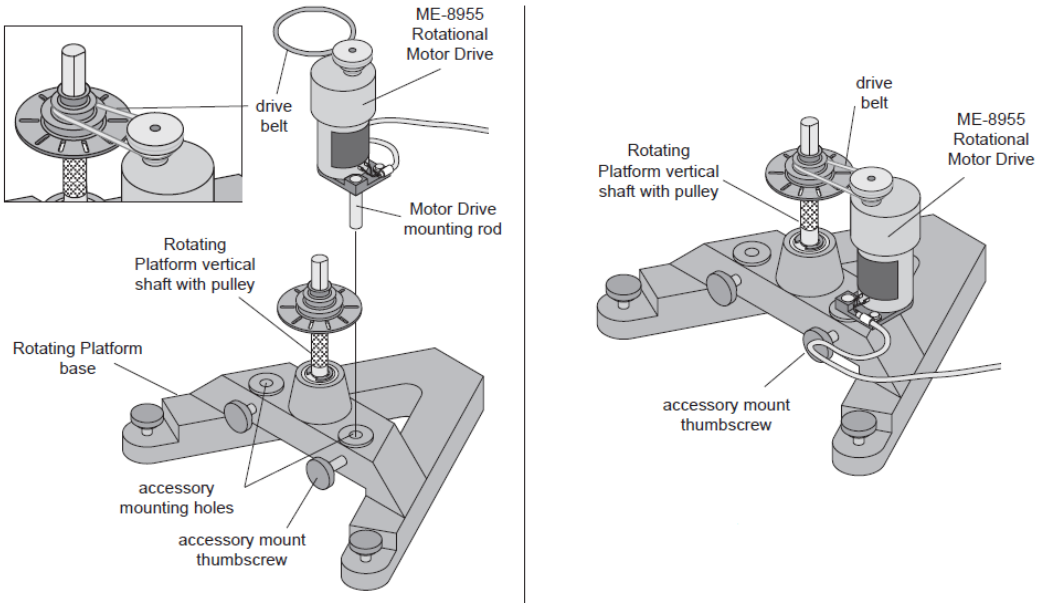


Figura 4.3. Colocación y ajuste del motor.

ADAPTADOR ROTACIONAL BASE "A" CI-6690
Y SENSOR DE MOVIMIENTO ROTATORIO CI-6538

El adaptador incluye un soporte, una polea de tres pasos, una correa de transmisión (anillo "O") y un tornillo.

1. Coloque el apoyo del adaptador en el orificio que queda libre sobre el soporte de la base "A".

2. Ajuste por el tornillo de la base.

3. Monte el sensor de movimiento en la barra del adaptador y sujételo mediante el tornillo del sensor.

4. Coloque la polea de tres pasos, la correa de transmisión y el tornillo en la parte superior de la base "A" tal como indica la figura 4.5. Conecte la correa (anillo "O") al eje giratorio.

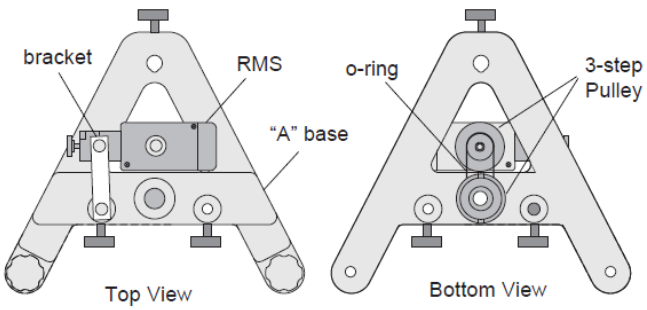


Figura 4.4. Montaje del sensor de movimiento en la base "A".

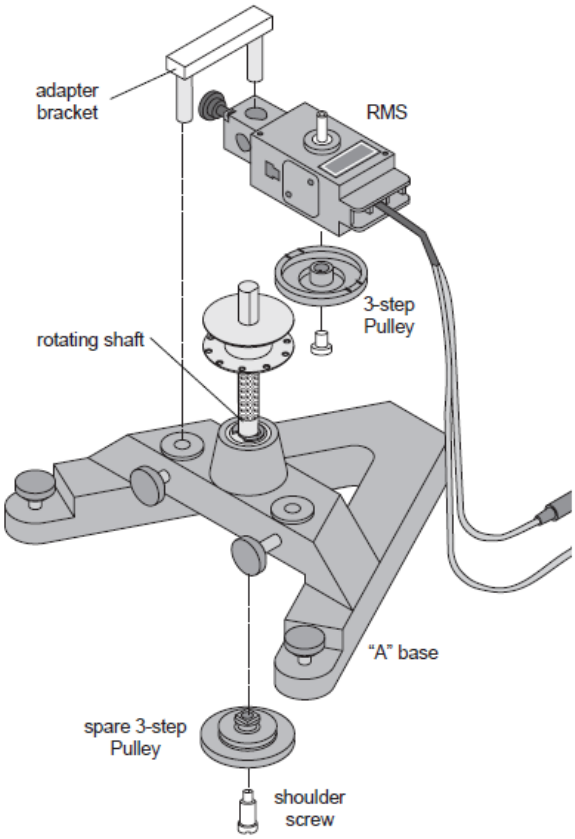


Figura 4.5. Ajuste del sensor de movimiento y del adaptador.

SENSOR DE FUERZA CI-6746
Y ACCESORIO DE FUERZA CENTRÍPETA ME-8089

1. Deslice el bloque pivote-polea a través de la ranura para situarlo en el centro de la plataforma. El bloque tiene dibujado una marca para facilitar su correcta posición en el cero de la cinta métrica pegada en la plataforma.

2. Deslice el *portamasas* fijo en la plataforma. Ajuste y apriete las tuercas en forma de "T" en la ranura.

3. Inserte el *portamasas* deslizante en la ranura, de modo que la cara con la señal mire la cinta métrica.

4. El sensor se sitúa sobre la plataforma giratoria. Como esta plataforma es alargada, el medidor debe sujetarse en una estructura auxiliar apoyada en otra base distinta de la base "A". La estructura se compone de dos barras verticales, sujetas en dos bases, y otra barra horizontal unida a esas dos barras. (figuras 4.7 y 4.8).

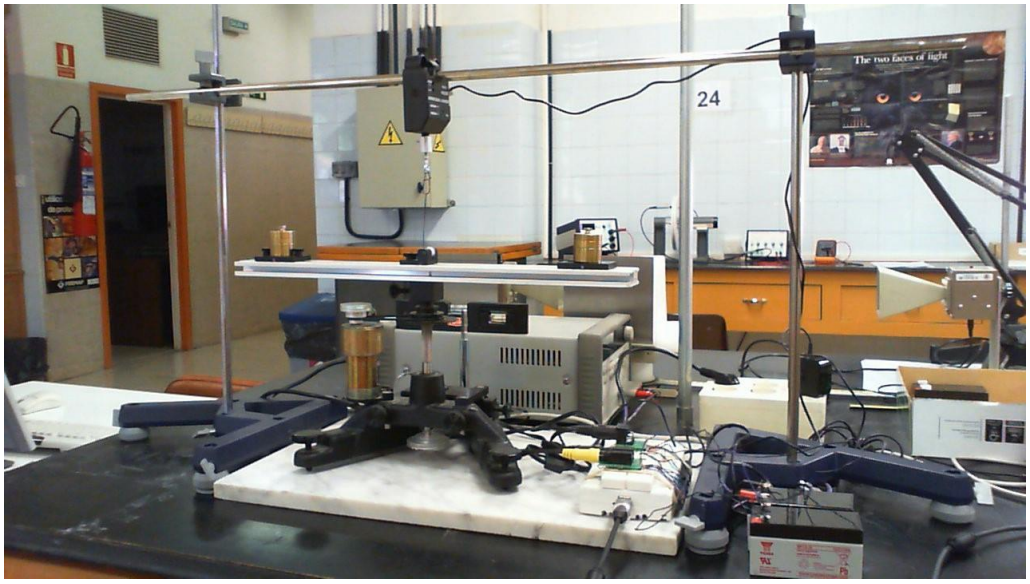


Figura 4.6. Equipo completo de medición de la fuerza centrípeta.

5. Deslice el sensor de fuerza sobre la barra horizontal y apriete el tornillo superior para sujetarlo. Asegúrese de que los cables del sensor se mantengan alejados de los brazos de la plataforma giratoria. Debe situarse exactamente encima de la polea de rodamiento giratorio, en el centro de la plataforma.

6. Fije el cable de conexión con rodamiento giratorio en la parte inferior del sensor.

7. Atraviese el cable unido en el bloque pivote-polea de rodamiento giratorio (figura 4.9).

Figura 4.7. Base de la estructura auxiliar.

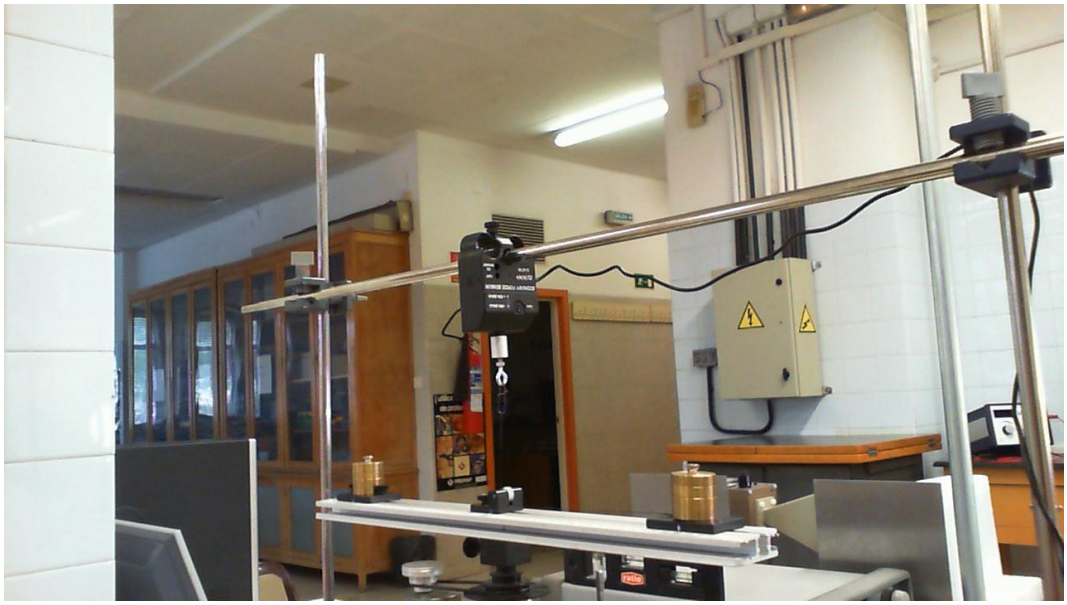


Figura 4.8. Posición del sensor de fuerza.

Figura 4.9. Cable que pasa por el bloque pivote-polea hacia el *portamasas* deslizante.

8. Añada una o varias masas sobre el *portamasas* deslizante. Siempre se coloca el cable antes de añadir la masa. Enrosque la tuerca para asegurar la masa.

9. Añada una o varias masas al *portamasas* fijo y sujétela mediante la tuerca. Sujete el *portamasas* mediante los tornillos.

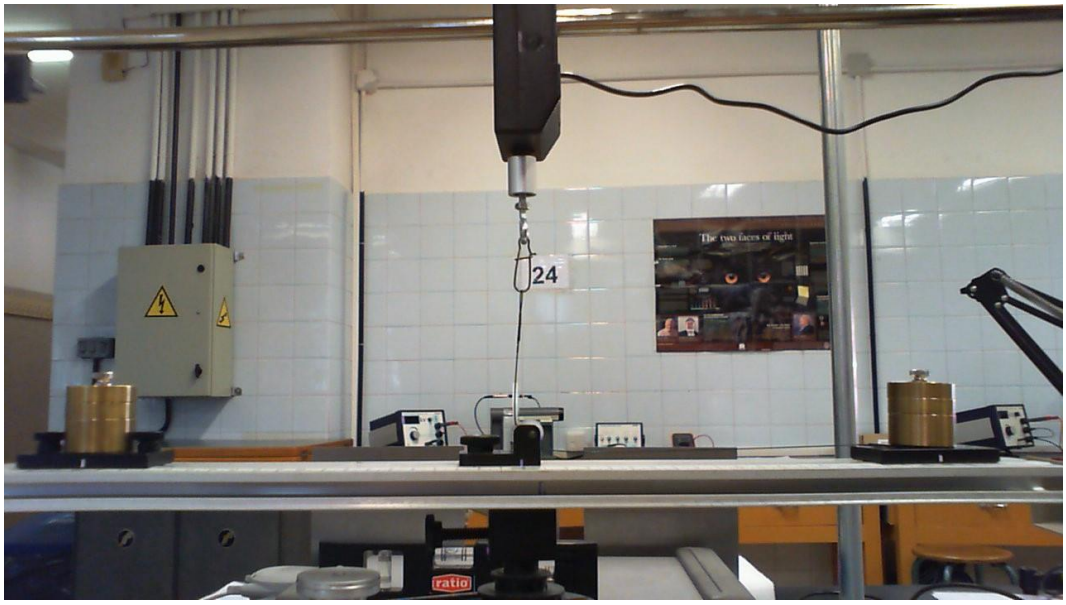


Figura 4.10. Ajuste de los dos *portamasas*.

10. El *portamasas* fijo y el *portamasas* deslizante tienen que situarse a la misma distancia del eje de rotación de la plataforma.

Es importante recordar que todos los cables deben mantenerse fuera del recorrido de los brazos de la plataforma.

CAPÍTULO 5.
ADQUISICIÓN DE DATOS Y CONFIGURACIÓN DE LOS SENSORES

5.1. Adquisición de datos

5.1.1. Conceptos básicos sobre los sistemas de adquisición de datos

Son muchas las aplicaciones donde es necesario el tratamiento de señales que proporcionen información sobre fenómenos físicos. En general, el tratamiento es imprescindible en grandes cantidades de información y con una elevada velocidad de procesado. Una computadora es la encargada de realizar estas tareas, debido a su excelente velocidad de procesado sobre grandes cantidades de información. Comúnmente, los dispositivos usados para la obtención de datos son las tarjetas de adquisición de datos, que proporcionan a la computadora la capacidad de adquirir y generar señales, ya sean analógicas o digitales. Sin embargo, estas no son las únicas funciones de estas tarjetas. Entre otras, también disponen de contadores y temporizadores.

Cuando se desea obtener información sobre fenómenos físicos es necesario introducir un nuevo elemento en el sistema que suministre un parámetro eléctrico a partir de un parámetro físico: el transductor, captador o sensor. Es el primer elemento que forma un sistema general de adquisición de señales. Generalmente, las señales eléctricas generadas por los transductores no son adecuadas o no son compatibles con las características de entrada de una tarjeta de adquisición de datos. En estos casos, es imprescindible el uso de dispositivos de acondicionamiento de señal que realizan un *pretratamiento* de la señal. Entre otras, las funciones más usuales de los acondicionadores son: amplificación, filtrado, aislamiento eléctrico, multiplexado e incluso *linealización*. La figura 5.1 muestra una configuración general de un sistema basado en la adquisición de datos.

Figura 5.1. Configuración general de un sistema PC basado en la adquisición de datos.

En cualquier máquina o proceso automatizado es fundamental disponer de elementos que indiquen el estado del proceso o el valor de la variable a controlar para que el sistema actúe correctamente. Estos elementos son los sensores, transductores y captadores.

El concepto de sensor y sonda puede ser distinto al de transductor o captador, pero normalmente está extendido utilizarlos de forma indistinta para referirse a la función que desempeñan en un sistema automatizado. En el caso de sensores sencillos pueden tener la función de sensor y transductor en el mismo elemento (tal es el caso de un simple final de carrera).

La función de estos elementos es adaptar las variables de entrada (magnitud física o química) en otro tipo de magnitud proporcional (normalmente en automatización es una variable eléctrica) que pueda ser interpretable por el sistema y así se pueda realizar el control del proceso. Esta variable física o química de entrada (temperatura, posición, peso, deformación, par de fuerza, velocidad…) y la magnitud de salida dan lugar a los diferentes tipos de transductores que se disponen en el mercado. Todos ellos tienen unas propiedades generales comunes y otras particularidades en función del tipo de magnitud medida.

El **sensor** o también llamado “sonda” es el elemento que se encuentra en contacto directo con la magnitud que se va a evaluar. Al interactuar con esta sufre cambios en sus propiedades. Por ejemplo, la magnitud física puede ser la temperatura y la propiedad alterada puede ser la resistencia eléctrica que varía proporcionalmente a la variable medida.

El **transductor** es un dispositivo que tiene la misión de traducir o convertir una señal física en otra distinta entendible por el sistema, es decir, convierte una señal no interpretable por el sistema en otra variable interpretable por dicho sistema.

La variación de resistencia que se obtiene en el sensor es un valor directo de la temperatura, pero en los sistemas de control no trabajan con estas señales, sino con tensión o intensidad, por lo que con ayuda de un transductor esta variación de resistencia se asocia a una variación de voltaje, que también es proporcional a la temperatura (o variable que se mida en cada caso). Una variación de resistencia en el sensor es leída por el transductor y asociada a una variación de voltaje. El transductor, por tanto, suele incluir al sensor.

El captador es básicamente un **transductor** incorporado en un lazo de control realimentado y su función es recoger o captar un tipo de información en el sistema para realimentarla.

Otra parte de un transductor es el amplificador. La variación de tensión que se registra a consecuencia de la variación de la temperatura es demasiado pequeña para ser digitalizada, y es por eso que se hace necesario amplificarla. Los valores óptimos de amplificación están relacionados con el digitalizador que se disponga, ya que se debe tener en cuenta el rango de lectura del digitalizador, pues si se cambia el voltaje por encima de este rango se perderán los datos.

El sensor y el transductor son elementos distintos, pero para muchos autores, en el trabajo diario, captador, sensor y transductor suelen referirse a lo mismo, aunque es importante saber que no es así exactamente.

FUNCIONES GENERALES DEL ACONDICIONAMIENTO DE SEÑAL

Dependiendo del tipo del transductor que se use, el uso de acondicionamiento de señal puede mejorar la calidad y las prestaciones del sistema de adquisición. Las funciones de acondicionamiento que se usan generalmente para cualquier tipo de señal son: amplificación, filtrado y aislamiento.

- **Amplificación**. Debido al bajo nivel de señal que suministran los transductores, el ruido puede jugar un papel importante en lo que a error de medida de señal se refiere. Una amplificación fuera del chasis del PC y cerca de la fuente de origen de la señal puede incrementar la resolución de la medida y reducir de una forma efectiva el efecto de ruido sobre la señal deseada.

  Es cierto que las tarjetas de adquisición de datos contienen un amplificador interno. No obstante, el uso de este amplificador es más común para adaptar los márgenes dinámicos de la señal y la tarjeta, y así aumentar la resolución para disminuir la influencia del ruido en la señal.

- **Filtrado**. El uso de filtros permite rechazar un cierto margen de frecuencias indeseables. Es muy común el uso de filtros banda-eliminada con frecuencia central de 50 Hz para eliminar el ruido de red procedente de fluorescentes, maquinaria, fuentes de alimentación, etc. También son muy comunes los filtros *antialiasing* que permiten que la señal que va a ser muestreada pueda ser reconstruida

perfectamente después de la adquisición. El ancho de banda de estos filtros debe coincidir con el ancho máximo de la señal deseada.

- Aislamiento. La incompatibilidad de masas entre las tarjetas y las señales a medir es la causa más común de los problemas de medida y puede llegar a dañar la tarjeta. El método más usado para el aislamiento consiste en la utilización de circuitos ópticos.

5.1.2. Tarjetas de adquisición de datos (TAD)

Actualmente se dispone de una gran variedad de tarjetas que permite llevar a cabo diversas aplicaciones. Sin embargo, es importante conocer cuáles son las prestaciones que ofrece cada tarjeta para que se adapte correctamente a la aplicación sin que sus prestaciones sean muy elevadas, pero tampoco muy bajas.

CONDICIONES GENERALES SOBRE LAS TAD

En este apartado se analizan las consideraciones que determinan las características hardware de las tarjetas de adquisición de datos, para tener un criterio de valoración de la efectividad de la TAD y de comparación entre diferentes placas.

- Entradas analógicas. Las prestaciones y precisión que proporciona una tarjeta, en cuento a entradas se refiere, son básicamente el número de canales de que dispone, la frecuencia de muestreo, la resolución y los niveles de entrada. Generalmente, muchos de estos parámetros se pueden configurar por software.

  El número de canales analógicos se ha de especificar tanto para entradas referenciadas a masa como para diferenciales. Las entradas referenciadas a masa también se las conoce como "*single-ended inputs*". Si entre el terminal de referencia y tierra existe una diferencia de potencial, esta se denomina "tensión en modo común", y es causante de muchos errores de medida. Esta configuración se utiliza en adquisición de señales de alto nivel, donde el error introducido por la señal en modo común es despreciable.

  Las señales diferenciales se basan en que los dos terminales de una entrada corresponden con dos terminales de entrada de la TAD, es decir, no existe ningún terminal referenciado a masa. De esta forma se elimina la tensión en modo común. Esta configuración de entrada es útil para la adquisición de señales de bajo nivel.

- **Frecuencia de muestreo**. Determina la velocidad a la que se producen las conversiones *Analog to Digital Converter* (ADC). Una frecuencia de muestreo elevada proporciona señales con mayor calidad de definición en tiempo y a la vez aumenta el flujo de datos hacia el procesador. Por tanto, se habrá de buscar un valor de compromiso que haga óptimo el funcionamiento del sistema. Es fundamental en toda adquisición respetar el teorema de Nyquist para el muestreo.

- **Resolución**. Indica el número de bits que utiliza el conversor ADC para cuantificar los niveles de señal analógica. Cuanto mayor sea el número de bits del ADC, mayor será el número de niveles de señal que se puede representar.

- **Niveles de entrada**. Son los límites de entrada de tensión de la TAD. Es muy común diferenciar entre señales unipolares y bipolares. Las señales unipolares admiten únicamente niveles de tensión positivos, mientras que las bipolares permiten las dos polaridades. La figura 5.2 muestra una señal adquirida por una entrada unipolar de 10 voltios mediante un conversor ADC con una resolución de 3 bits.

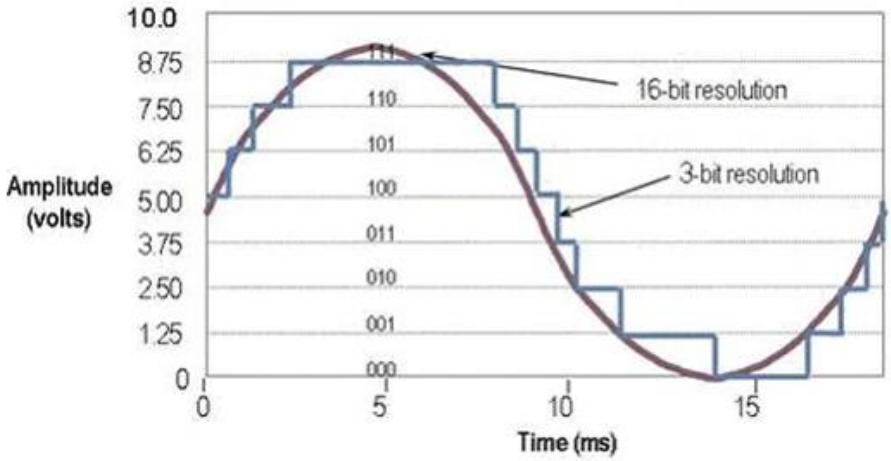


Figura 5.2. Representación de una señal analógica sinusoidal muestreada con un ADC de 3 bits de resolución.

Para disponer del máximo de resolución en la medida, el margen dinámico de señal de entrada debe coincidir con el margen de la TAD. Se puede utilizar el amplificador interno de la tarjeta con el objetivo de hacer esto posible.

- **Salidas analógicas**. Muchas TAD incorporan salidas analógicas. Básicamente, las características técnicas de las salidas analógicas son las comentadas para las entradas.

- **Puertos digitales**. Son líneas de entrada/salida digitales. Se utilizan para control de procesos, generación de modelos por testeo, comunicación con equipos periféricos, etc. Los parámetros más importantes que caracterizan los puertos digitales son el número de líneas disponibles, la velocidad a la cual se pueden transferir los datos y la capacidad de control de diferentes dispositivos (*handshacking*).

- **Temporizadores**. Son líneas útiles para muchas aplicaciones, tales como contar las veces que se produce un evento, generar bases de tiempos para procesos digitales o generación de pulsos.

DIAGRAMA DE BLOQUES GENERAL DE UNA TAD

La etapa de entrada de una TAD es muy común para todos los tipos y modelos. Fundamentalmente está compuesta por un multiplexor, que permite disponer de varios canales de entrada; seguido de un amplificador de instrumentación de ganancia programable. Este amplificador se conecta a otro amplificador de muestreo y retención (*Sample & Hold*) y finalmente este proporciona el valor de tensión al conversor ADC. La figura 5.3 muestra la etapa de entrada general de una TAD.

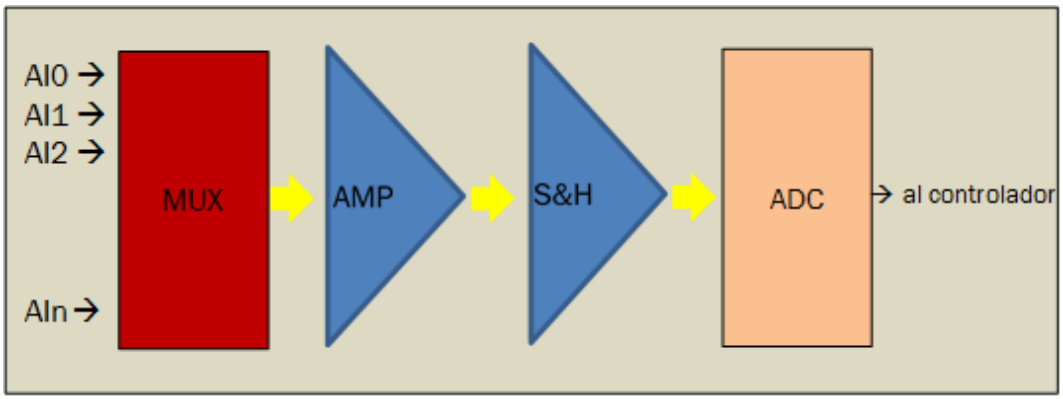


Figura 5.3. Etapa de entrada general de una TAD.

En cuanto a las salidas analógicas, estas se componen esencialmente de conversores *Digital to Analog Converter* (DAC) que se conectan directamente al bus interno del microprocesador. Para cada salida analógica se necesita un conversor DAC que normalmente tiene la misma resolución que los AD de la entrada.

5.1.3. Software de control de las TAD: NI-DAQ

Toda placa de adquisición de datos necesita de un software de control. Este control se puede llevar a cabo de tres formas distintas. La primera consiste en la programación directa de los registros. Es el método que permite más flexibilidad en cuanto a la capacidad de control, pero también es el más

costoso en cuanto a tiempo y dificultad de programación. Otro método es utilizar un driver o software de control de la tarjeta. Se trata de una serie de funciones que actúa sobre los registros de la tarjeta, pero a un nivel de programación superior al de la programación directa de registros. Proporciona la misma flexibilidad de programación que esta, pero el tiempo de desarrollo de la aplicación disminuye de una forma apreciable. Por último, cabe la posibilidad de controlar la tarjeta a través de un programa de nivel superior a los anteriores. Esto permite el desarrollo de aplicaciones potentes en cuanto a representación y análisis se refiere. Un ejemplo de este método sería el control de la tarjeta a través de LabVIEW.

Aunque se desee hacer uso de una aplicación de alto nivel, es imprescindible conocer la existencia y el funcionamiento básico del driver correspondiente si se desea aprovechar al máximo las prestaciones de la tarjeta. En general, cualquier aplicación se dedica a las funciones de este software. Es conveniente que el software de control de la tarjeta sea del mismo fabricante. National Instruments proporciona el software NI-DAQ para el control de dichas tarjetas. Se trata de un conjunto de más de 110 funciones para el control de las placas de adquisición.

NI-DAQ es una librería de rutinas que soporta todas las placas de adquisición de National Instruments. Gestiona las entradas y salidas analógicas y digitales, las señales de temporización, el control de DMA para lenguajes de programación convencionales, el control de los *triggers* y el multiplexado. También dispone de funciones de control para RTSI y SCXI de National Instruments. El número de servicios que se hacen servir para una aplicación depende de la complejidad de la aplicación que se desee gestionar y de las prestaciones de la tarjeta que se utilice. NI-DAQ está disponible para sistemas DOS, Windows, Windows NT, Macintosh y plataformas SUN.

El software NI-DAQ para Windows incorpora librerías DLL (*Dinamic Link Library*). Esto permite una gran flexibilidad para el control de la tarjeta utilizando diferentes lenguajes de programación. LabVIEW es uno de los lenguajes con posibilidades para su uso, destacando entre otros, como Borland C++, Turbo PASCAL para Windows, Visual Basic, etc.

Las posibilidades de programación que facilita NI-DAQ son, básicamente, gestión de buffers y datos, gestión de recursos y por último mensajes conducidos por eventos.

GESTIÓN DE BUFFERS Y DATOS

NI-DAQ puede gestionar transferencias de datos por DMA, interrupciones o software de estructura *polling*. NI-DAQ utiliza principalmente dos sistemas de

transferencia de datos: I/O programable y DMA. La velocidad de transferencia de datos está limitada tanto por el mecanismo de transferencia como por la computadora, la tarjeta y el sistema operativo.

El I/O programable es un método de software para la transferencia de datos desde la memoria de la computadora a un dispositivo de entrada/salida. Para cada dato, la CPU ha de ejecutar un código de transferencia de datos hacia la TAD. Por tanto, la CPU permanecerá ocupada mientras el dato esté siendo escrito o leído de la tarjeta, y no podrá atender a ningún código de aplicación; en definitiva, el funcionamiento general del sistema será más lento.

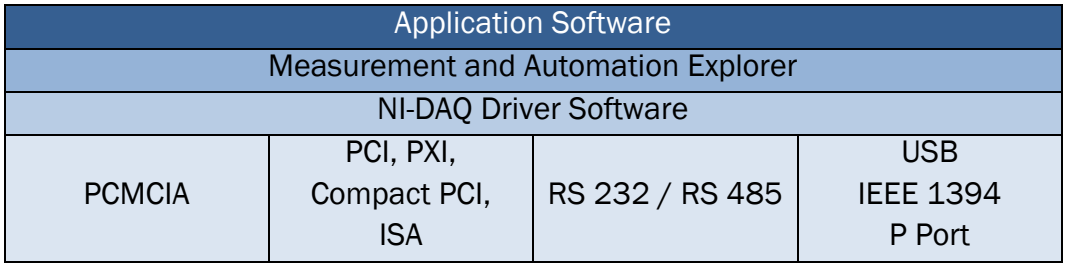


Figura 5.4. Estructura de software de un sistema de adquisición de datos con software NI-DAQ.

Por otro lado, el método de transferencia de datos por DMA hace uso del hardware en lugar del software para hacer las transferencias entre la memoria de la computadora y la tarjeta. Previamente, se programa el chip controlador de DMA. Este chip lleva a cabo las transferencias entre la memoria y los dispositivos I/O independientemente del procesador. Por tanto, este queda libre de tener que ejecutar un código para transferir cada dato, y permanece hábil para ejecutar otras aplicaciones. Sin embargo, la CPU y el controlador de DMA comparten el mismo bus, lo que conlleva una disminución de rendimiento del sistema, aunque no tanto como en el primer método.

GESTIÓN DE RECURSOS

Permite gestionar varias funciones al mismo tiempo y compartir recursos entre diferentes tarjetas de forma simultánea, incluyendo el control de múltiples canales de DMA y niveles de interrupción dentro del estándar RTSI de National Instruments, así como controlar dispositivos de acondicionamiento, como por ejemplo el SCXI.

El bus RTSI es un sistema para sincronizar el funcionamiento de diversas tarjetas de National Instruments al mismo tiempo.

MENSAJES CONDUCIDOS POR EVENTOS

En las técnicas de programación clásicas, los programas se ejecutan por secuencias y consultan el estado de los diferentes dispositivos y objetos de forma progresiva (*polling*), haciendo continuamente uso de estructuras de bucles hasta que se produzca un evento. Con el desarrollo de las interfaces gráficas para el usuario (GUI), como Windows, las aplicaciones empiezan a utilizar la programación conducida por eventos y orientada a objeto. Un evento es una operación que genera un mensaje tan pronto como se produce dicho evento y requiere una respuesta del programa de aplicación. De esta forma se evitan los tiempos que genera la programación convencional. NI-DAQ está capacitado para enviar mensajes conducidos por eventos a aplicaciones Windows.

5.1.3.1. Measurement and Automation Explorer (MAX)

Este programa (MAX) da acceso a todas las tarjetas y dispositivos DAQ, GPIB, IMAQ, IVI, Motion, VISA y VXI. Se accede a él mediante la siguiente ruta en LabVIEW: *Tools/Measurement & Automation Explorer...*

Con el MAX se puede configurar el hardware y software de National Instruments; añadir nuevos canales, interfaces e instrumentos virtuales; ejecutar test de diagnóstico del sistema; ver los dispositivos e instrumentos conectados a nuestro sistema; etc.

Hay que destacar que algunas de las opciones dependen de los dispositivos conectados en el equipo. Así, por ejemplo, las categorías *Data Neighborhood* y *Scales* sólo se mostrarán en el caso de que esté instalado un dispositivo DAQ. Del mismo modo, la categoría IVI aparecerá en aquellos dispositivos con IVI instalado.

CONFIGURACIÓN DEL MAX

La ventana de trabajo del MAX se divide en dos ventanas. A la izquierda se tiene *Configuration* (configuración, figura 5.5) y en la derecha el resultado de la selección que se ha hecho.

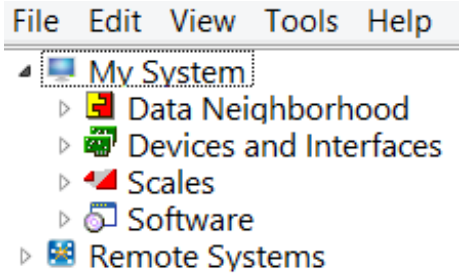


Figura 5.5. Configuración del MAX.

- *Data Neighborhood*: Crea canales virtuales en el dispositivo.
- *Devices and Interfaces*: Configura los dispositivos e interfaces instalados en el equipo.
- *Scales*: Configura y crea nuevas escalas existentes para los instrumentos virtuales.
- *Software*: Muestra, lanza y actualiza el software de National Instruments que se tenga instalado.

GENERACIÓN DE CANALES VIRTUALES DE ADQUISICIÓN Y GENERACIÓN DE DATOS

Otra vía para especificar el medio desde el que hacer la adquisición o generación de datos es la creación de una canal virtual mediante el software *Measurement & Automation Explorer* (MAX). De esta manera, la información del dispositivo y el canal vienen autocontenidos dentro de este elemento virtual, con la ventaja de que se puede cambiar tanto el dispositivo como el canal utilizados sin la necesidad de editar el código de adquisición en LabVIEW. Los pasos a seguir son los siguientes:

- Abrir la aplicación *Measurement & Automation Explorer (MAX)*.
- Seleccionar la carpeta de *Data Neighborhood*.
- Seleccionar la opción *Create New…*
- Escoger la opción de crear un NI-DAQmx *Global Virtual Channel* (NI-DAQmx librerías).
- A partir de este momento seleccionar si el canal virtual hace referencia a una entrada o salida analógica o digital; seleccionar un nombre y una descripción para el canal virtual; especificar el tipo de fuente de señal (tensión, corriente, temperatura…), las unidades, el rango y si es necesario algún tipo de escalado mediante un ajuste de offset y ganancia (=mx+b); y finalmente seleccionar la TAD y el canal.

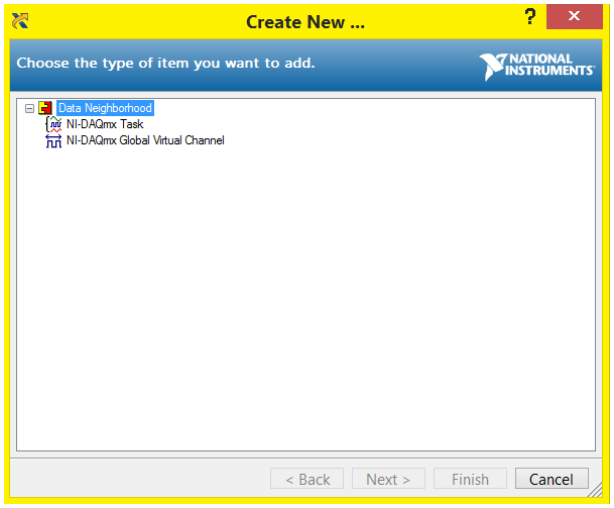


Figura 5.6. Ítems de *Data Neighborhood*.

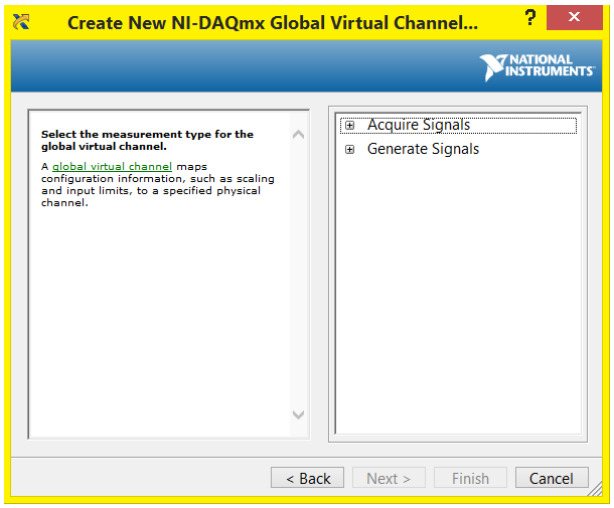


Figura 5.7. Selección de las características del canal virtual.

Si por cualquier causa se necesita cambiar el canal o sus características para realizar la adquisición, o al cambiar de sensor se necesita efectuar un escalado diferente de la señal de tensión, se puede modificar la propiedad del canal virtual desde el MAX sin necesidad de cambiar el código de la aplicación en LabVIEW. Se utilizan sobre todo canales virtuales de adquisición y generación para especificar principalmente la tarjeta a utilizar, el canal y el tipo de entrada o salida. Cuando se quiera especificar desde el MAX parámetros como la frecuencia de adquisición, opciones de sincronización o disparo, se debe utilizar una tarea de adquisición como se describe en el siguiente apartado.

CREACIÓN DE TAREAS (TASK) DE ADQUISICIÓN Y GENERACIÓN DE DATOS

El uso de las tareas de adquisición de datos se recomienda cuando se quieren definir parámetros más concretos sobre la adquisición o generación, tales como la frecuencia de muestreo y opciones de sincronismo y disparo. Para crear una tarea (NI-DAQmx *task*), se siguen los mismos pasos que para crear un canal, pero en este caso desde el MAX se selecciona una nueva NI-DAQmx *task* y se eligen determinadas opciones de configuración que ofrece el asistente.

Una vez creada la tarea y desde el MAX se tiene la opción de modificar los parámetros de la tarea y de comprobar su funcionamiento sin la necesidad de programar nada en LabVIEW, únicamente utilizando el MAX.

5.2. Configuración de los sensores y de la tarjeta

5.2.1. Conexiones

Los adaptadores Pasco ELVIS están diseñados para que los sensores Pasco funcionen en el software LabVIEW. Tanto el adaptador analógico como el adaptador digital permiten la recolección de datos y la alimentación de los sensores gracias a sus pines. Además, son compatibles con la mayoría de placas *protoboard*.

El adaptador analógico (CI-6718) sirve para conectar el cable del sensor de fuerza, mientras que el sensor digital (CI-6719), para el de movimiento. Este sensor dispone de dos conexiones para sus dos clavijas. Si se conecta la clavija amarilla al conector J1 y la negra al J2, los datos se refieren en sentido antihorario del sensor. Si se hace al revés, los datos se corresponden en el sentido horario.

Los adaptadores se insertan en la placa *protoboard* para conectar sus pines con la tarjeta de adquisición de datos y la alimentación mediante cables. Las conexiones se indican en la tabla 5.1 y figura 5.8.

DAPTADOR ANALÓGICO		Pin adaptador	Conector tarjeta	Número conector
AIN+	*Positive Sensor Analog Input Port*	(1) IN (+)	AI1	17
AIN–	*Negative Sensor Analog Input Port*	(2) IN (–)	AI9	18
+5 V	*+5 V Sensor Power Input*	(4) +5 V	+5 V	10
PGND	*Sensor Power Ground*	(5) PGND	DGND	11
+12 V	*+12 V Sensor Power Input*	(6) +12 V	Polo positivo batería A	-
–12 V	*–12 V Sensor Power Input*	(7) –12 V	Polo negativo batería B	-
N/C	*No Connection*	(3) No conectado	-	-
AOUT	*Analog Output*	(8) No conectado	-	-
ADAPTADOR DIGITAL		Pin adaptador	Conector tarjeta	Número conector
GND	*Digital Port #1 Power Ground*	GND	DGND	11
SIG1	*TTL Signal for Digital Port #1*	SIG1	P0.0 in	1
+5	*+5 V Sensor Power Input*	+5	+5	10
+5	*+5 V Sensor Power Input*	+5	+5	10
SIG2	*TTL Signal for Digital Port #2*	SIG2	P0.3 in	4
GND	*Digital Port #2 Power GND*	GND	DGND	5

Tabla 5.1. Tabla de conexiones.

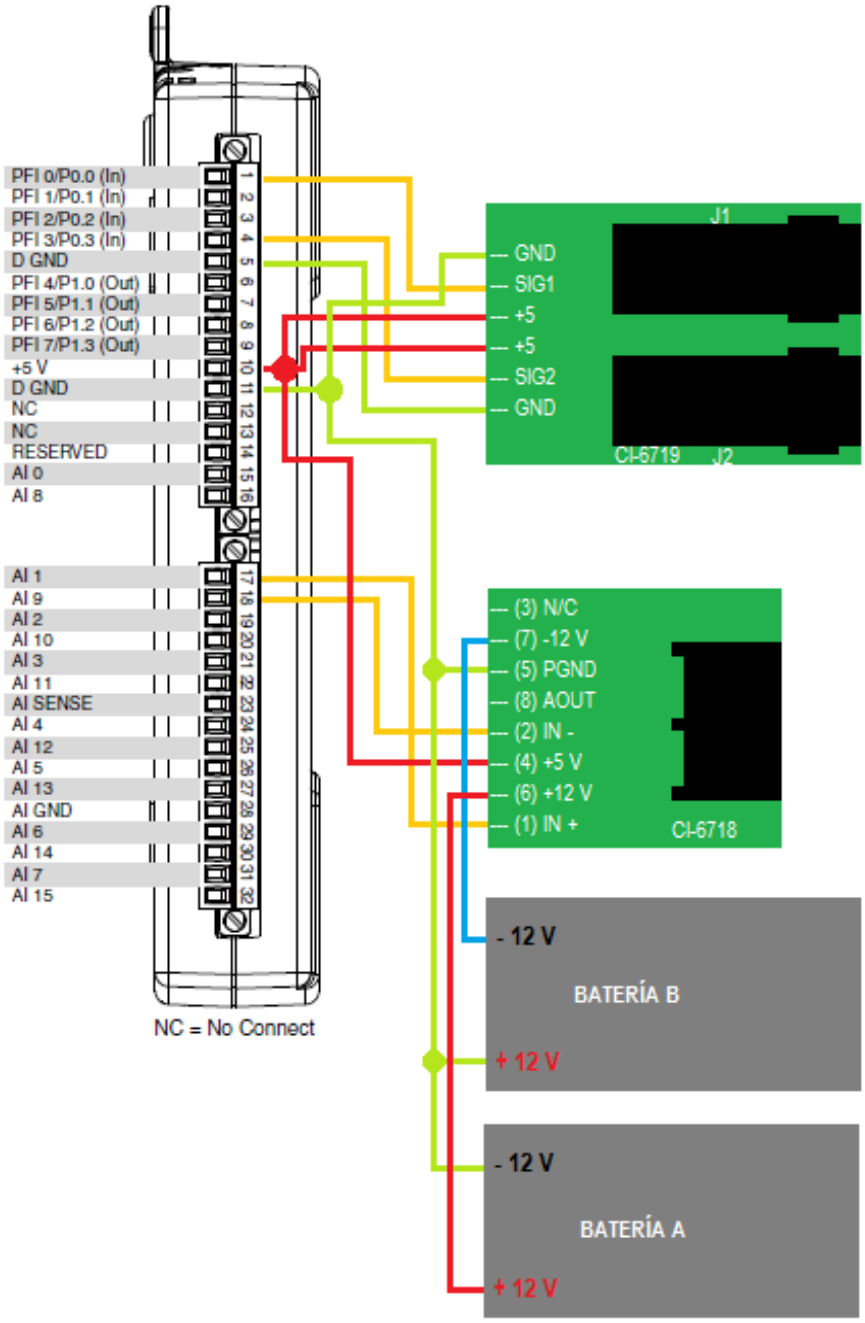


Figura 5.8. Diagrama de conexiones.

5.2.2. Alimentación

Muchas veces la alimentación está relacionada, en el caso de la tarjeta, con la familia lógica a la que pertenecen los circuitos digitales que tiene. La tarjeta de este Proyecto, la NI USB-6210, necesita 5 voltios positivos, provenientes del cable USB conectado con la computadora. Un led verde que parpadea indica que la unión está correcta. Por otra parte, existe una gran variedad de sensores. Por ello, la alimentación varía en función del tipo de medidor.

Aunque sean dispositivos distintos y formen parte del mismo sistema, demandan alimentaciones diferentes.

SENSOR DE FUERZA (CI-6718)
Se necesitan dos fuentes de alimentación iguales. El sensor requiere voltaje positivo y negativo, por un lado 12 voltios positivos y por otro, 12 negativos, ambos en corriente continua. La batería A suministra la tensión positiva, y su borne negativo se conecta a masa, mientras que la batería B da tensión negativa, y su borne positivo se enlaza con la masa. El valor de ±12 voltios se debe a que este sensor logra sus mejores prestaciones en este rango.

SENSOR DE MOVIMIENTO (CI-6719)
Los 5 voltios necesarios para su funcionamiento los proporciona la tarjeta.

5.2.3. Configuración

La configuración, calibración, adquisición y procesado de datos se puede realizar de dos modos distintos:

- A través del computador de interfaz universal de Pasco.
- Por medio de una tarjeta de adquisición de datos, de una computadora y del software LabVIEW.

El computador de Pasco ofrece, entre otras especificaciones, entradas digitales, salidas analógicas y generadores de funciones. Sin embargo, en este TFG se ha optado por usar una tarjeta de adquisición de datos. Mediante el uso de esta tarjeta, el presente Proyecto es más visual y educativo, porque la interfaz de Pasco incluye el software y lo único que hay que hacer es conectar los cables de los sensores y un cable USB a una computadora.

Figura 5.9. Computador *inteface* modelo 850 de Pasco.

5.2.3.1. Adquisición de datos en LabVIEW

LabVIEW es un programa adecuado para la adquisición de datos, entre otros motivos, por su total compatibilidad con las tarjetas de National Instruments. Su interfaz gráfica ofrece una gran potencia de visualización de señales y dispone de librerías de procesado para el tratamiento de las señales adquiridas. Para que todo esto sea posible, LabVIEW ofrece una librería de adquisición de datos que proporciona al usuario una herramienta de trabajo de fácil uso y que permite disponer de una mayor flexibilidad en cuanto al manejo de las TAD se refiere.

Desde la versión de LabVIEW 7.0 aparece la versión de NI-DAQ llamada NI-DAQmx, que resuelve algunos de los problemas de sincronización y multitarea que en ocasiones podía tener NI-DAQ tradicional (anterior a LabVIEW 7.0). NI-DAQmx se encuentra en la paleta *Functions*, en la familia *Measurement I/O*.

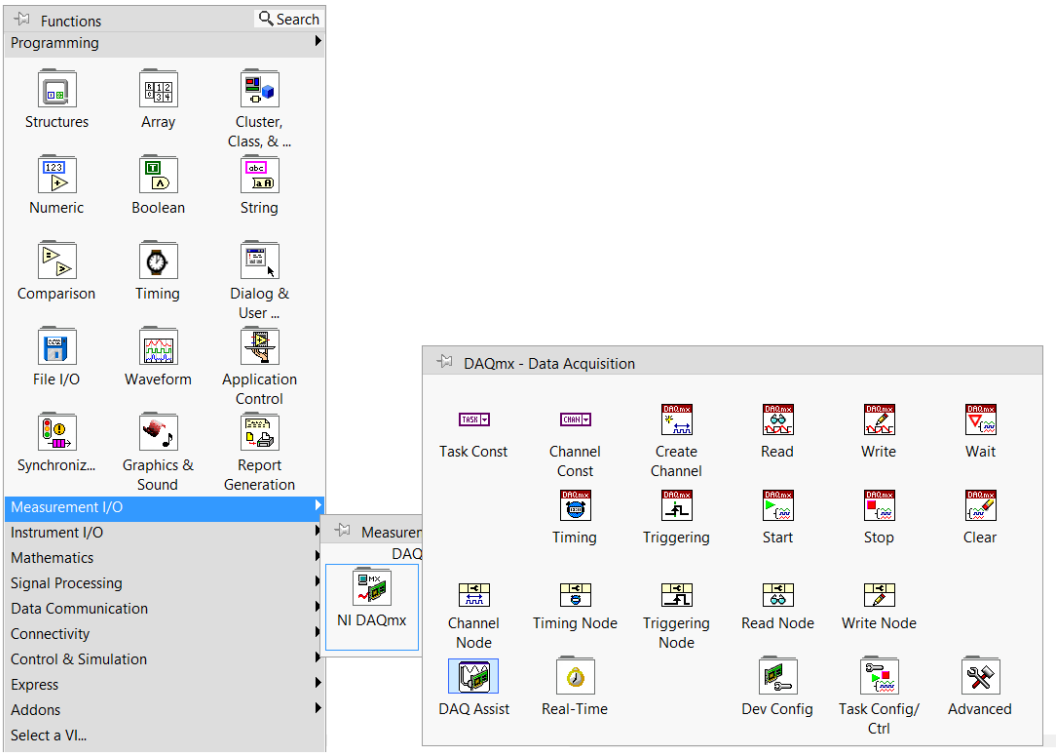


Figura 5.10. Paleta NI-DAQmx.

LIBRERÍAS DE ADQUISICIÓN DE DATOS CON NI-DAQmx

Al cliquear en NI-DAQmx, aparecen los bloques correspondientes a esta librería. Los bloques más importantes para la adquisición y generación de señales son *DAQmx Read.vi* y *DAQmx Write.vi*. Mediante el uso de VI (*Virtual Instrument*) polimórficos se selecciona la función específica a realizar, por ejemplo: adquisición de un valor analógico, de una forma de onda, de un valor

digital, del valor de un contador o de la generación de un valor analógico de salida, de una forma de onda, etc.

Para utilizar las funciones de NI-DAQmx es necesario crear un canal virtual (*channel*) o una tarea (*task*) donde se reflejen las propiedades básicas de la función, como el canal de entrada o de salida de la tarjeta, el modo de generación o muestreo (único o continuado), la velocidad de adquisición, los límites o el tipo de canal (referido a masa, diferencial).

Este Proyecto usa los bloques en vez de la aplicación *Measurement & Automation Explorer* (MAX) para crear y configurar los canales de comunicación de los sensores con la tarjeta y la computadora. Así, se muestra cómo configurar los bloques en vez de usar el MAX, para exponer el potencial del lenguaje gráfico de LabVIEW.

5.2.3.2. Comunicación con el sensor de fuerza

DAQmx Create Virtual Channel (VI)

Es un VI polimórfico. Crea un canal virtual o un conjunto de canales para una tarea. En la parte inferior se selecciona la función específica a realizar (cuadro *Polymorphic VI Selector*). La opción *AI Voltage* crea un canal o varios para medir voltaje.

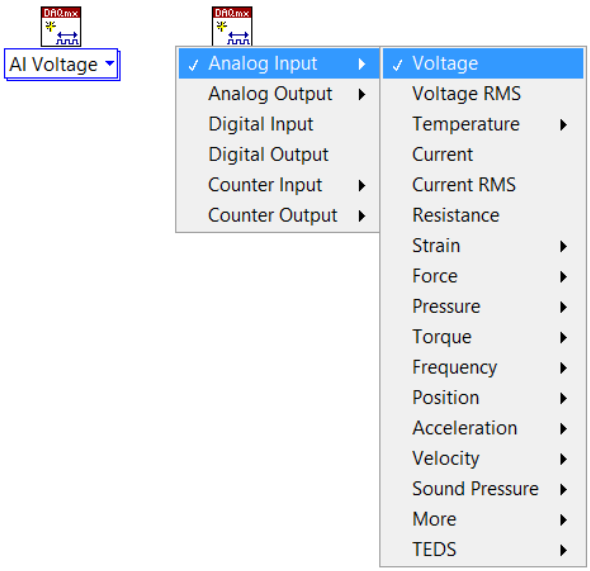


Figura 5.11. Canal *AI Voltage*.

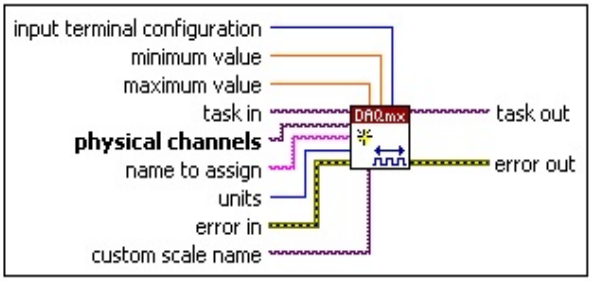


Figura 5.12. Bloque *DAQmx Create Virtual Channel (VI)*, canal *AI Voltage*.

Las entradas y salidas de este bloque que se usan en este Proyecto son las siguientes:

Physical channels: Especifica los nombres de los canales físicos para el uso de la creación del canal virtual. DAQmx enumera todos los canales físicos de los aparatos y módulos conectados. Para la tarjeta NI-USB 6210, los canales son AI1 (+,–), (Dev1/ai1).

Units: Especifica las unidades del voltaje. Se elige V (voltios) para la tarjeta.

Maximum value: Especifica en unidades el máximo valor que se espera medir. El sensor proporciona un valor máximo de 8 V.

Minimum value: Mínimo valor a medir. Sería de –8 V.

Input terminal configuration: Especifica la configuración de salida del terminal para el canal. Para la tarjeta se selecciona *default*.

Task out: Es la referencia a la tarea después de la ejecución de este bloque. La tarea contiene los canales virtuales creados. En este caso, al no conectar la entrada *task in*, NI-DAQmx crea automáticamente la referencia.

Error out: Contiene información de error.

DAQmx Start Task (VI)

Transición entre la tarea al estado de ejecución para empezar a medir. Es decir, se mide el voltaje de manera constante cada vez que se ejecute el bloque.

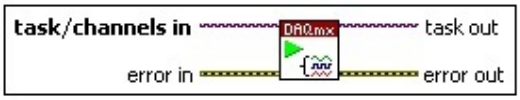


Figura 5.13. Bloque *DAQmx Start Task (VI)*.

Estas son sus entradas y salidas.

Task/channels in: Nombre de la tarea del canal virtual.

Task out: Referencia a la tarea.

Error in/out: Describe condiciones de errores.

DAQmx Read (VI)

Lee muestras de la tarea. Al igual que *DAQmx Create Virtual Channel*, es un VI polimórfico. Del mismo modo, en la parte inferior se selecciona la función específica a realizar (cuadro *Polymorphic VI Selector*). En este caso, adquisición de un valor analógico en una variable numérica de precisión doble. *Analog DBL 1Chan 1Samp* lee una única muestra de una tarea que contiene una única salida de canal analógico.

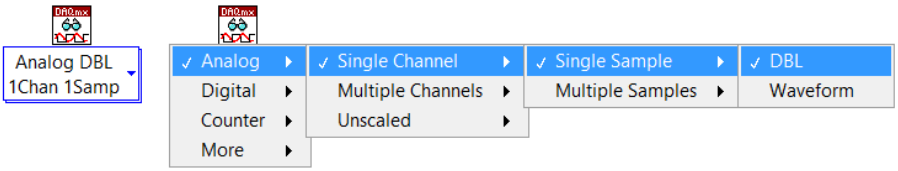


Figura 5.14. Función *Analog DBL 1Chan 1Samp*.

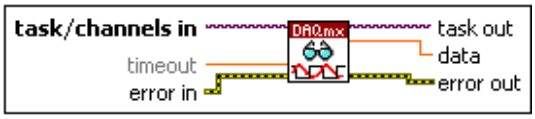


Figura 5.15. Bloque *DAQmx Read (VI)*, función *Analog DBL 1Chan 1Samp*.

Sus entradas y salidas empleadas se muestran a continuación.

Task/channels in: Nombre de la tarea del canal virtual.

Task out: Referencia a la tarea.

Data: Devuelve una muestra. Están en las unidades que se especificaron en el bloque *DAQmx Create Virtual Channel.*

Error in/out: Describe condiciones de errores.

DAQmx Clear Task (VI)
Limpia la tarea. Antes de la limpieza, este VI la aborta, si es necesario, y lanza algún recurso a la tarea reservada. No se puede usar esta tarea después de que se limpie, a menos de que se reproduzca.

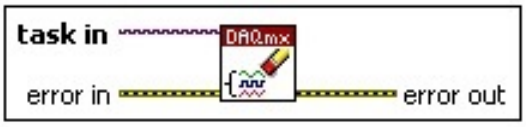


Figura 5.16. Bloque *DAQmx Clear Task (VI)*.

Se programa un código sencillo con estos bloques para tomar medidas del sensor. *Simple Error Handler.vi* indica si ocurre un error.

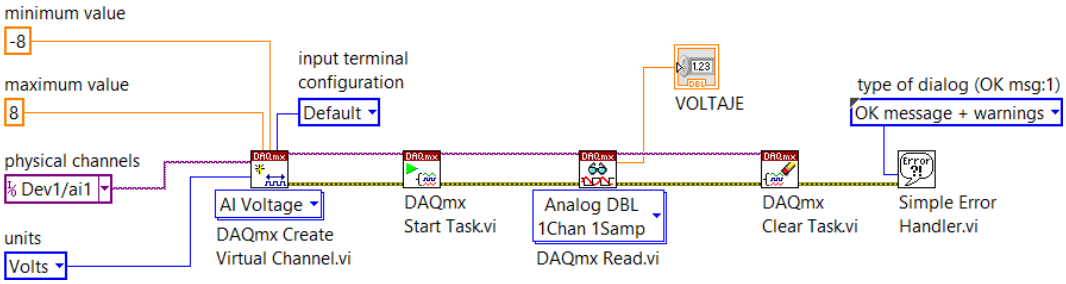


Figura 5.17. Código en LabVIEW para el sensor de fuerza.

5.2.3.3. Configuración y calibración del sensor de fuerza

El sensor tiene una salida de entre –8 y +8 voltios y un rango de entre –50 y +50 N. Es decir, genera –8 voltios para –50 N, 0 voltios para una fuerza "cero" y +8 voltios para 50 N. Un empuje o una compresión es considerado positivo y una fuerza de tracción es considerada negativa. El sensor tiene galgas extensiométricas montadas en un eje en forma de "S". El eje incorpora una protección para que no se dañe el sensor si se alcanzan fuerzas mayores de 50 N. Además, lleva incorporado un botón de tara para poner a cero el sensor.

El medidor está diseñado para producir cero voltios aproximadamente cuando hay ausencia de fuerza. Un cambio en la fuerza de un newton causa un cambio en la salida del voltaje de 160 milivoltios (0,160 voltios). Después de pulsar el botón de tara, un newton causa el mismo voltaje. Por ende, un voltaje de 1,60 voltios equivale a 10 N (compresión). De igual modo, un voltaje de –1,60 voltios resulta una fuerza de –10 N (tracción).

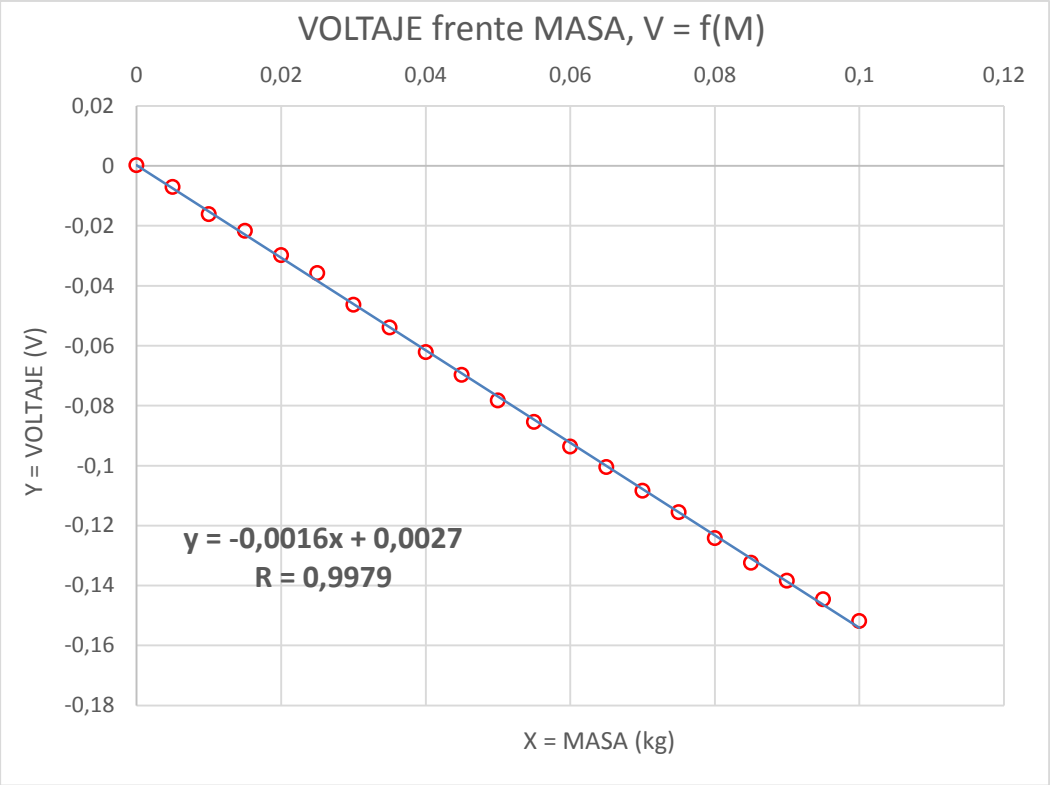
Para comprobar el correcto funcionamiento del sensor, hay que demostrar que su salida (voltaje) responde a una función matemática respecto a la señal de entrada, la fuerza. La calibración se realiza con varios valores de fuerzas.

Un polímetro mide la señal que ofrece el sensor, para observar que las lecturas que ofrece LabVIEW sean las mismas. Se toman 21 valores que corresponden a varias masas. Para cada masa, se anota el valor que ofrece LabVIEW y el polímetro, cuando la plataforma gira. Para cada valor, se pulsa el botón de tara. Después, se calcula la fuerza-peso correspondiente a cada masa. La tabla contiene los datos obtenidos con sus respectivos errores.

Con los datos obtenidos, se elabora una gráfica, voltaje (LabVIEW) frente masa.

MASA (kg) $\pm 10^{-5}$	FUERZA (N) $\pm 10^{-5}$	LabVIEW, V $\pm 10^{-5}$
0	0	$2{,}0 \cdot 10^{-4}$
$5{,}00 \cdot 10^{-3}$	$49{,}00 \cdot 10^{-3}$	$-7{,}00 \cdot 10^{-3}$
$10{,}00 \cdot 10^{-3}$	$98{,}00 \cdot 10^{-3}$	$-1{,}607 \cdot 10^{-2}$
$15{,}00 \cdot 10^{-3}$	$147{,}0 \cdot 10^{-3}$	$-2{,}166 \cdot 10^{-2}$
$20{,}00 \cdot 10^{-3}$	$196{,}0 \cdot 10^{-3}$	$-2{,}980 \cdot 10^{-2}$
$25{,}00 \cdot 10^{-3}$	$245{,}0 \cdot 10^{-3}$	$-3{,}580 \cdot 10^{-2}$
$30{,}00 \cdot 10^{-3}$	$294{,}0 \cdot 10^{-3}$	$-4{,}632 \cdot 10^{-2}$
$35{,}00 \cdot 10^{-3}$	$343{,}0 \cdot 10^{-3}$	$-5{,}389 \cdot 10^{-2}$
$40{,}00 \cdot 10^{-3}$	$392{,}0 \cdot 10^{-3}$	$-6{,}211 \cdot 10^{-2}$
$45{,}00 \cdot 10^{-3}$	$441{,}0 \cdot 10^{-3}$	$-6{,}967 \cdot 10^{-2}$
$50{,}00 \cdot 10^{-3}$	$490{,}0 \cdot 10^{-3}$	$-7{,}822 \cdot 10^{-2}$
$55{,}00 \cdot 10^{-3}$	$539{,}0 \cdot 10^{-3}$	$-8{,}546 \cdot 10^{-2}$
$60{,}00 \cdot 10^{-3}$	$588{,}0 \cdot 10^{-3}$	$-9{,}368 \cdot 10^{-2}$
$65{,}00 \cdot 10^{-3}$	$637{,}0 \cdot 10^{-3}$	$-1{,}005 \cdot 10^{-2}$
$70{,}00 \cdot 10^{-3}$	$686{,}0 \cdot 10^{-3}$	$-1{,}084 \cdot 10^{-2}$
$75{,}00 \cdot 10^{-3}$	$735{,}0 \cdot 10^{-3}$	$-1{,}155 \cdot 10^{-2}$
$80{,}00 \cdot 10^{-3}$	$784{,}0 \cdot 10^{-3}$	$-1{,}242 \cdot 10^{-2}$
$85{,}00 \cdot 10^{-3}$	$833{,}0 \cdot 10^{-3}$	$-1{,}324 \cdot 10^{-2}$
$90{,}00 \cdot 10^{-3}$	$882{,}0 \cdot 10^{-3}$	$-1{,}384 \cdot 10^{-2}$
$95{,}00 \cdot 10^{-3}$	$931{,}0 \cdot 10^{-3}$	$-1{,}446 \cdot 10^{-2}$
$100{,}00 \cdot 10^{-3}$	$980{,}0 \cdot 10^{-3}$	$-1{,}518 \cdot 10^{-2}$

Tabla 5.2. Valores de masa, fuerza y del sensor de fuerza.



Gráfica 5.1. Voltaje frente masa.

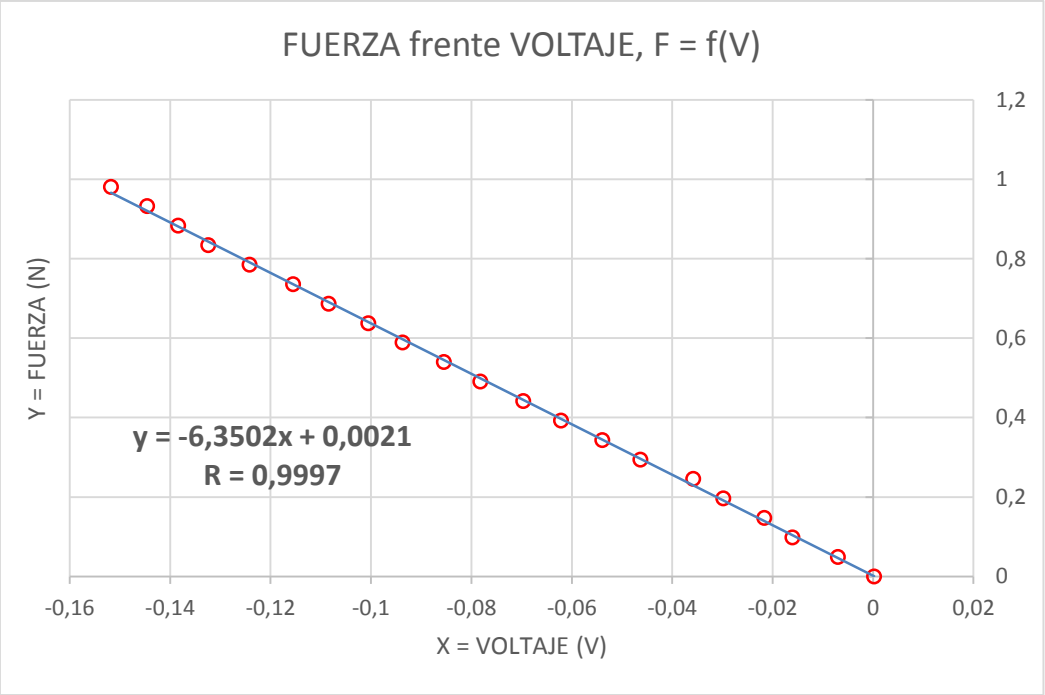
En esta gráfica se observa que los datos que ofrece el sensor muestran una relación matemática. El método de los mínimos cuadrados ofrece la ecuación:

$$y = -0,0016x + 0,0027 \qquad [38]$$

Además, se realiza otra gráfica. **Fuerza** frente **voltaje**:

DATOS VOLTAJE GRÁFICA (V) $\pm 10^{-5}$	FUERZA TEÓRICA (N) $\pm 10^{-5}$
$2{,}0 \cdot 10^{-4}$	0
$-7{,}00 \cdot 10^{-3}$	$49{,}05 \cdot 10^{-3}$
$-1{,}607 \cdot 10^{-2}$	$98{,}10 \cdot 10^{-3}$
$-2{,}166 \cdot 10^{-2}$	$147{,}15 \cdot 10^{-3}$
$-2{,}980 \cdot 10^{-2}$	$19{,}62 \cdot 10^{-3}$
$-3{,}580 \cdot 10^{-2}$	$245{,}25 \cdot 10^{-3}$
$-4{,}632 \cdot 10^{-2}$	$29{,}43 \cdot 10^{-3}$
$-5{,}389 \cdot 10^{-2}$	$343{,}35 \cdot 10^{-3}$
$-6{,}211 \cdot 10^{-2}$	$39{,}24 \cdot 10^{-3}$
$-6{,}967 \cdot 10^{-2}$	$441{,}45 \cdot 10^{-3}$
$-7{,}822 \cdot 10^{-2}$	$49{,}05 \cdot 10^{-3}$
$-8{,}546 \cdot 10^{-2}$	$539{,}55 \cdot 10^{-3}$
$-9{,}368 \cdot 10^{-2}$	$58{,}86 \cdot 10^{-3}$
$-1{,}005 \cdot 10^{-2}$	$637{,}65 \cdot 10^{-3}$
$-1{,}084 \cdot 10^{-2}$	$68{,}67 \cdot 10^{-3}$
$-1{,}155 \cdot 10^{-2}$	$735{,}75 \cdot 10^{-3}$
$-1{,}242 \cdot 10^{-2}$	$78{,}48 \cdot 10^{-3}$
$-1{,}324 \cdot 10^{-2}$	$833{,}85 \cdot 10^{-3}$
$-1{,}384 \cdot 10^{-2}$	$88{,}29 \cdot 10^{-3}$
$-1{,}446 \cdot 10^{-2}$	$931{,}95 \cdot 10^{-3}$
$-1{,}518 \cdot 10^{-2}$	$981{,}00 \cdot 10^{-3}$

Tabla 5.3. Valores de voltaje y fuerza teórica.



Gráfica 5.2. Fuerza frente voltaje.

Esta gráfica ofrece la siguiente ecuación:

$$y = -6,3502x + 0,0021 \qquad [39]$$

Si un cambio en la fuerza de un newton causa una alteración en la salida del voltaje de 160 milivoltios (0,160 voltios), un cambio de 1 voltio supone una fuerza de 6,25 N. La gráfica 5.2 corrobora esta proporción.

5.2.3.4. Comunicación con el sensor de movimiento

Los VI son los mismos que los utilizados en el caso del sensor de fuerza.

DAQmx Create Virtual Channel (VI)

La función seleccionada es *CI Cnt Edges*.

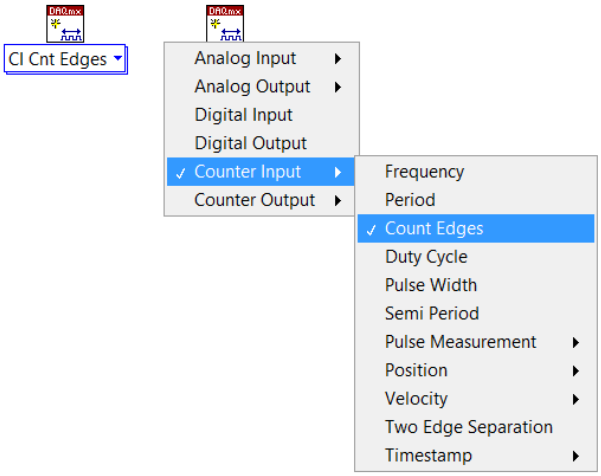


Figura 5.18. Función *CI Cnt Edges*.

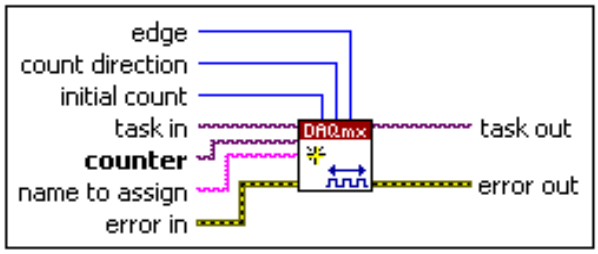


Figura 5.19. Bloque *DAQmx Read (VI)*, función *CI Cnt Edges*.

Entradas y salidas para esta aplicación:

Counter: Especifica el nombre del contador que se usa para crear el canal virtual. DAQmx enumera todos los canales físicos de los aparatos, incluidos contadores, y módulos conectados. Para la tarjeta NI-USB 6210, los canales son Dev1/ctr0.

Initial count: Es el valor desde el cual se empieza a contar. Se cuenta a partir de 0.

Count direction: Especifica si se incrementa o decrementa el contador en cada vuelta. Se elige *Count Up*.

Edge: Especifica si el pulso de la señal de salida se incrementa o decrementa según las vueltas. Se escoge *Rising*. De este modo, los grados se cuentan de manera ascendente.

Task out: Referencia a la tarea después de la ejecución de este VI.

Error out: Contiene información de error.

DAQmx *Start Task* (VI)

Mide los pulsos cuando funciona el bloque.

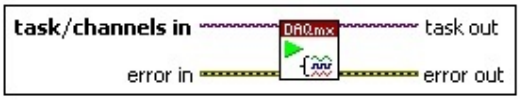


Figura 5.20. Bloque *DAQmx Start Task (VI)*.

Task/channels in: Nombre de la tarea del canal virtual.

Task out: Referencia la tarea.

Error in/out: Describe condiciones de errores.

DAQmx *Read (VI)*

Lee muestras de la tarea. Al igual que *DAQmx Create Virtual Channel*, es un VI polimórfico. Del mismo modo, en la parte inferior se selecciona la función específica a realizar (cuadro *Polymorphic VI Selector*). En este caso, adquisición de una muestra de 32 bits *unsigned* de un contador. *Counter U32 1Chan 1Samp* lee una única muestra de una tarea contador que contiene una única muestra en formato U32.

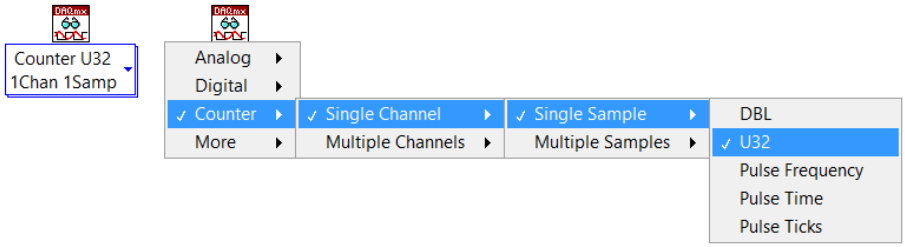


Figura 5.21. Función *Counter U32 1Chan 1Samp*.

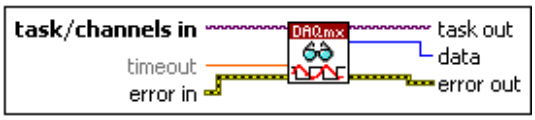


Figura 5.22. Bloque *DAQmx Read (VI)*, función *Counter U32 1Chan 1Samp*.

Las entradas y salidas conectadas son:

Task/channels in: Nombre de la tarea del canal virtual.

Task out: Referencia a la tarea.

Data: Devuelve una muestra.

Error in/out: Describe condiciones de errores.

DAQmx *Clear Task (VI)*
Limpia la tarea.

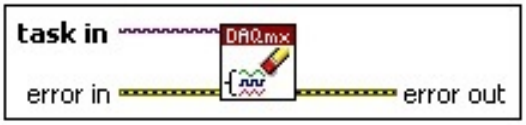


Figura 5.23. Bloque *DAQmx Clear Task (VI)*.

Como en el caso del sensor de fuerza, se diseña un pequeño programa en LabVIEW.

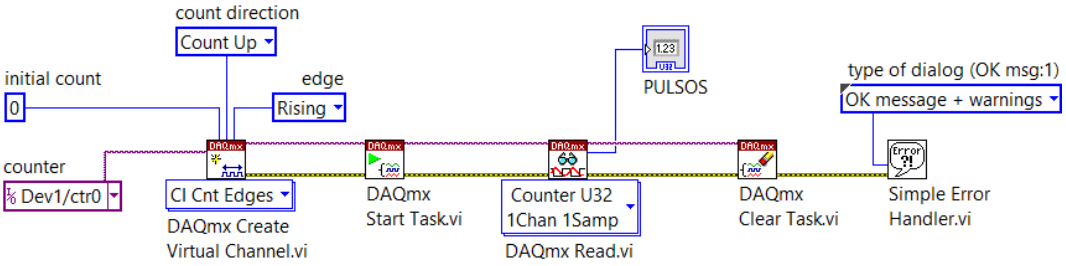


Figura 5.24. Código en LabVIEW para el sensor de movimiento.

5.2.3.5. Configuración y calibración del sensor de movimiento

Este sensor está diseñado para medir bidireccionalmente la posición en un movimiento circular. Contiene un *encoder* óptico que ofrece un máximo de 1440 conteos por revolución (360 grados) del eje del sensor. Esta resolución puede ajustarse en el software *ScienceWorkshop* de 360 a 1440 veces por revolución (1 grado o 1/4 grado). La dirección de la rotación también puede medirse. Sus dos clavijas pueden conectarse a dos canales digitales de la interfaz de la computadora *ScienceWorkshop*, de Pasco.

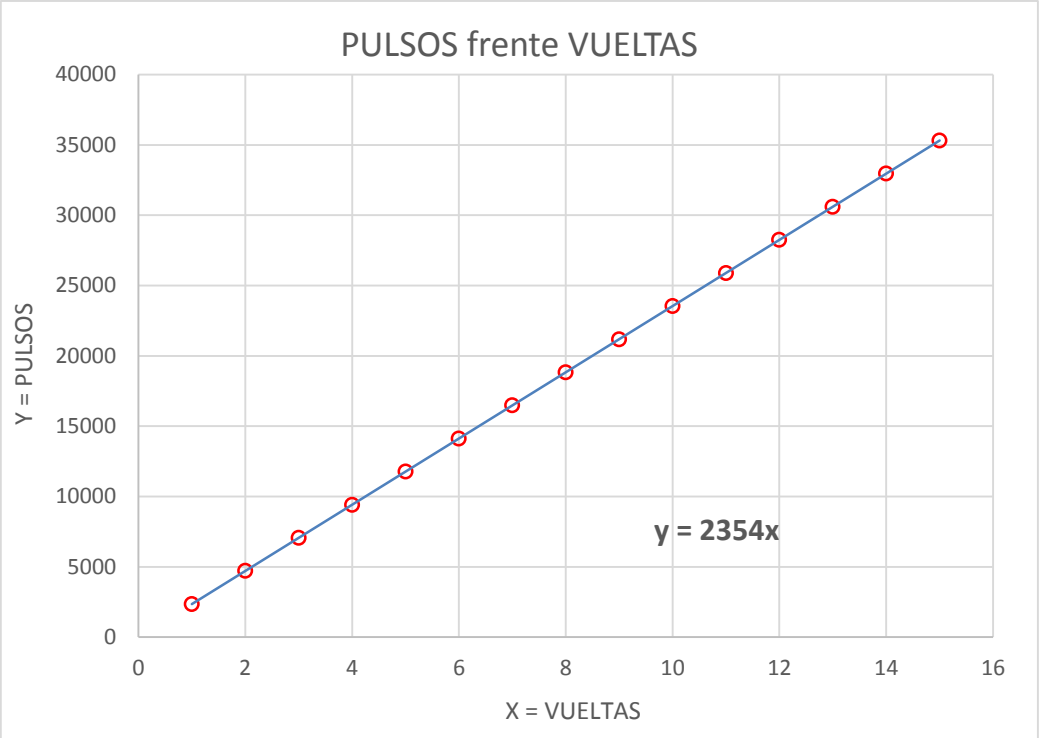
La abrazadera de la varilla puede ser montada en las tres caras del sensor, permitiendo que el medidor se monte en un soporte con diferente orientación. La polea de tres pasos en el eje giratorio permite montarse en cualquier

extremo del eje. Una correa de goma en forma de anillo "O" sirve para transmitir el movimiento.

Al igual que en el sensor de fuerza, es necesario comprobar el funcionamiento del sensor de movimiento. Este medidor tiene una resolución de 1 grado, de manera que cada pulso que produzca se corresponde a 1 grado. Se han tomado 15 valores, que son el número de vueltas contadas (de 1 a 15). Para marcar el inicio de las vueltas, en el extremo de la plataforma se coloca una señal (polea con abrazadera, del equipo ME-8952). Esta señal se alinea con un elemento externo para marcar el inicio de una vuelta. La plataforma se mueve manualmente. LabVIEW ofrece los valores cuando gira la plataforma un número determinado de vueltas. Así, se manifiesta que el número de pulsos es una función lineal respecto al giro.

VUELTAS	PULSOS
1	2354
2	4708
3	7062
4	9416
5	11770
6	14124
7	16478
8	18832
9	21186
10	23540
11	25894
12	28248
13	30602
14	32956
15	35310

Tabla 5.4. Valores de vueltas y pulsos.



Gráfica 5.3. Pulsos frente vueltas.

$$y = 2354x \qquad [40]$$

Estos valores indican que el sensor trabaja correctamente. Para traducir los valores, se dividen por la relación de diámetros entre las poleas del eje del equipo y el sensor. Los manuales de Pasco contienen las especificaciones. Así, la polea de tres pasos que dispone del anillo "O" mide 29 milímetros de diámetro. El resultado es en grados.

$$grados = \frac{número\,de\,pulsos}{\frac{189,6278}{29}} = \frac{29 \cdot número\,de\,pulsos}{189,6278} \qquad [41]$$

Si a este número se le divide por 360 grados, se obtiene el número de vueltas.

$$número\,de\,vueltas = \frac{\frac{29 \cdot número\,de\,pulsos}{189,6278}}{360} = \frac{29 \cdot número\,de\,pulsos}{360 \cdot 189,6278} \qquad [42]$$

5.2.3.6. Confrontación de los dos sensores

La ecuación correspondiente a la fuerza centrípeta es la expresión [37] deducida:

$$F_n = M\, a_n = M\, \frac{v^2}{R} = M\, R\, \omega^2$$

Relaciona la masa con la velocidad y el radio. Los sensores de fuerza y movimiento proporcionan sendos datos de fuerza (newtons) y de movimiento (número de vueltas, grados o radianes). Ambos medidores funcionan bien por separado, pero es necesario confirmar que funcionan a la perfección conjuntamente. Para ello, se ha realizado un pequeño programa en LabVIEW y se ha utilizado la fuente de alimentación para suministrar voltaje al motor. En todas las medidas se ha empleado el mismo voltaje, por lo que al aumentar la masa, la velocidad disminuye muy ligeramente. El radio es 0,20 metros. Se han tomado medidas de ambos sensores y se elabora la tabla 5.5. M representa la masa de cada medida.

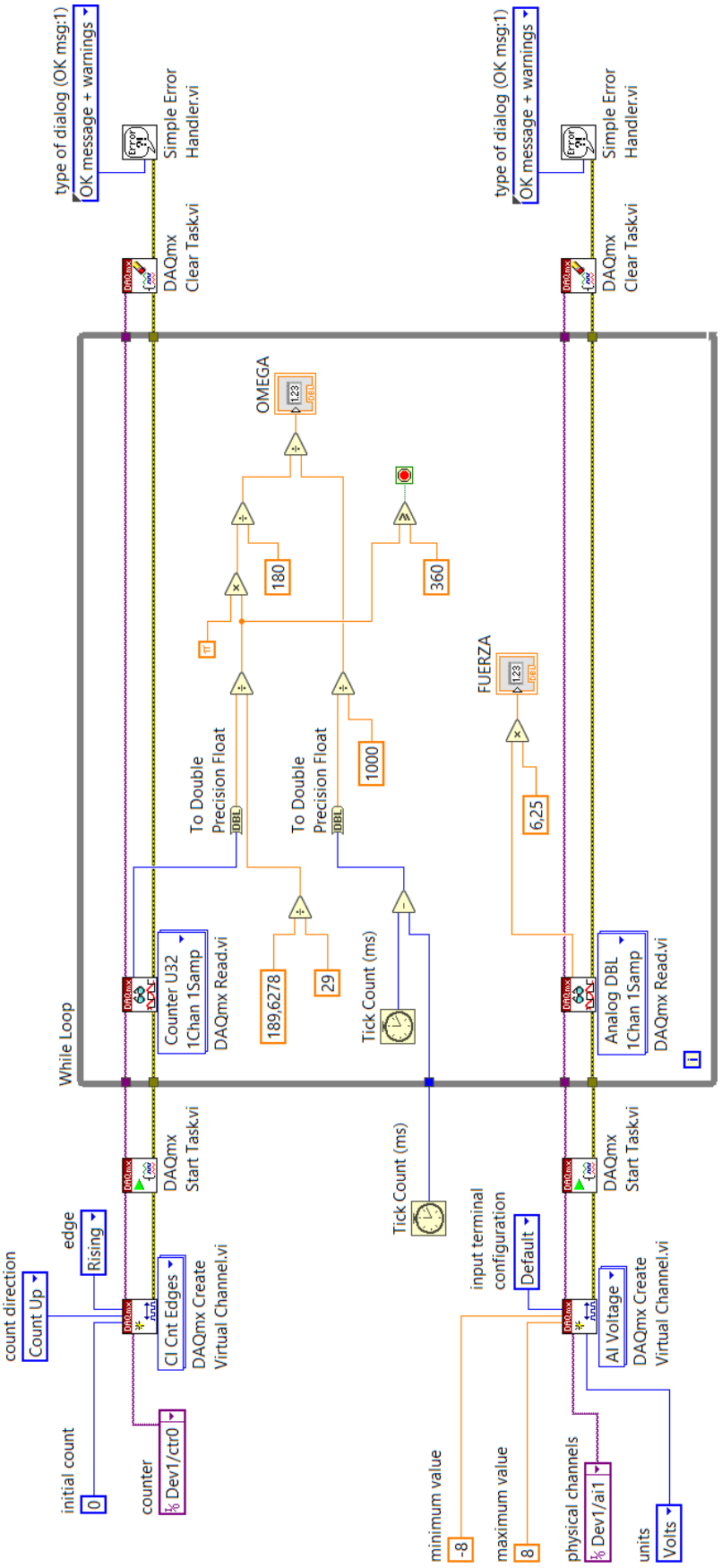
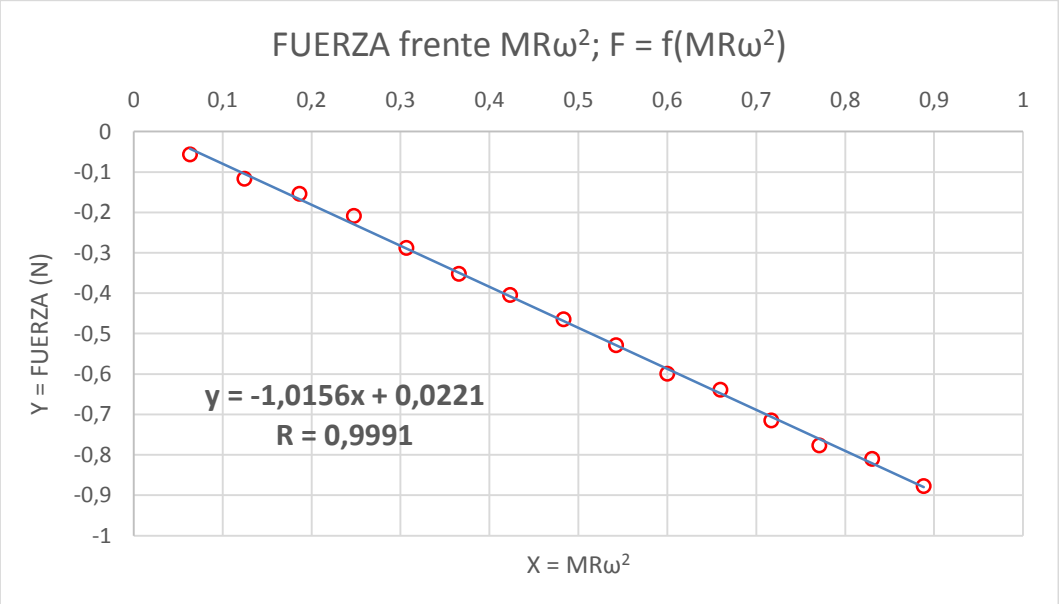


Figura 5.25. Código para los dos sensores.

M (kg) $\pm 10^{-5}$	ω (rad/s) $\pm 10^{-6}$	MRω² (N) $\pm 10^{-8}$	FUERZA (N) $\pm 10^{-8}$
$20{,}00 \cdot 10^{-3}$	3,981946	$6{,}342358 \cdot 10^{-2}$	$-5{,}656667 \cdot 10^{-2}$
$40{,}00 \cdot 10^{-3}$	3,948928	$1{,}2475226 \cdot 10^{-1}$	$-1{,}1686667 \cdot 10^{-1}$
$60{,}00 \cdot 10^{-3}$	3,942458	$1{,}8651570 \cdot 10^{-1}$	$-1{,}5386667 \cdot 10^{-1}$
$80{,}00 \cdot 10^{-3}$	3,934644	$2{,}4770277 \cdot 10^{-1}$	$-2{,}0866667 \cdot 10^{-1}$
$100{,}00 \cdot 10^{-3}$	3,916260	$3{,}0674185 \cdot 10^{-1}$	$-2{,}8790000 \cdot 10^{-1}$
$120{,}00 \cdot 10^{-3}$	3,904088	$3{,}6580567 \cdot 10^{-1}$	$-3{,}5253333 \cdot 10^{-1}$
$140{,}00 \cdot 10^{-3}$	3,889106	$4{,}2350407 \cdot 10^{-1}$	$-4{,}0460000 \cdot 10^{-1}$
$160{,}00 \cdot 10^{-3}$	3,889106	$4{,}8340001 \cdot 10^{-1}$	$-4{,}6486667 \cdot 10^{-1}$
$180{,}00 \cdot 10^{-3}$	3,882192	$5{,}4257093 \cdot 10^{-1}$	$-5{,}2860000 \cdot 10^{-1}$
$200{,}00 \cdot 10^{-3}$	3,873030	$6{,}0001446 \cdot 10^{-1}$	$-5{,}9986667 \cdot 10^{-1}$
$220{,}00 \cdot 10^{-3}$	3,872528	$6{,}5984482 \cdot 10^{-1}$	$-6{,}3873333 \cdot 10^{-1}$
$240{,}00 \cdot 10^{-3}$	3,864660	$7{,}1690865 \cdot 10^{-1}$	$-7{,}1490000 \cdot 10^{-1}$
$260{,}00 \cdot 10^{-3}$	3,851028	$7{,}7118167 \cdot 10^{-1}$	$-7{,}7676667 \cdot 10^{-1}$
$280{,}00 \cdot 10^{-3}$	3,850746	$8{,}3038171 \cdot 10^{-1}$	$-8{,}1016667 \cdot 10^{-1}$
$300{,}00 \cdot 10^{-3}$	3,848190	$8{,}8851398 \cdot 10^{-1}$	$-8{,}7730000 \cdot 10^{-1}$

Tabla 5.5. Valores de masa, MRω² y de los sensores de movimiento y de fuerza.

Después, se realiza la gráfica de la fuerza frente a MRω².



Gráfica 5.4. Fuerza frente MRω².

La ecuación que resulta es:

$$y = -1{,}0156x + 0{,}0221 \qquad [43]$$

Como la pendiente es prácticamente la unidad, se ratifica que los dos sensores funcionan perfectamente y la ecuación [37]:

$$F_n = M\, a_n = M\, \frac{v^2}{R} = M\, R\, \omega^2$$

CAPÍTULO 6.
DESARROLLO DE LA APLICACIÓN

6.1. LabVIEW

Las aplicaciones creadas en LabVIEW reciben el nombre de instrumentos virtuales o VI (*Virtual Instrument*). Cada VI está compuesto de tres componentes:

- Un **panel frontal** (*Front Panel*). Es la interfaz del usuario.
- Un **diagrama de bloques** (*Block Diagram*). Contiene el código fuente gráfico que define la funcionalidad del VI.
- **Icono y conector**. Identifica a cada VI, de modo que se puede utilizar dentro de otro VI. Un VI dentro de otro VI recibe el nombre de subVI.

PANEL FRONTAL Y DIAGRAMA DE BLOQUES

El panel frontal se compone de controles e indicadores, que son los terminales de entrada y salida, respectivamente. Los controles simulan elementos de ingreso al instrumento y proporcionan datos al diagrama de bloques. Los indicadores imitan elementos de salida del instrumento y visualizan los datos que el diagrama de bloques adquiere o genera.

El diagrama de bloques contiene el código fuente gráfico, compuesto por unas representaciones ilustrativas de funciones que controlan los objetos del panel frontal. Estos objetos aparecen como terminales en el diagrama de bloques.

MENÚS DE LabVIEW

La barra de menús de la parte superior de la ventana de un VI contiene diversos menús **Pull-down** (desplegables). Al hacer clic sobre un elemento de esta barra, aparece un menú debajo de ella. Dicho menú contiene elementos comunes a otras aplicaciones Windows, como **Open** (abrir), **Save** (guardar) y **Paste** (pegar), y muchas otras particularidades de LabVIEW. La siguiente figura muestra la barra de menús tanto para la ventana **Panel** como **Diagram**.

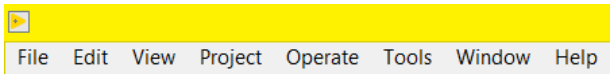


Figura 6.1. Barra de menús.

- *File* (archivo). En este menú se encuentran funciones para realizar operaciones básicas con los archivos, como abrirlos, cerrarlos guardarlos e imprimirlos. Destaca la función **VI Properties** (propiedad

del VI), que permite ver las características del VI, así como poner una contraseña.
- *Edit* (edición). Contiene funciones que permiten realizar búsquedas, así como modificar archivos de LabVIEW y sus componentes.
- *View* (ver). Aquí están disponibles las paletas que tienen los elementos del panel frontal y las funciones para el diagrama de bloques, además de la paleta de herramientas. También permite acceder a la función **Error List** (lista de errores). Además, muestra la jerarquía del VI, a quién llama, cuáles son sus subVI, los VI no abiertos o *breakpoints* (puntos de ruptura).
- *Project* (proyecto).
- *Operate* (función). Contiene elementos para controlar el funcionamiento de los VI. Una opción interesante en este menú es la de modificar los valores por defecto.
- *Tools* (herramientas). En este menú se encuentra todo aquello para pulir los VI, como por ejemplo herramientas de comparación entre VI, editor de librerías o generador de ejecutables. Otra de las opciones interesantes de este menú es **Options...**, que permite configurar un gran número de parámetros de LabVIEW.
- *Window* (ventana). Contiene funciones que permiten configurar la apariencia de las ventanas.
- *Help* (ayuda). Presenta ayuda sobre los diferentes iconos y otros aspectos de LabVIEW.

BARRA DE HERRAMIENTAS (TOOLBAR)

Esta barra sirve para editar o ejecutar los VI. Dependiendo de si se está en el modo de edición o en modo de ejecución se tienen más o menos opciones.

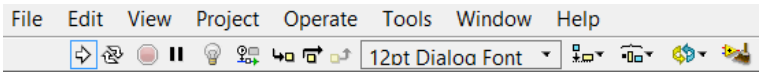


Figura 6.2. Barra de herramientas del diagrama de bloques.

La creación o modificación de un VI se realiza cuando se permanece en el modo **Edit**. Cuando se está listo para probar el VI, se selecciona **Change to Run Mode** (cambio al modo de ejecución) desde el menú **Operate**. Esto compila el VI y se pone en el modo **Run**. En esta situación se disponen de las opciones de depuración, ejecución del VI, diferentes modos de ejecución, impresión de datos, etc. Si se quiere ejecutar el VI desde el modo **Edit** sin pasar al modo **Run**, se cliquea sobre la flecha de ejecución. Si fuese necesario, LabVIEW compilaría primero el VI, después conmutaría el modo **Run**, ejecutaría el VI y volvería al modo **Edit** una vez que el VI se hubiese ejecutado.

La función de los diferentes botones es la siguiente (de izquierda a derecha):

- Botón **Run**. Ejecuta los VI. Si la flecha está rota, indica que se está creando un VI. Si sigue estando en esa situación después de dar por terminado el diagrama, se considera que el VI tiene errores y no se puede ejecutar.
- **Botón Run Continously**. Ejecuta indefinidamente el VI hasta que se aborta o se hace una pausa de ejecución.
- **Abort Execution**. Aborta la ejecución del VI.
- Botón **Pause**. Hace una pausa en la ejecución del VI.
- Botón **Ayuda**. Muestra en la parte superior izquierda una pantalla referida a la ventana **Context Help**.
- Botón **Enter**. Aparece para recordar que hay un nuevo valor disponible para reemplazar otro antiguo (por ejemplo, cuando cambiamos el valor de un control).
- **Higligth Execution**. Presenta una ejecución animada del diagrama de bloques.
- **Step into**. Ejecuta paso a paso el código y después hace una pausa. Si el siguiente elemento a ejecutar es un sub VI, lo abriría y continuaría la ejecución en el diagrama de bloques de este subVI.
- **Step over**. Es similar a *Step Into*. Cuando se llega a un subVI y otras funciones los ejecuta sin necesidad de abrirlos y de entrar dentro de su código.
- **Step Out**. Finaliza la ejecución del nodo actual y después hace una pausa; por ejemplo, si se tiene un bucle *For* de 500 repeticiones, las haría automáticamente y después pondría una pausa en el siguiente nodo.
- **Text Settings** (configuración de texto). Cambia la fuente del texto que se quiere escribir.
- **Align Objects** (alineación de objetos). Alinea los objetos que se seleccionen según unos ejes.
- **Distribute Objects** (distribución de objetos). Distribuye los objetos espaciándolos.
- **Reorder** (reordenación de objetos). Los reordena en relación uno a otro. Es útil cuando existen objetos superpuestos y hay que acceder al que está en el fondo.
- **Resize Objects** (redimensionar objetos). Redimensiona objetos del panel frontal al mismo tamaño.
- **Warning**. Aparecerá en el caso de que un VI presente un aviso o advertencia y se haya marcado la opción *Show Warnings* (mostrar avisos) en la ventana *Error list* (lista de errores).

CREACIÓN DE OBJETOS

Para elaborar el panel frontal o editar el diagrama de bloques, los controles, indicadores y bloques se encuentran en dos paletas: **Controls** (*Front Panel*) y **Functions** (*Block Diagram*). Ambas paletas son accesibles mediante dos procedimientos. El primero, hacer clic con el botón derecho del ratón sobre una parte del panel frontal o diagrama de bloques que carezca de objetos. El segundo, acceder al menú **View** y seleccionar **Control Palette** (panel) o **Functions Palette** (diagrama).

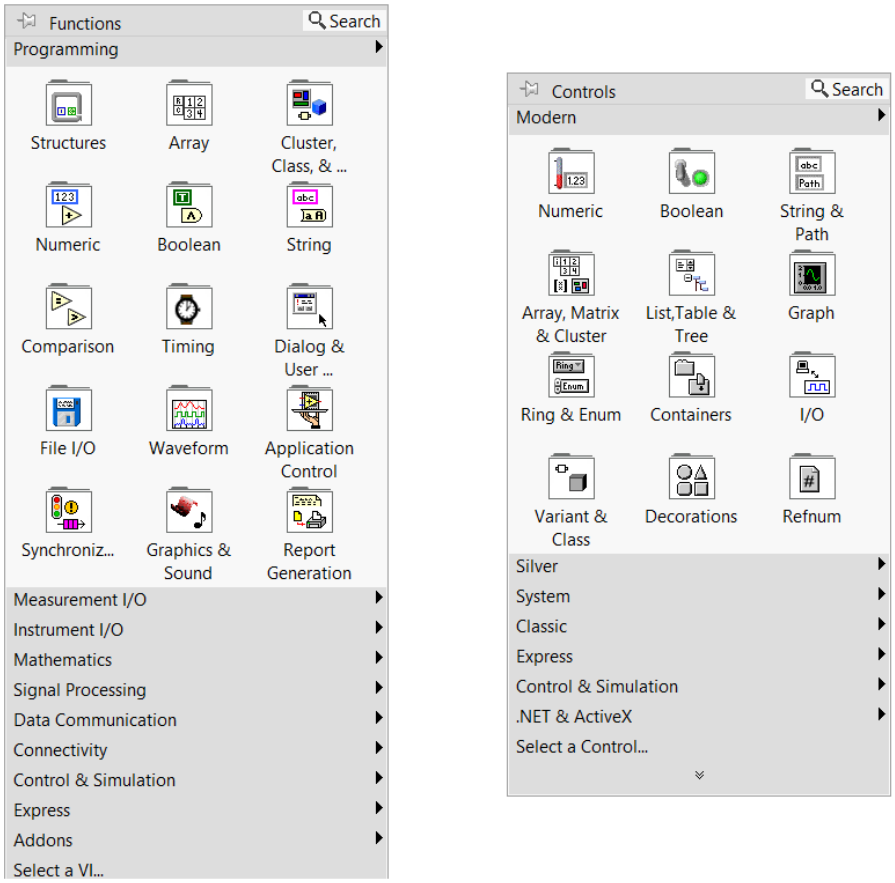


Figura 6.3. Paletas *Functions* y *Controls*.

Todos los objetos en LabVIEW tiene asociados menús *Pop-up*, los cuales se obtienen pulsando el botón derecho del ratón sobre dicho objeto. Mediante la selección de sus diferentes opciones se pueden configurar determinados parámetros, como el aspecto o comportamiento de ese objeto.

HERRAMIENTAS DE LabVIEW (*Tools*)

Una herramienta es un modo de funcionamiento especial del ratón. Se usa para llevar a cabo funciones específicas de edición o ejecución. Para acceder a ellas hay que seleccionar la opción **Tools Palette** del menú **View**. Una forma alternativa es pulsar <Shift> más el botón derecho del ratón. La descripción

de los botones viene a continuación (de izquierda a derecha y de la parte superior a la inferior).

Figura 6.4. Paleta de herramientas.

- **Automatic Tool Selection** (selección automática de herramienta). Permite habilitar la tecla <Tab> para cambiar entre las cuatro herramientas más comunes de la paleta *Tools*.
- **Operate Value** (valor operativo). Maneja los controles del panel frontal (y los indicadores del modo **Edit**). Es la única herramienta disponible en el modo **Run**.
- **Position/Size/Select** (situación/tamaño/selección). Selecciona, mueve y redimensiona objetos.
- **Edit Text** (edición de texto). Crea y edita textos.
- **Connect Wire** (conexión de cables). Enlaza objetos del diagrama de bloques y asigna a los terminales del conector del VI los controles e indicadores del panel frontal.
- **Object Shortcut Menu** (menú *Pop-up* del objeto). Despliega el menú *Pop-up* asociado al objeto. Tiene el mismo efecto que si se pulsa el botón derecho del ratón sobre el objeto.
- **Scroll Window** (desplazamiento de la pantalla). Desplaza la pantalla en la dirección que deseemos para ver posibles zonas ocultas.
- **Set/Clear Breakponit** (establecer/quitar puntos de ruptura). Permite poner tantos puntos de ruptura como se desee a lo largo del diagrama de bloques. Cuando durante la ejecución se llega a uno de ellos, LabVIEW conmuta automáticamente al diagrama de bloques. Esta herramienta también sirve para quitar los puntos.
- **Probe Data** (sonda de datos). Permite capturar resultados intermedios en la ejecución de un VI que da resultados inesperados o dudosos.
- **Get Color** (capturar color). Permite saber de manera específica qué color tiene un objeto, texto u otros elementos.
- **Set Color** (colorear). Colorea diversos objetos y los fondos.

6.1.1. Tipos de datos en LabVIEW

LabVIEW ofrece una gran cantidad de tipos de datos con los que se puede trabajar, respondiendo así a las necesidades reales de aplicaciones. Uno de los aspectos más significativos de LabVIEW es la diferenciación que efectúa en el diagrama de bloques entre los diferentes tipos de controles o indicadores, basada en que cada uno de ellos tiene un color propio. De esta manera, y como consecuencia de una memorización o asimilación práctica, es muy fácil identificarlos y reconocer inmediatamente si se está trabajando con el tipo de datos adecuado. Se distinguen los siguientes tipos, los cuales pueden funcionar como controles o como indicadores (entre paréntesis queda reflejado el color con el que quedan representados en el diagrama de bloques):

- *Boolean* (verde claro). Los tipos de datos *booleanos* son enteros de 16 bits. El bit más significativo contiene el valor *booleano*. Si el bit 15 se pone a 1, entonces el valor del control o indicador es **TRUE** (verdadero); por el contrario, si este bit 15 vale 0, el valor de la variable *booleana* será **FALSE** (falso).

- **Numéricos**. Hay diferentes tipos.
  - *Extended* (naranja). Utiliza números de coma flotante de precisión extendida según el modelo de computadora.
  - *Double* (naranja). Los números en coma flotante de doble precisión cumplen con el formato de doble precisión IEEE de 64 bits. Es el valor por defecto en LabVIEW.
  - *Single* (naranja). Los números en coma flotante de precisión simple cumplen con el formato de precisión simple IEEE de 32 bits.

  - *Log Integer* (azul). Los números enteros largos tienen un formato de 32 bits, con o sin signo.
  - *Word Integer* (azul). Estos números tienen un formato de 16 bits, con o sin signo.
  - *Byte Integer* (azul). Tienen un formato de 8 bits, con o sin signo.
  - *Unsigned Long* (azul). Entero largo sin signo.
  - *Unsigned Word* (azul). Palabra sin signo.
  - *Unsigned Byte* (azul). *Byte* sin signo.

  - *Complex Extended* (naranja). Número complejo con precisión extendida.
  - *Complex Double* (naranja). Complejo con precisión doble.
  - *Complex Single* (naranja). Complejo con precisión simple.

- *Arrays* (depende del tipo de datos que contenga). Es un conjunto de datos del mismo tipo. LabVIEW almacena el tamaño de cada dimensión de un *array* como **Long Integer** seguido por el dato. Los *arrays booleanos* se almacenan de manera diferente a los *booleanos* escalares. Estos *arrays* se almacenan como bits empaquetados. El tamaño de la dimensión viene dado en bits en lugar de *bytes*. El bit 0 se guarda en la posición más alta de memoria (215), y el bit 15 en la posición más baja (20).

- *Strings* (rosa). LabVIEW almacena los *strings* como si fueran un *array* uni-dimensional de *bytes* enteros (caracteres de 8 bits).

- *Handles*: Un *handle* es un puntero que apunta a un bloque de memoria localizable. Un *handle* sólo apunta a datos definidos por el usuario. LabVIEW no reconoce qué es lo que hay en ese bloque de memoria. Es especialmente útil para pasar de un bloque de datos por referencia entre nodos de interfaz de código (*Code Interface Nodes* o CIN).

- *Paths* (verde oscuro). LabVIEW almacena las componentes tipo y el número de un *path* en palabras enteras, seguidas inmediatamente por las componentes del *path*. El tipo de *path* es 0 para un *path* absoluto y 1 para un *path* relativo. Cualquier otro valor indicaría que el *path* no es válido. Cada componente del *path* es una cadena Pascal (*P-string*), en la cual el primer *byte* es la longitud de la *P-string* (sin incluir el *byte* de longitud).

- *Clusters* (marrón o rosa). Un *cluster* almacena diferentes tipos de datos de acuerdo a las siguientes normas: los datos escalares se almacenan directamente en el *cluster*; los *arrays*, *strings*, *handles* y *paths* se almacenan indirectamente. El *cluster* almacena un *handle* que apunta al área de memoria en la que LabVIEW ha almacenado realmente los datos.

6.1.2. Estructuras

LabVIEW permite ejecutar un mismo conjunto de sentencias un número determinado de veces, o bien que se repitan mientras se cumplen ciertas condiciones. También puede ocurrir que se quiera ejecutar una orden u otra según unas condiciones fijadas, así como que funcionen siempre antes que otras.

El programa dispone de cuatro estructuras básicas disponibles en la paleta *Structures*, en la ventana *Block Diagram*.

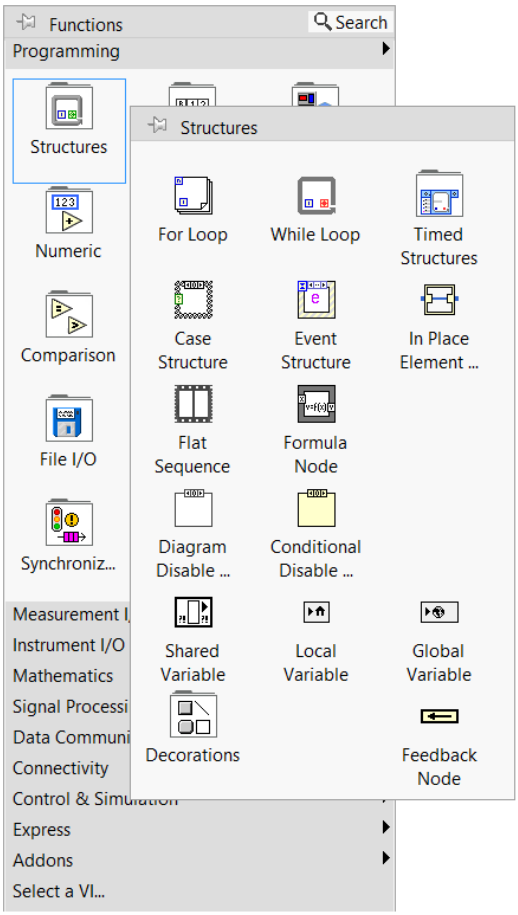


Figura 6.5. Paleta de estructuras.

Bucle *While*

Esta estructura sirve para repetir una operación mientras una determinada condición sea cierta o falsa.

Consta de dos terminales:

- Terminal condicional.
  Se le conecta la condición para ejecutar el subdiagrama. LabVIEW comprueba el estado de este terminal al final de cada iteración. Este terminal se puede configurar de manera que pare si la condición es cierta (*Stop if True*) o bien que pare si la condición es falsa (*Continue if True*).

- Terminal de iteración.
  Indica el número de veces que se repite el bucle y que, como mínimo, siempre es una (i = 0).

Figura 6.6. Bucle *While*.

Bucle *For*
Sirve para que una operación se repita un número determinado de veces.

Tiene asociados dos terminales.

- Terminal contador.
  Contiene el número de veces que se ejecuta el subdiagrama creado en el interior de la estructura. El valor del contador se fija externamente.

- Terminal de iteración.
  Indica el número de veces que actúa el programa: cero durante la primera iteración, uno durante la segunda y así hasta N–1.

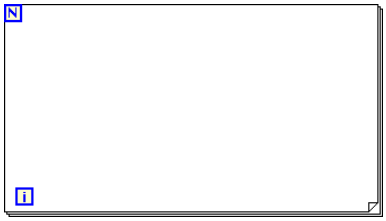


Figura 6.7. Bucle *For*.

Estructura *Case*
Se utiliza en las situaciones en las que el número de alternativas disponibles sean dos o más. Según qué valor tome el selector dentro de los *n* valores posibles, se ejecuta en correspondencia uno de los *n* subdiagramas.

Este tipo de estructura consta de un terminal llamado selector y un conjunto de subdiagramas, cada uno de los cuales está dentro de un *case* o suceso y etiquetado por un identificador del mismo tipo que el selector. En cualquier caso siempre se tiene que cubrir todo el rango de posibles valores, y al menos tiene que haber un *case* por defecto, el cual se ejecutará en caso de que el selector no corresponda a ninguno de los previstos.

Además, *Case* no cuenta con los registros de desplazamiento (*Shift Register*) de las estructuras iterativas (*While* y *For*), pero se pueden crear túneles para extraer o introducir datos. Si un *case* o suceso proporciona un dato de salida a una determinada variable es necesario que todos los demás también lo hagan. Si no ocurre de esta manera no trabaja el programa.

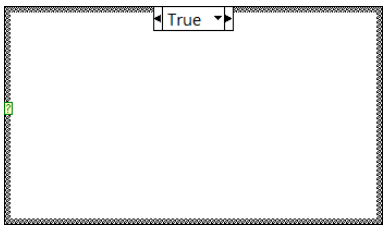


Figura 6.8. Ejemplo de estructura *Case*.

Estructura *Sequence*

En LabVIEW una función se ejecuta cuando tiene disponible todos los datos de entrada. Así, se produce una dependencia de datos que obliga a que la función que recibe un dato directa o indirectamente de otra se efectúe siempre después, creándose un flujo de programa.

Hay ocasiones en que esta dependencia de datos no existe y es necesario que un subdiagrama se ejecute antes que otro. En estos casos se usa la estructura *Sequence* para forzar un determinado flujo de datos. Cada subdiagrama está contenido en un *frame* o marco. Estos operan en orden de aparición: primero el *frame* 0 o marco 0, después el *frame* 1 y así sucesivamente hasta el último.

Al contrario de *Case*, si un *frame* aporta un dato de salida a una variable los demás marcos no tienen por qué hacerlo. Hay que tener en cuenta que el dato está solamente disponible cuando se ejecute el último *frame* y no cuando opere el *frame* que transfiere el dato.

Si la sentencia se ejecuta en orden sin paso de datos entre *frames*, no hay secuencias locales y se emplea *Flat Sequence*. La estructura *Stacked Sequence* es menos visual y era la única disponible en versiones antiguas, aunque las versiones modernas también disponen de ella.

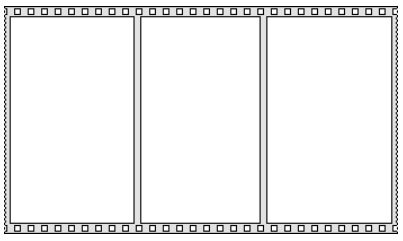


Figura 6.9. Ejemplo de estructura *Flat Sequence*.

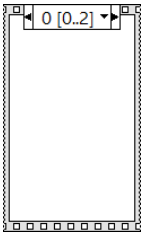


Figura 6.10. Ejemplo de estructura *Stacked Sequence*.

6.2. Diagrama de bloques

El código fuente se sintetiza mediante el siguiente esquema.

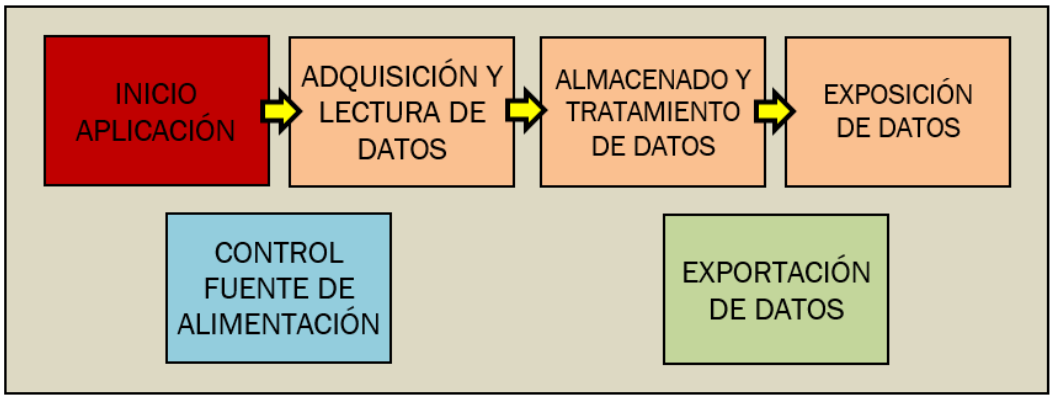


Figura 6.11. Esquema general de la aplicación.

El diagrama de bloques general se muestra en la siguiente figura. Por su gran extensión, está dividido en ocho zonas para su análisis.

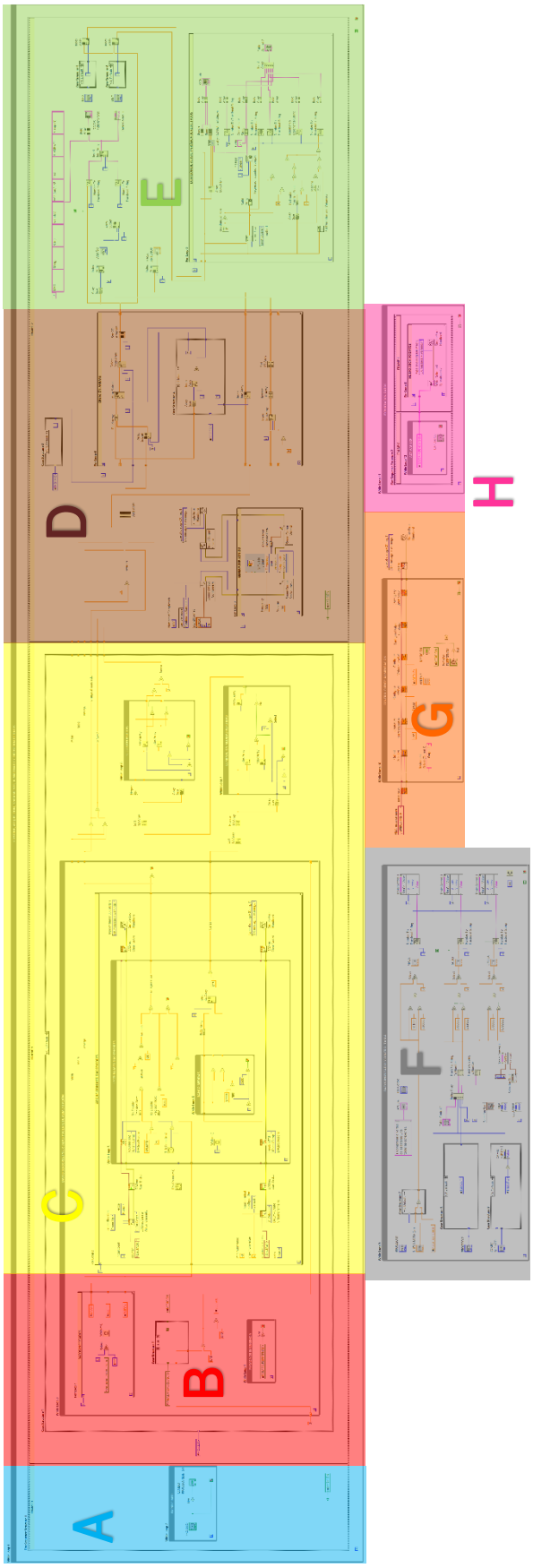


Figura 6.12. Zonas del código.

- El bucle *While* 1 abarca las zonas A, B, C, D y E.
  Contiene prácticamente todos los bloques y estructuras del programa. En él se encuentran los procesos de control, adquisición, tratamiento y exposición de los datos proporcionados por los sensores e introducidos manualmente.

  El terminal condicional está configurado en modo *Stop if True* y tiene conectado una constante FALSE (paleta *Boolean*). Esto significa que este bucle siempre se ejecuta. De este modo, se puede realizar un número indeterminado de mediciones.

  También incluye la estructura *Flat Sequence* 1, compuesta por tres *frames* o pasos. Cada *frame* tiene objetivos concretos:

  - *Frame* 0 (zona A).
    Inicio de las mediciones.

  - *Frame* 1 (zonas B y C).
    Recibe los datos trasmitidos por los sensores a través de la tarjeta de adquisición y los tecleados por el usuario o pasa al siguiente *frame* para la eliminación de valores.

  - *Frame* 2 (zonas D y E).
    Los datos se almacenan en sus correspondientes *arrays*, se tratan y se exponen por medio de una tabla y una gráfica.

- El funcionamiento de la tabla de datos está, en la zona F, así como la ayuda.

- El control de la fuente de alimentación ocupa la zona G.

- La exportación de datos en diferentes tipos de extensiones de archivo se corresponde con la zona H.

El lenguaje cuenta con un total de 29 bucles y estructuras. Están repartidas según la siguiente tabla:

BUCLE/ESTRUCTURA	NOMBRE/*FRAMES* O *CASES*	ZONA
Bucle *While* 1	CONTROL, ADQUISICIÓN, TRATAMIENTO, ALMACENADO Y EXPOSICIÓN DE DATOS	A, B, C, D y E
Estructura *Flat Sequence* 1	*FRAME* 0	A
	FRAME 1	B y C
	FRAME 2	D y E
Bucle *While* 2	EMPIECE MED	A
Estructura *Case* 1	*CASES 0* y *1*	B y C
	CASES 2 y 3	B y C
Bucle *While* 3	DATOS SENSORES, INSERTADOS Y FUENTE DE ALIMENTACIÓN	B y C
Bucle *For* 1	DATOS INSERTADOS	B
Estructura *Case* 2	TRUE	B
	FALSE	B
Bucle *While* 4	PARADA DE EMERGENCIA	B
Bucle *For* 2	ADQUISICIÓN MEDIDAS SENSORES	C
Bucle *While* 5	LECTURA MEDIDAS SENSORES	C
Bucle *While* 6	RANGO MEDIDAS	C
Bucle *While* 7	ELIMINACIÓN MEDIDAS REPETIDAS	C
Bucle *While* 8	SELECCIÓN MEDIDAS	C
Bucle *For* 3	GENERADOR SONIDO	D
Bucle *For* 4	CREACIÓN *ARRAYS*	D
Estructura *Case* 3	*CASES* 0 y 3	D
	CASES 1 y 2	D
Estructura *Case* 4	CUATRO *CASES*	D
Estructura *Case* 5	CUATRO *CASES*	E
Estructura *Case* 6	CUATRO *CASES*	E
Bucle *For* 5	ECUACIÓN DE AJUSTE Y GRÁFICA DE RESULTADOS	E
Bucle *While* 9	DATOS INSERTADOS Y CONTROL TABLA	F
Estructura *Case* 7	TRES *CASES*	F
Estructura *Case* 8	*CASE* 0	F
	CASES 1 y 2	F
	CASE 3	F
Estructura *Case* 9	*CASE* 0	F
	CASES 1 y 2	F
	CASE 3	F
Bucle *While* 10	CONTROL FUENTE ALIMENTACIÓN	G
Bucle *While* 11	EXPORTACIÓN DATOS	H
Estructura *Flat Sequence* 2	*FRAME* 0	H
	FRAME 1	H
Bucle *While* 12	ACTIVAR EXP	H
Bucle *For* 6	NUEVO ARCHIVO DATOS	H

Tabla 6.1. Bucles y estructuras del código.

6.2.1. Comienzo de las mediciones

ZONA A

El *frame* 0 contiene un bucle *While* que sirve para controlar las mediciones. En concreto, se encarga de dar comienzo a la toma de datos gracias al botón PROCESO, configurado como un pulsador con retorno. Cuando se acciona (ON) detiene la ejecución de la estructura, la condición del bucle pasa a TRUE y continúa el flujo del programa al siguiente *frame*, el 1. Además, un indicador *booleano* advierte del funcionamiento del programa. Cuando el bucle está activo, el piloto luce en verde; y cuando no lo está, en rojo. Este indicador recuerda cuándo se debe realizar el cambio de proceso y el *reset* del sensor de fuerza. Está precedido por el bloque *Not*.

Por otra parte, también se sitúa en este *frame* una variable local, que pertenece al indicador *booleano* MEDICIÓN. Para facilitar la limpieza del código, se emplean variables locales. En las variables locales los datos se almacenan en algunos de los controles o indicadores existentes en el panel frontal del VI creado. Estas variables no sirven para intercambiar datos entre VI. La principal utilidad de estas variables radica en el hecho de que una vez creada la variable local no importa que proceda de un indicador o de un control, ya que se podrá utilizar en un mismo diagrama, tanto de entrada como de salida.

Este tipo de variables se encuentra en la paleta *Structures* (ventana *Block Diagram*, figura 6.5). Una vez creada, haciendo clic con el botón derecho del ratón sobre ella, se debe elegir el control relacionado con la variable. Para ello, se pulsa en *Select Item*. Es imprescindible que el control tenga asignado un nombre de identificación. Si en algún momento se cambia el nombre del control origen, no es necesario modificar el nombre de la variable local, porque LabVIEW actualiza los cambios. Es importante destacar la opción *Change To Read* o *Change To Write*, que permite escoger entre leer o escribir en el control.

Otra forma de crear una variable local es cliquear en el control o indicador con el botón derecho y seleccionar *Create/Local Variable*. Después, se cliquea igualmente con el botón derecho en la variable local y se elige *Change To Read o Change To Write*. Las variables locales se distinguen por un símbolo de una casa negra.

En esta variable se selecciona la escritura en ella. Así, una constante FALSE ordena a este indicador que permanezca en OFF, y el indicador se ilumina en rojo. Además, controla la fuente de alimentación, obligando a que

una de las condiciones para el funcionamiento de esta fuente no se cumpla (bucle *While* 10).

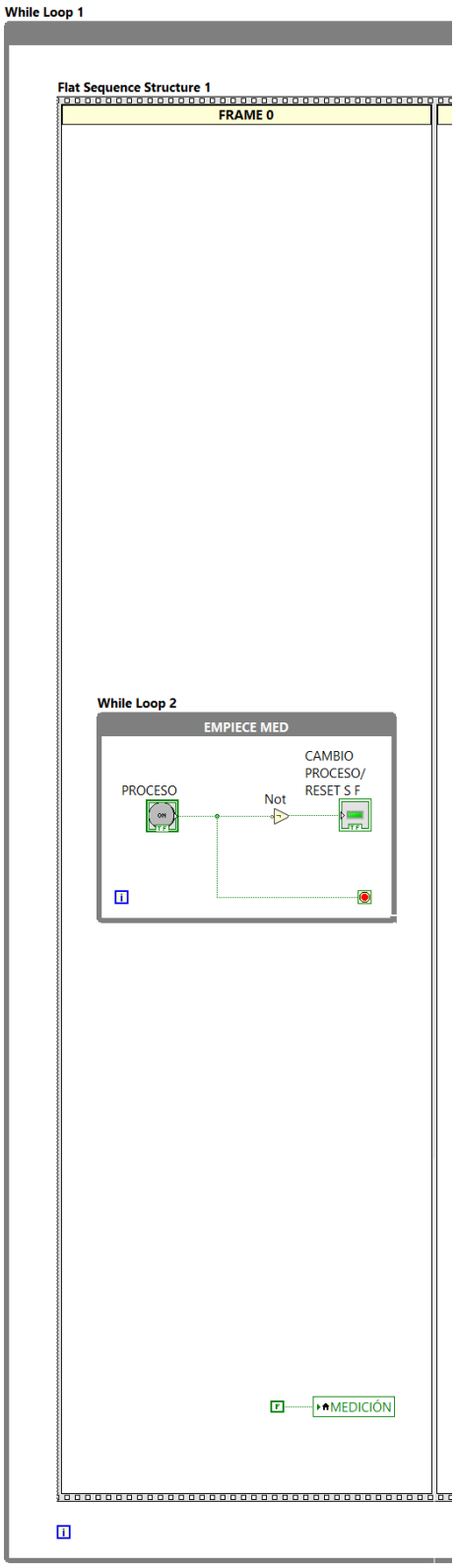


Figura 6.13. Zona A, *frame* 0.

6.2.2. Adquisición y lectura de los datos

El siguiente paso acoge una estructura *Case*, cuyo propósito radica en escoger cuál es el método de las mediciones. Consta de dos sucesos (*cases* 0 y 1, nuevas mediciones y cambio de estas, por defecto; y *cases* 2 y 3, supresión de mediciones y eliminación de todas las medidas de la tabla). Existen cuatro alternativas, pero los *cases* 0 y 1 y los *cases* 2 y 3 comparten el mismo conjunto de subdiagramas. Se extiende en las zonas B y C.

El terminal selector está conectado al elemento *Ring* (PROCESO), un menú desplegable donde se selecciona la acción que se desee. Los nombres de las opciones se editan en los ajustes del control, cliqueando con el botón derecho y seleccionando *Edit Items…*

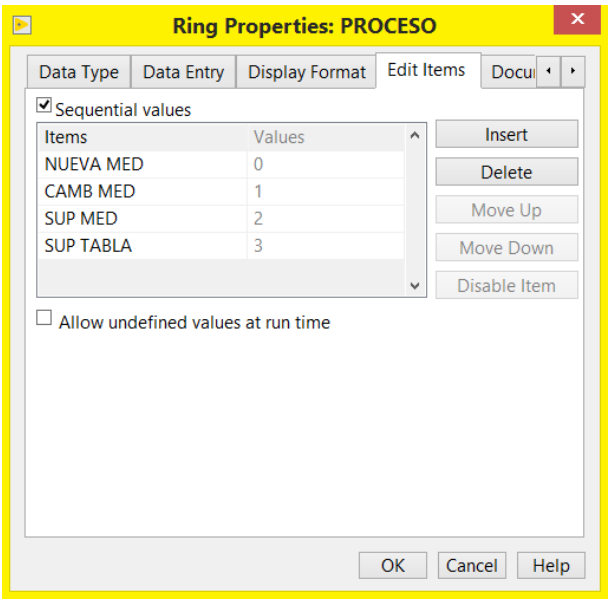


Figura 6.14. Modificación de los ítems del elemento *Ring*.

- *Cases* 0 y 1.
  - Bucle *While* 3.

    Su finalidad consiste en adquirir los datos de los sensores de movimiento y fuerza, insertar los valores introducidos por el usuario (masa, radio y omega) y proporcionar el valor del voltaje para la fuente de alimentación.

    ZONA B

    Los datos insertados manualmente (masa, radio y omega) provienen del bucle *While* 9 (zona F). Tres variables locales situadas en el bucle *For* 1 se encargan de leer estos datos. También esta estructura dispone del bloque *Wait (ms)*, paleta *Timing*. Se dispara cuando el bucle funciona y cuenta los

milisegundos establecidos. La entrada de *Wait (ms)* está conectada a otro bloque, *Select*, paleta *Comparison*, que devuelve un valor numérico de sus dos entradas dependiendo del estado de una condición (TRUE o FALSE). La entrada TRUE es 0, mientras que la entrada FALSE es 1000. La elección se realiza mediante el control PARADA EMERGENCIA, a través de una variable local.

Tras ese tiempo, los datos de la masa, radio y omega salen del bucle para continuar por el flujo del diagrama. Los 1000 milisegundos sirven para establecer un período mínimo entre mediciones, mientras que los 0 milisegundos para que el bucle termine de inmediato. El *For* 1 se ejecuta una única vez.

Cuando un cable atraviesa los límites del bucle, aparece en el borde un nuevo terminal llamado túnel, que es la conexión entre el interior y el exterior, de forma que los datos fluyen a través de él después de cada iteración. Con esto, es posible guardar de esta manera no sólo el último valor de todas las iteraciones, sino también los valores intermedios. Los bucles *For* y *While* tienen esta posibilidad.

Existen tres modos de túnel: *Last Value*, para el último valor; *Indexing*, para los valores de todas las iteraciones; y *Concatenating*, que une conjuntos de datos de todas las iteraciones de un bucle anterior. Además, a estas salidas se les puede añadir una condición (*Conditional*).

El voltaje de la fuente de alimentación se calcula en la estructura *Case* 2, que funciona gracias a la variable local PARADA EMERGENCIA. En el estado OFF, se selecciona el caso FALSE, donde se permite la suma iterativa de 0,10 gracias a un registro de deslazamiento o *Shift Register*.

Son variables locales que permiten transferir los valores finales de una iteración al principio de la siguiente. Tiene un par de terminales colocados a ambos lados de un bucle. El terminal de la derecha almacena el valor de la iteración hasta que una nueva hace que este valor se desplace al terminal de la izquierda, quedando en el de la derecha el nuevo valor. Es posible tener más de un terminal en el lado izquierdo, con el objetivo de tener más valores de iteraciones anteriores almacenados. En este caso, no es necesario.

Por ello, aumenta un valor de 0,10 en cada iteración del bucle *While* 3. Este valor se suma al número inicial 3,40. De este modo, en la iteración 0, se obtiene 3,50; en la siguiente iteración, 3,60; y así sucesivamente. El voltaje mínimo de la fuente se sitúa en 3,50 voltios, para que el motor pueda girar la plataforma dentro del rango de masas y omegas que permite la aplicación. El terminal izquierdo está conectado a una constante (0), para que después de posicionar el botón PARADA EMERGENCIA en OFF, vuelva a sumar 0,10 a 0.

Si la estructura está en caso TRUE, el valor que suma a 3,40 es cero, para que cuando se cambie al caso FALSE, el primer número del voltaje sea 3,50. Cada valor del voltaje entra en una variable local de escritura (VOLTAJE). También esta estructura controla la fuente de alimentación. En FALSE, envía TRUE a la variable local MEDICIÓN, y al revés en el caso TRUE.

El bucle *While* 4 funciona como una barrera para pasar a la siguiente iteración de la estructura DATOS SENSORES, INSERTADOS Y FUENTE DE ALIMENTACIÓN. Se activa por el control PARADA EMERGENCIA, mediante una variable local. Está unida a un bloque *Not*. Si está en ON, el bucle *While* 4 se ejecuta y no se avanza a la siguiente iteración, con la correspondiente parada del motor. Por el contrario, si permanece en OFF, termina el funcionamiento de este bucle y el programa sigue su curso.

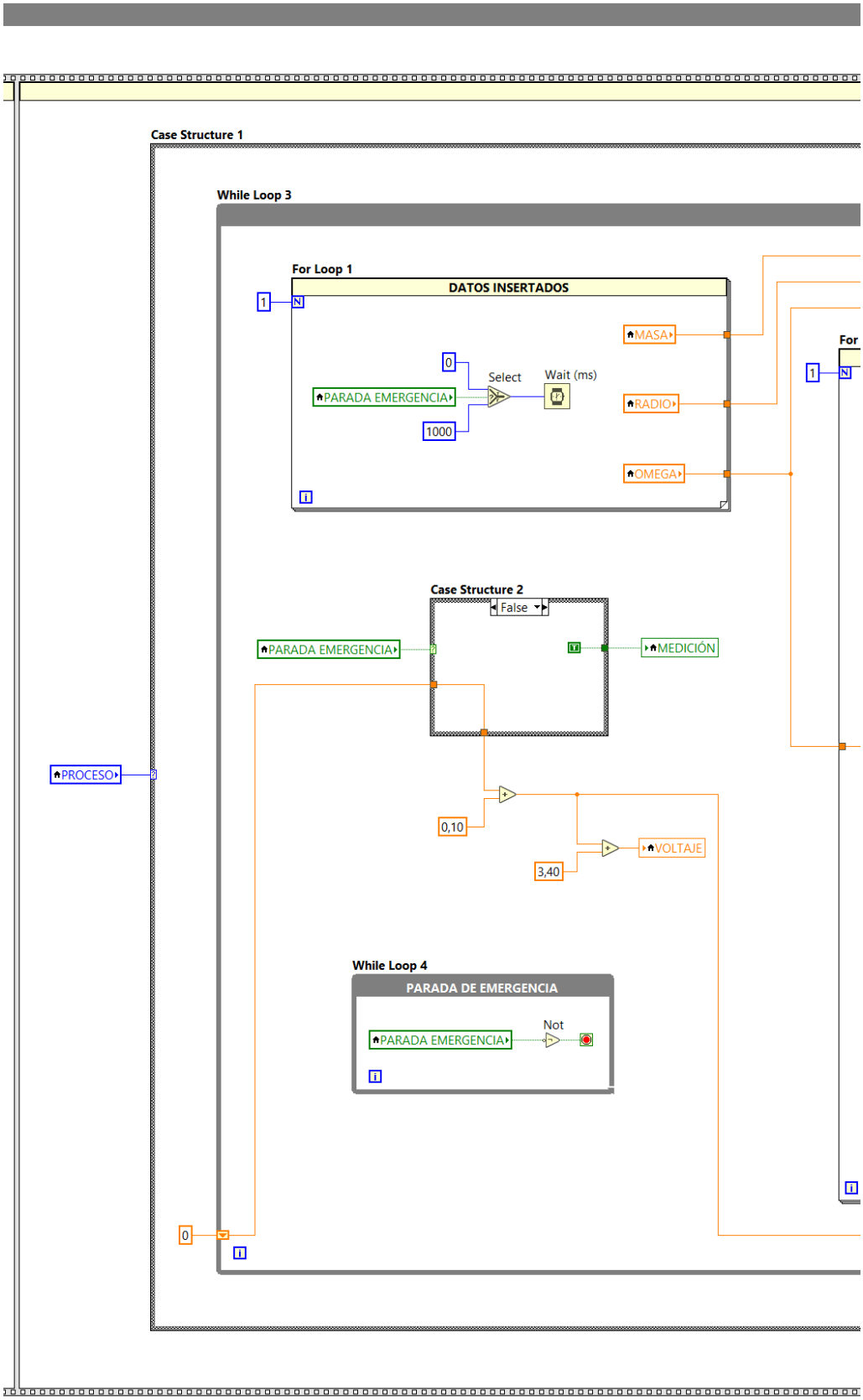


Figura 6.15. Zona B.

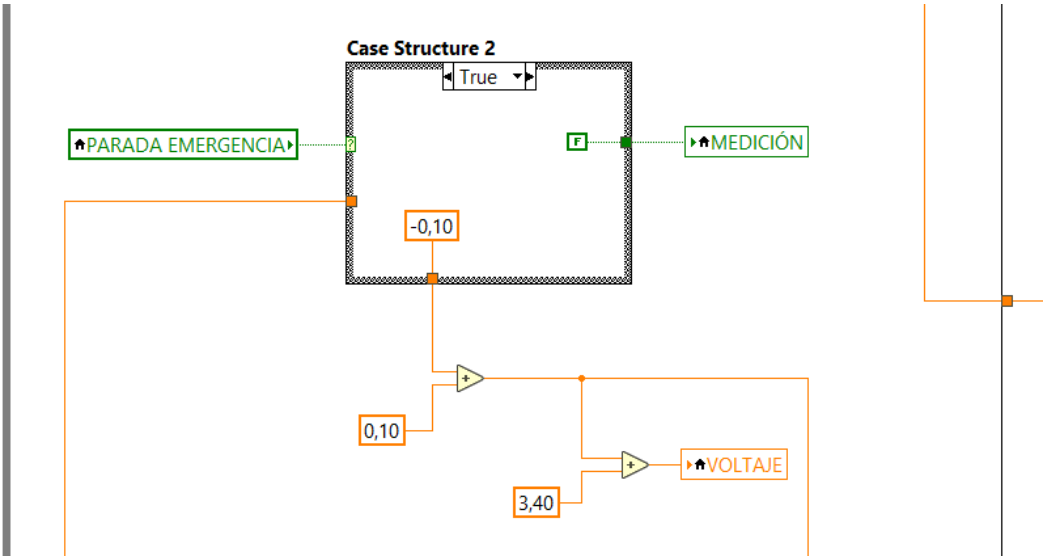


Figura 6.16. Detalle de la zona B, *Case* 2 en posición TRUE.

ZONA C

La adquisición de los datos a través de los sensores de movimiento y de fuerza se encuentra en el bucle *For* 2, con una única iteración y dos salidas. Los VI *DAQmx Create Virtual Channel.vi* y *DAQmx Start Task.vi* funcionan para que la computadora establezca contacto con los sensores a través de la tarjeta de adquisición de datos. Se crean sendos canales de comunicación y el inicio de la toma de datos. Los detalles de estos VI se encuentran en los epígrafes *5.2.3.2. Comunicación con el sensor de fuerza* y *5.2.3.4. Comunicación con el sensor de movimiento*.

Una vez establecida la comunicación y el comienzo de las medidas, los valores se dirigen al bucle *While* 5. Está configurado para efectuar la lectura gracias a *DAQmx Read.vi*. A continuación, se realiza una serie de operaciones matemáticas para convertir los datos en radianes y en newtons (paleta *Numeric*).

La conversión se detalla en los epígrafes *5.2.3.3. Configuración y calibración del sensor de fuerza* y *5.2.3.5. Configuración y calibración del sensor de movimiento*. Para obtener radianes, los grados se multiplican por el número pi y luego se dividen por 180. El bloque *Wait (ms)* significa que las muestras se reciben cada milisegundo. El bloque *To Double Precision Float* (paleta *Conversion*), tiene como cometido cambiar el formato de un número a doble precisión.

La medida del tiempo transcurrido se realiza con dos bloques *Tick Count (ms)* (paleta *Timing*). Cuenta *ticks*, de un milisegundo cada uno. Al colocar uno dentro y otro fuera del bucle *While* 5, se toma el tiempo cuando el bucle funciona. Así, cuando se entra a esta estructura, el tiempo del *Tick Count (ms*) de fuera pasa por un túnel y se le resta el tiempo del *Tick Count (ms)* de dentro. Esto cuenta el tiempo transcurrido, que está en milisegundos, y se le divide entre 1000 para convertirlo en segundos.

La condición de parada se determina con la siguiente relación: Si la omega de consigna es *r* radianes entre un segundo, para una omega de 2π radianes se tendrá un tiempo *t* (una vuelta se realizará en un determinado tiempo). Así:

$$\omega \text{ de consigna}: \left(\frac{\text{r radianes}}{\text{1 segundo}}\right) = \frac{2\pi \text{ radianes / vuelta}}{\text{t segundos / vuelta}}$$

Por ello, la parada del bucle se establece cuando el tiempo sea igual o mayor que 2π dividido por la omega de consigna (la omega que se desee en cada medición). La velocidad del motor varía en cada iteración del bucle *While* 3, por lo que el tiempo medido cambia y ofrece una omega distinta en cada iteración.

Respecto al sensor de fuerza, cada valor obtenido se trata en el bucle *While* 6. La estructura permite ampliar el rango de medidas, debido al ruido de salida, el *slew rate* y el límite del ancho de banda. El modo túnel se compone de un *Last Value* y dos *Indexing*. Estas tres salidas se juntan en el bloque *Insert Into Array*, (paleta *Array*), que introduce un elemento (dato) o un *subarray* (conjunto de datos) en un *array* en el punto que se especifique en *index*. Las entradas y salidas utilizadas de este bloque son las siguientes:

N-dim array: Es el *array* en el que se quiere insertar un elemento, fila, columna o página. La entrada procede del túnel *Last Value*. Sin embargo, como es un dato, para transformarlo en *array* se utiliza el bloque *Build Array* (paleta *Array*), que concatena varios *arrays* o añade elementos a un *array*. Para redimensionarla, se arrastra la base del bloque hacia arriba o abajo.

Index 0... N–1: Concreta la posición del *array* en el que se introduce el elemento. Aquí es la 1, que es la posición inmediatamente posterior del dato del sensor.

N o *N–1 dim array*: Es el elemento, fila, columna o página que se inserta en el *array* especificado en *n-dim array*. En este caso, son las dos salidas *Indexing*.

Output array: Es el *array* con los datos insertados.

Este *array* (conjunto de datos) es multiplicado por la constante 6,25, para convertir los datos en voltios a newtons. Los cables que representan a un *array* son más gruesos.

Como salidas del *While* 5, existe un total de seis. Cuatro salidas, las que proceden de *DAQmx Read.vi*, desembocan en los VI *DAQmx Clear Task.vi* y *Simple Error Handler.vi* (paleta *Dialog & User Interface*). Tienen como fin limpiar las tareas de la lectura y de visualizar un mensaje cuando se genere un error, tanto para el sensor de movimiento como para el de fuerza. Las dos restantes se reparten para los valores de la velocidad angular u omega y la fuerza. Esta última es de modo *Concatenating*, para que permita agrupar los conjuntos de datos generados por el sensor de fuerza.

La condición de paro del bucle *While* 3 se define cuando la omega es mayor o igual que la omega de consigna. Además, las salidas de este bucle son cuatro: masa, radio, omega y fuerza.

Los valores de la masa actual, radio y omega se utilizan para efectuar una serie de operaciones matemáticas que ofrece como resultado omega al cuadrado y $MR\omega^2$.

El *array* de la fuerza se dirige al bloque *Sort 1D Array*. Los elementos se ordenan según su valor, no por la posición del *array*. Es decir, las primeras posiciones se ocupan con elementos cuyos valores son bajos, mientras que las últimas, con valores altos. Después, el flujo de datos sigue hacia el bloque *Reverse 1D Array*, cuya función es la de revertir el orden de los elementos de un *array*. Los elementos de las primeras posiciones cambian a las últimas; y los elementos de las últimas posiciones, a las primeras.

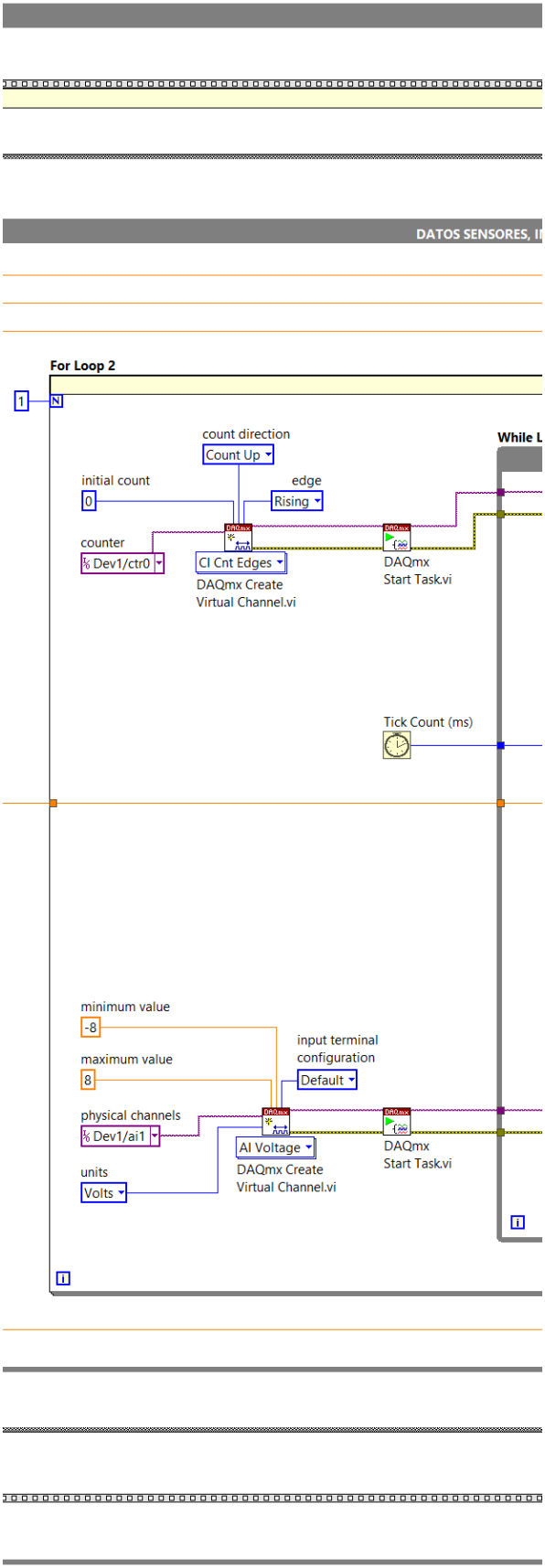


Figura 6.17. Zona C, adquisición de los datos de los sensores.

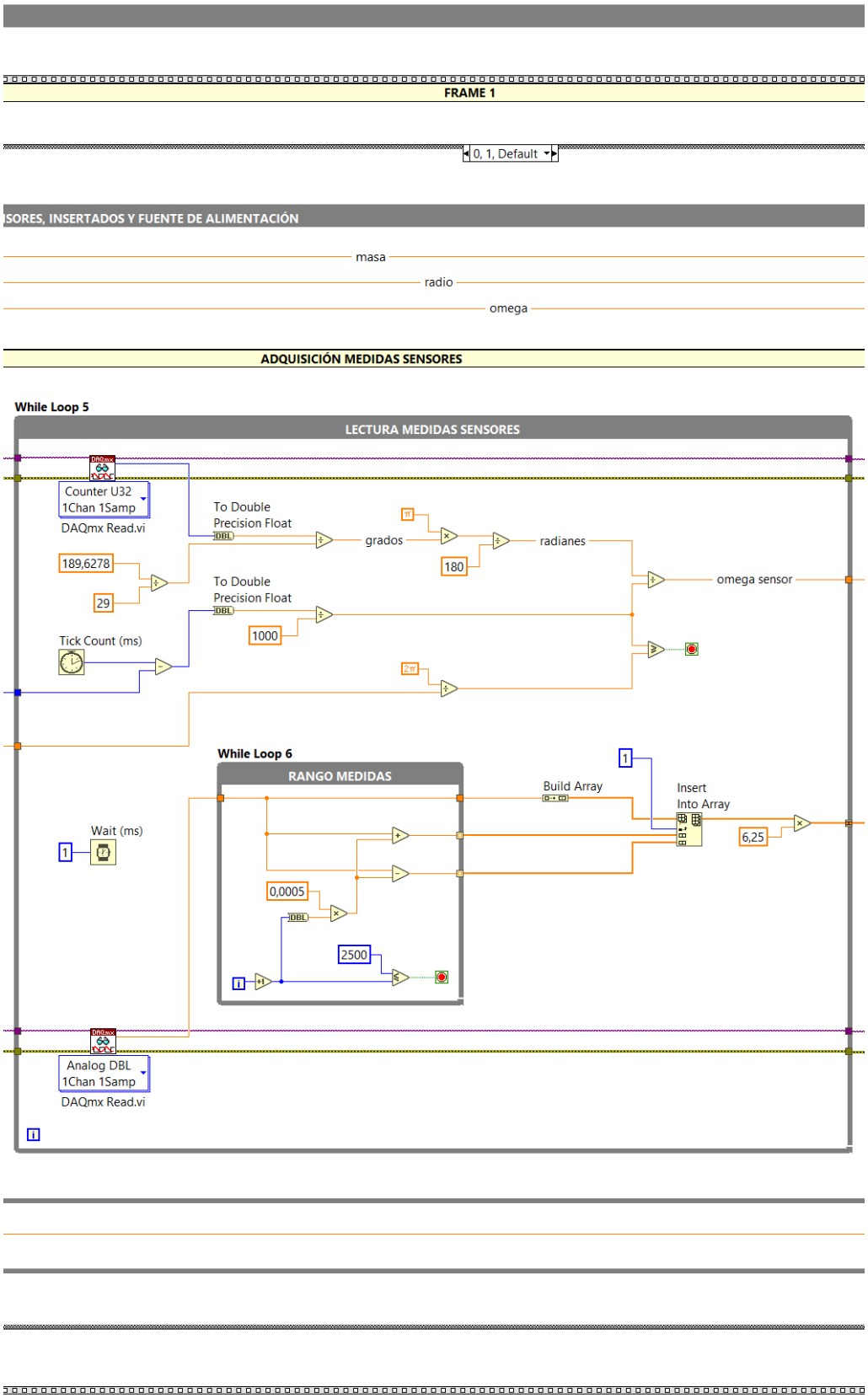


Figura 6.18. Zona C, lectura de medidas de los sensores.

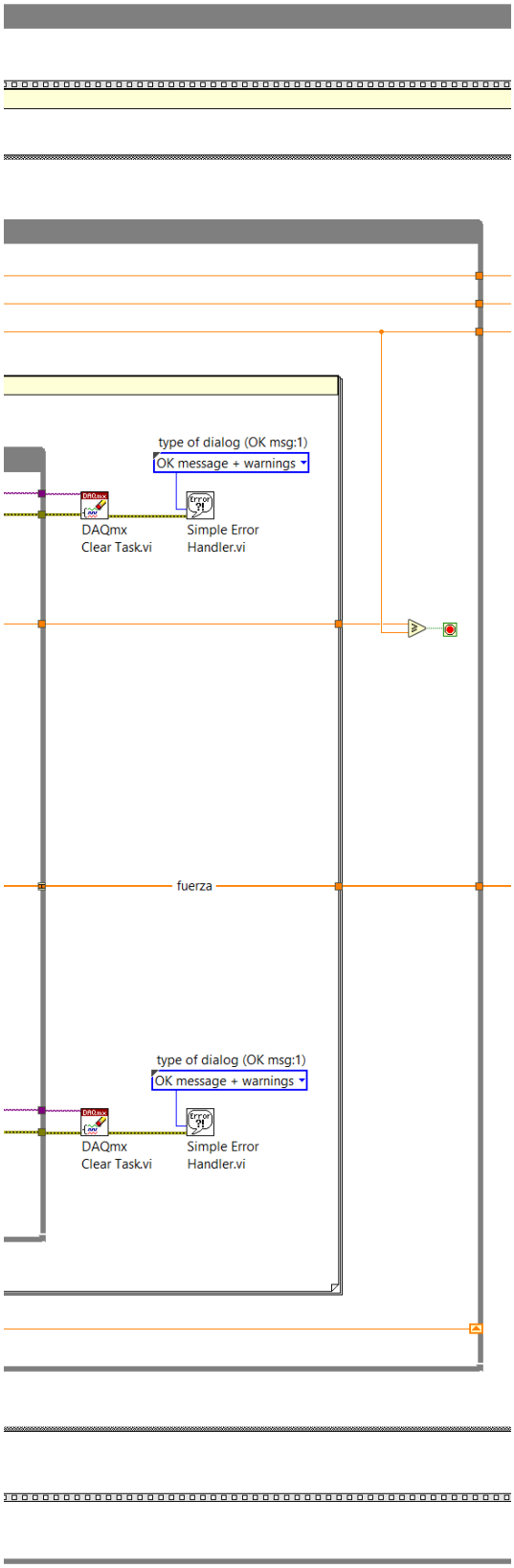


Figura 6.19. Zona C, finalización de la lectura.

- Bucle *While* 7.
  ZONA C
  Tiene como cometido crear un *array* sin repetir elementos. Los datos ordenados son la entrada de dos bloques *Index Array*.

  El bloque *Index Array* (paleta *Array*) devuelve el elemento o *subarray* de la posición indicada en *index*. Las entradas y salidas usadas se muestran a continuación.

  N-dimension array: El *array* que entra en el bloque.

  Index 0… N–1: Concreta la posición en la que se quiere extraer el elemento o *subarray*. En esta ocasión, es el número de iteración del bucle o la iteración más uno.

  Element o *subarray*: Devuelve el elemento o *subarray*.

  Un bloque extrae el elemento *i*, mientras que el otro, el elemento *i+1*. Estos dos elementos se comparan y el resultado es la condición del bloque *Select*. Si la condición es TRUE, devuelve –1; por el contrario, si es FALSE, el número de posición que ocupa el valor *i* (iteración del bucle). El valor –1 es una posición que no existe, por lo que no se extrae ningún valor.

  Esta salida del bucle *Select* es la entrada de *index* de otro bloque *Index Array*, cuya otra entrada es el *array* de la fuerza. La salida de este bloque es también la del bucle, de tipo *Indexing* y *Conditional*. La condición para que el dato salga del bucle la proporciona la comparación de los elementos *i* e *i+1*. Si son iguales, el *array* que se genera no aumenta de tamaño.

  El bucle se detiene cuando el tamaño del *array* de la fuerza es mayor o igual que el número de iteraciones más uno. Para conocer la cantidad de elementos almacenados en el *array*, se usa el bloque *Array Size*. Después, el *array* se introduce en el siguiente bucle, el *While* 8, tras pasar por un *Sort 1D Array*.

- Bucle *While* 8.
  ZONA C
  El objeto de esta estructura es conseguir dos valores próximos a $MR\omega^2$.

La estructura tiene dos *Index Array*. Uno de ellos recorre el *array* desde la posición 0 hasta la última (0, 1, 2,...), mientras que el otro recorre desde la posición N–1 hasta la 0 (N–1, N–2, etc.), siendo N el número de elementos. Las salidas de estos dos bloques son también las de la estructura, que además tienen condiciones. La condición de la primera salida es que los datos tienen que ser menores que el número negativo de $MR\omega^2$. Como el túnel tiene modo *Last Value*, el último dato extraído es inmediatamente menor al negativo de $MR\omega^2$. Sin embargo, la condición de la segunda salida estriba en que los valores son mayores que el valor negativo de $MR\omega^2$, y el dato en ese caso es inmediatamente mayor. Estos dos elementos se suman y se dividen entre dos. Así resulta el valor de la fuerza.

Las salidas de los *cases* 0 y 1 se corresponden a los siguientes datos: masa, radio, omega, omega al cuadrado, $MR\omega^2$ y fuerza.

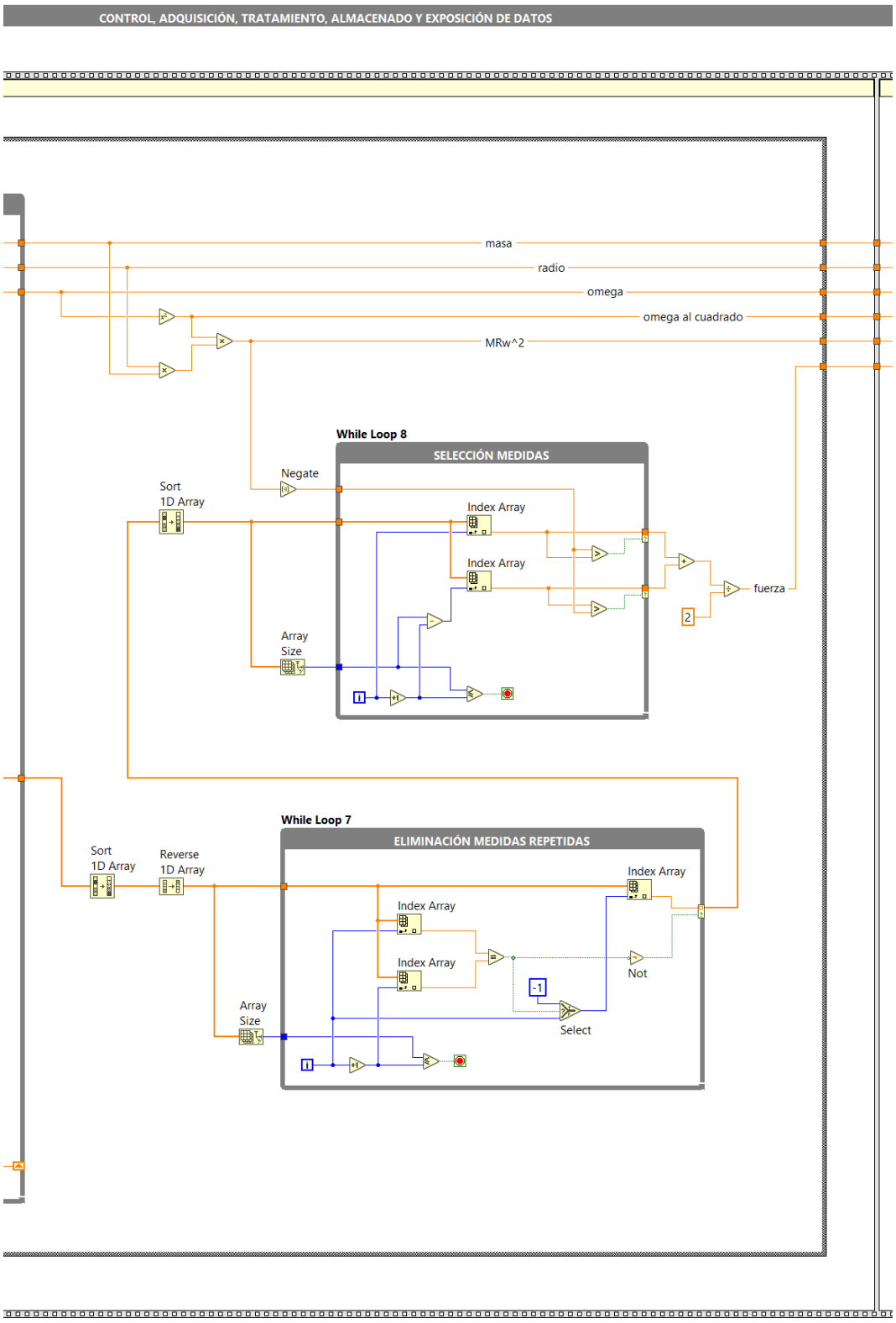


Figura 6.20. Zona C, procesado de las medidas del sensor de fuerza.

- *Cases* 2 y 3.
  ZONAS B y C
  Esta opción se produce cuando se quieren suprimir medidas o borrar todas las medidas de la tabla de datos. Si se selecciona, proporciona el valor 0 a los cables de la masa, radio, omega, omega al cuadrado, $MR\omega^2$ y fuerza. Otra de las acciones que realiza es la de entregar la constante FALSE a la variable local MEDICIÓN. Así, el motor no gira y no se adquieren datos.

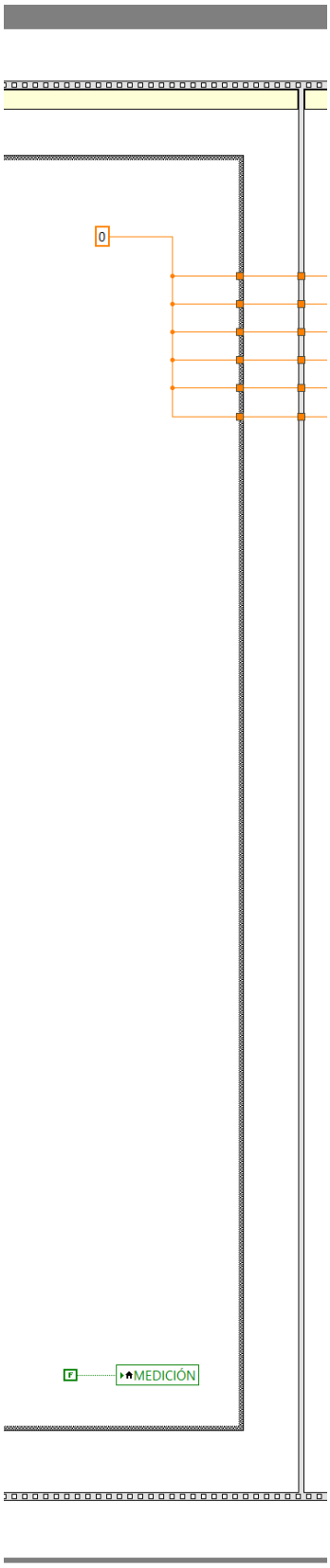


Figura 6.21. *Cases* 2 y 3 de las zonas B y C.

6.2.2.1. Tabla y datos introducidos manualmente

ZONA F

El bucle *While* 9 contiene los elementos que controlan la tabla de datos. El indicador *Table Control* 1 (paleta *List, Table & Tree*) se cablea a *Index Array*, para extraer dos elementos de la tabla (el radio y el omega tecleados en ella). Las columnas son constantes, mientras que las filas varían en función de la estructura *Case* 8:

- *Case* 0.
  El número de fila se corresponde con el número de filas totales que dispone la tabla, gracias a una variable local (TAMAÑO).

- *Cases* 1 y 2.
  Si el valor introducido en el panel de control (PROCESO) es igual o menor que el número de filas de la tabla de datos, se cumple la condición TRUE del bloque *Select*, y al valor se le resta uno. Así coincide con la posición de la tabla (la fila uno es la posición 0; la fila 2, posición 1; etc.). Si la condición es falsa, *Select* entrega 1.

- *Case* 3.
  La estructura entrega 0.

Las dos salidas de *Index Array* se llevan a dos bloques *Fract/Exp String To Number* (paleta *Number String Conversion*), que convierte el carácter *string* en número. La entrada y la salida usadas son:

String: Puede ser un *string*, un *cluster* de *string*s, un *array* de *string*s o un *array* de *clusters* de *string*s. Si *string* contiene los caracteres Inf o NaN, este bloque devuelve Inf o Nan, respectivamente. En esta aplicación, devuelve la posición seleccionada de la tabla.

Number: Puede ser un número, un *cluster*, un *array* de números o un *array* de *clusters*, dependiendo de la estructura del *string*. En este caso, un número.

Los valores del radio y del omega se comparan. De esta manera se evita que se inserte cualquier número en la tabla. Así, para el radio, el valor está comprendido entre 0,06000 y 0,22000 metros; mientras que para la omega, entre 5,00000 y 15,00000 rad/s. Si el número está fuera de este rango, se devuelve 0. Después, los números se transforman a formato *string* (*Number To Fractional String*). En concreto, este bloque convierte un número en un

formato *F-format* (notación fraccional) de *string* punto flotante. Las entradas y salidas consideradas se muestran a continuación.

Use system decimal point: Define cómo es la separación de la parte entera de la parte decimal. Si el equipo informático tiene configurado el punto como separador, es indiferente este control *booleano*. Por el contrario, cuando el símbolo es una coma, en el caso de TRUE, el separador es una coma; si es FALSE, un punto. Se decide por TRUE.

Number: Es el dato de entrada. Puede ser un escalar, un *array* o un *cluster*.

Precision: Es un escalar. Precisa cuántos dígitos tiene la parte decimal del dato de salida. Se elige 5.

F-format string: Es el dato de salida con las condiciones de entrada del bloque.

Después, los datos transformados en cadena de caracteres desembocan en dos sendos *Invoke Node* (*Set Cell Value*), que realiza una acción sobre una referencia, en este caso, la tabla de datos. *X Index* sirve para definir la columna, *Y Index*, la fila; y *Value* recibe el dato. El *Invoke Node* se crea con este menú contextual (figura 6.22).

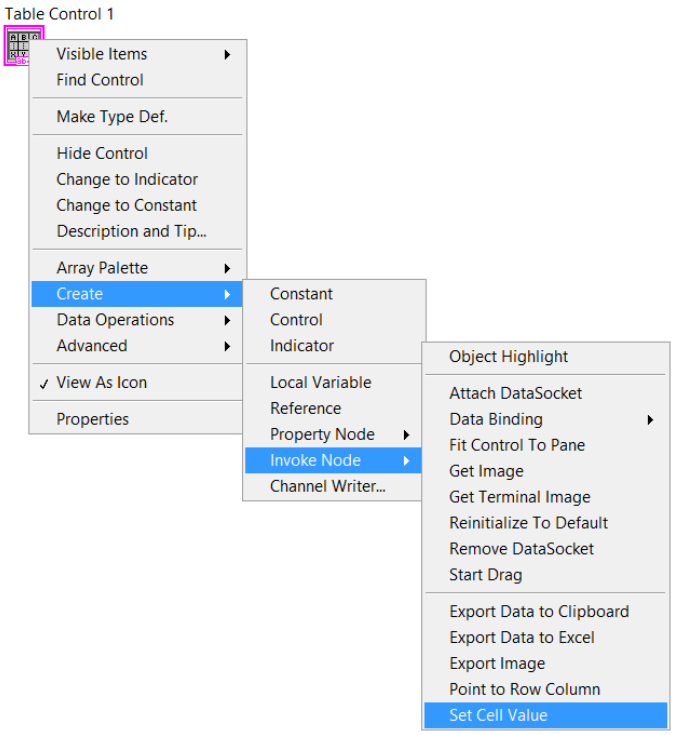


Figura 6.22. Creación de un *Invoke Node*.

La tabla de datos enmarca con color azul las celdas donde se teclean los datos del radio y el omega. Para ello, es necesario insertar un *Property Node*. Los *property nodes* o nodos de propiedad son variables que dependen únicamente del terminal a partir del cual se han creado y que permiten leer o modificar atributos del panel frontal de un control o indicador (cambio de color, desactivación, o lectura de posiciones de cursores, por ejemplo). Para crear un *Property Node* se cliquea con el botón derecho del ratón sobre cualquier control del panel frontal o terminal del diagrama de bloques. Se selecciona la opción *Create/Property Node*.

El nodo puede ser de escritura como de lectura. Una pequeña flecha a la izquierda del nodo indica que es de escritura, mientras que una flecha a la derecha informa de que es de lectura. Con el botón derecho del ratón se especifica el tipo de *Property Node* asociado a dicho control (*Table Control* 1). En este caso, *SelSize* y *SelStart*, y de escritura.

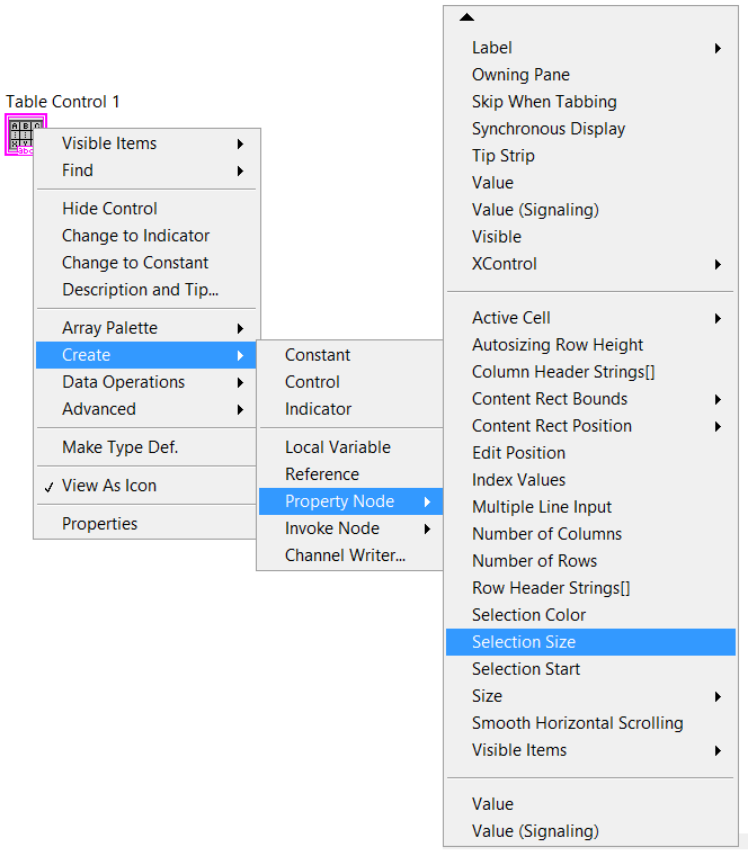


Figura 6.23. Creación de un *Property Node*.

Al insertar el *Property Node Selection Size*, se redimensiona el bloque y aparece *Selection Start*. *SelSize* define cuántas celdas se enmarcan. En este caso, dos celdas por fila. *SelSart* establece la posición donde comienza el enmarque de las celdas. El valor proviene de otro *Propery Node*, del *cluster*

Selection Start. Este *cluster* contiene dos números. Uno es una constante, 2, para la columna. El otro valor proviene de la estructura *Case* 8, para la fila.

Para la masa, la tabla muestra la masa total en cada fila y la última masa usada en la nueva medida. El usuario inserta el valor de la masa añadida en el control numérico y elige si sumarlo o restarlo a la masa total. También puede cambiar la masa total por el valor introducido. Así, la estructura *Case* 7 cuenta con las siguientes opciones:

- *Case* 0.
  Por defecto. Suma el valor de la última masa con el valor de la masa que se le incorpora.

- *Case* 1.
  Resta el valor de la última masa con el valor de la masa que se extrae.

- *Case* 2.
  No realiza ninguna operación.

La salida de esta estructura se compara (el valor debe estar comprendido entre 0,01000 y 0,20000 kg). Fluye por el bloque *Number To Fractional String* y entra en un *Set Cell Value*.

Una estructura *Case* (la número 9) se encarga de comprobar y de mostrar el número de fila introducido en el panel de control.

- *Case* 0.
  Es el número totales de filas más uno. Sirve para indicar que los datos de la nueva medida se introducen después de la última fila.

- *Case*s 1 y 2.
  El usuario introduce el número de fila a cambiar o suprimir. Si el valor introducido excede del número total de filas, el control muestra 1.

- *Case* 3.
  Devuelve 0.

En cada opción, aparece un *Property Node*, *Value*, para comprobar el dato introducido o que no se cambie el valor mostrado.

Por otra parte, el bucle también contiene un indicador *String*, unido a una constante *String* que alberga una ayuda para el funcionamiento del programa:

1. Comprobar el voltaje de las baterías y de todas las conexiones.

2. Una vez pulsado el botón RUN, se introducen los datos del radio y del omega en la tabla. La masa se teclea en su cajetín y se selecciona la operación.

3. Se escogen en la gráfica los parámetros de los ejes X e Y.

4. Se elige el tipo de proceso. Si se cambia o eliminan medidas, se teclea el número de fila. Se acciona el pulsador.

5. Los datos pueden exportarse tras finalizar cualquier medida.

6. PARADA EMERGENCIA se acciona para detener la plataforma. Tras desactivar el pulsador, vuelve a girar con el mismo tipo de proceso.

También está el elemento *Tab Control*, con dos pestañas, una para la tabla de datos y otra para la ayuda. El bucle *While* 9 funciona continuamente y se ejecuta cada 500 milisegundos.

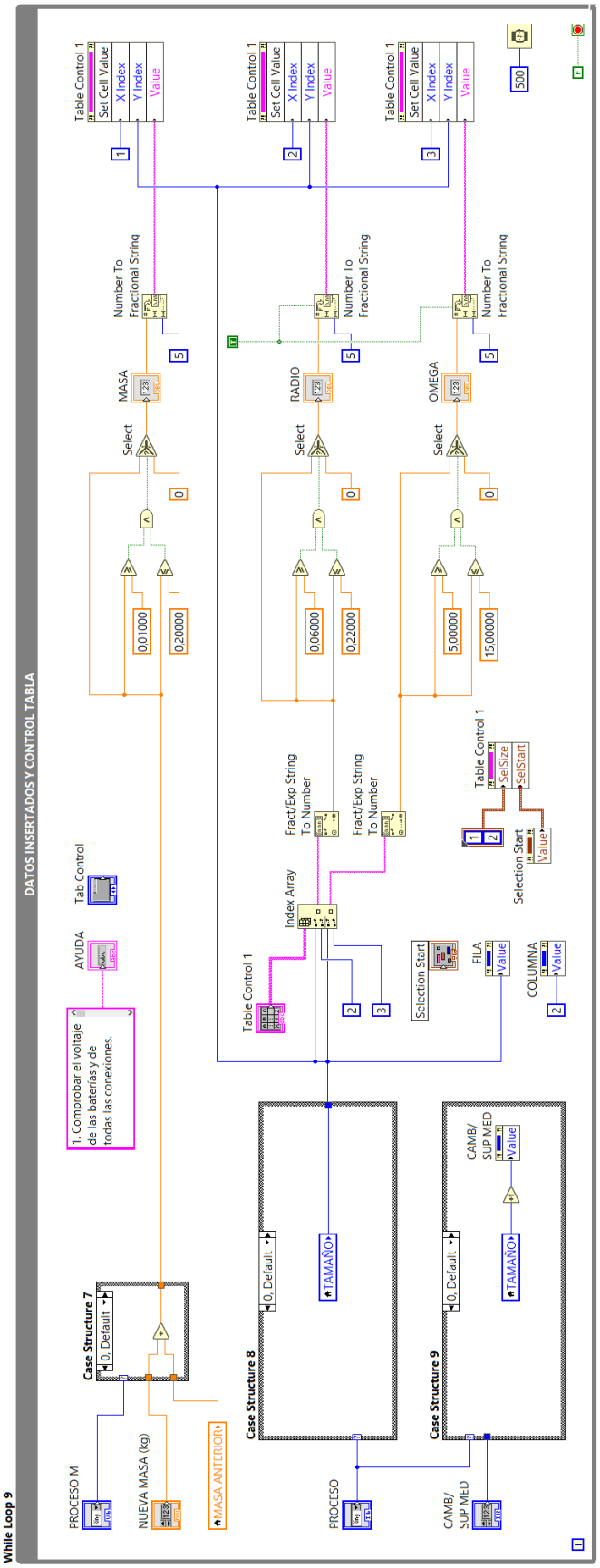


Figura 6.24. Zona F.

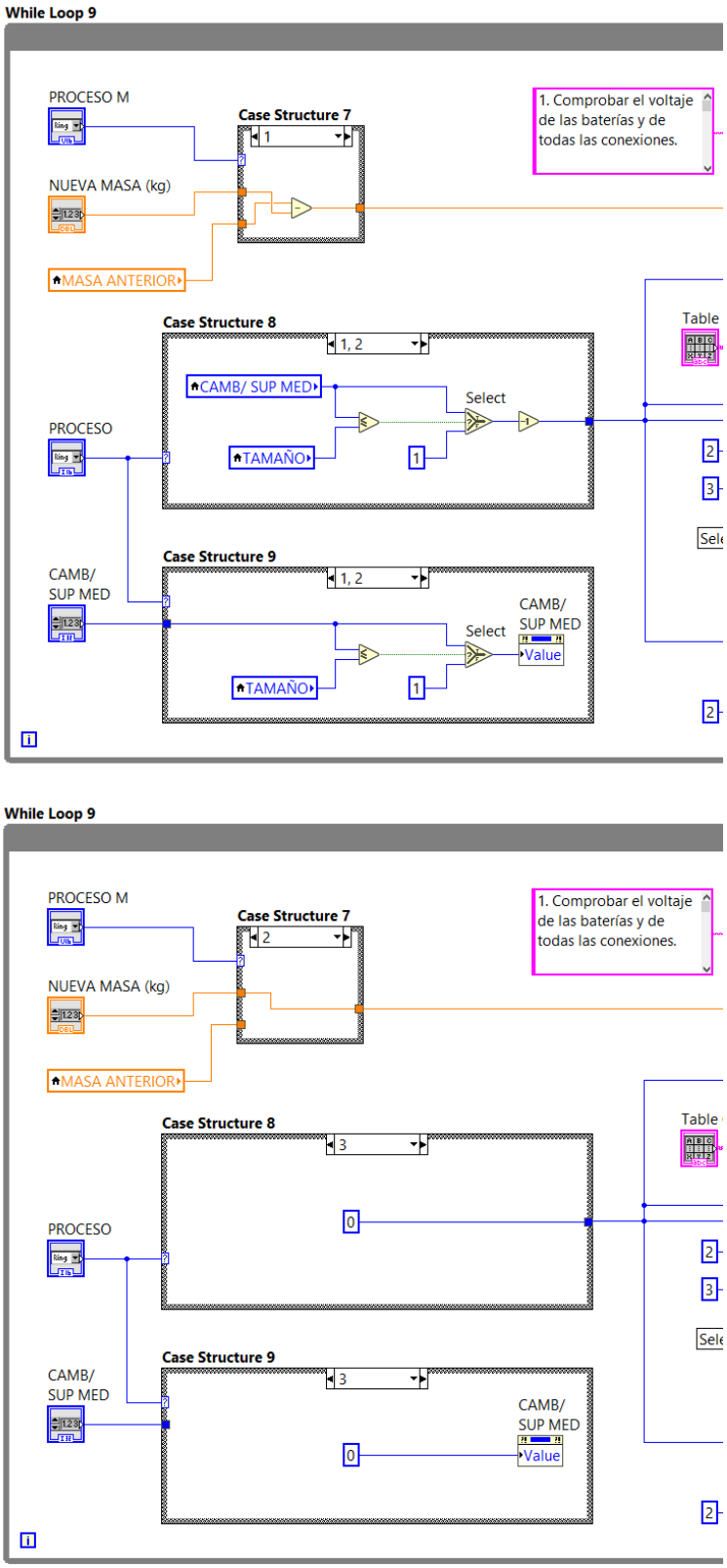


Figura 6.25. Zona F, opciones 1 y 2 del *Case* 7 y opciones 1, 2 y 3 de los *Cases* 8 y 9.

6.2.3. Acumulación y tratamiento de los datos

El *frame* 2 (*While* 1) agrupa varios bucles y estructuras. Se encuentra en las zonas **D** y **E**. Además se ubica otra variable local, MEDICIÓN, con una constante FALSE. La zona **D** se encarga de registrar y ordenar los datos, así como de advertir del fin de las mediciones.

ZONA **D**

El bucle *For* 3 establece un aviso sonoro para avisar de que las medidas han sido adquiridas y la plataforma deja de girar.

La señal sonora se construye de la siguiente manera. Primeramente, un VI, *Sound Output Configure.vi*, determina un sonido de salida. Es importante destacar que Windows debe tener instalado DirectX 8.0 o versiones posteriores para usar este VI. Además, es obligatorio que Linux disponga del driver *Open Sound* System (OSS). Sus entradas y salidas empleadas son estas:

Number of samples/channels: Especifica el número de muestras por canal en el *buffer*. Un número grande de muestras sirve para operaciones continuas, mientras que un número pequeño es ideal para usar poca memoria. Se ajusta a 1000.

Sample mode: Establece muestras finitas o continuas. Las finitas se relacionan con el número de muestras determinado en la entrada de este VI. Las continuas se repiten. Se elige *Continuous Samples*.

Sound format: Ajusta la tasa, el número de canales y los bits por muestra en la operación de sonido. El valor de cada uno de ellos depende de la tarjeta de sonido. En *Sample rate* (S/s), comúnmente por defecto, es 22050 S/s. El número de canales está relacionado con los que la tarjeta soporte. Para la mayoría de ellas, 1 es mono y 2, estéreo. Los bits por muestra determinan la calidad de cada muestra en bits. Normalmente son de 8 o 16 bits. En la aplicación, los datos son 22050, 2 y 16, respectivamente.

Task ID: Devuelve un número de identificación relacionado con la configuración.

Error out: Contiene información sobre errores.

Después, dentro del bucle, *Sound Output Set Volume.vi* ajusta el volumen. En la entrada *volume*, un 0 es silencio, mientras que 100 significa lo más alto.

Se opta por 100. A continuación, *Simulate Signal Express* genera una onda senoidal, cuadrada, triangular, de diente de sierra o ruido. Se escoge onda senoidal. Hay que cliquear dos veces para configurar este VI. La salida es DC. Además, hay que establecer la frecuencia, en este caso, de 1000 Herzios. Esta salida pasa por *Convert from Dynamic Data Express*, que transforma una señal dinámica, en este caso una onda senoidal, en un conjunto de datos numéricos.

El dato, tras salir del bucle, se dirige a *Sound Output Write.vi*. Las entradas y salidas utilizadas son:

Task ID: Recibe la señal configurada previamente.

Data: Escribe el sonido al *buffer* interno.

Task ID out: Es el sonido manipulado.

Error in/out: Contiene información de los errores.

Tras salir del bucle, el dato entra en *Sound Output Clear.vi*, que limpia la tarea de sonido. Finalmente, *Simple Error Handler.vi* muestra por pantalla un mensaje de error si se produce. Los VI de sonido se encuentran en las paletas *Graphics & Sound*, *Waveform* y *Dialog & User Interface*. También están disponibles en uno de los ejemplos de LabVIEW, desde el menú *Help/Find Examples...* (panel frontal o diagrama de bloques). En *NI Examples Finder*, cliquear en *Browse* y seleccionar *Directory Structure*. En la ruta de carpetas se sigue la siguiente: *Graphics and Sound/Sound*. El archivo es Generate Sound.vi.

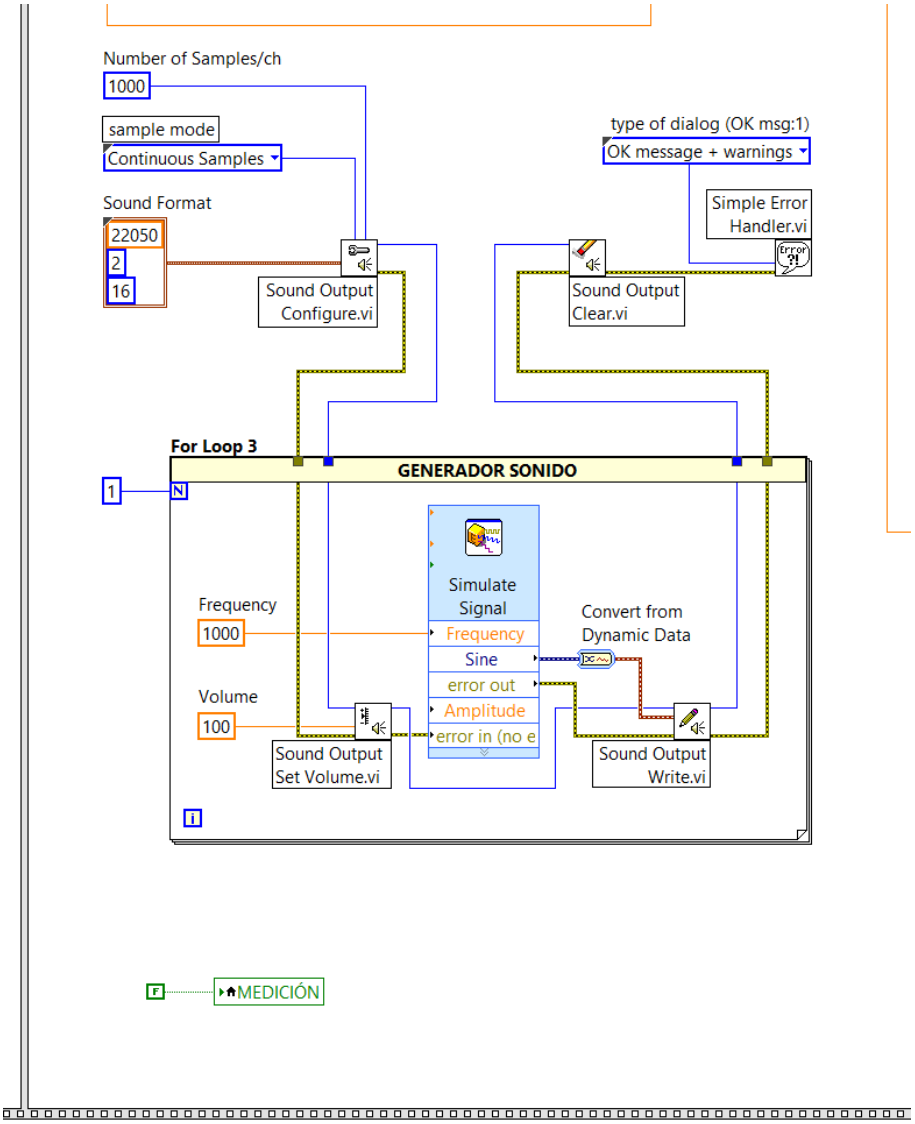


Figura 6.26. Zona D, estructura de sonido.

Del *frame* 1 proceden los datos de la masa, radio, omega, omega al cuadrado, $MR\omega^2$ y fuerza. El período se calcula dividiendo 2π por omega. Después, se agrupan en un *array* de nueve elementos tras su paso por el bloque *Build Array*. Este *array* fluye al bucle *For* 4.

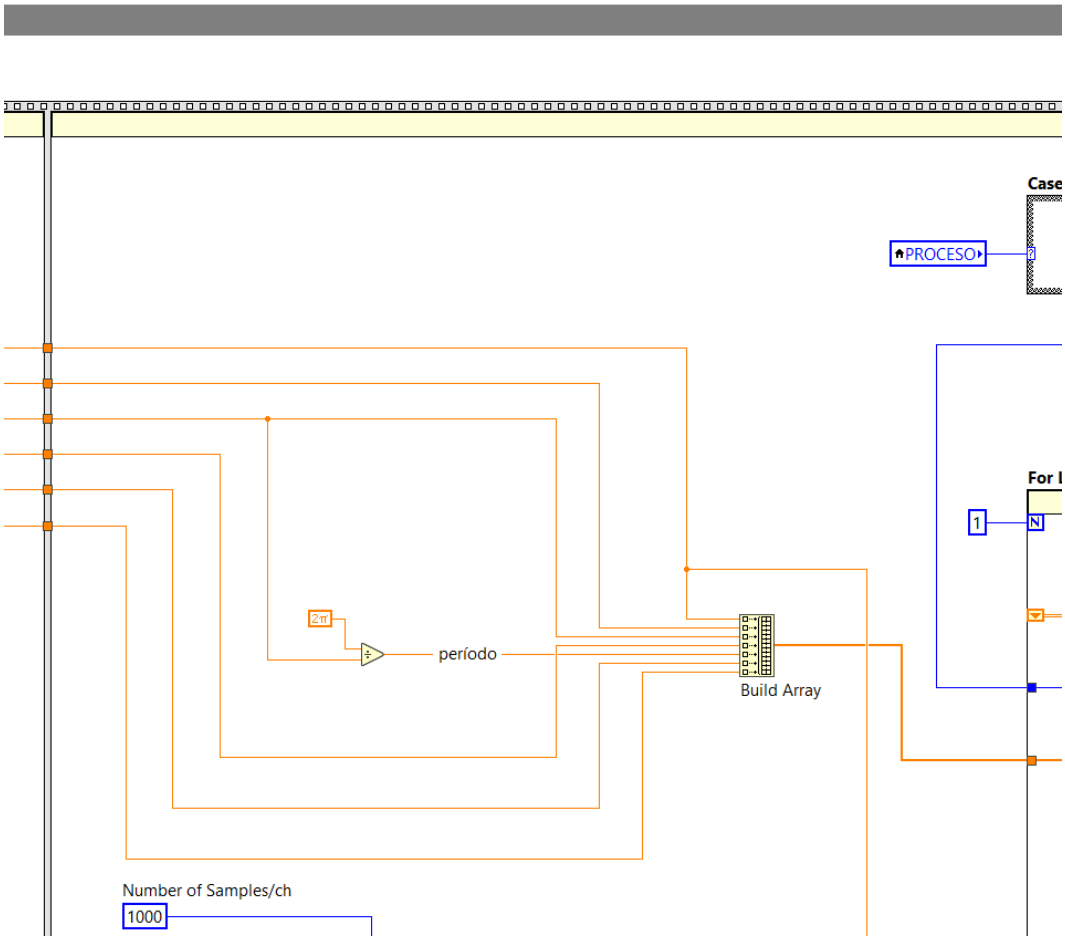


Figura 6.27. Detalle de la zona D, agrupación de datos.

El bucle *For* 4, con una única iteración, acumula, cambia o elimina los datos de cada medición. Crea tres *arrays*: uno para los datos, otro para la masa y otro para el número de medidas.

El *array* conteniendo los datos de la última medición realizada entra al bloque *Array Subset* (paleta *Array*), que extrae una parte del *array* desde la posición y con el número de elementos que se le indique. Las entradas y salidas se describen a continuación.

Array: Puede ser un *array N-dimensional* de cualquier tipo.

Index: Especifica el primer elemento, fila, columna o página a incluir en la parte del *array* que se quiere obtener. Si *index* es menor que 0, el bloque entiende que es un 0. Si *index* es mayor o igual que el tamaño del *array* de entrada, la función devuelve un *array* vacío. Se usa 0.

Length: Concreta cuántos elementos, filas, columnas o páginas se incluyen en el *array* de salida. Si *index* más *length* es mayor que el tamaño

del *array*, el bloque devuelve solamente los datos disponibles. El valor predeterminado de *length* es desde *index* hasta el fin del *array*. Si *length* es menor que 0, la función la trata como 0. La estructura *Case* 4 suministra el valor.

Subarray: Es del mismo tipo que el *array* de entrada.

Después, la salida de este bloque se conecta a una de las entradas de *Build Array*. La otra entrada procede de un *Shift Register* o registro de desplazamiento. Un mismo bucle puede tener varios registros de desplazamientos. De hecho, la estructura *For* 4 cuenta con un total de cuatro. Puede trabajar con cualquier tipo de datos, siempre y cuando los datos que se conecten a cada terminal sean del mismo tipo. Al finalizar la ejecución de todas las iteraciones, el último valor queda en el terminal de la derecha.

El bloque *Build Array* construye un *array 2D*, compuesto por filas y columnas. Cada fila contiene los datos de cada medición, mientras que cada columna los datos de cada magnitud. El cable se representa por dos líneas finas paralelas. Este *array* es una de las entradas de *Array Subset*, que además tiene estas otras entradas y salidas:

Array: Puede ser un *array N-dimensional* de cualquier tipo.

Index: Se le conecta 0, para que el bloque trabaje desde la primera fila.

Length: La estructura *Case* 4 provee el valor.

Subarray: Es del mismo tipo que el *array* de entrada.

Los otros *index* y *length* se refieren a las columnas. Estas entradas están desconectadas, para que todas las columnas estén incluidas.

A continuación, el bloque *Delete From Array* (paleta *Array*) se encarga de eliminar un elemento o *subarray* de *N-dim array* a partir de una determinada posición.

N-dim array: Es el *array* en el cual se quiere eliminar un elemento, fila, columna o página. La entrada es el *array 2D*.

Length: Determina cuántos elementos, filas, columnas o páginas se suprimen. Se conecta 1.

Index 0... N–1 (row): Especifica la posición del elemento, fila, columna o página a eliminar. Esta posición se determina mediante la estructura *Case* 3.

Index 0... N–1 (col): Esta entrada está deshabilitada, ya que no se suprimen columnas.

Array w/subset deleted: Devuelve el *array* sin el elemento, fila, columna o página sustraídos. Si la estructura *Case* 3 está en la opción 1 y 2, sería el *array* sin una determinada fila.

Tras pasar por estos bloques, el *array 2D* cruza por el VI *Sort 2D Array.vim*. Reordena las columnas o filas de un *array 2D* según los elementos de una columna o fila. El orden se realiza de manera ascendente.

2D Array: Especifica el *array 2D* a ordenar. Esta entrada acepta *arrays* de cualquier tipo, excepto *refnums*.

Dimension to index (column): Concreta la dimensión del *array 2D*. No es necesario habilitar esta entrada.

Index: Determina la posición de la columna o fila con los elementos que se quieren ordenar. La entrada es la constante 5, correspondiente a la columna de los datos de MRω^2.

Sorted 2D Array: Devuelve el *array* ordenado.

Otra de las entradas de este bucle *For* es la que se concierne a la masa. El dato se dirige al bloque *Insert Into Array* (paleta *Array*).

N-dim array: La entrada está conectada a un *Shift Register*.

La dimensión del *array* se determina por el número de datos acumulados. La estructura *For* 4 es de una iteración, por lo que el *array* aumenta de tamaño en cada iteración del bucle general *While* 1.

Index 0... N–1: Concreta la posición del *array* en el que se introduce el elemento. Esta posición viene de la estructura *Case* 4. Los datos insertados se desplazan hacia las posiciones mayores. Es decir, cada vez que entra un dato, los elementos almacenados del *array* se desplazan a su respectiva posición siguiente.

N o *N–1 dim array*: El dato llega desde el *frame* anterior.

Output array: Es el *array* con los datos insertados.

Por otra parte, el número de mediciones se calcula mediante un *Shift Register*, que guarda el resultado de la suma del último valor más uno. Empieza sumado 0 más 1, después 1 más 1, 2 más 1,... y así sucesivamente. Cada valor es la entrada del bloque *Insert Into Array*, también conectado a otro *Shift Register* para la creación de un *array*. Después, este *array* transita por el bloque *Reverse 1D Array*. Después, cruza el bloque *Sort 1D Array*, para ordenar los elementos de menor a mayor.

La estructura *Case* 3 facilita el número de la posición del bloque *Delete From Array*, situado en el bucle *For* 4 La selección de casos se realiza mediante la variable local del *Ring* PROCESO.

- *Cases* 0 y 3.
  Devuelve –1. Esto significa que no se elimina ningún valor, porque las posiciones de los *arrays* están comprendidas entre 0 y N–1 (siendo N el número de elementos del *array*).

- *Cases* 1 y 2.
  Aquí se le resta 1 al valor introducido en el *Numeric Control* del PROCESO (variable local), para enviar el número de posición en la que se eliminan las medidas. En un *array*, la primera posición es la 0; la segunda, la 1; etcétera.

La estructura *Case* 4 tiene cuatro opciones y está controlada por la variable local PROCESO. Sus tres salidas contribuyen a la creación del *array 2D* de los datos, del *array* de la masa y del *array* del número de mediciones.

- *Case* 0.
  La primera salida está unida a la constante 7. Cuando se introduce el *array* de siete elementos al bucle *For* 4, el *Array Subset* devuelve esos mismos elementos.

  La segunda también tiene la misma función. Establece el número de filas del *array 2D*. Un *Array Size* entrega un *array* con los números de los tamaños correspondientes a los *arrays* que componen el *array 2D*. Así, el *array* tiene dos elementos, uno con el tamaño del *array 2D*, y el otro con el tamaño del *array 1D* de los siete elementos.

  Un bloque *Index Array* extrae la posición 0 (donde está el valor del tamaño del *array 2D*) y se le suma 1. Cuando entra una nueva

medición con su *array* de siete elementos, aumenta el tamaño del *array 2D*.

La última salida establece la posición donde entra el valor de la masa en el *array*. El nuevo dato se coloca en la posición 0.

- *Case* 1.
  La primera salida también está conectada a la constante 7, con el mismo propósito. Por su parte, la segunda también especifica el número de filas del *array 2D*. Como se cambia una fila, el número corresponde con el tamaño del *array 2D* considerando el *array* de la última medición. Para la tercera salida, se le conecta –1.

- *Case* 2.
  La primera salida está conectada a 0. La segunda, extrae el tamaño del *array 2D* sin el *array* a introducir, ya que esta opción es para la eliminación. La última salida entrega –1, para que así no entre el valor al *array* de la masa.

- *Case* 3.
  El valor 0 está unido a la primera y segunda salidas, y –1 a la tercera.

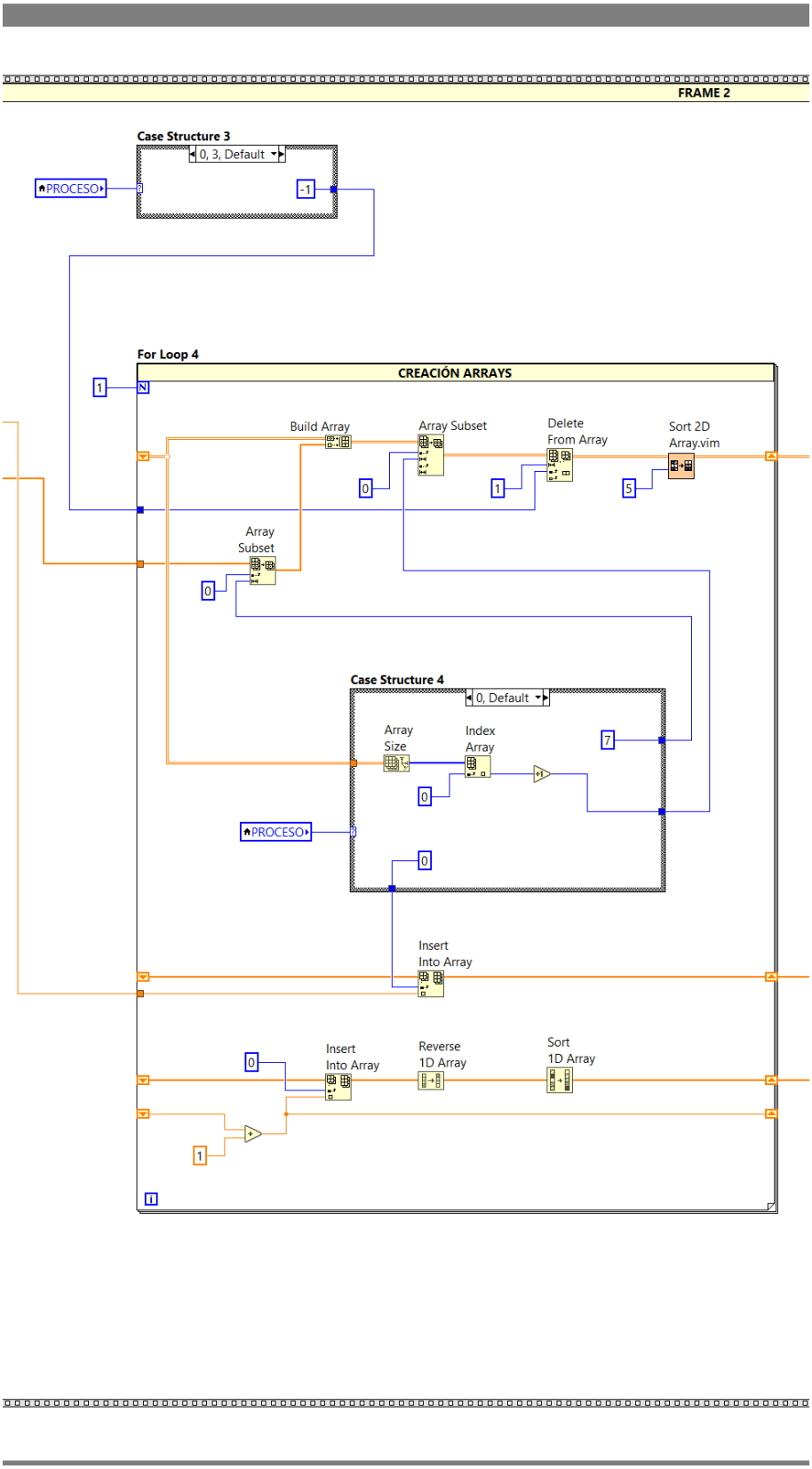


Figura 6.28. Zona D, creación de *arrays* con las estructuras *Cases* 3 y 4 con las opciones por defecto.

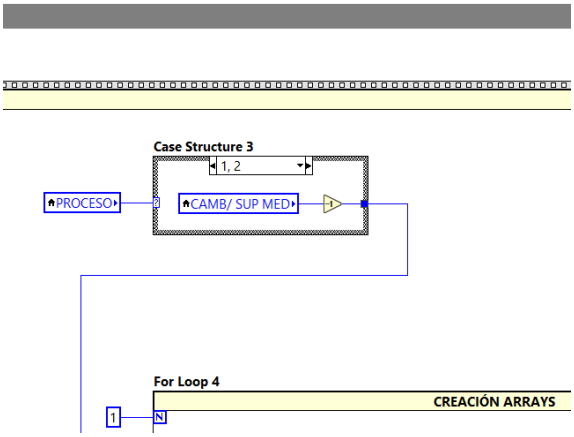


Figura 6.29. Detalle de la zona D, *cases* 1 y 2 de la estructura *Case* 3.

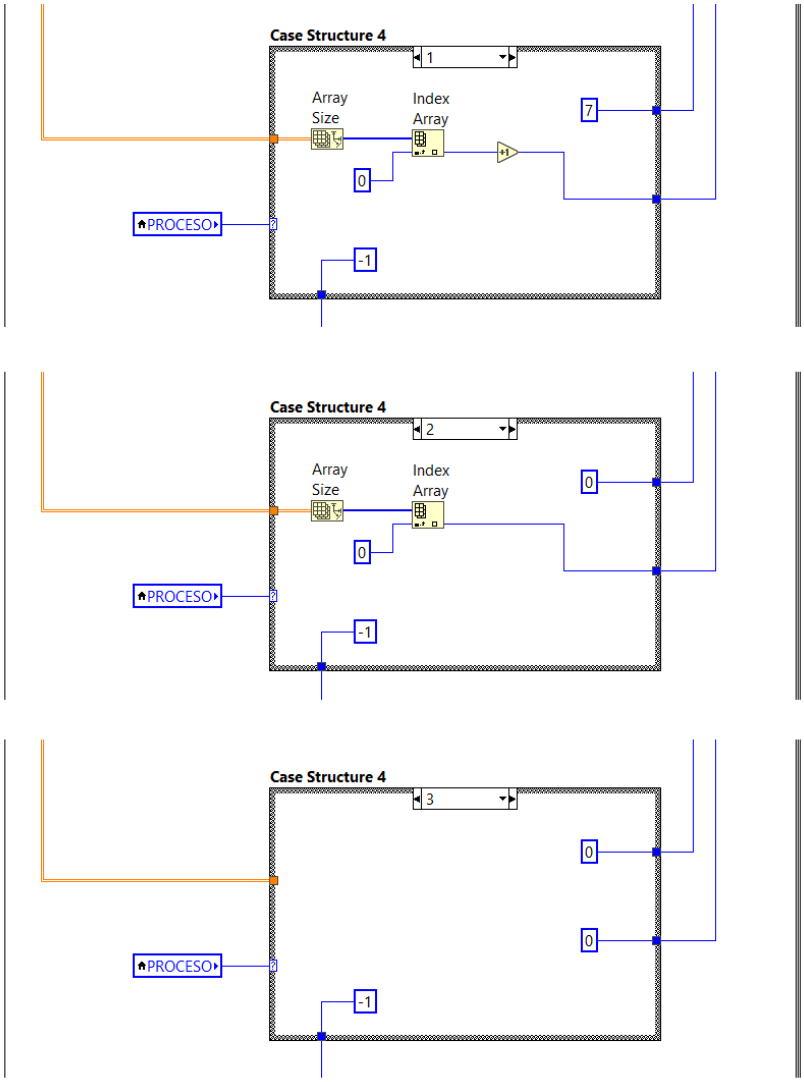


Figura 6.30. Detalle de la zona D, opciones 1, 2 y 3 de la estructura *Case* 4.

6.2.4. Exposición de los datos

ZONA E

El bucle *For* 4 dispone de tres salidas. La primera de ellas contiene el *array 2D* de los datos de cada medición. Un ramal se dirige a los bloques *Array Size* e *Index Array* para conocer el número de filas del *array 2D*.

La segunda salida guarda las masas introducidas. Este *array* sirve para conocer el último valor insertado, que se usa para el bucle *While* 9. El *array* entra en *Index Array* y se extrae este valor.

La tercera salida, correspondiente al *array* del número de medidas, fluye hacia el bloque *Array Subset*, cuyas entradas y salidas usadas son estas:

Array: Puede ser un *array N-dimensional* de cualquier tipo. Se origina en el bucle *For* 4.

Index: Se usa 0.

Length: La entrada proviene de *Index Array*.

Subarray: Es del mismo tipo que el *array* de entrada.

Este bloque se configura así para conocer cuántas medidas se realizan, descartando las eliminaciones o los cambios.

Los datos se muestran en una tabla y una gráfica tras cada medición. Así, se analiza en tiempo real la experimentación, sin esperar hasta la finalización de un número determinado de mediciones. Por un lado, los datos se tabulan para mostrarlos en una tabla, situada en el panel de control. Por otra parte, una estructura calcula la ecuación y el ajuste de mínimos cuadrados, además de graficar las medidas.

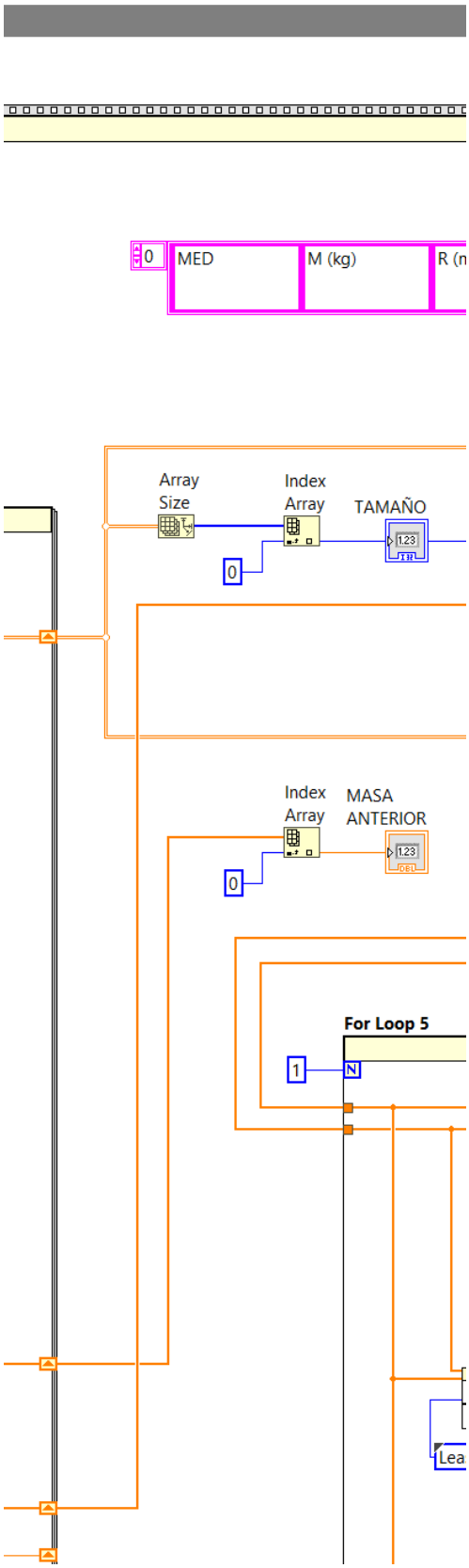


Figura 6.31. Zona E, salidas del bucle *For* 4.

PRESENTACIÓN DE LOS DATOS

El *array 2D* y el del número de medidas se direccionan a sus respectivos bloques *Number To Fractional String*. Las entradas y salidas empleadas son las siguientes:

Use system decimal point: Se establece TRUE.

Number: Es el dato de entrada. Puede ser un escalar, un *array* o un *cluster*.

Precision: Para el *array 2D*, es 5; y para el número de mediciones, 0.

F-format string: Es el dato de salida con las condiciones de entrada del bloque.

Después, los datos transformados en cadena de caracteres se agrupan gracias a *Insert Into Array*. Está configurado para que el *array* del número de mediciones sea la primera columna.

Finalmente, el conjunto de datos desemboca en un *Property Node*. Primero se ha insertado una tabla (*Table Control* 1, Zona F) en el panel de control. Después, se crea el *Property Node*. En este caso, *Value*, y de escritura.

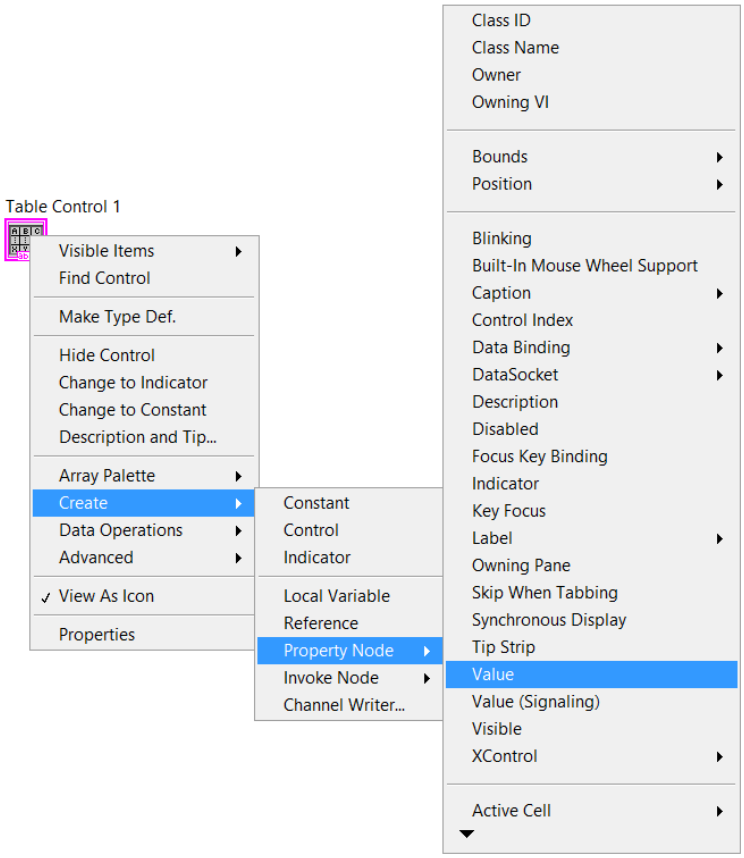


Figura 6.32. *Property Node* de *Table Control* 1.

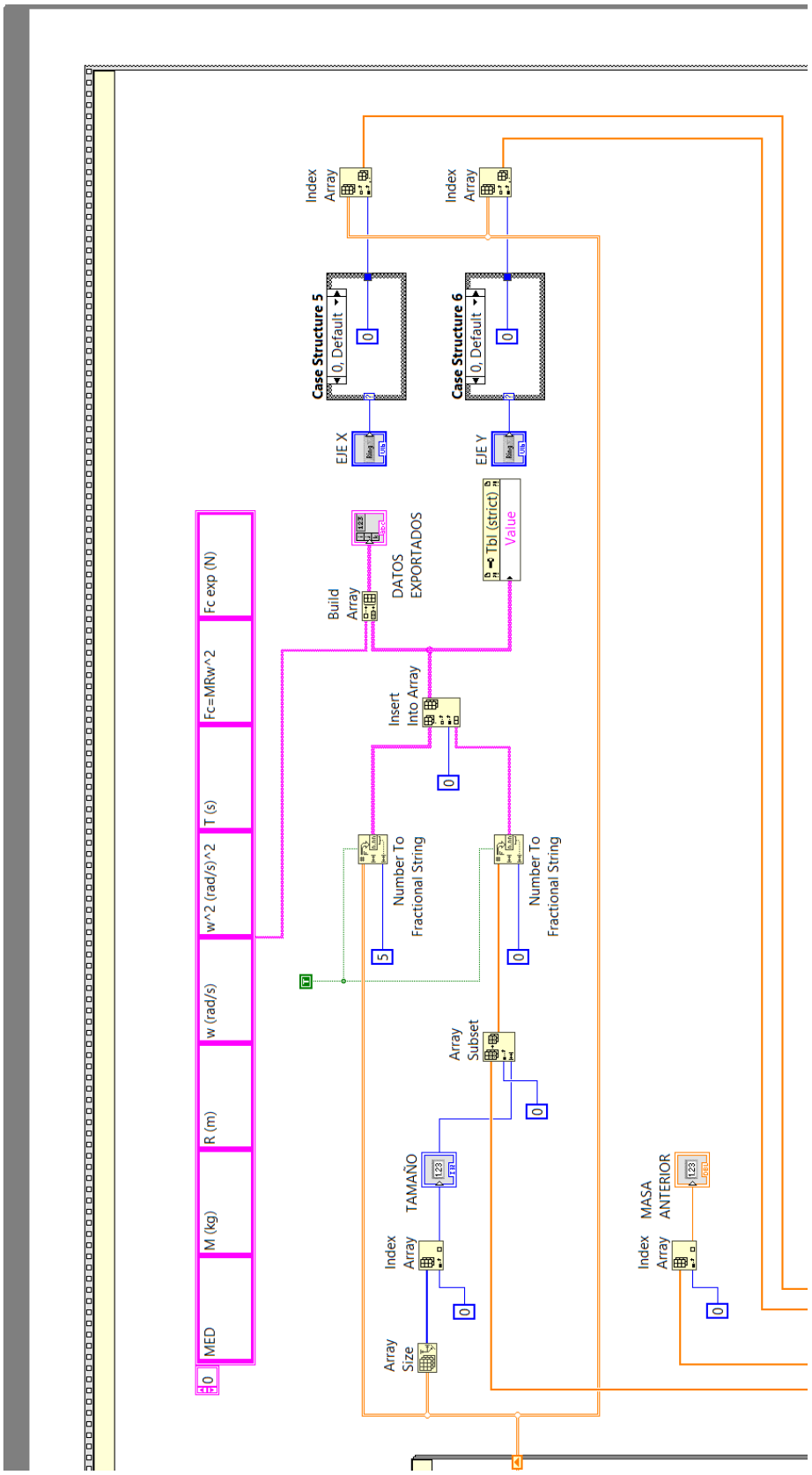


Figura 6.33. Zona E, presentación de los datos y estructuras *Cases* 5 y 6.

VISUALIZACIÓN DE LOS DATOS

El *array 2D* se bifurca para ir a dos *Index Array*. Cada uno de ellos sirve para escoger los datos del *array 2D* referidos a los ejes *X* e *Y* de la gráfica. Las entradas y salidas configuradas son:

N-dimension array: Es la entrada del bloque, que puede ser cualquier *array* de *N-dimension*. En este caso, es el *array 2D* de los datos.

Index 0... N–1 (col): Especifica el número de la posición del *array* de entrada. En este caso, se determina la columna.

La gráfica muestra los datos que el usuario elija. Así, se programan dos estructuras *Case* con cuatro alternativas, cada una de ellas correspondientes a los siguientes casos:

0- Masa.
1- Radio.
2- Omega al cuadrado.
3- Fuerza experimental.

Las dos estructuras (*Case* 5 y *Case* 6) permiten elegir los valores representados en la gráfica, tanto para el eje *X* como para el eje *Y*. Las opciones se eligen gracias a sendos controles (menú *Ring*).

Element o *Subarray*: Tiene los elementos seleccionados por el bloque y los coloca en un *array* de dimensión N.

Las dos salidas de los dos bloques *Index Array*, con los dos conjuntos de datos seleccionados, se llevan al bucle *For* 5, que trabaja para representar en una gráfica los datos escogidos por el usuario y mostrar la ecuación de ajuste por mínimos cuadrados y sus parámetros. Este bucle se repite una vez por cada medida, de modo que los datos son mostrados cada vez que termine el proceso.

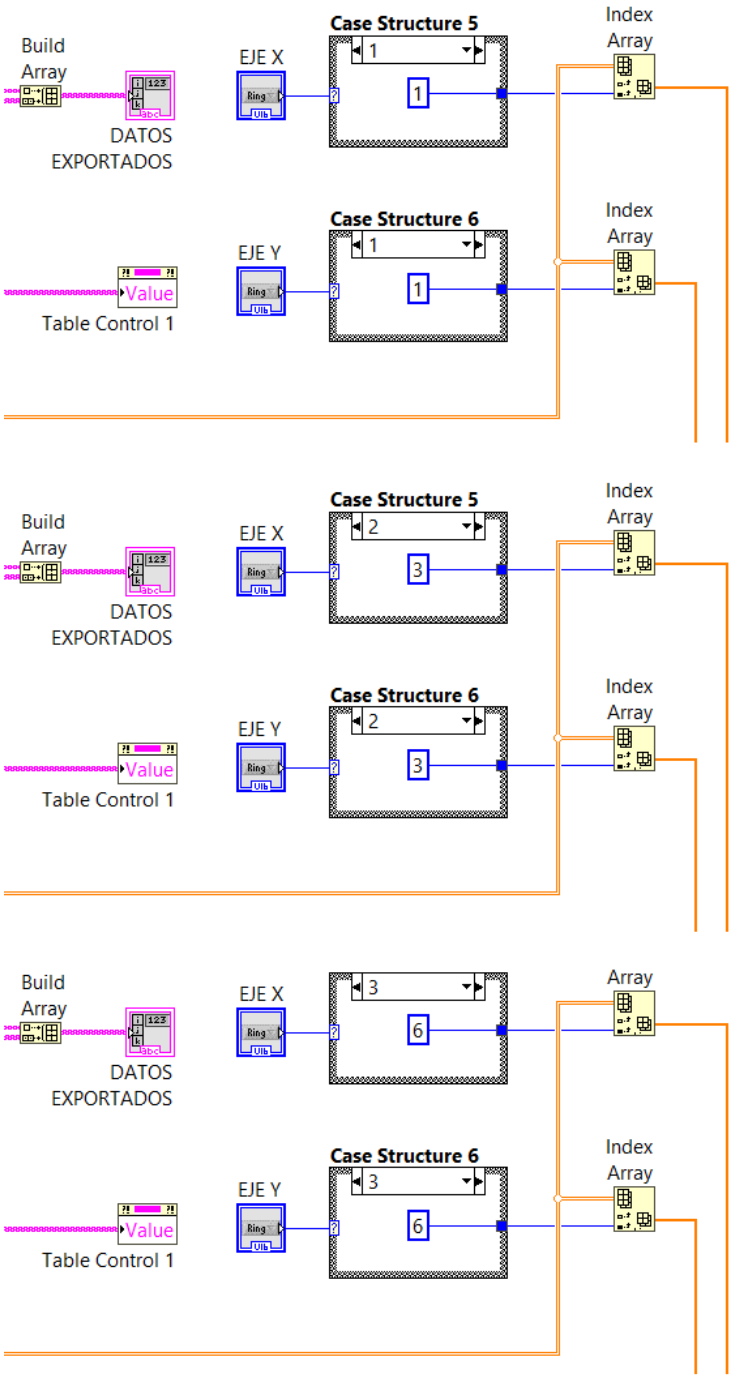


Figura 6.34. Zona E, opciones 1, 2 y 3 de las estructuras *Case* 5 y 6.

Los datos del eje *X* y del eje *Y* entran en la estructura para fluir a varios bloques. Primeramente, se dirigen a *Linear Fit.vi*. En este caso, calcula el ajuste por mínimos cuadrados de los datos. Se usan tres salidas.

Best Linear Fit: Son los valores de los datos del eje *Y* ajustados.

Slope: Es la pendiente del ajuste.

Intercept: Es el punto de intercepción en la ordenada del ajuste.

Estas dos salidas (*Slope* e *Intercept*) se unen en el bloque *Build Array*, para construir un *array* numérico. Constituye una de las entradas de *Regression Equation String.vi*. Este VI no se encuentra en ninguna paleta. Debe seleccionarse en uno de los ejemplo de LabVIEW. Para ello, se accede mediante el menú de *Help/Find Examples...* desde el panel frontal o el diagrama de bloques. En el cuadro *NI Examples Finder*, se selecciona en *Browse*, *Task*. Se cliquea en *Analysis, Signal Processing and Mathematics*, luego en *Curve Fitting* y se abre el archivo Regression Solver.vi. Por último se accede a su diagrama de bloques. Estas son sus entradas y salidas.

Fit tipe: Es el tipo de ajuste, que varía entre lineal, polinómico, exponencial y *power* (base-exponente). Se elige *Linear*.

Precision: El número de dígitos de la parte decimal, tanto para el punto de intercepción como para la pendiente. Se inserta 5.

Coefficients: Los datos para realizar la ecuación.

Formula: La salida, en formato *string* (“ A + B*x”, siendo A el punto de intercepción y B la pendiente).

La salida *Formula* es la entrada 1 del bloque *Concatenate Strings* (paleta *String*), que une en un único *string* varios *strings* o *arrays* de una dimensión. La entrada 0 es una constante *string* (“y = ”). La salida (*concatenated string,* “y = A + B*x”) se dirige a un *Build Array*, para mostrar la ecuación de ajuste en *Table Control* 2.

Para calcular el coeficiente R, se usa el VI *Goodness of Fit.vi* (paleta *Mathematics*). Calcula tres parámetros del ajuste de los datos: SSE (la suma del error al cuadrado), R al cuadrado y RMSE (la raíz del error al cuadrado). Estos parámetros describen el grado de ajuste de los datos. Las entradas y salidas utilizadas son:

Y: Los datos del *array* que corresponden al eje *Y*.

Best Fit: Los valores del eje *Y* ajustados.

R-square: El coeficiente R al cuadrado. Se le conecta al bloque *Square Root*, que devuelve la raíz cuadrada.

La grafica del panel de control muestra dos grupos de elementos. El primer grupo se corresponde con dos conjuntos de datos para los ejes *X* e *Y*. Muestran las medidas seleccionadas por el usuario mediante dos controles *RING*. Las medidas se representan mediante circunferencias de color rojo.

Para el segundo grupo se considera la ecuación del ajuste. Los números del eje *X* se obtienen de esta forma. Se usa el bloque *Array Max & Min* (paleta *Array*) para obtener los valores máximos y mínimos. Estos dos datos son las entradas para el VI *Eval y=f(x) Optimal Step.vi* (paleta *Mathematics*) que tiene también la entrada procedente del VI *Regression Equation String.vi*. Los valores del eje *Y* se obtienen de *Eval y=f(x) Optimal Step.vi*. La ecuación del ajuste se muestra en línea azul.

Los dos grupos pasan por sendos bloques *Bundle (paleta Cluster, Class & Variant)*, encargado de unir cada grupo de elementos en un *cluster*. Los dos *clusters* creados se fusionan en el bloque *Build Array*, cuya salida se conecta a una gráfica *XY Graph*.

Los errores de la pendiente y de la ordenada se calculan de acuerdo a las siguientes fórmulas:

$$error\ pendiente = \frac{pendiente}{R} \cdot \sqrt{\frac{1-R^2}{(número\,de\,datos\,de\,X)-2}} \qquad [44]$$

$$error\,ordenada = error\ pendiente \cdot \sqrt{\overline{X}^2 + \sigma^2} \qquad [45]$$

Siendo $\overline{X}$ la media aritmética y σ la varianza.

Para conseguir estos números, se realiza una serie de operaciones aritméticas con bloques de la paleta *Numeric*. Además, son necesarios otros elementos, como *Array Size* (paleta *Array*) y *Std Deviation and Variance.vi* y *Mean.vi* (paleta *Probability & Statistics*).

Tras el cálculo de la ordenada, el error de la ordenada, la pendiente, el error de la pendiente, el coeficiente de correlación y la ecuación de ajuste, estos datos se convierten en *strings* mediante bloques *Number To Exponential String* o *Number To Fractional String*. Varios bloques *Build Array* convierten en *array* los datos y los agrupan, para finalmente conectarlos a *Table Control* 2 y mostrarlos en el panel de control.

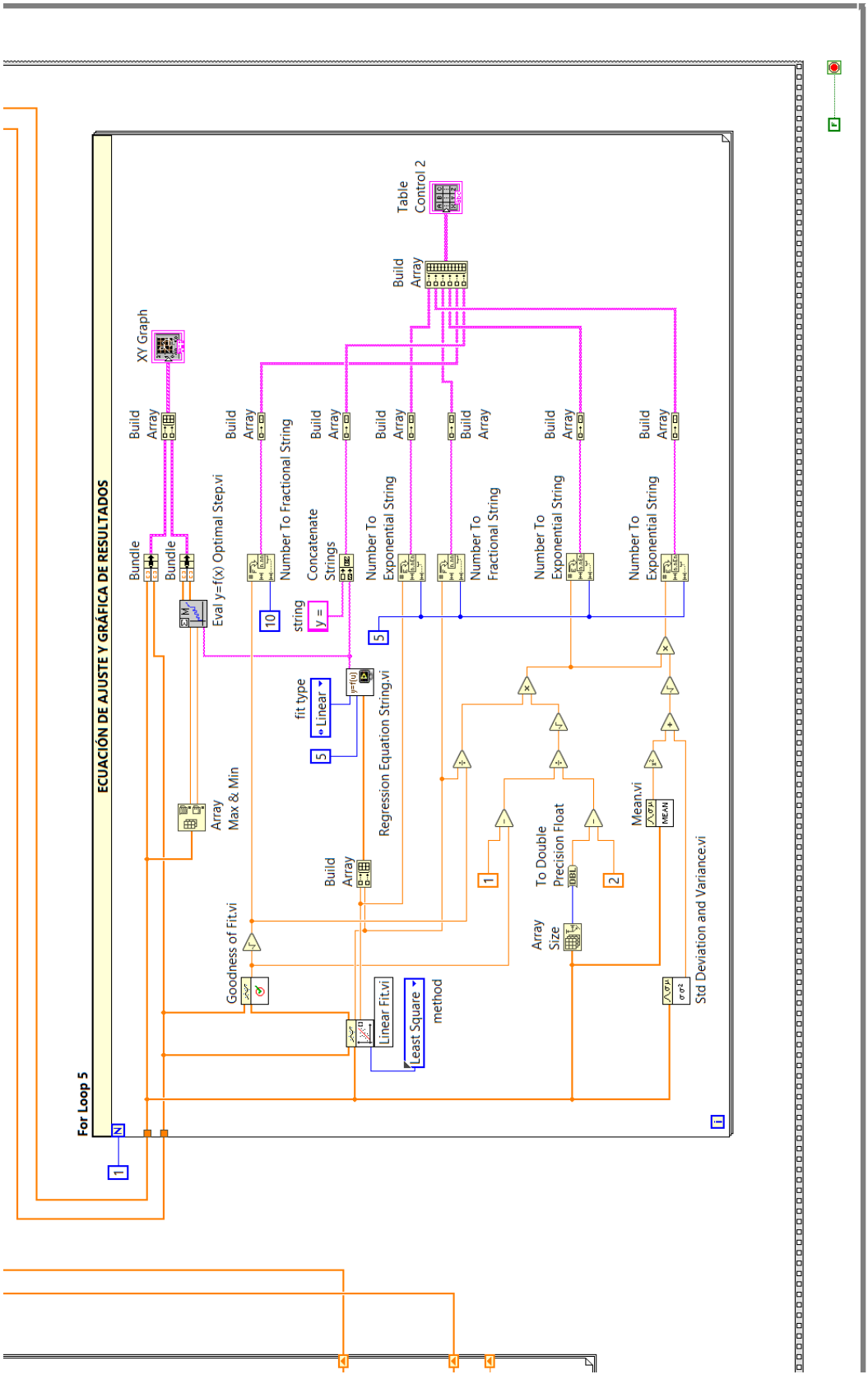


Figura 6.35. Zona E, bucle *For* 5.

6.2.5. Control de la fuente de alimentación

ZONA G

En primer lugar, es necesario disponer de las librerías de la fuente de alimentación. Para ello, en la ventana principal de LabVIEW, se selecciona *Help/Find Instruments Drivers…* La carpeta donde se instalarán las librerías se muestra en una ventana.

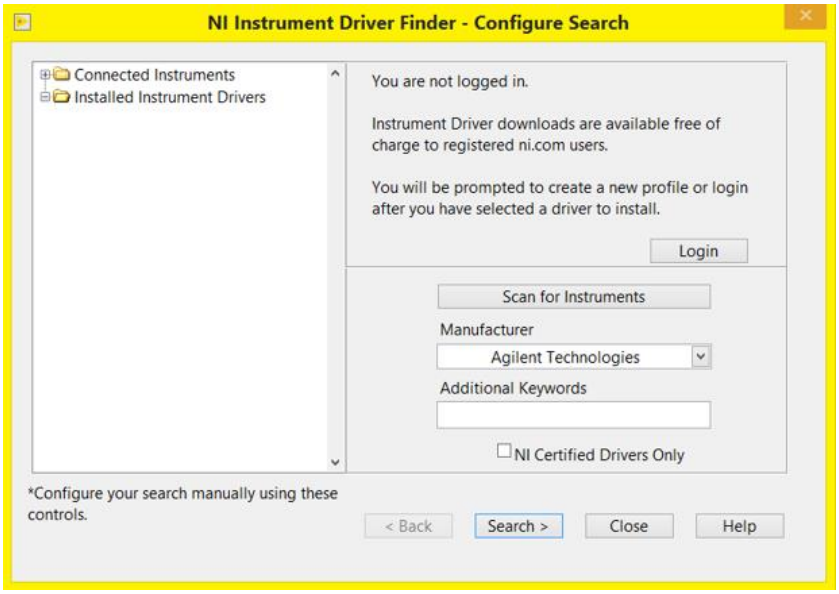


Figura 6.36. Ventana principal de *NI Instrument Driver Finder*.

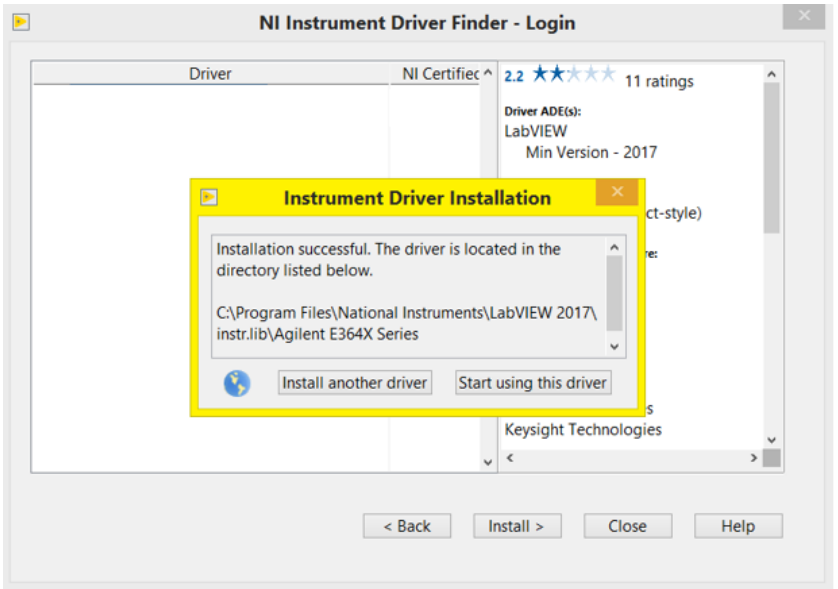


Figura 6.37. Directorio de la instalación de las librerías.

Después de situar las librerías para la fuente, se programa el código.

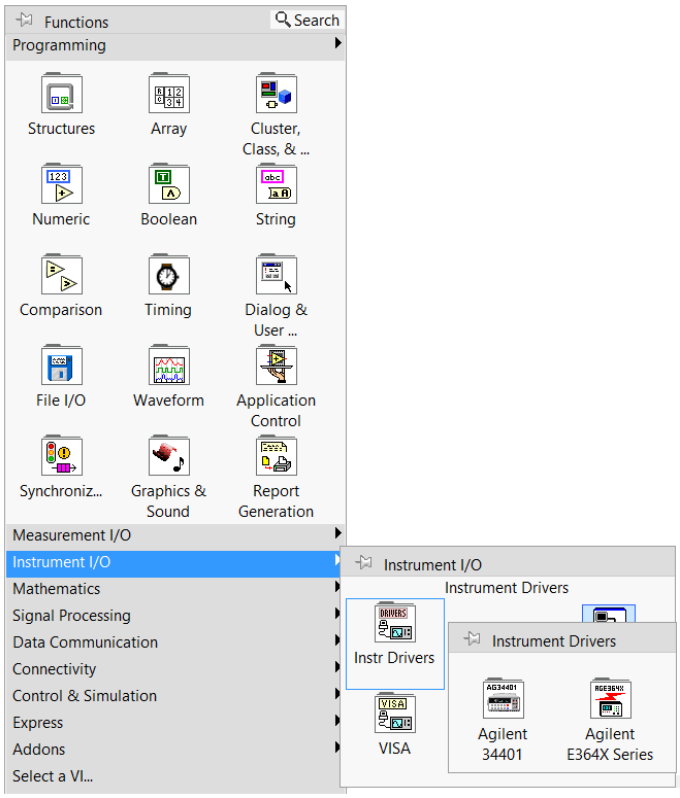


Figura 6.38. Paleta *Instrument Drivers*.

Los VI se encuentran en la paleta *Instrument Drivers* (Agilent E364X Series) o bien en la carpeta instr.lib. Hay que inicializar la fuente con *Initialize.vi*, cuya función es similar a *DAQmx Create Virtual Channel*, es decir, establece la comunicación. Su entrada es *VISA resource name*, que configura la referencia al instrumento y lo identifica. La fuente tiene como dirección GPIB0::5::INSTR. Las dos salidas usadas del VI (*VISA resource name out* y *Error out*) entran en el bucle *While* 10, cuyo primer elemento es *Output Select.vi*. Selecciona el canal de la fuente. Se utiliza la salida 1, por lo que se escoge *Output 1*.

Después, los VI *Configure Current Limit.vi* y *Configure OVP.vi* sirven para limitar la corriente y la tensión, respectivamente. Así, se asegura que el motor no reciba demasiada corriente o sobretensión para no dañarlo. La intensidad máxima es 1,5 amperios y la tensión, 15 voltios.

El siguiente VI es *Configure Voltage Level.vi*. Su entrada es una variable local de VOLTAJE, cuyo control numérico se sitúa en el mismo bucle. A continuación, *Configure Output Enabled.vi* configura si la señal que la fuente de alimentación produce en el canal seleccionado aparece en el conector de salida. Es decir, habilita o deshabilita dicha salida. Esta acción está

controlada por una condición *booleana*. Para que esté habilitada, se deben cumplir dos condiciones a la vez (función *and*): que MEDICIÓN permanezca en TRUE y que PARADA EMERGENCIA esté en OFF. MEDICIÓN es un indicador *booleano* y para conectarlo se crea una variable local. Además, dependiendo del flujo de la estructura *Flat Sequence* 1, cambia a TRUE o FALSE.

Finalmente, dos VI se encargan de mostrar cualquier error producido en el control y de liberar la conexión de la fuente (*Error Query (multiple).vi* y *Close.vi*). Si un error ocurre, *Simple Error Handler.vi* lo indica. Dentro del bucle la condición de parada es una constante FALSE para la terminal condicional S*top if True*. El bucle siempre se ejecuta.

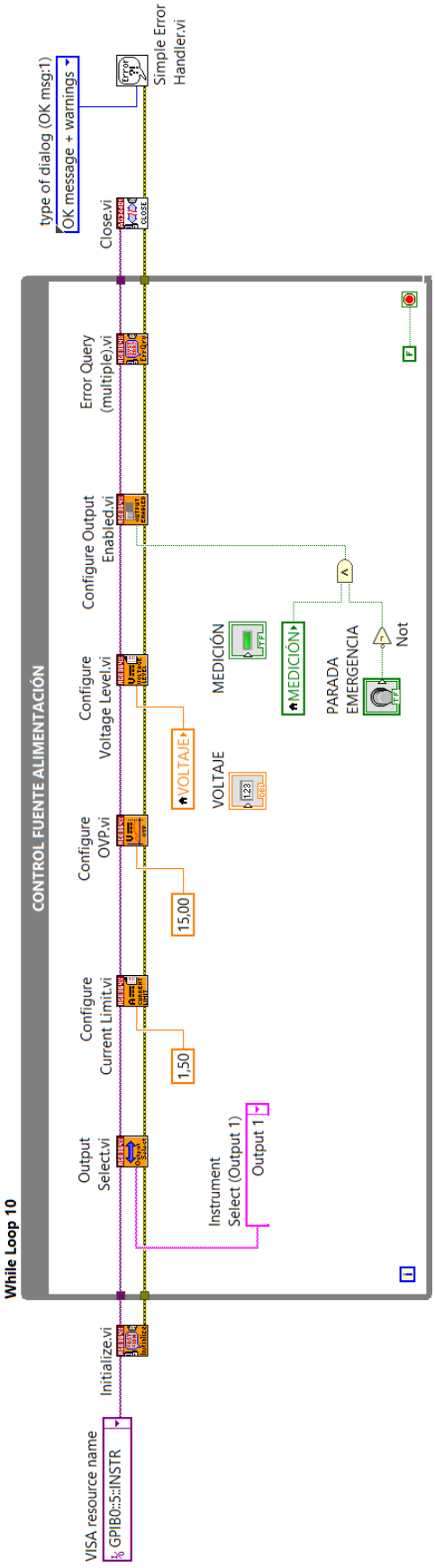


Figura 6.39. Zona G.

6.2.6. Exportación de los datos

ZONA H

La aplicación permite guardar los datos tras realizar cada medición. Para ello, se cablea desde la salida del bloque *Insert Into Array* hasta el bloque *Build Array*, que tiene dos entradas, la de los datos, y un *Array Constant* (paleta *Array*), formado por ocho *strings*. Cada uno de estos *strings* son los encabezados de las columnas de sus respectivos datos. La salida del *Build Array* se dirige a un indicador, un *array* (DATOS EXPORTADOS). Este *array* está conectado al bucle *While* 11 mediante una variable local.

El bucle *While* 11 está constituido por la estructura *Flat Sequence* 2, compuesta por dos *frames*. El *frame* 0 tiene un bucle *While*. Este bucle deja pasar los datos mientras el interruptor EXP DATOS esté en ON. Este control funciona como un interruptor con retorno. Cuando se pulsa, pasa a ON y automáticamente luego vuelve a OFF.

El *frame* 1 cuenta con un bucle *For*. Los datos provienen del anterior *frame* y entran a *Write Delimited Spreadsheet.vi* (paleta *File I/O*). Convierte los datos de un *array* de *strings*, o de otro tipo de elemento, a un *string* de texto y lo escribe a un archivo. Para ello, cuando se activa el bucle, aparece una ventana en la pantalla donde se elige el directorio, el nombre del archivo y la extensión (.xls, .txt, etc.). Así se exportan y guardan los datos. Este VI está conectado a *Simple Error Handler.vi*, que indica si se ha generado un error mediante un mensaje.

El bucle *For* 6 se ejecuta una vez. Sin embargo, la condición de paro del bucle *While* 11 es FALSE, por lo que cada vez que se realicen mediciones, el usuario tiene la posibilidad de generar un archivo para guardarlos.

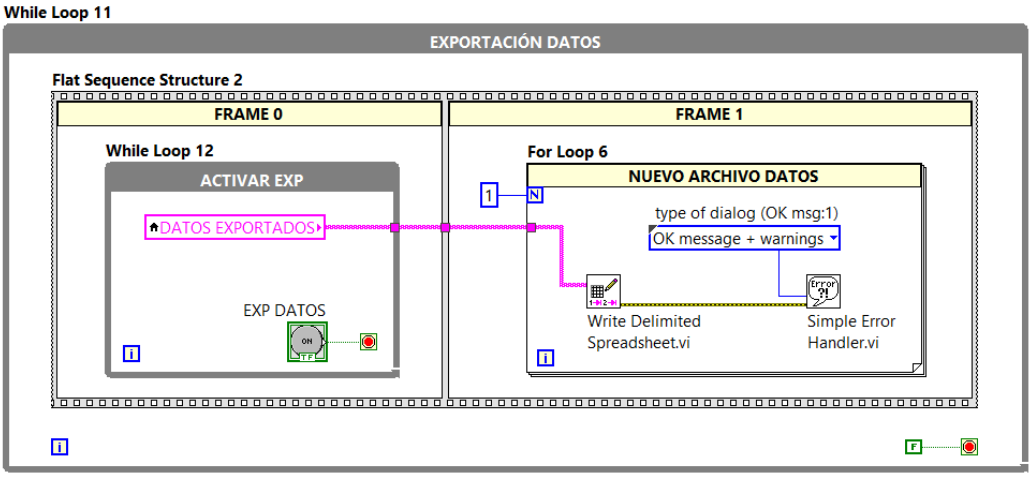


Figura 6.40. Zona H.

6.3. Panel frontal

El panel frontal o panel de control sirve para manejar el programa y ver los datos de los diferentes parámetros.

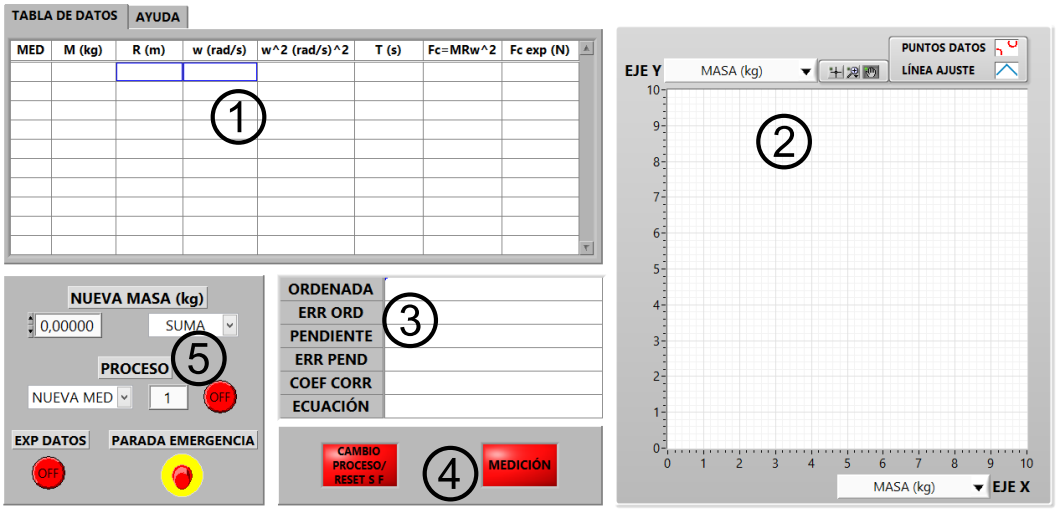


Figura 6.41. Panel de control.

Se compone de cinco zonas.

- Zona ①. *Tabla de control de dos pestañas*.

  El elemento *Tab Control* (paleta *Containers*) contiene páginas o pestañas. Cada una de ellas puede albergar elementos de control o indicadores que no pueden compartir páginas. Cuando se está en una pestaña, se ejecuta su contenido. Así pues, *Tab Control* tiene muchas similitudes con una estructura *Case*, cuyo selector es la parte superior de cada página.

  La primera pestaña posee una tabla de ocho columnas, cada una de ellas con los datos a estudio, y once filas. La primera fila es el encabezado. Si se realizan más de diez mediciones, aumentan las filas. Una barra de desplazamiento vertical permite moverse por la tabla. Además, las celdas seleccionadas del Radio y la Omega son de escritura de los datos.

  La segunda cuenta con un breve manual de ayuda con instrucciones. El usuario así conoce cómo proceder en las medidas y cómo actuar ante cualquier posible fallo.

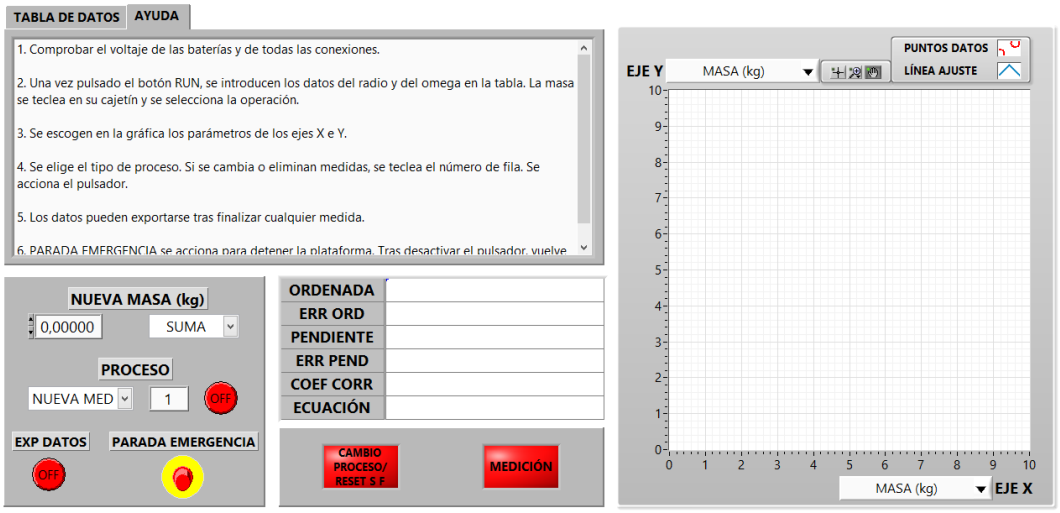


Figura 6.42. Panel de control con la pestaña AYUDA.

- Zona ②. *Gráfica*.

  En los ejes *X* e *Y*, el usuario selecciona qué datos desea que la gráfica (indicador *XY Graph*, paleta *Graph*) muestre, gracias a dos menús *Ring*. Los datos a escoger son los mimos tanto para *X* como para *Y*:

  - MASA (kg).
  - RADIO (m).
  - OMEGA^2 (rad/s)^2.
  - FUERZA EXPER (N).

  Estos datos son puntos de color rojo en la gráfica. Asimismo, también se grafica la línea que corresponde a la ecuación de ajuste de mínimos cuadrados, de color azul. Tras cada proceso, se actualiza.

  La escala de ambos ejes (abscisas y ordenadas) se escalan de forma automática, según se registren datos. Es decir, si en el eje *X* los datos varían de 0 a 10, dicho eje ajusta su escala desde 0 hasta 10.

  Además, gracias a una paleta (*Graph Palette*), el usuario puede profundizar en el estudio de los datos en la gráfica, como hacer selecciones de una determinada área, ampliar una zona, etc.

- Zona ③. *Tabla de datos de la gráfica*.

  En esta tabla se tabulan diferentes términos y coeficientes de la ecuación de ajuste por mínimos cuadrados (línea azul de la gráfica). Los datos se renuevan después del proceso. Así, se muestran:

- ORDENADA.
- ERROR ORDENADA.
- PENDIENTE.
- ERROR PENDIENTE.
- COEFICIENTE DE CORRELACIÓN.
- ECUACION DE AJUSTE.

- Zona ④. *Indicadores*.

  Los indicadores advierten de dos eventos. El primer indicador luce en verde (ON) en los momentos en los que se debe seleccionar una opción del menú PROCESO y de pulsar el *reset* del sensor de fuerza. Si luce en rojo (OFF), no se deben realizar estas acciones. El segundo, MEDICIÓN, informa sobre cuándo se adquieren datos desde los sensores. Ambos son de tipo *Square LED* (paleta *Boolean*).

- Zona ⑤. *Controles*.
  - NUEVA MASA (kg).
    Se introduce el valor añadido o extraído a la masa total que aparece en la tabla, o bien que el valor aparezca en la celda, mediante un selector. Este selector permite sumar, restar o introducir dicho valor a la celda de la tabla. Cuenta con cinco dígitos decimales. El rango comprende desde los 0 hasta los 0,20000 kg. El incremento y el decremento son de 0,00001.

  - PROCESO.
    Un menú *Ring* selecciona la acción de la medición (nueva medida, cambio medida, suprimir medida o suprimir todas las medidas de la tabla). Al lado de este menú, un controlador numérico indica la fila de la medida afectada por el proceso. La nueva medida se ejecuta tras la última. Si se opta por el cambio o eliminación, el usuario debe introducir el número de fila, menor o igual que el número de filas totales de la tabla. Si se elige suprimir tabla, aparece 0.

    En la parte derecha aparece un controlador (*Labeled Round Button*, paleta *Boolean*). Sirve para iniciar el proceso. Es un interruptor con retorno. Se pulsa cuando el indicador CAMBIO PROCESO/RESET S F está en ON y MEDICIÓN, en OFF.

- EXP DATOS.
  Este controlador se puede accionar tras una medición. Es decir, si CAMBIO PROCESO/RESET S F está en ON y MEDICIÓN, en OFF. También está configurado como interruptor con retorno.

- PARADA EMERGENCIA.
  En la parte derecha e inferior está colocado otro controlador del mismo tipo que los anteriores. La diferencia radica en que cuando se pulsa, no retorna. Cuando se activa (ON), la plataforma se para y no se toman datos de los sensores. En OFF, el flujo del programa continúa. No es posible cambiar el tipo de proceso tras su activación.

6.4. Funcionamiento de la aplicación

En primer lugar hay que comprobar el voltaje de las baterías. Después, todas las conexiones de los elementos (sensores, tarjeta, fuente de alimentación) y las eléctricas. Tienen que estar correctas. Tras esto, se abre el archivo de la aplicación, con extensión .vi.

Se cliquea con el ratón en *Run*.

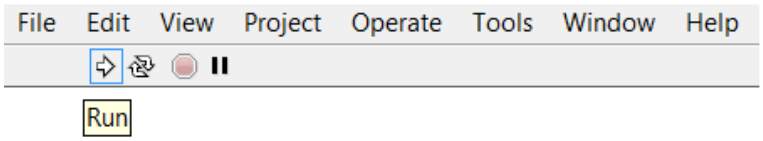


Figura 6.43. Botón *Run*.

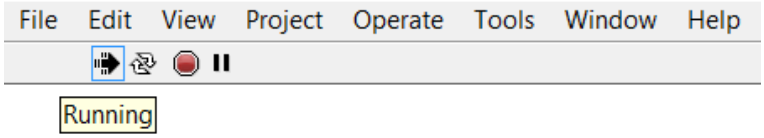


Figura 6.44. Indicador del estado *Running*.

La aplicación está lista, y pasa a estado *Running*. Si se quiere finalizar el programa, se pulsa *Abort Execution*.

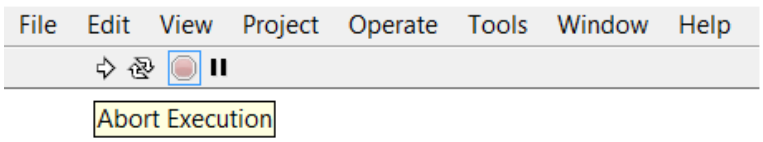


Figura 6.45. Botón *Abort Execution*.

6.4.1. Fuerza centrípeta frente masa

La masa varía en esta experimentación, y el radio y la omega son constantes. El *portamasas* fijo debe de contener la misma masa que el *portamasas* del extremo de la plataforma. En modo *Running*, se introducen los datos. Como ejemplo, la primera masa es de 0,01004 kg. Se suma y aparece la misma cifra, puesto que es la medida número uno, y no existen acumuladas. En la tabla se inserta 0,20000 metros y 7,00000 rad/s. En PROCESO, se selecciona NUEVA MED. Para el eje *Y* de la gráfica se escoge FUERZA EXPER (N), mientras que en el eje *X*, MASA (kg).

El indicador CAMBIO PROCESO/RESET S F está en verde. Se pulsa el botón del PROCESO y la plataforma gira. Cuando la velocidad angular llega a 7 rad/s, se para, la tabla de datos muestra los resultados y la gráfica un punto (figura 6.47).

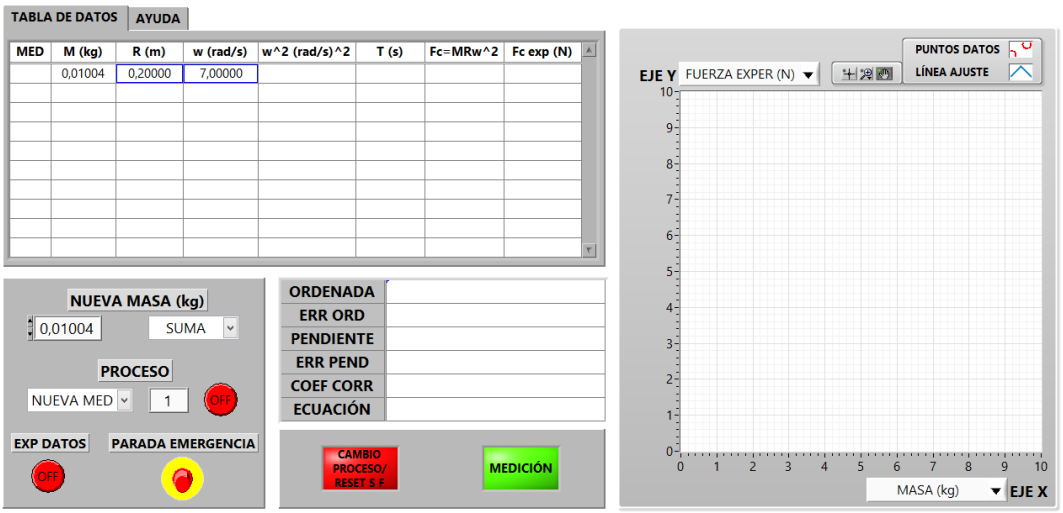


Figura 6.46. Programa adquiriendo datos de los sensores.

Después, los datos de la medida 2. Se teclea la masa a añadir, como ejemplo, 0,00998 kg, con operación suma. El radio y la omega son los mismos.

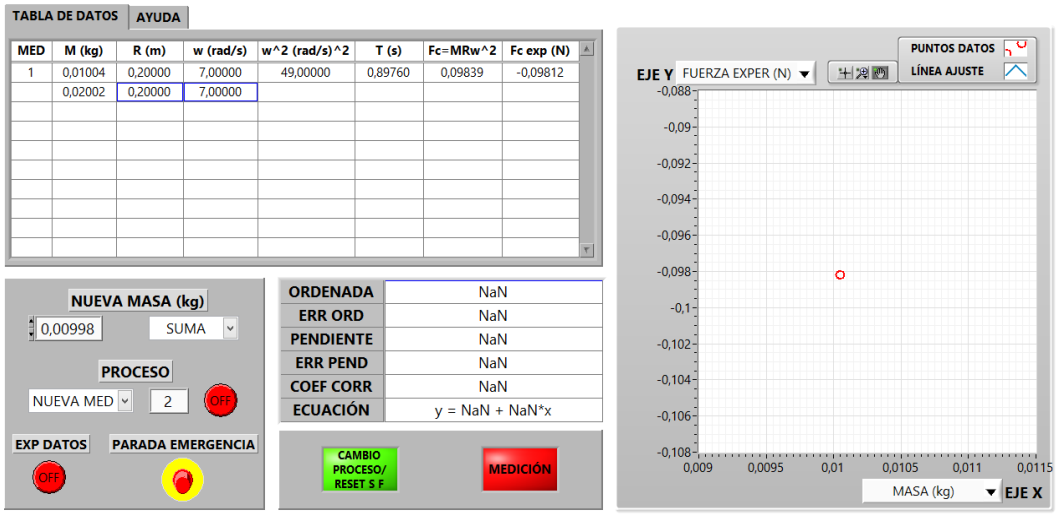


Figura 6.47. Datos para la medida 2.

Se observa el resultado de las dos medidas en la figura 6.48. Para la medida 3, se repite la misma operación. Y cuando muestre por pantalla los datos, se introducen los datos para la medida 4.

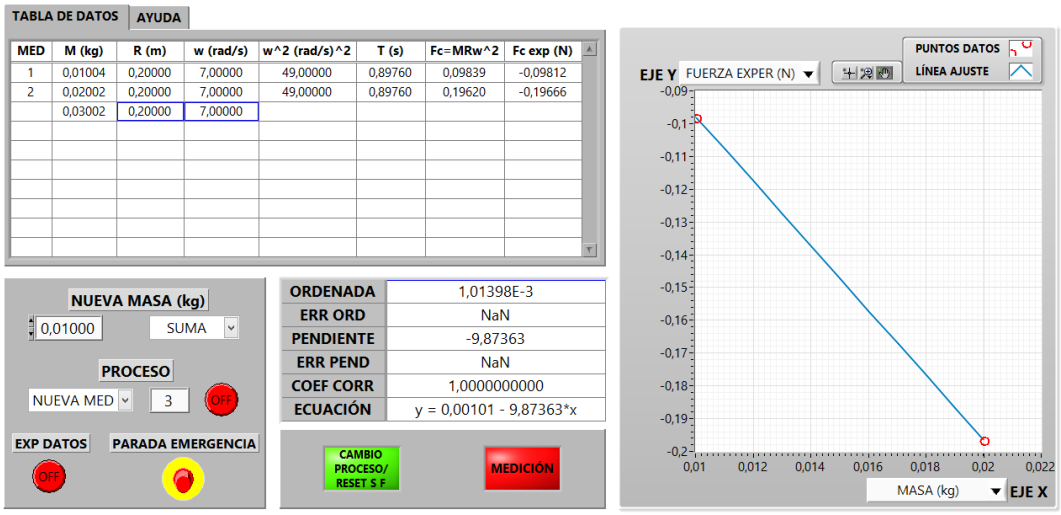


Figura 6.48. Aplicación con dos medidas y datos para la tercera medida.

Con siete medidas, la pantalla ofrece este aspecto:

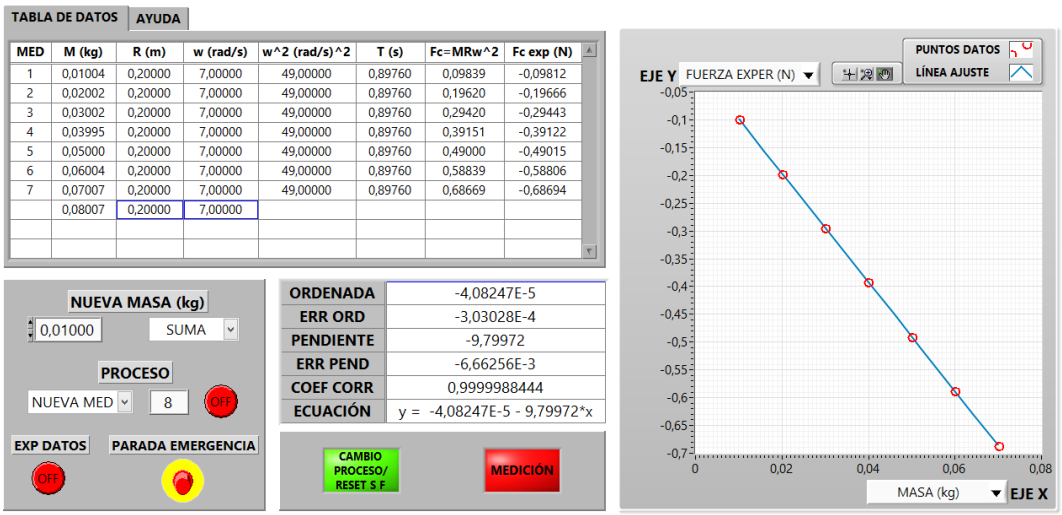


Figura 6.49. Tabla con siete medidas realizadas.

Se introducen las medidas que quiera el usuario. Así, para diez medidas, la aplicación muestra las siguientes tablas y gráfica.

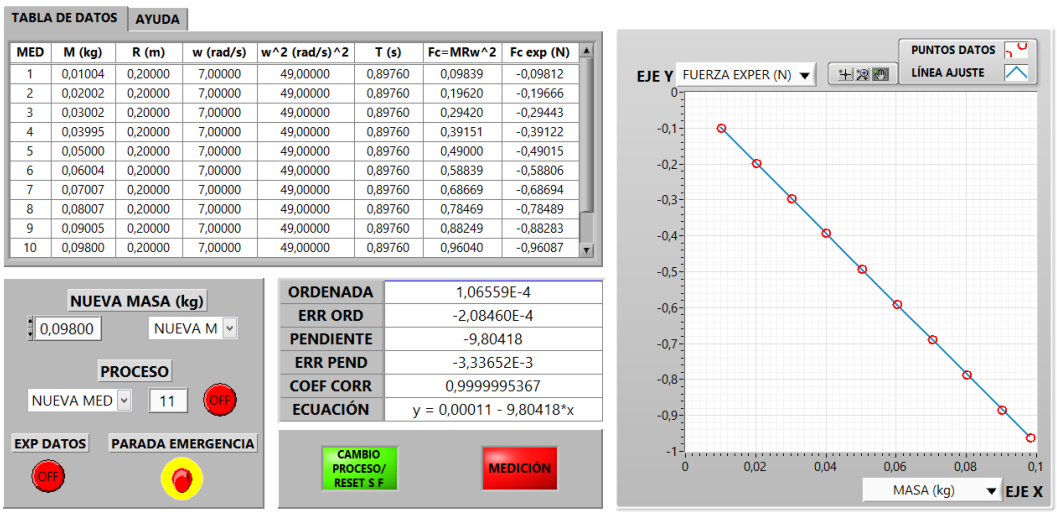


Figura 6.50. Diez mediciones.

Si se elimina una medida, por ejemplo la seis (FC=MRw^2 con 0,58839), se produce lo siguiente:

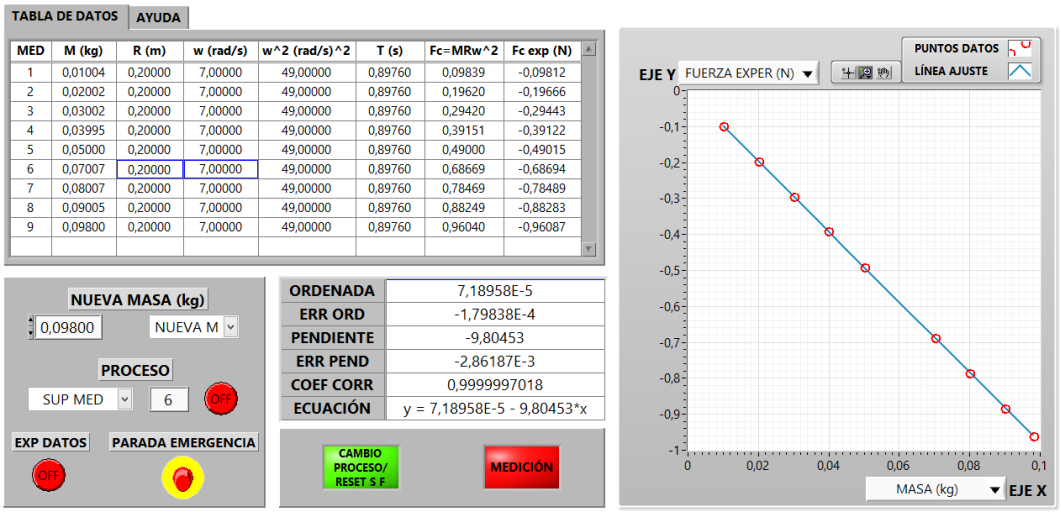


Figura 6.51. Aplicación tras la eliminación de la sexta medida.

Si se pulsa EXP DATOS, se abre una ventana para guardar los datos de las distintas medidas.

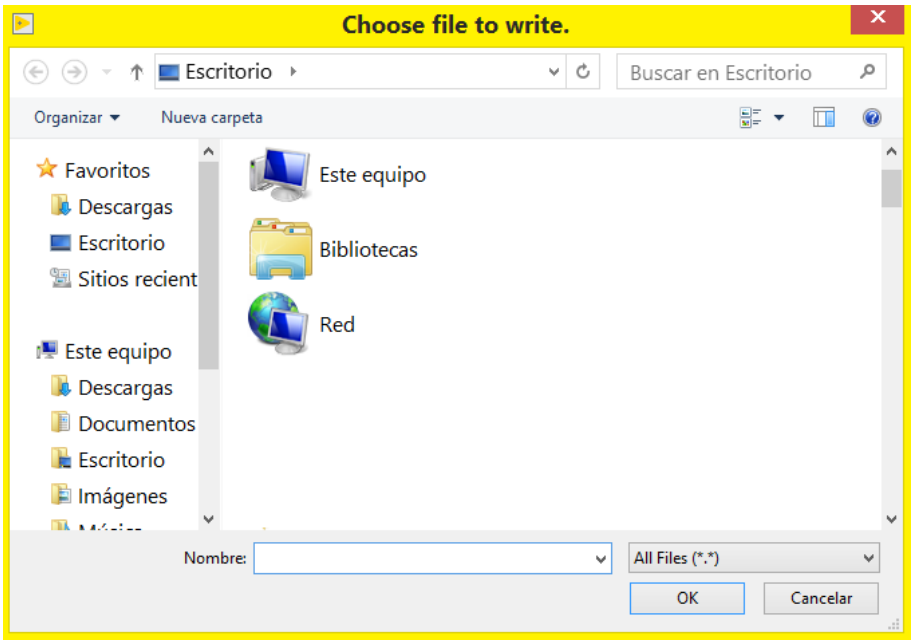


Figura 6.52. Ventana para exportar medidas.

6.4.2. Fuerza centrípeta frente radio

El radio es una variable, mientras que la masa y la omega son constantes. Tras activar la aplicación (modo *Running*), se introducen los datos, se selecciona el proceso y las magnitudes de los ejes *X* e *Y*. El radio se ajusta bajando o subiendo la barra horizontal, donde está sujeto el sensor de fuerza.

Se comprueba la horizontalidad con el nivel. Se comienza con un radio de 0,06000 metros, y se incrementa este valor en 0,01500 metros en las siguientes medidas.

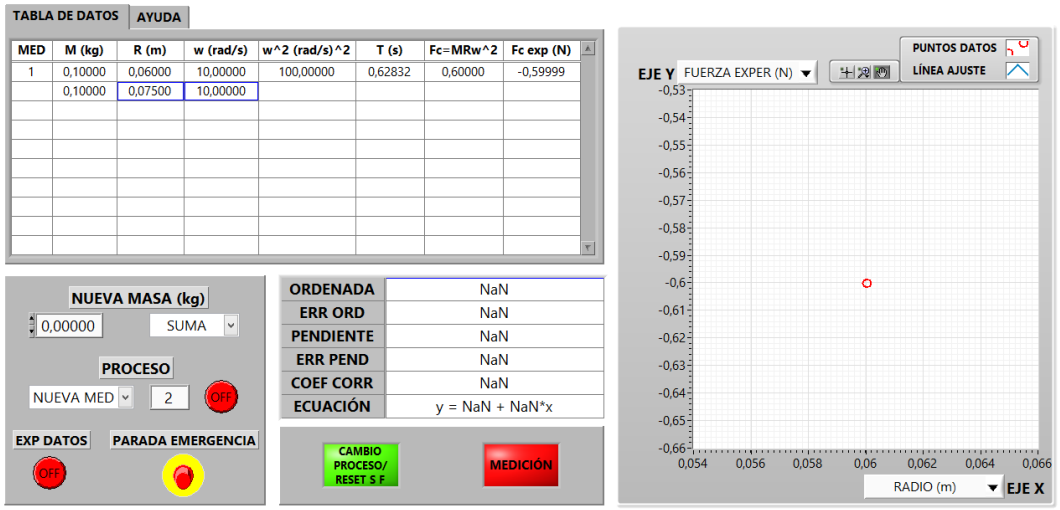


Figura 6.53. Primera medida y datos para la segunda.

Con dos medidas, se observa lo siguiente:

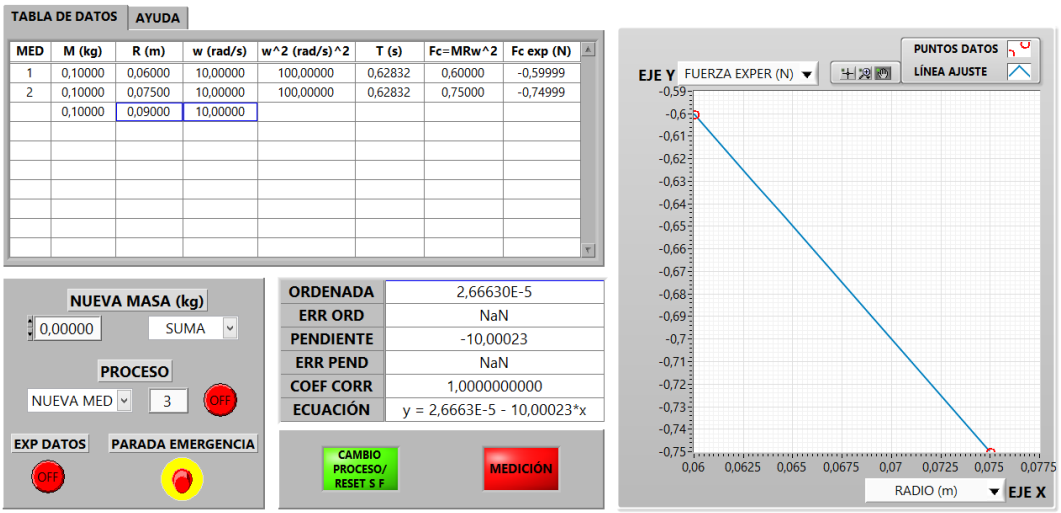


Figura 6.54. Dos medidas.

En las medidas número 3 y número 7:

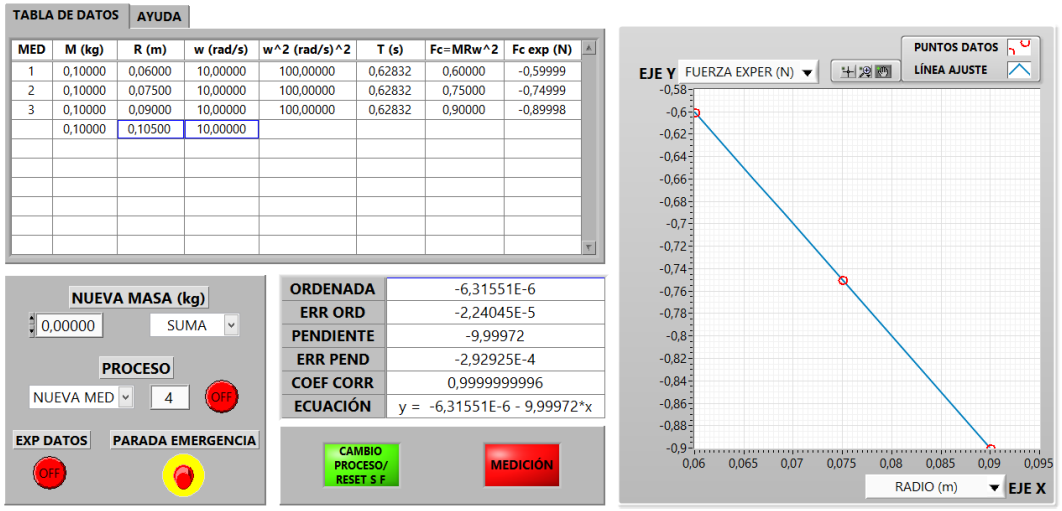


Figura 6.55. Tabla con tres medidas.

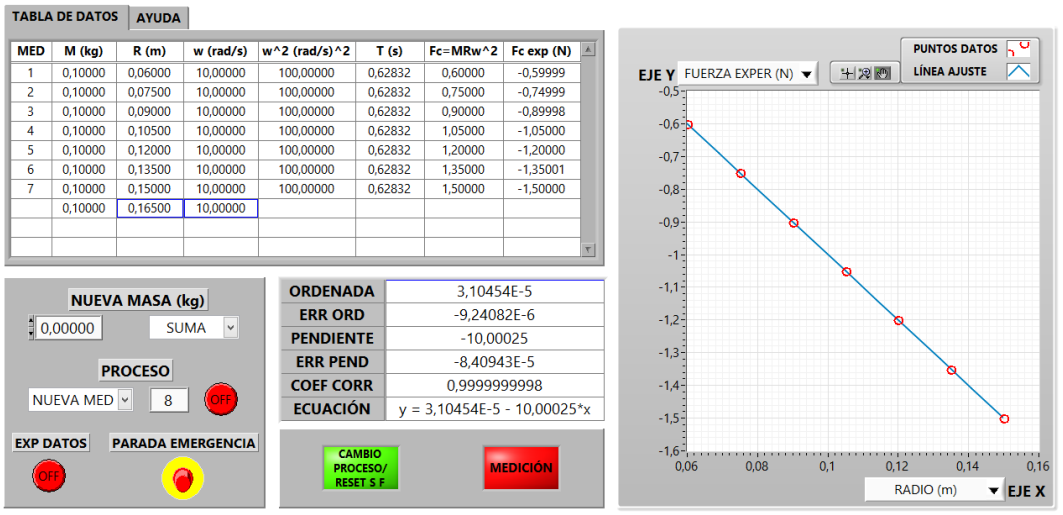


Figura 6.56. Siete medidas.

Con diez medidas, este es el resultado.

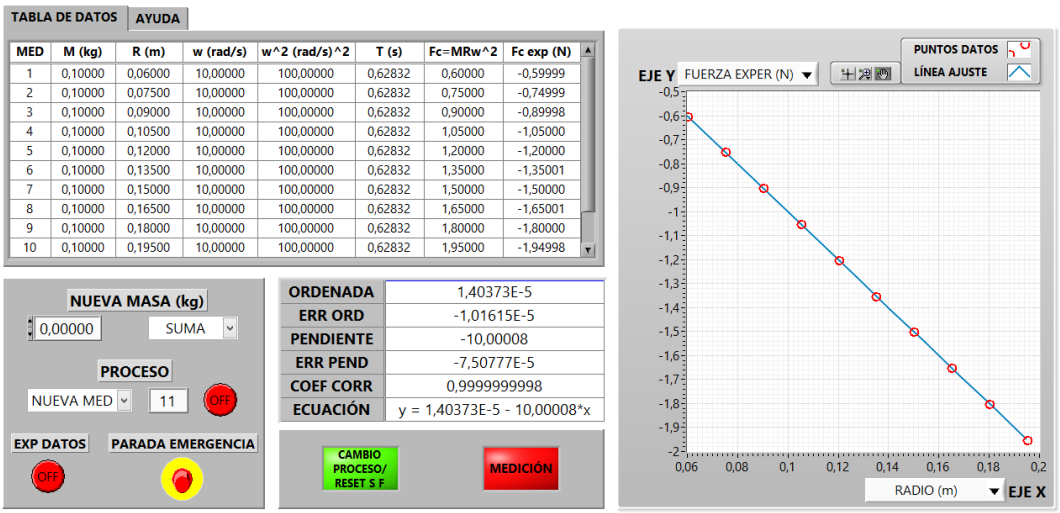


Figura 6.57. Aplicación con diez medidas.

Si se realiza un *zoom* en la medida número 3, la gráfica muestra el siguiente aspecto.

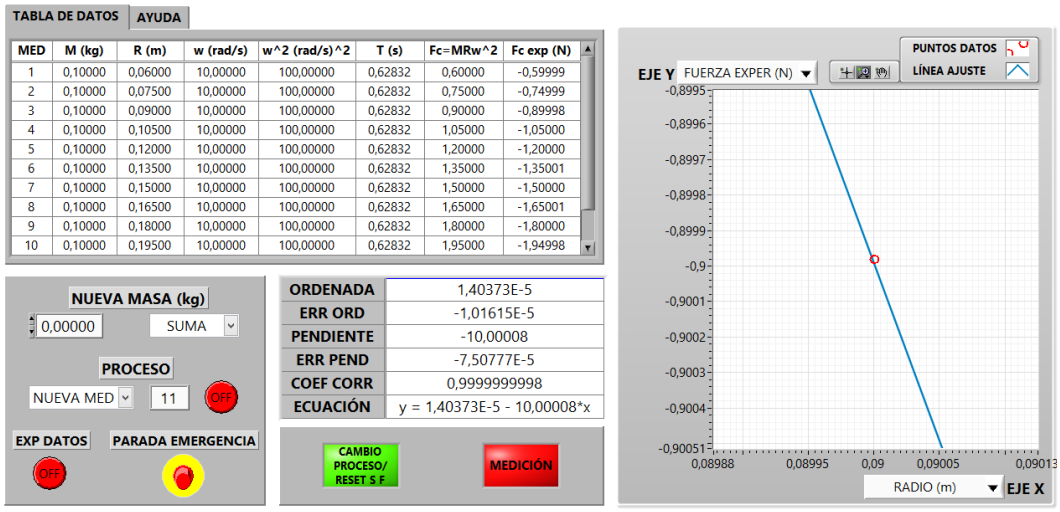


Figura 6.58. *Zoom* en la tercera medida.

6.4.3. Fuerza centrípeta frente omega al cuadrado

En esta experimentación se varía la velocidad. El usuario introduce una omega y la plataforma gira hasta alcanzar ese valor. La masa y el radio permanecen constantes, mientras que la velocidad difiere en cada medida. Los primeros datos se insertan tras activar la aplicación (*Running*). La masa es de 0,05000 kg, y el radio mide 0,20000 metros. Se seleccionan las magnitudes en la gráfica y el tipo de proceso. Se empieza con una omega de 5 rad/s. Se aumenta en 1 rad/s para cada medida. Así,

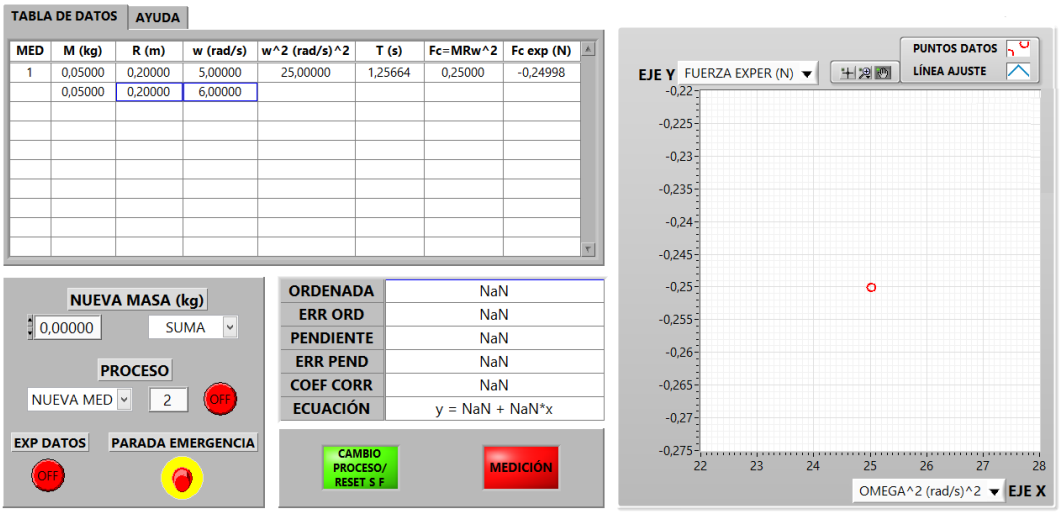


Figura 6.59. Primera medida y datos para la segunda.

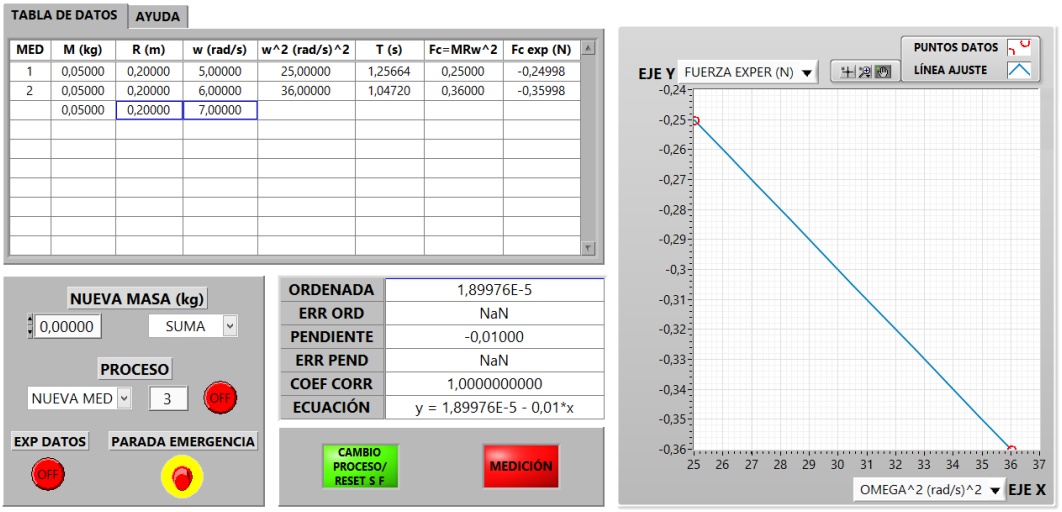


Figura 6.60. Dos primeras mediciones efectuadas.

Con tres y siete medidas se muestra en pantalla las siguientes figuras:

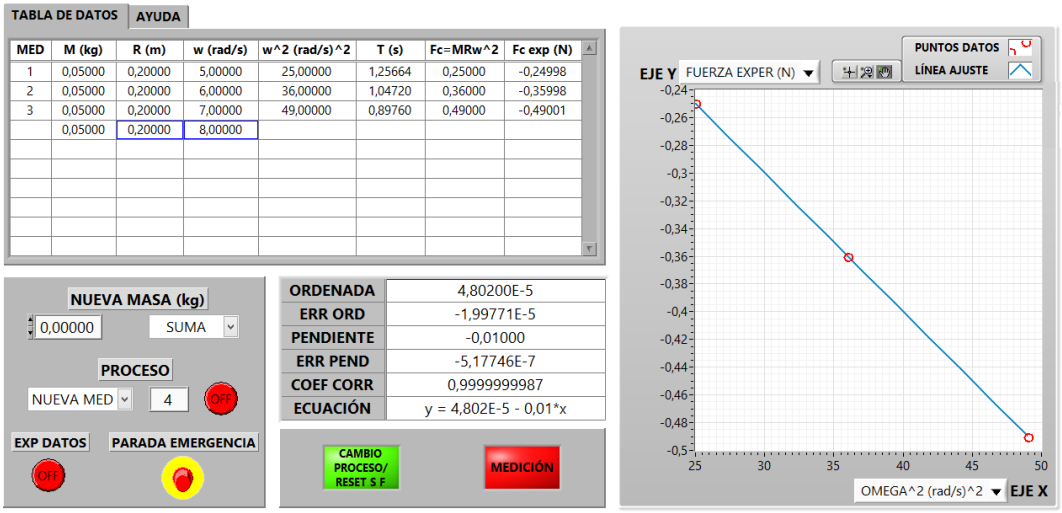


Figura 6.61. Aplicación con tres medidas.

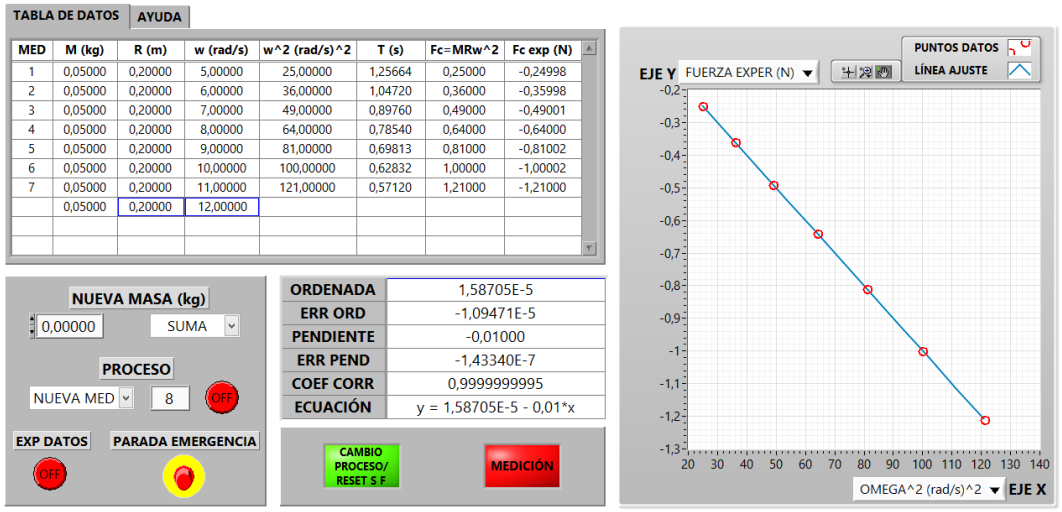


Figura 6.62. Aplicación con siete medidas.

Con diez medidas, el resultado es el siguiente:

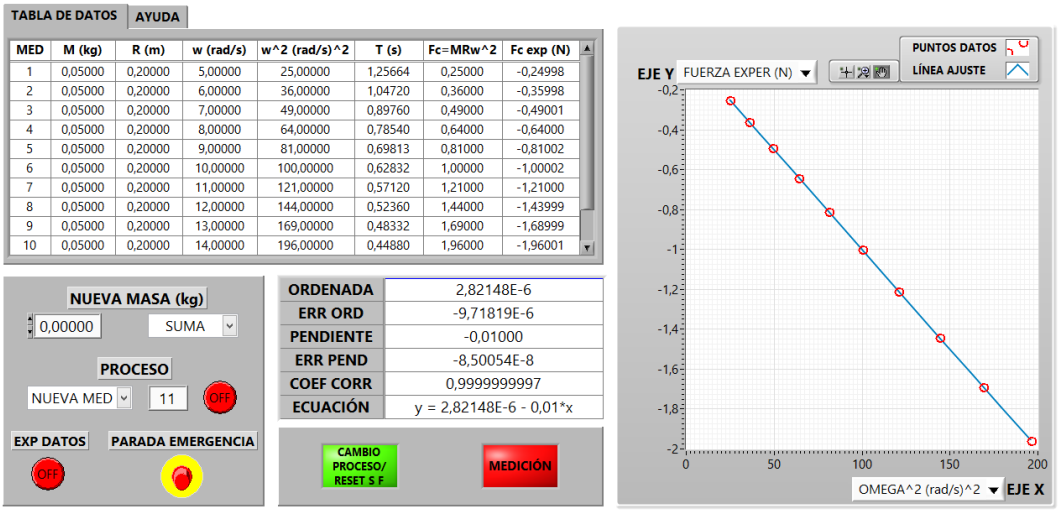


Figura 6.63. Diez mediciones realizadas.

Si se desea cambiar una medida, se selecciona CAMB MED y, a continuación, el número de fila que se desea eliminar para introducir la nueva medida. En este caso, se cambia la fila sexta. La masa es la misma, y se teclea en la tabla el radio y la omega (0,20000 metros y 10,00000 rad/s).

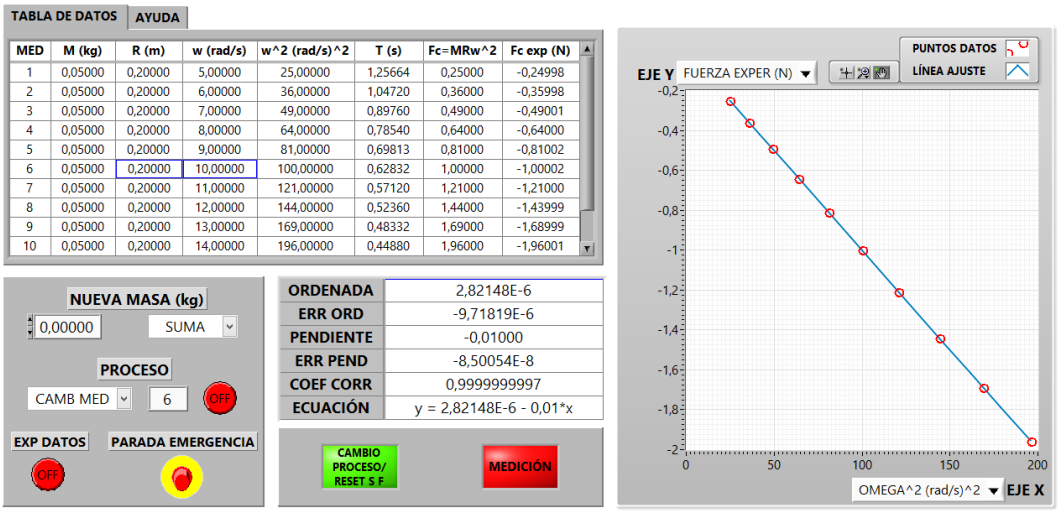


Figura 6.64. Nuevos datos para la sexta medida.

Tras activar el proceso, las tablas y la gráfica se actualizan. Se presenta esta pantalla:

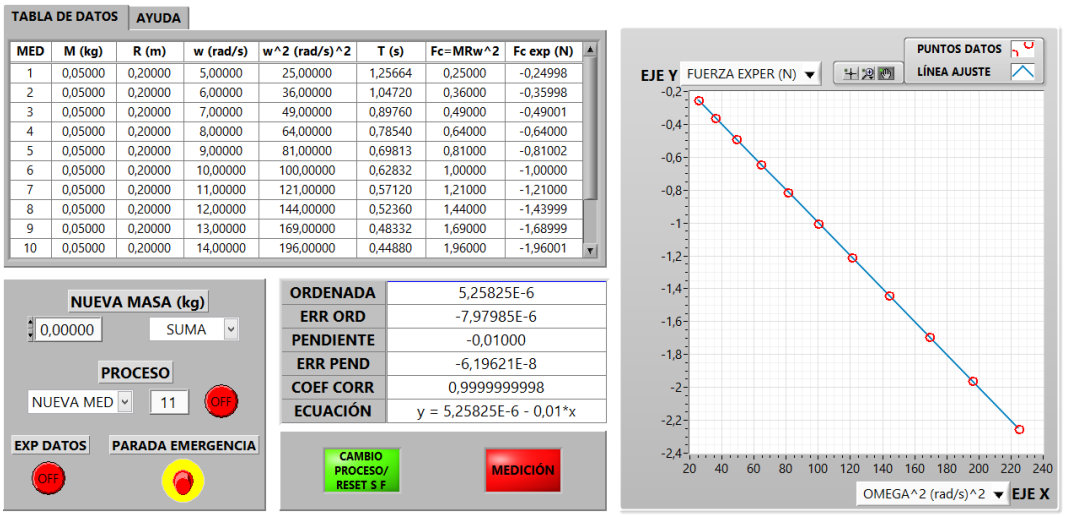


Figura 6.65. Aplicación tras el cambio de la sexta medida.

CAPÍTULO 7.
CONCLUSIONES

- Se ha comprobado que dos sensores (uno analógico y otro digital) funcionan a la vez correctamente. El uso de estos dos elementos es indispensable para la elaboración de este Proyecto. Sin embargo, el sensor analógico de fuerza no trabaja con tanta estabilidad como el sensor digital de movimiento. Para la solución de este punto, se elaboraron subdiagramas en LabVIEW que permiten alcanzar la precisión con el objeto de crear una aplicación robusta. Además, se ha aprendido a cómo utilizar un sensor para la toma de medidas, una función clave en el mundo de la automática.

- También se ha aprendido el funcionamiento de una tarjeta de adquisición de datos. Este dispositivo juega otro papel importante en el paso de este Proyecto. Así, se encarga de convertir las señales eléctricas procedentes de los sensores en lenguaje de bits para la computadora. Se ha estudiado la tarjeta para saber la alimentación y realizar las conexiones. Para ello, se ha usado el manual, que explica sus características y datos técnicos.

- El uso del software LabVIEW ha permitido el desarrollo de este TFG. Se ha conocido el enorme potencial de este software, usado en importantes empresas de electrónica, computación o automoción. Dispone de una amplia gama de herramientas para la consecución de diversos objetivos, como adquisición y tratamiento de datos, control de procesos, desarrollo de interfaces gráficas de usuario, etc. En este caso, se ha diseñado un código, donde parte de él se centra en la adquisición de los datos suministrados por los dos sensores. Se ha trabajado con las paletas y bloques para conseguir la comunicación y se han analizado las diversas maneras de establecer el contacto y la adquisición, comprendiendo cómo efectuar este proceso, muy importante para la toma de datos.

- La adquisición de datos es una parte de este Proyecto, pero también constituye un factor decisivo el tratamiento de las medidas, así como su exposición. Estas tareas se han configurado en LabVIEW para que después de cada medición se muestren en pantalla y se acumulen las medidas anteriores. La programación del código se ha elaborado para que sea fácil su comprensión, tanto el diagrama de bloques como el panel frontal. Se ha usado el menor número de estructuras y bloques

para conseguir un código limpio y claro. Además, se ha aprendido a manejar este software y se han asimilado importantes conceptos de este lenguaje gráfico, que van desde el funcionamiento de una estructura básica hasta el uso de bloques para exportar datos a diversos formatos, como .xls.

- Asimismo, se ha diseñado el panel frontal o de control para que al usuario le resulte fácil trabajar con la aplicación. Se han colocado los botones necesarios con el propósito de que el programa funcione. También se ha tenido en cuenta las posibles situaciones en las que se pueda enfrentar el usuario. Así, el programa dispone de un botón de parada de emergencia para que la plataforma pare su giro ante una circunstancia que pueda dañar al motor u otra situación peligrosa.

  Este panel se ha elaborado de forma sencilla para resumir el funcionamiento. También se considera la muestra de datos: dos tablas, una para las medidas y otra para los parámetros de la ecuación de ajuste. Estas tablas se actualizan cada vez que realice un proceso. La gráfica muestra los puntos y la línea de ajuste con nitidez, y de igual modo se renueva en cada proceso. La solución que se ha adoptado se ajusta a los requerimientos del Proyecto. Así, la aplicación en LabVIEW está hecha a medida. Es una de las ventajas de este software. Permite diseñar códigos y paneles de control según las necesidades del usuario, es decir, un programa personalizado de acuerdo a las especificaciones requeridas.

- Por otra parte, la evolución del Proyecto se ha basado en establecer objetivos determinados, y cuando esos objetivos se han alcanzado, se crean otros. En esta aplicación, cada estructura tiene una función, del mismo modo que las piezas de un motor cumplen tareas concretas. La organización para trabajar de esta manera ha resultado satisfactoria, pues cada parte del código puede considerarse un módulo que funciona sin las demás partes, pero que juntas realizan la aplicación.

- El objetivo de este Proyecto se ha alcanzado, que es la elaboración de una aplicación en LabVIEW para la medición de la fuerza centrípeta en un equipo de laboratorio, gracias a dos sensores. Sin embargo, igual de importante es la actitud para realizar este TFG. En muchas ocasiones no se conseguían los valores esperados, el código no funcionaba como se quería o se desconocía el funcionamiento exacto de los sensores. A pesar de estas adversidades, con esfuerzo y constancia se ha llegado a la meta. Además, se ha comprendido la gran funcionalidad de la automática.

- En definitiva, se ha logrado la creación en LabVIEW de un programa para el estudio de la fuerza centrípeta en un equipo de laboratorio, y en el transcurso de este Proyecto se ha aprendido a manejar el software, a comprender los distintos dispositivos para adquirir datos y a establecer objetivos de manera organizada, todo ello con trabajo y perseverancia.

BIBLIOGRAFÍA

- MARTÍN BRAVO, Mª Ángeles. *Fundamentos de Física: mecánica, electromagnetismo*. Valladolid: Universidad, Secretariado de Publicaciones, 1993.

- TIPLER, Paul A; MOSCA, Gene. *Física para la ciencia y la tecnología. Mecánica, oscilaciones y ondas, termodinámica*. Barcelona: Reverté, 2010.

- RESNICK, Robert; HALLIDAY, David; S. KRANE, Kenneth. *Física, volumen 1*. México: Patria, 2009.

- ALONSO, Marcelo; FINN, Edward J. *Física*. Argentina [etc.]: Addison-Wesley Iberoamericana, 1995.

- LÁZARO, Antoni Mánuel; DEL RÍO FERNÁNDEZ, Joaquín. *LabVIEW 7.1. Programación gráfica para el control de instrumentación*. Madrid: Paraninfo, 2005.

- LAJARA VIZCAÍNO, José Rafael; PELEGRÍ SEBASTIÁ, José. *LabVIEW. Entorno gráfico de programación*. Barcelona: Marcombo, 2007.

- SISTEMA MECÁNICO ROTACIONAL ME-8950 DE PASCO https://www.pasco.com/prodCatalog/ME/ME-8950_complete-rotational-system/index.cfm, 2016. Online; accedido el 19-Septiembre-2016.

- PLATAFORMA DE ROTACIÓN ME-8951 DE PASCO https://www.pasco.com/prodCatalog/ME/ME-8951_rotating-platform/index.cfm, 2016. Online; accedido el 26-Septiembre-2016.

- MOTOR ROTACIONAL ME-8955 DE PASCO https://www.pasco.com/prodCatalog/ME/ME-8955_rotational-motor-drive/index.cfm, 2016. Online; accedido el 10-Octubre-2016.

- ADAPTADOR ROTACIONAL BASE "A" CI-6690 DE PASCO https://www.pasco.com/prodCatalog/CI/CI-6690_a-base-rotational-adapter/index.cfm, 2016. Online; accedido el 7-Noviembre-2016.

- SENSOR DE MOVIMIENTO ROTATORIO CI-6538 DE PASCO https://www.pasco.com/prodCatalog/CI/CI-6538_rotary-motion-sensor/index.cfm, 2016. Online; accedido el 5-Diciembre-2016.

- SENSOR DE FUERZA CI-6746 DE PASCO
  https://www.pasco.com/prodCatalog/CI/CI-6746_economy-force-sensor/index.cfm, 2017. Online; accedido el 9-Enero-2017.

- ACCESORIO DE FUERZA CENTRÍPETA ME-8089 DE PASCO
  https://www.pasco.com/prodCatalog/ME/ME-8089_computer-based-centripetal-force-accessory/index.cfm, 2017. Online; accedido el 23-Enero-2017.

- ADAPTADOR ANALÓGICO ELVIS CI-6718 DE PASCO
  https://www.pasco.com/prodCatalog/CI/CI-6718_analog-elvis-adapter/index.cfm, 2017. Online; accedido el 20-Febrero-2017.

- ADAPTADOR DIGITAL ELVIS CI-6719 DE PASCO
  https://www.pasco.com/prodCatalog/CI/CI-6719_digital-elvis-adapter/index.cfm, 2017. Online; accedido el 20-Febrero-2017.

- ADQUISICIÓN DE DATOS EN LabVIEW
  https://www.ni.com/data-acquisition/esa/, 2017. Online; accedido el 20-Marzo-2017.

- MANUAL DE LA TARJETA DE ADQUISICIÓN DE DATOS NI USB-6210
  https://www.ni.com/pdf/manuals/371931f.pdf, 2017. Online; accedido el 10-Abril-2017.

- FUENTE DE ALIMENTACIÓN E3646A DE AGILENT
  https://www.keysight.com/en/pd-836435-pn-E3646A/60w-dual-output-power-supply-two-8v-3a-or-20v-15a?cc=ES&lc=spa, 2017. Online; accedido el 8-Mayo-2017.

- BENCHVUE SOFTWARE (LIBRERÍAS Y CONTROLADORES DE LA FUENTE DE ALIMENTACIÓN)
  https://www.keysight.com/main/software.jspx?cc=US&lc=eng&nid=-11143.0.00&id=1184883&pageMode=CV, 2017. Online; accedido el 22-Mayo-2017.

- CABLE DE CONEXIÓN GPIB/USB 82357B DE AGILENT
  https://www.keysight.com/en/pd-851808-pn-82357B/usb-gpib-interface-high-speed-usb-20?cc=ES&lc=spa, 2017. Online; accedido el 5-Junio-2017.

- TEORÍA DE ERRORES
  ttp://webpersonal.uma.es/~JMPEULA/teoria_de_errores.html#introduccion, 2017. Online; accedido el 12-Junio-2017.