DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE GRADO

CONTROL DE UN ACTUADOR DE RIGIDEZ VARIABLE

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Autor: David Gómez Martínez

Tutor: José Medina Hernández

Leganés, Septiembre de 2016

RESUMEN

En el presente documento se detalla el diseño, implementación y puesta en marcha de un sistema de control para monitorizar un actuador de rigidez variable. Contando con el actuador ya instalado (en un banco de ensayos que se encuentra disponible para la realización de este trabajo de fin de grado), se planteó tanto la lógica de control del actuador como la adquisición de datos procedentes de los sensores disponibles, de tal forma que el dispositivo se mantenga totalmente rígido mientras no se sobrepase un cierto valor de carga umbral, siendo este umbral ajustable según las necesidades de la operación que se esté realizando.

Para conseguir controlar el sistema, se utilizó como base el programa Labview, una plataforma y entorno de desarrollo para diseñar sistemas con un lenguaje de programación gráfico. Esta aplicación es procesada por un dispositivo embebido de bajo coste (NI myRIO), que cuenta con un procesador programable que ejecuta un sistema operativo en tiempo real, así como un FPGA personalizado. El elemento principal para el que se diseñó el sistema de control es el motor que se encarga del movimiento del robot, además de los encoders magnéticos utilizados para la medición de desplazamientos relativos.

Una vez realizado el sistema de control (con varios niveles de control como son el control de corriente, velocidad y fuerza del actuador), se han realizado diferentes ensayos para comprobar que los resultados experimentales obtenidos se asimilan a las simulaciones previas disponibles sobre el actuador, así como la capacidad de controlar un sistema completo a través de un dispositivo embebido de bajo coste.

INDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. PLANTEAMIENTO

1.1. Introducción

En los últimos años, se ha ido incrementado el interés en aplicaciones relacionadas con la interacción humano-robot en el entorno de trabajo. En el campo industrial, la seguridad de esta interacción se centra en reducir al máximo la posibilidad de contacto entre robot y humano, solo permitiéndose esta interacción en acciones de aprendizaje o mantenimiento de los propios robots [1]. Debido a ello, gran parte del desarrollo tecnológico en robótica y automática, se ha centrado en la manipulación rápida y precisa a través de materiales no flexibles y pesados con el objetivo de que el mecanismo robótico esté dotado de una rigidez estructural. El problema de estos materiales es que implican altos valores de inercia en los movimientos, y esto deriva en serios problemas para gestionar imprevistos, como por ejemplo la aparición de un objeto nuevo en escena o que el entorno haya sido modificado.

Sin embargo, dentro del ámbito de la robótica de servicio, la interacción física entre los robots y los humanos y el medio en general es cada vez más frecuente y necesaria, por lo que los avances tecnológicos no se pueden centrar solo en evitar las colisiones, sino que también debemos disminuir las consecuencias de las mismas, centrándonos en ello desde la etapa de diseño.

Como solución a este nuevo entorno de trabajo, donde los robots y los humanos comparten el mismo espacio, se ha planteado como una solución viable la utilización de los llamados robots con articulaciones flexibles (RAF). Los RAF están basados en el concepto de actuador que se ajusta a las fuerzas o pares externos que se le solicitan, a través de un movimiento suave de adaptación, y entendiendo la flexibilidad como una ventaja mecánica.

Dentro de los RAF ya explicados, se encuentran los actuadores de rigidez variable, conocidos como VSA (por las siglas en inglés: Variable Stiffness Actuator).

Los VSA se basan en la idea de desacoplar elásticamente la inercia del rotor de la inercia del eslabón, de tal manera que la inercia del primero no esté contribuyendo a aumentar la fuerza generada en una posible colisión accidental [2].

Controlar el robot en situaciones de colisión se ha convertido en una línea de investigación en dispositivos robóticos que actúan coordinadamente entre sí o con seres humanos. Más concretamente, el problema más típico que se estudia es aquel en el que se presentan situaciones donde se producen movimientos inesperados y donde es necesario garantizar siempre la seguridad física del ser humano. Las soluciones que plantean los VSA para este problema combinan tanto la reducción de los parámetros inerciales que permiten una limitación de las fuerzas en el impacto del robot, como la adición de componentes flexibles en su estructura. Por lo tanto, si el impacto finalmente se produjera, los daños se reducirían notablemente comparados con los de un robot tradicional rígido, debido a los siguientes aspectos:

- Los RAF son robots más ligeros, por lo que su energía cinética será menor cuando impacte con un ser humano.

- Su energía cinética, en el caso de que se produzca una colisión, se transforma principalmente en energía potencial elástica del VSA, por lo que se minimizan los daños por la colisión.

- Cuando se produce un impacto en un robot rígido, la fuerza de reacción en su extremo aumenta bruscamente en cuestión de microsegundos. Sin embargo, para el caso de un robot con rigidez variable, esta fuerza de reacción crece más lentamente, alcanzando valores pico menores, y en cuestión de milisegundos, por lo que nos podríamos centrar en un control de fuerza (en lugar de un control de posición) que anule el efecto dañino del impacto.

De esta forma, podremos cambiar la rigidez en función de la operación que se esté realizando, siendo alta cuando no haya peligro de colisión, y baja en el caso de sobrecargas o de impacto [3].

Además de las ventajas que ofrecen en los campos explicados hasta ahora, estos actuadores también están presentes en otros campos en auge, como la robótica de rehabilitación o las prótesis. Una característica de los VSA (aparte de reducir las elevadas fuerzas originadas en la articulación) es su capacidad de almacenar y devolver la energía gracias a los elementos elásticos pasivos de los que están compuestos. De esta forma, el uso de este tipo de actuadores se está ampliando en aplicaciones donde la eficiencia energética se puede incrementar, modificando la frecuencia natural del actuador. Se puede ver un ejemplo del desarrollo en este ámbito en los robots caminantes, donde se puede ajustar la rigidez de la articulación según la velocidad del paso. Esta circunstancia permite un movimiento más natural y más eficiente en términos energéticos.

Para conseguir que los VSA se utilicen en diferentes aplicaciones que en la actualidad están reservadas para robots rígidos, debemos seguir estudiando los sistemas con elementos flexibles y rigidez variable, así como las técnicas de control. De esta forma, conseguiremos eliminar inconvenientes y que la interacción humano-robot se produzca de forma armónica y con una alta seguridad. Debido a ello, este trabajo de fin de grado está centrado en el control de un actuador de rigidez variable dentro de un entorno real (disponiendo de un banco de ensayos para realizar las distintas pruebas), para así conseguir nuevos datos y mejoras en el comportamiento de este tipo de dispositivos, además de la justificación de que se pueden controlar a través de sistemas embebidos de bajo coste.

1.2. Motivaciones y origen del proyecto

En la actualidad, la presencia de los robots está creciendo cada vez más, además, cada vez con más frecuencia, se utilizan para realizar funciones relacionadas con la asistencia sanitaria y la ayuda a los discapacitados.

Debido a ello, una de las áreas de investigación más interesantes dentro de la robótica es la interacción humano-robot. Tanto los expertos como el público en general están de acuerdo en que los robots deben mostrar un comportamiento inteligente, especialmente si este comportamiento está involucrado con un entorno humano, ya que así se consigue establecer una confianza hacia el robot. Existen criterios como la "seguridad física" o "sensación de seguridad" que llevan a analizar de manera conceptual parámetros intrínsecos del diseño, que de forma directa o indirecta tienen que ser considerados.

Los actuadores de rigidez variable, que mezclan electrónica y mecánica, pueden ser combinados también con un conocimiento de la anatomía humana, lo que generaría una mayor calidad de vida para personas que tienen partes de su cuerpo amputadas, que han sufrido un accidente cerebro-vascular o que tienen una discapacidad permanente. Estos dispositivos pueden ser incorporados en exoesqueletos, prótesis de rodillas, brazos mecánicos, o robots que pueden ayudar a una persona en el día a día (Figura 1.1).

Con todas estas motivaciones, y centrándonos en el estudio de los VSA, lo que se pretende es que los robots salgan del entorno industrial, consiguiendo aumentar las aplicaciones en las que se pueden usar robots para facilitar la vida de las personas. Esto también contribuye a iniciar una inmensa cantidad de soluciones que pueden ser válidas en un futuro no muy lejano.

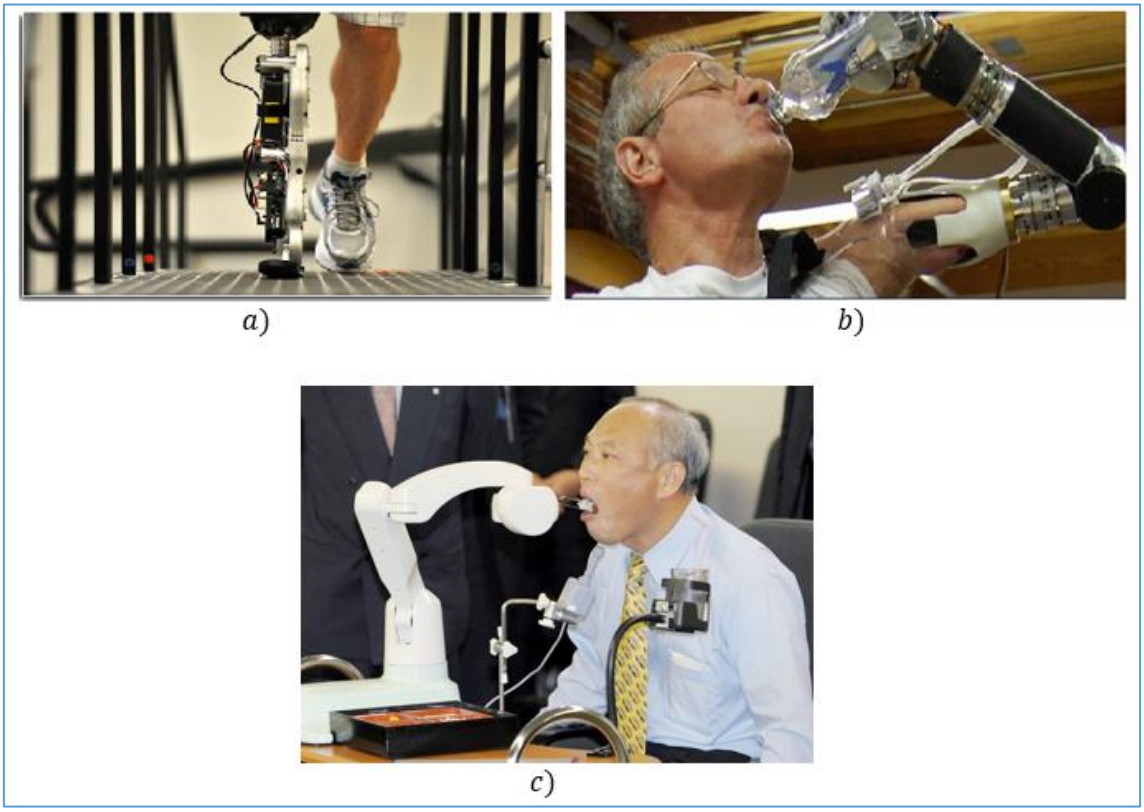

Figura 1. 1: Aplicaciones de los actuadores VSA: a) Prótesis de rodilla flexible [4], b) Prótesis de brazo robótico con codo de rigidez variable [5] y c) Robot asistencial con control de la fuerza [6]

Debido a la necesidad de seguir consolidando la seguridad de los dispositivos robóticos, la Universidad Carlos III de Madrid (UC3M) proyectó y construyó el dispositivo con pre-tensión MMJS (Mechanism of Multiple Joint Stiffness) cuyo diseño está basado en el cambio de rigidez pasivo mediante muelles pre-comprimidos (Figura 1.2). Este dispositivo incorpora un elemento flexible en el mecanismo de transmisión, que puede ser comprimido de manera independiente por un segundo actuador. Al tener una característica fuerza-elongación no lineal, la compresión aplicada sobre el resorte traslada el punto de operación del mismo a diferentes valores de rigidez [7].

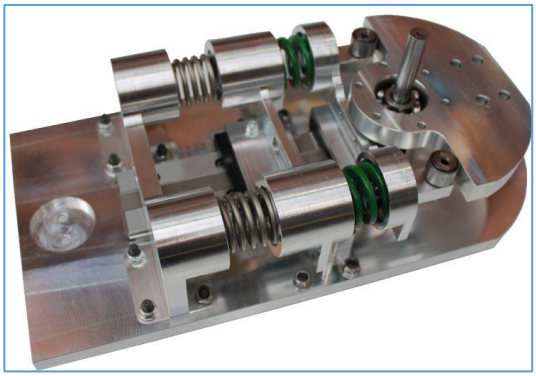

Figura 1. 2: Actuador de rigidez variable: MMJS

Gracias al banco de ensayos fabricado por la UC3M en el laboratorio de robótica de la misma universidad, este dispositivo podrá ser testeado, y de esta forma poder realizar una comparación entre los resultados experimentales y los estudios teóricos previos. Este dispositivo será evaluado en diferentes situaciones y configuraciones, y podremos conseguir obtener nuevos datos sobre comportamientos ante situaciones reales gracias al control de los mismos, ayudando a que comprendamos mejor este tipo de dispositivos.

1.3. Objetivos

El principal objetivo de este trabajo de fin de grado es realizar un control tanto a alto como a bajo nivel para un sistema articular con rigidez variable, más concretamente, para el dispositivo MMJS. Además de esto, se quiere implementar con un diseño embebido de bajo coste que permita que el sistema disponga de una alta flexibilidad ante posibles cambios en el banco de ensayos (cambios de motor, de sensores, etc.). Gracias al diseño de control embebido a través de la FPGA del dispositivo myRIO, también se intentará conseguir trabajar en tiempo real, condición importante en este tipo de dispositivos, que necesitan obtener una respuesta correcta en un tiempo determinado y relativamente rápido.

Para conseguir estos objetivos, las metas que se propusieron fueron las siguientes:

- Adaptación de las librerías disponibles en maxon al dispositivo myRIO para que éste pueda controlar el proceso.
- Diseño del sistema de control para la actuación del sistema MMJS.
- Implementación de las pruebas de control, en sus diferentes configuraciones, para el sistema MMJS.
- Desarrollo de un sistema de adquisición de datos (con el mismo dispositivo myRIO) procedentes de los sensores disponibles del banco.
- Adquisición de datos y tratamiento de los mismos, así como el análisis de los resultados.

1.4. Presupuesto

A continuación, en la Tabla I, se muestra el presupuesto aproximado del coste de todos los materiales utilizados para la consecución del trabajo de fin de grado propuesto.

Tabla I: Presupuesto del proyecto

Ítem	*Descripción*	*Cantidad*	*Precio*
1	**Controlador para motores Maxon EPOS2.** *Versión 70/10.*	1	*571,54 €/ud.*
2	**Motor Maxon con escobillas de grafito.** *Modelo 370354 RE-50 200 W.*	1	*428,00 €/ud.*
3	**Reductor planetario Maxon para motor.** *Modelo 110505 62 a 50 Nm.*	1	*503,09 €/ud.*
4	***Encoder 3 canales con Line Driver para motor.*** *Modelo MR 500 ppv*	1	*90,91 €/ud.*
5	**Encoder magnético ASM.** *Modelo POSIROT PMIS4.*	1	*123,02 €/ud.*
6	**Software NI-Labview.** *Versión profesional 2014.*	1	*1443,75 €/ud.*
7	**Dispositivo NI-myRIO Académico.** *Modelo 1900*	1	*560 €/ud.*
		PRESUPUESTO TOTAL	***3720,31 €***

1.5. Estructura de la memoria

A continuación se muestra cómo están dispuestos los diferentes capítulos del documento, así como la información que contiene cada uno de ellos:

Capítulo 1: El primer capítulo de este documento detalla el planteamiento general del proyecto. En él se plasma una breve introducción donde se explica el significado de los actuadores de rigidez variable, así como una explicación acerca de las líneas principales del proyecto. Posteriormente se explican las motivaciones y razones que mueven a la realización de dicho proyecto, y los objetivos que se pretenden alcanzar con la realización del mismo. En último lugar se expone un presupuesto detallado del proyecto y se especifica la estructuración de los capítulos del presente documento junto con sus correspondientes contenidos.

Capítulo 2: En este capítulo se realiza una descripción del estado del arte actual de las articulaciones flexibles y de los dispositivos de control utilizados, realizando un primer acercamiento formal al lector y dándole a conocer los avances que se han producido, estableciendo comparaciones con otros conocimientos paralelos

Capítulo 3: En este capítulo se explica la parte práctica del proyecto, ya que se expone el diseño del sistema, donde se profundiza y se revisan los componentes que integran y forman parte del banco de ensayos, mostrando una visión particular y general del funcionamiento del mismo. Además también se explica el sistema de control usado y los controladores utilizados para conseguir la comunicación y control del motor.

Capítulo 4: En este capítulo se continúa con la parte técnica, donde se explica todo el software diseñado para realizar el sistema de control del actuador de rigidez variable. En cuanto al control, se realizan varias estrategias de control de corriente, velocidad, posición y fuerza, que serán explicadas en detalle.

Capítulo 5: En el quinto capítulo se realizan los ensayos de control para el dispositivo MMJS, incluyendo las gráficas de los resultados obtenidos con los mismos y la comparación de éstos con los estudios teóricos.

Capítulo 6: Para finalizar, en el último capítulo, se comentan las conclusiones generales adquiridas durante el desarrollo de todo el proyecto y las futuras líneas de investigación y posibles futuros trabajos sobre este proyecto.

2. ESTADO DEL ARTE

2.1. Introducción

En la actualidad, es un gran reto que los actuadores mecánicos consigan las capacidades de rigidez y flexibilidad que se dan en los seres humanos, ya que un ser humano tiene la capacidad de contraer los músculos para reaccionar con la rigidez adecuada a las perturbaciones, y relajarlos de forma casi instantánea posteriormente. Para que un sistema técnico pueda imitar esas capacidades, debe contar con varios niveles de rigidez y una oposición al movimiento alta o baja, dependiendo de cada situación, pero por el momento, no se han conseguido actuadores que cumplan todos los requisitos al mismo tiempo.

Para lograr este reto, se han construido actuadores flexibles pasivos con rigidez variable como el dispositivo utilizado en este trabajo de fin de grado conocido como MMJS.

Este dispositivo (Figura 2.1), cuyo funcionamiento se explicará con más detalle a continuación, es un mecanismo pasivo, ya que no tiene un segundo sistema de accionamiento en la misma articulación. Sin embargo, se trata de un sistema conjunto de rigidez variable, debido a que tiene tres fases de funcionamiento distintas, donde cada fase está definida por un valor de rigidez diferente, en función del par elástico que se produce en la articulación.

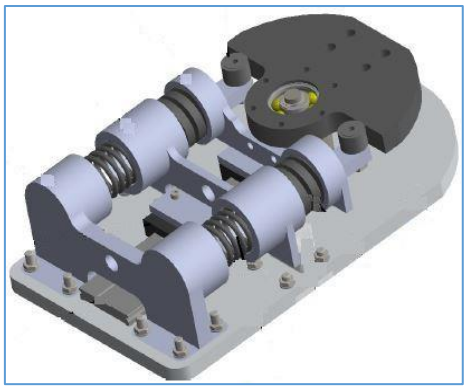

Figura 2. 1: Modelo CAD del actuador de rigidez variable MMJS

Para conseguir realizar un control tanto a bajo como a alto nivel de este actuador de rigidez variable, se ha decidido utilizar un sistema embebido, ya que gracias a ello se consigue un diseño de control sencillo, con una alta flexibilidad, y relativamente de bajo coste, y que además, permitiría trabajar en tiempo real, aspecto muy importante en el mundo de la robótica.

La elección de este sistema embebido fue el NI myRIO 1900, que se explicará con detalle en el apartado 3.4, ya que cumplía con todos los requisitos deseados para el sistema de control, y además tiene la opción de trabajar con un programa como Labview, en el que se utiliza un

lenguaje de programación gráfico, que es mucho más intuitivo que cualquier lenguaje tradicional.

Además del controlador maestro (NI myRIO), el sistema dispone de un controlador esclavo, que en el caso de nuestro sistema será el dispositivo EPOS2 70/10, que es el encargado de comunicarse con el motor principal con las órdenes que reciba del NI myRIO, para que de esta forma se consiga un control del sistema.

Con este conjunto que se ha introducido, se obtiene el sistema completo, con el que se controlará el motor que mueve al actuador de rigidez variable MMJS, y se realizarán las pruebas necesarias en el mismo para sacar una conclusión del avance conseguido en este aspecto.

A continuación, se detallan todos los conceptos introducidos en este apartado, para que el lector tenga una visión más específica del estado actual del sistema disponible para la realización de este trabajo de fin de grado.

2.2. Articulación elástica serie

2.2.1. Definición

Matthew Williamson, uno de los principales investigadores sobre el concepto de articulación elástica serie (SEA, por sus siglas en inglés que significan Series Elastic Actuation), afirma que la principal ventaja de introducir un elemento elástico a la salida de un actuador es que aumenta la calidad del control de la fuerza [8]. Este tipo de actuador se contradice con la robótica convencional, ya que siempre ha estado basada en la maximización de la rigidez para obtener un ancho de banda de control alto y pequeños errores en el seguimiento del robot.

Los actuadores más típicos para el accionamiento de un robot son los motores eléctricos. Sin embargo, este tipo de motores no son capaces de generar grandes fuerzas con una baja velocidad, que es lo más usado en las tareas robóticas. Por lo tanto, este tipo de motores requieren la utilización de reductoras para superar este problema. La introducción de una reductora tiene algunos inconvenientes como la fricción, el ruido y posibles reacciones violentas. Debido a ello, la implementación de un elemento elástico a continuación de la reductora reduce los inconvenientes que se introducían con ésta, y también se crea la posibilidad de mejorar la detección de fuerza, que a su vez hará que se mejore el control de la misma. Otra de las ventajas de utilizar una SEA para la robótica es la mejora de la seguridad con respecto a los propios robots y al entorno.

En la Figura 2.2, se puede observar el modelo de control básico en bucle cerrado para una SEA. El sistema de control es similar al de cualquier actuador.

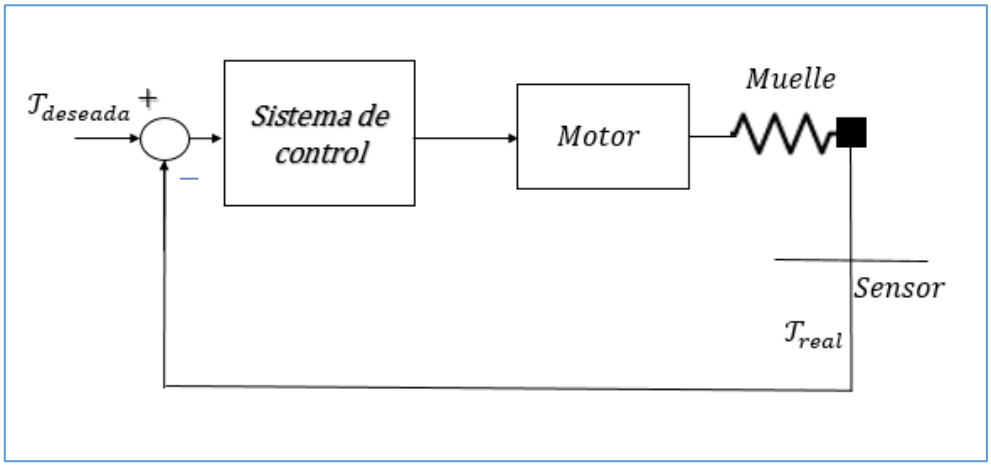

Figura 2. 2: Sistema de control para una actuador elástico serie

2.2.2. MMJS (Mecanismo de múltiple rigidez articular)

El diseño del dispositivo MMJS está basado en un conjunto de resortes lineales que están precargados e instalados para que consigan obtener las particularidades de los actuadores elásticos serie y limiten la fuerza máxima del sistema. Este dispositivo es un actuador pasivo, ya que no dispone de un segundo elemento de accionamiento en la misma articulación. Sin embargo, es un mecanismo de rigidez variable gracias a las tres zonas de operación en las que es capaz de trabajar. Cada una de las zonas está caracterizada por un valor de rigidez diferente, variando en función del par elástico generado en la articulación [9].

En la Figura 2.3a se muestra el mecanismo capaz de transformar el movimiento articular en movimiento lineal. El MMJS está compuesto por una manivela central, un rodillo, un pasador y un pistón. La manivela puede rotar con respecto al eje de rotación O. Con esta rotación, genera un ángulo de deflexión β alrededor del eje x. La corredera interior hace posible que el rodillo pueda deslizarse. Además, el rodillo actúa como un apoyo al pistón, que tiene su movimiento limitado solo al eje y.

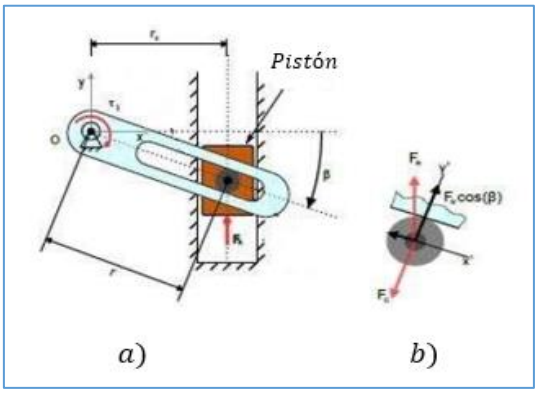

Figura 2. 3: a) Mecanismo de corredera doble y b) fuerzas en el pasador

Si se aplica una fuerza de par $\mathcal{T}_i$ al MMJS, el contacto entre la corredera interior y el rodillo produce una fuerza F_C que es transmitida al pistón a través del pasador. Si lo que se busca es el equilibrio estático del sistema, se debe aplicar una fuerza F_R adecuada sobre el pistón, en la dirección del eje y positivo, como se muestra en la Figura 2.3a. La relación entre la fuerza aplicada sobre el pistón F_R y la fuerza de par $\mathcal{T}_i$, se obtiene realizando el balance de fuerzas que actúan sobre el pasador (Figura 2.3b). Si tenemos en cuenta que la distancia entre el eje giratorio y el pasador r varía de acuerdo con la siguiente ecuación, $r = r_0 \cdot \sec\beta$, se puede demostrar que la relación entre ambas fuerzas es la siguiente:

$$\frac{\mathcal{T}_i}{F_R} = r_0 \tag{1}$$

Esta relación es independiente del ángulo de desviación, y es igual a la distancia mínima de separación del eje de rotación desde el centro de la segunda.

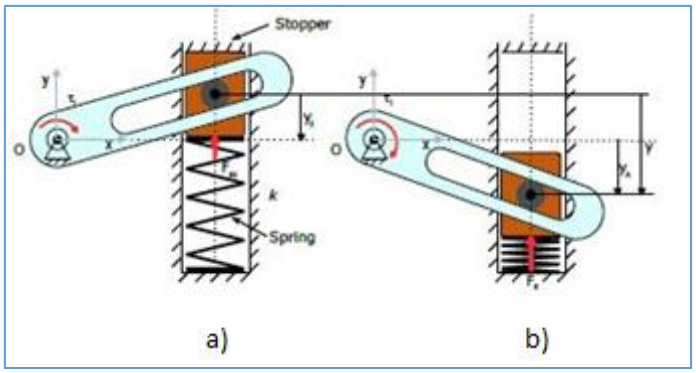

Figura 2. 4: Mecanismo con flexibilidad: a) Estado inicial de equilibrio y b) configuración general.

Para conseguir que el sistema tenga un comportamiento elástico, hay un resorte conectado entre el pistón y el extremo inferior de la segunda corredera (Figura 2.4a). Además, también se añade un tope en el extremo superior para limitar el movimiento del pistón y así mantener al resorte siempre con una compresión mínima.

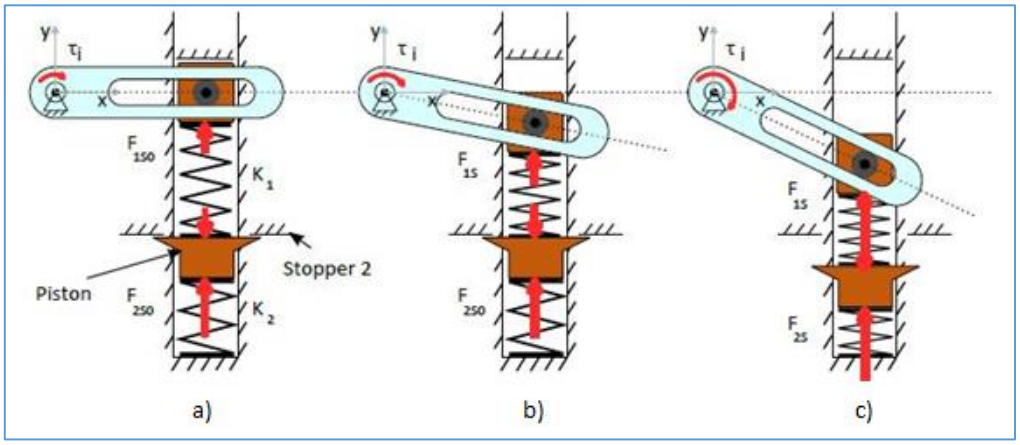

Figura 2. 5: Mecanismo de doble muelle: a) Equilibrio inicial, b) flexibilidad debida a una primera fuerza y c) flexibilidad debida a una segunda fuerza

La articulación robótica ha sido diseñada con el objetivo de que dicha articulación disponga de una múltiple rigidez, y por ello está basada en el principio de funcionamiento de doble resorte. Esta articulación, como se muestra en la Figura 2.6, está formada por una placa base, una leva, dos seguidores, tres soportes principales, muelles principales y secundarios, topes, una guía de carriles y dos carros. La leva es capaz de girar alrededor del eje de rotación fijo que se encuentra en la placa base. Cuando ésta gira, su contorno empuja a uno de los dos seguidores, dependiendo de en qué dirección está rotando. Ambos seguidores están fijados a los primeros soportes, pudiendo empujar al sistema de muelles.

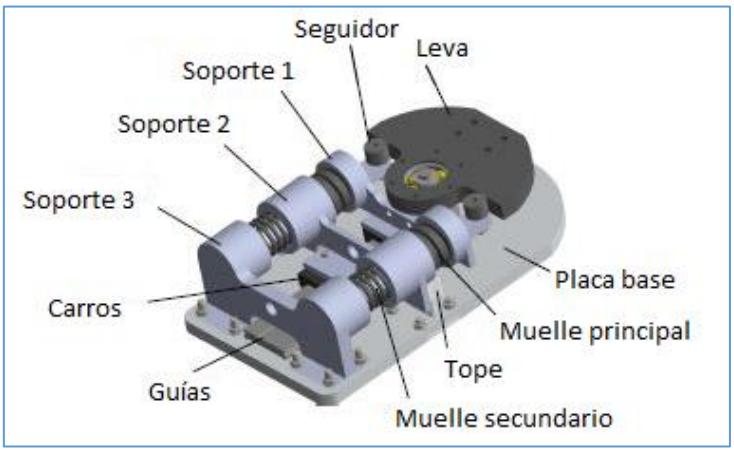

Figura 2. 6: Prototipo en CAD del MMJS

La corredera de la Figura 2.5, que permite a los dos pistones poder deslizarse sobre el mismo eje, ha sido implementada a través de unos carriles guía y dos carros (véase la Figura 2.6). El carro más cercano al eje sobre el que gira la leva está apoyado sobre el primer soporte, y su función es transmitir la fuerza que aplica la leva al conjunto de muelles de mayor rigidez. El segundo carro se encuentra apoyado al segundo soporte, y su objetivo es trasladar la fuerza de

los muelles de mayor rigidez superiores a los muelles de menor rigidez inferiores. La distancia que existe entre los topes y el tamaño de los muelles seleccionados determina la precarga existente tanto en los resortes rígidos (superiores) como en los flexibles (inferiores).

Si se aplica una carga externa como se puede observar en la Figura 2.7, se genera un ángulo de desviación, y esto provoca que se produzca el deslizamiento de los soportes 1 y 2 en la dirección de la guía.

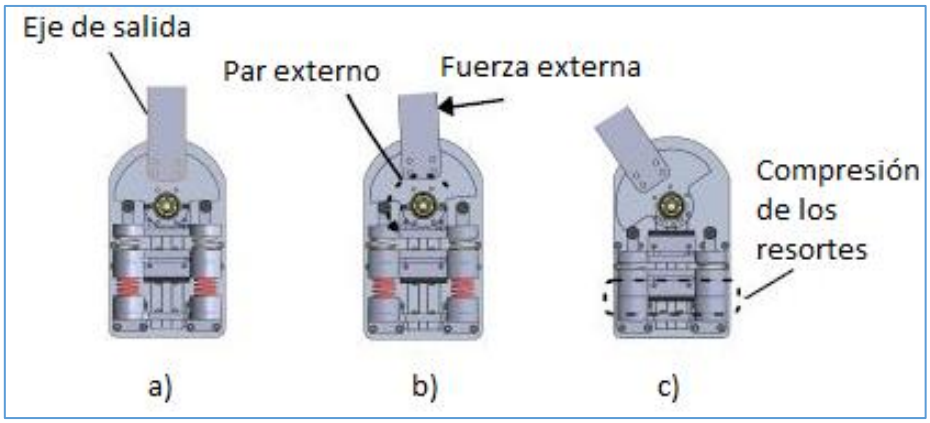

Figura 2. 7: a) Posición inicial, b) pequeña fuerza externa que produce la compresión de los resortes superiores y c) gran fuerza externa que produce la compresión de los resortes superiores e inferiores

Cuando los resortes inferiores logran alcanzar su máxima compresión, se produce la deflexión máxima, y es en este punto donde los resortes obtienen su longitud sólida.

Con todo esto, se logra obtener una primera área de operación de alta rigidez diseñada para pares bajos. Esto hace que la precisión del posicionamiento sea mayor de forma más sencilla, y su valor puede ser configurado a través de los muelles principales. Existe una segunda zona de operaciones de media-baja rigidez, y está caracterizada por reducir el consumo de energía en los movimientos cíclicos.

Al disponer de la característica de poder cambiar la rigidez rápidamente de un valor a otro, conseguimos una reducción del tiempo de reacción antes de una posible colisión.

2.3. Sistemas embebidos

2.3.1. Definición

Un sistema embebido (Figura 2.8) es un sistema basado en uno o varios microprocesadores que están construidos para controlar una función o un conjunto de funciones en tiempo real, y que no está diseñado para ser programado por el usuario final de la misma manera que el sistema de un ordenador. En un sistema embebido, un usuario puede tomar decisiones respecto a su funcionalidad, pero no puede cambiar la funcionalidad del sistema mediante la

sustitución del software. Con un ordenador, esto es exactamente lo que un usuario puede hacer [10].

Estos sistemas están diseñados para realizar una tarea en particular, pero teniendo la posibilidad de abordar esa tarea con diferentes opciones. Este punto es importante, ya que es lo que diferencia a los sistemas embebidos de los ordenadores, donde el usuario puede cubrir un amplio rango de necesidades.

Un punto muy importante dentro de los sistemas embebidos es el tiempo real, entendiéndose por sistemas en tiempo real a aquellos sistemas en los que el control del tiempo es vital para el correcto funcionamiento. Los sistemas en tiempo real necesitan realizar ciertas operaciones o cálculos en un límite de tiempo. Donde ese límite de tiempo resulta crucial. Un ejemplo claro de un sistema de tiempo real es el control de tráfico aéreo [11].

Por lo general, los sistemas embebidos pueden ser programados en el lenguaje ensamblador del microprocesador, pero también se puede utilizar compiladores específicos, pudiendo así usar lenguajes más avanzados como C o C++.

Si el tiempo de respuesta en la aplicación que estemos diseñando no es un factor crítico (es decir, no necesitamos trabajar en tiempo real), también podemos utilizar el lenguaje JAVA.

Figura 2. 8: Sistema embebido [12]

2.3.2. Arquitectura

2.3.2.1. Microprocesador

En la definición dada anteriormente sobre sistema embebido, se da por hecho que éste alberga uno o más microprocesadores, que son los que se encargan de aportar la “inteligencia al sistema”. Especificando más, el microprocesador se encarga de las operaciones de cálculo principales del sistema, además de ejecutar el código necesario para realizar una tarea y dirigir el funcionamiento del resto de elementos por los que está rodeado.

En la Figura 2.9 se puede observar el aspecto que tiene un microprocesador de los que se utilizan para el diseño de los sistemas embebidos.

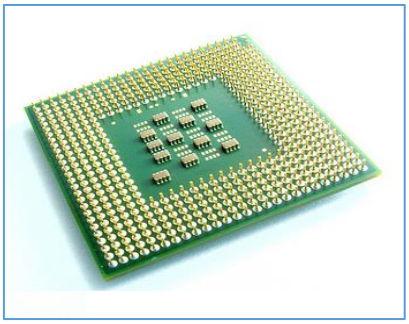

Figura 2. 9: Microprocesador [13]

2.3.2.2. Memoria RAM

Este elemento (Figura 2.10) es el encargado de almacenar el código de los programas que el sistema puede ejecutar, así como los datos del mismo. Una de sus principales características es que debe tener un acceso de lectura y escritura muy rápido, ya que así conseguimos que el microprocesador no pierda tiempo en tareas que no son de cálculo.

Debido a que es una memoria volátil, el sistema necesita un soporte donde se almacenen los datos incluso cuando no dispongamos de alimentación o energía.

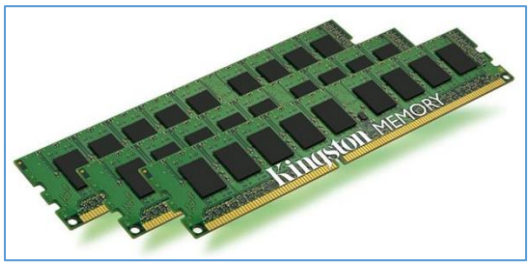

Figura 2. 10: Memoria RAM [14]

2.3.2.3. Memoria caché

Esta memoria (Figura 2.11) es más rápida que la RAM, y se utiliza para almacenar los datos y el código que han sido usados o solicitados con más frecuencia. Debido a que el sistema realiza microtareas (que en muchas ocasiones son repetitivas), la memoria caché consigue que se ahorre tiempo, ya que si el dato ya se encuentra dentro de la misma caché, el microprocesador no necesitará ir a la memoria principal a recoger ese dato.

Cuando el microprocesador necesita leer o escribir en una ubicación de la memoria principal, primero verifica que no existe una copia de esos datos en la memoria caché, ya que si es así, el microprocesador lee o escribe de inmediato en la caché, debido a que se realiza de forma mucho más rápida.

Figura 2. 11: Memoria caché [15]

2.3.2.4. Memoria no volátil

De forma habitual, a esta memoria se le conoce como "Disco Duro" (Figura 2.12). En él, la información es no volátil y además puede conseguir capacidades muy elevadas. A diferencia de la memoria RAM, que es de estado sólido, éste suele ser magnético en aplicaciones para ordenador, pero su excesivo tamaño y falta de robustez mecánica lo suele hacer inviable para sistemas embebidos, ya que suelen estar expuestos a condiciones de vibración.

Gracias a los avances tecnológicos, se ha conseguido resolver este problema para los sistemas embebidos, ya que se han desarrollado discos duros en estado sólido, y ahora, si es viable su utilidad en los mismos.

Figura 2. 12: Disco duro [16]

2.3.2.5. BIOS-ROM

BIOS (Basic Input & Output System, cuya traducción sería Sistema básico de entrada y salida) es un código necesario para que se inicialice el sistema y para poner en comunicación todos los elementos de la placa madre. La memoria ROM (Read Only Memory, memoria de solo lectura no volátil) es un chip donde se encuentra almacenado el código BIOS (Figura 2.13).

Figura 2. 13: BIOS-ROM [17]

2.3.2.6. CMOS-RAM

Es un chip de memoria que se puede utilizar para lectura y escritura. Está alimentado por una pila donde se almacena el tipo y la ubicación de los dispositivos conectados a la placa madre (disco duro, puertos de entrada y salida, etc.). Además, contiene un reloj en permanente funcionamiento que ofrece al sistema la fecha y hora.

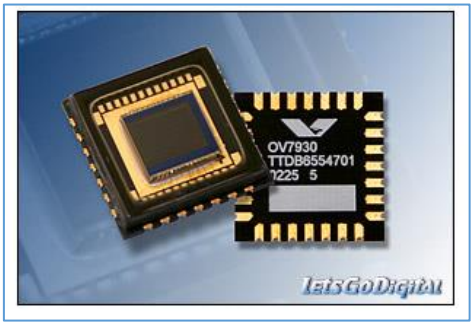

Figura 2. 14: CMOS-RAM [18]

2.3.2.7. Chipset

Este chip (Figura 2.15) se encarga de controlar las interrupciones que están dirigidas al microprocesador, el acceso directo a la memoria (DMA, que permite a cierto tipo de componentes del sistema acceder a la memoria para leer o escribir independientemente del microprocesador principal) y al bus ISA, además de ofrecer temporizadores, etc. Es muy frecuente que nos encontremos la CMOS-RAM y el reloj en tiempo real en el interior de este chip.

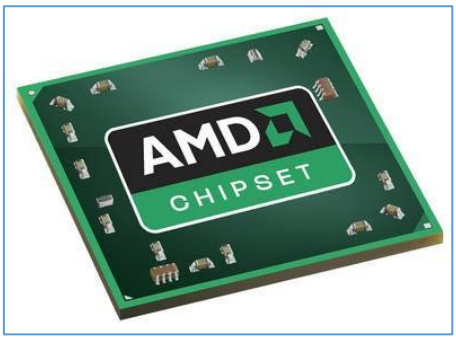

Figura 2. 15: Chipset [19]

2.3.2.8. Entradas/salidas al sistema

La mayoría de los sistemas embebidos están dotados de una serie de puertos de entradas y salidas tanto analógicas como digitales, que son nodos en los que los dispositivos periféricos se pueden conectar y pueden intercambiar información con la memoria y el microprocesador central.

Los puertos contienen en sí mismos un número definido de registros, los cuales son utilizados para el almacenamiento temporal de varios tipos de datos. Las direcciones de los registros y sus funciones están definidas con precisión. Entre estos puertos se encuentran los puertos serie, donde los datos se transfieren bita bit de forma secuencial, los puertos paralelos, donde los datos se transfieren en paralelo, y los puertos universales, como puede ser el USB.

2.3.3. Tiempo real

El término tiempo real dentro de los sistemas embebidos, significa que el sistema es capaz de interaccionar repetidamente con su entorno físico y responde a los estímulos que recibe del mismo dentro, dentro de un plazo de tiempo determinado [20].

Para que el funcionamiento del sistema sea correcto no basta con que las acciones que realiza sean correctas (validez lógica), sino que también necesitamos que se ejecuten dentro del intervalo de tiempo especificado (instante en que se produce la corrección).

Todas las actividades que realiza un sistema de tiempo real son conocidas como tareas, y éstas poseen diferentes tipos de requisitos:

- Funcionales: básicamente se refiere a los requisitos funcionales como lo "qué hace" el sistema cuando realiza esa tarea.

- No funcionales: dentro de estos requisitos se engloban tanto los temporales (cuándo se ejecutan, cuánto tardan…) como requisitos de fiabilidad y seguridad. También se incluyen aquí requisitos como el coste, consumo de energía, etc.

En la Figura 2.16, se puede observar cómo sería la ejecución de una tarea en un sistema en tiempo real.

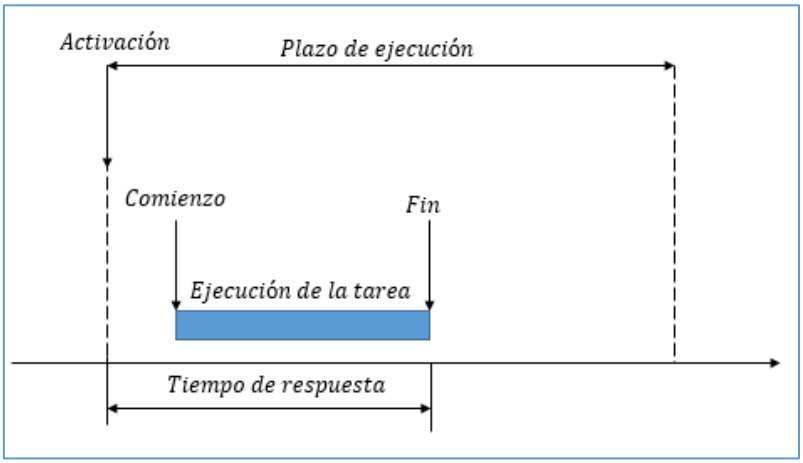

Figura 2. 16: Ejecución de una tarea en tiempo rea

Si el tiempo de respuesta es menor que el plazo de ejecución, la tarea se estará ejecutando en tiempo real (como en el caso de la Figura 2.16). Sin embargo, si por cualquier motivo este tiempo de respuesta superará al plazo de ejecución, el sistema no estaría cumpliendo los requisitos de tiempo real que se necesitan, y por lo tanto, la ejecución de esa tarea no se está realizando de forma correcta.

Dentro de los sistemas en tiempo real, hay tareas que necesitan que la respuesta sea exacta, por lo que el requisito temporal se consideraría tiempo real estricto (como por ejemplo en un control de frenado). Sin embargo, hay tareas que no necesitan cumplir los plazos de forma tan exigente, por lo que también existe un tiempo real flexible, donde se pueden perder algunos plazos de tiempo de vez en cuando, y tiempo real firme, donde solo se pueden perder plazos de tiempo ocasionalmente. Por lo tanto, en un mismo sistema en tiempo real, puede haber tareas con diferentes requisitos temporales.

En el caso de la robótica, hablar de tiempo real es muy importante, ya que cualquier tarea realizada a través de un robot (ya sea robótica industrial, humanoide, asistencial, etc.) necesita que la respuesta del mismo sea en un plazo de tiempo muy corto, ya que si esto no sucede así, las consecuencias pueden ser graves.

Debido a ello, en este proyecto se decidió elegir un sistema embebido para el control del actuador de rigidez variable, ya que este tipo de sistemas ofrecen la característica del tiempo real, y por lo tanto, permiten trabajar con un sistema flexible, donde el espacio que ocupa dentro del banco de ensayos es mínimo, y que además, consigue que el sistema a controlar trabaje justo en el tiempo que requiere la aplicación.

Por consiguiente, el control de todo el banco a través de un sistema embebido hará que se cumplan los objetivos que se habían marcado antes de realizar este trabajo de fin de grado. Además, en este caso particular, el sistema embebido utilizado da la opción de trabajar directamente sobre una FPGA o sobre el CPU, por lo que se puede realizar un control tanto a bajo como a alto nivel.

2.4. Labview y los sistemas RIO

2.4.1. Definición

Labview es un entorno de desarrollo integrado y diseñado específicamente para ingeniería. Este programa utiliza un lenguaje de programación gráfica (G) que usa un modelo de flujo de datos en lugar de líneas secuenciales de código de texto.

La base de este tipo de programación intuitiva es que permite escribir código funcional utilizando un diseño visual que se asemeja a su proceso de pensamiento. Esta característica se traduce en que la persona que lo utiliza trabaja más tiempo resolviendo los problemas que verdaderamente importan, a diferencia de los lenguajes de programación tradicionales, que requieren el empleo de cantidades grandísimas de tiempo en aprender la sintaxis necesaria que se asocia con el lenguaje.

Entre sus principales ventajas, ofrece una robusta capacidad de integración con una gran cantidad de dispositivos de hardware, brinda cientos de bibliotecas integradas para análisis avanzado y visualización de datos, provee de una variada gama de características y herramientas de asistencia al desarrollador, así como multitud de interfaces de usuario configurables.

Gracias a ello, la plataforma Labview se ha hecho líder en la industria desde su introducción en 1986.

2.4.2. Diferencias con lenguajes tradicionales

Labview tiene bastantes diferencias con los lenguajes de programación tradicionales, pero principalmente existen dos vertientes.

Por una parte, la programación se desarrolla cableando iconos gráficos en un diagrama que compila directamente a código máquina. Aunque su representación se haga gráficamente en lugar de con texto, contiene los mismos conceptos de programación que se pueden encontrar en la mayoría de los lenguajes tradicionales. Por ejemplo, incluye todas las construcciones estándar tales como tipos de datos, bucles, eventos, variables, recursividad y programación orientada a objetos (Figura 2.17).

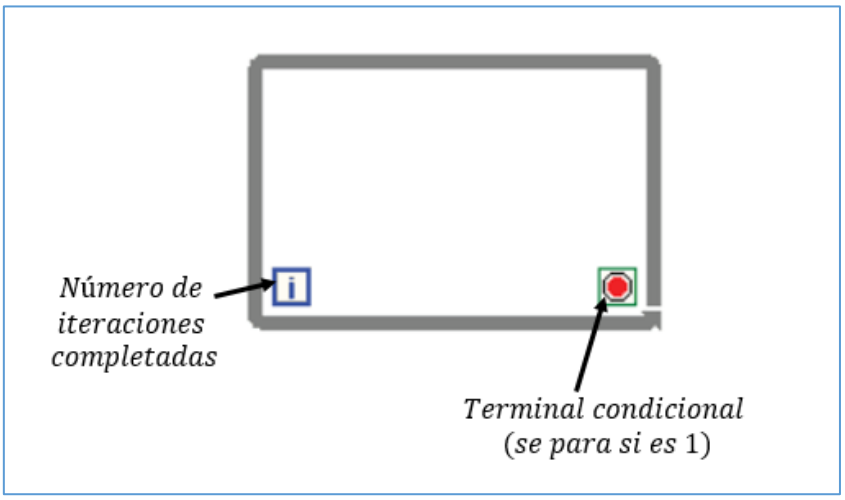

Figura 2. 17: Bucle while en Labview

Por otra parte, los nodos en un programa de LabVIEW (palabras, funciones y estructuras como bucles y subrutinas) tienen entradas, procesan datos y generan salidas. Una vez que todas las entradas de los nodos contienen un dato válido, dicho nodo ejecuta su lógica, produce datos de salida y transmite los datos al siguiente nodo en la secuencia del flujo. Un nodo que recibe datos de otro, puede ejecutarse si y solo si, el primero ha completado su ejecución [21].

2.4.3. Paralelismo en la programación

Los lenguajes de flujo de datos como LabVIEW permiten "paralelizar" automáticamente. A diferencia de los lenguajes secuenciales como C y C++, los programas gráficos contienen de forma inherente información sobre qué partes del código pueden ser ejecutadas paralelamente. Por ejemplo, un patrón común de diseño es el Productor/Consumidor (Figura 2.18), en el que dos bucles while se ejecutan independientemente: el primero es el responsable de la producción de datos y el segundo del procesamiento. En la ejecución en

paralelo los datos interactúan entre los bucles usando procesos de colas, las cuales, son estructuras de datos estándar en los lenguajes de programación de propósito general [22].

El paralelismo es importante en la programación, ya que desbloquea las ganancias de rendimiento relativas a los programas secuenciales debido a cambios recientes en el diseño de los procesadores. Como resultado, los fabricantes han diseñado nuevas arquitecturas con múltiples núcleos de procesamiento.

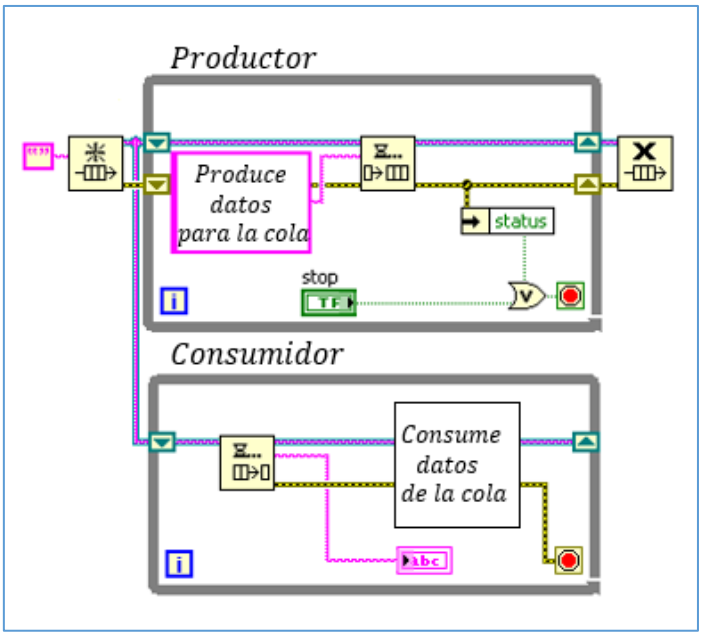

Figura 2. 18: Patrón de diseño Productor/Consumidor de Labview

El multihilo se usa para sacar el máximo provecho al rendimiento disponible en procesadores con varios núcleos. Si se utilizan los tradicionales lenguajes basados en texto, se deben crear y administrar hilos para implementar el paralelismo, un desafío de envergadura para programadores no expertos.

Por el contrario, la naturaleza paralela de LabVIEW hace que la multitarea y el multihilo sean fáciles de implementar. Además, el compilador trabaja continuamente para identificar secciones paralelas del código.

2.4.4. Sistemas RIO

La arquitectura de Labview en los sistemas RIO (Reconfigurable I/O) es una parte integral de la plataforma de diseño gráfico de sistemas de National Instruments. El diseño gráfico de sistemas es un método moderno para el diseño, creación de prototipos y despliegue de sistemas de monitorización y control, y combina el entorno de la programación gráfica de NI

Labview con el hardware abierto disponible en el mercado para simplificar enormemente el desarrollo, lo cual se traduce en diseños de mayor calidad con la posibilidad de incorporar un diseño personalizado [23].

Esta arquitectura está basada principalmente en cuatro componentes, que son los siguientes:

- Procesador: es utilizado para implementar el código con el fin de comunicarse con otras unidades de procesamiento, tales como FPGAs, interfaces con periféricos, datos de registro y ejecutar aplicaciones. National Instruments ofrece sistemas de hardware RIO en diversos formatos. Por una parte ofrece sistemas multinúcleo de alto rendimiento con multiprocesamiento simétrico (SMP), que ejecutan el sistema operativo Microsoft Windows. Por otra parte ofrece sistemas pequeños y embebidos de tiempo real como NI myRIO y CompactRIO.

- FPGA: la FPGA reconfigurable es el núcleo de la arquitectura del sistema de hardware RIO. Se utiliza para evitar cargar al procesador con tareas intensivas y proporcionar una ejecución determinista con un rendimiento extremadamente alto. La FPGA está directamente conectada a los módulos de E/S para permitir un acceso de alto rendimiento a la circuitería de cada módulo y una flexibilidad ilimitada de temporización, disparo y sincronización. Además, la FPGA ejecuta todo el código en el hardware, lo cual proporciona alta fiabilidad y determinismo.

- E/S modulares: los módulos de E/S de la serie C contienen aislamiento, circuitos de conversión, acondicionamiento de señales y conectividad integrada para la conexión directa a sensores/actuadores industriales. Al ofrecer diversas opciones de cableado e integración de la caja de conexiones del conector de los módulos, un sistema RIO reduce significativamente los requisitos de espacio y los costes del cableado en campo. En el caso de este proyecto, como se ha utilizado el dispositivo NI myRIO, no se dispone de E/S modulares, ya que vienen integradas en el mismo dispositivo.

- Plataforma de desarrollo Labview: National Instruments ofrece una solución completa para el desarrollo del diseño gráfico de sistemas de aplicaciones embebidas con el fin de que se puedan diseñar, crear prototipos y desplegar sistemas de manera eficiente con una sola plataforma de software. Gracias al software de diseño gráfico de sistemas Labview, se pueden desarrollar aplicaciones para el procesador, sintetizar el propio circuito de medida personalizado en la FPGA y luego integrar perfectamente los dos con E/S modulares para crear una solución completa RIO.

Estos cuatro componentes combinados, proporcionan la capacidad de crear rápidamente circuitos de hardware personalizado con E/S de alto rendimiento y una flexibilidad sin precedentes en el control de la temporización del sistema.

2.5. Controlador EPOS

A continuación se hará una definición general de los controladores EPOS, así como su arquitectura y los modos de operación con los que se trabajarán en este trabajo de fin de grado.

2.5.1. Definición

Los sistemas EPOS (Easy Position System) están basados en un controlador digital de posicionamiento con estructura modular (Figura 2.19). Estos sistemas resultan adecuados para motores de corriente continua de imán permanente, con y sin escobillas y con encoder incremental en un rango de potencia desde 1 hasta 700 W. Cuentan con un gran número de modos operativos, así como diferentes interfaces que permiten su control, consiguiendo así un uso flexible en los diversos sistemas de motores utilizados tanto en la tecnología de la automatización como en la mecatrónica [24].

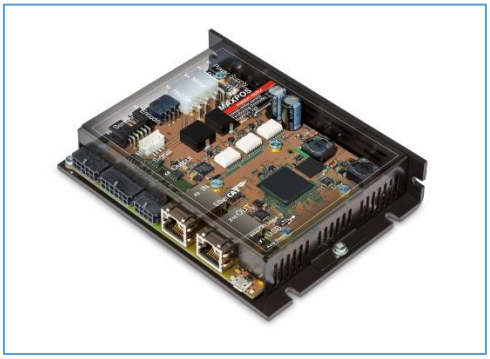

Figura 2. 19: Dispositivo EPOS con estructura modular

2.5.2. Arquitectura

2.5.2.1. Control

El control de velocidad es la principal función de estos controladores. Con este control lo que se intenta conseguir es que se mantenga una velocidad predeterminada tan estable como sea posible, independientemente de las fluctuaciones en la carga. Los circuitos electrónicos del controlador comparan de forma constante la velocidad de control (deseada) con la velocidad actual (real). El controlador detecta la diferencia que hay entre ambas y, actuando sobre la etapa de potencia, consigue compensarla. Esto se conoce como un control de velocidad en bucle cerrado (Figura 2.20).

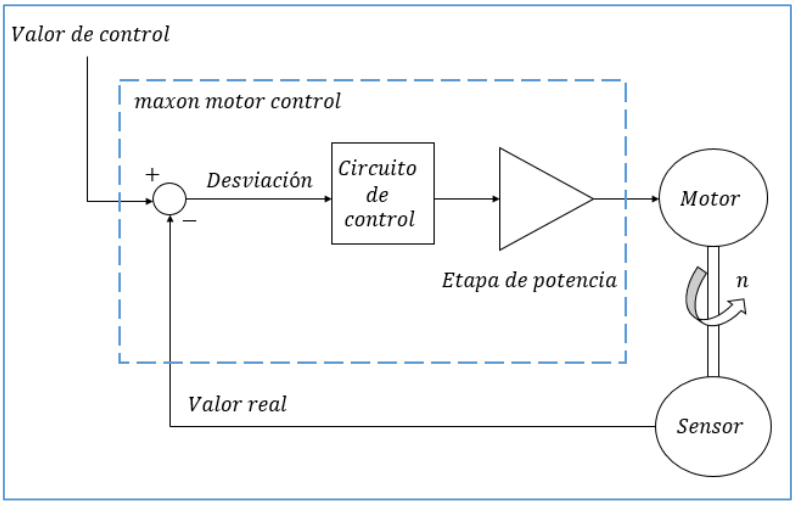

Figura 2. 20: Esquema del circuito de control

Por otra parte, estos controladores también permiten un control de posición, en el que se asegura que la posición medida a través de la información que el controlador recibe del encoder digital coincida con la posición deseada (Figura 2.21).

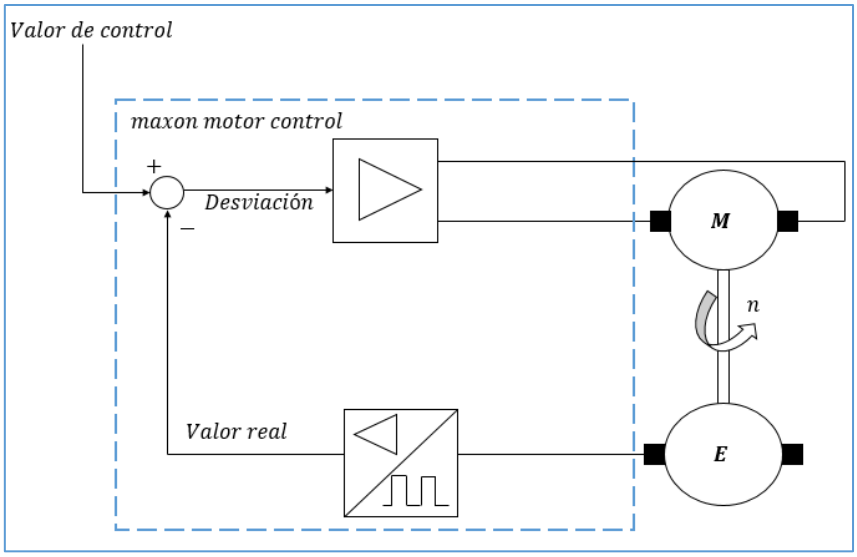

Figura 2. 21: Esquema de control por encoder

Por último, también se puede realizar un control de corriente. Se le proporciona al motor la intensidad correspondiente a una señal que se desea controlar. Como consecuencia de ello, el par del motor cambia según cambie valor de esta señal de control. Con el control de corriente se consigue mejorar las prestaciones de un sistema superior de control tanto de posición como de velocidad (Figura 2.22).

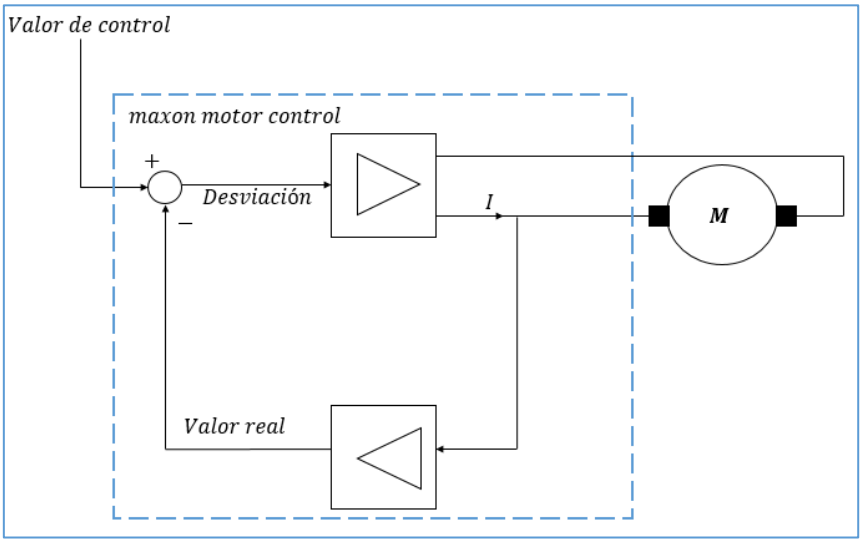

Figura 2. 22: Esquema de control de corriente

2.5.2.2. Comunicación

Estos dispositivos pueden estar comunicados y controlados mediante red CANopen, USB y RS232, así como otras interfaces de comunicación. En la aplicación de este trabajo de fin de grado, la comunicación tanto desde el ordenador como desde el dispositivo myRIO ha sido por medio de USB, debido a que los drivers que se encuentran disponibles solo permitían este tipo de comunicación.

2.5.3. Modos de operación

2.5.3.1. Profile Position Mode

Este modo de operación permite posicionar el eje del motor desde un punto A hasta un punto B mediante la generación automática de una trayectoria controlada. Este posicionamiento puede ejecutarse con respecto al origen absoluto del eje o con respecto a la posición actual u origen relativo. Cualquier modificación de los puntos inicial y final en el movimiento, originarán un ajuste de la trayectoria que se genera para adaptar los parámetros de aceleración y velocidad al valor más óptimo posible.

2.5.3.2. Position Mode

Este modo de operación es el más directo desde el punto de vista conceptual para alcanzar una posición deseada, ya que no hay generador de trayectoria ni interpolación intermedia. Directamente, el software del controlador intentará colocar el eje del motor en la posición predeterminada a su máxima capacidad, fijando la velocidad y la aceleración al límite seleccionado previamente por el operador.

Debido al funcionamiento tan directo de este modo, una mala planificación del movimiento por parte del operador puede causar una parada del sistema, ya que el motor opera al máximo de su capacidad y esto provoca el aumento de la probabilidad de sufrir grandes desfases entre los valores de posición, velocidad y aceleración deseados y los medidos por medio de los sensores. Si esto sucediese, el software produciría un mensaje de error que provocaría la detención de este modo.

2.5.3.3. Profile Velocity Mode

Al igual que el modo Profile Position Mode, genera una rutina de cálculo interna que, automáticamente, calcula nuevas posiciones del eje del motor teniendo en cuenta la posición actual, punto final, aceleración y la velocidad en un determinado momento. Es decir, calcula y controla las trayectorias necesarias para poder alcanzar la velocidad predeterminada por el operador.

De esta manera, el eje del motor sigue y mantiene una consigna de velocidad, hasta que esté reciba una nueva orden.

2.5.3.4. Velocity Mode

Este modo, al igual que ocurre en el Position Mode, consigue que el eje del motor gire a una velocidad deseada de forma directa a través del software del controlador, sin generar ninguna trayectoria para conseguirlo.

Este modo puede ser útil cuando se está realizando un control de posición maestro.

2.5.3.5. Current Mode

Este último modo de operación se basa en la función de control en corriente, en la cual se recibe y compara el valor de corriente demandada actual, con el valor máximo definido previamente. Este es el único modo en el que el actuador realizará el movimiento basándose en la regulación de la corriente y no en posición o velocidad.

3. DESCRIPCIÓN DEL SISTEMA

3.1. Introducción

A lo largo de este capítulo, se mostrará al lector las diferentes partes que conforman el banco de ensayos, el modelo de control que se ha utilizado para la monitorización del sistema y la descripción de los dispositivos y programas usados para construir el software que controla todo el proceso. Con ello, se podrá formar una imagen clara y general del conjunto de componentes que interactúan entre sí para lograr el objetivo de controlar y adquirir datos del actuador MMJS, así como de los ensayos realizados.

Como ya se ha introducido previamente, el actuador MMJS ha sido implementado mediante un solo motor que le permitirá moverse, encontrando un nuevo método pasivo basado en la pre-compresión de los muelles en el cambio de su rigidez, permitiendo de esta forma la eliminación del uso de un segundo motor.

Este actuador ha sido instrumentado mediante un diverso conjunto de sensores diseñados para la medida de posición, velocidad, corriente y fuerza. Además, se dispondrá de una estación de computación encargada de realizar el enlace entre el banco y el operador. Dicha estación permitirá monitorizar y controlar el actuador MMJS a través del dispositivo myRIO en todo momento durante las situaciones de prueba, permitiendo una forma de comprobación rigurosa, transparente y flexible de lo que está sucediendo.

3.2. Banco de ensayos

3.2.1. Introducción

El banco de ensayos cuenta con varias zonas bien diferenciadas entre sí. Existe una zona diseñada para el montaje y movimiento del actuador flexible con rigidez variable que hará las funciones del brazo robot (zona 1). En la parte más baja del banco se habilitó un espacio (zona 2), que estará destinado a los elementos electrónicos de instrumentación y control. Por último, existe un espacio dedicado a la monitorización, control y operación del banco por parte del operador (zona 3). Esta zona está apartada del propio banco para obtener mayor seguridad durante los ensayos realizados.

Figura 3. 1: Banco de ensayos completo: Zona 1: Actuador, zona 2: Mesa de instrumentación y zona 3: Puestos de control

La componente mecánica del banco de ensayos es básicamente el brazo robótico flexible, cuya función la realiza el dispositivo MMJS, que ya fue explicado anteriormente en el apartado 2.2.2.

Se han utilizado varios elementos de medida para realizar los ensayos necesarios, además de realizar un control lógico de la parte mecánica del banco para conseguir que los movimientos y niveles de rigidez sean los deseados en el actuador, así como para adquirir y tratar los datos procedentes de los elementos de medida. Todo ello será explicado en los apartados 3.2.2, 3.2.3 y 3.2.4

3.2.2. Sensores

Por una parte, el diseño del sistema del motor usado, permite obtener datos intrínsecos en tiempo real acerca de su funcionamiento, a través de un encoder acoplado en el propio conjunto del motor. En el motor utilizado, se pueden monitorizar tres datos fundamentales: posición, velocidad y corriente.

La posición será obtenida por la lectura del encoder acoplado de la Figura 3.2, que tiene unas características que se pueden observar en la Tabla II.

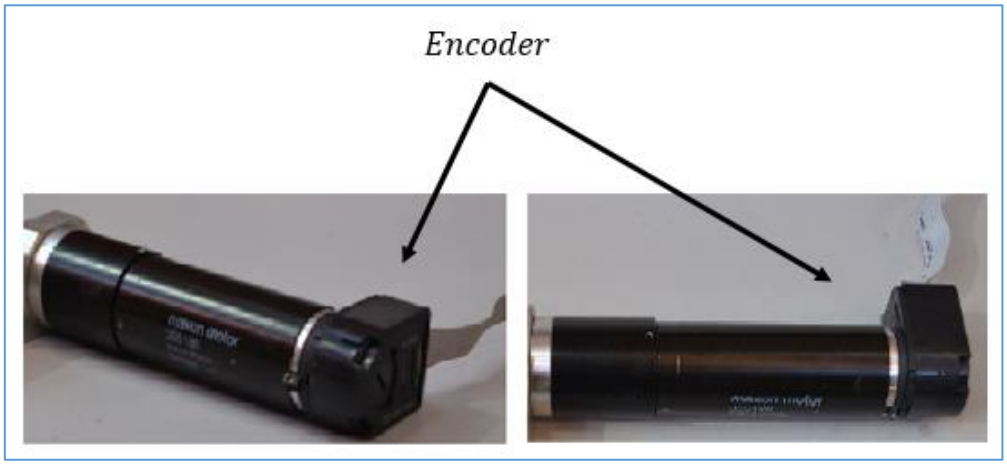

Figura 3. 2: Encoder acoplado en el motor

Tabla II: Características del encoder intrínseco

Tipo	**Valor**
Número de pulsos por vuelta	500
Número de canales	3
Line Driver	DS26LS31
Máxima velocidad	12000 rpm
Diámetro del eje	3 mm
Datos Técnicos	**Valor**
Tensión de alimentación	5 V ± 10 %
Máxima aceleración angular	250000 rad/s^2
Desfase	90°
Tiempo de flanco de subida	180 ns
Tiempo de flanco de bajada	40 ns

Además, gracias a la disposición geométrica de la bobina y el circuito magnético interno se consigue determinar de forma detallada como se transforma la potencia eléctrica de entrada (corriente y tensión), que la suministra directamente el dispositivo EPOS2, en potencia mecánica de salida (velocidad y par). De esta forma, se obtienen los datos de velocidad y corriente por medio del controlador, y los datos de posición por medio del encoder, que serán guardados y tratados por el sistema.

Además de este encoder interno, también se encuentra disponible un encoder externo al conjunto del motor, que es necesario para la realización de una correcta monitorización de todo el sistema. El modelo de encoder utilizado en este proyecto (Figura 3.3) es POSIROT PMIS4-20-512 ASM magnético, que está compuesto por un sensor y una rueda (cuyo modelo

es PMIR4-20-90). Las características de ambos componentes se pueden observar en las Tablas III y IV respectivamente.

Figura 3. 3: Encoder magnético (sensor y rueda)

Tabla III: Características del sensor

Datos Técnicos	**Valor**
Salida	TTL/24 V
Voltaje de excitación	10 a 30 VDC
Corriente de excitación	50 a 300 mA
Periodo magnético	2 mm
Espacio entre sensor y rueda	0,10 a 0,80 mm
Tolerancia	± 1 mm
Linealidad	± 0,1°
Repetitividad	± 1 dígito
Frecuencia de pulso	50 kHz, máxima 480 kHz
Peso	30 ± 5 g

Tabla IV: Características de la rueda

Datos Técnicos	**Valor**
Periodo de la señal	45000
R.P.M (a 480 kHz)	512

3.2.3. Actuadores

El motor utilizado en el banco de ensayos es un motor CC (Corriente Continua) de doscientos vatios cuyo fabricante es Maxon Motor. Será el encargado del movimiento de todo el brazo flexible. Este tipo de motores están caracterizados por la facilidad a la hora de controlarlos en cualquier posición y por la ausencia de no-linealidades en su comportamiento.

Además, como está fabricado sin núcleo de hierro, se evita que se produzcan pérdidas ni efectos de saturación.

El modelo del motor utilizado en este sistema, es el Modelo RE - 50 37354 (Figura 3.4) con las siguientes características (Tabla V):

Figura 3. 4: Motor de CC

Tabla V: Características del motor de CC

Datos del motor	**Valor**
Voltaje nominal	24 V
Velocidad sin carga	5950 rpm
Corriente sin carga	236 mA
Velocidad nominal	5690 rpm
Par nominal	406 $N \cdot m$
Corriente nominal	10,9 A
Especificaciones	**Valor**
Termal time constan winding	71,70 s
Termal time constan motor	1370 s
Máxima velocidad permitida	9500 rpm
Número de pares de polos	1
Máxima eficiencia	94 %

El sistema completo del motor no solo incluye el propio motor, sino que también incluye una reductora, que es necesaria para conseguir producir una potencia mecánica con un alto par, y con una reducción de velocidad. Concretando un poco más, las reductoras planetarias son las más indicadas para realizar dicha transmisión de pares elevados (Figura 3.5). Además, tienen un reducido tamaño y peso, que es útil para las aplicaciones más exigentes.

Figura 3. 5: Reductora planetaria

Algunas de sus características principales son:

-Pares de salida de hasta 180 N/m.

-Relaciones de reducción disponibles desde 4:1 hasta 6285:1.

-Diámetro externo de 6 - 81 mm.

-Altas prestaciones en espacios reducidos.

-Elevadas reducciones en poco espacio.

-Salida y entrada concéntrica de ejes.

Para completar el sistema del motor, solo habría que incluir el sensor intrínseco explicado en el apartado 3.2.1. El sistema completo quedaría tal y como se puede observar en la Figura 3.6.

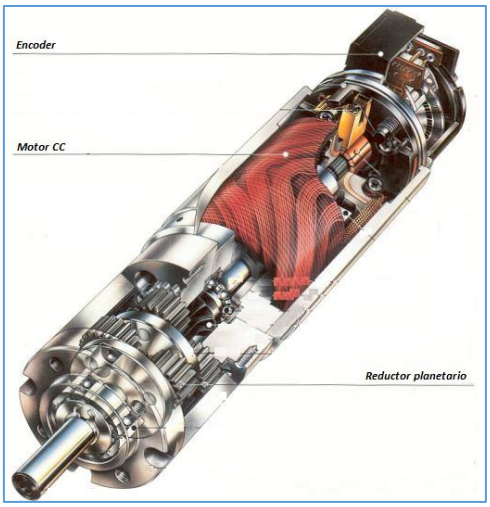

Figura 3. 6: Esquema de los elementos que componen el motor

3.2.4. Controladores

3.2.4.1. Dispositivo EPOS2

EPOS2 es un controlador de posición construido de forma modular. Gracias a que dispone de varios modos de operación, el controlador se puede adaptar de forma flexible a un amplio conjunto de accionamientos, que principalmente estarán destinados a los campos de la automatización y de la mecatrónica.

El funcionamiento de este controlador (que ya se explicó con más detalle en el apartado 2.5) está basado en la realización de un control programable combinando la retroalimentación y la prealimentación (feed forward). Esta característica permite un comportamiento ideal en el movimiento. Además, cuenta con una serie de librerías estándar para poder controlar el movimiento y conseguir una adquisición de datos para monitorizar y controlar los procesos por vía USB o CANopen, facilitando de manera considerable la integración en sistemas más amplios.

En el banco de ensayos, hay una unidad instalada y conectada a través de USB al dispositivo myRIO, con el que se controla todo el proceso a través de Labview. Esta conexión se puede observar en la Figura 3.7.

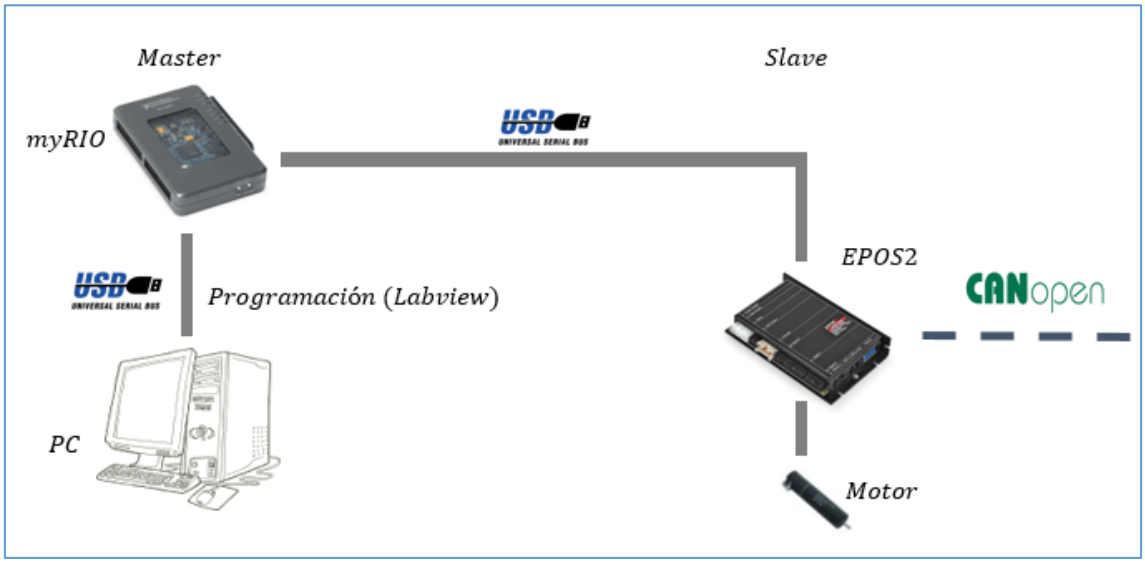

Figura 3. 7: Esquema de la red creada entre myRIO y EPOS2 a través de USB

El controlador utilizado en este proyecto ha sido el EPOS2 70/10, que es apto para motores CC de escobillas con encoder o para motores EC (Electronically Commutated) sin escobillas, con sensores Hall y encoder desde 80 hasta 700 W [25] (Tabla VI).

Tabla VI: Características de EPOS2 70/10

Datos eléctricos	**Valor**
Tensión de alimentación	11 – 70 VDC
Máxima corriente de salida	25 A
Frecuencia de conmutación	50 kHz
Velocidad de muestreo del PI = control de corriente	10 kHz
Velocidad de muestreo del PI = control de velocidad	1 kHz
Velocidad de muestreo del PID = control de posición	1 kHz
Entradas	**Valor**
Señales de los sensores	H1, H2, H3
Señales de encoder	A,A\,B,B\,I,I\ (máximo 5 MHz)
Entradas digitales	10 (7 optoacopladas, 3 diferenciales)
Entradas analógicas	2 (diferenciales) Resolución 12 bit, 0… + 5 V
CAN-ID	Configurable con interruptor DIP 1…7
Salidas	**Valor**
Salidas digitales	5 (4 optoacopladas, 1 diferencial)
Salidas analógicas	---
Voltajes de salida del encoder	+5 VDC, máximo 100 Ma
Interfaces	**Valor**
RS232	RxD; TxD (máximo 115200 bit/s)
CAN	High; low (máximo 1Mbit/s)
USB 2.0/3.0	Data+; data+- (máximo 12 Mbit/s)
Indicador	**Valor**
LED verde= READY, rojo= ERROR	LED verde, LED rojo
Datos mecánicos	**Valor**
Peso	Aproximadamente 330 g
Dimensiones (L x a x a)	150 x 93 27 mm

En la Figura 3.8 se puede observar el controlador instalado en el banco de ensayos.

Figura 3. 8: EPOS2 junto a myRIO instalado en el banco de ensayos: myRIO (izquierda) y EPOS2 70/10 (derecha)

3.2.4.2. Dispositivo myRIO

Este controlador es un dispositivo hardware embebido de National Instruments encargado de ejecutar el software de control alojado en su memoria sobre el dispositivo EPOS2 al que está conectado. Este dispositivo será explicado con más detalle en el apartado 3.4.1. En particular, se ha utilizado el modelo NI myRIO-1900, cuyas características se pueden observar en la Tabla VII.

Tabla VII: Características de myRIO-1900

Datos eléctricos	**Valor**
Tensión de alimentación	6 – 16 VDC
Máxima potencia de consumo	14 W
Potencia consumida nominal	2,6 W
Entradas	**Valor**
Entradas digitales	40 líneas E/S
Entradas analógicas	10 canales Resolución 12 bits, 0… + 5 V
Salidas	**Valor**
Salidas digitales	40 líneas E/S
Salidas analógicas	6 canales Resolución 12 bits, 0… + 5 V
Datos mecánicos	**Valor**
Peso	Aproximadamente 193 g
Memoria	**Valor**
No volátil	512 MB
DDR3	256 MB
Procesador	**Valor**
Tipo	Xilinx Z-7010
Velocidad	667 MHz
Núcleos	2

Además, en la Figura 3.8, se puede ver el myRIO instalado en el banco de ensayos, así como su conexión con el controlador EPOS2.

3.3. Sistema de control

3.3.1. Modelo del motor

Un actuador muy común en cualquier sistema de control es un motor de corriente continua, ya que proporciona directamente un movimiento de rotación. Además, junto con las ruedas o tambores y cables, puede proporcionar también el movimiento de traslación. El circuito equivalente eléctrico del inducido y el diagrama de cuerpo libre del rotor se muestra en la Figura 3.9 [26].

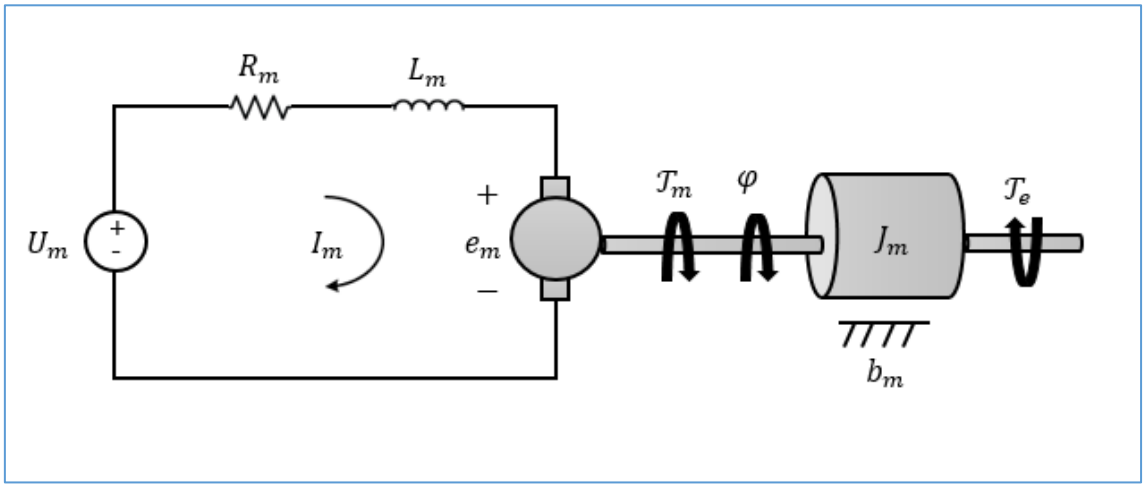

Figura 3. 9: Circuito eléctrico del motor y diagrama de cuerpo libre del rotor

En general, el par generado por un motor de corriente continua es proporcional a la corriente del inducido y a la fuerza del campo magnético. En este proyecto, se da por supuesto que el campo magnético es constante y, por lo tanto, que el par motor es proporcional solamente a la corriente del inducido Im, por un factor constante Km, como se muestra en la siguiente ecuación. Esto se conoce como un motor con la corriente del inducido controlada.

$$\mathcal{T}_m = K_m \cdot I_m \tag{2}$$

La fuerza contra-electromotriz, em, es proporcional a la velocidad angular del eje multiplicada por un factor constante Ka.

$$e_m = K_a \cdot \dot{\varphi} \tag{3}$$

A través de la Figura 3.9, se puede deducir las siguientes ecuaciones que gobiernan el sistema, basadas en la segunda ley de Newton y en las leyes de voltaje de Kirchhoff.

$$J_m\ddot{\varphi} + b_m\dot{\varphi} = \mathcal{T}_m + \mathcal{T}_e \tag{4}$$

$$U_m = R_m I_m + L_m \dot{I_m} + K_a\dot{\varphi} \tag{5}$$

Para obtener el diagrama de bloques del sistema, se ha aplicado la transformada de Laplace, y se han obtenido las siguientes ecuaciones en función de la variable s.

$$I_m = \frac{U_m - K_a\dot{\varphi}}{R_m + L_m s} \tag{6}$$

$$\dot{\varphi} = \frac{\mathcal{T}_m - \mathcal{T}_e}{J_m s + b_m} \tag{7}$$

A partir de estas ecuaciones, en la Figura 3.10 se implementa el siguiente diagrama de bloques del sistema representado en la Figura 3.9.

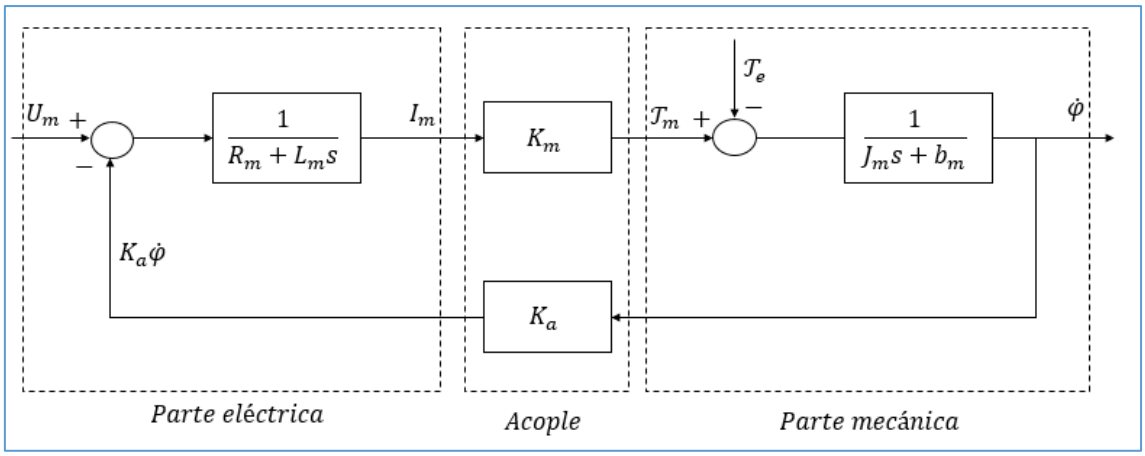

Figura 3. 10: Diagrama de bloques del sistema del motor

3.3.2. Control en cascada para el motor de corriente continua

Un lazo de control clásico necesita un regulador dispuesto como se indica en la Figura 3.11, donde se controla la velocidad del eje del motor. Para controlar una variable de salida de un sistema, en este caso la velocidad del motor $\dot{\varphi}$, se debe actuar sobre las variables de entrada que se puedan manipular, que en el caso del motor de corriente continua, sería la tensión de entrada U_m. El control de $\dot{\varphi}$ se verá además afectado por las perturbaciones que entran en el sistema, en este caso el par de carga $\mathcal{T}_e$. Debido a que se quiere controlar con precisión el sistema y además evitando el efecto de las perturbaciones, el regulador más conveniente a utilizar sería un PI [27].

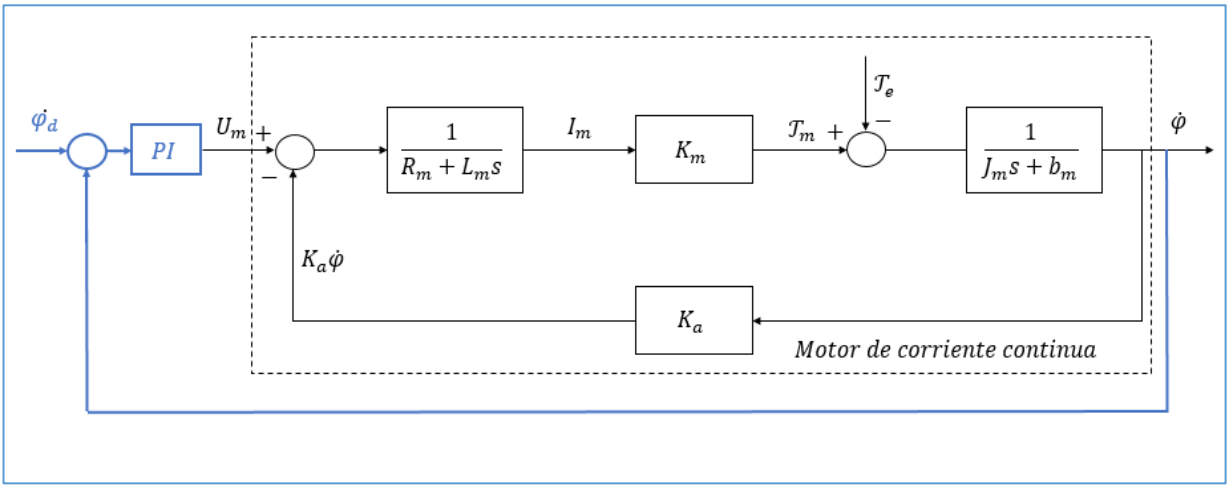

Figura 3. 11: Control mono-variable del motor de corriente continua

La utilización de reguladores con acción diferencial no es muy aconsejable en aplicaciones que incluyan motores, ya que este tipo de sistemas tienen el problema del ruido inherente, y podrían llevar al mismo a la inestabilidad debido a las reacciones bruscas de la acción diferencial del regulador. Además, la sintonización de este tipo de regulador no es nada sencilla, y ofrece unas prestaciones dinámicas bajas.

Frente a esto, el esquema más utilizado para el control preciso de motores es el control en cascada. El control en cascada se puede aplicar en sistemas en los que haya variables internas (de estado) y con dinámica bien diferenciada. En el caso del sistema propuesto en este proyecto, con el motor eléctrico, la parte eléctrica es mucho más rápida que la parte mecánica. Hablar de mayor rapidez es indicativo de una mayor facilidad para variar la magnitud de una variable.

Si se quiere realizar un control de la corriente que se genera en el motor, el problema se traduciría en manipular directamente la tensión de entrada, y por tanto, la posibilidad de tener un control sencillo de la corriente no sería una hipótesis, sino que es inmediata, como se muestra en la Figura 3.12.

Al hacer un control de corriente, a su vez se está haciendo un control de par, ya que son directamente proporcionales con constante de proporcionalidad K_m, tal y como se puede observar en la Ecuación (2). Debido a que se considera que el control de la corriente está aislado, sin tener en cuenta otras variables del motor, se puede estimar que la fuerza contraelectromotriz del motor e_m $(K_a \cdot \dot{\varphi})$, es una perturbación de entrada para este subsistema.

Teniendo en cuenta estas condiciones, el regulador de corriente (C_C) más adecuado sería del tipo PI, ya que este regulador elimina tanto el error de posición como el efecto de las perturbaciones en régimen permanente.

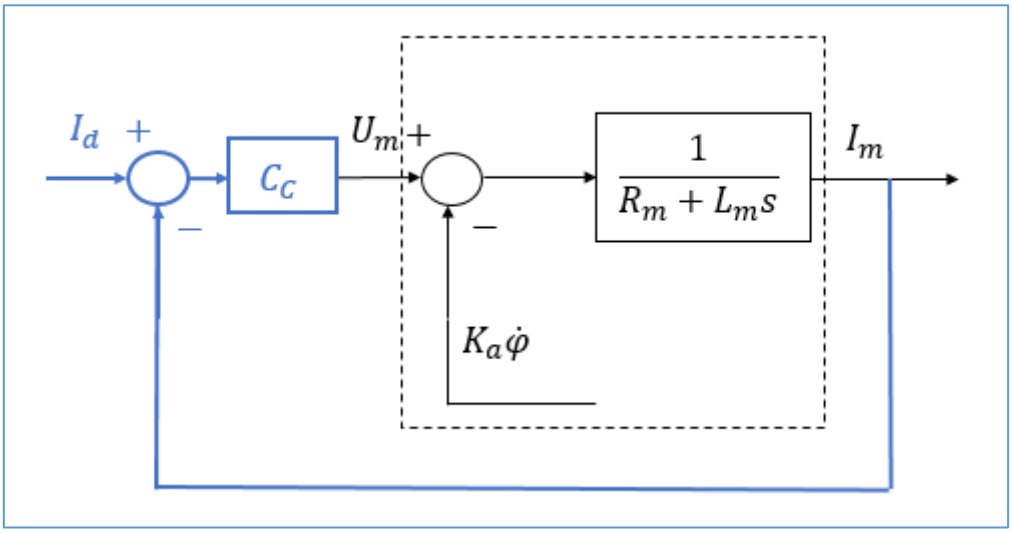

Figura 3. 12: Control de corriente

Una vez realizado el lazo de control de la corriente explicado anteriormente, se ha llegado a la conclusión de que sería posible manipular directamente el par eléctrico $\mathcal{T}_m$ a través de la constante de proporcionalidad K_m, y no solo la tensión de entrada, por lo que el control de la velocidad se vuelve extremadamente sencillo, ya que su esquema estaría reducido al control de un sistema de primer orden, como se muestra en la Figura 3.13.

Debido a que el sistema a controlar es de tipo 0, será necesario que el regulador de velocidad (C_V) sea de tipo PI para poder anular el error de posición en régimen permanente. Además, este sistema se ve afectado por una perturbación de entrada, el par de carga $\mathcal{T}_e$, por lo tanto, como se ha comentado antes, este regulador PI también es el indicado para el rechazo de este tipo de perturbaciones en régimen permanente.

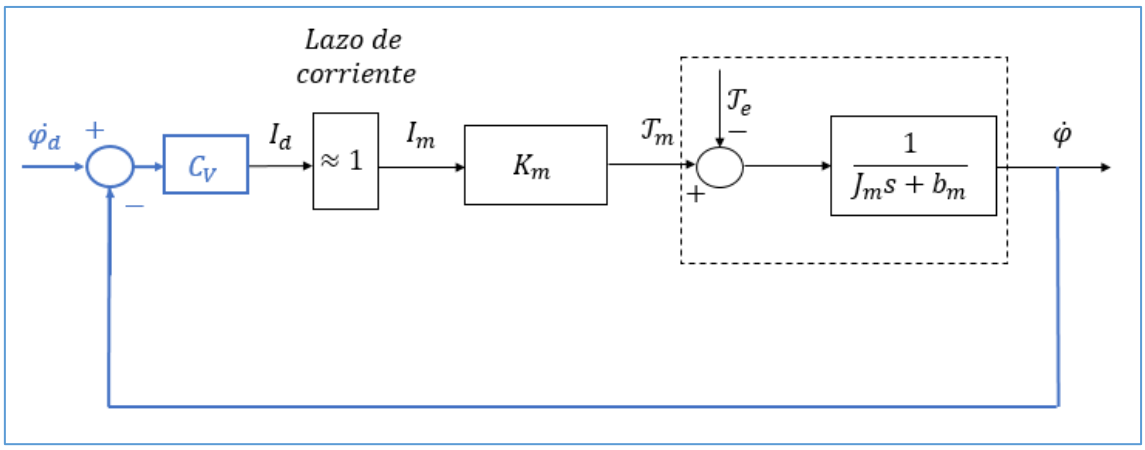

Figura 3. 13: Control de velocidad

El lazo de corriente se considera como una constante igual 1 en la Figura 3.13, debido a que si se sintoniza el bucle de control de la corriente con un ancho de banda muy superior (>4) al deseable para el control de velocidad, se puede suponer que $I_d \approx I_m$, desde el punto de vista del lazo de velocidad.

Una vez obtenido el control de velocidad, el control de posición se convierte en algo muy sencillo, ya que el sistema a controlar es un sistema de tipo I no expuesto a perturbaciones de carga. Por tanto, el diseño del regulador de posición (C_P) podría ser simplemente de tipo proporcional. El esquema de control de la posición del motor se puede observar en la Figura 3.14.

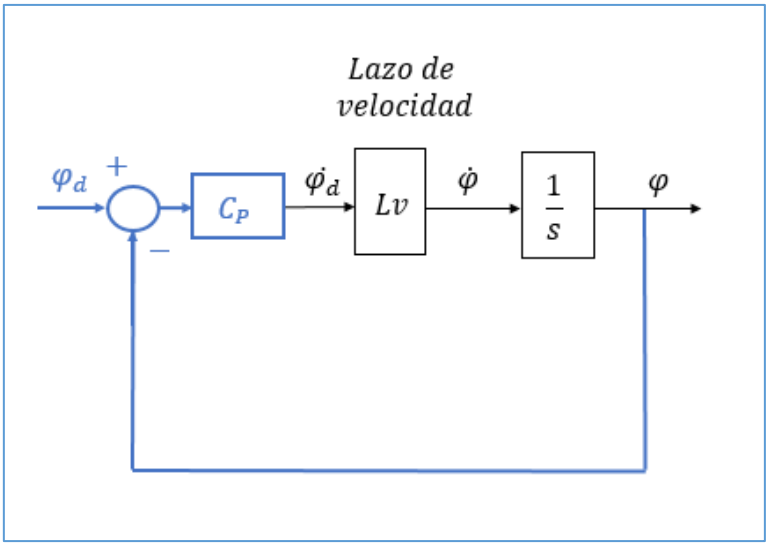

Figura 3. 14: Control de posición

3.3.3. Modelo elástico

Los manipuladores robóticos están generalmente diseñados de tal manera que el acoplamiento entre el sistema de accionamiento (motor y reductora) y los enlaces, podría ser tan rígido como sea posible. Sin embargo, las articulaciones utilizadas en la robótica incorporan un componente elástico entre el actuador y el enlace (Figura 3.15). De ese modo, la articulación puede ser considerada como un sistema que contiene dos masas acopladas, una de ellas se forma debido a la inercia reflejada del rotor en la salida del sistema de transmisión y la otra, está relacionada con la inercia equivalente del enlace. El movimiento relativo entre ambas masas produce una deformación elástica, con una rigidez que es fija en el caso de las SEA, o variable en el caso de los VSA [9].

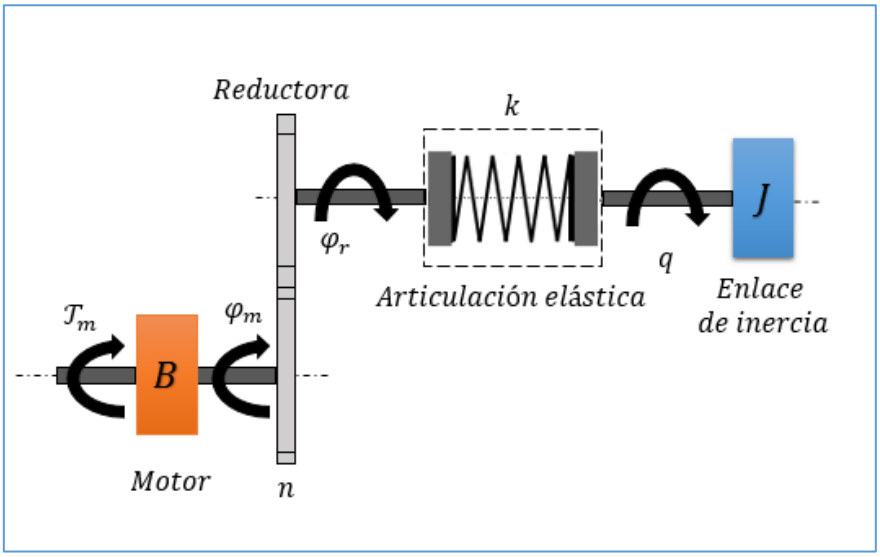

Figura 3. 15: Esquema de una articulación robótica elástica

El mecanismo propuesto en la Figura 3.15 se trata de una articulación robótica con una rigidez en las articulaciones que podría ser considerada constante en algunos tramos. Se compone de un conjunto de resortes lineales y un sistema de transmisión que permiten transformar el movimiento de la articulación en un movimiento lineal.

El sistema dispone de dos tipos de resortes lineales, que definen la función de la rigidez de las articulaciones a través de su rigidez y su precarga. Es un mecanismo pasivo diseñado para mejorar el rendimiento y la seguridad de los brazos robóticos, especialmente de aquellos cuya finalidad es compartir su espacio de trabajo con otros robots o seres humanos.

En la Figura 3.16 se muestra el par externo requerido para equilibrar el DSM (Double Spring Mechanism, cuya traducción es Mecanismo de doble resorte), de acuerdo con la deflexión angular.

Como se puede observar, se distinguen tres estados diferentes de trabajo, el estado inicial se mantiene mientras que el par no es superior a $\mathcal{T}_{th}$, la precarga del muelle principal evita un desplazamiento importante de la rotura (alta rigidez).

Se deja de mantener el primer estado una vez se supera $\mathcal{T}_{th}$, y en este punto, sólo el resorte principal se comprime. Mientras que el par externo no supere a $\mathcal{T}_{th2}$, se mantendrá este estado de trabajo.

Cuando el par externo supera $\mathcal{T}_{th2}$, se inicia el siguiente estado de trabajo, donde el resorte secundario se incorpora al sistema y elige un entorno en el que se cumple que k1 (del primer muelle) > k2 (del segundo muelle), lo que produce una mayor compresión del resorte secundario en comparación con el principal, mientras que aumenta el par (baja rigidez). Además, en la Figura 3.16 también se señalan otros puntos de interés tales como βmin, βmax y βth, que permiten describir el rango de trabajo de estos estados.

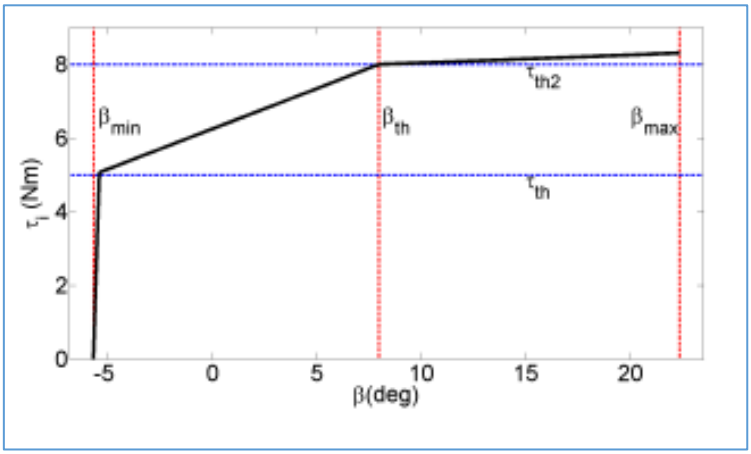

Figura 3. 16: Grafica de par externo vs deflexión angular del DSM

3.3.4. Control de fuerza

En la figura 3.17 se muestra un esquema del control de par elástico, a partir del control de posición que se consigue en la Figura 3.14. Como ya había sido considerado en el apartado 3.3.2, es posible manipular directamente la velocidad del motor y la posición a través de la misma, e incluso también se puede hacer mínima su dinámica gracias a la buena elección de los reguladores tanto en el lazo de corriente como en el de velocidad. Por lo tanto, el control del par elástico se reduce solamente a controlar la deflexión angular de la articulación expuesta en la Figura 3.15.

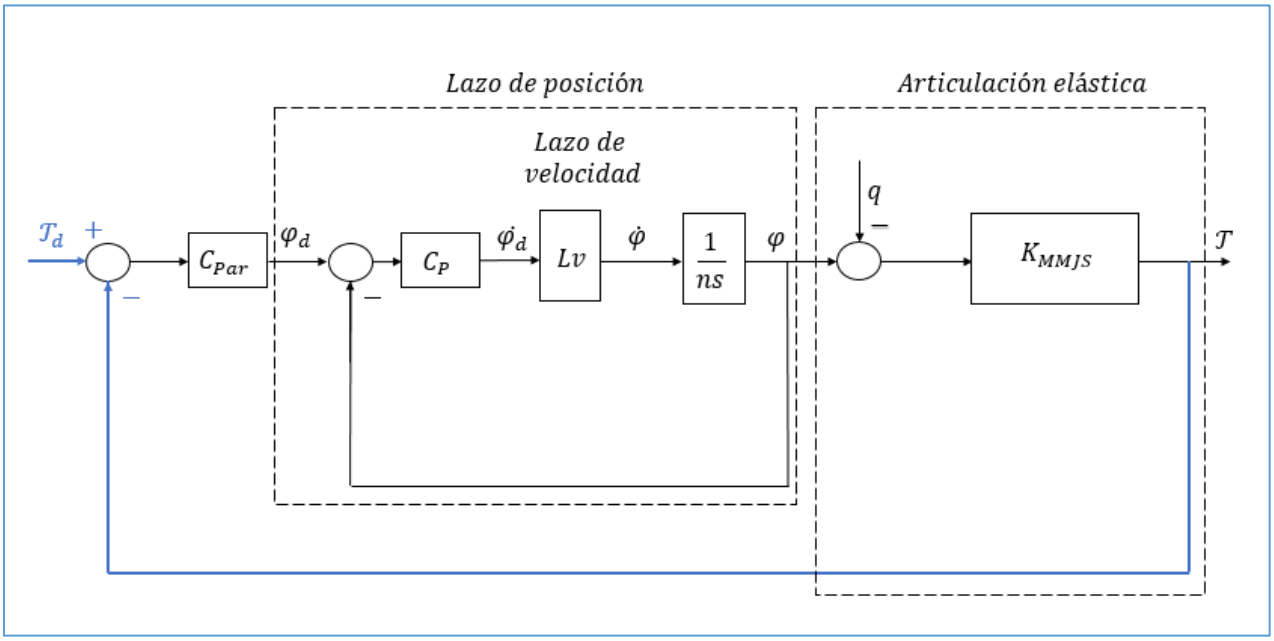

Figura 3. 17: Control de par elástico

El sistema representado en la Figura 3.17 es un sistema de tipo I, dentro de un lazo de fuerza, y está expuesto a una perturbación q que está producida debido al movimiento del eslabón de salida. La constante K_{MMJS} representa el par elástico con respecto a la deflexión angular de la articulación elástica, mientras que C_{par} es el regulador necesario para controlar el lazo de par.

En las simulaciones que se hicieron para determinar el ancho de banda teórico del sistema, se diseñó el regulador C_{par} como un regulador de tipo PI con un integrador lento, y el eslabón de salida se ha fijado rígidamente para conseguir que la perturbación de entrada q no realice efecto en el sistema.

En la Figura 3.18a se muestran las simulaciones que se realizaron aplicando varios escalones de par elástico para estimar el ancho de banda del sistema. El sistema representado muestra un tiempo máximo para alcanzar la señal deseada de 37 ms. En la Figura 3.18b se muestra una representación del diagrama de bode de las diferentes funciones de transferencia dependiendo del escalón que se aplicaba (las simulaciones han sido realizadas por el grupo VSA en el laboratorio de robótica).

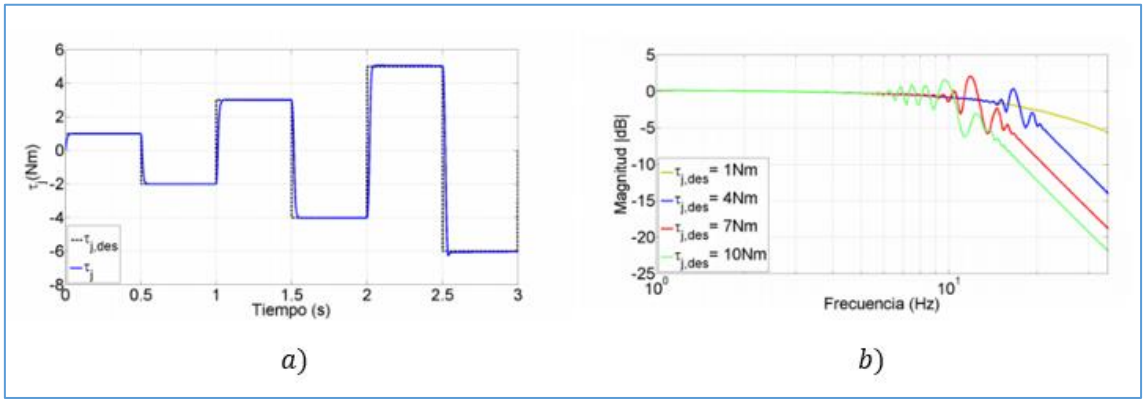

Figura 3. 18: Control de par articular con eslabón de salida fijo: a) Respuesta ante escalones y b) estimación de la función de transferencia para diferentes escalones

La rigidez articular equivalente conocida como K_{MMJS} no influye en el ancho de banda, ya que una disminución en la rigidez articular se compensa con un aumento en la ganancia de control, por lo que el ancho de banda podría mantenerse constante sin que esté condicionado por esta constante.

Para el caso particular del dispositivo MMJS, el ancho de banda máximo que se ha conseguido es de 22 Hz para el caso del escalón de 1 Nm, y según aumenta el valor del escalón, disminuye el ancho de banda, por lo que se pone de manifiesto que la alta flexibilidad, cuando las amplitudes son mayores, hace que el ancho de banda caiga notablemente debido a efectos de saturación en el motor, además de efectos no lineales que están presentes en el mecanismo y que no se tienen en cuenta. Sin embargo, el rendimiento del MMJS se puede comparar con los

mejores valores dentro de los actuadores elásticos serie, por lo que los resultados son bastante satisfactorios.

En el apartado 5.5 se explicarán las pruebas experimentales que han sido realizadas para poder compararlas con los resultados obtenidos aquí, y así poder determinar si de verdad los resultados experimentales son los esperados, o sin embargo, las estimaciones previas no eran del todo correctas.

3.4. Descripción del sistema NI MyRIO

3.4.1. Definición

National Instruments myRIO es un microcontrolador integrado con Xilinx Zynq Z-7010, que dispone de una arquitectura de multiprocesador reconfigurable. Sus dos procesadores tienen su propia memoria, además de periféricos, y ambos pueden ser programados de manera independiente. En la Figura 3.19 se puede ver el dispositivo myRIO embebido.

Figura 3. 19: Dispositivo NI myRIO

3.4.2. Arquitectura

Por una parte, el dispositivo myRIO tiene un procesador fijo, que es un ARM Cortex-A9 MPCore. Este procesador de doble núcleo implementa la arquitectura del conjunto de instrucciones ARMv7 (ISA) e incluye un conjunto fijo de periféricos. Está preconfigurado en la fábrica con una distribución de Linux con extensiones en tiempo real. Linux es un popular sistema operativo en tiempo real (RTOS) que tiene una programación y comportamiento más determinista que el sistema Windows, y se utiliza en una amplia gama de aplicaciones embebidas [28].

Por otra parte, dispone de un procesador reconfigurable, que es el Xilinx Artix-7 FPGA. La FPGA se compone de unidades de lógica, memoria y otros bloques de construcción fundamentales que pueden ser reconfigurados a nivel de hardware. Un FPGA puede implementar periféricos de hardware tales como buses de comunicación, generadores PWM, interfaces de encoders, algoritmos de procesamiento de señales, procesamiento de vídeo y decodificación, e incluso otras arquitecturas de procesamiento.

La FPGA de myRIO tiene una parte configurada de forma fija, conocida como personalidad fija, es decir, una configuración de un FPGA que está destinado a ser distribuido sin modificaciones. Esta personalidad fija es un archivo binario compilado, conocido como archivo de bits (bitfile), que puede ser distribuido sin el código fuente. La personalidad fija de la FPGA que se incluye con myRIO implementa una serie de periféricos, como son la UART, I2C, PWM y otros muchos, además de las entradas y salidas digitales y analógicas.

En la Figura 3.20 se puede observar la arquitectura de myRIO, así como los diferentes periféricos de los que está compuesto.

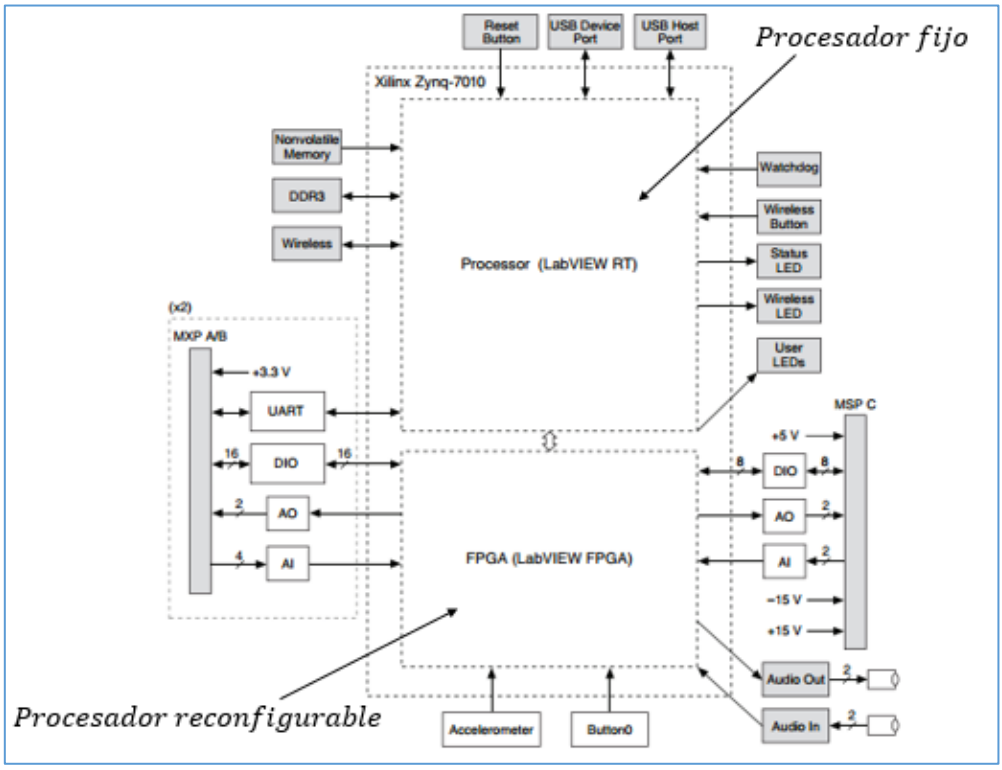

Figura 3. 20: Arquitectura del dispositivo NI myRIO [29]

3.4.3. Software soportado

Tanto el procesador fijo como el reconfigurable del dispositivo myRIO normalmente son programados a través de Labview (cuyo funcionamiento ya fue explicado en el apartado 2.4),

ya que este programa dispone de módulos que permiten trabajar en tiempo real, y dispone también de librerías específicas para myRIO, por lo que hace muy fácil su programación.

Además, la personalidad fija de la FPGA que viene por defecto en myRIO viene programada con Labview FPGA, que es un módulo que permite interactuar directamente con la FPGA desde el programa, y liberada como código abierto.

Sin embargo, myRIO no solo puede ser programado a través de Labview, ya que LabVIEW FPGA también puede generar código en C, por lo que se podría importar este código al programa llamado Eclipse, y de esta forma conseguiríamos interactuar directamente con la FPGA utilizando lenguaje C.

4. DESCRIPCIÓN DEL SOFTWARE

4.1. Introducción

Para poder realizar las pruebas experimentales necesarias para evaluar el comportamiento del dispositivo MMJS, era necesario realizar un control a varios niveles del mismo. En este capítulo se explicarán cada una de las estrategias de control que se han utilizado, así como las diferentes librerías y generadores de señales usados para conseguir que el control y las pruebas realizadas sean los correctos.

Debido a que se necesita no solo realizar un esquema de control del dispositivo, sino que también realizar pruebas y medidas, el mejor programa para conseguir este objetivo era Labview, ya que de esta forma se combina el control a bajo y alto nivel en un entorno gráfico, y además se realiza a la misma vez un programa de control del dispositivo y una interfaz gráfica para el usuario que quiera manejarlo. Esto se cumple gracias a que Labview dispone de dos partes en sus programas:

- Panel frontal: esta parte del programa es la interfaz con el usuario. Con esta parte el usuario podrá interactuar con el programa a la vez que se está ejecutando, y será capaz de ver los resultados obtenidos en tiempo real. Además podrá modificar los controles (señales de consigna, ganancias de los reguladores, etc.) y podrá visualizar los indicadores (salidas del sistema, gráficas con los resultados, etc.).

- Diagrama de bloques: en esta parte es donde se realiza la implementación del programa de control, con una serie de bloques interconectados entre sí y que realizan una función específica.

Gracias a esta estructura en los programas de Labview, se consigue a la vez construir el esquema de control del dispositivo MMJS y la interfaz con el usuario para el manejo del mismo, y así poder realizar las pruebas de forma simple.

4.2. Requerimientos

Debido a que el banco de ensayos dispone de varios elementos que servirán de ayuda para el control en todos los niveles del dispositivo MMJS, se decidió dividir el diseño del software en varios niveles, que son los siguientes:

- Control del motor principal: en este punto, el objetivo es que el actuador responda de forma a correcta a las diferentes configuraciones usadas, ya sea control de corriente, de velocidad o de posición. Para ello, se utiliza del dispositivo NI myRIO, que actúa como el controlador maestro y será el encargado de configurar las comunicaciones, al controlador esclavo (EPOS2 70/10), los parámetros de los motores y la configuración de los diferentes modos de operación. Una vez configurado el controlador esclavo, éste será el encargado de enviar las órdenes al motor, y de esta forma se controla el

motor a través de un sistema embebido de bajo coste, que era el objetivo a cumplir. Esta parte del control se explicará en el apartado 4.5.

- Sensores: a través del encoder extrínseco disponible, se es capaz de conocer la deflexión angular del actuador, por lo que gracias a esto, es posible realizar un control de fuerza (par elástico) gracias a la constante Kmmjs, con la que se pasa de deflexión angular a par elástico, como ya se explicó anteriormente. Con este control de fuerza, se cierra el último lazo de control en el que se trabaja en este trabajo de fin de grado. Esta parte del control se explicará en el apartado 4.6.

- Adquisición de datos: para poder realizar la adquisición de datos, se ha implementado un programa dentro del módulo de FPGA de myRIO, por lo que se están tomando datos en tiempo real, y por lo tanto, se podrán analizar los resultados con garantías. En los siguientes apartados se explicará con detalle la lógica utilizada para conseguir esta adquisición, y que de esta forma el lector comprenda mejor el programa. Esta parte del control se explicará en el apartado 4.6.

- Gestión de errores: el usuario que interactúe con los programas de control, será capaz de visualizar los errores que se produzcan en el mismo para poder actuar ante ellos, y de esta forma ser capaz de solucionarlos de una manera rápida y simple.

4.3. Librerías

Una librería es un fichero que contiene un conjunto de funciones, escritas en un lenguaje de programación y que puede ser utilizada por un programa. De esta forma, si un programa necesita usar cualquiera de las funcionalidades disponibles en la librería, puede acceder a la misma sin tener que reescribir el código en el programa [30].

Maxon proporciona una librería con extensión DLL (Dynamic Link Library) para poder comunicarse con el controlador EPOS2 a través de Labview. Este tipo de librería se basa en un archivo que dispone de código ejecutable que solo puede ser cargado bajo el sistema operativo de Windows. También proporciona la misma librería, pero con extensión SO (Shared Object), que es igual a la librería DLL, pero que solo puede ser cargada bajo el sistema operativo de Linux.

Como ya se explicó con anterioridad, el dispositivo myRIO dispone de su propio sistema operativo Linux, por lo que solo se puede trabajar con la librería SO que proporciona Maxon. Sin embargo, no es posible introducir la librería en la memoria del dispositivo desde un entorno Windows, por lo que se debe utilizar otra estrategia para poder usar estas librerías desde myRIO.

El programa Eclipse, permite conectar el myRIO como un elemento externo, y una vez que está conectado, permite abrir un terminal para interactuar con la consola como en cualquier máquina que disponga de un sistema operativo Linux. Gracias a esta característica explicada, a través de una serie de instrucciones a seguir, detalladas en [31], se consigue introducir la

librería de Maxon con extensión SO en la memoria del dispositivo, y que de esta forma, esta librería sea reconocida por el mismo.

Una vez que se ha introducido la librería dentro del myRIO, el siguiente paso es conseguir llamar a la funciones de la misma desde Labview. Dentro del programa Labview, solo se encuentran disponibles las funciones de la librería DLL antes citada, por lo que la única forma de utilizarlas desde el myRIO, es modificándolas. Para ello, es necesario indicar que el lugar de donde provienen no es la librería DLL, sino que es la librería SO, por lo que solo se debe abrir cada función de la librería que se vaya a utilizar para el proyecto, y modificar el lugar del que proviene (path) (Figura 4.1). Dentro del programa Labview, es tan sencillo como indicarle a la función que llama a la librería, que el "path in" (destino de la librería) es libEposCmd.so (Figura 4.2). De esta forma, la librería ya no buscará la función dentro de la DLL, sino que la buscará dentro de la extensión SO, y como ya está introducida en la memoria del myRIO, será reconocida, y por lo tanto, se podrá usar perfectamente.

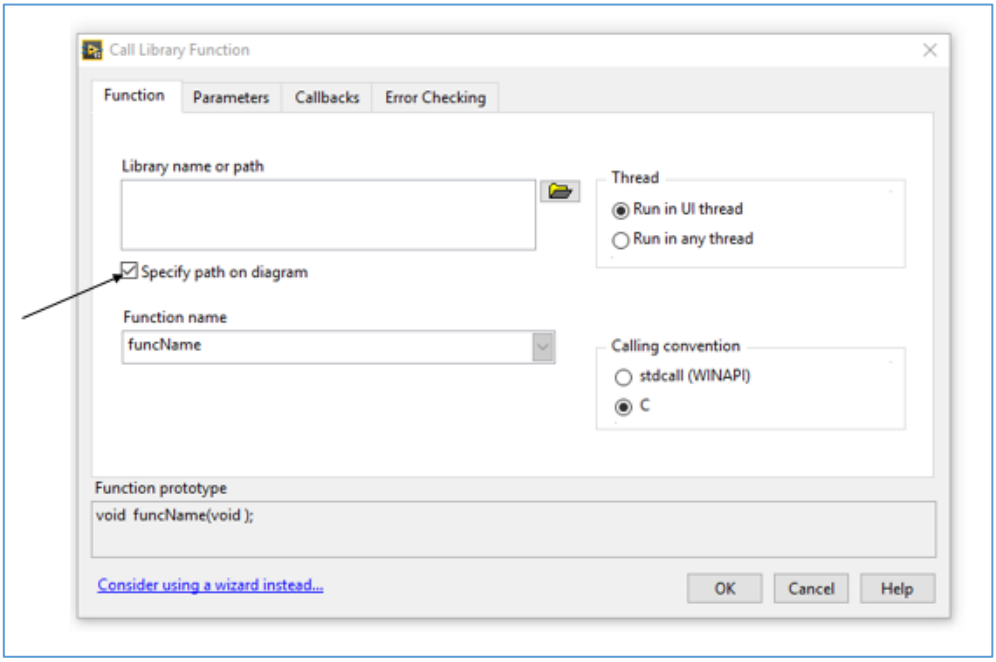

Figura 4. 1: Señalar que quieres introducir externamente el valor del path

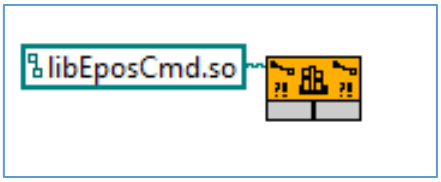

Figura 4. 2: Indicar la librería .so para que funcione en myRIO

A continuación, se explicará de forma detallada cada una de las funciones que han sido modificadas para poder utilizarlas desde el dispositivo myRIO, dividiéndolas según la función que realiza en el control del motor.

4.3.1. Funciones de configuración

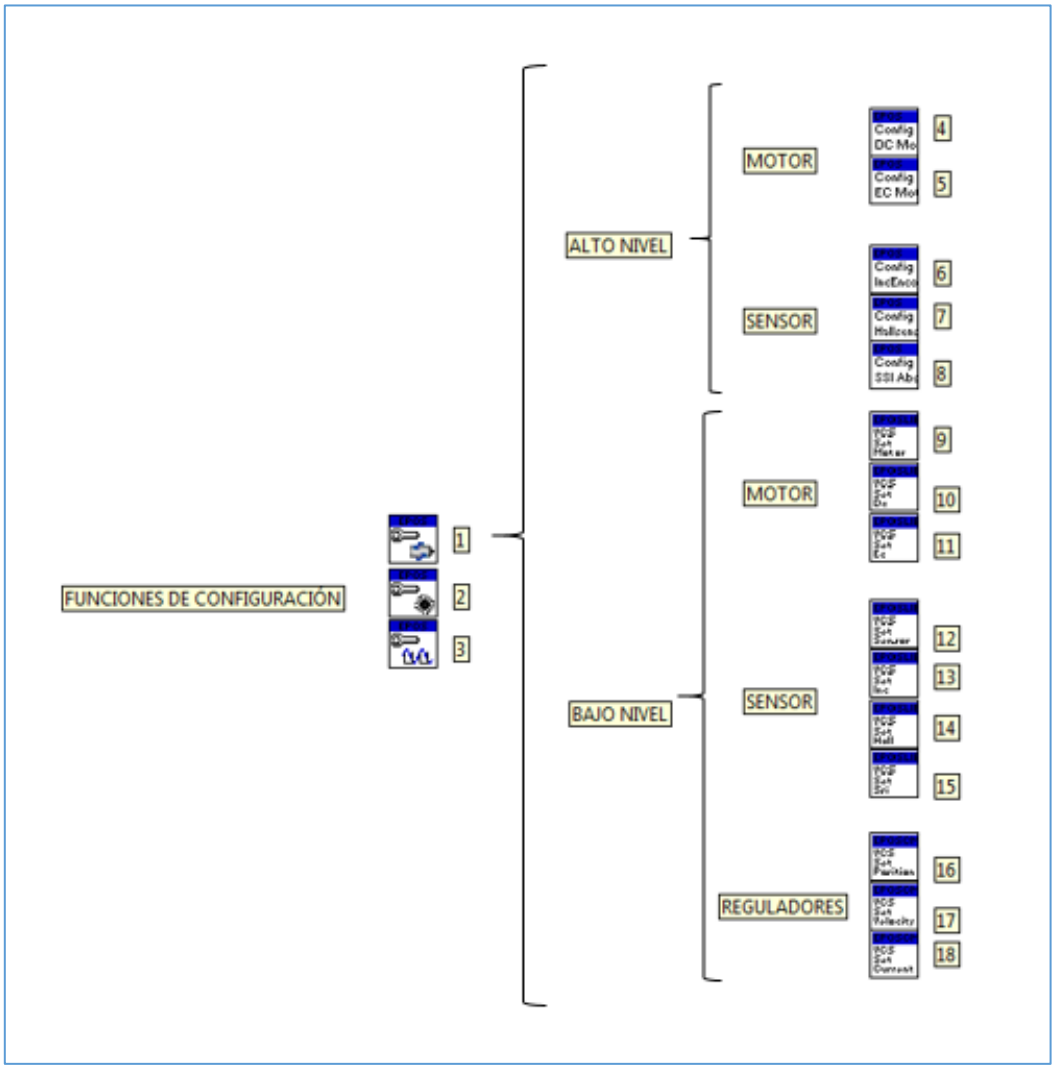

Figura 4. 3: Funciones de configuración de la librería de Maxon

Como se puede observar en la Figura 4.3, se han modificado 18 funciones de configuración, cuyo funcionamiento es el siguiente:

- Funciones generales: el subsistema 1, es conocido como “ConfigureMotorType”, y sirve para indicarle al controlador el tipo de motor que se va a utilizar. El subsistema 2, es conocido como “ConfigureSensorType”, y su función es la de configurar el tipo de sensor interno que tiene disponible el motor. El subsistema 3, es llamado

"ConfigureRegulators", y es utilizado para configurar las diferentes ganancias de los reguladores utilizados en el sistema de control.

- Funciones de alto nivel: el subsistema 4, es conocido como "ConfigureDcMotorType", y sirve para configurar las características del motor en el caso de que el motor utilizado sea de tipo DC. El subsistema 5, es conocido como "ConfigureEcMotorType", y tiene la misma función que la anterior, con la única diferencia de que se utiliza cuando el motor es de tipo EC. El subsistema 6, llamado "ConfigureIncEncoderSensorType", se utiliza para configurar las características del encoder incremental. Los subsistemas 7 y 8 tienen la misma función que el subsistema 6, pero cada uno configura el tipo de sensor que le corresponde. Sus nombres son "ConfigureHallsensorSensorType" y "ConfigureSsiAbsEncoderSensorType" respectivamente.

- Funciones de bajo nivel: el subsistema 9, llamado "VCS Set Motor Type", se utiliza para indicarle al controlador el tipo de motor que se usa en la aplicación. El subsistema 10, llamado "VCS Set Dc Motor Parameter", es utilizado para indicarle al controlador los parámetros característicos del motor de tipo DC. El subsistema 11, llamado "VCS Set Ec Motor Parameter", tiene la misma función que el subsistema 10, pero para los motores de tipo Ec. El subsistema 12, conocido como "VCS Set Sensor Type", tiene la misma lógica que el subsistema 9, pero para el sensor interno del conjunto del motor. El subsistema 13, que se llama "VCS Set Inc Encoder Parameter", tiene el mismo funcionamiento que el subsistema 10, pero en el caso del encoder incremental. Los subsistemas 14 y 15, al igual que el subsistema 13, tienen la misma lógica de funcionamiento, pero cada uno con el tipo de sensor correspondiente. Sus nombres, respectivamente, son "VCS Set Hall Sensor Parameter" y "VCS Set Ssi Abs Encoder Parameter". Por último, los subsistemas 16, 17 y 18, se utilizan para indicarle las ganancias de posición, velocidad y corriente, respectivamente, de los reguladores utilizados en el sistema. Sus nombres, por orden numérico, son "VCS Set Position Regulator Gain", "VCS Set Velocity Regulator Gain" y VCS Set Current Regulator Gain".

4.3.2. Funciones de acción/estado

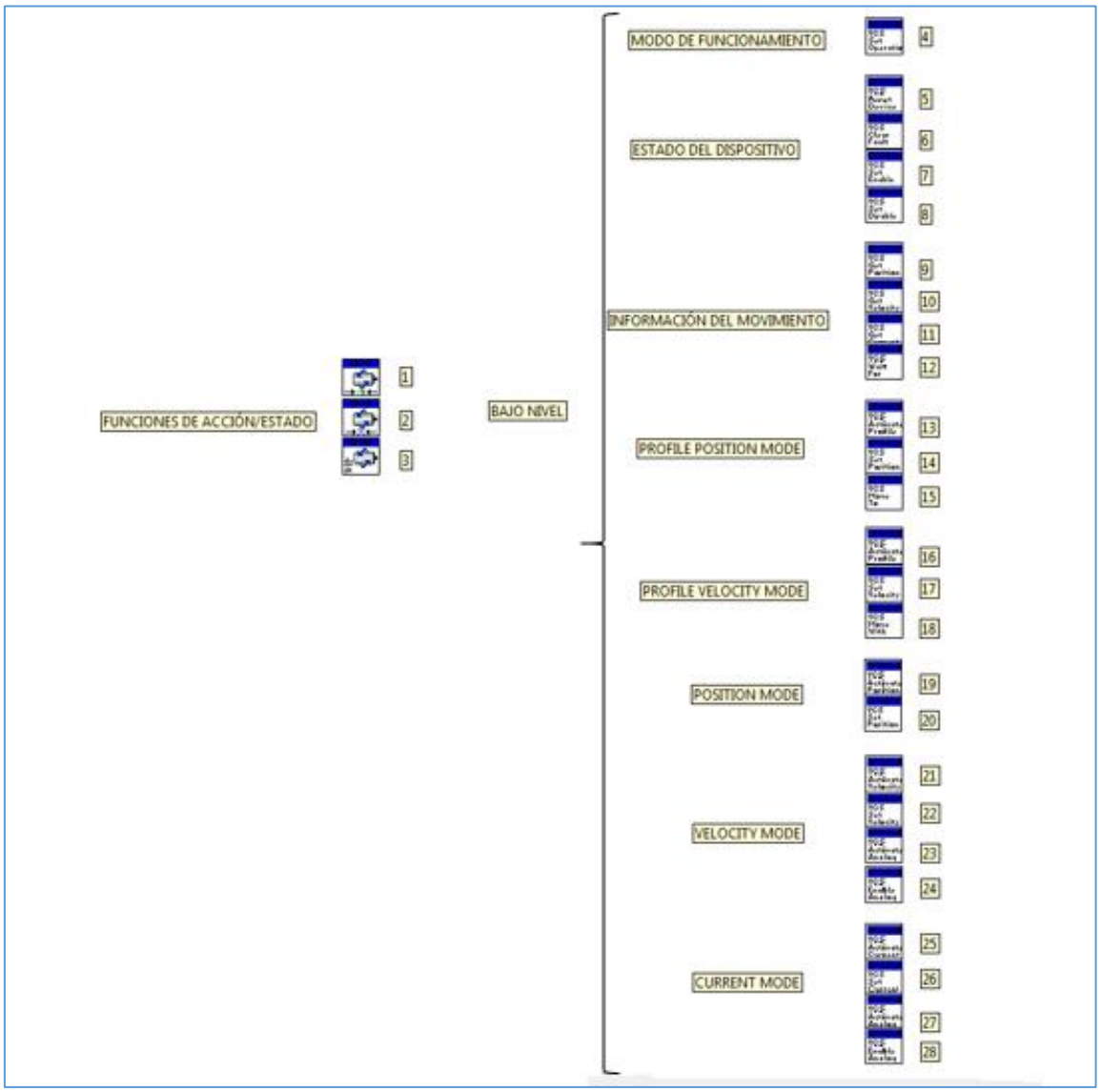

Figura 4. 4: Funciones de acción/estado de la librería de Maxon

Como se puede ver en la Figura 4.4, se han modificado 27 funciones de acción/estado, cuyo funcionamiento se explica a continuación:

- Funciones generales: dentro de las funciones generales de acción/estado, se encuentran los tres primeros subsistemas. El subsistema 1, llamado “EnableAxis”, se utiliza para habilitar los ejes y así conseguir que el motor pueda moverse. El subsistema 2, conocido como “DisableAxis” es utilizado para deshabilitar los ejes una vez se termina el programa, para evitar que se queden bloqueados. El subsistema 3, llamado “MovewithVelocity”, se utiliza dentro del modo de operación Profile Velocity Mode para conseguir que el motor se mueva con la velocidad de consigna.

- Funciones de bajo nivel: el subsistema 4, llamado “VCS Set Operation Mode”, se utiliza para indicarle al controlador el modo de operación en el que debe trabajar el motor. El subsistema 5, llamado “VCS Reset Device”, es usado para reiniciar el dispositivo EPOS2. El subsistema 6, llamado “VCS Clear Fault” se utiliza para limpiar al dispositivo de errores y deshabilitarlo. El subsistema 7, que se conoce como “VCS Set Enable State”, sirve cambiar el estado del dispositivo a “habilitado”. Por otra parte, el subsistema 8, llamado “VCS Set Disable State” cambia el estado del dispositivo a “deshabilitado”. Los subsistemas 9, 10 y 11 devuelven la posición, velocidad y corriente real del motor, respectivamente, y son llamados “VCS Get Posicion Is”, “VCS Get Velocity Is” y “VCS Get Current Is”. El subsistema 12, llamado “VCS Wait For Target Reached” espera hasta que el estado del dispositivo es el deseado o hasta que se agota un tiempo indicado por el operador.

  Dentro de las funciones de bajo nivel, existen funciones específicas para el modo de operación conocido como “Profile Position Mode”, que son las siguientes: el subsistema 13, conocido como “VCS Activate Profile Position Mode”, sirve para activar este modo de operación. El subsistema 14, llamado “VCS Set Position Profile”, sirve para indicarle al controlador el perfil de posición que debe de seguir el motor. El subsistema 15, llamado “VCS Move to Position”, sirve para que el motor se mueva a la posición deseada con el perfil que antes indicado.

  Por otra parte, los subsistemas 16, 17 y 18 realizan las mismas funciones que los subsistemas anteriores pero en el modo de operación “Profile Velocity Mode”, y cuyos nombres son, respectivamente, “VCS Activate Profile Velocity Mode”, “VCS Set Velocity Profile” y “VCS Move with Velocity”.

  Al igual que para estos dos modos de operación, existen las mimas funciones para los modos de operación “Position Mode”, “Velocity Mode” y “Current Mode”, donde siempre hay una función que activa el modo de operación, y otra que introduce la consigna a la que debe funcionar el motor. Para el modo de posición, están disponibles las funciones 19 y 20, cuyos nombres son “VCS Activate Position Mode” y “VCS Set Position Must”. Para el modo de velocidad, se encuentran las funciones 21 y 22, cuyos nombres son “VCS Activate Velocity Mode” y “VCS Set Velocity Must” y para el modo de corriente, se dispone de las funciones 25 y 26, cuyos nombres son “VCS Activate Current Mode” y “VCS Set Current Must”.

  Para terminar, dentro de los modos de velocidad y corriente, existe la opción de introducir una consigna externa al programa, y para ello se utilizan, en el modo de velocidad, las funciones 23 y 24, cuyos nombres son “VCS Activate Analog Velocity Setpoint”, que activa el modo de operación de velocidad pero con una consigna externa, y “VCS Enable Analog Velocity Setpoint”, que habilita una máscara que permite que funcione este modo de operación. Al igual que en el modo de velocidad, sucede en el modo de corriente, donde las funciones utilizadas para el mismo objetivo son los subsistemas 25 y 26, cuyos nombres son “VCS Activate Analog Current Setpoint” y “VCS Enable Analog Current Setpoint”.

4.3.3. Funciones de comunicación

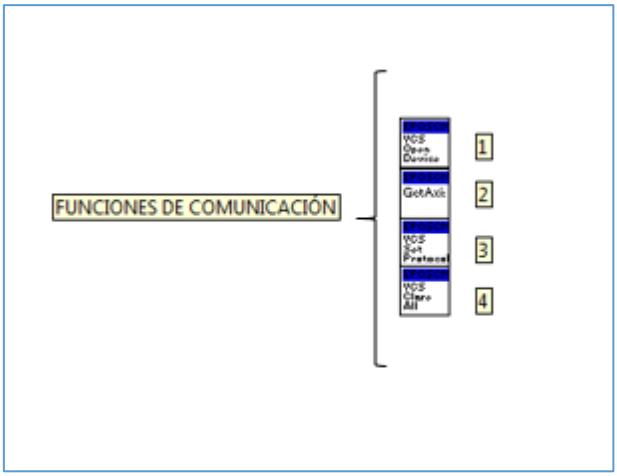

Figura 4. 5: Funciones de comunicación de la librería de Maxon

En la Figura 4.5 se puede observar las funciones de comunicación modificadas para el sistema propuesto, que son las siguientes:

- El subsistema 1, conocido como "VCS Open Device", abre el dispositivo EPOS2 y permite que se empiece a interaccionar con él.

- El subsistema 2, conocido como "Get Axis", se utiliza para obtener el estado de los ejes en cualquier momento de la ejecución.

- El subsistema 3, llamado "VCS Set Protocol Stock Settings", sirve para seleccionar la velocidad a la que se transmiten los datos y para indicar el tiempo de espera.

- El subsistema 4, llamado "VCS Close All Devices" sirve para cerrar todos los dispositivos que estén abiertos. Se suele utilizar para terminar el programa que se esté ejecutando.

4.3.4. Funciones de alto nivel

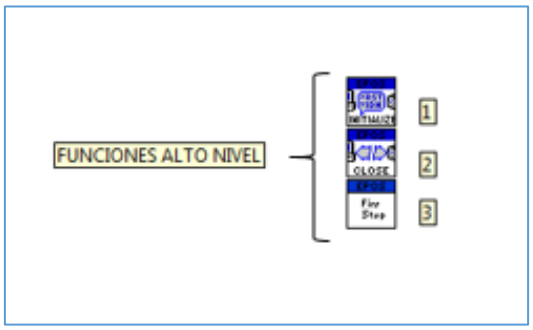

Figura 4. 6: Funciones de alto nivel de la librería de Maxon

En la Figura 4.6, se pueden observar las funciones de más alto nivel utilizadas en los programas. El subsistema 1, conocido como "Initialize", es usado para inicializar todos los aspectos referentes al dispositivo EPOS2. Se abre el dispositivo, indicándole al mismo todos los aspectos de la comunicación, se le indica el modo de operación en el que se va a trabajar, y se limpia el dispositivo de cualquier error anterior que pueda tener. En la Figura 4.7 se puede observar su funcionamiento con más detalle.

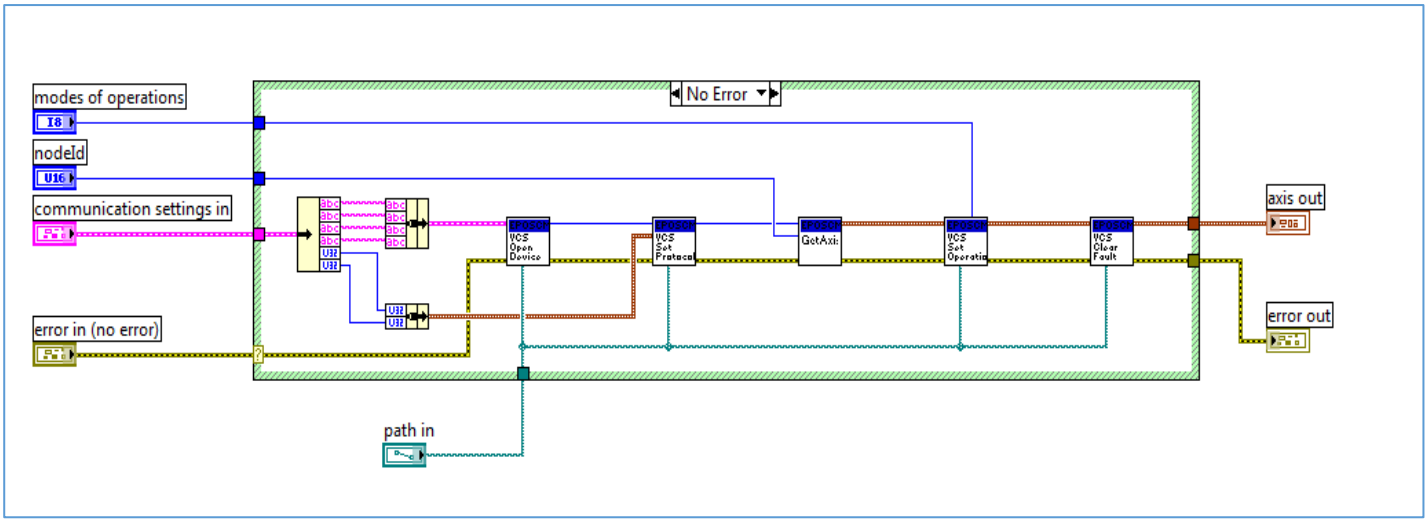

Figura 4. 7: Función "Initialize" de la librería de Maxon

El subsistema 2, es una función de más alto nivel que la que explicada anteriormente llamada "VCS Close All Devices", pero cuya función es exactamente la misma que la mencionada aquí.

Por último, el subsistema 3, conocido como "First Step", es el primer paso en cada uno de los programas que se explicarán más adelante en este mismo capítulo, y lo que se hace en él es, por orden, inicializar el dispositivo, reiniciarlo para evitar que compilaciones anteriores afecten

al programa, indicar al controlador todos los parámetros característicos del motor utilizado, hacer lo mismo para el sensor interno y por último, configurar los diferentes reguladores de posición, velocidad y corriente utilizados. En la Figura 4.8, se puede observar el diagrama de bloques con detalle.

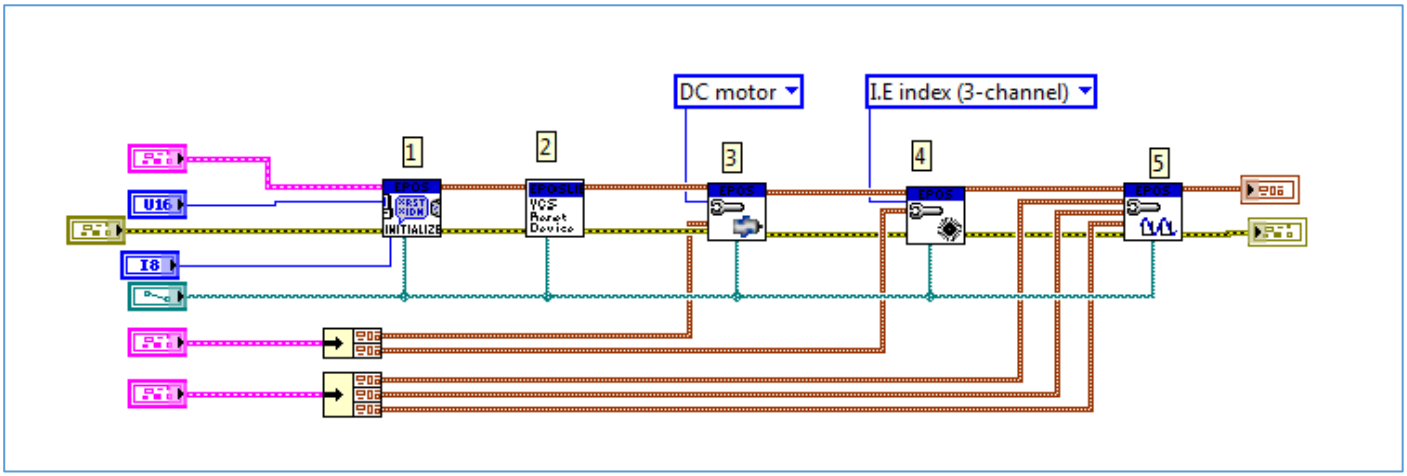

Figura 4. 8: Función "First Step" utilizada en todos los programas de control

4.4. Generadores de señales

4.4.1. Alto nivel

Se consideran generadores de señal de alto nivel a los que han sido diseñados desde la CPU del dispositivo NI myRIO. En este aspecto, se ha diseñado un solo generador de señales, pero que, a través de un selector, es posible seleccionar que tipo de señal se desea utilizar para la consigna, teniendo en el mismo generador la posibilidad de crear una señal constante, sinusoidal, sinusoidal con frecuencia variable y cuadrada. A continuación, se explicará con más detalle cada uno de estos tipos de señal.

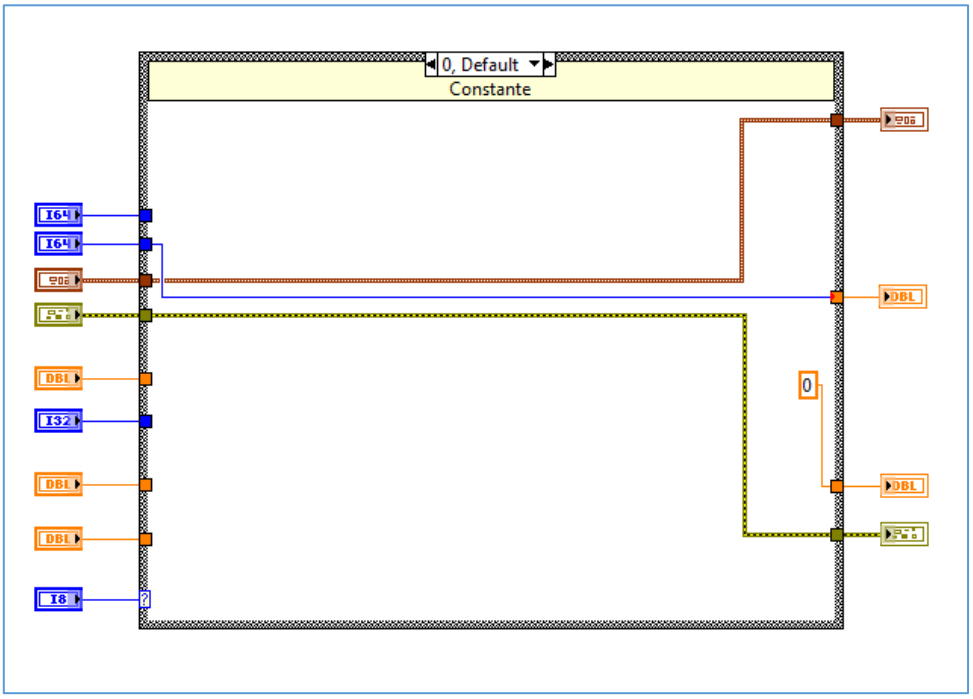

Figura 4. 9: Señal constante

Como se puede observar en la Figura 4.9, si se selecciona que la señal de consigna generada sea una constante, directamente el valor de amplitud indicado por el operador al generador, será su salida, por lo que siempre se mantendría ese valor constante.

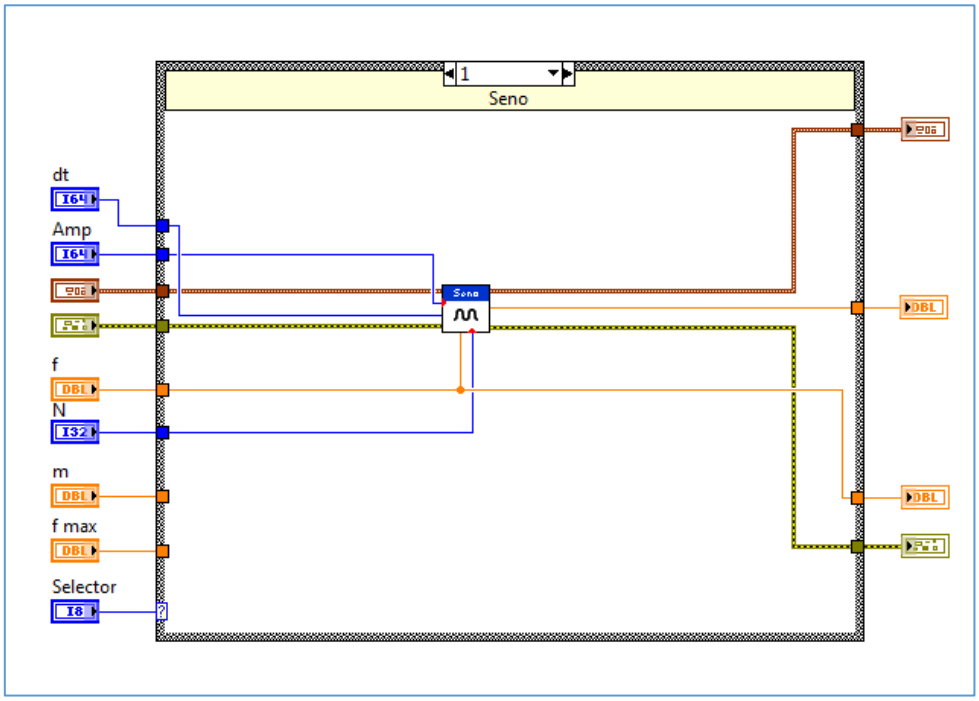

Figura 4. 10: Señal sinusoidal

En la Figura 4.10, se puede observar el diagrama de bloques del generador cuando se selecciona la señal sinusoidal. Al bloque llamado Seno, se le introducen los valores de dt (tiempo de muestreo), Amp (Amplitud), f (frecuencia del seno) y N (número de muestras). El valor de la salida del generador será el seno con la amplitud indicada en la variable Amp, y con la frecuencia indicada en la variable f. El funcionamiento de este bloque se puede observar en la Figura 4.11.

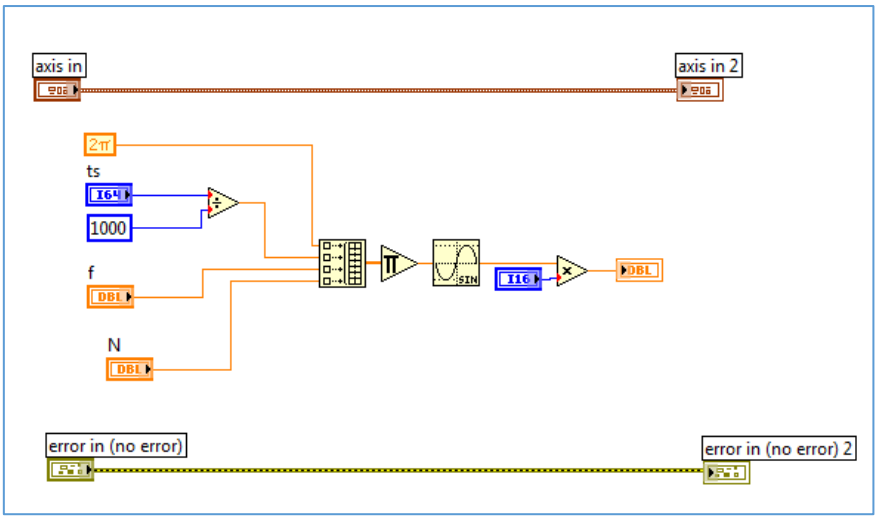

Figura 4. 11: Funcionamiento del bloque llamado "Seno"

En la Figura 4.12, se puede observar el diagrama de bloques del generador para el caso en el que la señal generada sea una sinusoidal con frecuencia variable. El diagrama es igual que en el caso de la señal sinusoidal, lo único que cambia, es que se añade el bloque que hace que la frecuencia sea variable, llamado "Modula frecuencia", por lo tanto, en lugar de introducirle valor de la variable f al bloque "Seno", se introduce la salida del bloque "Modula frecuencia".

En la Figura 4.13, se puede observar el diagrama de bloques del subsistema "Modula frecuencia", para poder entender cómo se consigue que la frecuencia vaya creciendo dependiendo del valor de la variable m.

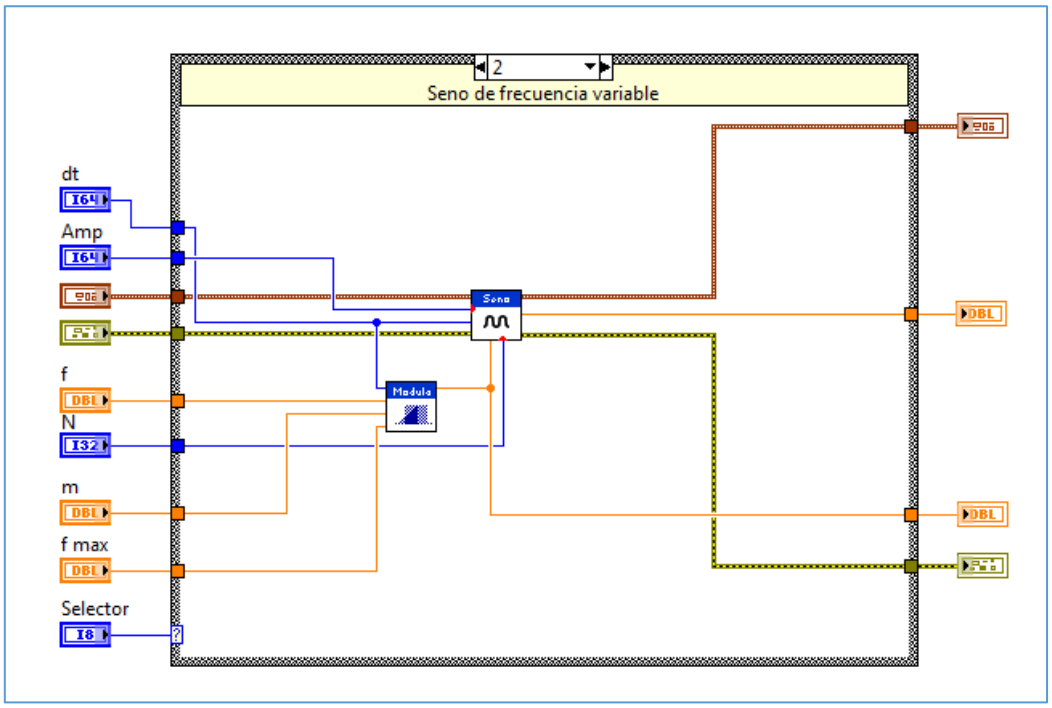

Figura 4. 12: Señal sinusoidal con frecuencia variable

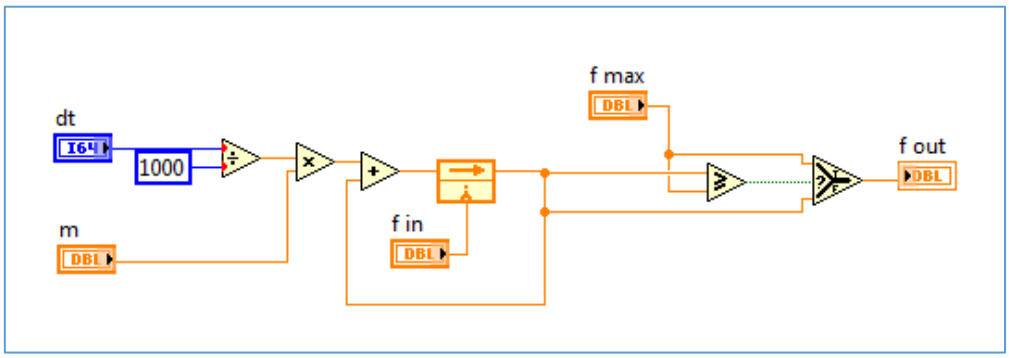

Figura 4. 13: Funcionamiento del bloque llamado "Modula frecuencia"

En la Figura 4.13, se puede observar que la frecuencia irá creciendo con una pendiente igual a "m" mientras que el valor de f sea menor que la frecuencia máxima. Cuando el valor de la frecuencia sea igual al de la frecuencia máxima, el valor de ésta se mantendrá constante.

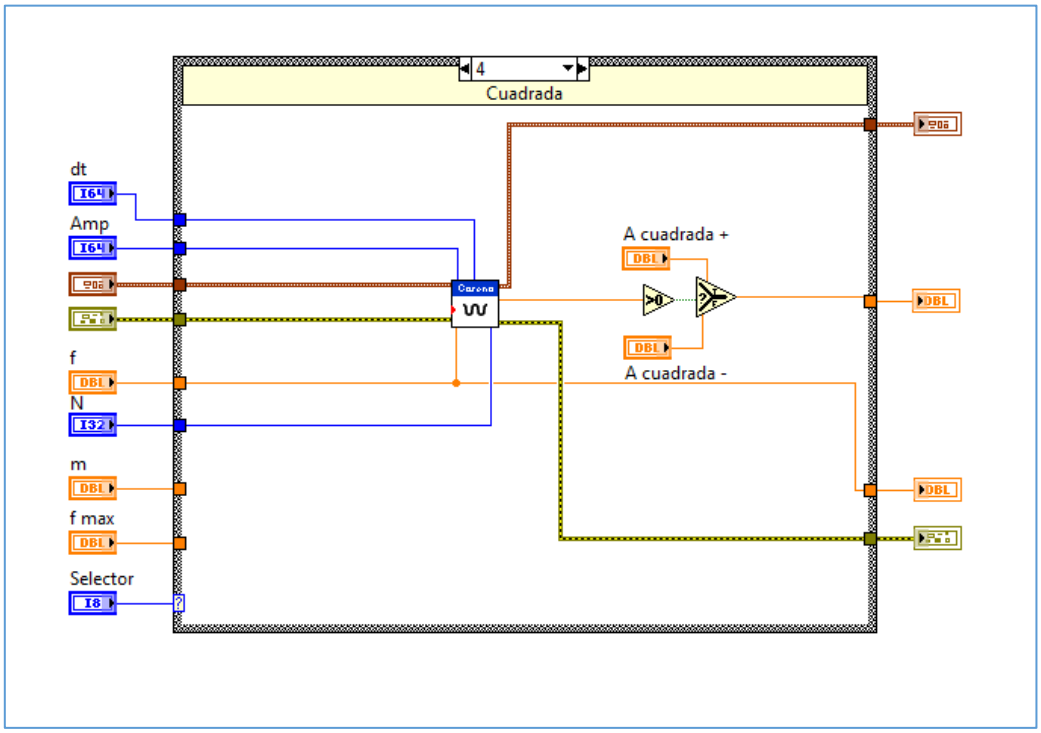

Figura 4. 14: Señal cuadrada

En la Figura 4.14, se puede observar el diagrama de bloques del generador en el caso de que la señal generada sea cuadrada. Con los parámetros introducidos en el bloque llamado "Coseno", se genera un coseno con la amplitud igual a la variable Amp. El valor de salida del bloque llamado "Coseno" se compara con cero. Cuando el valor del coseno es mayor que 0, el valor de la señal será igual al valor de la Amplitud positiva, y cuando el valor del coseno es menor que 0, el valor de la señal será igual al valor de la Amplitud negativa. De esta forma, se consigue generar una señal cuadrada.

4.4.2. Bajo nivel

Se consideran generadores de bajo nivel a aquellos donde la señal se genera a través de la FPGA de myRIO, convirtiendo a la CPU en un instrumento para modificar las variables internas de los programas que trabajan en la FPGA. De esta forma, se consigue que la señal se genere en tiempo real, y la CPU del dispositivo myRIO es liberada de realizar el trabajo de la generación de la señal, consiguiendo así mayores rendimientos.

Dentro de este tipo de generadores, se ha diseñado un programa llamado “Seno fpga.VI”, con el que se genera un seno que puede variar entre 0 y 5 V, para introducir su salida por una salida analógica del dispositivo myRIO, y así pueda llegar esta señal a una entrada analógica del controlador EPOS2 70/10. Este programa será el utilizado para generar la señal de consigna en los programas “Corriente con consigna externa” y “Velocidad con consigna externa”, que se explicarán más adelante. En la Figura 4.15, se puede observar la parte del programa que trabaja sobre la FPGA del dispositivo myRIO.

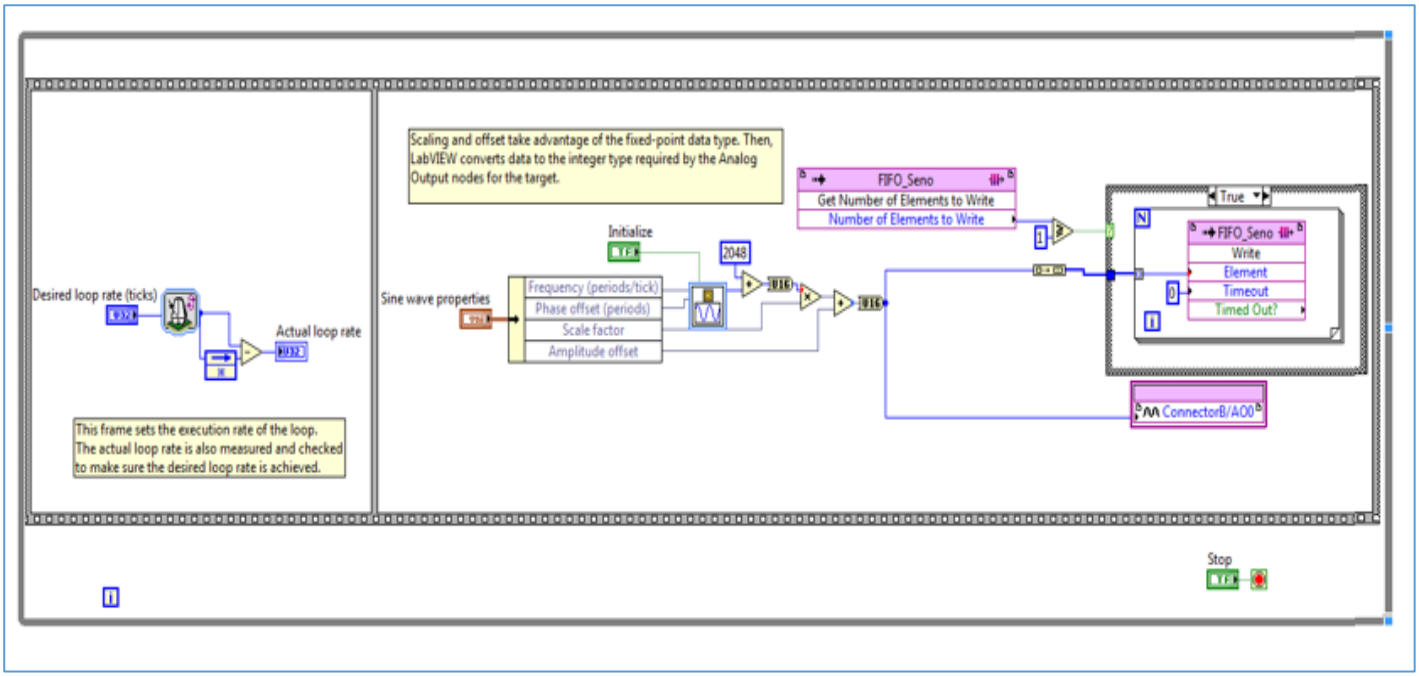

Figura 4. 15: Señal sinusoidal entre 0 y 5 V generada en la FPGA del dispositivo myRIO

Como se puede observar en la Figura 4.15, la señal de salida es almacenada en una FIFO (First In First Out), y además se introduce en la salida analógica AO0, como ya se ha explicado anteriormente. La variable “Sine wave properties” es la variable que será modificada desde la CPU de myRIO para generar el seno según los objetivos de la aplicación. De esta forma, la CPU de myRIO solo está encargada de la modificación de esta variable, y es liberada del trabajo de generar la señal y de la introducción de la misma en una salida analógica (aspecto importante, ya que la carga de trabajo sería demasiada para la CPU, y no conseguiríamos trabajar en tiempo real). En el primer subdiagrama de la estructura de secuencia, se marca el tiempo de muestreo con el que funcionará el programa. En las Figura 4.16, 4.17, 4.18 y 4.19 se puede observar la parte del programa de generación de la señal diseñada en la CPU.

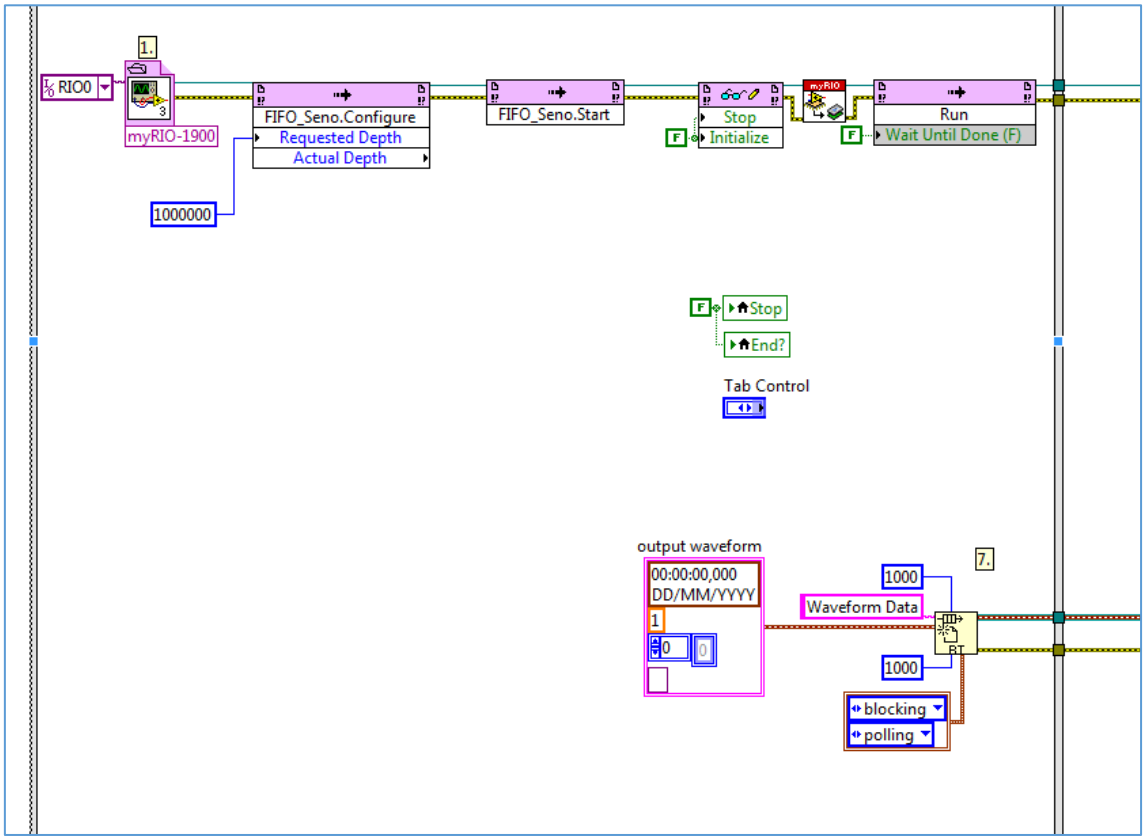

Figura 4. 16: Zona de inicialización del programa de la señal sinusoidal en la CPU

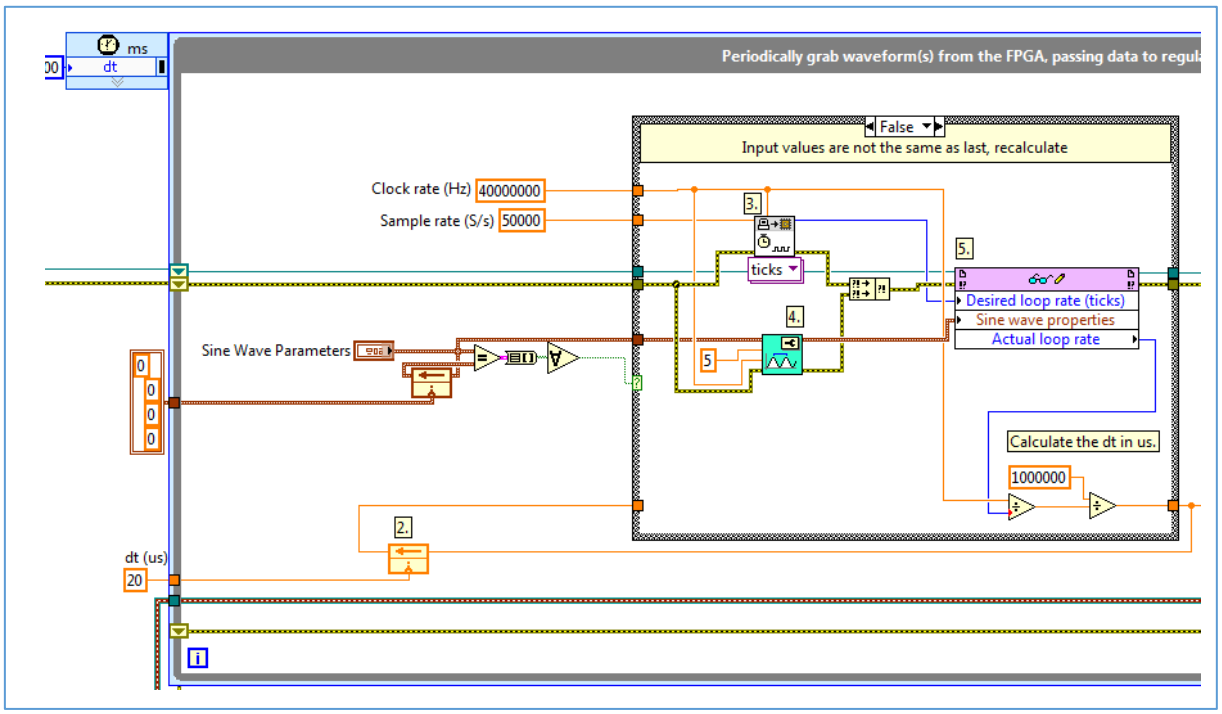

Figura 4. 17: Zona de asignación de las propiedades de la señal sinusoidal

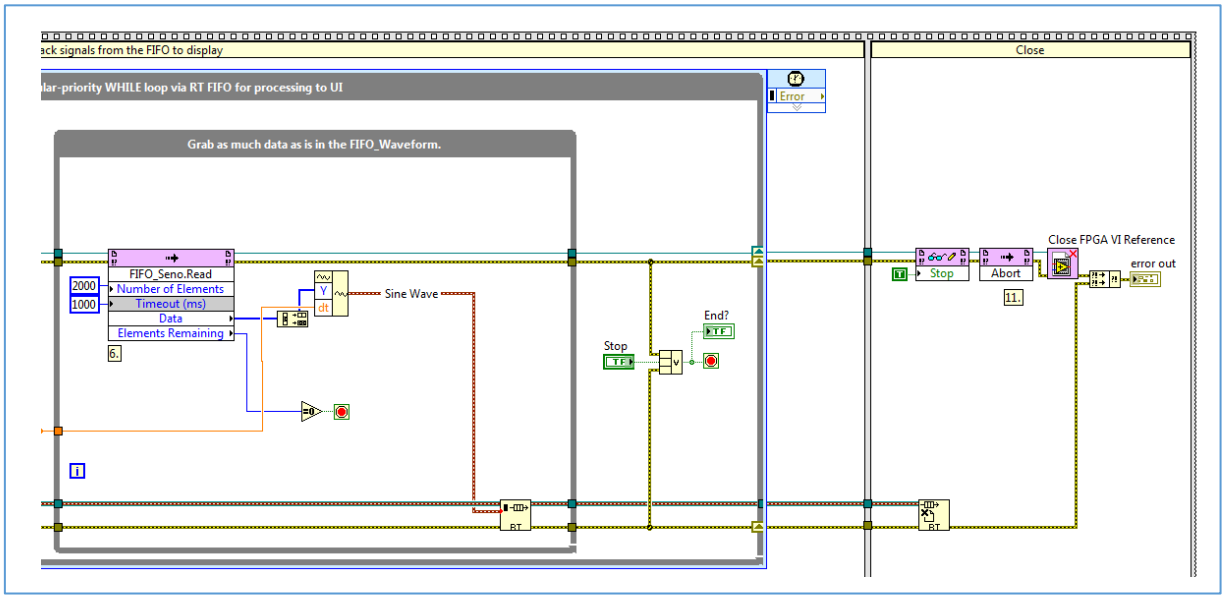

Figura 4. 18: Zona de lectura de la FIFO y guardado de datos para graficarlos

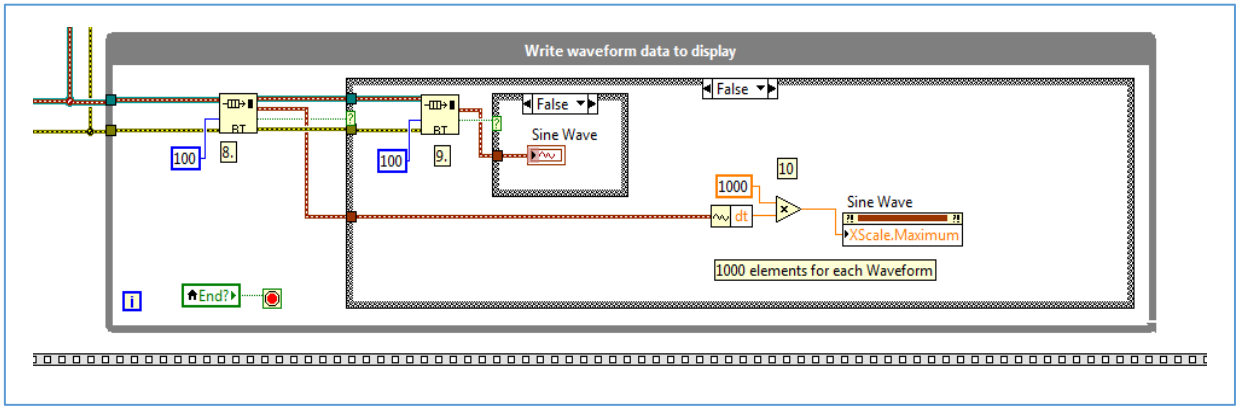

Figura 4. 19: Zona de construcción de la gráfica

En la Figura 4.16, se puede ver la zona de inicialización del programa. En ella se abre la referencia al programa explicado en la Figura 4.15, para así poder trabajar con las variables que él utiliza, y además poder ejecutarlo en un segundo plano. Además de esto, se inicia la FIFO donde se guardan los datos del seno, para que comience así el guardado de datos. Los valores de stop y reset se inicializan a 0, y se crea el bitfile. También se configuran de forma paralela los parámetros que servirán para la construcción de la gráfica. Por último, se ejecuta el programa utilizando el método "Run" de la paleta de funciones de la FPGA.

En la Figura 4.17, se modifica la variable antes citada "Sine wave properties", para generar el seno con la amplitud, frecuencia y offset sean necesarios para el sistema. El subsistema 4, es el encargado de indicar al generador que la señal de salida tendrá una amplitud de entre 0 y 5 V,

para que ésta salga por la salida analógica de forma correcta (la salida analógica solo permite señales de entre 0 y 5 V).

En la Figura 4.18, se puede observar cómo se leen los datos de salida del seno generado, para guardarlos en otra FIFO y así poder graficarlos después, para que el usuario tenga una mejor visión de que la señal que genera es la correcta. Además, en la Figura 4.18 también se puede ver el cierre del programa, modificando el valor de stop a 1, y cerrando la referencia con el programa de la FPGA, para que éste deje de ejecutarse en segundo plano.

Por último, en la Figura 4.19, se leen los datos guardados antes en la FIFO, y se grafican los resultados de la señal generada.

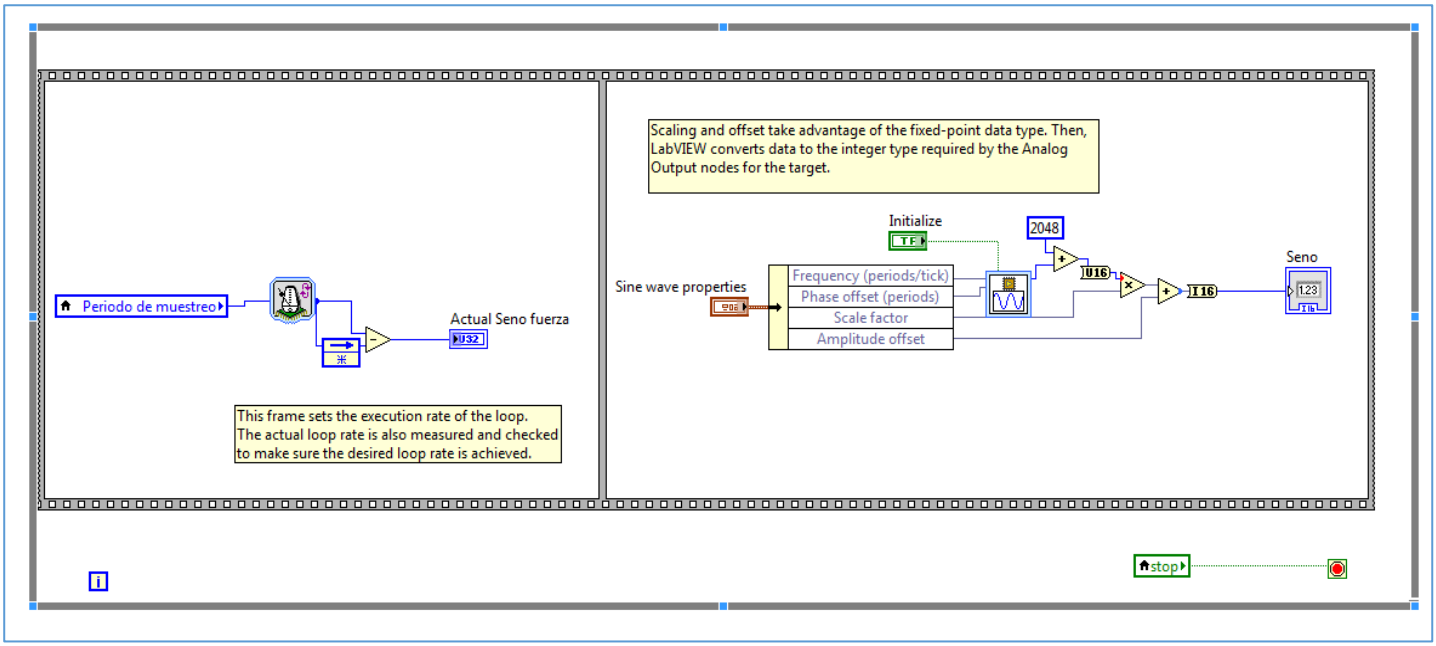

Figura 4. 20: Señal sinusoidal de par elástico de la FPGA

Al igual que se ha explicado la generación de la señal sinusoidal que se utilizará en los programas de corriente y velocidad externos, también se genera una señal sinusoidal que servirá para el control de fuerza, como se puede observar en la Figura 4.20 (este diagrama se encuentra dentro del programa que descansa en la FPGA llamado “Encoder Propio”, que será explicado más adelante). Este valor de salida llamado “Seno”, se introducirá directamente en la señal de consigna de par elástico, obteniendo así un seno con las características que se deseen.

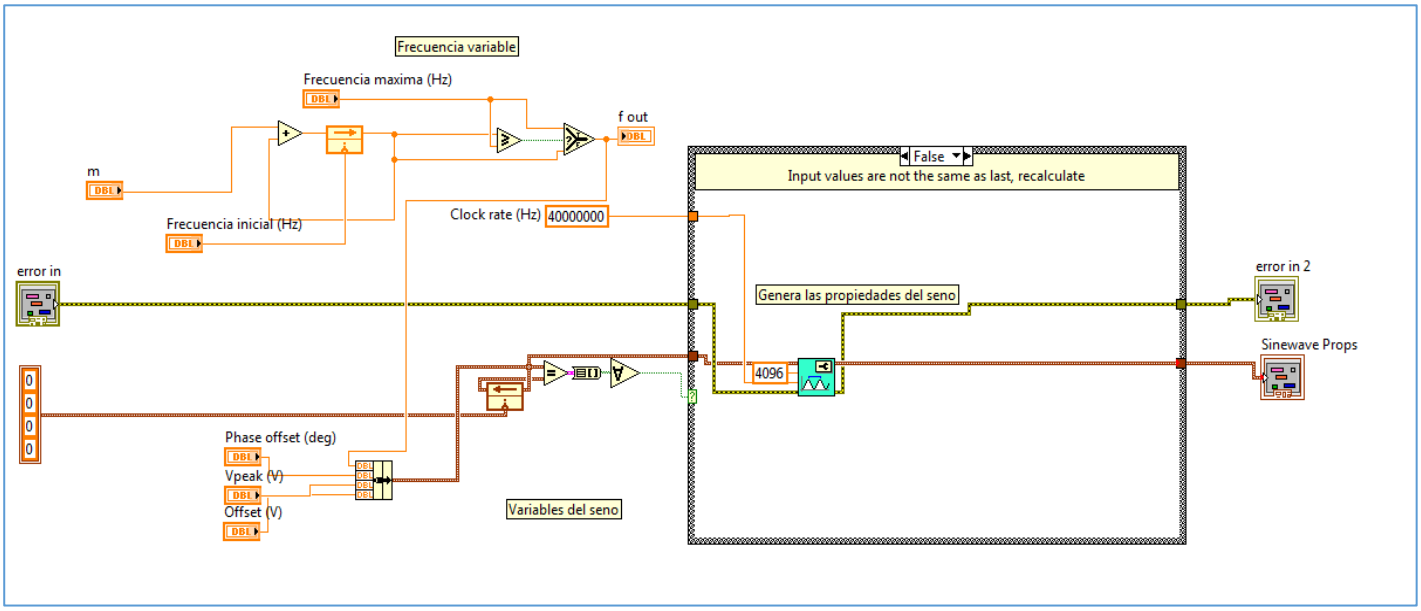

Figura 4. 21: Señal sinusoidal de par elástico de la CPU

En la Figura 4.21, se puede observar el diagrama de bloques de la parte del generador sinusoidal de fuerza implementado en la CPU de myRIO. En este caso, las características conocidas como “Sine Wave Properties”, se han separado y generado en un cluster, ya que la frecuencia, puede ser generada de forma variable si se le introduce a la variable m un valor diferente a 1, o por el contrario, se puede generar un seno con la frecuencia constante si la variable m es igual a 1. El valor de salida de este programa, es “Sine Wave Properties”, que son las propiedades del seno de par que generamos desde la FPGA. Más adelante se explicará el programa de control de fuerza, donde se introducirá esta salida al programa explicado en la Figura 4.20 y así se obtendrá un seno con las características que se deseen según la aplicación.

4.5. Modos de operación

4.5.1. Corriente con consigna directa

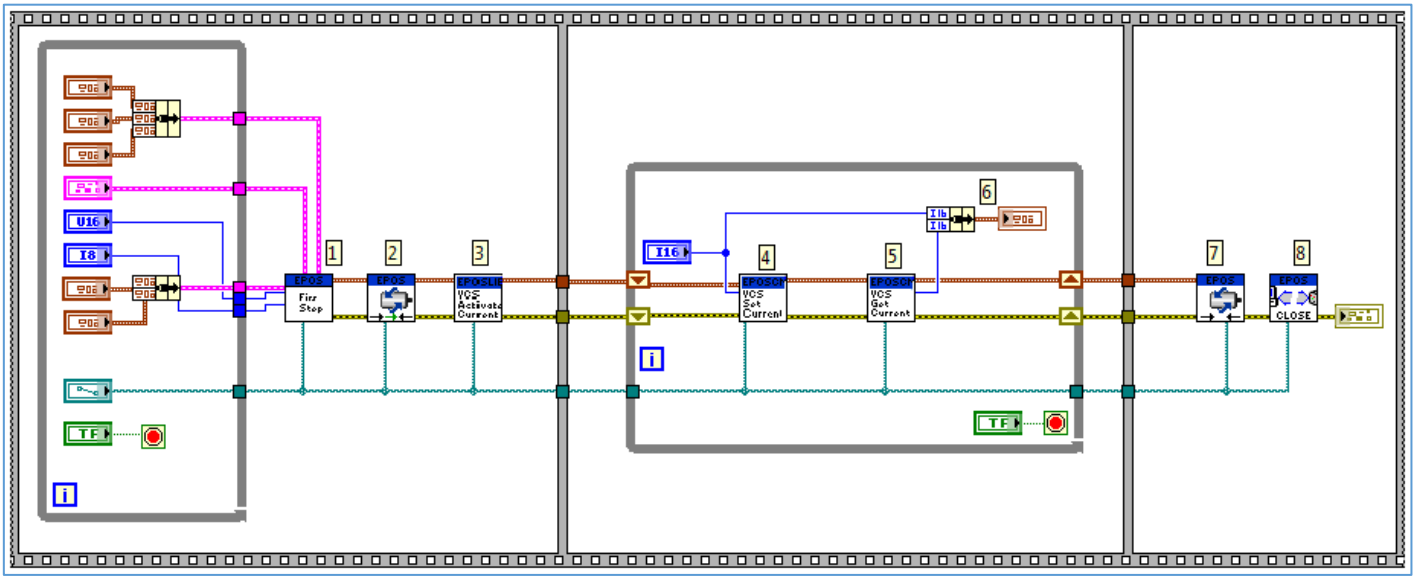

Figura 4. 22: Modo corriente con consigna directa

En la Figura 4.22 se puede observar el diagrama de bloques del modo de operación conocido como "Current Mode". Este diagrama se divide en 7 subsistemas, y a su vez, se divide en tres subdiagramas que componen la estructura de secuencia que completa el programa. A continuación se explicará con detalle el funcionamiento del mismo:

- En el primer subidagrama de la estructura de secuencia se encuentran los tres primeros subsistemas, que son conocidos como el proceso de inicialización tanto del dispositivo EPOS2 como del modo de operación en el que vayamos a trabajar.

  En el subsistema 1, llamado "First Step", se inicializa el dispositivo EPOS2, se introduce el tipo de motor y los principales parámetros del mismo y se configuran los parámetros de los reguladores PID para las variables de posición, corriente y velocidad, como ya explicamos en el apartado 4.3.

  En el subsistema 2, llamado "Enable Axis", se habilitan los ejes del motor, para permitir que éste pueda moverse.

  En el subsistema 3, llamado "Activate Current Mode", se activa el modo de operación Current Mode, de esta forma el dispositivo EPOS2 reconoce los comandos relacionados con este modo de operación.

- En el segundo subdiagrama de la estructura de secuencia se encuentran los dos siguientes subsistemas, que son necesarios para que el motor se mueva con la

corriente demandada y también se obtiene la corriente real que está dando el motor, además de graficar los resultados obtenidos.

El subsistema 4, llamado "Set Current Must", es el encargado de indicarle al motor la corriente a la que debe trabajar.

El subsistema 5, llamado "Get Current Is", se encarga de obtener la corriente real a la que está trabajando el motor. Con los datos obtenidos tanto de la corriente demandada como de la corriente real que tiene el motor, se realiza una gráfica para poder analizar la respuesta del motor.

- En el tercer subdiagrama de la estructura de secuencia se encuentran los dos últimos subsistemas, que se necesitan para parar el motor y cerrar el dispositivo.

  El subsistema 7, llamado "Disable Axis", deshabilita los ejes del motor para que éste deje de moverse. Este subsistema es necesario para evitar que los ejes del motor queden bloqueados cuando se cierra el programa.

  Por último, el subsistema 8, llamado "Close", cierra todos los dispositivos que estén abiertos, liberándolos así para otras aplicaciones.

4.5.2. Corriente con consigna externa

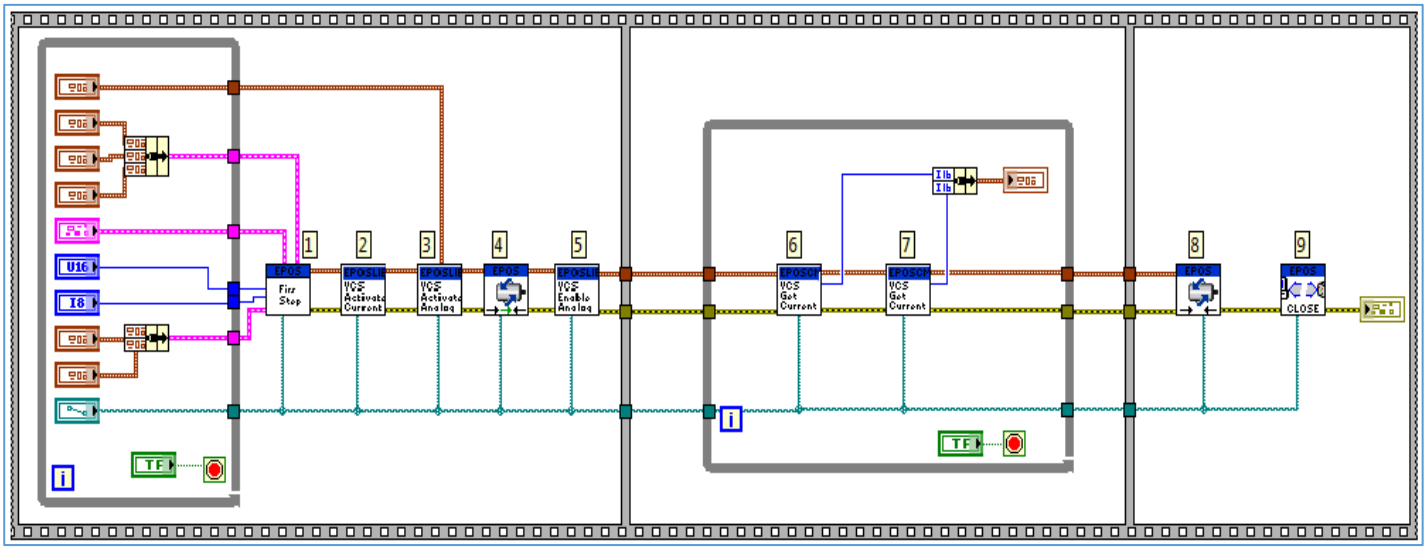

Figura 4. 23: Modo corriente con consigna externa

En la Figura 4.2 se puede observar el diagrama de bloques del modo de operación conocido como "Current Mode Externo". Este diagrama se divide en 9 subsistemas, y a su vez, se divide

en tres subdiagramas que componen la estructura de secuencia que completa el programa. A continuación se explica con detalle el funcionamiento del mismo:

- En el primer subdiagrama de la estructura secuencial se encuentran los primeros 5 subsistemas del programa. Este subdiagrama, como ya se explicó anteriormente es conocido como el proceso de inicialización, y la única diferencia que se encuentra con respecto al modo de operación "Current Mode" está en el subsistema 3 y en el 5. El subsistema 3, llamado "Activate Analog Current Setpoint" se utiliza para configurar la entrada analógica del dispositivo EPOS2 seleccionada, para así poder introducir externamente la corriente demandada para el motor. El subsistema 5, llamado "Enable Analog Current Setpoint" es utilizado para habilitar la máscara que permite la ejecución de la corriente configurada en el subsistema 3.

- En el segundo subdiagrama de la estructura secuencial se encuentran los subsistemas 6 y 7, que son los encargados de obtener la corriente que deseamos en el motor y la corriente real del motor respectivamente. Los datos obtenidos de ambos subsistemas se grafican para analizar los resultados.

- En el tercer subdiagrama de la estructura secuencial se encuentran los subsistemas 8 y 9, que al igual que en el modo de corriente antes explicado se utilizan para cerrar el programa.

Debido a que este modo de operación funciona a través de una señal externa, se ha generado un seno con el programa llamado "Seno fpga.VI" tal y como se explicó en el apartado 4.4.2. Esta señal generada sale del dispositivo myRIO a través de una salida analógica y entra en la entrada analógica del EPOS2 configurada en el programa de la Figura 4.23, y con este mismo programa, se consigue transformar esa señal en una consigna de corriente que debe seguir el motor.

4.5.3. Velocidad con consigna directa

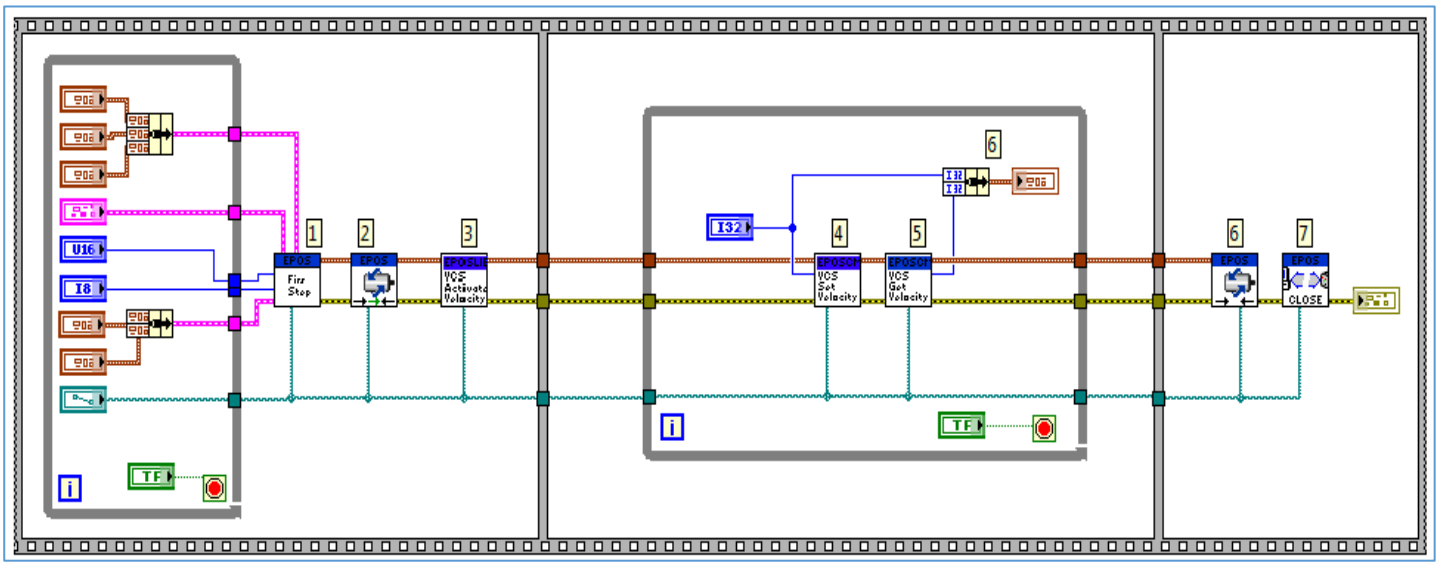

Figura 4. 24: Modo velocidad con consigna directa

En la Figura 4.24 se puede observar el diagrama de bloques del modo de operación conocido como “Velocity Mode”. Al igual que en los anteriores modos de operación ya explicados, el diagrama se divide en tres subdiagramas que componen la estructura de secuencia del programa, y dentro de estos tres subdiagramas, se encuentran los 7 subsistemas que son explicados a continuación:

- Los tres primeros subsistemas del primer subdiagrama de la estructura de secuencia, como ya se explicó en los anteriores modos, se conocen como el proceso de inicialización, y realizan la misma función explicada anteriormente, con la única excepción que en el subsistema 3, se utiliza la función llamada “Activate Velocity Mode”, ya que en este caso se trabaja en modo velocidad.

- Los dos siguientes subsistemas, que se encuentran en el segundo subdiagrama de la estructura de secuencia, realizan la misma función que realizaban en el modo corriente con consigna directa, con el único cambio que ahora la consigna es de velocidad (a través del subsistema 4, llamado “Set Velocity Must”) y los datos obtenidos por parte del motor son de velocidad real (en rpm) (a través del subsistema 5, llamado “Get Velocity Is”).

- Los dos últimos subsistemas, que se encuentran en el tercer subdiagrama de la estructura de secuencia, son los mismos explicados con anterioridad en el modo corriente, y realizan la función de cerrar el programa.

4.5.4. Velocidad con consigna externa

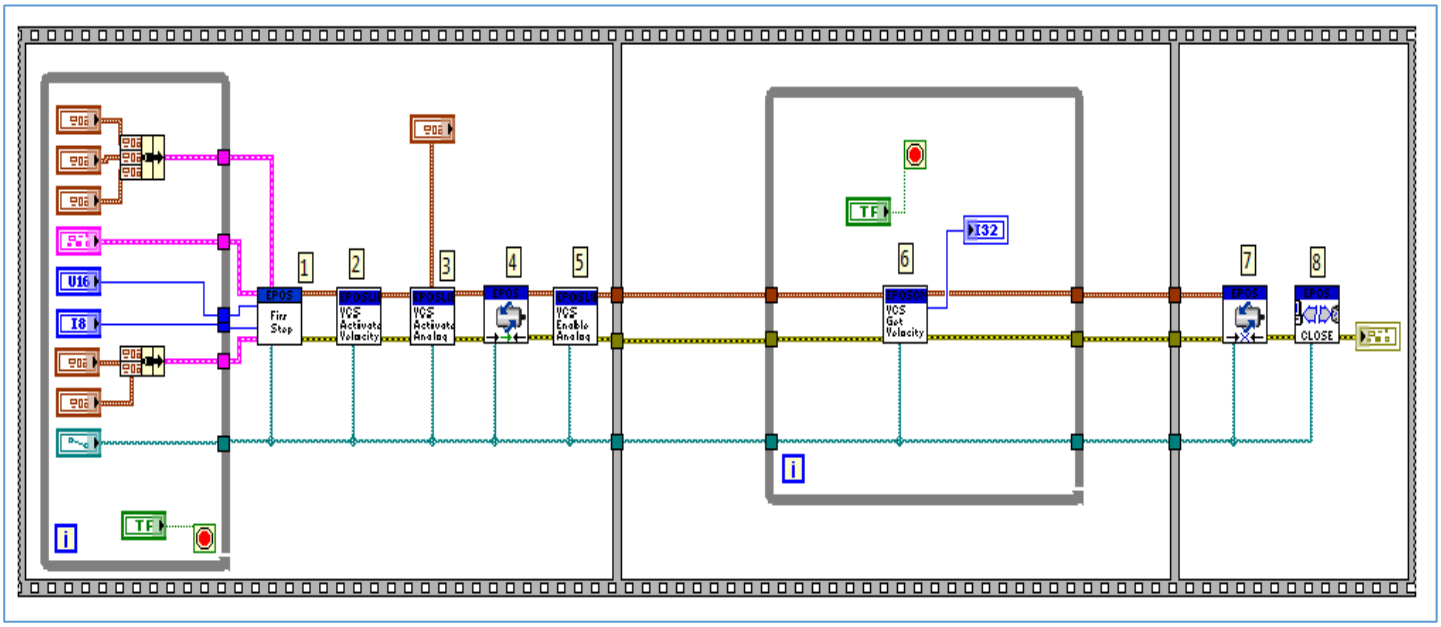

Figura 4. 25: Modo velocidad con consigna externa

En la Figura 4.25 se puede observar el diagrama de bloques del modo de operación conocido como “Current Mode Externo”. Este diagrama se divide en 8 subsistemas, y a su vez, se divide en tres subdiagramas que componen la estructura de secuencia que completa el programa. A continuación se explica con detalle el funcionamiento del mismo:

- En el primer subdiagrama de la estructura secuencial se encuentran los primeros 5 subsistemas del programa. Este subdiagrama, realiza la misma función que en el caso del modo “Corriente con consigna externa”, sustituyendo los subsistemas 3 y 5 por los subsistemas “Activate Analog Velocity Setpoint” y “Enable Analog Velocity Setpoint” respectivamente.

- En el segundo subdiagrama de la estructura secuencial se encuentra el subsistema 6, que es el encargado de obtener la velocidad real del motor. Los datos obtenidos del subsistema se grafican para analizar los resultados.

- En el tercer subdiagrama de la estructura secuencial se encuentran los subsistemas 7 y 8, que al igual que en el resto de modos se utilizan para cerrar el programa.

La señal externa se genera de la misma forma que en el caso del modo corriente con consigna externa, es decir, con el programa llamado “Seno fpga.VI”, y tiene la misma lógica de funcionamiento, lo único que cambia es que la señal se traduce en velocidad para el motor y no en corriente.

4.5.5. Velocidad con consigna auto-generada

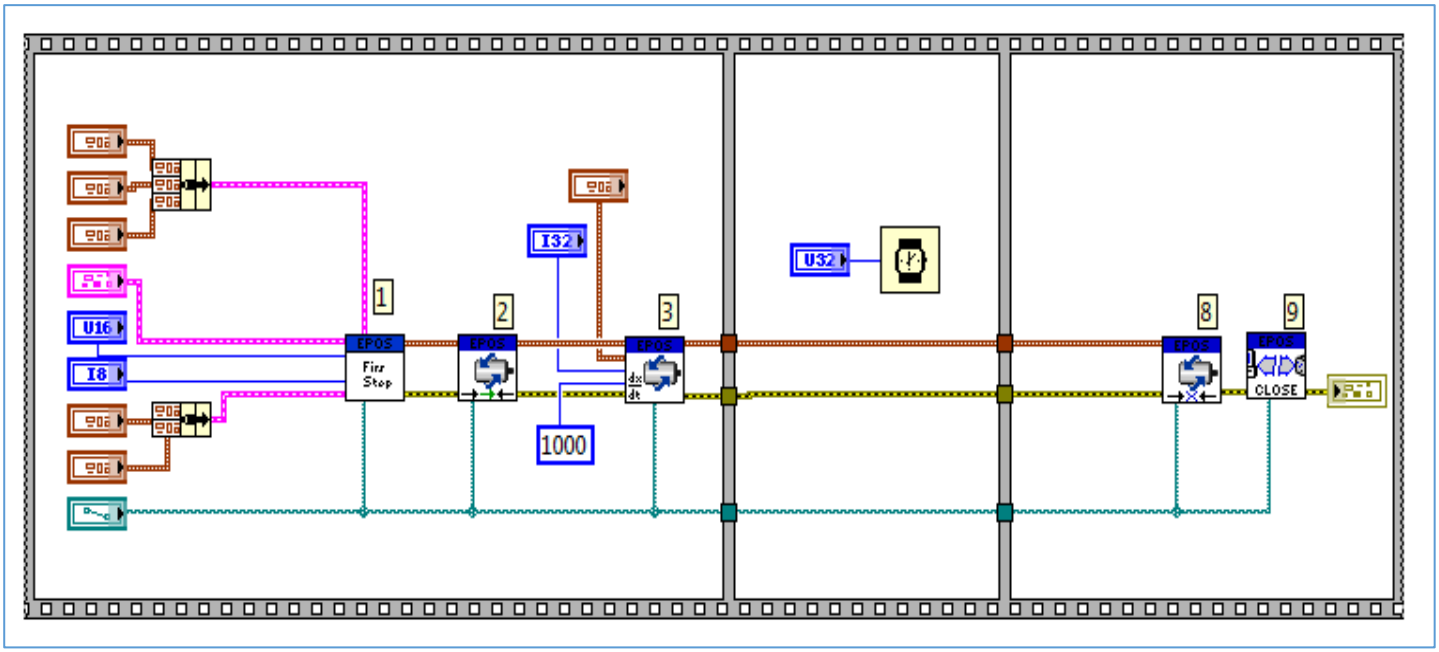

Figura 4. 26: Modo velocidad con consigna auto-generada

En la Figura 4.26 se puede observar el diagrama de bloques del modo de operación conocido como “Profile Velocity Mode”. Este diagrama se divide en 5 subsistemas, y a su vez, se divide en tres subdiagramas que componen la estructura de secuencia que completa el programa. A continuación se explica con detalle el funcionamiento del mismo:

- En el primer subdiagrama de la estructura de secuencia, donde se encuentran los 3 primeros subsistemas, es conocido como el proceso de inicialización. Los dos primeros subsistemas son los mismos que han sido utilizados en todos los modos explicados anteriormente, mientras que el subsistema 3 es conocido como “Move with Velocity” y es utilizado para indicarle al motor el perfil de velocidad que debe seguir, y de esta forma que el controlador construya internamente la trayectoria para seguirla.

- El segundo subdiagrama de la estructura de secuencia es utilizado para indicar el tiempo durante el cual el motor tiene que seguir la trayectoria marcada en el anterior subdiagrama.

- En el tercer subdiagrama de la estructura de secuencia se encuentran los subsistemas 4 y 5, que son los encargados de cerrar el programa.

4.5.6. Posición con consigna directa

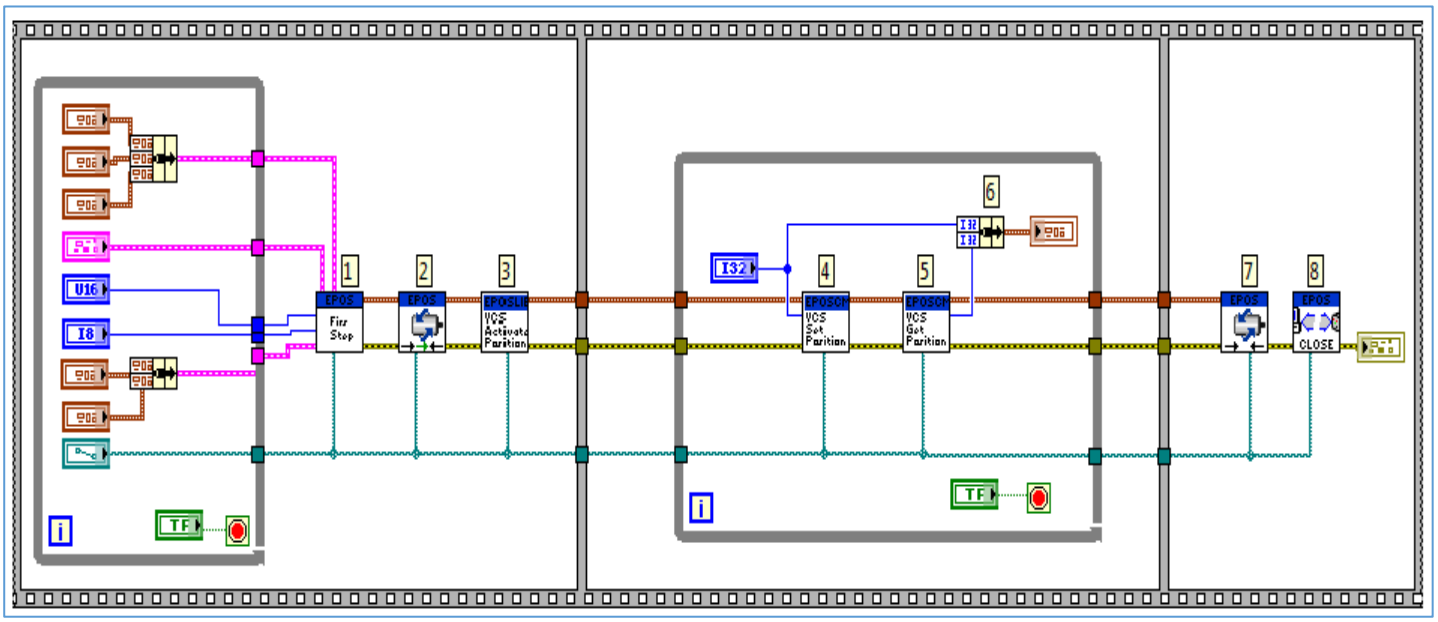

Figura 4. 27: Modo posición con consigna directa

En la Figura 4.27 se puede observar el diagrama de bloques del modo de operación conocido como “Position Mode”. Este diagrama se divide en 8 subsistemas, y a su vez, se divide en tres subdiagramas que componen la estructura de secuencia que completa el programa. A continuación se explica con detalle el funcionamiento del mismo:

- Los tres primeros subsistemas del primer subdiagrama de la estructura de secuencia tienen el mismo funcionamiento que en los anteriores casos. El único cambio que se aprecia se encuentra en el subsistema 3, ya que ahora se utiliza la función llamada “Activate Position Mode”.

- Los dos siguientes subsistemas, que se encuentran en el segundo subdiagrama de la estructura de secuencia, realizan la misma función que realizaban los anteriores modos. Con ellos se obtiene tanto la posición deseada como la posición real del motor, y son graficadas para analizar los resultados obtenidos.

- Los dos últimos subsistemas, que se encuentran en el tercer subdiagrama de la estructura de secuencia, son los mismos explicados con anterioridad.

4.5.7. Posición con consigna auto-generada

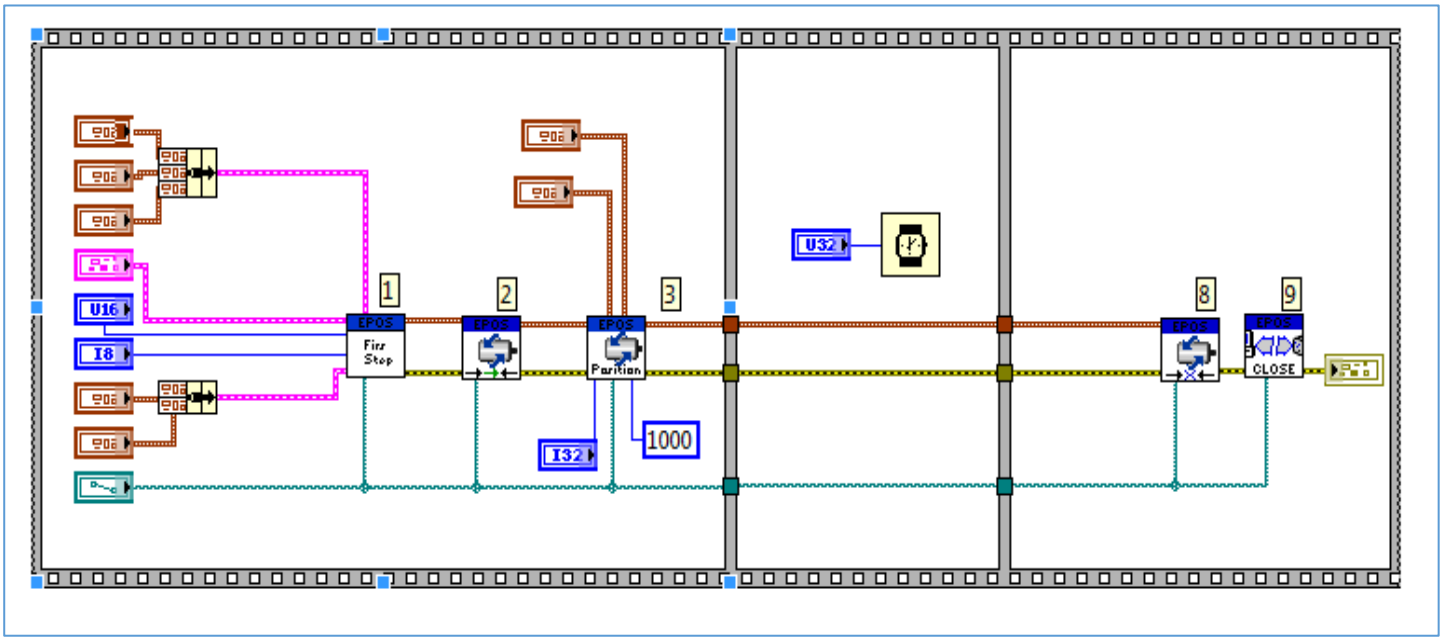

Figura 4. 28: Modo posición con consigna auto-generada

En la Figura 4.28 se puede observar el diagrama de bloques del modo de operación conocido como “Profile Position Mode”. Este diagrama se divide en 5 subsistemas, y a su vez, se divide en tres subdiagramas que componen la estructura de secuencia que completa el programa.

El funcionamiento de este modo es idéntico al funcionamiento del modo velocidad con consigna auto-generada. El único cambio apreciable se encuentra en el subsistema 3, llamado “Move to Position“. Con esta función se crean los perfiles de posición para que se genere la trayectoria correcta y el motor pueda conseguir la posición deseada en el tiempo indicado en el segundo subdiagrama de la estructura de secuencia.

4.6. Control de fuerza

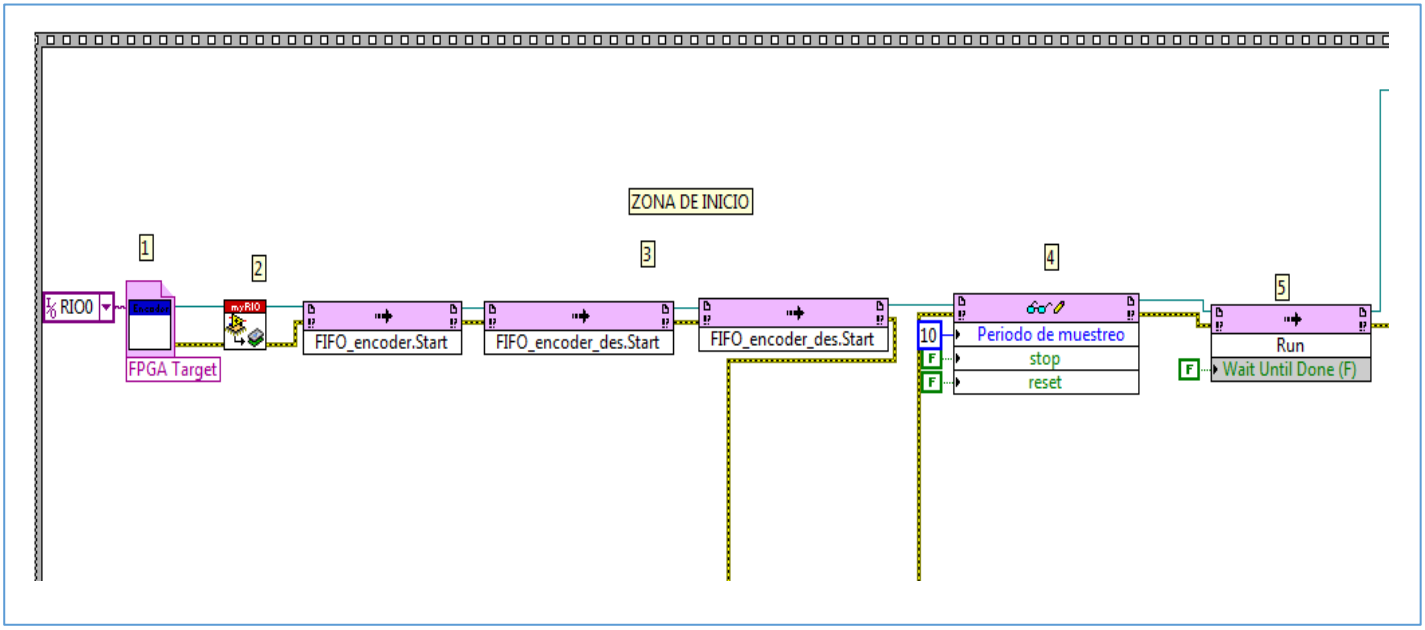

Figura 4. 29: Zona de inicialización en el diagrama del control de fuerza

En la Figura 4.29 se puede observar la zona de inicialización dentro del programa que conforma el control de fuerza del sistema. El programa, como en los casos anteriores que han sido explicados, está formado por tres subdiagramas que completan la estructura secuencial.

La zona de inicialización se encuentra en el primer subdiagrama de la estructura de secuencia, y se puede observar que está formada por 5 partes bien diferenciadas, que se explicarán a continuación:

- La primera parte está formada por un subsistema conocido como "Open FPGA Reference", que es el encargado de abrir la referencia al programa de la FPGA llamado "Encoder propio" (que es el encargado de la obtención de la deflexión angular y del guardado de datos para el análisis de los resultados), y de esta forma poder ejecutar este programa de forma paralela y modificar sus variables desde la CPU para conseguir el objetivo marcado.

- La segunda parte está formada por un subsistema conocido como "Set Custom Bitfile", que realiza la función de crear el archivo de bits (bitfile) correspondiente al programa que ha sido abierto con el anterior subsistema. Esto es necesario para generar el programa como código abierto.

- La tercera parte está formada por 3 subsistemas llamados "Invoke Method"(se utilizan para llamar a un método que se encuentra en el programa abierto de la FPGA). En todos ellos se inicializan las FIFOs utilizadas para el guardado de los datos correspondientes, para así después poder hacer un análisis de los resultados.

- La cuarta parte está formada por un subsistema llamado “Read/Write Control” (llama a variables del programa de la FPGA, para así poder modificarlas), que es el encargado de inicializar las variables Periodo de muestreo, reset y stop, ya que si no tendrían el valor de la anterior compilación.

- Por último, la quinta parte está formada por otro subsistema “Invoke Method”, en el que se ha invocado al método “Run”, que lo que hace es ejecutar el programa de la FPGA.

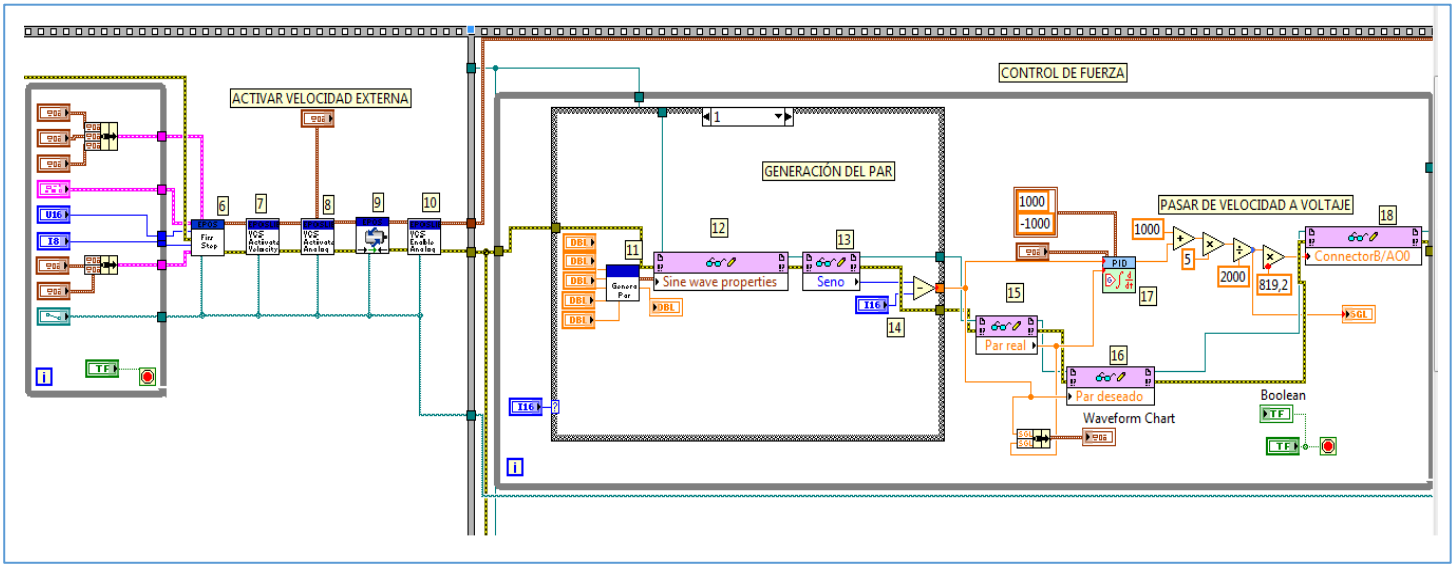

Figura 4. 30: Zona de control en el diagrama de control de fuerza

En la Figura 4.30, se puede observar la zona del control de fuerza. A continuación de la zona de inicialización, y todavía dentro del primer subdiagrama de la estructura de secuencia, se inicializa el modo de operación “Velocidad con consigna externa”, es decir, se realizan exactamente los mismos pasos que en el primer subdiagrama de la estructura de secuencia del programa de la Figura 4.25.

Una vez hecho esto, comienza el segundo subdiagrama de la estructura de secuencia. El subsistema 11, llamado “Generador fuerza”, es el subsistema explicado en la Figura 4.21, y por lo tanto, genera las características del seno de par elástico que se desea, y su salida se introduce en la variable interna del programa de la FPGA “Sine wave properties”, a través del subsistema “Read/Write Control”, como ya se explicó en el apartado 4.4.2.

La variable “Seno” (subsistema 13), es la salida del seno que se ha generado en la FPGA (la salida del programa que se está ejecutando en paralelo desde la FPGA), por lo que esta salida es directamente la variable de par deseado, y se introduce como Setpoint (consigna) en el PID (subsistema 17).

Por otra parte, dentro del programa "Encoder propio", también se obtiene el par real con el que está trabajando el actuador, por lo que la variable "Par real" obtenida del programa de la FPGA, se introduce también en el PID como Process Variable (señal medida), para que así el PID contrarreste la diferencia que existe entre el par real y el deseado. Además, estas dos variables son graficadas para analizar los resultados obtenidos.

La salida del PID se traduce en velocidad del motor, pero como el programa trabaja con una consigna externa, hay que transformar esta velocidad en voltaje, y enviarlo a través de una salida analógica al dispositivo EPOS2. Con las características del programa de velocidad con consigna externa, se consigue que el voltaje que llega por la entrada analógica del dispositivo EPOS2, sea traducido en velocidad de nuevo, y así el motor se mueve con una mayor velocidad cuanto mayor sea la diferencia entre el par real y el par deseado. Cuando esta diferencia sea cero, significará que el actuador ha llegado al par de consigna, y por lo tanto, la velocidad del motor será cero, ya que no necesita moverse más para llegar a ese par.

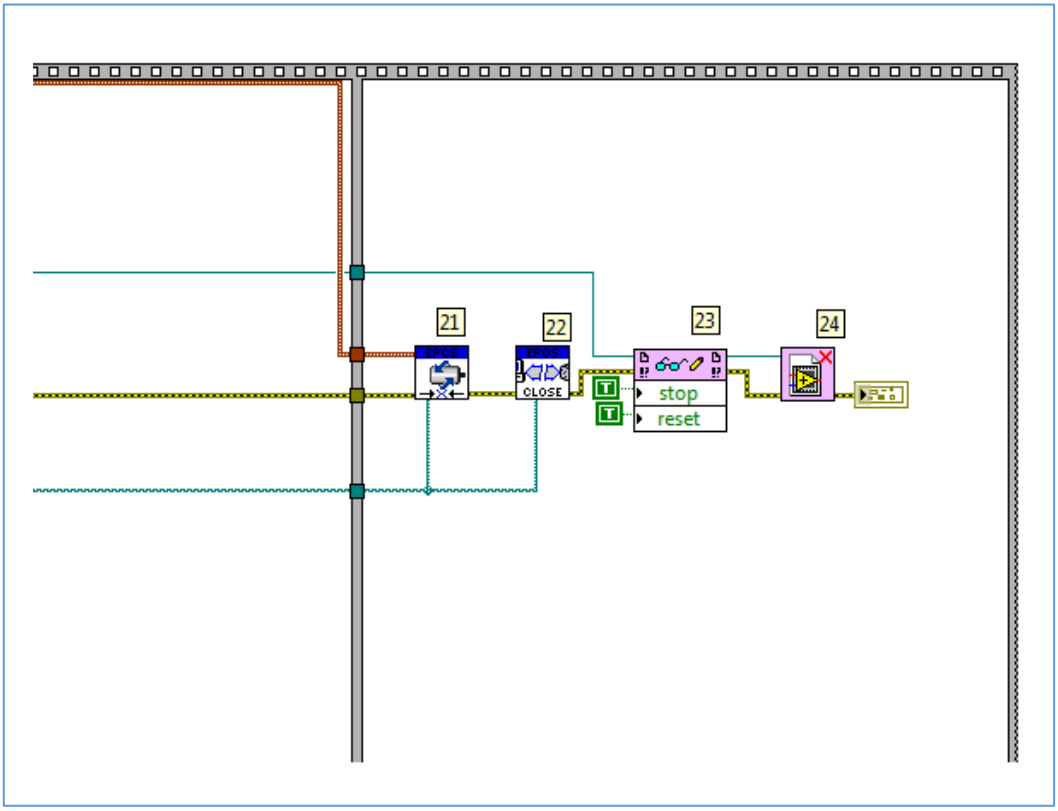

Figura 4. 31: Zona de cierre del esquema de control de fuerza

Por último, para finalizar el control de fuerza, es necesario, como en todos los anteriores casos explicados, cerrar el programa. En la Figura 4.31, lo único nuevo que se observa con respecto a otros programas ya explicados, es que no solo hay que cerrar el programa de velocidad, sino que también hay que cerrar la referencia que se había abierto al programa de la FPGA,

además, se cambian los valores de stop y reset a 1, para que la ejecución del mismo sea detenida.

4.6.1. Estimación del par elástico

Gracias al encoder extrínseco disponible, se tiene la capacidad de conocer la deflexión angular del actuador elástico, sin embargo, no se conoce el par elástico. Para poder trabajar con el par elástico, hay que fijarse en la gráfica que relaciona la deflexión angular (eje x) con el par elástico del actuador (eje y). La grafica se puede observar en la Figura 4.32.

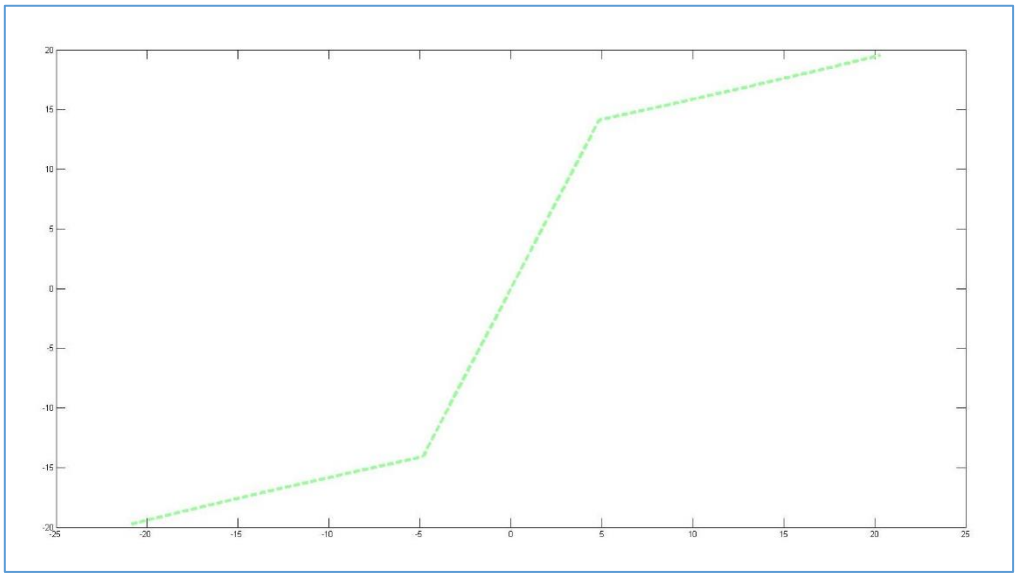

Figura 4. 32: Relación entre la deflexión angular y el par elástico

Como se puede observar en la Figura 4.32, existen tres zonas diferenciadas en la gráfica. La zona media, que está comprendida entre una deflexión angular de 4.9 grados y -4.9 grados, la zona superior, que se produce cuando la deflexión angular es superior a 4.9 grados, y la zona inferior, que se produce cuando la deflexión angular es inferior a -4.9 grados.

Para la transformación de la deflexión angular obtenida con el encoder al par elástico, solo hay que tratar a cada zona como una recta, y así se obtiene una relación sencilla para calcular el par elástico del actuador (Figuras 4.33, 4.34 y 4.35).

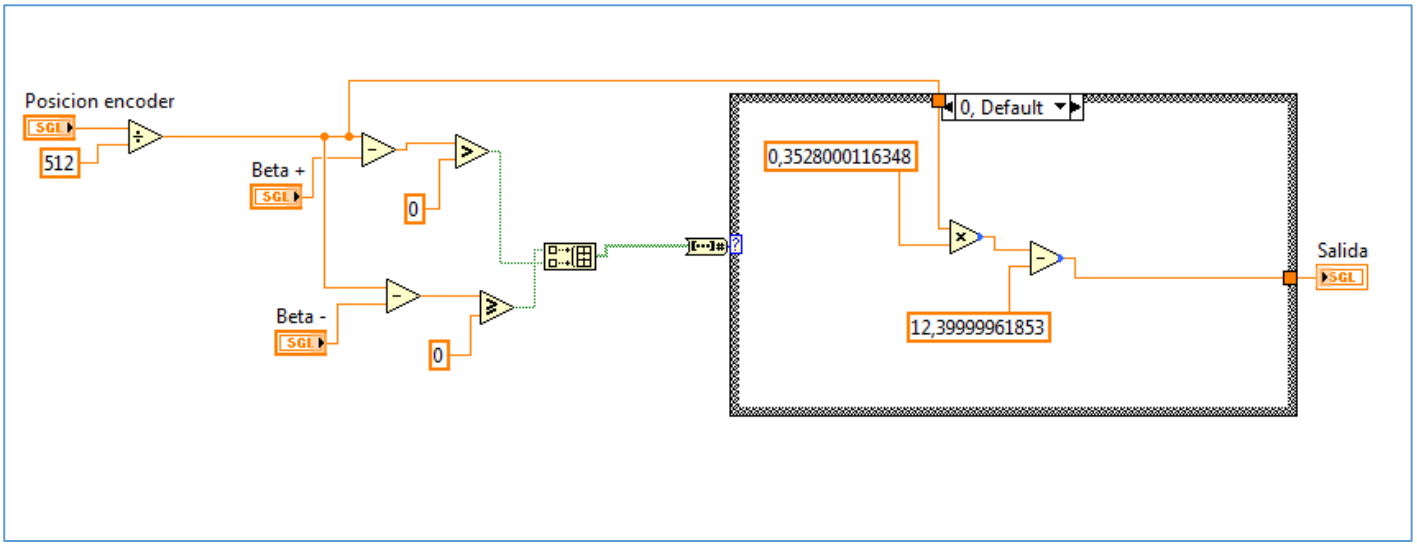

Figura 4. 33: Caso en el que la deflexión angular es inferior a -4.9 grados

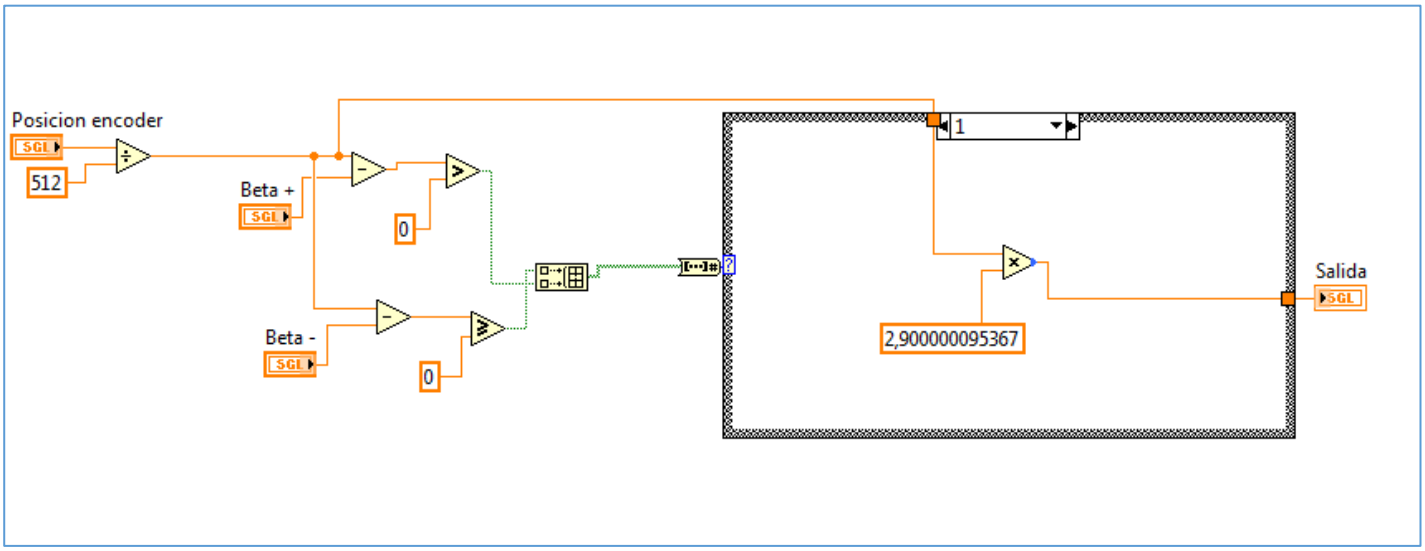

Figura 4. 34: Caso en el que la deflexión angular se encuentra entre -4.9 y 4.9 grados

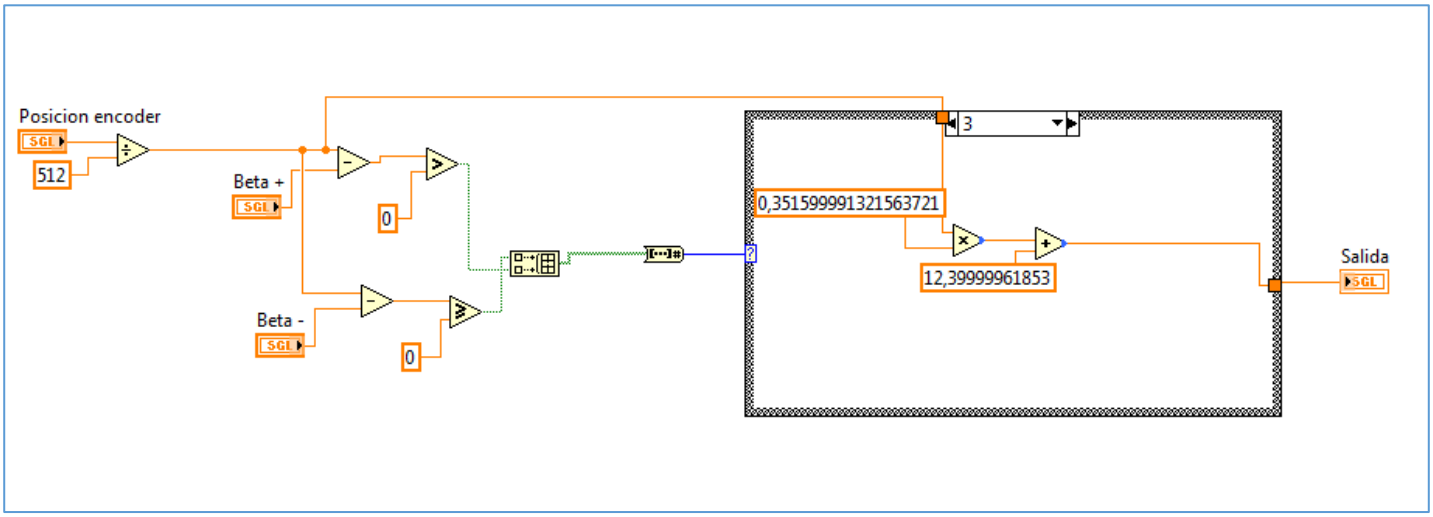

Figura 4. 35: Caso en el que la deflexión angular es superior a 4.9 grados

En las Figuras 4.33, 4.34 y 4.35 se observa la lógica utilizada para decidir en qué zona de la gráfica se encuentra la deflexión angular de entrada. La posición real del encoder, es dividida entre 512 para obtener la deflexión angular en grados (que es como se representa en la Figura 4.32), y una vez obtenida, se compara ésta con 4.9 grados (Beta +) y con -4.9 grados (Beta -).

En el caso de que la Deflexión real sea superior a 4.9 grados, el comparador de arriba, dará como resultado un uno. A su vez, la deflexión real también es superior a -4.9 grados, por lo que el segundo comparador también dará como resultado un uno, y por lo tanto, el resultado que obtenido será 11 (en binario), que se traduce a 3 en decimal, y se ejecutará la opción de la Figura 4.35. En este caso, la recta sigue la siguiente ecuación:

$$Salida\ (par\ elástico) = (0{,}352 * Deflexión\ real) + 12{,}4 \qquad (8)$$

En el caso de que la deflexión real sea inferior a -4.9 grados, el comparador de arriba dará un cero, y el comparador de abajo también dará un cero, por lo tanto, se obtiene como resultado 00, que es 0 en decimal, y se ejecutará la opción de la Figura 4.33. En este caso, la recta sigue la siguiente ecuación:

$$Salida\ (par\ elástico) = (0{,}353 * Deflexión\ real) - 12{,}4 \qquad (9)$$

Por último, en el caso de que la deflexión real esté comprendida entre 4,9 y -4,9 grados, el primer comparador dará un cero, mientras que el segundo dará un uno, y de esta forma se obtiene el valor 01, que en decimal es 1, y por lo tanto, se ejecutará la opción de la Figura 4.34. En este caso, la recta sigue la siguiente ecuación:

$$Salida\ (par\ elástico) = 2{,}9 * Deflexión\ real \quad (10)$$

4.6.2. Adquisición de datos

La adquisición de datos está compuesta por dos partes. La primera de ellas se da en el programa de la FPGA "Encoder propio" (Figuras 4.36 y 4.37), donde se obtiene el par real del actuador elástico, el par deseado (que es introducido en el programa explicado en el apartado 4.6) y el tiempo que tarda en realizarse cada iteración. Estos tres datos son guardados cada uno de ellos en una FIFO diferente, para ir acumulando todos los datos obtenidos en tiempo real.

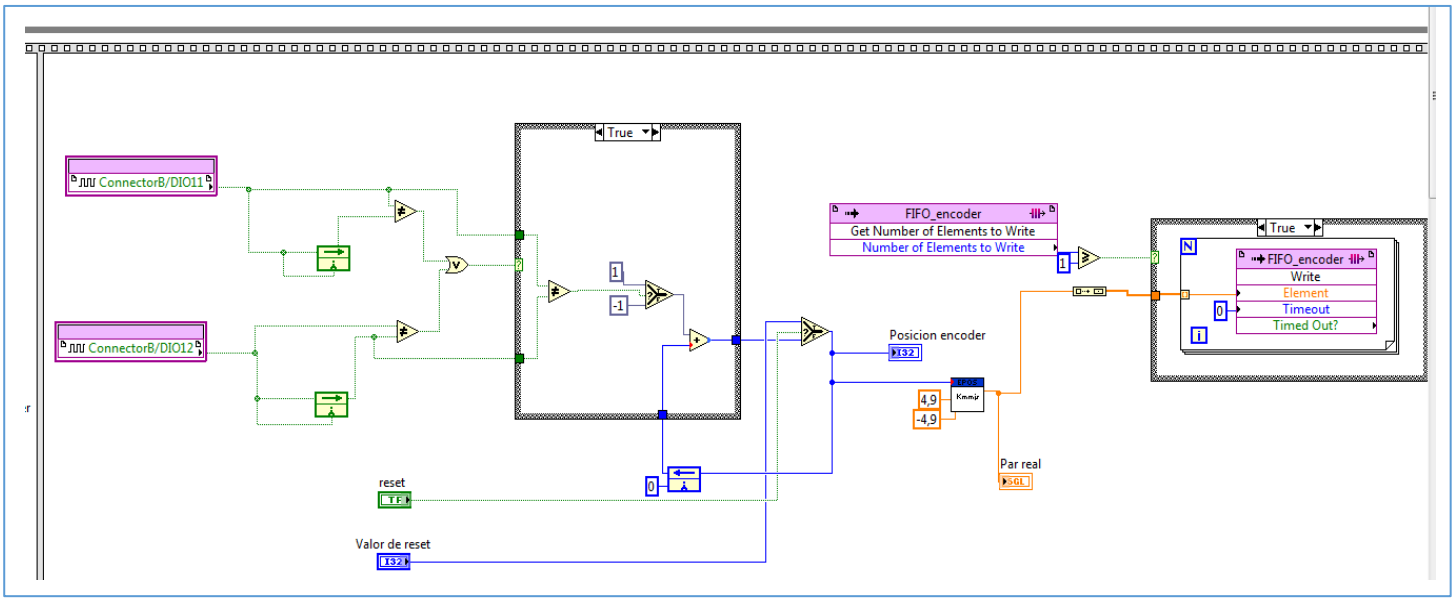

Figura 4. 36: Adquisición de los datos del par real del actuador

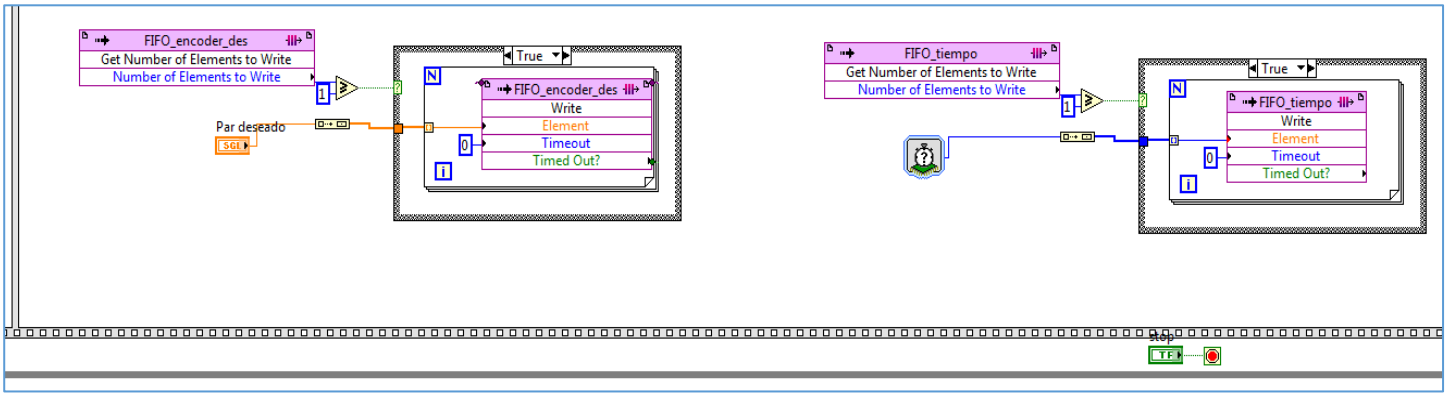

Figura 4. 37: Adquisición de los datos del par deseado y del tiempo

En la Figura 4.36 se puede observar el diseño del programa de medición de un encoder, por el medio del cual se obtiene la variable de salida llamada “Posición encoder” (deflexión angular del actuador). Esta variable es introducida al programa descrito en las Figuras 4.33, 4.34 y 4.35, y a través de él, como ya se explicó con anterioridad, se calcula el par real del actuador. Este par real es introducido a la FIFO llamada “FIFO_encoder”, para acumular ahí todos los datos que se vayan obteniendo.

En la Figura 4.37, se puede observar que se introduce la variable de par deseado en la FIFO llamada “FIFO_encoder_des”, con el mismo objetivo que en el caso anterior, y también se introduce el valor del tiempo (en microsegundos) que tarda cada iteración en otra FIFO, llamada “FIFO_tiempo”.

Con estos tres datos guardados, se podrá realizar un análisis de los resultados obtenidos cuando se realicen las pruebas necesarias para el control de fuerza.

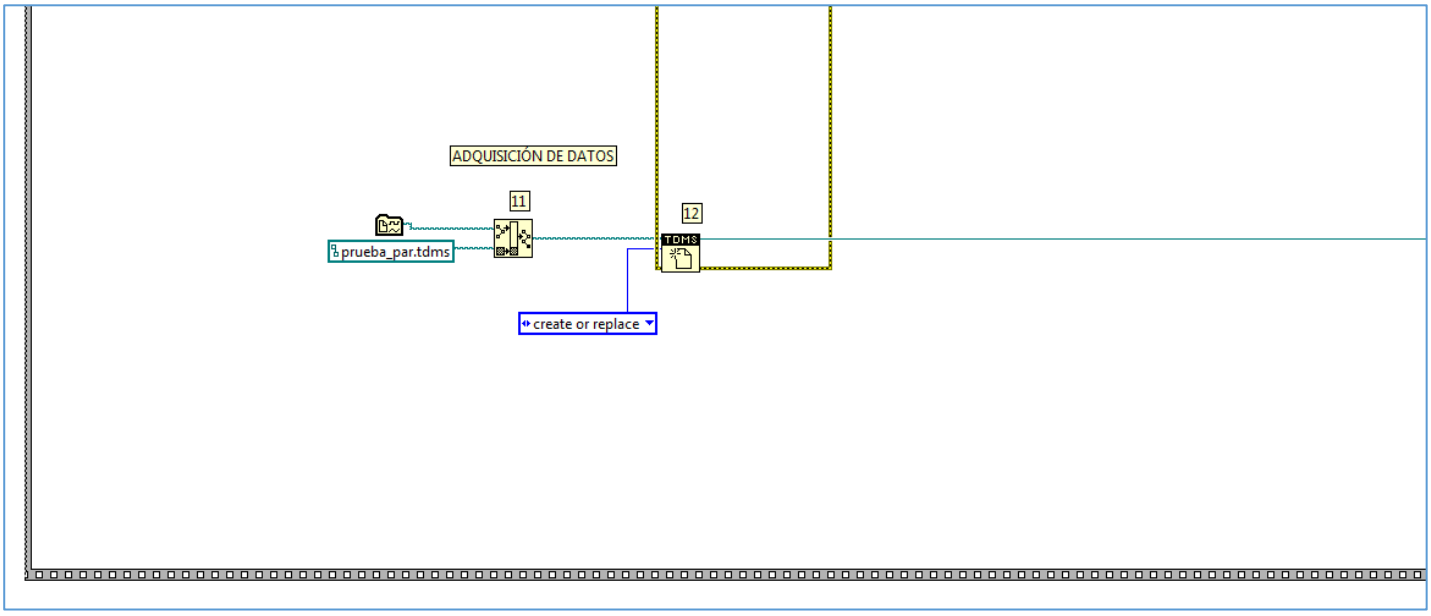

Figura 4. 38: Inicialización en el proceso de crear un archivo TDMS

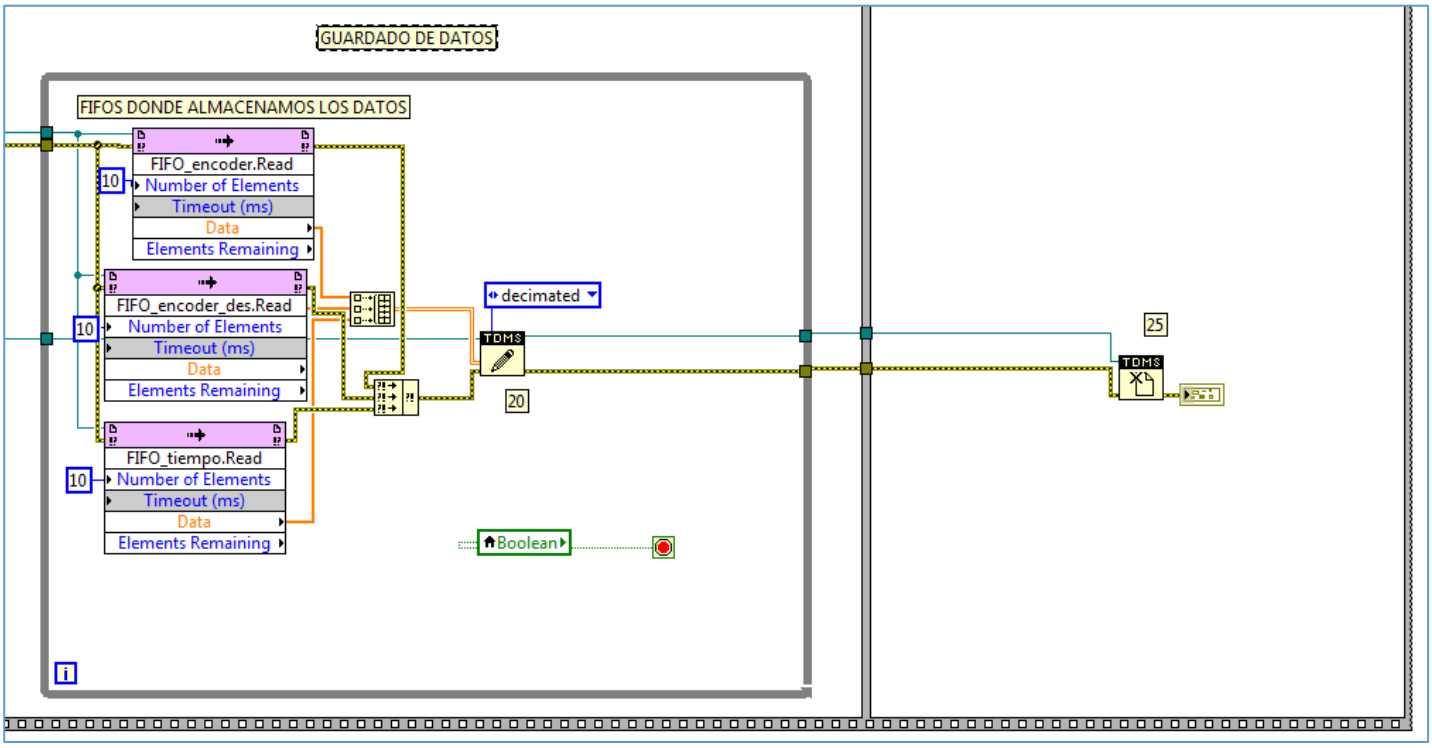

Figura 4. 39: Guardado de datos para el archivo TDMS y cerrado del mismo

La otra parte de la adquisición de datos (la parte diseñada en la CPU), es la se puede ver en las Figuras 4.38 y 4.39. En la Figura 4.38, se muestra el proceso de inicialización para la creación de un archivo TDMS. El subsistema 12 se encarga de abrir el archivo TDMS, con el nombre y la

función que se le indique (en este caso, el archivo TDMS que crea se llamará prueba_par, y cada vez que se ejecute el programa, lo creará en caso de que no exista, o lo reemplazará en caso de que sí). El archivo será guardado en el destino por defecto dentro de la memoria del dispositivo NI myRIO gracias al subsistema 11.

En la Figura 4.39, se puede observar cómo se leen los datos de las FIFOs creadas anteriormente para el guardado del par real, par deseado y el tiempo. Estos datos leídos son introducidos en el archivo TDMS, para después poder analizarlo. Además, con el subsistema 25, se cierra la creación del archivo TDMS, para asegurar su correcto funcionamiento. Este archivo creado es básicamente un Excel con todos los datos obtenidos en las diferentes pruebas que se realizarán para el control de fuerza. De esta forma, se podrán graficar los resultados de par obtenidos con respecto al tiempo, y así poder analizar si han sido o no satisfactorios, como ya se explicará en el Capítulo 5.

Tanto la Figura 4.38 como la 4.39, forman parte del programa de control de fuerza explicado en el apartado 4.6, por lo que en el mismo programa, de forma paralela, se está realizando el control de fuerza y la adquisición de datos, característica típica de los programas construidos con Labview, donde el paralelismo es un elemento muy importante.

5. RESULTADOS EXPERIMENTALES

5.1. Introducción

Una vez finalizado el desarrollo del software, el banco estaba operativo para realizar los ensayos pertinentes que confirmen que el actuador MMJS instalado estaba siendo controlado de forma correcta en todos los modos de operación que hay disponibles gracias al controlador EPOS2, y además comprobar que se está realizando un control de fuerza del actuador, que eran los principales objetivos que se habían marcado al empezar el trabajo de fin de grado.

A lo largo de este capítulo, se analizarán los resultados obtenidos en los ensayos de cada uno de los programas explicados en el capítulo 4, así como la comparación de los mismos con los cálculos teóricos disponibles, para comprobar el grado de capacidad que tiene un dispositivo embebido de bajo coste de controlar un sistema completo.

5.2. Control de corriente

5.2.1. Directo

Se han realizado dos ensayos de corriente, uno con el regulador PID autosintonizado por el propio dispositivo EPOS2, y otro con un regulador PID calculado experimentalmente. El ensayo ha consistido en aplicar varios escalones de corriente al motor para comprobar su respuesta temporal ante los mismos. El resultado de los ensayos se muestra en la Figura 5.1.

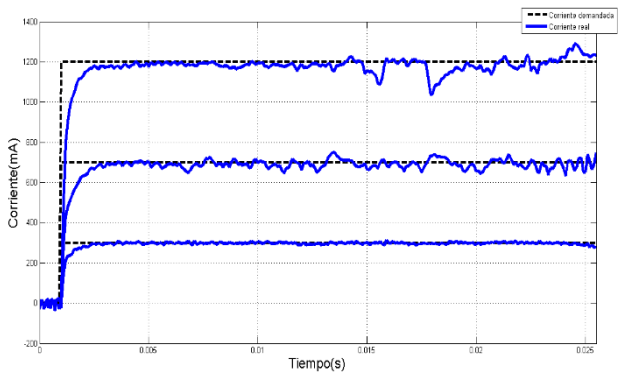

Figura 5.1a: Ensayo de corriente con PID auto-sintonizado

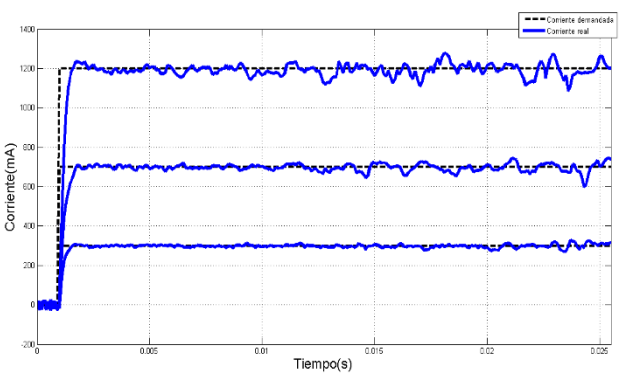

Figura 5.1b: Ensayo de corriente con PID propio

Figura 5. 1: Ensayos de corriente con consigna directa

Los escalones aplicados han sido de 300, 700 y 1200 mA en ambos ensayos. En la Figura 5.1, se puede observar que la línea discontinua de color negro sería la corriente de consigna para el motor, mientras que la línea azul marina sería la corriente real del motor.

Se puede ver que la respuesta del motor con los valores del PID propio es más rápida que con los valores del PID auto-sintonizado, por lo que, a primera vista, se podría concluir que con PID propio se obtiene un mayor ancho de banda, que es el objetivo principal.

A partir de los resultados obtenidos, utilizando el "System Identification Tool" de Matlab, se ha estimado una función de transferencia para cada sistema, y se muestran a continuación:

$$FT_{PID\_auto-sintonizado} = \frac{8536\,s + 2{,}163 \cdot 10^7}{s^2 + 1{,}417 \cdot 10^4\,s + 2{,}182 \cdot 10^7} \tag{11}$$

$$FT_{PID\_propio} = \frac{6146\,s + 1{,}635 \cdot 10^7}{s^2 + 8458\,s + 1{,}648 \cdot 10^7} \tag{12}$$

Las dos funciones de transferencia ((11) y (12)) obtenidas han sido utilizadas para analizar la respuesta de los sistemas ensayados ante la aplicación de un escalón unitario. Los resultados se muestran en la Figura 5.2.

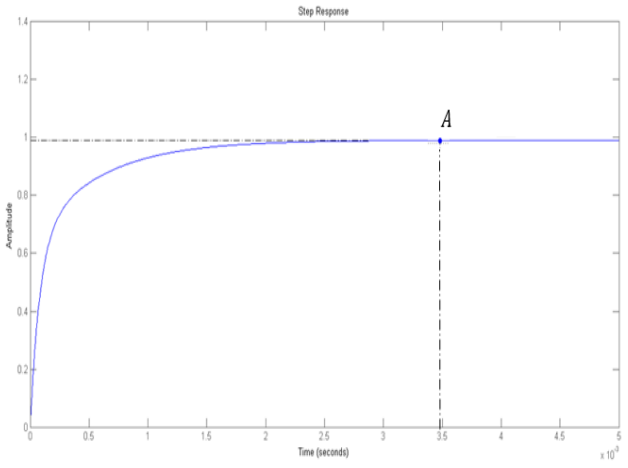

Figura 5.2a: Respuesta a escalón con PID auto-sintonizado

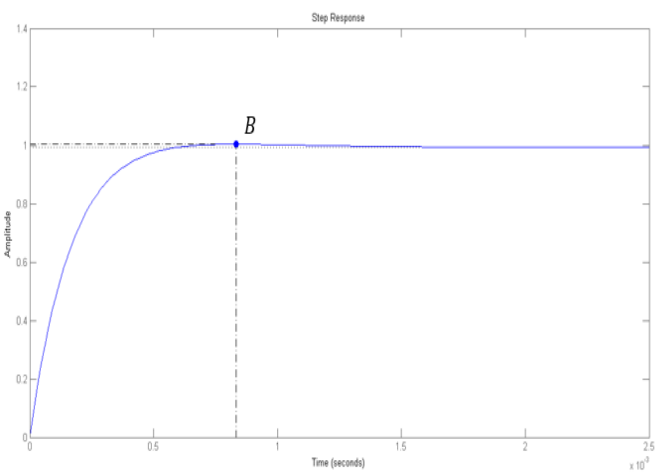

Figura 5.2b: Respuesta a escalón con PID propio

Figura 5. 2: Respuesta de los ensayos de corriente con consigna directa ante un escalón unitario

Tanto el punto A como el punto B que se pueden ver en las Figuras 9a y 9b respectivamente, son el valor máximo de amplitud que alcanza la respuesta de cada uno de los sistemas. Gracias a estos dos puntos, es posible calcular el coeficiente de amortiguamiento de ambos sistemas. Este coeficiente se calcula utilizando la siguiente fórmula:

$$M_P = 1 + e^{\frac{\xi \cdot \pi^2}{\sqrt{1-\xi^2}}} \tag{13}$$

Donde M_P es el valor de pico dado por los puntos A y B, y ξ el coeficiente de amortiguamiento que queremos calcular.

En el sistema de la Figura 5.2a, el valor de M_P (punto A) es igual a 0,991. Sustituyendo en la ecuación (13), se obtiene un coeficiente de amortiguamiento mayor que 1, por lo tanto este sistema es sobreamortiguado.

En el sistema de la Figura 5.2b, el valor de M_P (punto B) es igual a 1. Sustituyendo en la ecuación (13), se obtiene un coeficiente de amortiguamiento igual a 1. Este valor de amortiguamiento da lugar a un sistema críticamente amortiguado, que es un estado frontera entre el sobreamortiguamiento y el subamortiguamiento.

Además de realizar un estudio de la respuesta de los sistemas ante un escalón unitario, también se ha querido realizar una caracterización de la respuesta frecuencial de ambos, por lo que se han obtenido los diagramas de Bode correspondientes, que se muestran en la Figura 5.3.

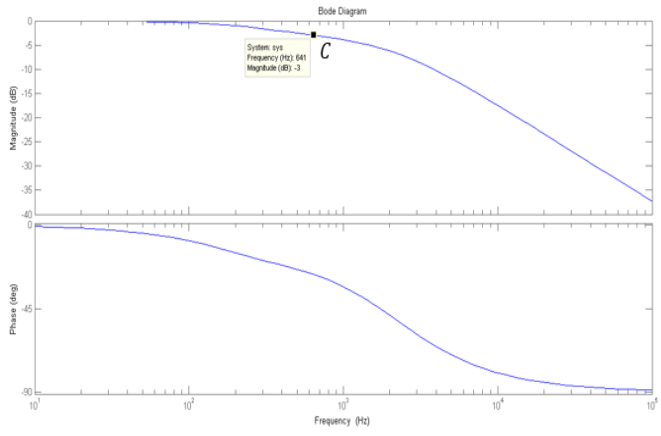

Figura 5.3a: Diagrama de bode del ensayo de corriente con PID auto-sintonizado

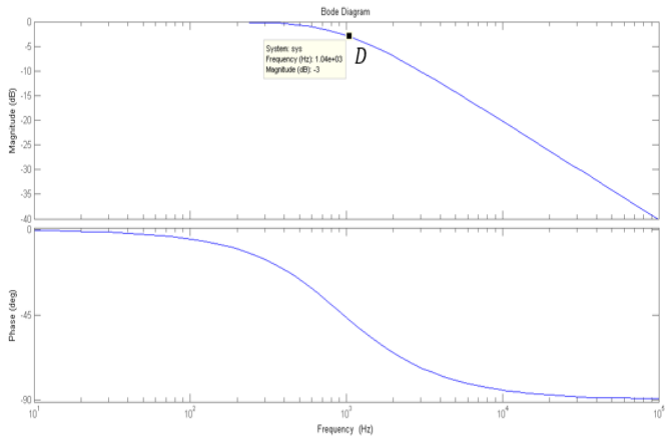

Figura 5.3b: Diagrama de bode del ensayo de corriente con PID propio

Figura 5. 3: Diagramas de bode de los ensayos de corriente con consigna directa

Los puntos C y D indican el ancho de banda de los sistemas. En el caso del sistema con PID auto-sintonizado este ancho de banda es igual a 641 Hz, que es muy inferior al deseado (1000 Hz) para realizar el ensayo de corriente de forma correcta, y así evitar que en el control de velocidad, afecte el lazo de corriente.

En el caso del sistema con PID propio, el ancho de banda es igual a 1040 Hz. Se puede observar que este ancho de banda es incluso superior al deseado, por lo que, se podría decir que los valores de las ganancias del PID calculado de forma experimental para el control de corriente se acercan mucho más a lo que se considera "ideal" según las simulaciones disponibles.

Para analizar la estabilidad de ambos sistemas, es necesario calcular los márgenes de fase y de ganancia.

El margen de ganancia es la cantidad de ganancia, en decibelios, que se puede añadir al lazo de control antes de que el sistema en lazo cerrado se vuelva inestable [32]. Este margen se calcula a través del corte del diagrama de fase con -180 grados, y en ambos casos la gráfica no llega a los -180 grados, por lo que en los sistemas obtenidos, el margen de ganancia es infinito. Esto significa que el sistema no se vuelve inestable aunque se aumente la ganancia en el lazo.

El margen de fase es la cantidad de retardo puro que se puede añadir al sistema antes de que el propio sistema en lazo cerrado se vuelva inestable [32]. Este margen se calcula como el ángulo que le falta a -180 grados para llegar a la fase cuando la ganancia es de 0 dB. En este caso, debido a que los dos sistemas comienzan con una ganancia igual a 0 dB y por lo tanto sus gráficas no cortan con este valor, tienen un margen de fase infinito.

Debido a que en ambos casos los márgenes de fase y de ganancia son positivos (y además infinitos), los dos sistemas son estables.

5.2.2. Externo

Se ha realizado un ensayo de corriente en el que se genera una señal de consigna externa a través de la FPGA del dispositivo myRIO. Esta señal generada es un seno de amplitud de 5v, pero que a través del programa explicado en la Figura 4.23 se consigue que la corriente demandada para el motor sea un seno con amplitud de 300 mA.

Gracias a que la generación de la señal se realiza a través de la FPGA de forma externa al EPOS2, los datos generados no tienen adquirirse por el puerto USB, y por lo tanto, se pueden guardar los datos de la señal generada con un ancho de banda mucho mayor.

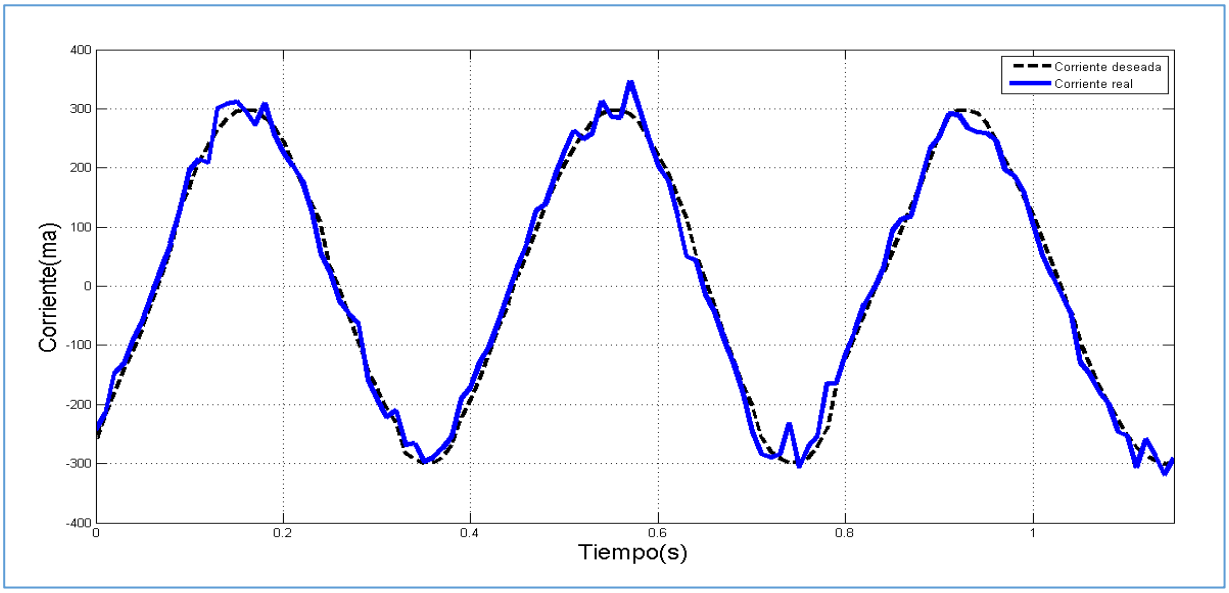

Figura 5. 4: Ensayo de corriente con consigna externa

Se puede observar en la Figura 5.4 que la señal de consigna es seguida aproximadamente de forma correcta por el motor. Se ha realizado el ensayo a una frecuencia baja (entre 2 y 3 Hz), ya que, aunque la señal externa generada si puede muestrearse a frecuencias muy altas, los datos provenientes de la corriente real del motor, son enviados al dispositivo myRIO por medio de USB, y por lo tanto, la adquisición de datos no tiene el ancho de banda deseado.

El valor ideal de muestreo de la corriente (ancho de banda del sistema) es 1 kHz, por lo que se ha decidido no realizar más pruebas en este modo de operación, ya que el ancho de banda disponible es mucho menor y no existe la posibilidad de conseguir ese ancho de banda ideal a través de los medios disponibles.

5.3. Control de velocidad

5.3.1. Directo

Para analizar los resultados del control de la velocidad del motor, se han realizado dos ensayos de velocidad, al igual que se hizo en el control de corriente. Uno de los ensayos se ha realizado con el regulador PID autosintonizado por el propio dispositivo EPOS2, y otro con un regulador PID calculado de forma experimental. El ensayo ha consistido en aplicar varios escalones de velocidad al motor para comprobar su respuesta temporal ante los mismos. El resultado de los ensayos se muestra en la Figura 5.5.

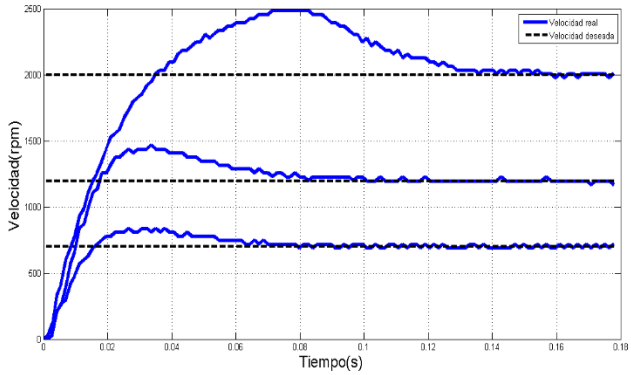

Figura 5.5a: Ensayo de velocidad con PID auto-sintonizado

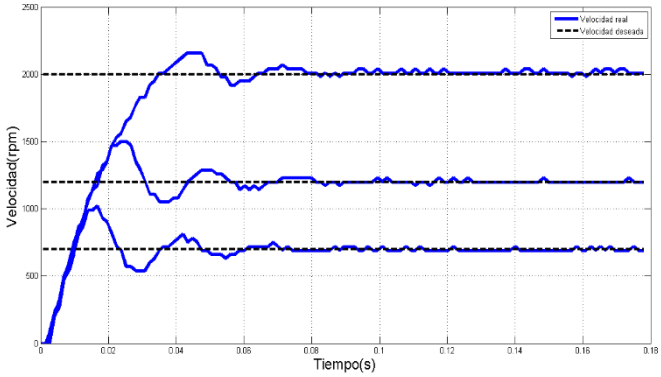

Figura 5.5b: Ensayo de velocidad con PID propio

Figura 5. 5: Ensayos de velocidad con consigna directa

Los escalones aplicados han sido de 700, 1200 y 2000 rpm en ambos ensayos. En la Figura 5.5, se puede observar que la línea discontinua de color negro sería la velocidad de consigna demandada al motor, mientras que la línea azul marina sería la velocidad real obtenida del motor.

Se puede observar que el tiempo de establecimiento de los ensayos con el PID propio es menor que con el PID auto-sintonizado, por lo tanto, la respuesta ante el escalón aplicado es más rápida con el PID propio, consiguiendo así un ancho de banda mayor en este sistema.

A partir de los resultados obtenidos, utilizando el "System Identification Tool" de Matlab, al igual que en el ensayo de corriente, se ha estimado una función de transferencia para cada sistema, que se muestran a continuación:

$$FT_{PID\_auto-sintonizado} = \frac{60{,}49\ s + 5370}{s^2 + 93{,}52\ s + 5301} \tag{14}$$

$$FT_{PID\_propio} = \frac{131{,}5\ s + 6{,}113 \cdot 10^4}{s^2 + 90{,}09\ s + 6{,}128 \cdot 10^4} \tag{15}$$

Las dos funciones de transferencia obtenidas han sido utilizadas, al igual que en el ensayo de corriente, para analizar la respuesta de los sistemas ensayados ante la aplicación de un escalón unitario. Los resultados se muestran en la Figura 5.6.

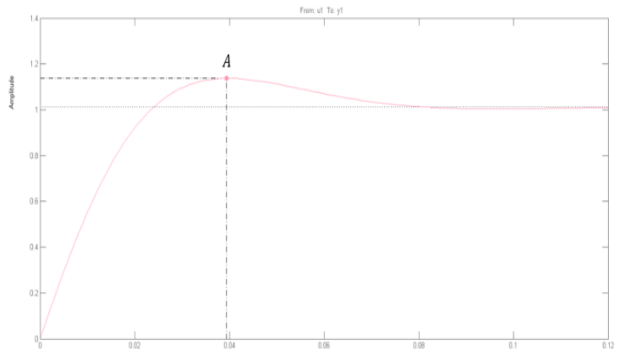

Figura 5.6a: Respuesta a escalón con PID auto-sintonizado

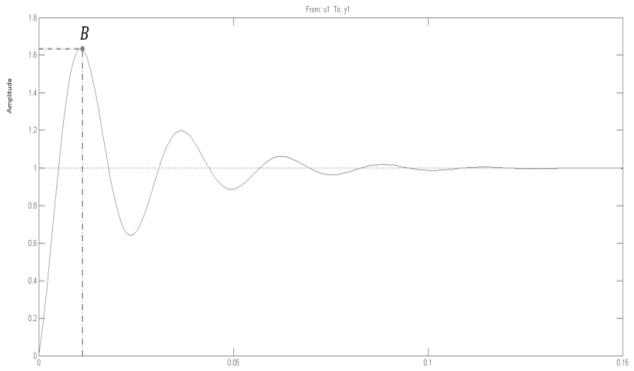

Figura 5.6b: Respuesta a escalón con PID propio

Figura 5. 6: Respuesta de los ensayos de velocidad con consigna directa ante un escalón unitario

Tanto el punto A como el punto B que se pueden ver en las Figuras 5.6a y 5.6b respectivamente, son el valor máximo de amplitud que alcanza la respuesta de cada uno de los sistemas. Gracias a estos dos puntos, se puede calcular el coeficiente de amortiguamiento de ambos sistemas utilizando la ecuación (13).

En el sistema de la Figura 5.6a, el valor de M_P (punto A) es igual a 1,14. Despejando en la ecuación (13), se obtiene un coeficiente de amortiguamiento, ξ, igual a 0,53, por lo que se trata de un sistema subamortiguado.

En el sistema de la Figura 5.6b, el valor de M_P (punto B) es igual a 1,63. Realizando el mismo calculo que anteriormente, se obtiene un coeficiente de amortiguamiento igual a 0,1455, por lo que, al igual que antes, es un sistema subamortiguado.

Además de realizar un estudio de la respuesta de los sistemas ante un escalón unitario, también se ha querido realizar una caracterización de la respuesta frecuencial de ambos, por lo que han sido obtenidos los diagramas de Bode correspondientes, que se muestran en la Figura 5.7.

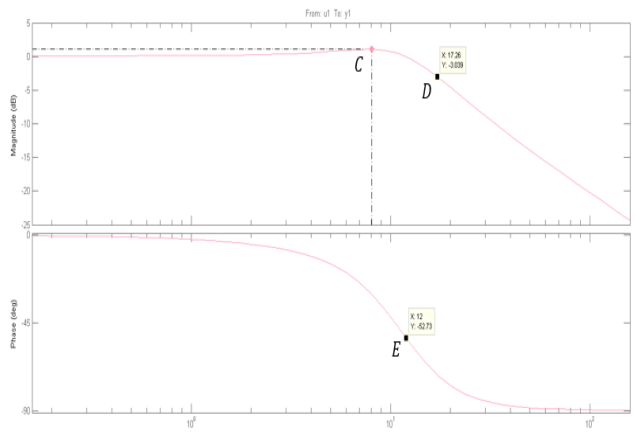

Figura 5.7a: Diagrama de Bode del sistema con PID auto-sintonizado

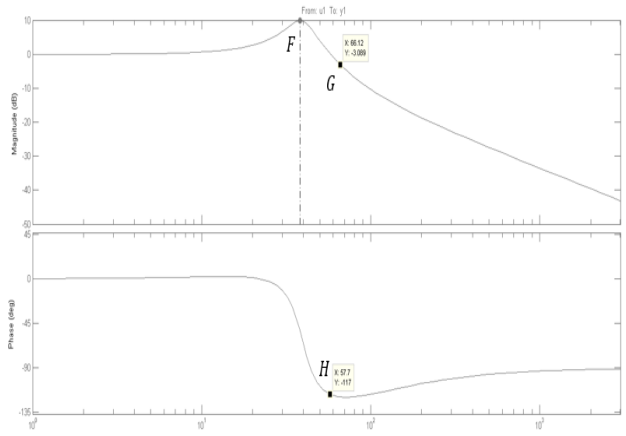

Figura 5.7b: Diagrama de Bode del sistema con PID propio

Figura 5. 7: Diagramas de bode de los ensayos de velocidad con consigna directa

Los puntos C y F de las Figuras 5.7a y 5.7b, respectivamente, son los que indican el pico máximo de amplitud (en decibelios) de ambos sistemas ensayados. Este pico de resonancia existe debido a que los dos sistemas son subamortiguados (ξ<0.707). El sistema con PID auto-sintonizado tiene un valor de pico igual a 1,06 dB, mientras que el sistema con PID propio tiene un valor de pico igual a 9,92 dB. Con estos dos puntos también se podría calcular el coeficiente de amortiguamiento calculado anteriormente, utilizando la siguiente fórmula:

$$M_P = \frac{1}{2\xi\sqrt{1-2\xi^2}} \tag{16}$$

Por otro lado, los puntos D y G nos indican el ancho de banda de los sistemas. En el caso del sistema con PID auto-sintonizado este ancho de banda es igual a 17,26 Hz, que es muy inferior al deseado (80 Hz) para realizar el ensayo de velocidad de forma correcta.

En el caso del sistema con PID propio, el ancho de banda es igual a 66,12 Hz. Se puede ver que este ancho de banda si se acerca mucho más al ancho de banda deseado para el control de velocidad, por lo que se puede decir que los valores del PID calculado de forma experimental son más acertados que los que calculados automáticamente por el programa EPOS Studio.

Por último, para analizar la estabilidad de estos sistemas se ha calculado el margen de ganancia y el margen de fase, cuya definición fue explicada en el ensayo de corriente.

En ambos casos la gráfica no llega a los -180 grados, por lo que en estos sistemas el margen de ganancia es infinito. Esto significa que el sistema no se vuelve inestable aunque se aumente la ganancia en el lazo, al igual que pasaba en el caso de la corriente.

El margen de fase viene representado por los puntos E y H de las Figuras 5.7a y 5.7b respectivamente. En el caso del sistema con PID auto-sintonizado, el margen de fase sería igual a 127,27 grados, mientras que en el sistema con PID propio el margen sería igual a 63 grados.

En ambos casos tanto el margen de fase como el de ganancia son positivos, por lo que los dos sistemas son estables.

5.3.2. Externo

Al igual que se hizo en el control de corriente, ha sido realizado un ensayo de velocidad en el que la generación de la señal de consigna se realiza de forma externa, a través de la FPGA del dispositivo myRIO. Esta señal generada es un seno de amplitud de 5v, pero que a través del programa explicado en la Figura 4.23 se consigue que la velocidad demandada para el motor sea un seno con amplitud de 300 rpm.

Al generar la señal a través de la FPGA, y transmitirla por medio de una salida analógica, conseguimos que los datos no sean adquiridos por parte del dispositivo myRIO por medio del puerto USB, por lo que la generación de la señal de consigna podría tener una frecuencia muy alta, ya que el ancho de banda que se obtiene es grande.

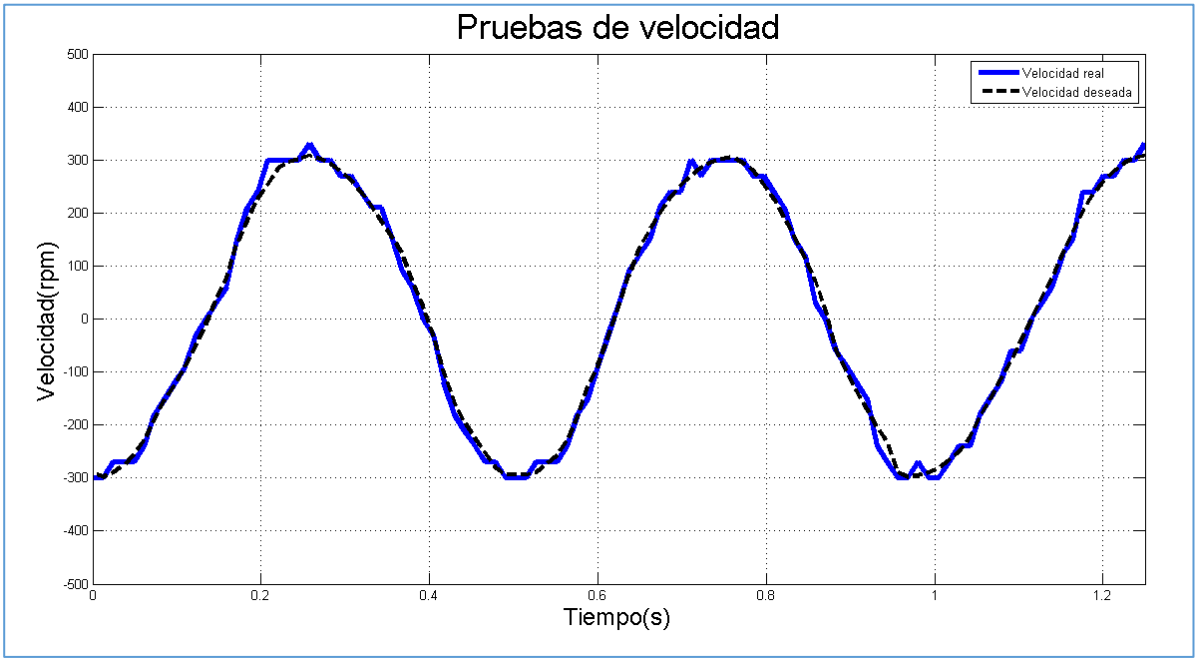

Figura 5. 8: Ensayo de velocidad con consigna externa

En la Figura 5.8, se observa que la señal de consigna generada externamente es seguida aproximadamente de forma correcta por el motor. El ensayo ha sido realizado a una frecuencia baja (2 Hz), ya que, aunque la señal generada si puede muestrearse a frecuencias

muy altas, la adquisición de datos provenientes de la velocidad real del motor, al igual que pasaba en el caso de la corriente, se realiza por medio de USB, y por lo tanto, el ancho de banda es muy inferior al deseado. Si se intentara generar la señal de consigna con una frecuencia superior, la adquisición de datos no sería capaz de mostrarnos unos resultados lógicos, ya que su ancho de banda no es suficiente, por lo que se decidió no realizar más pruebas en este modo de operación.

5.3.3. Auto-generado

Se ha realizado una prueba de velocidad donde la trayectoria que realiza el motor para conseguir la velocidad de consigna se genera de forma automática por el controlador EPOS2 70/10. Esta prueba ha consistido en indicar una velocidad de consigna y el perfil de velocidad y aceleración que debe seguir el motor para alcanzarla. Los resultados obtenidos se muestran en la Figura 5.9.

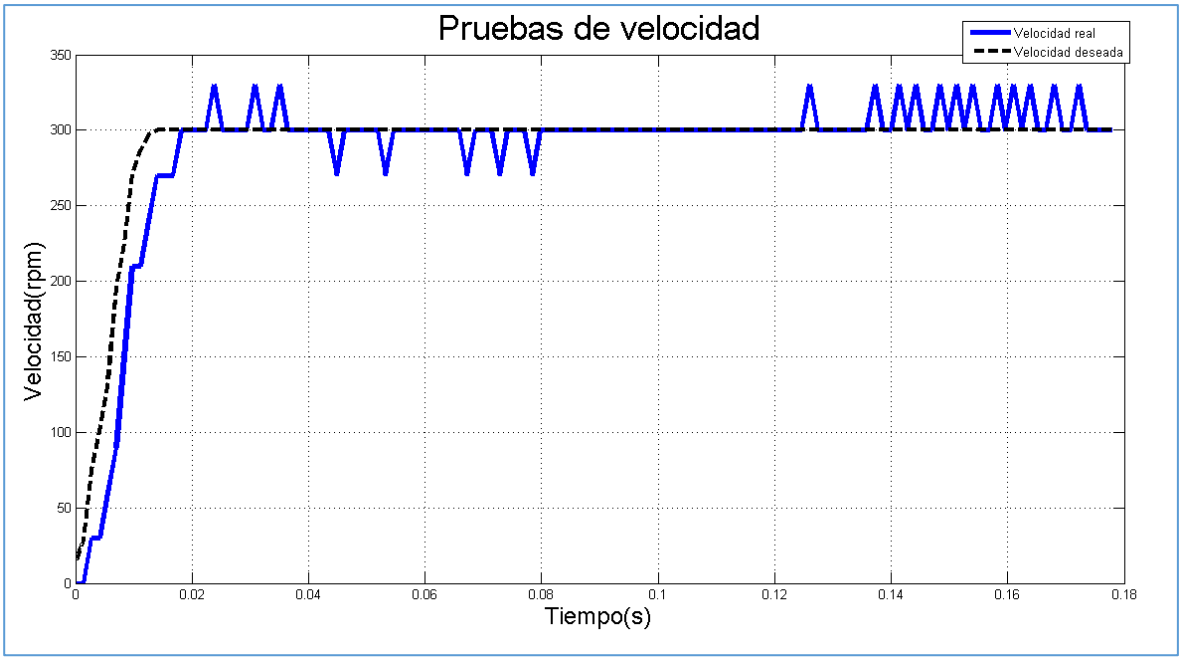

Figura 5. 9: Ensayo de velocidad con trayectoria auto-generada

Como se puede observar en la Figura 5.9, se alcanza la consigna deseada de forma mucho más suave que en el modo de velocidad con consigna directa. Esta característica se debe a que en este modo de operación, la trayectoria que realiza el motor para alcanzar la consigna es calculada internamente por el controlador EPOS2, y por lo tanto, realiza la trayectoria más adecuada dependiendo de la consigna introducida, mientras que en el modo de operación directo, no hay generación de trayectoria por parte del controlador, por lo que el motor

intenta conseguir la consigna lo más rápido posible (al máximo de su capacidad), y por lo tanto, la consecución de la velocidad deseada se realiza de forma mucho más brusca.

5.4. Control de posición

5.4.1. Directo

Aunque el control de posición no ha sido uno de los objetivos del proyecto, se han realizado pruebas para comprobar que el motor es capaz de responder perfectamente ante la aplicación de un escalón de posición.

En este caso, no se ha estimado una función de transferencia para el sistema ensayado, y por lo tanto, tampoco se ha realizado una caracterización de la respuesta frecuencial del mismo, ya que en el control de posición no se dispone de ninguna simulación teórica previa, por lo que no se puede realizar una comparación de los resultados experimentales obtenidos.

Los resultados obtenidos ante la aplicación del escalón de posición se muestran en la Figura 5.10.

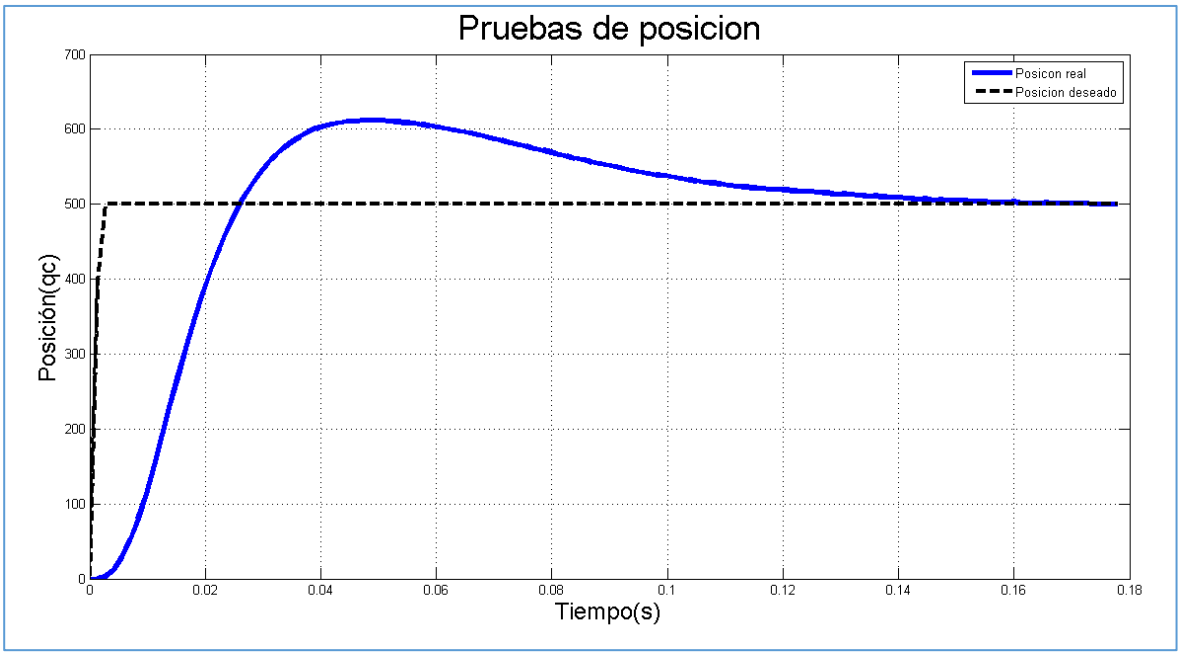

Figura 5. 10: Ensayo de posición con consigna directa

5.4.2. Auto-generado

Al igual que con el control de velocidad, en el control de posición también se ha realizado un ensayo en el que la trayectoria que se genera para conseguir una consigna de posición se calcula de forma interna por el controlador, por lo que, como también pasaba en el caso de la velocidad, la respuesta del mismo será mucho más suave, ya que la trayectoria es la más adecuada para conseguir esa posición. Los resultados obtenidos se pueden observar en la Figura 5.11.

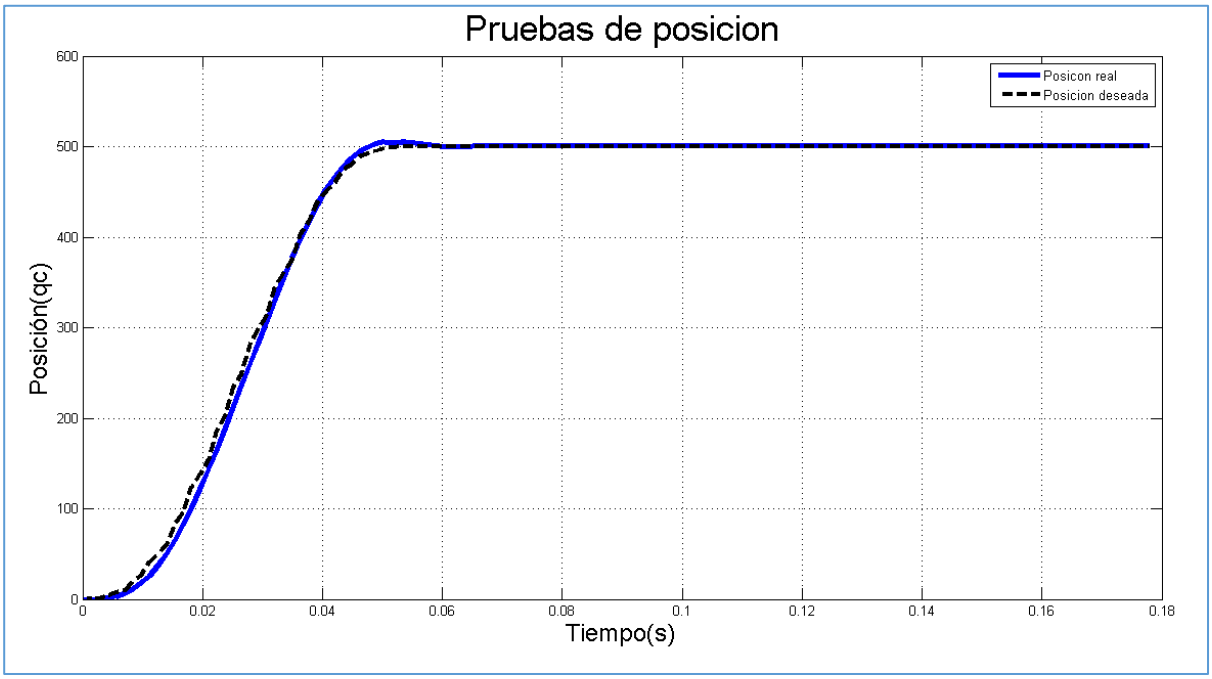

Figura 5. 11: Ensayo de posición con trayectoria auto-generada

Se puede observar en la Figura 5.11 que posición real obtenida es muy similar a la posición deseada, por lo que se podría decir que el control en este modo de operación es bastante satisfactorio, ya que no se produce ningún pico de posición brusco, y la trayectoria real es casi idéntica a la trayectoria deseada.

5.5. Control de fuerza

Para analizar los resultados del control de fuerza, ha sido realizado un ensayo en el que se aplican varios escalones de par para comprobar cómo es la respuesta temporal del actuador ante los mismos. Los resultados obtenidos del ensayo se observan en la Figura 5.12.

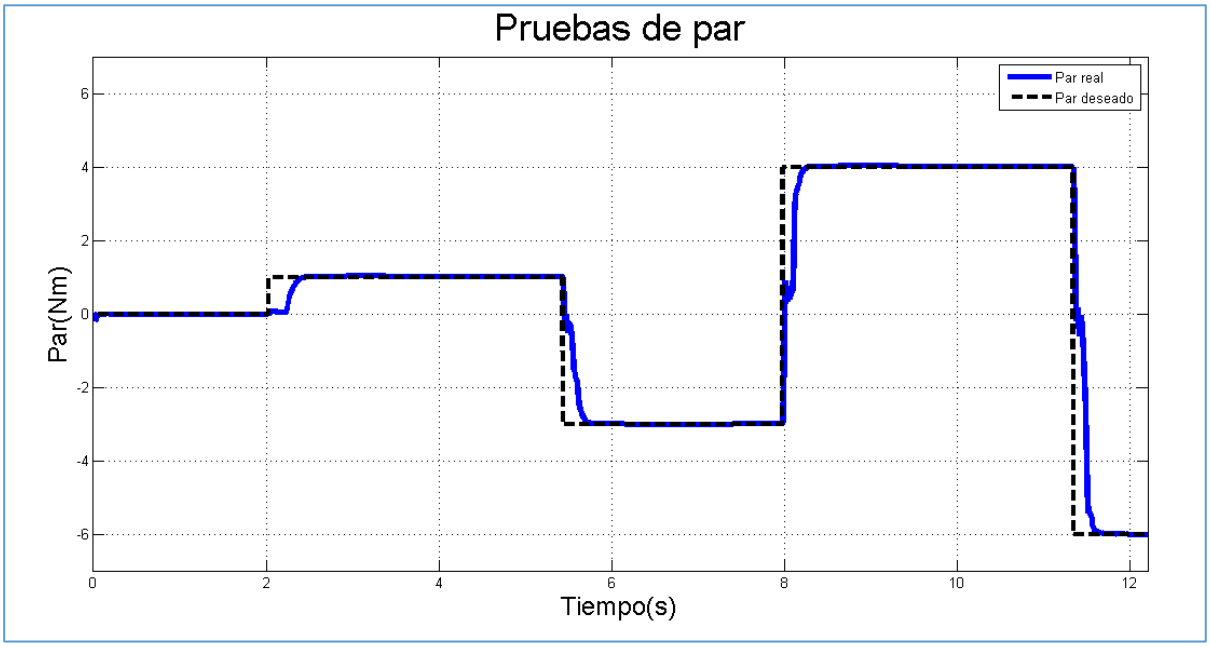

Figura 5. 12: Ensayo para el control de par elástico del actuador

En la Figura 5.12, la línea negra sería el par de consigna deseado para el actuador, y la línea azul sería el par real que se está obteniendo por parte del actuador. Como se puede ver, la respuesta obtenida se estabiliza rápidamente y no tiene picos que sobrepasen el par deseado.

Al igual que en los casos de corriente y velocidad, se ha estimado una función de transferencia para el sistema de la Figura 5.12 a través de la aplicación "System Identification Tool" de Matlab. La función de transferencia obtenida es la siguiente:

$$FT_{par} = \frac{-2603\ s + 1{,}91 \cdot 10^5}{s^3 + 83{,}95\ s^2 + 7802\ s + 1{,}893 \cdot 10^5} \tag{17}$$

La función de transferencia obtenida ha sido utilizada, al igual que en el ensayo de corriente y velocidad, para analizar la respuesta del sistema ensayado ante la aplicación de un escalón unitario. Los resultados se muestran en la Figura 5.13.

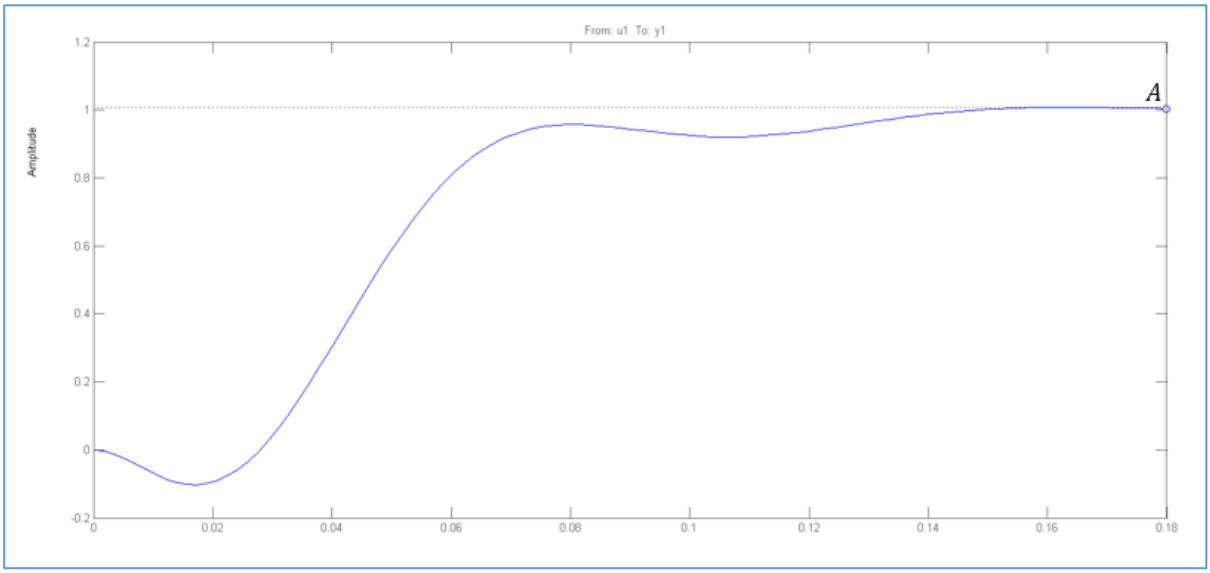

Figura 5. 13: Respuesta del sistema de fuerza ante un escalón unitario

Se puede observar que la respuesta llega a la unidad de forma mucho más lenta que en los ensayos de velocidad y corriente, pero este aspecto es lógico debido a que el ancho de banda teórico calculado para el control de fuerza, es muy inferior al ancho de banda teórico calculado para el control de velocidad y corriente. El punto máximo que alcanza la gráfica (punto A de la Figura 5.13) es 1, por lo que, utilizando la ecuación (13), se obtiene un coeficiente de amortiguamiento igual a 1, y por lo tanto, el sistema es críticamente amortiguado, que como ya se dijo anteriormente, es un sistema frontera entre el subamortiguamiento y el sobreamortiguamiento.

Además de realizar un estudio de la respuesta del sistema ensayado ante un escalón unitario, también se ha querido realizar una caracterización de la respuesta frecuencial del mismo, por lo que ha sido obtenido el diagrama de bode correspondiente al sistema ensayado, que se muestra en la Figura 5.14.

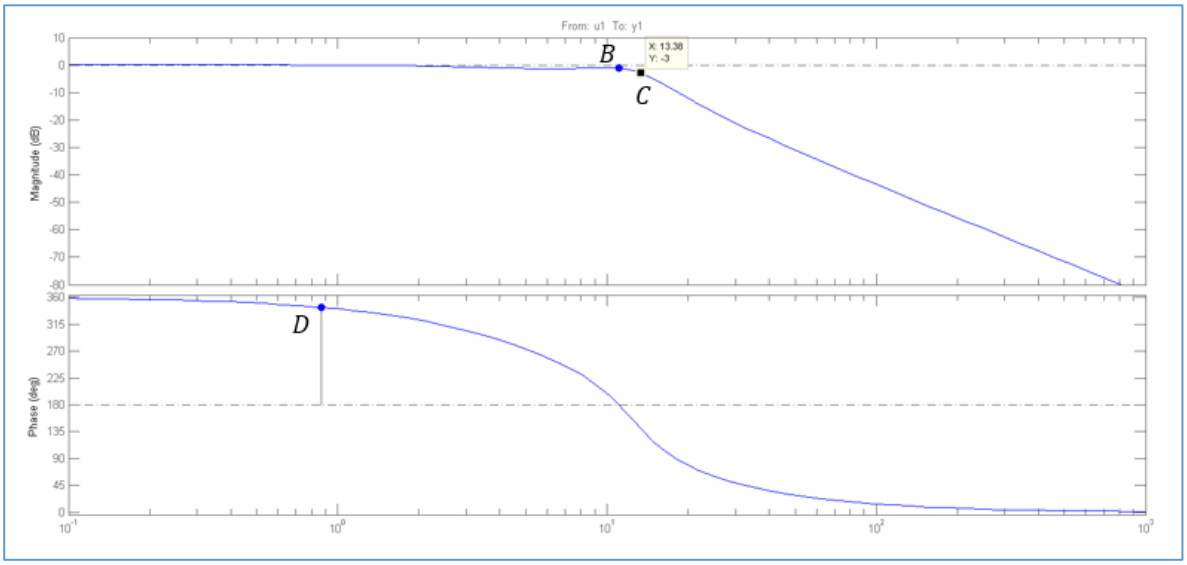

Figura 5. 14: Diagrama de Bode del ensayo de fuerza

El punto C de la Figura 5.14, indica el ancho de banda del sistema ensayado para el control de par. Se observa que el ancho de banda obtenido es igual a 13,38 Hz, un valor que es inferior al calculado en las simulaciones teóricas previas (donde se obtuvo un ancho de banda máximo de 22 Hz), pero que es bastante similar. Teniendo en cuenta que el sistema que se simula previamente es "ideal", los resultados obtenidos en estos ensayos experimentales son muy positivos.

Para analizar la estabilidad del sistema ensayado, es necesario calcular los márgenes de fase y de ganancia, al igual que en el control de corriente y velocidad.

El punto B de la Figura 5.14 sirve para obtener el valor del margen de ganancia, que como ya sabemos, se calcula a través del corte del diagrama de fase con -180 grados. En este caso, este margen de ganancia es igual a 1,12 dB, que aunque no es muy alto, es positivo, que es el objetivo.

El margen de fase viene representado por el punto D de la Figura 5.14, y en este caso es igual a 163 grados.

En el sistema obtenido a través de los ensayos, tanto el margen de fase como el de ganancia son positivos, por lo concluye que el sistema ensayado es estable.

6. CONCLUSIONES

Una vez realizados los diferentes ensayos, ha sido posible llegar a una serie de conclusiones sobre el proyecto que se ha desarrollado, así como los posibles trabajos futuros en los que se podría seguir avanzando en este aspecto.

Uno de los puntos más importantes de este proyecto, era comprobar la capacidad para el dispositivo myRIO de soportar el Real Time, es decir, comprobar si era posible conseguir trabajar en tiempo real utilizando un sistema embebido de bajo coste.

Una vez analizados los resultados, se llega a la conclusión de que, por ejemplo, en los ensayos de corriente y velocidad, el dispositivo myRIO no era capaz de adquirir datos con el ancho de banda ideal, por lo que no se han podido realizar pruebas que fueran largas en el tiempo (que habrían ayudado a obtener unos resultados más concluyentes del comportamiento del actuador), ya que, como los datos reales provenientes del motor, se transmiten al dispositivo myRIO por medio del controlador EPOS2, no es posible conseguir las velocidades necesarias, debido a que la comunicación es de tipo USB, y este aspecto ralentiza el proceso.

Una de las posibles soluciones a este problema, habría sido utilizar otro tipo de comunicación entre ambos dispositivos, como por ejemplo la comunicación CAN Open (que alcanza una mayor velocidad que la comunicación USB). Sin embargo, las librerías que proporciona Maxon, estaban implementadas para la comunicación USB, por lo que, si se hubiera utilizado la comunicación CAN, se habría tenido que diseñar todo el conjunto de librerías que han sido utilizadas, y esto supondría una carga de trabajo muy grande.

Otra posible solución a este problema, habría sido utilizar más sensores que midieran la corriente y velocidad real del motor, y que de esta forma, no fuera necesario adquirir datos que se transmitan por medio del controlador EPOS2. Con esta solución, se conseguiría que la ralentización que produce la comunicación USB se elimine, ya que los sensores irían conectados directamente al dispositivo myRIO, y la adquisición de datos tendría un ancho de banda mucho mayor.

A parte de los problemas encontrados en los ensayos de velocidad y corriente, se puede concluir que en el control de fuerza, los resultados obtenidos han sido muy satisfactorios, ya que, como se puede observar en el capítulo 5, el ancho de banda para la adquisición de datos de par elástico es bastante cercano al calculado de forma teórica, y por lo tanto, en este lazo de control, el dispositivo myRIO es capaz de trabajar en tiempo real. Esto es debido a que el par elástico es obtenido a través de un sensor (encoder magnético) conectado directamente al myRIO, por lo que la comunicación entre ambos es más rápida, ya que no tiene que pasar por el puerto USB.

Por otra parte, otro de los aspectos importantes a analizar en este capítulo, es la facilidad que da la programación gráfica disponible en Labview a la hora de programar los diferentes lazos de control en los que se trabaja en este proyecto. Gracias a este tipo de programación, se construyen programas que son mucho más intuitivos que en cualquier lenguaje de programación tradicional (C, C++, etc.), y además, la programación de los mismos se hace de

forma mucho más sencilla, ya que la carga de trabajo mental también es menos pesada (la sintaxis en los lenguajes de programación tradicionales supone un trabajo muy pesado).

En cuanto al sistema de control utilizado, se decidió realizar un control en cascada. Con este método de control, se están usando dos sistemas de medición (como son el control de corriente y de velocidad, por ejemplo), para manipular un solo elemento final de control (como es el caso de la velocidad, o un lazo más avanzado sería en el caso del par elástico). El controlador primario (lazo externo) o maestro, se corresponde con una variable cuya importancia es mayor que el resto, en este caso particular, la velocidad o el par, mientras que los controladores secundarios (lazo interno) o esclavos, son variables de menor importancia, y su valor afecta a la variable del control primario. Con este tipo de control, se reducen los efectos de las posibles perturbaciones que afecten al control, haciendo así que el sistema sea robusto, y por otra parte, se reducen los efectos de los retardos de tiempo.

El primer lazo que se debe sintonizar sería el lazo interno, y una vez que este lazo esta sintonizado y disponible para su control, se sintoniza el lazo externo, que debe ser más rápido (al menos 3 veces) para conseguir que el lazo interno no afecte en el mismo.

De esta forma, se obtienen tres niveles de control (corriente, velocidad y fuerza), que se podrían controlar tanto de forma separada como conjunta, ya que, gracias a la estrategia de control utilizada, el nivel más interno no afecta al nivel más externo o avanzado, consiguiendo así una optimización del sistema de control.

REFERENCIAS

[1] J. Echávarri Otero, G. Carbone, M. Ceccarelli y J. L. Muñoz Sanz, Criterios para la seguridad en el uso de robots, Cusco, 2007.

[2] A. Giménez Fernández y y otros, Diseño y simulación de un actuador de rigidez variable, 2012.

[3] J. López Martínez y y otros, Acoplamiento flexible de seguridad para la interacción hombre-robot, 2014.

[4] «Demedicina,» 27 Septiembre 2013. [En línea]. Available: http://demedicina.com/pierna-bionica-controlada-por-la-mente/. [Último acceso: Junio 2016].

[5] «El Tiempo,» 14 Mayo 2014. [En línea]. Available: http://eltiempo.com.ve/tiempo-libre/tecnologia/autorizan-venta-de-brazo-robotico-en-estados-unidos/138666. [Último acceso: Junio 2016].

[6] 26 Mayo 2010. [En línea]. Available: http://www.rehabilitacionblog.com/2010/05/los-robots-cuidaran-ancianos-antes-de.html. [Último acceso: Junio 2016].

[7] S. Wolf y G. Hirzinger, A New Variable Stiffness Design: Matching Requirements of the Next Robot Generation, IEEE International Conference on Robotics and Automation, 2008.

[8] T. v. Tilburg, Control of a 1 degree-of-freedom Series Elastic Actuator on a Humanoid Robot, Eindhoven, 2010.

[9] P. F. Lozano Vallés, J. Medina , A. Jardón y C. Balaguer, Mechanism of multiple joint stiffness for the better performance and the safety physical interaction human robot, 2015.

[10] S. Heath, Embedded systems design, Newnes, 2003.

[11] A. Nadal Galiana, Sistemas embebidos, Universidad Politécnica de Valencia, 2005.

[12] «SemanticWebBuilder,» [En línea]. Available: http://www.semanticwebbuilder.org.mx/es_mx/swb/Sistemas_Embebidos_Innovando_hacia_los_Sistemas_Inteligentes_. [Último acceso: Agosto 2016].

[13] «Ministerio de Educación, Cultura y Deporte,» [En línea]. Available: http://www.ite.educacion.es/formacion/materiales/60/cd/02_elestudio/12_elementos_bsicos.html. [Último acceso: Agosto 2016].

[14] «PC actual,» [En línea]. Available: http://www.pcactual.com/noticias/trucos/todo-

sobre-memoria-ddr3-2_8780. [Último acceso: Agosto 2016].

[15] «Soporte-Leyner,» 18 Septiembre 2014. [En línea]. Available: https://soporteleynerpoli.wordpress.com/2014/09/18/memoria-cache/. [Último acceso: Agosto 2016].

[16] «Neoteo,» 31 Julio 2009. [En línea]. Available: http://www.neoteo.com/mantenimiento-y-diagnostico-del-disco-duro. [Último acceso: Agosto 2016].

[17] 25 Febrero 2016. [En línea]. Available: http://computadoras.about.com/od/placa-base/a/Qu-E-Es-La-Bios-Y-Para-Que-Se-Utiliza.htm. [Último acceso: Agosto 2016].

[18] [En línea]. Available: http://www.letsgodigital.org/en/news/articles/story_5415.html. [Último acceso: Agosto 2016].

[19] «InformaticaHoy,» 2015. [En línea]. Available: http://www.informatica-hoy.com.ar/aprender-informatica/La-importancia-del-chipset-en-la-PC.php. [Último acceso: Agosto 2016].

[20] J. M. Cruz, «Simposio Argentino de Sistemas Embebidos,» 16 Agosto 2013. [En línea]. Available: http://www.sase.com.ar/2013/files/2013/09/SASE2013-Tiempo_Real.pdf. [Último acceso: Septiembre 2016].

[21] «National Instruments,» [En línea]. Available: http://www.ni.com/academic/students/learnlabview/esa/gprogramming.htm.

[22] «National Instruments,» 7 Octubre 2015. [En línea]. Available: http://www.ni.com/white-paper/3023/es/. [Último acceso: Septiembre 2016].

[23] «National Instruments,» 5 Diciembre 2013. [En línea]. Available: http://www.ni.com/white-paper/10894/es/. [Último acceso: Agosto 2016].

[24] M. Motor, EPOS2 Positioning Controller Getting Started, 2016.

[25] «Maxon Motor,» [En línea]. Available: http://www.maxonmotor.es/maxon/view/product/control/Positionierung/375711. [Último acceso: Agosto 2016].

[26] «CONTROL TUTORIALS FOR MATLAB & SIMULINK,» [En línea]. Available: http://ctms.engin.umich.edu/CTMS/index.php?aux=Home. [Último acceso: Mayo 2016].

[27] I. Díaz, A. B. Diez y J. M. Guerrero, Prepráctica: Control en cascada, 2007.

[28] J. Cameron Jensen, E. Ashford Lee y S. Arunkumar Seshia, An Introductory Lab in Embedded and Cyber-Physical Systems, 2014.

[29] N. Instruments, USER GUIDE AND SPECIFICATIONS NI myRIO-1900.

[30] «OVTOASTER,» 17 Marzo 2014. [En línea]. Available: http://ovtoaster.com/las-librerias-compartidas-en-linux-y-su-gestion/. [Último acceso: Septiembre].

[31] m. m. control, EPOS Command Library, Octubre, 2014.

[32] «Universidad de Oviedo,» [En línea]. Available: http://isa.uniovi.es/docencia/ra_marina/UCLM_TEMA10.PDF. [Último acceso: Septiembre 2016].