2020

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Implementación de controladores dinámicos para un robot paralelo de 4 grados de libertad sobre un controlador empotrado industrial de tiempo real y FPGA

PROYECTO DE FIN DE GRADO DE INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

AUTOR: JESÚS FERRÁNDIZ ALARCÓN

TUTORA: MIQUEL VALLÉS, MARINA

COTUTOR: PULLOQUINGA ZAPATA, JOSE LUIS

AGRADECIMIENTOS

En primer lugar, agradecer a mi tutora Marina por sus valiosos consejos y orientación, sobre todo teniendo en cuenta las dificultades que ha supuesto este año. En segundo lugar, a mi cotutor y compañeros Jose, Rafa y Pau, por su gran ayuda durante el trabajo. También a todos los compañeros y profesores que me han acompañado a lo largo de estos años. Y, por último, a mi familia, por darme siempre la oportunidad de estudiar y su apoyo incondicional.

RESUMEN

El objetivo de este trabajo es la implementación de controladores para un robot paralelo de cuatro grados de libertad. Para ello, se ha seleccionado un hardware certificado que cumple con los requerimientos temporales y computacionales de la planta y las estrategias de control. Posteriormente, se ha organizado el diseño de la arquitectura del bucle de control de manera que se cumpla siempre el periodo de muestreo y no haya pérdidas de datos. Por último, esta implementación se ha validado a través de una simulación con el modelo del robot.

RESUM

L'objectiu d'aques treball és la implementació de controladors per a un robot paral·lel de quatre graus de llibertat. Per a això, s'ha seleccionat un hardware certificat que cumpleix amb els requeriments temporals i computacionals de la planta i les estratègies de control. Posteriorment, s'ha organitzat el diseny de la arquitectura del bucle de control de manera que es compleixi sempre el període de mostreig i no hi hagi pérdua de dades. Finalment, aquesta implementació s'ha validat mitjançant una simulació amb el model del robot.

ABSTRACT

The objective of this project is to implement controllers for a four degrees of freedom parallel robot. In order to do this, a certificated hardware that meets the timing and computational requirements of the process and the control strategies has been selected. Subsequently, the design of the control loop architecture has been organized so that the sample period is always accomplished and there is not any data loss. Finally, this implementation has been validated through a simulation using a model of the robot.

2020

Memoria: Implementación de controladores dinámicos para un robot paralelo de 4 grados de libertad sobre un controlador empotrado industrial de tiempo real y FPGA

DOCUMENTO 1: MEMORIA

AUTOR: FERRÁNDIZ ALARCÓN, JESÚS

TUTORA: MIQUEL VALLÉS, MARINA

COTUTOR: PULLOQUINGA ZAPATA, JOSE LUIS

TABLA DE CONTENIDO

1. OBJETO

El presente proyecto de fin de grado tiene como objeto desarrollar un control de posición para un robot paralelo de cuatro grados de libertad diseñado por el Instituto de Automática e Informática Industrial de la Universitat Politècnica de València. El desarrollo del control incluye tanto la programación como la selección y el diseño del hardware que compone la unidad de control.

1.1. INTRODUCCIÓN

La ingeniería de control es una disciplina cuyo estudio se centra en proporcionar e implementar teorías para planificar el comportamiento de una variable, generalmente de naturaleza física o química y obtener una respuesta deseable. A esta se le denomina variable controlada y puede se puede tratar de temperatura, posición, nivel de pH, cantidad de glucosa en sangre, etc.

Para modificar la variable controlada, es necesario aplicar una señal denominada señal de control o variable manipulada. Esta variable es aplicada a un actuador, el objeto que permite interactuar con el entorno. Al conjunto del actuador y del sistema a controlar se le denomina planta. Así pues, la variable controlada sería una señal de la propia planta.

Sin embargo, en estos sistemas aparecen perturbaciones con mucha frecuencia, estas se tratan de señales que afectan negativamente al valor de salida de la planta. En el caso del control de velocidad de un vehículo, una perturbación podría ser el viento. Además, para saber con exactitud qué cantidad de señal de control hay que aplicar para obtener una salida deseada, es necesario conocer la planta a la perfección, o lo que es lo mismo, tener un modelo matemático ideal de esta.

Gracias a muchas técnicas de identificación de parámetros de procesos es posible obtener modelos que se acerquen con una gran exactitud a la realidad, pero no es posible obtener un modelo perfecto. Debido a esto, la variable controlada puede tener un ligero error al esperado.

Por estos motivos, hay sistemas que requieren de un control realimentado. Se trata de un control en bucle cerrado en el que, mediante un sensor se mide la variable controlada y se aplica una señal de control que es función del error que existe entre el valor que se desea y el valor medido.

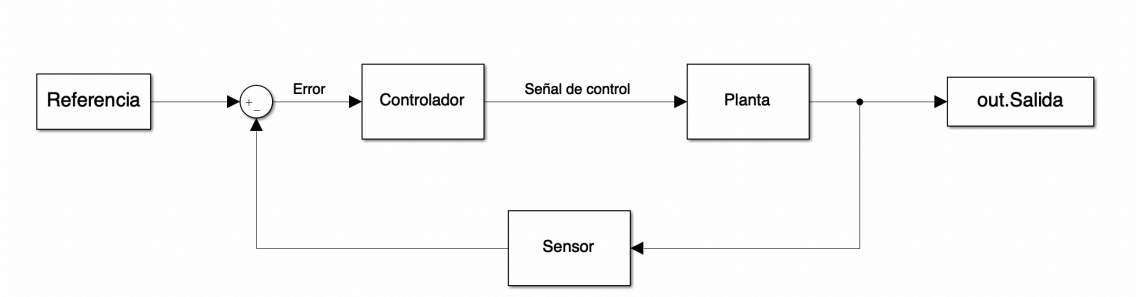


Figura 1: Bucle de control simple

Gracias a esta realimentación, el sistema puede ser prácticamente insensible frente a perturbaciones y a las variaciones o errores en el modelo de los parámetros del sistema. Como desventaja, un bucle cerrado añade más complejidad al diseño de los elementos que lo integran, sobre todo del controlador, el elemento cuya finalidad es aplicar una señal de control a partir del error medido tal que se obtenga una salida deseada disminuyendo el propio error. [1]

Así pues, una rama muy importante de la ingeniería de control es el diseño y la implementación de controladores dinámicos, traducidos en algoritmos e implementados en sistemas computacionales capaces de realizar operaciones matemáticas. Los controladores dinámicos se diseñan en función del tipo de respuesta que se desea y su forma.

El control tiene numerosas aplicaciones en campos muy diversos siendo uno de los más relevantes el control de robots. Gracias a la implementación de bucles cerrados y de la teoría de control es posible controlar las trayectorias que realizan los elementos del robot.

En este proyecto de fin de grado, se ha trabajado con un robot constituido por 4 actuadores, en este caso motores, que funcionan de forma conjunta para mover una plataforma en el espacio. Esta plataforma se utiliza para la rehabilitación de miembros inferiores como tobillos y rodillas a través de proporcionar movimientos que favorezcan la recuperación de tendones y articulaciones.

Figura 2: Robot paralelo de 4 grados de libertad. Fuente: https://imbio3r.ai2.upv.es/lista-galerias

1.2. FINALIDAD Y OBJETIVOS DEL PROYECTO

La finalidad del presente trabajo de fin de grado consiste en la implementación de controladores dinámicos para un robot de 4 grados de libertad. Para ello, es necesario establecer una serie de objetivos:

- Seleccionar un hardware cuyas prestaciones cumplan con los requerimientos del control.
- Realizar las conexiones entre los actuadores y sensores del robot con los elementos hardware que integran el control y la alimentación.
- Diseñar una arquitectura de control apropiada para el robot y el hardware.
- Implementar los controladores dentro de la arquitectura de control.
- Validar la implementación mediante simulación.

2. ANTECEDENTES Y JUSTIFICACIÓN

2.1. JUSTIFICACIÓN DEL PROYECTO

Para este robot paralelo, ya se han desarrollado e implementado controladores a partir de un ordenador y tarjetas de adquisición de datos. Sin embargo, en este momento se pretende obtener una patente o poder comercializarlo en el futuro. Por tanto, es necesario emplear un hardware y software que cumplan con las normativas y tengan las certificaciones necesarias para ello.

Así pues, este proyecto tiene como objetivo implementar los controladores ya diseñados para este robot en una nueva unidad de control cuyo hardware esté certificado.

2.2. MARCO TEÓRICO

2.2.1. INTRODUCCIÓN A LA ROBÓTICA

Un robot puede ser definido como un sistema que permite controlar los diferentes grados de libertad de un cuerpo rígido respecto a una base fija. Este cuerpo rígido en robótica se denomina efector final y sus grados de libertad no pueden superar 6, tres movimientos traslacionales y tres rotatorios. Así pues, se puede describir su posición y su orientación.

El sistema físico que permite el movimiento y el dinamismo es un sistema mecánico definido por cadenas cinemáticas. Existen dos tipos de cadenas, ambas aplicables a la arquitectura de los robots. Por un lado, se tienen las cadenas cinemáticas simples o cerradas. En estas, cada cuerpo tiene un grado de conexión menor o igual a dos. Por otro lado, también existen las cadenas cinemáticas cerradas, que se forman cuando uno de los elementos distinto de la base fija tiene un grado de conexión mayor o igual a tres.

2.2.2. ROBOTS SERIE

Un robot serie es uno formado por una cadena cinemática abierta en la que sus elementos tienen un grado de conexión igual a dos, exceptuando la base y el efector final, cuyos grados son uno. Estos se han traducido en brazos robóticos con una estructura muy similar a la del brazo humano. Cada cuerpo rígido que lo forma está unido al siguiente y al anterior por medio de articulaciones de un solo grado de libertad, es decir, permiten un movimiento traslacional o rotatorio con respecto de un solo eje.

Figura 3: Brazo robótico. Fuente: https://new.abb.com/products/robotics/es/robots-industriales/irb-4400

En la industria son los más utilizados, pero tienen dos inconvenientes que los hacen inviables para cierto tipo de aplicaciones. Tienen una relación carga/potencia muy pequeña, es decir, no pueden manipular cargas muy superiores a su propio peso. La segunda desventaja es su precisión en la posición del efector final. Para el control de los robots es necesario incorporar sensores, no obstante, estos pueden presentar errores en las medidas, que pueden ser pequeños, pero en una arquitectura serie, un error de centésimas de grado en una de las primeras articulaciones, se puede traducir en milímetros de error en la posición del efector final. A este error se le añaden las deformaciones físicas que se pueden producir en el interior de los elementos y articulaciones.

Por estos motivos, para tareas en las que se requiera una alta precisión, así como manipular cargas muy pesadas, estos robots serie son inapropiados.

2.2.3. ROBOTS PARALELOS

Aquellos robots formados por una estructura mecánica constituida por una cadena cerrada en la que el efector final se une a la base por al menos dos cadenas cinemáticas independientes se denominan paralelos. Generalmente, el efector final de este tipo de robots es una plataforma que es controlada en el espacio. Gracias a esta arquitectura, la carga que soporta el efector final se redistribuye entre cada unión que lo conecta con la cadena cinemática sostenida en la base fija, por ello, solo soportan una fracción del peso de la carga. Los accionamientos de potencia están directamente conectados con la plataforma y al poder actuar de manera simultánea pueden manipular cargas muy superiores. Así, los robots paralelos tienen una relación carga/potencia mucho mayor que los serie.

Figura 4: Robot paralelo. Fuente: http://www.mectrol.com.br/mectrol/pt/produto/visualizar/codproduto/161/robo-delta-rd403.html

Además de esta ventaja, los errores de los sensores internos afectan a la posición de la plataforma considerablemente menos que en los robots de cadena cinemática abierta. Concretamente, si todos los sensores presentan el mismo error, el cálculo de la posición de la plataforma tendrá un solo error en su eje vertical cuya amplitud será similar al error presente en un único sensor. Por último, en comparación con cualquier otro tipo de estructura, pueden presentar elevadas velocidades de operación. [2]

Sin embargo, también presentan inconvenientes muy relacionadas con el control. La cinemática de los robots paralelos es mucho más compleja y en ocasiones, se plantea la cuestión de utilizar sensores redundantes a pesar del error en sus medidas para poder establecer un lazo de control cerrado. Además de esto, no existe un modelo dinámico general para estos robots a diferencia de la configuración serie que sí que tiene. Esto dificulta el uso de algoritmos de control de carácter general. A esto se le añade que el problema de resolver las configuraciones singulares es mucho más complejo. En estas configuraciones, se pierden grados de libertad del efector final que se traducen en un control prácticamente imposible del mismo y, por tanto, es necesario conocer cuándo el robot está próximo a una para poder evitarlas.

En este proyecto, se plantea precisamente resolver el problema del control de posición de un robot paralelo para el cual es necesario plantear una serie de problemas que surgen a partir de este.

2.2.3.1. PROBLEMA DE LA CINEMÁTICA INVERSA

El modelado cinemático inverso consiste en la obtención de la posición de las articulaciones dadas las coordenadas del efector final. A diferencia de los robots serie, para los paralelos es un problema más sencillo que se puede resolver con consideraciones geométricas de carácter general. [3]

Las ecuaciones que definen la cinemática inversa del robot paralelo del proyecto se pueden resolver de manera analítica sin necesidad de recurrir a métodos numéricos.

2.2.3.2. PROBLEMA DE LA CINEMÁTICA DIRECTA

Este problema consiste en calcular las coordenadas de la plataforma a partir de la posición de los actuadores del robot. Para el caso de los robots paralelos, este es un problema más complicado que puede llevar a diferentes soluciones analíticas. Una forma de abordarlo es a partir de la notación de Denavit-Hartenbert, pero se pueden llegar a obtener sistemas no lineales difíciles de resolver analíticamente.

Es por ello que se recurre a métodos numéricos como el método de Newton-Raphson. Este es un método computacionalmente sencillo que resulta en una solución única. No obstante, se debe asegurar su convergencia y requiere una aproximación tan cercana como sea posible al valor de la solución. [4]

2.2.4. CONTROL DE LOS ROBOTS

Una vez resueltos estos problemas, es necesario implementar los algoritmos en un sistema computacional capaz de controlar el robot. Este sistema debe procesar las señales de los sensores, debe tomar decisiones y, en consecuencia, proporcionar una señal a los actuadores. El sistema encargado de esta tarea, en robótica se denomina unidad de control y, dependiendo de la aplicación, existen dos tipos de unidades.

Los robots comerciales tienen una unidad de control cerrada, es decir, no se pueden modificar las estrategias de control que ya llevan implementadas, o es difícil incorporar sensores distintos a los que ofrece el fabricante como soportados. En cambio, los robots con una unidad abierta, sí que ofrecen la posibilidad de cambiar las estrategias de control y de implementar otros tipos de sensores.

2.2.5. PROCESADORES DE TIEMPO REAL

Estos sistemas computacionales es posible que estén operando en entornos o aplicaciones en las que es crítico realizar una tarea, generalmente de naturaleza física, en un tiempo determinado. Entonces, se podrán decir que forman parte de un sistema de tiempo real que se puede definir como un sistema de proceso de la información que tiene que responder a un estímulo de entrada generado externamente en un período finito y especificado.

Dependiendo de lo crítico que sea la aplicación, se pueden distinguir entre dos sistemas de tiempo real, estrictos (hard) y no estrictos (soft). En los estrictos es absolutamente esencial que las respuestas se produzcan en el tiempo especificado, mientras que, en los no estrictos, aunque no se cumplan los tiempos establecidos, el sistema puede seguir funcionando de manera correcta.

La información de estos sistemas es manipulada por procesadores de tiempo real que tienen características propias del tiempo real. Son deterministas, si se especifica que una acción debe hacerse en cierto tiempo, se hará y en el caso de que no se pueda satisfacer, es capaz de detectar el error y de responder. Además, es capaz de asegurar la concurrencia de procesos, aunque estos al ser de naturaleza física existan en paralelo. También deben ser extremadamente seguros y fiables, tanto el hardware como el software que manejan. Se deben de añadir

interfaces de usuario muy cuidadas que minimicen el error humano. Finalmente, al tratarse de aplicaciones de ingeniería, es necesario que puedan manipular números reales o de coma flotante. [5]

2.2.6. FPGA (FIELD PROGRAMMABLE GATE ARRAY)

Las FPGA son ciruitos integrados VLSI (very-large-scale integration) que están formados por un elevado número de bloques lógicos simples interconectados entre sí. Estos bloques se pueden programar internamente y las conexiones entre los mismos. Dependiendo del fabricante, la arquitectura puede ser distinta, pero tienen características en común. Todas están formadas por bloques lógicos configurables denominados CLB (Configurable Logic Block), entre estos existen líneas de interconexión configurables y alrededor de la matriz incorporan bloques de entrada y de salida configurables. Además, pueden tener bloques integrados tales como memorias RAM, multiplicadores, administradores de reloj, etc.

Actualmente, la mayor parte de estos dispositivos están basados en la tecnología de programación SRAM, es decir, se pueden reconfigurar tantas veces como sea necesario. La programación generalmente se lleva a cabo mediante lenguajes de descripción de hardware como VHDL y VERILOG, los más utilizados. Una vez escrito el código, este se sintetiza en estructuras lógicas y estas se traducen en subbloques preparados para ser implementados. Así pues, se colocan físicamente y se configuran las conexiones entre ellos.

Trabajar con estos dispositivos forma parte de un diseño semipersonalizado, no es necesario conocer los detalles internos del chip, así utilizar las FPGA tienes ventajas como: Poder realizar rediseños de una forma rápida y sirven de prototipos para diseñar hardware más personalizado. Además, dotan de una arquitectura paralela, de una gran flexibilidad y permiten la implementación de grandes funciones complejas. Como inconvenientes, tienen una menor velocidad que los circuitos con aplicaciones más específicas, el circuito puede presentar retrasos que dependen de la manera en que se hayan implementado físicamente y requieren software más complejo que las CPLD. [6]

2.3. ESTUDIOS TÉCNICOS PREVIOS

El Instituto de Automática e Informática Industrial ya ha llevado a cabo un amplio estudio teórico del modelado del robot y de la resolución teórica de la cinemática inversa y directa de este robot. Las ecuaciones de estos dos problemas se utilizarán a lo largo del proyecto cuando sea necesario.

También se ha implementado una arquitectura de control previa basada en un ordenador industrial equipado con tarjetas de adquisición de datos utilizando el middleware OROCOS de tiempo real y ROS. [7]

3. FACTORES A CONSIDERAR

Para desarrollar este proyecto, hay que tener en cuenta una serie de necesidades que se van a clasificar en factores técnicos y de gestión. Estos se van a explicar a continuación.

3.1. FACTORES TÉCNICOS

- La unidad de control que se escoja debe ser compatible con el robot de cuatro grados de libertad del proyecto. Debe tener la capacidad de controlar cuatro motores de corriente continua a través de salidas analógicas y también se requiere que tenga las suficientes entradas para leer las señales procedentes de 4 encoders con dos canales cada uno. En el caso de que se utilice un sistema cuyo rango de tensiones no sea compatible con los sensores, se deberán acondicionar las señales necesarias.
- Se debe utilizar un controlador con el que se puedan implementar algoritmos de control más complejos que los PID estándares, en los que se necesitan realizar operaciones matriciales.
- Se requiere que la unidad de control a implementar sea abierta, es decir, que ofrezca la posibilidad de implementar distintas estrategias de control y poder utilizar varios tipos de sensores, en este caso, sensores de fuerza para tener en cuenta la resistencia que ejerce el paciente al movimiento. Esta unidad también debe permitir el desarrollo de interfaces de usuario sencillas de utilizar.
- El controlador debe garantizar requerimientos y características de tiempo real.
- Para el control del robot es necesario que el bucle de control tenga un tiempo de muestreo de como máximo 10 milisegundos.
- Con el fin de estudiar la respuesta del controlador, se deben poder monitorizar los datos de cada iteración del bucle de control sin ninguna pérdida. Estos datos son: el tiempo en el que se ha ejecutado, la posición de los actuadores y la acción de control utilizada.

3.2. FACTORES DE GESTIÓN

- Se debe utilizar un hardware que permita poder ser comercializable. Para ello, debe cumplir con toda la normativa necesaria e incorporar la Conformidad Europea, en otras palabras, ha de tener el marcado CE.

4. SOLUCIONES ALTERNATIVAS

4.1. PLC INDUSTRIAL

Los controladores lógicos programables (PLC) son herramientas muy robustas que permiten controlar una gran cantidad de actuadores y obtener señales de múltiples sensores, ambos tanto de naturaleza digital como analógica.

Además, estos cuentan con capacidades de tiempo real, gracias a su procesador que está destinado para ello. Sin embargo, la mayoría de estos solamente ofrecen la posibilidad de implementar algoritmos de control basados en PIDs estándares. Para este proyecto es necesaria una mayor capacidad de cómputo y un equipo en el que sea posible implementar controladores más complejos.

4.2. MICROCONTROLADOR

Otra solución podría ser utilizar un microcontrolador. Actualmente, muchos microcontroladores tienen una arquitectura que soporta un sistema operativo de tiempo real. Un ejemplo es la arquitectura ARM, de la cual se basan microcontroladores con buenas prestaciones como los de la familia STM32 de STMicroelectronics.

Como ventajas, estos microcontroladores son sistemas de bajo coste, pequeños y compactos que además consumen muy poca energía. Sin embargo, las señales de entrada y de salida de los microcontroladores son de bajo voltaje, por lo que sería necesario incorporar una etapa para amplificar las salidas analógicas que aplican la acción de control en los motores y también habría que atenuar la señal procedente de los encoders de 12 Voltios.

No obstante, la mayor desventaja al utilizar estos dispositivos es que tienen una baja capacidad de cómputo. [8] Esto resultaría en limitaciones a la hora de implementar controladores más avanzados que un PID. Entre los STM32, los productos F7 son los que mejores prestaciones ofrecen. Entre estas prestaciones, se tiene una memoria SRAM de 512 Kbytes, esta sería insuficiente para manejar las cantidades de datos que se necesita en un algoritmo de control avanzado. En relación con la baja capacidad de cómputo, muchos microcontroladores no incorporan un coprocesador matemático, lo que es fundamental para realizar operaciones en coma flotante y en casos más avanzados, operaciones trigonométricas y exponenciales. [9] [10] [11]

4.3. ORDENADOR CON TARJETA DE ADQUISICIÓN DE DATOS

Se podría utilizar un ordenador junto con una tarjeta de adquisición de datos que se ajuste a las necesidades de entradas y salidas del robot, esta solución permitiría cumplir con todas las necesidades de cómputo al poder utilizar un ordenador con los componentes cuyas prestaciones satisfacen esta necesidad.

Sin embargo, no se tendría un sistema compacto y no se puede garantizar la Conformidad Europea para el hardware del ordenador completo.

4.4. CONTROLADOR EMPOTRADO DE TIEMPO REAL

Como última alternativa se propone utilizar un sistema empotrado. Estos sistemas tienen prestaciones similares a las de un ordenador con la ventaja de que son muy compactos. Tienen una mayor capacidad de memoria RAM y de disco duro que los microcontroladores y que los dispositivos PLC. Además, pueden incluir procesadores que implementen sistemas operativos de tiempo real y cumplir con los requerimientos y seguridad de tiempo real.

Una serie de desventajas con respecto a los microcontroladores es el mayor consumo que los sistemas empotrados como estos conllevan y un coste mucho mayor.

Un ejemplo de estos sistemas son los dispositivos Compact RIO de National Instruments. Este es un sistema empotrado que incorpora un procesador de tiempo real y una FPGA a la que se le pueden introducir múltiples módulos de entrada y salida dependiendo de la necesidad. Por tanto, la modularidad que tienen con respecto al microcontrolador y a los PLC es mayor. De esta manera, se podrían implementar distintas estrategias de control y varios tipos de sensorización como añadir sensores de fuerza.

Por otro lado, estos productos se programan con el lenguaje LabVIEW de National Instruments, el cual ofrece la posibilidad de realizar interfaces de usuario de manera sencilla y eficaz.

En último lugar, los productos Compact RIO han sido probados y validados con el fin de además de ser seguros, cumplir toda la normativa necesaria para poder tener la Conformidad Europea.

5. DESCRIPCIÓN DETALLADA DE LA SOLUCIÓN APORTADA

5.1. SOLUCIÓN HARDWARE

Se ha optado por utilizar en este proyecto el controlador empotrado Compact RIO de National Instruments, debido principalmente a las mayores prestaciones con respecto a los microcontroladores y dispositivos PLC en el sentido de potencia de cálculo, debido también a su modularidad y a la garantía de tener un producto ya validado y certificado, cuestión que no se puede asegurar con un ordenador.

En los siguientes apartados se van a exponer las características técnicas del controlador y sus módulos además de las conexiones realizadas con este y el robot.

5.1.1. CONTROLADOR NI COMPACT RIO

El controlador Compact RIO es un sistema inteligente de control de alto rendimiento y adquisición de datos que incorpora un procesador de tiempo real y una FPGA reconfigurable que, además, funciona de interfaz entre el procesador y los módulos de entradas y salidas del controlador.

Los productos Compact RIO además cuenta con certificaciones estándares en la industria, así como con la Conformidad Europea al cumplirlas siguientes directivas europeas: 2014/35/EU Seguro con bajos voltajes, 2014/30/EU compatibilidad electromagnética (EMC) y 94/9/EC poder trabajar en atmósferas potencialmente explosivas (ATEX). [12]

Así pues, la Compact RIO es un sistema embebido o empotrado cuya finalidad es el control de sistemas físicos en condiciones en las que la respuesta debe ser determinista y el tiempo es crítico. Otra característica con la que cuenta es modularidad. Existe una gran cantidad de productos que permiten personalizar el controlador dependiendo entre otros de la velocidad de procesado, la temperatura de trabajo, el tipo de señales que es necesario medir.

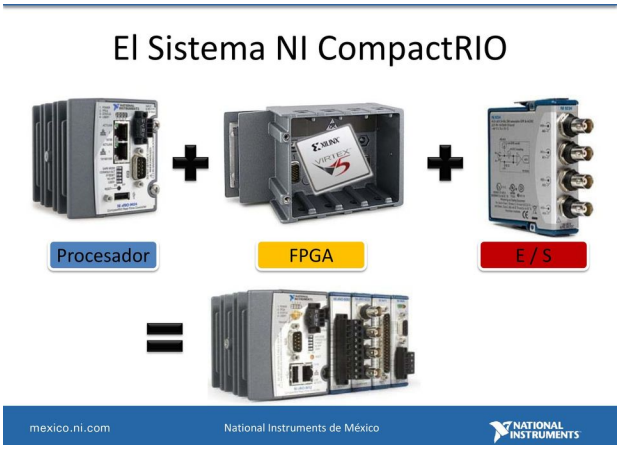


Figura 5: Sistema NI CompactRIO. Fuente: National Instruments de México, mexico.ni.com

La arquitectura global del sistema sería la siguiente, el procesador de tiempo real obtiene información de la FPGA a través de un bus PCI, a su vez, la FPGA obtiene información de los módulos de entradas y salidas capaces de leer sensores o actuar sobre salidas, esta arquitectura se puede ver en la siguiente imagen:

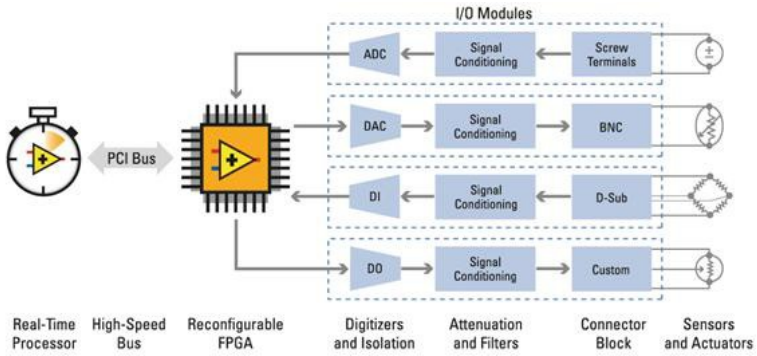


Figura 6: Arquitectura del sistema RIO. Fuente: https://www.ni.com/es-es/innovations/white-papers/13/the-labview-rio-architecture--a-foundation-for-innovation.html

5.1.1.1. PROCESADOR DE TIEMPO REAL

El controlador de tiempo real que se ha utilizado es la cRIO-9004. Este es un sistema empotrado para Compact RIO. El controlador cuenta con las siguientes características: CPU de 195 MHz, 64 MB DRAM y 512 MB de memoria no volátil.

El procesador tiene instalado un sistema operativo de Tiempo Real, concretamente el sistema Phar Lap ETS.

Figura 7: Controlador cRIO-9004. Fuente: https://www.ni.com/es-es/support/model.crio-9004.html

5.1.1.2. CHASIS, FPGA

Una vez elegido el procesador del controlador, es necesario añadir un chasis para poder incorporar los módulos de entradas y salidas que se requieran. Estos chasis además tienen implementados una FPGA que es capaz de acceder a estos módulos. En este caso, se ha seleccionado el chasis 9104.

Este chasis en concreto tiene capacidad para ocho módulos de la serie C de National Instruments, estos se introducen en los compartimentos indicados por el número 4 en la imagen siguiente:

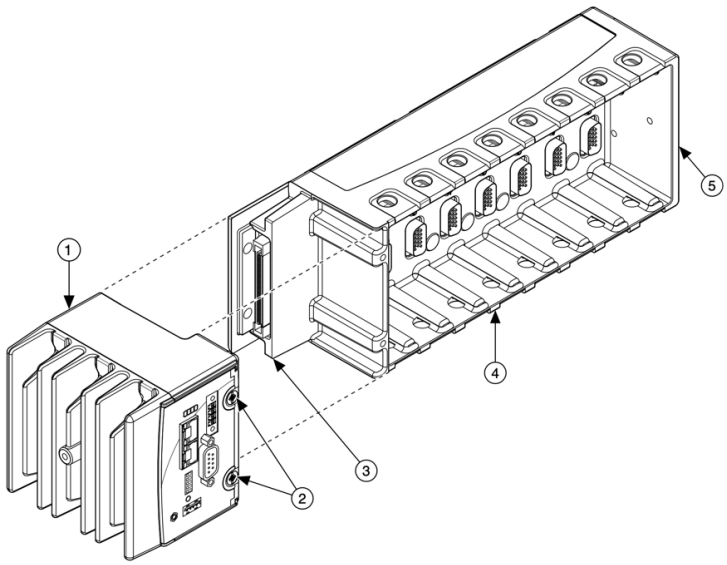


Figura 8: Controlador y chasis. Fuente: User manual and specifications NI cRIO-9101/9102/9103/9104

Como se mencionó anteriormente, el sistema incluye una FPGA Xilinx Virtex II XC2V3000-4FG6761 que cuenta con tres millones de celdas. [13]

5.1.1.3. MÓDULOS DE ENTRADA Y SALIDA

Al chasis cRIO 9104 se le pueden incorporar módulos de la serie C de National Instruments, se puede elegir entre más de 100 módulos de control, adquisición y comunicación. Los que se han incorporado han sido los siguientes:

- NI 9201:
  Este módulo cuenta con ocho entradas analógicas cuyo rango se encuentra entre los +10 V y -10 V y cuenta con una rapidez de muestreo de 500kS/s.
  Cada entrada es escaneada, aislada y muestreada por un único ADC. Se pueden leer todos los canales a la vez ya que cada canal de entrada tiene una ruta de señal independiente y un ADC. La resolución de la medida es de 12 bits. [14]
  Se han utilizado dos módulos 9201 que se ha conectado en los terminales 4 y 6.
- NI 9263:
  El módulo 9263 contiene 4 salidas analógicas entre un rango de +10V y -10V con una resolución de 16 bits y una velocidad de muestreo de 100kS/s/ch. [15] Se han incorporado dos módulos NI 9263 a los terminales 3 y 5.

- NI 9421:
  Se ha utilizado un módulo 9421 que consiste en 8 canales de entrada digitales del tipo sinking, es decir, proporciona una conexión a tierra para la carga. Estos canales son compatibles con señales de 24 V. [16]
  Se ha conectado este módulo al terminal 2 del chasis 9104.

- NI 9474:
  Este es un módulo de ocho salidas digitales compatibles con un rango de voltaje comprendido entre 5 a 30 V. [17] Se ha incorporado este módulo al terminal 1.

Figura 9: Módulos de la serie C. Fuente: https://mafiadoc.com/compactrio-developers-guide_59eedbb01723ddb706530585.html

5.1.2. INTERCONEXIONADO DE ELEMENTOS

En este apartado se va a explicar la conexión entre los elementos del controlador Compact RIO con el cuadro de control. Para simplificar el conexionado, se ha diseñado un circuito impreso, que, además, amplifica las señales procedentes de los encoders (+12 V) para que sea compatible con el módulo de entrada digital de +24 V de la Compact RIO.

El esquema del electrónico es el siguiente:

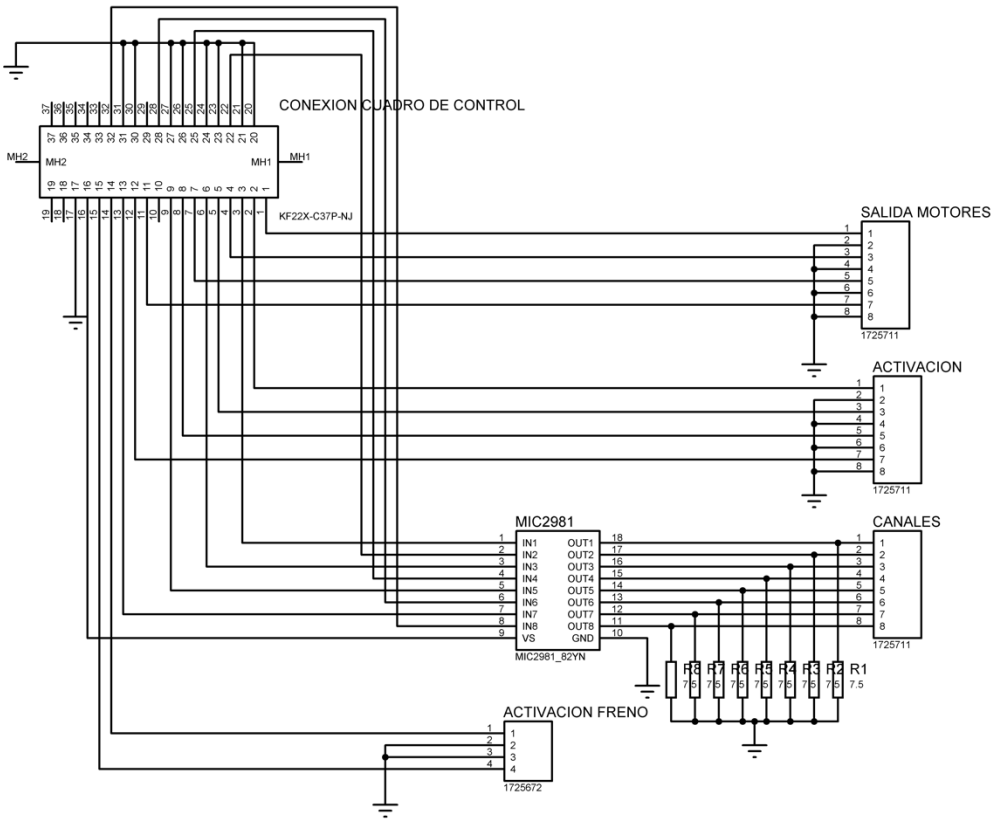


Figura 10: Diagrama eléctrico del circuito impreso

Este circuito impreso permite que gracias al conector KF22X-C37P-NJ y la redistribución del cableado, las señales de control de la Compact RIO puedan conectarse con el tablero de control del robot.

Además, como se ha mencionado anteriormente, ha sido necesario amplificar las señales de los canales de los encoders (0 V en estado bajo y 12 V en estado alto) para que sea compatible con la entrada digital del módulo NI 9421 (0 V en estado bajo y 24 V en estado alto). Esta amplificación se ha realizado con el circuito integrado MIC2981. Este es un controlador de corriente de 8 canales pensado para conmutar cargas TTL, CMOS o PMOS. Los niveles altos de las entradas digitales pueden comprender un rango de entre 5 V a 15 V.

Las resistencias se han diseñado teniendo en cuenta la corriente máxima de salida del MIC2981 [18] y las características de los niveles lógicos del módulo NI 9421 [17]. Por una parte, para asegurar que funcione el circuito integrado correctamente, se debe cumplir que:

$$R > \frac{Vs}{Ic_{máx}} > \frac{24\ V}{500\ mA} > 48\ \Omega$$

Por otra parte, para asegurar un correcto nivel lógico bajo, se debe cumplir también que:

$$R < \frac{V_{ILmáx}}{I_{ILmáx}} < \frac{5\ V}{300\ \mu A} < 16666{,}67\ \Omega$$

Considerando las dos inecuaciones:

$$48\Omega < \mathrm{R} < 16666{,}67\ \Omega$$

Finalmente, se ha optado por un valor intermedio, R = 7,5 kΩ.

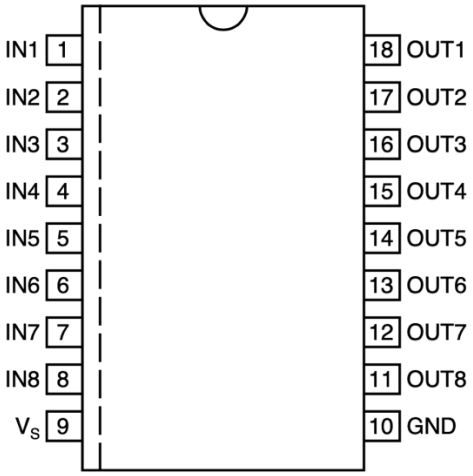


Figura 11: MIC2981. Fuente: MIC2981/2982 Datasheet

Así pues, alimentando la entrada Vs a +24 V, cuando se produzca un nivel alto de voltaje en la entrada, se tendrán +24 voltios en la salida del circuito integrado. En estas salidas se han añadido unas resistencias para limitar la corriente. El circuito impreso diseñado tiene la siguiente forma:

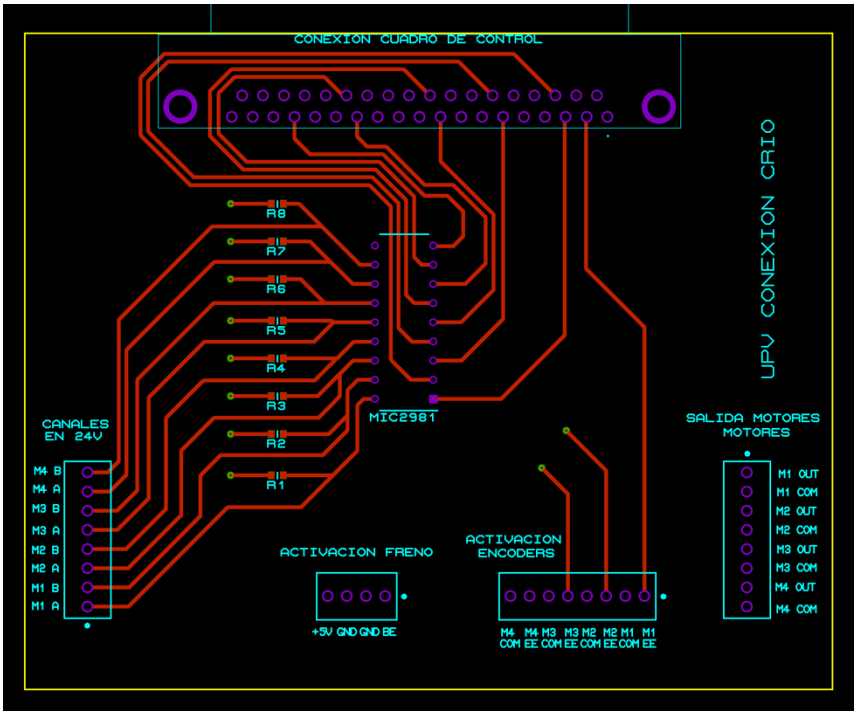


Figura 12: Circuito impreso conexión Compact RIO con el cuadro de control. Capas de arriba.

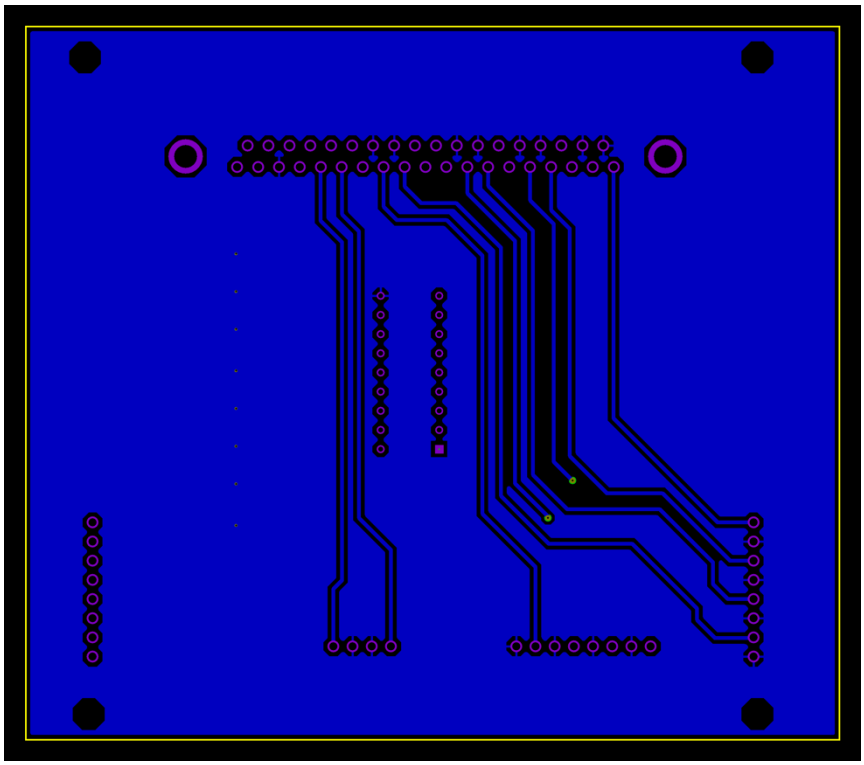


Figura 13: Circuito impreso conexión Compact RIO con el cuadro de control. Capas de abajo.

De esta manera, las conexiones a realizar entre esta placa y el controlador son las siguientes:

- Salida de motores:

NI 9263 Slot 3 CRIO	Placa conexión CRIO
AO0	M1 OUT
COM	M1 COM
AO1	M2 OUT
COM	M2 COM
AO2	M3 OUT
COM	M3 COM
AO3	M4 OUT
COM	M4 COM

Tabla 1: Conexiones entre NI 9263 y placa de conexión.

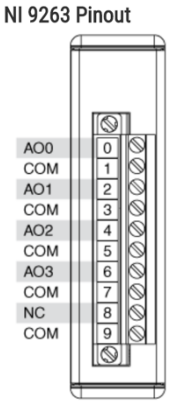


Figura 14: NI 9263 Pinout. Fuente: http://zone.ni.com/reference/en-XX/help/373197L-01/criodevicehelp/ni_9263/

- Activación de encoders y del freno:

NI 9474 Slot 1 CRIO	Placa conexión CRIO
DO0	M1 EE
DO1	M2 EE
DO2	M3 EE
DO3	M4 EE
DO4	BE
DO5	-
DO6	-
DO7	-
Vsup	+5V
COM	GND

Tabla 2: Conexiones entre NI 9474 y placa de conexión.

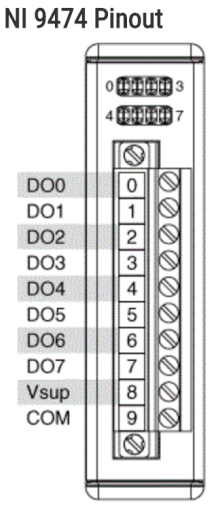


Figura 15: NI 9474 Pinout. Fuente: Conexiones entre NI 9263 y placa de conexión.

- Canales de los encoders:

NI 9421 Slot 2 CRIO	Placa conexión CRIO
DI0	M1 A
DI1	M1 B
DI2	M2 A
DI3	M2 B
DI4	M3 A
DI5	M3 B
DI6	M4 A
DI7	M4 B
COM	GND

Tabla 3: Conexiones entre NI 9421 y placa de conexión.

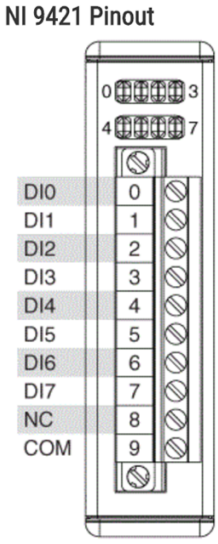


Figura 16: NI 9421 Pinout. Fuente: http://zone.ni.com/reference/en-XX/help/373197L-01/criodevicehelp/ni_9421/

5.2. SOLUCIÓN SOFTWARE

5.2.1. LENGUAJE DE PROGRAMACIÓN LABVIEW

La programación del control del robot se ha desarrollado con LabVIEW (Laboratory Virtual Instrument Engineering Workbench). Este es un lenguaje de programación gráfico desarrollado por National Instruments. Los sistemas Compact RIO, son productos pensados para ser programados mediante este lenguaje. LabVIEW está diseñado para aplicaciones de ingeniería como instrumentación y control de procesos.

Un programa de LabVIEW se denomina Virtual Instrument (VI) y consta de un panel frontal, que funciona como interfaz de usuario, y del código. Los elementos del código están programados internamente en C y BASIC. [19]

Así pues, mediante LabVIEW se puede desarrollar la programación del controlador Compact RIO y de su FPGA [20]. Este lenguaje incorpora un módulo específico para la programación de las FPGA y el entorno incorpora compiladores y programadores para estas.

El controlador Compact RIO del proyecto es compatible con la versión 2016 de LabVIEW y las anteriores. Se ha utilizado la 2016 de 32 bits y ha sido necesario instalar los siguientes módulos, todos en la versión también de 2016 y 32 bits:

- Módulo NI-RIO: Este corresponde a los controladores para manejar los dispositivos Compact RIO.
- Módulo LabVIEW Real-Time: Añade funcionalidades de tiempo real para aplicaciones de sistemas distribuidos de tiempo real. La temporización es mucho más precisa utilizando un sistema operativo de tiempo real.
- Módulo LabVIEW FPGA: Mediante este módulo se pueden implementar aplicaciones basadas en FPGA. Permite el acceso a las entradas y salidas del dispositivo con la lógica de LabVIEW. Además, se puede simular el comportamiento del programa de la FPGA.

5.2.2. COMUNICACIÓN TCP/IP

Ante la necesidad de recibir en un ordenador los datos de cada periodo de muestreo como el tiempo en el que se ha ejecutado, las posiciones de los actuadores o la acción de control para su monitorización y estudio de la respuesta, se debe de establecer una comunicación entre el controlador y el ordenador. Además, servirá también para enviar las referencias que debe seguir el robot desde el ordenador.

Se comunicarán, por tanto, los programas que se ejecuten en el ordenador y los que se ejecuten en la Compact RIO. El protocolo de comunicaciones elegido para llevar a cabo esta operación es el TCP/IP.

El protocolo TCP (Transmission Control Protocol) tiene como función asegurar una comunicación segura entre dos sistemas sin errores y sin pérdidas, a través de dividir la información en paquetes numerados de un tamaño adecuado con el fin de reconstruir el mensaje de destino en el orden correcto, y en caso de que falte un dato, se solicita a la máquina de origen los datos que faltan. Por otra parte, el protocolo IP se encarga de identificar los dispositivos conectados en la red. Cada sistema que se conecta a la red obtiene un identificador propio (dirección IP). [21]

El último identificador necesario es el puerto, en la comunicación TCP/IP, se utilizan los puertos para identificar distintos destinos dentro de una misma máquina. Además, se define Socket como una conexión entre dos máquinas con dirección IP.

Por último, la comunicación TCP/IP se basa en una arquitectura cliente-servidor. Múltiples clientes solicitan un intercambio de información a un único servidor, este intercambio puede ser tanto una operación de lectura como de escritura.

Este tipo de comunicación se ha implementado en los programas a través del mismo software LabVIEW que ofrece funciones para llevarla cabo y se ha seguido la estructura que se muestra a continuación:

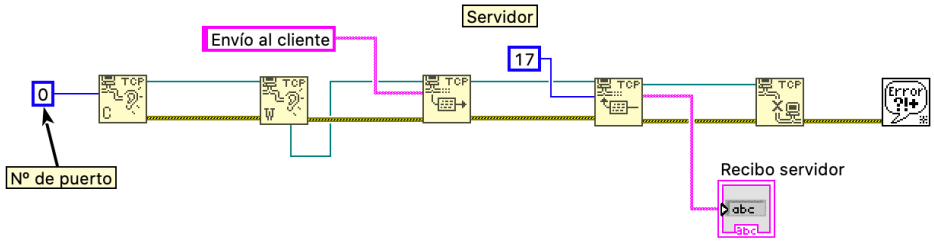


Figura 17: Estructura simple de un servidor TCP/IP en LabVIEW

Esta estructura corresponde al servidor, en este caso, será el ordenador. En un primer lugar, se deja al servidor escuchando por un puerto determinado y se espera hasta que un cliente se conecte en dicho puerto. Posteriormente, se realizan las operaciones de escritura y lectura necesaria y, finalmente, se cierra la conexión.

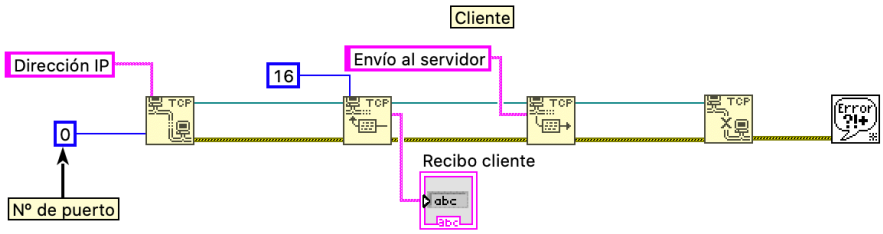


Figura 18: Estructura simple de un cliente TCP/IP en LabVIEW

Esta figura muestra la estructura del cliente, en un primer momento, se solicita y abre una conexión en una máquina con una dirección IP en un puerto determinado. A continuación, Se realizan las escrituras y lecturas que se requieran y se cierra la conexión. En este proyecto, el controlador actuará de cliente.

5.2.3. CINEMÁTICA INVERSA

A lo largo de este proyecto, va a ser necesario conocer la posición de las articulaciones teniendo los datos de la posición central de la plataforma del robot, ya sea para convertir los datos de referencias cartesianas en valores de articulaciones prismáticas o para obtener el valor de todas las articulaciones (variables activas) y poder establecer un control más complejo que se abordará en próximos apartados.

Así pues, se ha desarrollado un programa que realiza esta función. Este programa se puede utilizar tantas veces como sea necesario, se puede introducir dentro de otro como un subbloque. Se han implementado dos variantes distintas, en la primera, se calcula solamente las posiciones de las articulaciones prismáticas, mientras que, en la segunda, se calculan todas las juntas activas. Se ha realizado de esta manera para disminuir el tiempo de cálculo en el caso de que se requieran conocer solamente las posiciones de las articulaciones prismáticas.

A grandes rasgos, la estructura del sistema que calcula la cinemática inversa para las articulaciones prismáticas es la siguiente:

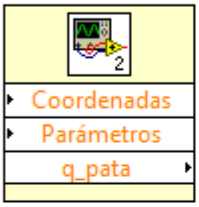


Figura 19: Bloque de cinemática inversa simple, CinInversa3UPS_RPU_Sequence.vi

La entrada “coordenadas” corresponde con un vector de longitud 4 con las coordenadas cartesianas de la posición del centro de la plataforma, estas son: X_m, Z_m y los ángulos Theta

y Psi. “Parámetros” corresponde con otro vector de longitud 11 cuyos valores son los parámetros que definen la configuración del robot. Es necesario conocer estos parámetros para poder aplicar la cinemática inversa. Las configuraciones posibles son:

Configuración	R1	R2	R3	ds	BetaFD	betaFI	Rm1	Rm2	Rm3	betaMD	betaMI
0	0,4	0,4	0,4	0	$\frac{50\pi}{180}$	$\frac{40\pi}{180}$	0,2	0,2	0,2	$\frac{30\pi}{180}$	$\frac{40\pi}{180}$
1	0,4	0,4	0,4	0,15	$\frac{90\pi}{180}$	$\frac{45\pi}{180}$	0,3	0,3	0,3	$\frac{50\pi}{180}$	$\frac{90\pi}{180}$
2	0,4	0,4	0,4	0,15	$\frac{90\pi}{180}$	$\frac{45\pi}{180}$	0,25	0,25	0,25	$\frac{5\pi}{180}$	$\frac{90\pi}{180}$
3	0,3	0,3	0,3	0	$\frac{5\pi}{180}$	$\frac{90\pi}{180}$	0,2	0,2	0,2	$\frac{70\pi}{180}$	$\frac{30\pi}{180}$
4	0,4	0,4	0,4	0,15	$\frac{90\pi}{180}$	$\frac{5\pi}{180}$	0,15	0,15	0,15	$\frac{45\pi}{180}$	$\frac{90\pi}{180}$
5	0,3	0,3	0,3	0,15	$\frac{50\pi}{180}$	$\frac{90\pi}{180}$	0,15	0,15	0,15	$\frac{90\pi}{180}$	$\frac{45\pi}{180}$
6	0,25	0,25	0,25	0,15	$\frac{5\pi}{180}$	$\frac{90\pi}{180}$	0,15	0,15	0,15	$\frac{80\pi}{180}$	$\frac{75\pi}{180}$
7	0,35	0,35	0,35	0,15	$\frac{90\pi}{180}$	$\frac{25\pi}{180}$	0,2	0,2	0,2	$\frac{60\pi}{180}$	$\frac{90\pi}{180}$
8	0,35	0,35	0,35	0,15	$\frac{35\pi}{180}$	$\frac{30\pi}{180}$	0,15	0,15	0,15	$\frac{5\pi}{180}$	$\frac{90\pi}{180}$

Tabla 4: Configuraciones de los parámetros del robot

La salida “q_pata” es un vector de tamaño 4 cuyos elementos son las soluciones de cada articulación prismática.

Las ecuaciones que definen la cinemática inversa y que ya se obtuvieron a partir de estudios técnicos previos a este proyecto han sido implementadas mediante el bloque de LabVIEW “Formula Node”, el cual permite introducir líneas de código en lenguaje C.

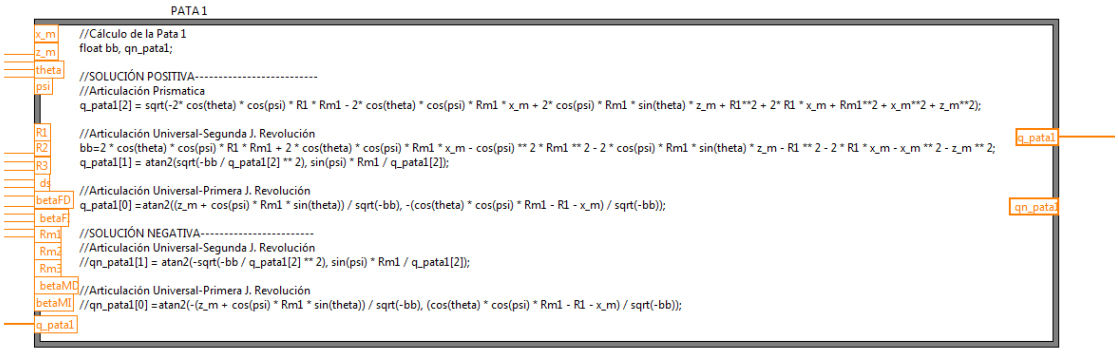


Figura 20: Cinemática inversa en bloque "Formula node".

Finalmente, la estructura general del bloque que calcula la cinemática inversa al completo, es decir, la posición de todas las articulaciones activas, es la siguiente:

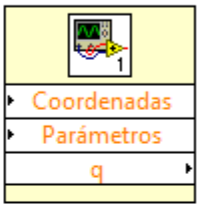


Figura 21: CinInversa3UPS_RPU_Completa.vi

Las entradas son las mismas que las del bloque anterior, mientras que la salida en este caso es una matriz de tamaño 4x3. Cada fila corresponde con la posición de las articulaciones de una pata. Por ejemplo, en la fila 3, el primer elemento es la solución de la posición de la articulación prismática, el segundo elemento la solución de la articulación universal-segunda J. revolución y el tercer elemento a la articulación universal-primera J. revolución. Solamente se diferencia la cuarta pata que tiene dos soluciones, la primera una articulación prismática y la segunda una articulación de revolución. Así pues, el elemento 4x3 de la matriz siempre es 0.

5.2.4. CINEMÁTICA DIRECTA

La cinemática directa es el procedimiento por el que se calculan las coordenadas del efector final, en este caso del centro de la plataforma a partir de las posiciones de articulaciones activas, concretamente, de las articulaciones prismáticas. Debido a la complejidad de las ecuaciones que rigen el comportamiento de la cinemática directa del robot, estas se deben resolver utilizando un método numérico. El método empleado ha sido el de Newton-Raphson.

Este es un método iterativo cuya finalidad es obtener una solución aproximada de una ecuación o un conjunto de ecuaciones. En cada iteración se obtiene un error en la solución, así pues, la función que calcula la cinemática directa itera hasta que el error es menor que una cierta tolerancia. En este caso se ha seleccionado 1e-7, pero este parámetro se puede modificar.

Si el error durante un cierto número de iteraciones no es menor que este parámetro también se terminarán las iteraciones. Este segundo número también se puede seleccionar. La estructura general del bloque que calcula la cinemática directa es la siguiente:

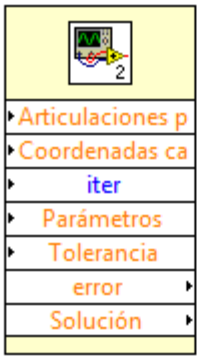


Figura 22: Bloque de cinemática directa, CinDirectaPos3UPS_RPU.vi

La primera entrada, “Articulaciones prismáticas”, es un vector de tamaño 4 cuyos elementos son las posiciones de estas articulaciones. La segunda entrada, “Coordenadas cartesianas anteriores”, es otro vector de tamaño 4 que agrupa la solución inmediatamente anterior de las coordenadas cartesianas X_m, Z_m, Theta y Psi. “Iter” es el número máximo de iteraciones mencionado anteriormente y “Tolerancia”, el error mínimo a partir del cual se considera que se ha obtenido una solución. Los “Parámetros” es el mismo vector de parámetros utilizado en la cinemática inversa. Por último, como salidas se tiene la solución, un vector que incluye las coordenadas cartesianas y el error obtenido a la hora de calcularla.

El diagrama de funcionamiento de la resolución es el siguiente:

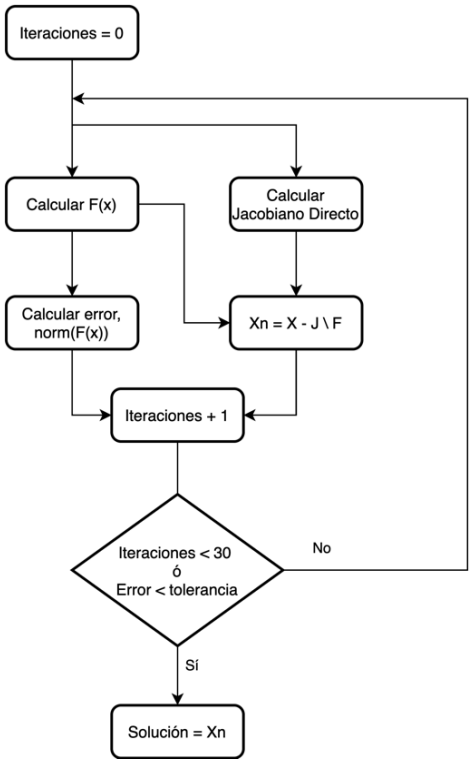


Figura 23: Diagrama de funcionamiento de la cinemática directa

Para calcular la matriz F(x) y la matriz Jacobiana Directa se han desarrollado las siguientes funciones:

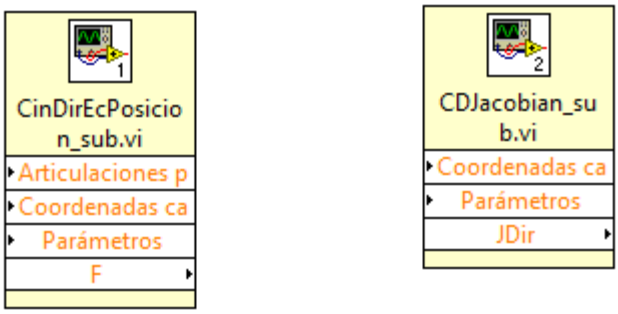


Figura 24: Funciones CinDirEcPosicion_sub.vi y CDJacobian_sub.vi

El primer bloque admite como entradas las posiciones de las articulaciones prismáticas, las coordenadas cartesianas anteriores y los parámetros. A través de esto calcula la matriz F, que está formada por las ecuaciones que definen la cinemática directa.

En segundo lugar, el bloque que calcula el Jacobiano directo lo hace a partir de las coordenadas cartesianas anteriores y de los parámetros.

5.2.5. LECTURA DE POSICIÓN Y ESCRITURA EN ACTUADORES MEDIANTE LA FPGA

La FPGA realiza de interfaz entre el bucle de control y las salidas y entradas del controlador. Así pues, se ha implementado un programa en la FPGA que lee los pulsos de las señales de los encoders, es decir, de las entradas digitales del módulo 2, 9421 (Digital Input) y convierte estas señales a valores de posición en metros que pueda utilizar el controlador. En segundo lugar, en la FPGA también se realiza la escritura de la acción de control en las salidas analógicas del módulo 3, 9263 (Analog Output). Finalmente, también se escribe el valor en las salidas digitales del módulo 9474 para los frenos del robot y la señal de activar los encoders.

Los encoders utilizados son incrementales y tienen dos canales para que sea posible distinguir el sentido de giro. Los cronogramas de las señales son los siguientes:

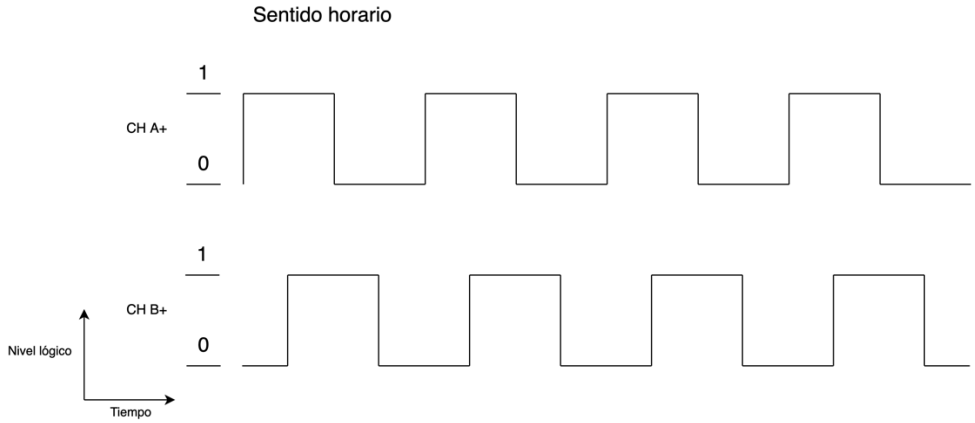


Figura 25: Cronograma de las señales de un encoder en sentido horario

Cuando el motor está girando en sentido horario, el canal A+ del encoder estará adelantado con respecto al canal B+ una fase de 90º.

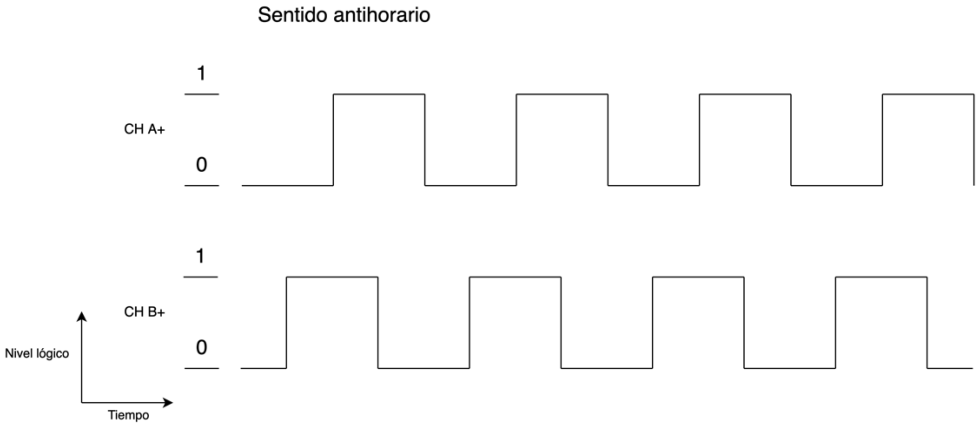


Figura 26: Cronograma de las señales de un encoder en sentido horario

Por el contrario, durante el sentido antihorario del motor, el canal B+ estará adelantado una fase de 90º con respecto al canal A+.

Teniendo en cuenta las formas de onda de las señales, se ha implementado una conversión en el programa de la FPGA que convierte estas señales en valores de posición de la siguiente forma:

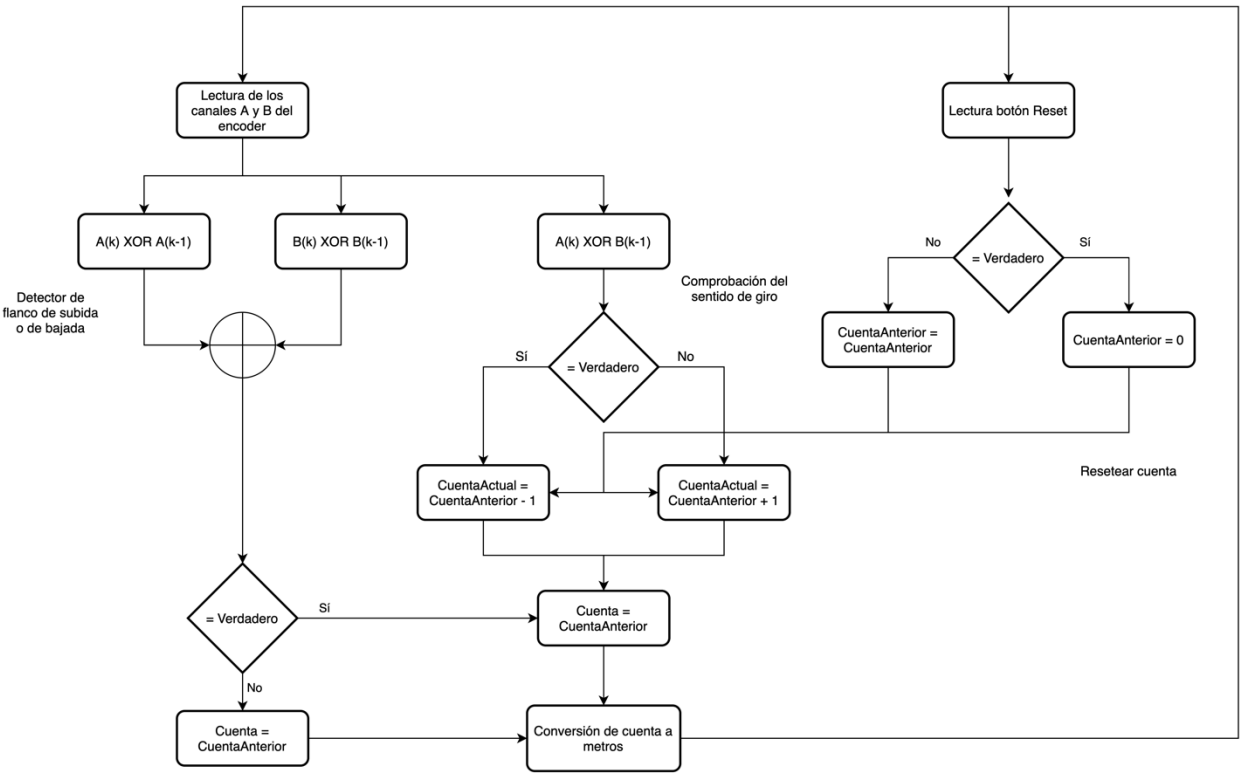


Figura 27: Diagrama de flujo de conversión de medida de encoder a posición en la FPGA

En primer lugar, realizando la operación XOR entre el valor actual y el valor anterior de la señal de uno de los canales de los encoders se puede detectar un flanco de subida o de bajada. Paralelamente, se procede a comprobar el sentido de giro, de nuevo utilizando la operación XOR esta vez entre la señal del canal A actual y B anterior. Se utiliza el valor anterior del canal B para que la operación lógica funcione tanto si se encuentra en sentido horario como antihorario. Si esta operación es igual a verdadero, se disminuye la cuenta y si es falsa, se aumenta la cuenta de pulsos. Además, también se incluye un botón de reset que pone a cero la cuenta del encoder.

Una vez se tiene el valor de la cuenta actualizado, si se ha detectado un flanco mediante la comprobación explicada en primer lugar, la cuenta de pulsos es igual al valor de la cuenta actualizado. En cambio, si no se ha detectado ninguno, se utiliza el valor de la cuenta anterior.

En cualquier caso, al final se convierte el valor de la cuenta a valores de posición en metros a través del siguiente factor de conversión: 5E-6 metros/cuentas.

En paralelo con este proceso, se lleva a cabo la escritura en las salidas de la acción de control. La FPGA accede a la variable acción de control del programa que se ejecuta en el procesador de la Compact RIO y la escribe en las salidas analógicas. También se lleva a cabo la escritura en la señal de los frenos del robot.

5.2.6. BÚFER PARA GUARDAR DATOS

Escribir datos en disco duro es una operación relativamente lenta, sobre todo teniendo en cuenta que la velocidad con la que se reciben los datos, la cual coincide con el periodo de muestreo del control, es del orden de milisegundos. Debido a esto, si las recepciones se escriben directamente en el disco duro, puede suceder que entre dos recepciones seguidas de muestras no se tenga el tiempo suficiente para escribirlas y se pierde el dato.

En cambio, las memorias volátiles RAM son más rápidas y sí que son compatibles con manejar esta información a esta velocidad. Por este motivo, se ha desarrollado un búfer de entrada de los datos procedentes del controlador en el ordenador. Los datos recibidos se almacenan en una variable para posteriormente, cuando el procesador pueda ejecutar la tarea, se escriban en el fichero de muestras. En el siguiente esquema se puede ver el funcionamiento:

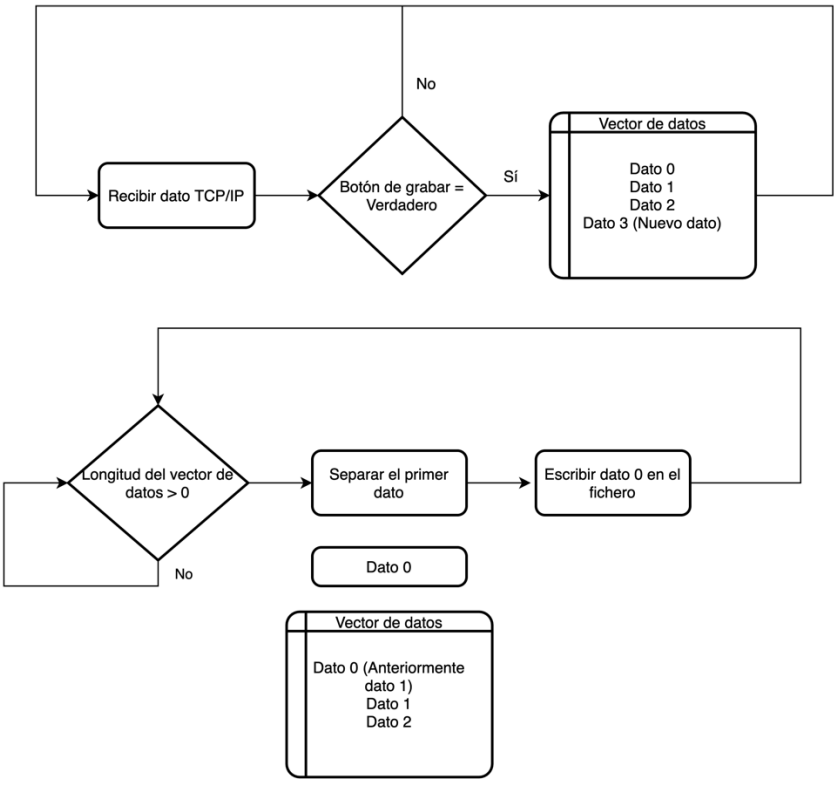


Figura 28: Diagrama de funcionamiento del búfer

Ambos procesos son bucles que funcionan en paralelo, sin embargo, el primer bucle, el encargado de recibir los datos y almacenarlos en una variable tiene una prioridad mayor que el

segundo, el que se encarga de escribir los datos en el disco duro, y también tiene una temporización que coincide con el periodo de muestreo. Al utilizar este método, no se pierde ninguna muestra.

5.2.7. ARQUITECTURA DEL BUCLE DE CONTROL

Para el desarrollo de la arquitectura del bucle de control se han tenido en cuenta las siguientes condiciones: Las referencias son ficheros de texto con valores numéricos ordenados en 4 columnas, una para cada actuador en caso de que los valores de referencias se traten directamente de la posición de las articulaciones. También es necesario leer valores de referencias que corresponden a las cuatro coordenadas cartesianas que definen la posición del efector final del robot.

Se puede observar en el siguiente esquema la arquitectura general del bucle de control:

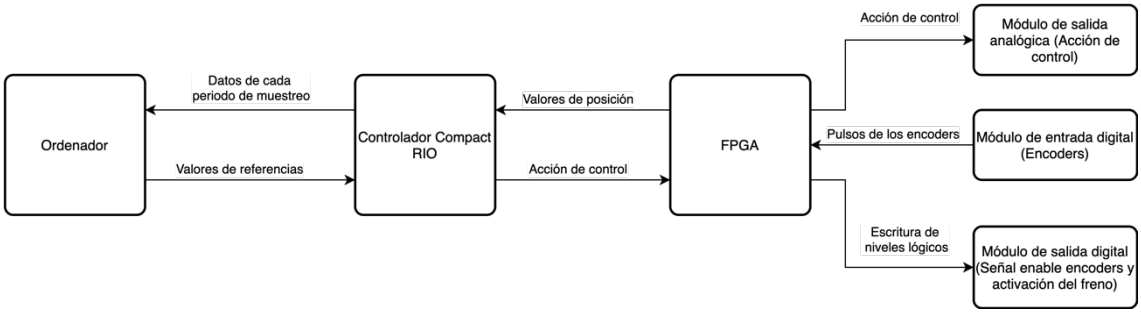


Figura 29: Arquitectura general del bucle de control

De una manera general, la función del ordenador sería enviar las referencias al controlador, además de recibir los datos de cada período de muestreo. El controlador recibe las referencias y el valor de posición de las articulaciones, para entonces, calcular la acción de control. Finalmente, la FPGA obtiene el valor del voltaje de los encoders, se convierte a un valor de posición y obtiene a partir del controlador, el valor de acción de control a aplicar en los motores.

Durante el diseño de la arquitectura, han surgido variaciones de carácter más específicas que se van a explicar en los siguientes apartados, así como una explicación más detallada de la arquitectura final.

5.2.7.1. ENVÍO DE FICHERO DE REFERENCIAS DE ARTICULACIONES

En un primer momento, se desarrolló una arquitectura en la que el ordenador lee por completo el fichero de referencias de posiciones de los actuadores directamente y se envía mediante TCP/IP a la Compact RIO.

Una vez realizado el envío, los datos de referencias se guardan en una variable y se comienza con el bucle de control. En cada iteración del bucle, primero se obtiene el valor de referencia a partir del número de iteraciones y se procede a la lectura de la posición de la FPGA. Posteriormente, se calcula la señal de error a partir de la diferencia entre ambos y se calcula la acción de control (en un primer momento con un regulador proporcional). Finalmente, se escribe el valor de la acción de control en la variable correspondiente en la FPGA que la escribirá en las salidas analógicas para los actuadores.

Este proceso se puede observar en el siguiente esquema:

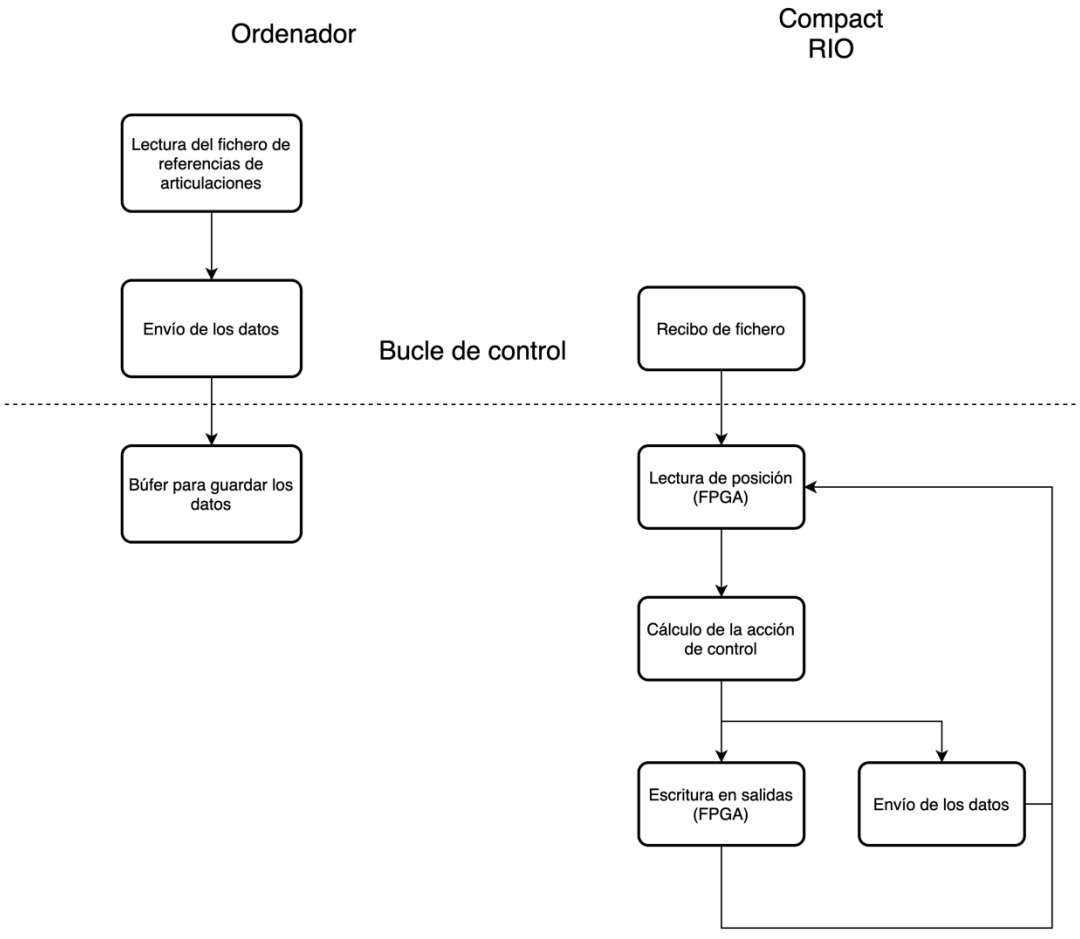


Figura 30: Envío de referencias de articulaciones

La temporización del bucle de control se ha realizado mediante la estructura de LabVIEW "Timed Loop", que permite ejecutar iteraciones de un código en un tiempo determinado, en este caso, el periodo de muestreo. De esta forma, con el procesador de tiempo real de la Compact RIO, se asegura la ejecución determinista de cada ciclo del bucle de control.

En cuanto al diagrama anterior, cabe destacar que el valor obtenido por la variable posición de la FPGA es un valor incremental, es decir, con la lectura de los encoders no se obtiene una posición absoluta, si no que, en un primer momento, cuando se ejecuta el programa, el valor de esta es de 0 independientemente del valor de la posición real de los actuadores al principio. Una vez los actuadores se muevan, se obtendrá un valor de posición con respecto al inicial que es 0.

Como las referencias se encuentran en valores absolutos, para evitar problemas en el control, a cada valor obtenido por la variable posición de la FPGA se le sumará el primer valor del fichero de referencias. Lo que es lo mismo, se considera que la posición inicial del robot corresponde con la primera posición del fichero de referencias.

5.2.7.2. ENVÍO DE FICHERO DE REFERENCIAS CARTESIANAS

La intención del proyecto es que los datos de los ficheros de referencias se encuentren en coordenadas cartesianas, precisamente en las cuatro coordenadas que definen la posición del efector final del robot. Sin embargo, como para calcular las acciones de control es necesario conocer las posiciones de las articulaciones prismáticas para cada coordenada cartesiana determinada, hay que realizar la cinemática inversa.

Como ya se ha mencionado anteriormente, con la cinemática inversa, dadas las coordenadas cartesianas del efector final, se pueden obtener la posición de las articulaciones. Así pues, primero se envía el fichero de referencias al controlador y antes de comenzar con el bucle de control, se realiza la cinemática inversa para cada referencia del fichero.

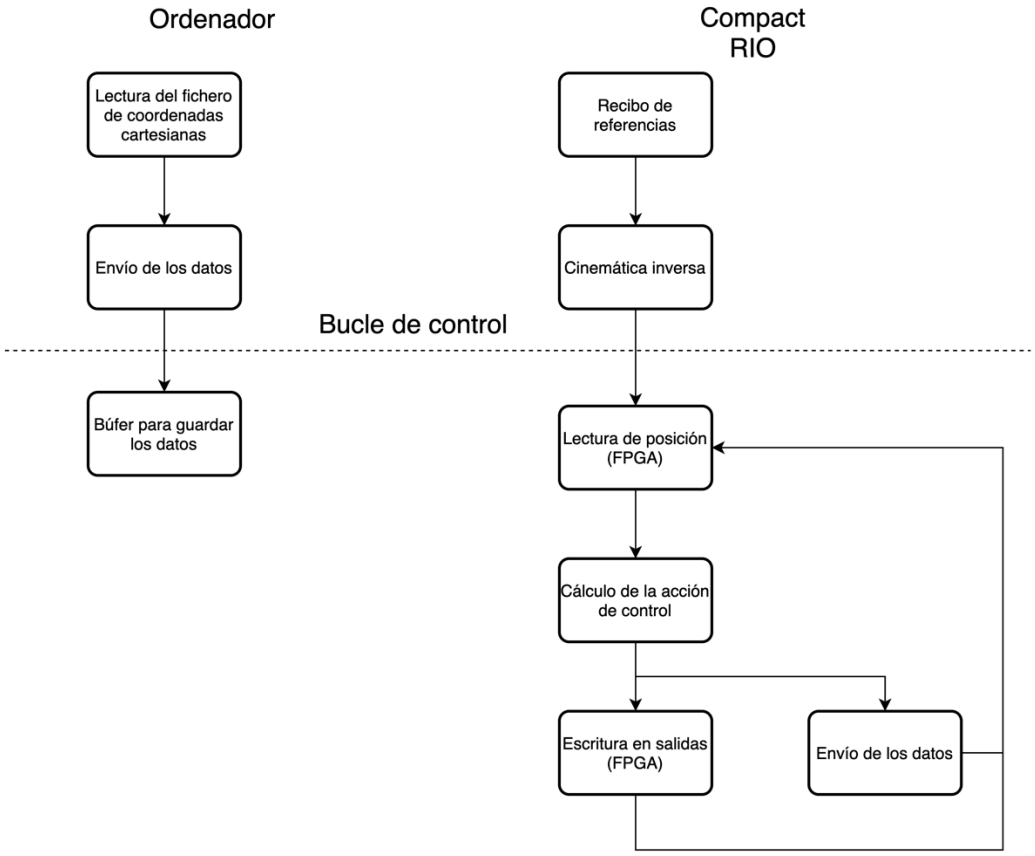


Figura 31: Envío de referencias cartesianas

Sin embargo, de esta forma aparecía un problema a la hora de realizar las pruebas en el laboratorio. Para realizar la cinemática inversa, es necesario convertir los datos del fichero que se encontraban en el tipo "string" a un vector de valores numéricos decimales. El fichero que se utilizó tenía un tamaño de 7001x4, en este caso, la trayectoria tendría 7001 iteraciones y las 4 coordenadas cartesianas. Así pues, a la hora de convertir este vector a posiciones de las articulaciones, este proceso requería mucho tiempo, concretamente entre 5 y 10 minutos. Lo cual es un motivo para prescindir de esta arquitectura, ya que los ficheros de referencias no tienen por qué ser de este tamaño, si no que pueden ser mayores, demorando más tiempo el proceso.

Este problema es probable que se deba a la falta de memoria volátil del procesador de la Compact RIO para tratar datos de gran tamaño.

5.2.7.3. ENVÍOS ÚNICOS DE REFERENCIAS CARTESIANAS

Con el fin de solucionar el problema anterior, en este caso, el ordenador envía una única referencia en cada iteración del bucle de control. De esta forma en el controlador ya no hay ninguna variable almacenada de un tamaño excesivo, ya que, en cada ciclo, las variables se actualizan. Además, el funcionamiento ahora es independiente del número de iteraciones que tenga la trayectoria de referencia, se pueden manejar trayectorias de un mayor tamaño.

No obstante, como desventaja, el bucle de control ahora depende del ordenador también, que en cada ciclo de control debe enviar la referencia correspondiente. El diagrama de funcionamiento es el siguiente:

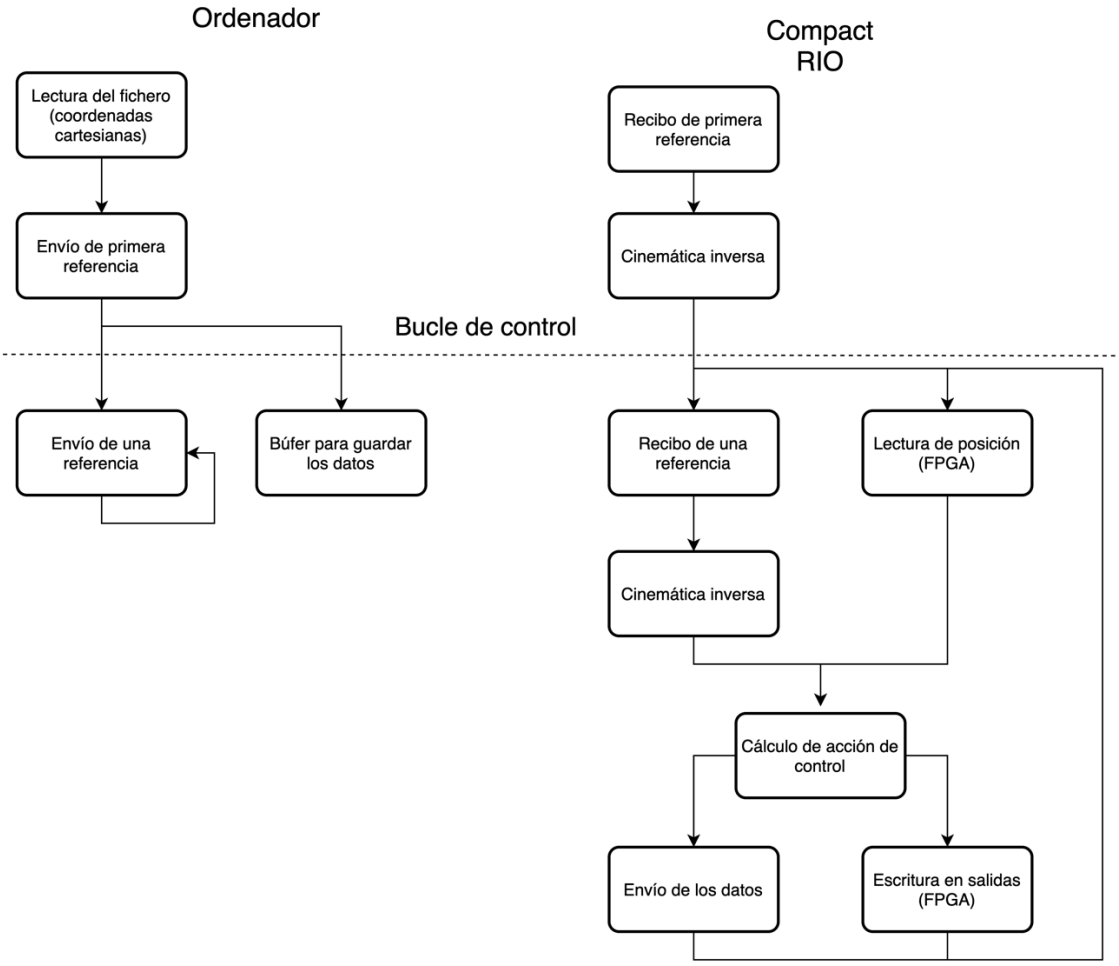


Figura 32: Único envío de referencias cartesianas

Tras realizar pruebas con tiempos de muestreo pequeños, entre 1 y 4 milisegundos, hay cada cierto tiempo iteraciones del ciclo de control que no se llegan a completar.

5.2.7.4. ARQUITECTURA FINAL: ENVÍO DE MÚLTIPLES LÍNEAS

Para disminuir el tiempo de ejecución de cada ciclo y solucionar el problema planteado anteriormente, se puede eliminar el tiempo de recepción de los datos TCP/IP cada un número determinado de ciclos. Es decir, se pueden enviar múltiples líneas de referencias en un solo envío y en un solo ciclo de control para posteriormente utilizar estos datos hasta que se apliquen todas las referencias y sea necesario recibir de nuevo otro conjunto de referencias.

Con este principio surge esta arquitectura, el programa del ordenador calcula qué líneas de referencias debe mandar en cada envío a partir del fichero de referencias cartesianas.

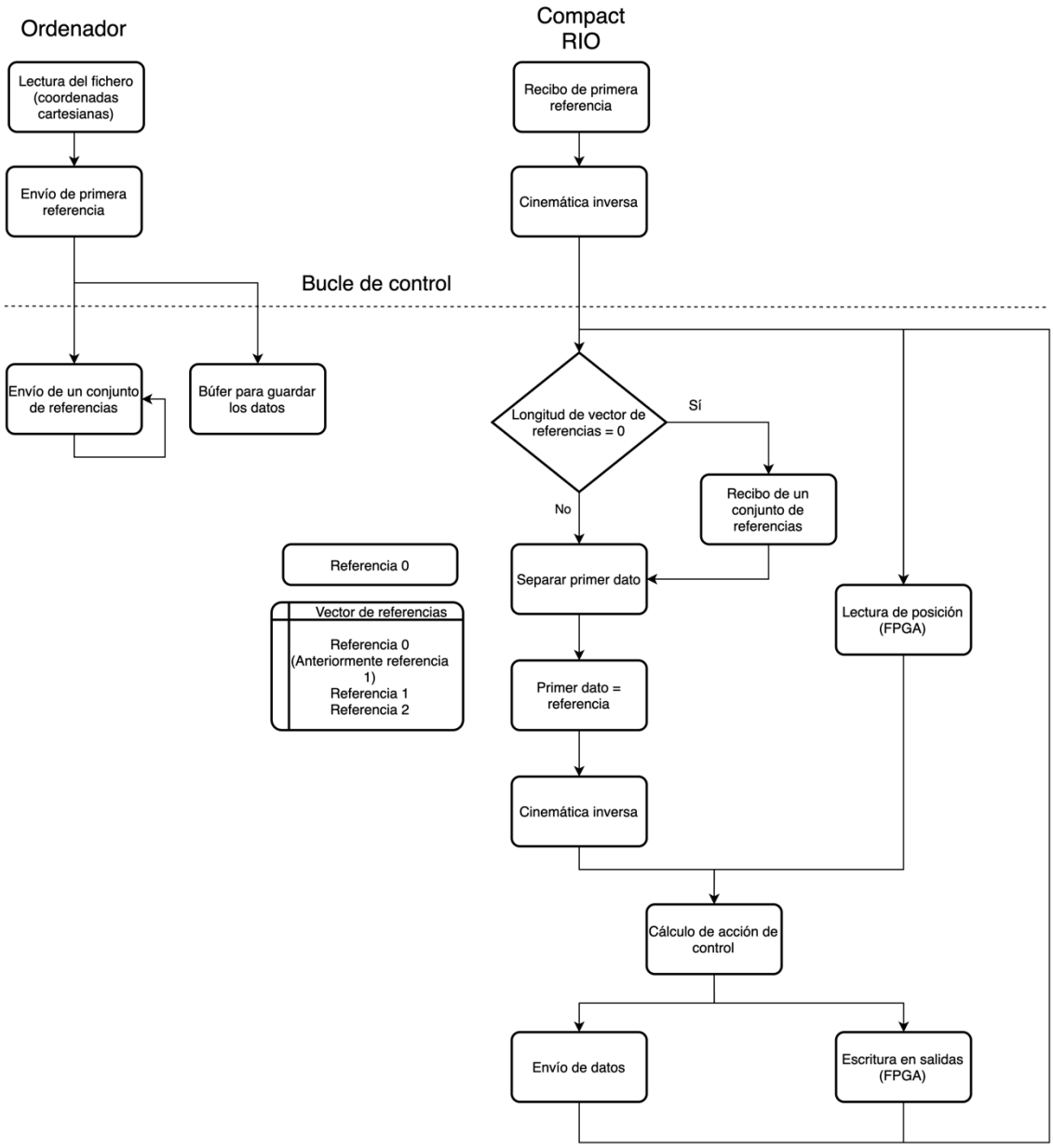


Figura 33: Envío de múltiples líneas de referencias

El número de referencias en un envío es variable y se puede seleccionar. Así pues, en el programa ejecutado en el ordenador, se calcula a partir de esta variable la longitud de los datos que supondría el envío de cada conjunto de referencias. Esta longitud se envía antes de comenzar el bucle de control al controlador para que este dato se utilice para recibir correctamente las referencias.

Además, en cada iteración del bucle de envío de referencias, se calcula a qué posición de los datos de todas las referencias se debe acceder con el fin de enviar las líneas que correspondan a dicha iteración.

5.2.8. ALGORITMOS DE CONTROL

Los algoritmos implementados para controlar la posición de la plataforma del robot paralelo son los que se explican a continuación.

5.2.8.1. CONTROLADOR PID

En primer lugar, se ha implementado un algoritmo PID cuya acción de control responde a la siguiente ecuación:

$$u(k) = Kp \cdot e(k) + Ki \cdot i(k) - Kv \cdot \hat{v}(k)$$

Como cualquier controlador PID, una fracción de la acción de control es directamente proporcional al error entre la referencia y la posición de los actuadores, otra fracción corresponde con una acumulación del error (aportación integral). El último término consiste con una realimentación de la velocidad de los actuadores. A falta de un sensor de velocidad, ha sido necesario estimarla, en este caso se ha realizado una derivada discreta:

$$\hat{v} = \frac{\Delta \mathrm{q}}{\Delta t} = \frac{q(k) - q(k-1)}{Ts}$$

El esquema dinámico es el siguiente:

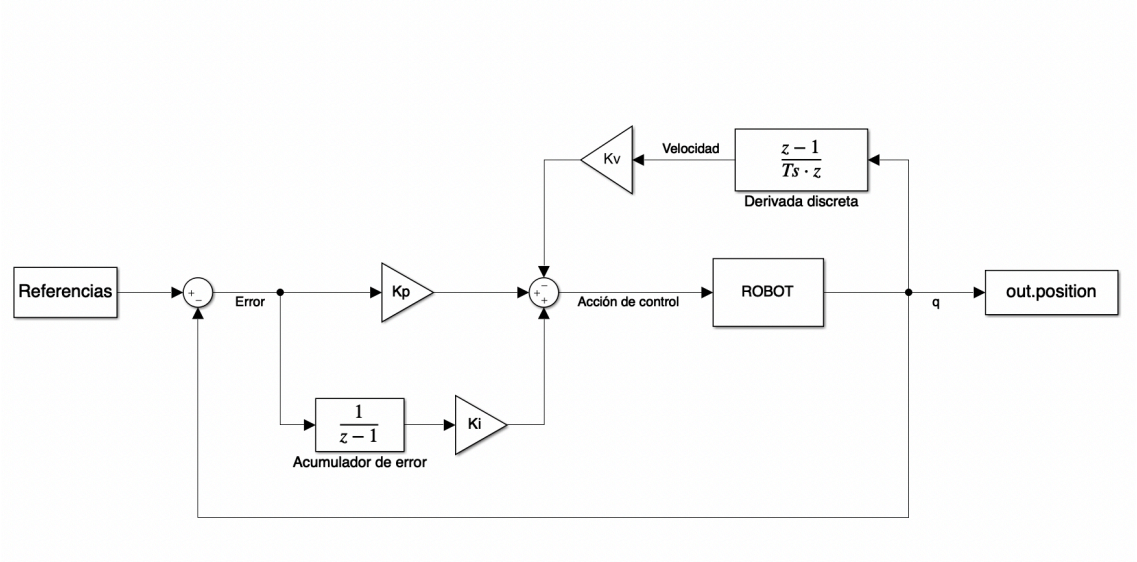


Figura 34: Diagrama del algoritmo de control PID

En el apartado de simulación y resultados se comentarán las ventajas y desventajas de este tipo de controlador aplicado al control de posición del robot paralelo del proyecto.

5.2.8.2. CONTROLADOR PD CON COMPENSACIÓN DE LA GRAVEDAD

El segundo regulador que se ha implementado es un controlador PD con compensación de la gravedad. Con respecto al anterior, se ha sustituido el término de acción integral por un término gravitacional que depende de la posición de los actuadores. Así pues, dependiendo de esta, se aporta energía con el fin de compensar el efecto que la gravedad tiene sobre el robot.

La acción de control corresponde con la siguiente ecuación:

$$u(k) = Kp \cdot e(k) - Kv \cdot \hat{v} + G(q_{(k)})$$

Para calcular $G(q_{(k)})$ ha sido necesario realizar las operaciones que se indican en el siguiente diagrama:

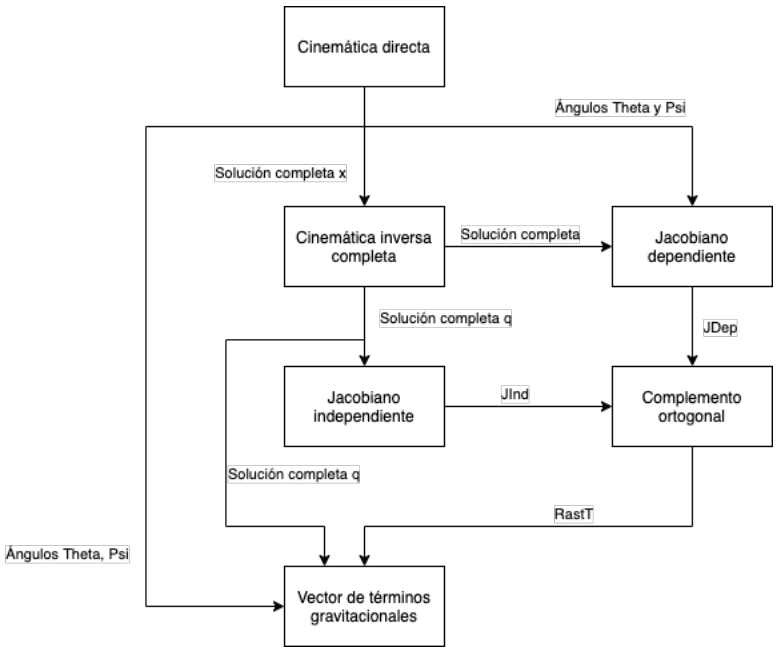


Figura 35: Diagrama del cálculo del vector de términos gravitacionales

Cada bloque de este diagrama corresponde con un subbloque implementado en LabVIEW, en apartados anteriores ya se han expliacado las funciones de la cinemática directa y de la cinemática inversa completa, ahora se van a explicar los restantes:

- Jacobiano dependiente:

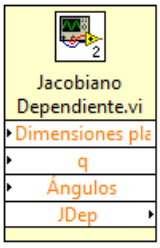


Figura 36: Bloque que calcula el Jacobiano dependiente, Jacobiano Dependiente.vi

Como entradas se tienen las dimensiones de la plataforma móvil, que corresponde con los elementos Rm1, Rm2, Rm3, betaMD y betaMI del vector de parámetros. La siguiente entrada es la solución completa de las juntas activas, la matriz solución del bloque de la cinemática inversa completa. La última entrada son los elementos Theta y Psi del vector de coordenadas cartesianas.

Este bloque calcula la matriz Jacobiana Dependiente función de las entradas mencionadas.

- Jacobiano independiente:

Figura 37: Bloque que calcula el Jacobiano Independiente, Jacobiano Independiente.vi

Este bloque calcula la matriz Jacobiana Independiente que es función de la entrada q, la matriz de juntas activas completa.

- Complemento ortogonal:

Figura 38: Subbloque que realiza el complemento ortogonal, Complemento Ortogonal.vi

En este bloque calcula la matriz R^* necesaria para calcular el vector de términos gravitacionales. Las ecuaciones implementadas en este bloque son las siguientes:

$$R^*_{[1:11,:]} = -JDep \backslash JInd$$

$$R^*_{[12:15,:]} = I_{4x4}$$

$$RastT = R^{*T}$$

- Vector términos gravitacionales

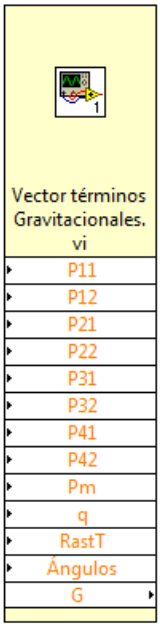


Figura 39: Función que calcula el vector de términos gravitacionales, Vector términos Gravitacionales.vi

Por último, este bloque calcula el vector de términos gravitacionales, es decir, la acción de control necesaria para compensar la gravedad. El cálculo se lleva a cabo a partir de las matrices q y RastT calculadas anteriormente además de los ángulos Theta y Psi, y de una serie de parámetros que corresponden con ciertas propiedades de las patas y de la plataforma móvil del robot como las masas y los centros de gravedad.

A partir de estos datos se calcula el Vector gravitacional (VQGrav) y a partir de este y la matriz RastT se obtiene el Vector gravitacional compacto (G), la salida de este subbloque.

$$G = RastT * VQGrav$$

De esta manera, ya se tiene el término de la acción de control necesaria para compensar la gravedad en el robot. El esquema del controlador final es el siguiente:

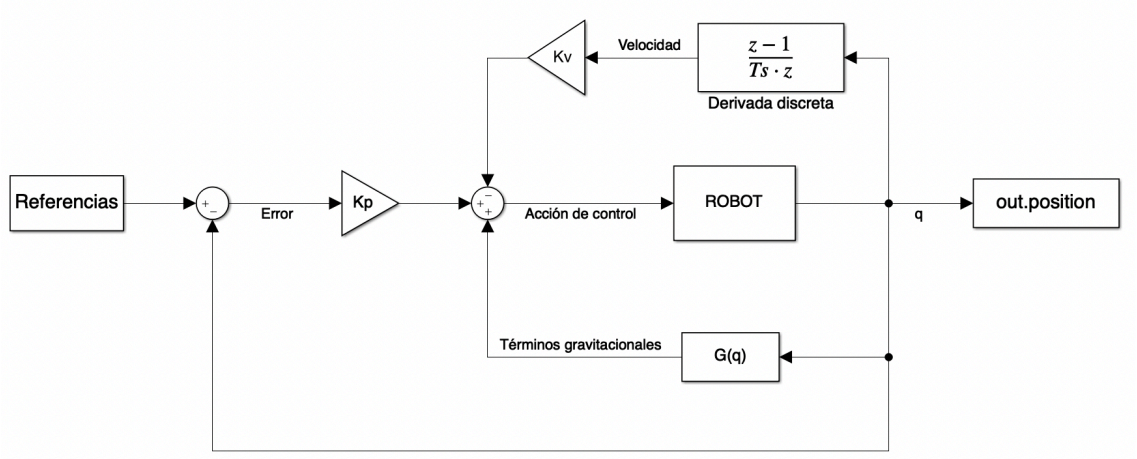


Figura 40: Diagrama del algoritmo de control de PD + Compensación de la gravedad

5.2.9. INTERFAZ DE USUARIO

Se han elaborado las interfaces de usuario a partir de los propios paneles frontales que ofrecen los programas de LabVIEW. En estos, se pueden añadir visualizaciones de datos numéricos y de gráficas, y también se pueden emplear entradas como botones, palancas y valores numéricos.

Cada programa tiene su propio panel de usuario, por ello, a continuación, se van a mostrar los paneles frontales de los programas realizados para el Servidor (ordenador) y el controlador (Compact RIO).

5.2.9.1. INTERFAZ PARA UN MOTOR

En la figura siguiente se muestra el panel frontal del programa del servidor para las arquitecturas de control de envío del fichero de referencias cartesianas y de articulaciones:

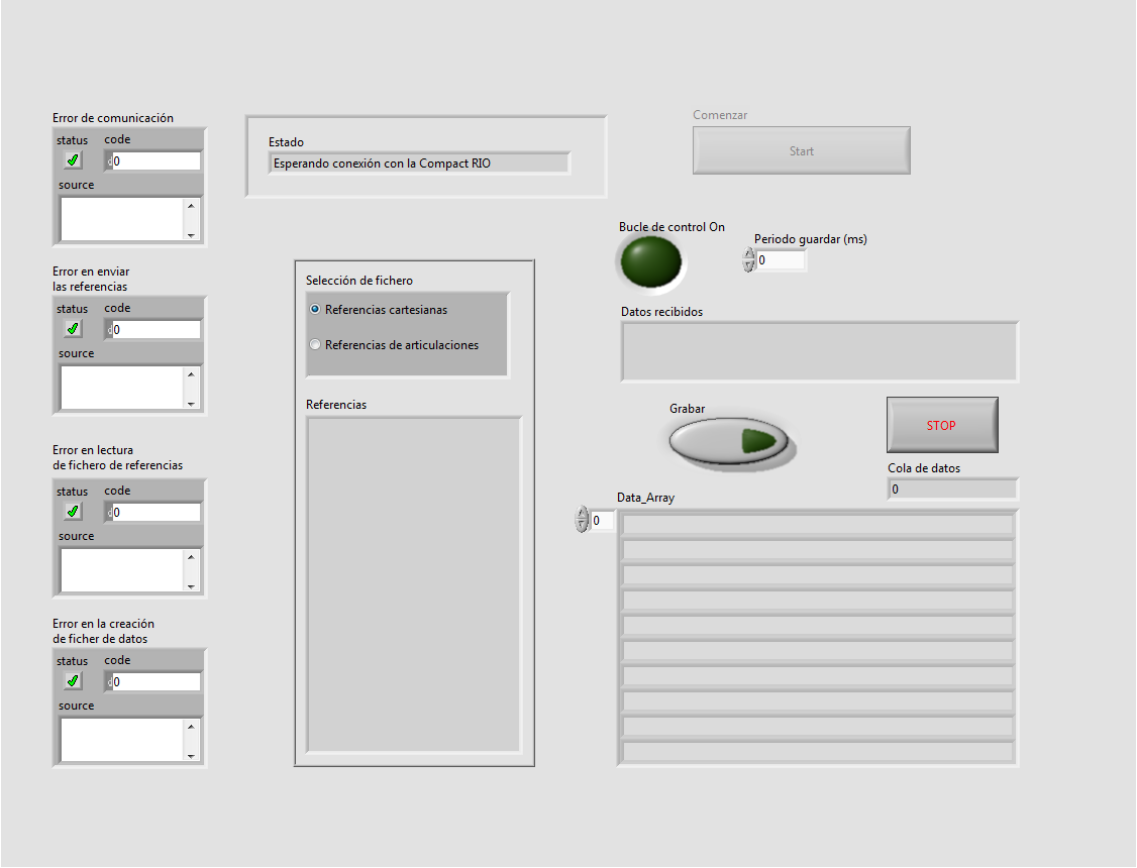


Figura 41: Panel frontal servidor para el envío del fichero

Gracias a las entradas de esta interfaz se puede elegir entre utilizar el fichero de referencias cartesianas y el fichero de referencias de articulaciones. Además, cuando se establezca la comunicación con el controlador, el botón de comenzar estará disponible y se podrá empezar con el experimento.

A continuación, se muestra el panel frontal del programa de la Compact RIO para las dos arquitecturas de control anteriores:

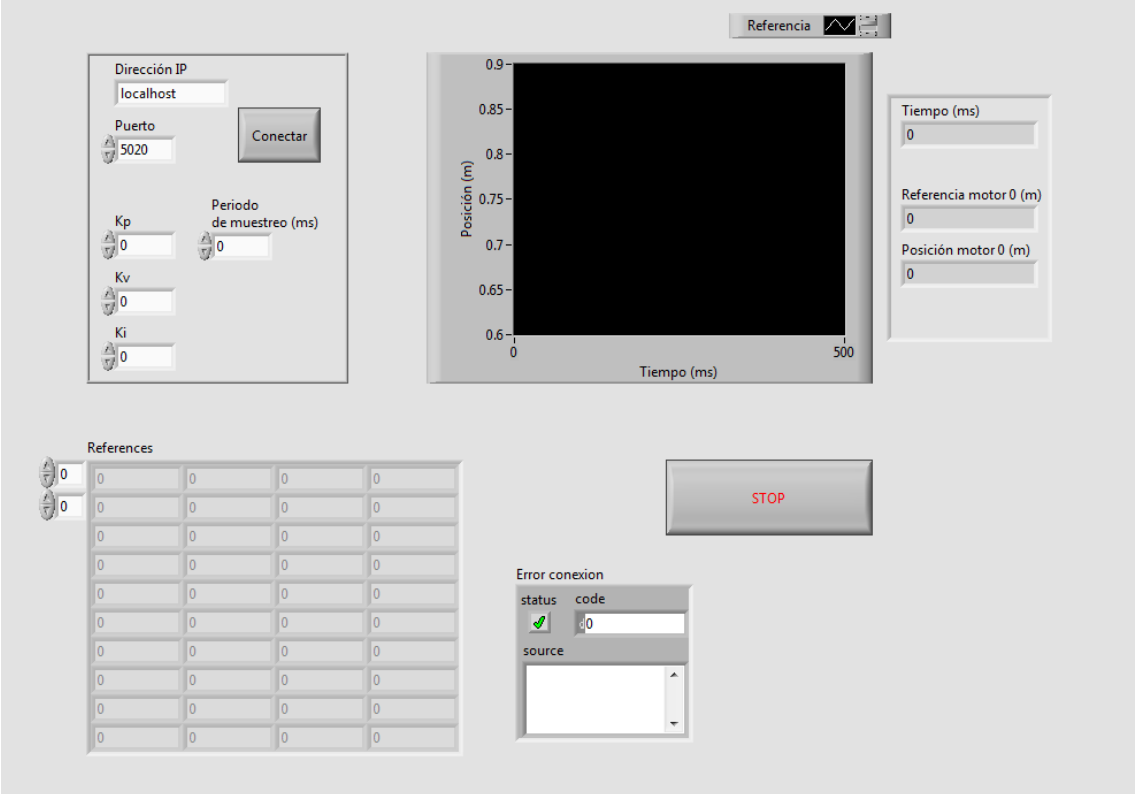


Figura 42: Panel frontal del programa de la Compact RIO para el envío del fichero

Como se observa, se pueden modificar los parámetros del controlador y la dirección IP y puerto del servidor. Además, hay una gráfica para visualizar la referencia (en blanco) y la posición de la articulación (en rojo). También hay un botón de paro en caso de emergencia.

Los programas relacionados con la arquitectura de control de un solo envío tienen los siguientes paneles de control, ambos son muy similares a las dos figuras anteriores.

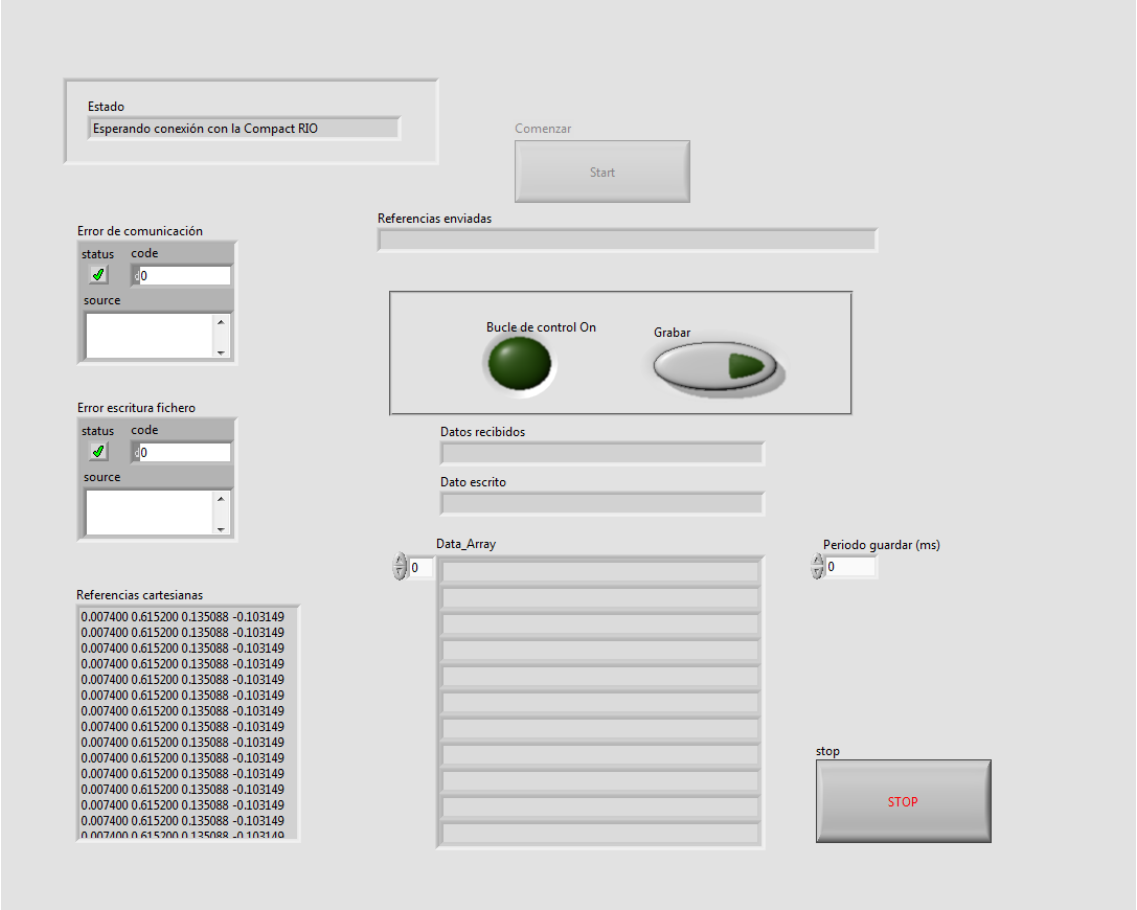


Figura 43: Panel frontal del servidor para un solo envío

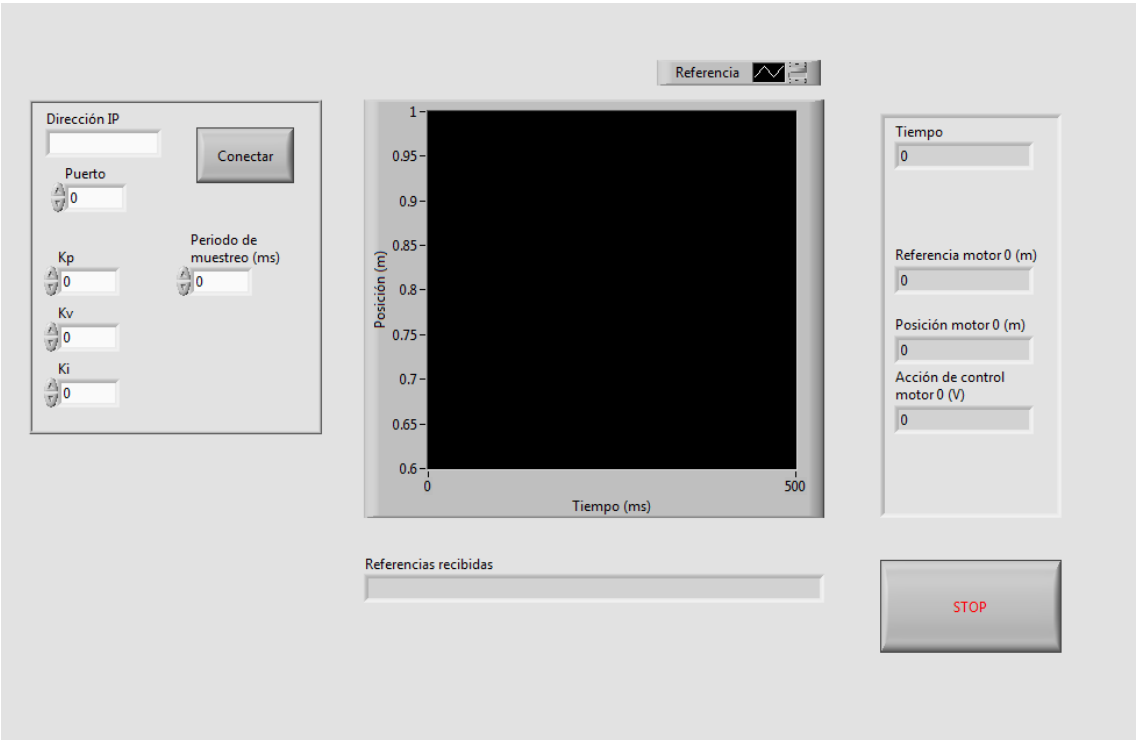


Figura 44: Panel frontal del controlador para un solo envío

5.2.9.2. INTERFAZ PARA EL ROBOT COMPLETO

El panel frontal del programa del servidor para el robot completo es el siguiente:

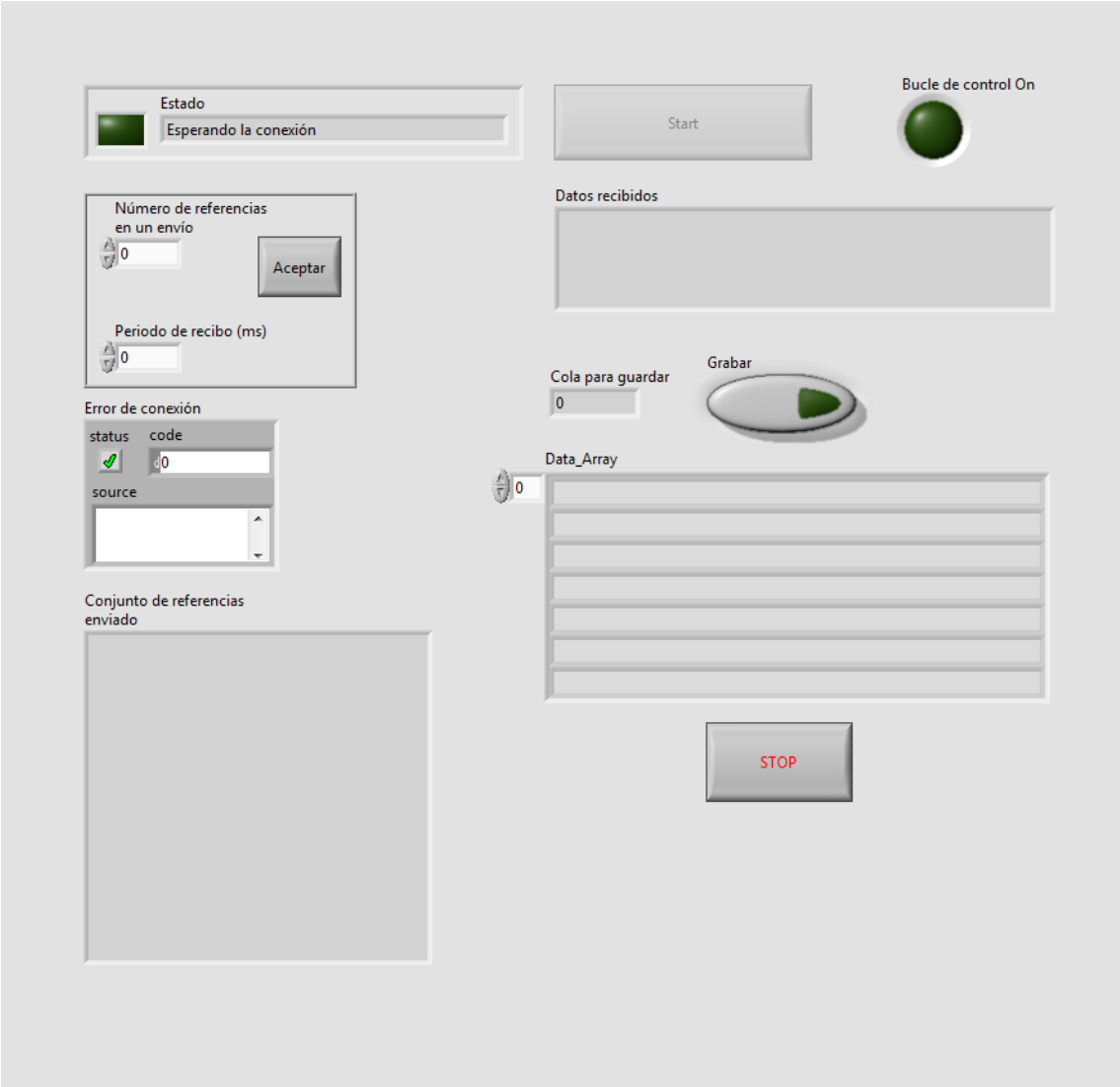


Figura 45: Panel frontal para el programa del servidor del robot completo

El panel es muy similar a las interfaces de los servidores anteriores. Sin embargo, en este se precisa seleccionar el número de referencias que se realiza en un envío ya que este utiliza la arquitectura final, la del envío de múltiples líneas de referencias.

En cuanto al panel frontal del programa ejecutado en el controlador, el resultado es el que se muestra en las siguientes figuras:

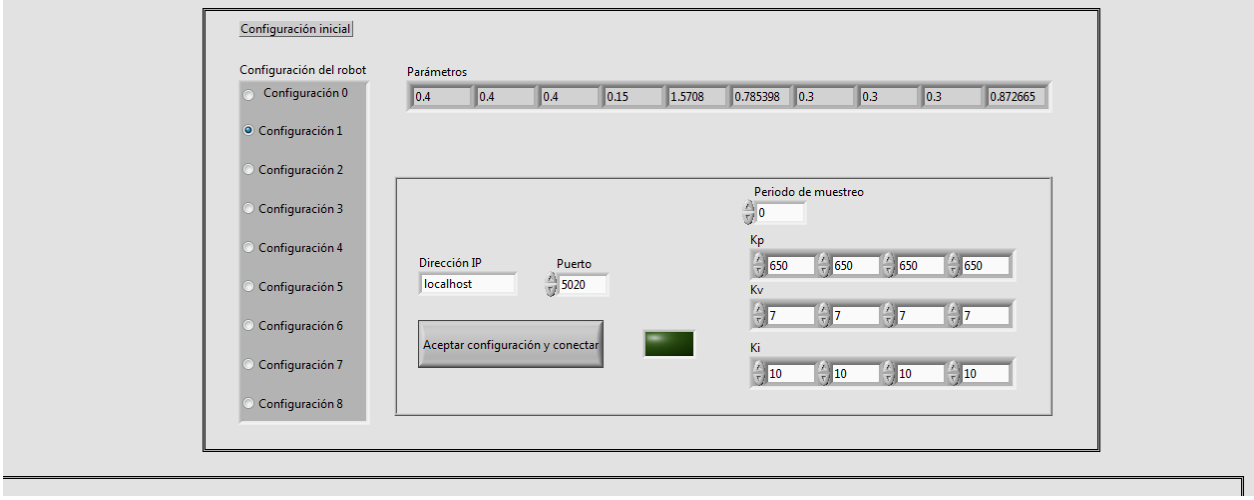


Figura 46: Panel frontal del controlador, configuración inicial

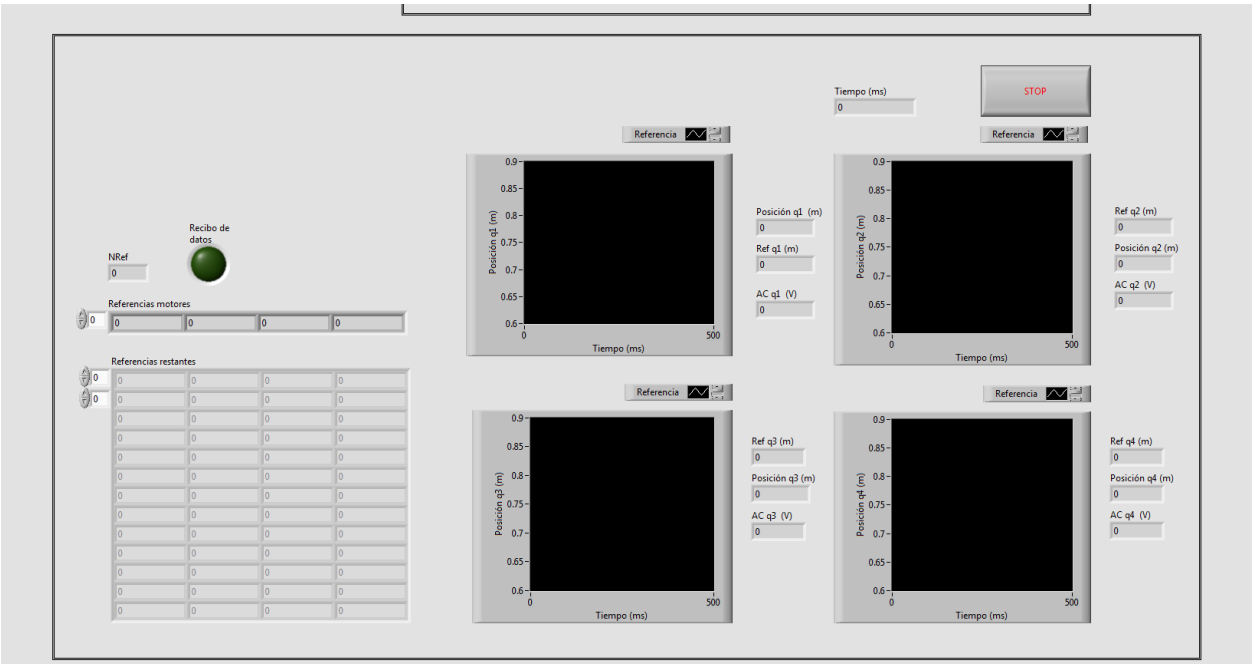


Figura 47: Panel frontal del controlador, bucle de control

Como se muestra en la primera figura, en este caso, además de seleccionar la dirección IP y el puerto del servidor, también es necesario seleccionar la configuración del robot, es decir, la serie de parámetros que definen la cinemática inversa, directa y el resto de las ecuaciones del robot. Estos parámetros se han mostrado en la tabla 4.

Posteriormente, en la segunda figura, se tienen las visualizaciones de cada articulación del robot en las gráficas. Además, hay un botón de paro de emergencia que activa los frenos del robot.

6. SIMULACIÓN Y RESULTADOS

Ante la imposibilidad de acudir a la Universitat Politècnica de València, la validación de los controladores se ha realizado mediante una simulación. Esta se ha desarrollado utilizando la herramienta Simulink de MATLAB haciendo uso de un modelo de la planta del robot que ya se había desarrollado previamente a este proyecto.

También se utilizará un modelo de un único motor y un eje para simular las primeras pruebas que se hicieron en el laboratorio al comienzo del trabajo.

6.1. ESQUEMA DE LA SIMULACIÓN

Para hacer posible la simulación ha sido necesario realizar una serie de cambios en el programa que se ejecuta en la Compact RIO, este ahora se ejecuta en el mismo ordenador que el programa del servidor y se ha sustituido la lectura y escritura de los módulos por una lectura y una lectura TCP/IP de los datos necesarios de Simulink.

El nuevo esquema es el siguiente:

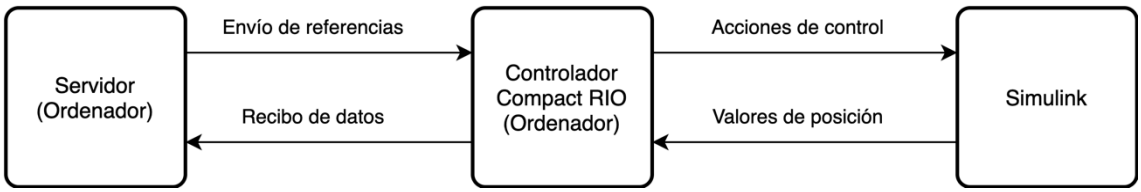


Figura 48: Arquitectura de control de la simulación

El ordenador sigue actuando como servidor del programa del controlador. Pero, por otro lado, el controlador funciona como servidor del modelo de Simulink, que actúa de cliente. El puerto utilizado para la comunicación entre el Servidor y el Controlador es el 5020 y para la comunicación de la simulación el 5021.

El diagrama de Simulink empleado para el control de un motor es el siguiente:

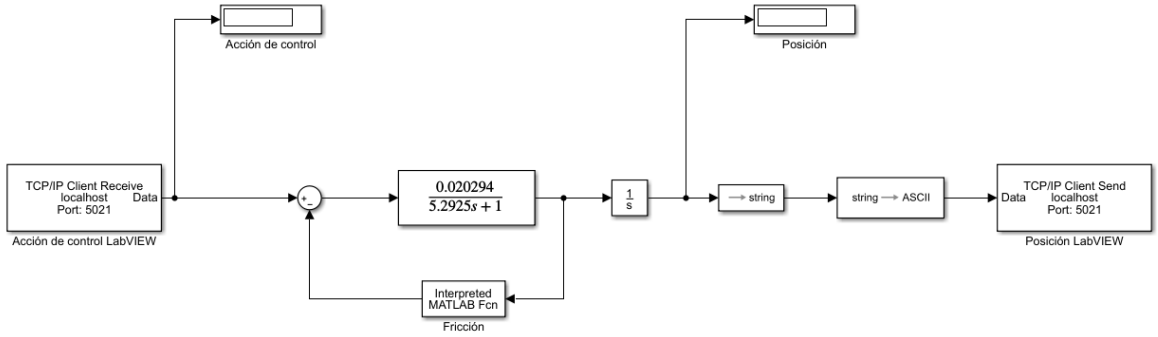


Figura 49: Diagrama de la simulación de un motor

En segundo lugar, el diagrama de Simulink utilizado para simular el comportamiento del robot es el siguiente:

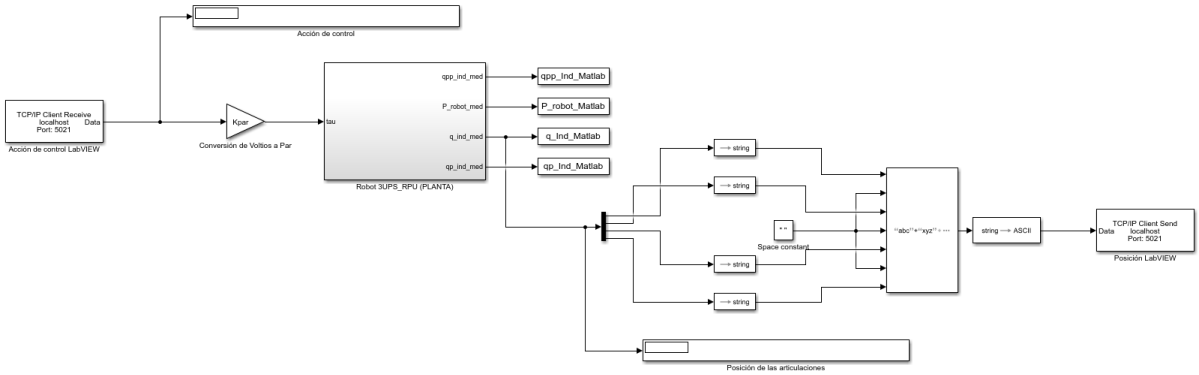


Figura 50: Diagrama de la simulación del robot completo

Realizando pruebas, se ha visto que no se realizaba correctamente la comunicación durante los dos primeros segundos, había muestras que faltaban. Este error se ha solucionado añadiendo una espera de dos segundos antes de comenzar con el bucle de control. De esta manera, no se pierde ninguna muestra.

6.2. RESULTADOS

A continuación, se van a exponer los resultados de los experimentos llevados a cabo con todos los programas y controladores desarrollados. La simulación se ha realizado con los diagramas de Simulink explicados en el apartado anterior.

Los experimentos se han realizado con las siguientes trayectorias cartesianas (en este caso se han muestreado cada 10 ms):

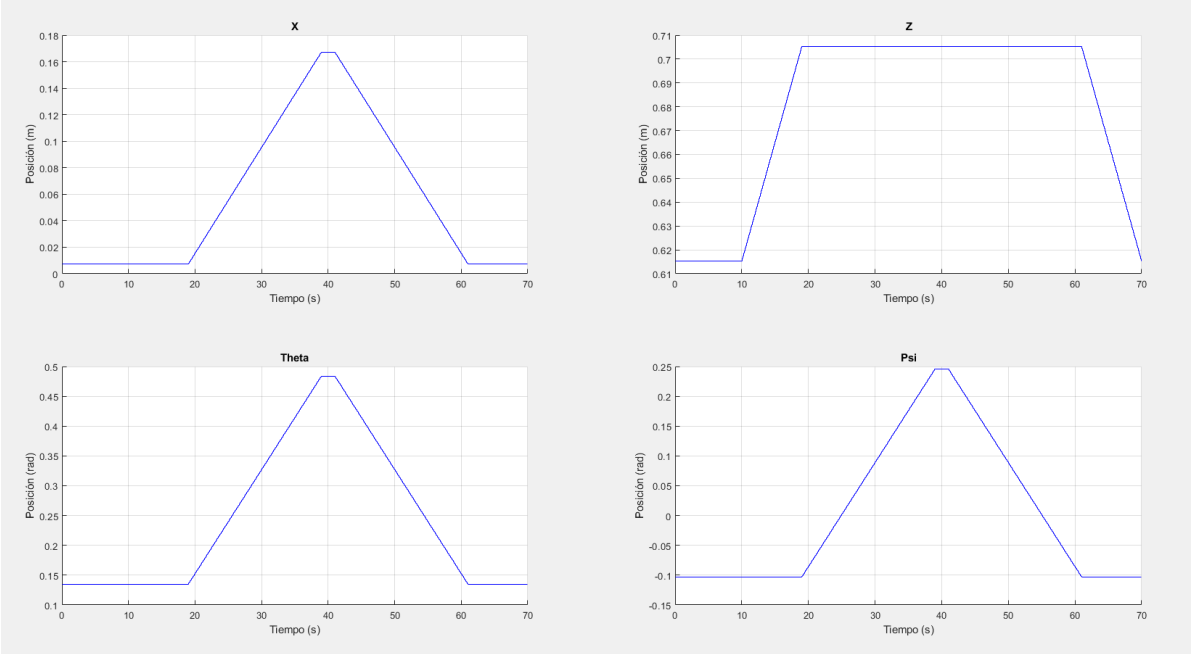


Figura 51: Trayectoria en coordenadas cartesianas de los experimentos

Estas trayectorias corresponden a las siguientes en posiciones de las articulaciones:

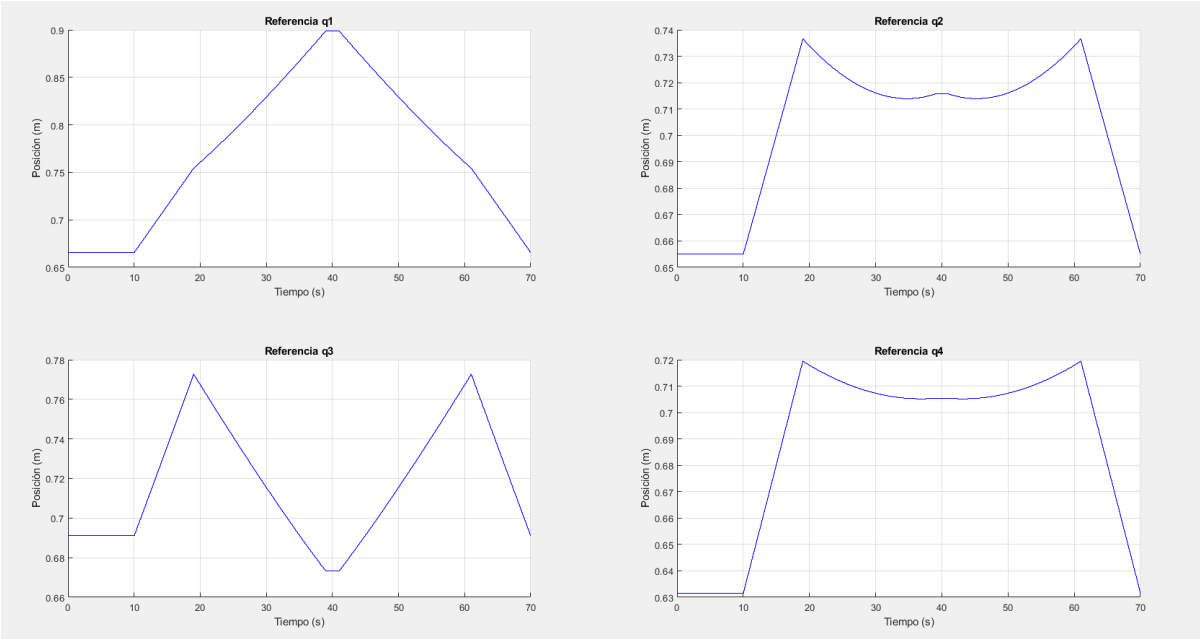


Figura 52: Trayectoria en posiciones de articulaciones de los experimentos

Por último, para cuantificar el error en los experimentos, se ha utilizado el Criterio integral del error cuadrático (CIEC):

$$CIEC = \sum e(k)^2$$

6.2.1. CONTROL DE UN MOTOR

El control de posición de un solo motor se ha realizado con un algoritmo PID como el explicado en el apartado 5. En cuanto a la referencia, se ha utilizado la trayectoria q1. Además, se ha incluido una saturación en la acción de control del controlador de +4 V y -4 V.

En primer lugar, se ha realizado una simulación solamente con una ganancia proporcional (Kp = 1300) y con un periodo de muestreo de 10 ms.

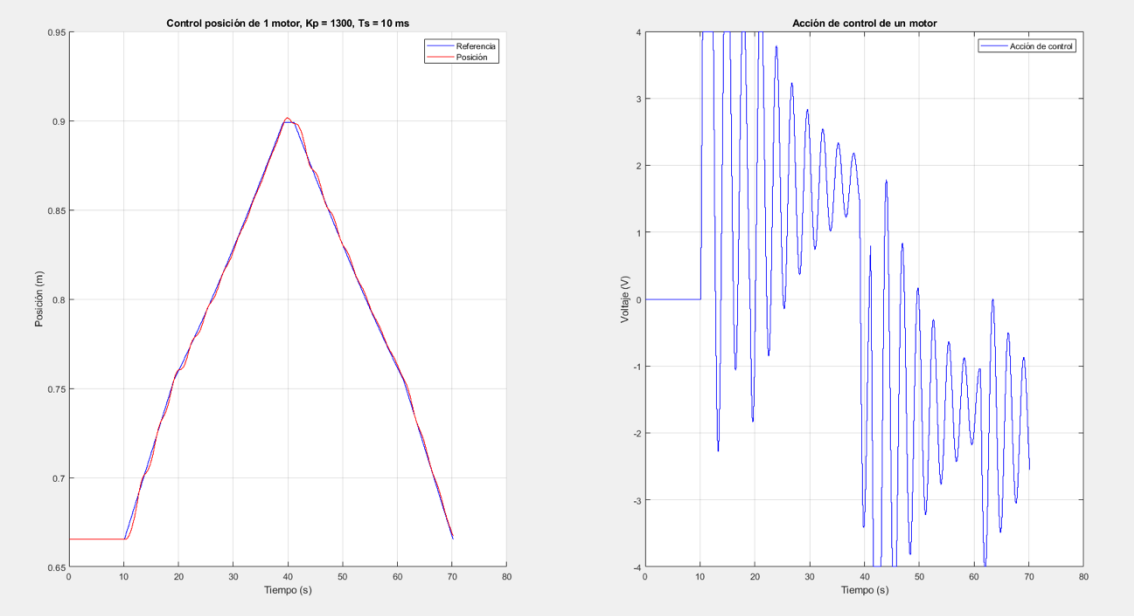


Figura 53: Experimento 1, kp = 1300, Ts = 10 ms

En este caso, el CIEC obtenido es de 0,026.

En el siguiente experimento, se ha aumentado la ganancia proporcional a 3000 y se ha añadido el término derivativo con kv = 200.

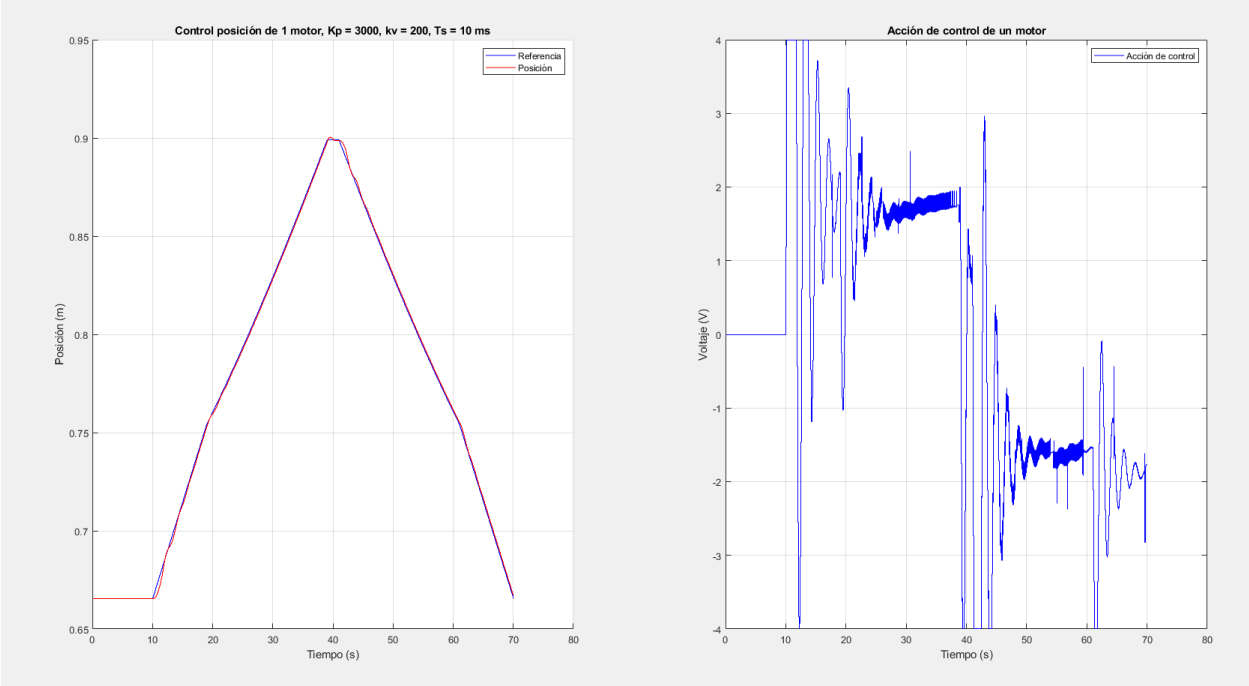


Figura 54: Experimento 2, kp = 3000, kv = 200, Ts = 10 ms

El CIEC resultante es de 0,0118, inferior a la mitad que el anterior.

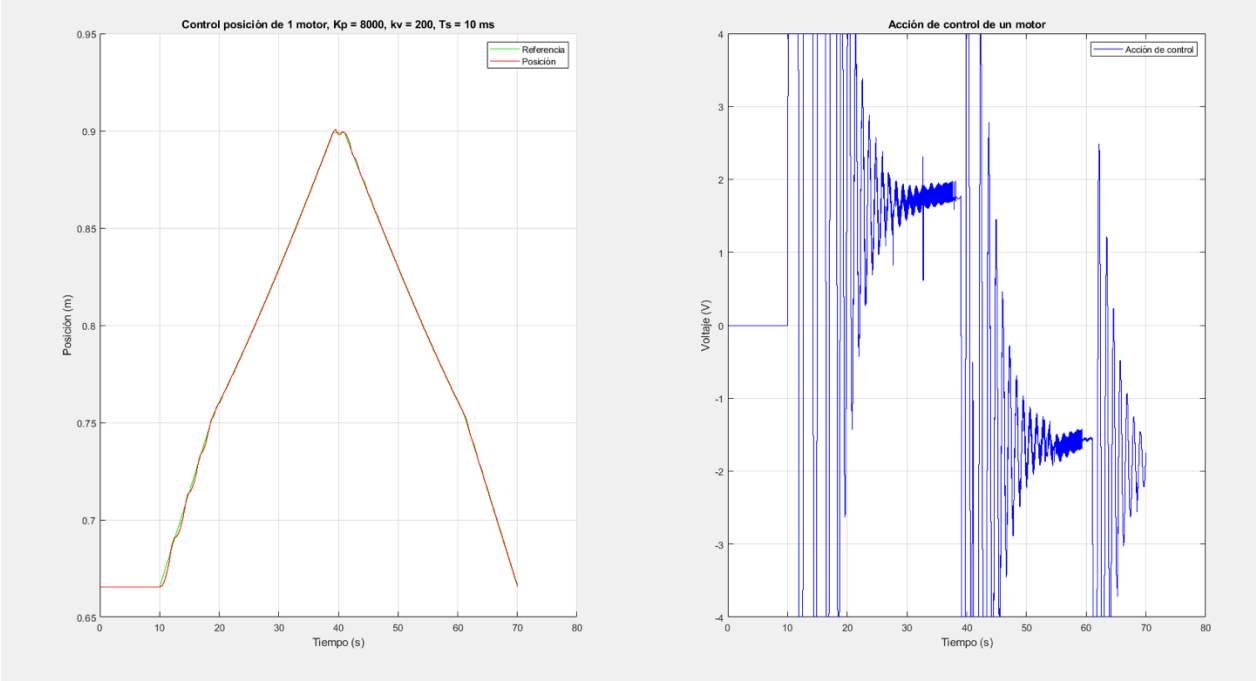


Figura 55: Experimento 3, kp = 8000, kv = 200, Ts = 10 ms

En este tercer experimento, se ha aumentado la ganancia proporcional a 8000. Así, se ha disminuido el CIEC hasta 0,0063.

Por último, a pesar de que las trayectorias fueron diseñadas para ser muestreadas cada 10 ms, para comprobar el funcionamiento, se ha realizado una simulación con un periodo de muestro de 5 ms. El resultado es el siguiente:

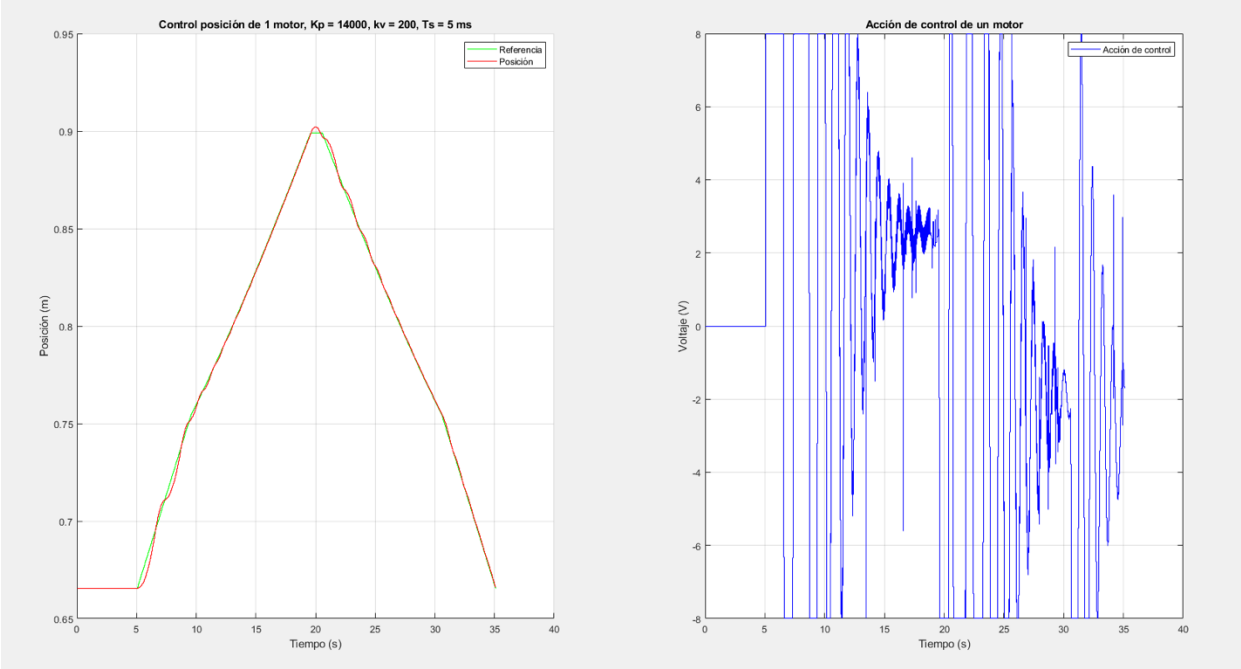


Figura 56: Experimento 4, kp = 14000, kv = 200, Ts = 5 ms

Se ha obtenido un error CIEC de 0,0233.

Tras realizar estos experimentos, como no hay error de posición ni de velocidad al utilizar un controlador PD, no se ha considerado utilizar una acción integral debido a que ralentizaría el sistema y empeoraría los transitorios.

6.2.2. CONTROL DEL ROBOT

Los expermientos que se presentan a continuación se han llevado a cabo utilizando el modelo del robot completo.

En primer lugar, se ha realizado una prueba con un tiempo de muestreo de 10 ms y utilizando un controlador PID cuyos parámetros son: Kp = 650, Kv = 6 y Ki = 10 (Para todos los actuadores). El resultado del control es el siguiente:

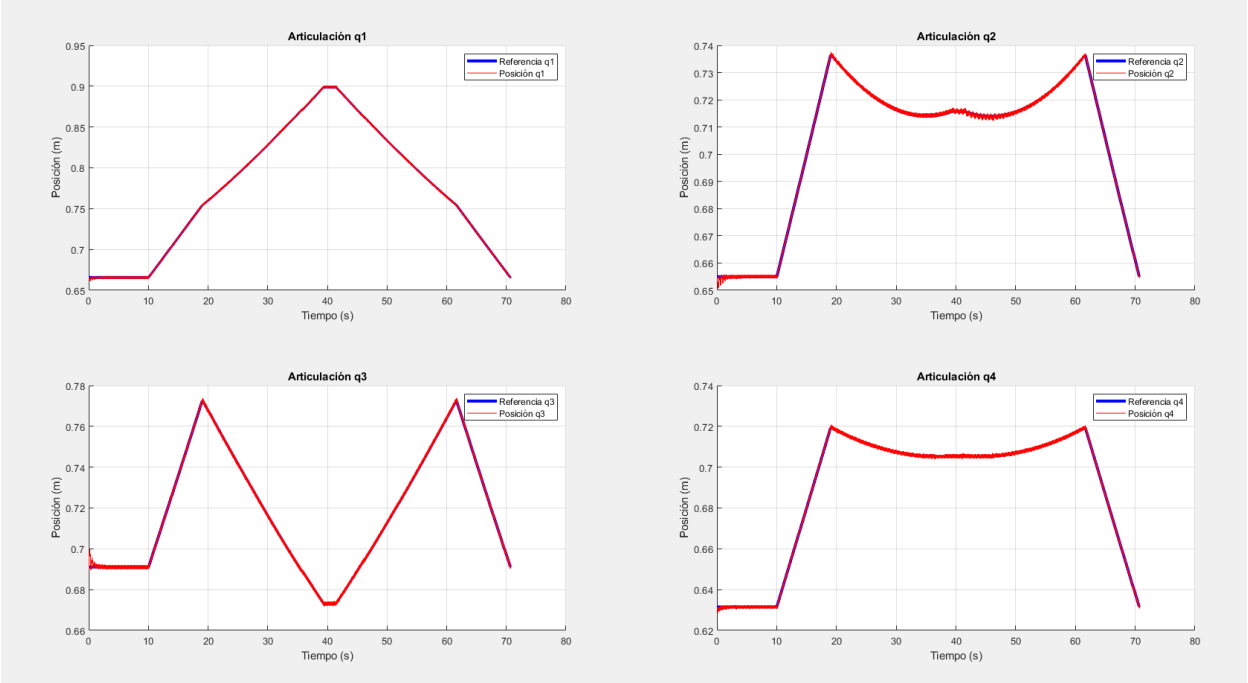


Figura 57: Representación de la referencia y de la posición de las articulaciones con un control PID

Se ha sumado el CIEC obtenido por cada articulación y el resultado es de 0,0223.

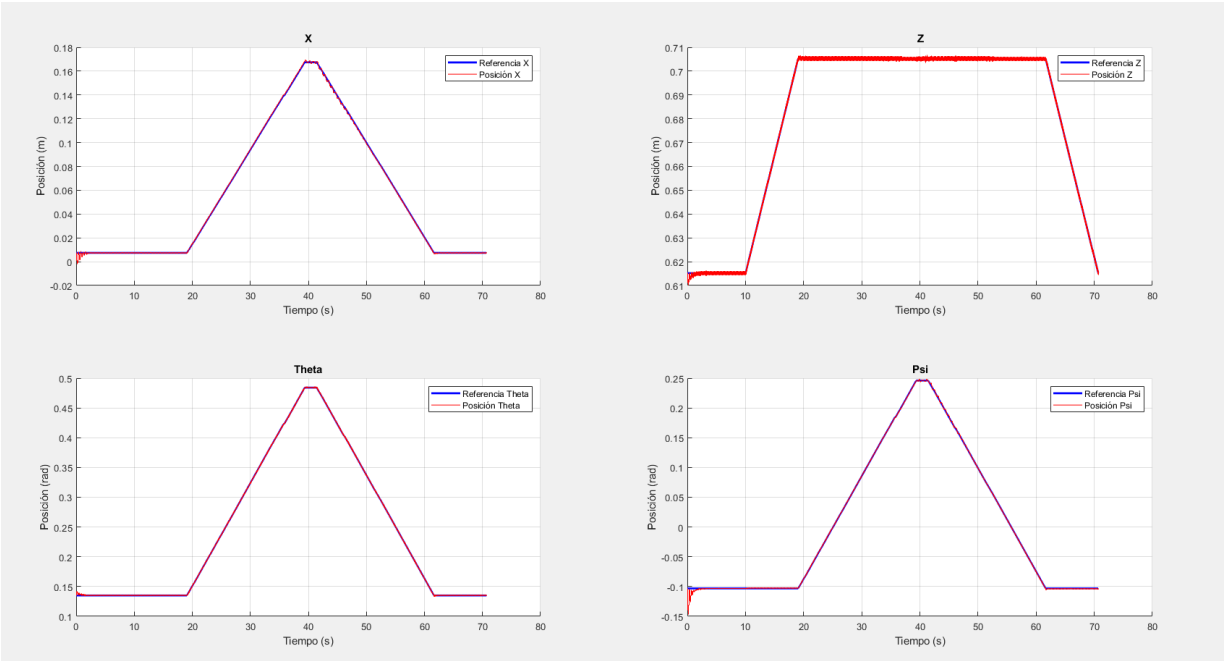


Figura 58: Representación de la referencia y posición cartesiana de la plataforma con un control PID

Los errores obtenidos en cada una de las coordenadas cartesianas se pueden ver en la siguiente tabla:

Errores PID	X	Z	Theta	Psi
CIEC	0,004	0,0054	0,0051	0,0385

Tabla 5: Errores en las coordenadas cartesianas para un control PID

Por último, la acción de control efectuada es la siguiente:

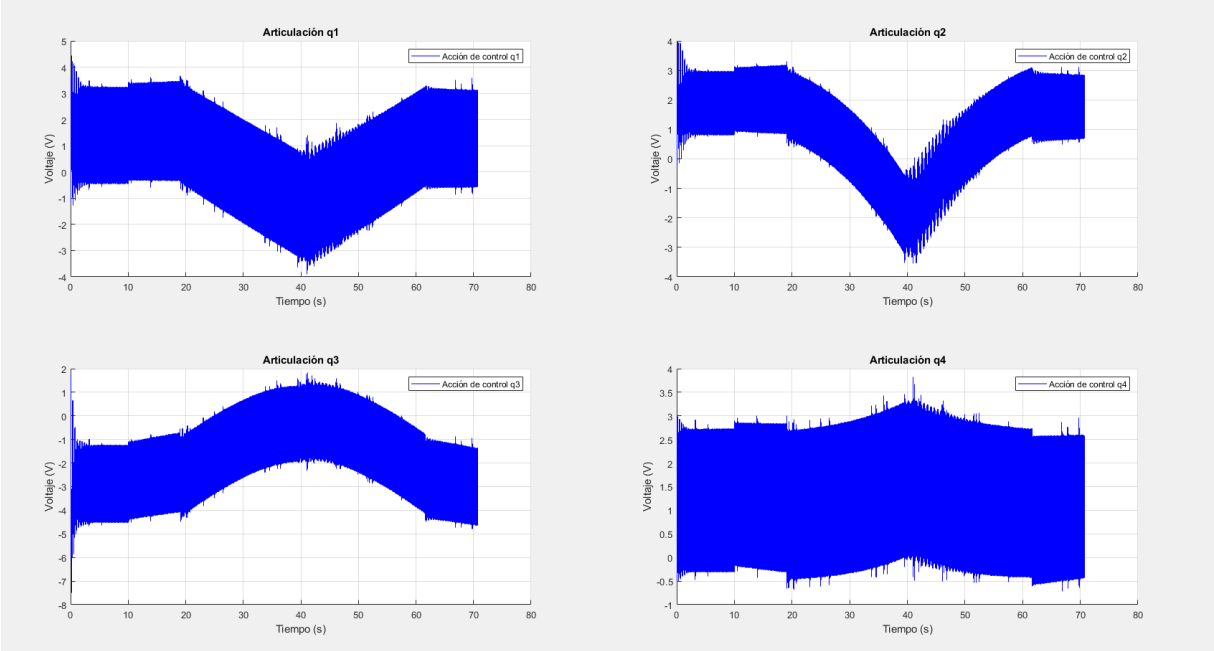


Figura 59: Representación de la acción de control para el control PID

Se puede observar, en las gráficas de posiciones de articulaciones y cartesianas, que hay un transitorio al comienzo de la prueba. Este se debe a la acción que la gravedad ejerce sobre el robot y finalmente es compensado por la acción integral. Sin embargo, este transitorio se puede mejorar con un controlador PD con compensación de la gravedad el cual se expone a continuación (Kp = 650, Kv = 7 y de nuevo Ts = 10 ms):

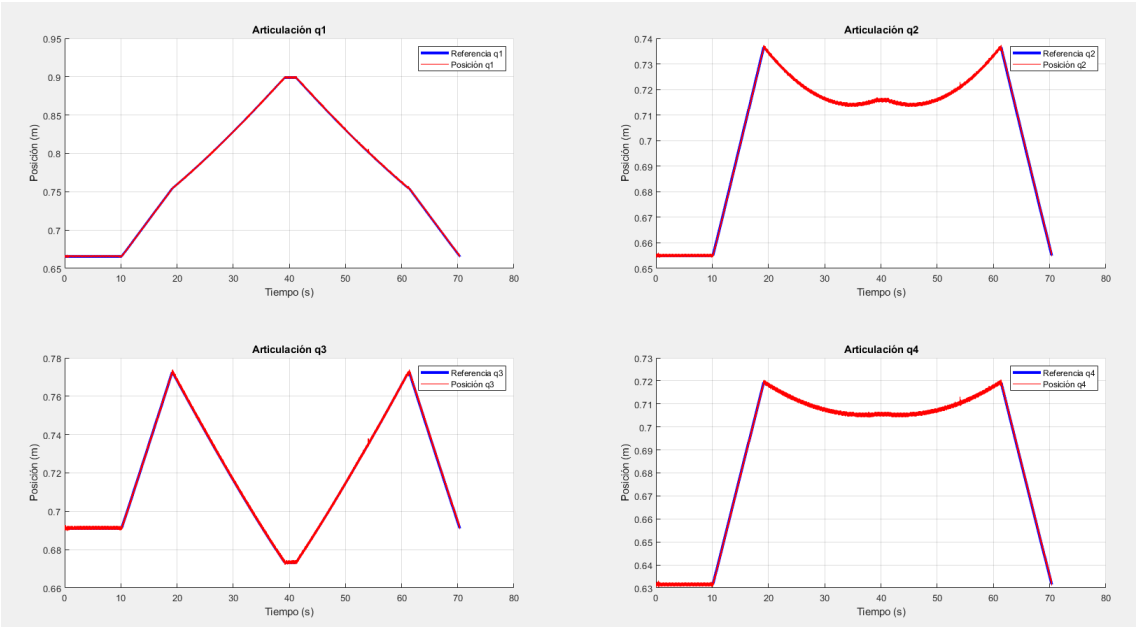


Figura 60: Representación de la referencia y las posiciones de las articulaciones con un control PDG

Para este caso, el error obtenido en la posición de las articulaciones es de 0,0229.

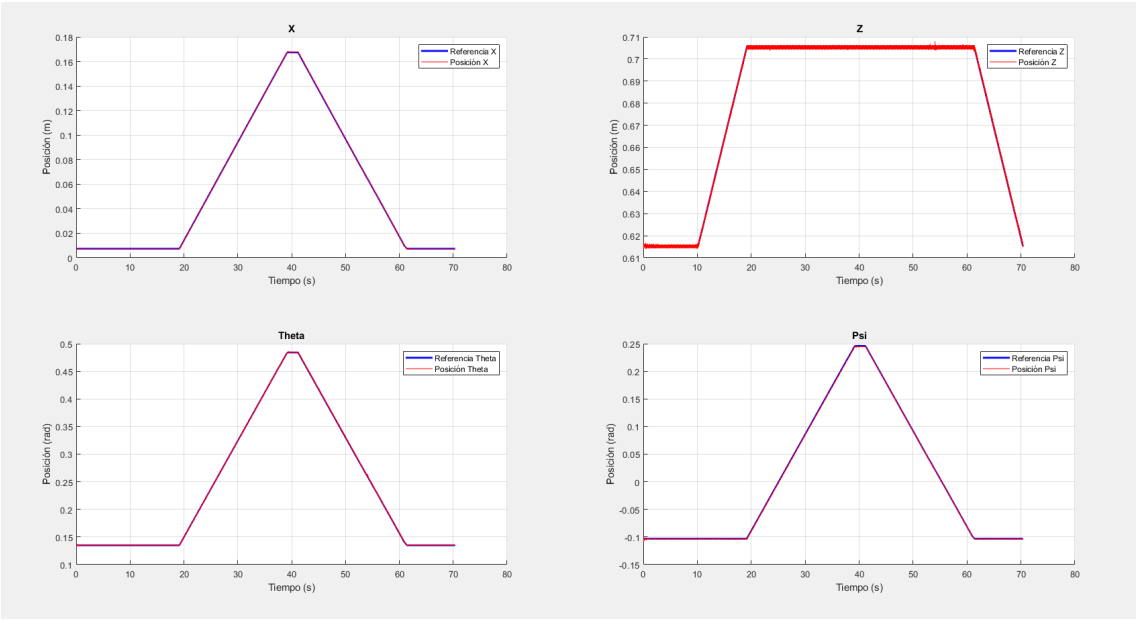


Figura 61: Representación de la referencia y posición cartesiana de la plataforma con un control PDG

Se puede observar que el transitorio ha disminuido considerablemente. Por tanto, una compensación de la gravedad funciona más rápido que un control integral ya que la compensación depende directamente de la posición de las articulaciones activas. Sin embargo, es necesario tener un modelo físico del robot muy fiel a la realidad, ya que, en el caso de que hubiera inexactitudes, se producirían errores de posición.

En este caso, los errores obtenidos son los siguientes:

PD + G (10ms)	X	Z	Theta	Psi
CIEC	0,00042	0,0053	0,0059	0,0076

Tabla 6: Errores en las coordenadas cartesianas para un control PD con compensación de la gravedad y Ts = 10 ms

La acción de control ejercida es la siguiente:

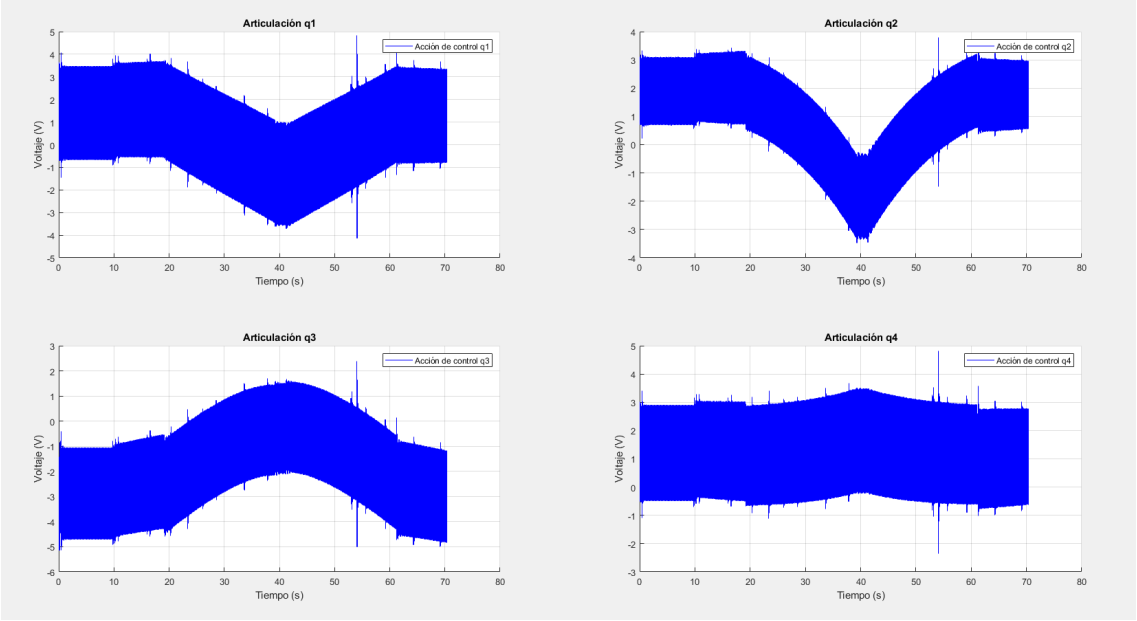


Figura 62: Representación de la acción de control con un control PDG

Finalmente, se ha realizado una prueba controlando el robot con un tiempo de muestreo de 5 ms y con el mismo controlador que en la simulación anterior. El resultado se muestra a continuación:

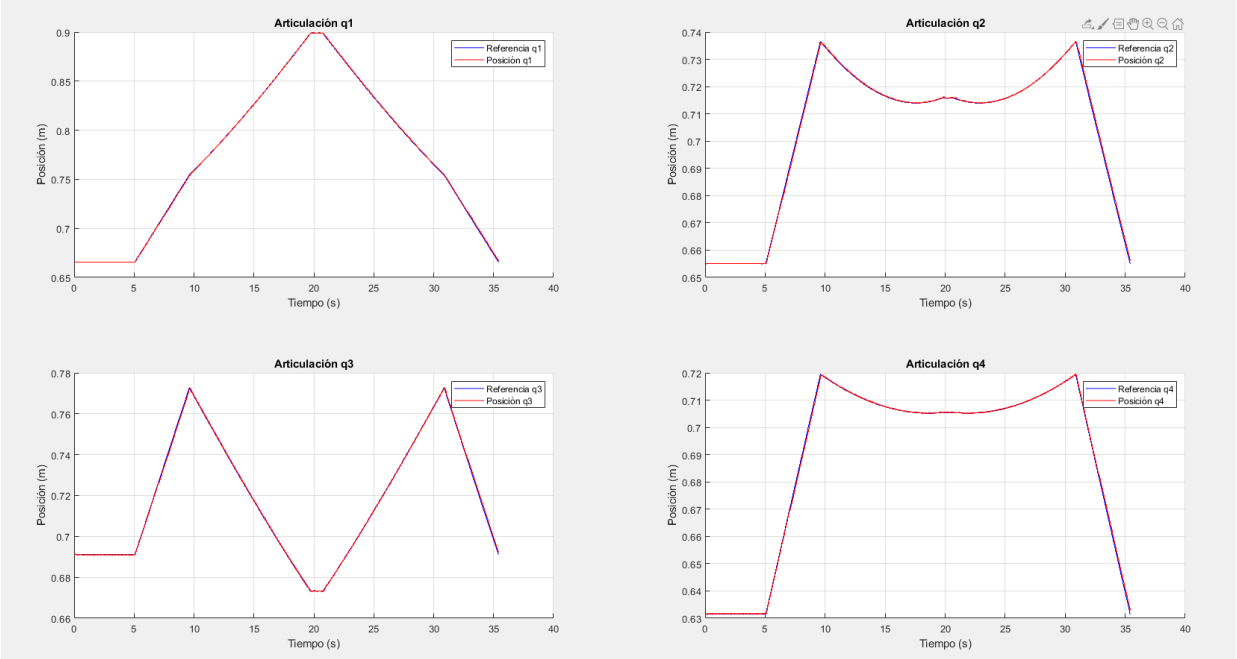


Figura 63: Representación de la referencia y las posiciones de las articulaciones con un control PDG y Ts = 5 ms

Ahora, la suma del error ha disminuido hasta 0,0101.

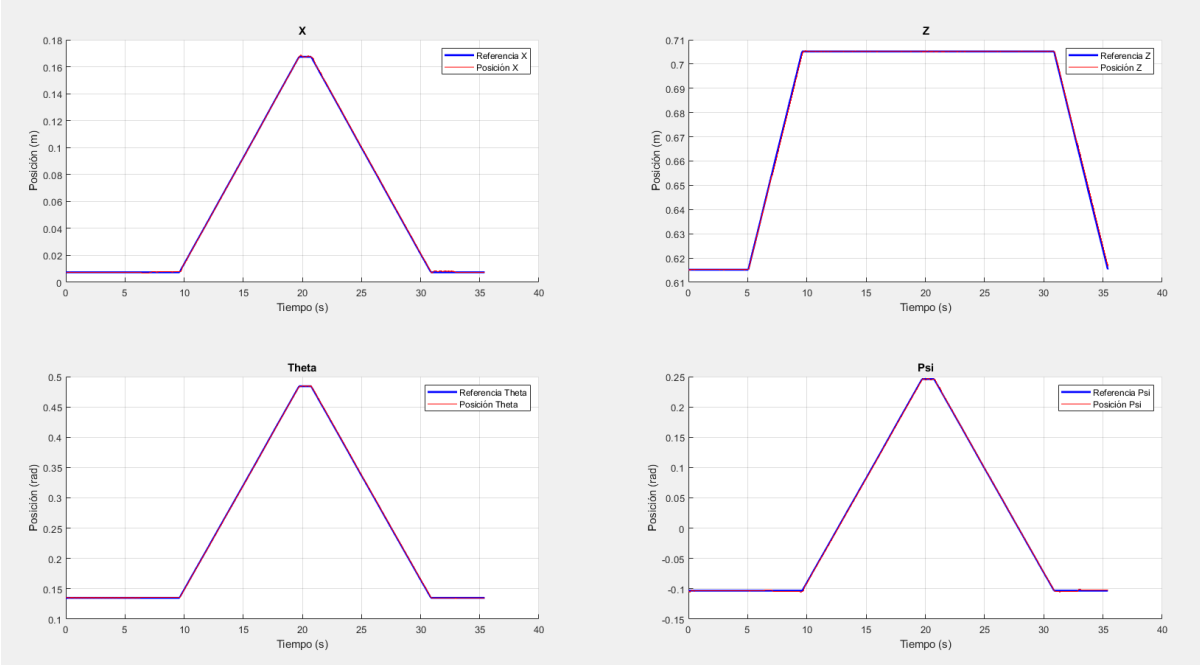


Figura 64: Representación de la referencia y de la posición cartesiana de la plataforma con un control PDG y Ts = 5 ms

Los errores obtenidos en la posición de las coordenadas cartesianas son los siguientes:

Errores PD+G (5 ms)	X	Z	Theta	Psi
CIEC	0,0016	0,0026	0,0058	0,0088

Tabla 7: Errores en las coordenadas cartesianas para un control PD con compensación de la gravedad y Ts = 5 ms

Por último, la acción de control es la siguiente:

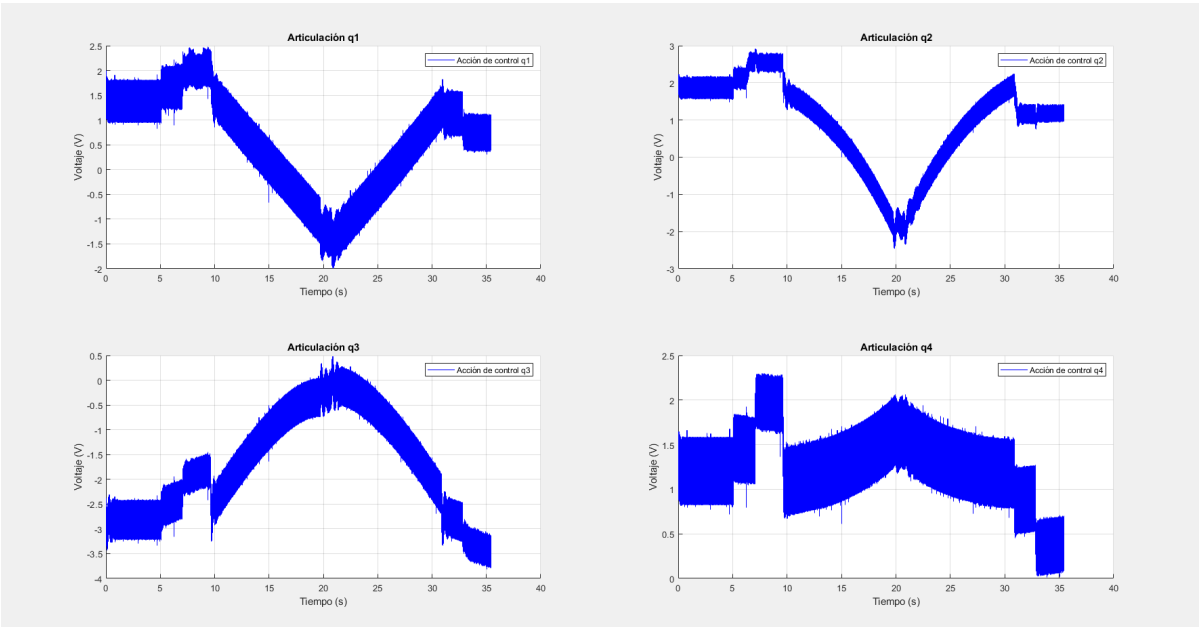


Figura 65: Representación de la acción de control con un control PDG y Ts = 5 ms

7. CONCLUSIONES

En este proyecto de fin de grado se ha diseñado una arquitectura de control para un robot paralelo de cuatro grados de libertad. Se ha seleccionado un hardware que cumple con las especificaciones y se han estudiado los requerimientos temporales necesarios en el bucle de control, para, en función de estos, organizar la implementación, las labores de comunicación y de almacenaje de datos de manera que se cumpla siempre el periodo de muestreo y no se pierda ninguna muestra. Dentro de esta arquitectura, se han implementado algoritmos para controlar la posición de la plataforma del robot paralelo.

Durante el periodo de diseño de la arquitectura de control, se puede extraer que no conviene almacenar variables de un tamaño grande, por ejemplo, vectores de más de mil datos, en la memoria de la Compact RIO. Por este motivo, enviar las referencias desde el ordenador al controlador cada cierto tiempo ofrece mejores resultados que almacenar el vector de referencias completo.

En segundo lugar, se ha diseñado una placa de circuito impreso, que además de simplificar las conexiones y reducir el cableado entre el controlador y el robot, adecua las señales entre el módulo de entrada digital de la Compact RIO y de la salida de los encoders.

Al no disponer del proceso real, se ha comprobado el funcionamiento computacional de los controladores implementados con una arquitectura similar a la real, pero estableciendo una comunicación con Simulink mediante TCP/IP.

Faltaría por probar la implementación con el robot real, sin embargo, se espera un funcionamiento correcto debido a que los controladores implementados ya se habían probado con el robot real antes de este trabajo y ahora, con el modelo, se ha comprobado el buen funcionamiento a nivel computacional en LabVIEW. Y también debido a que ya se estableció el control de un motor real con la Compact RIO antes de la crisis del COVID-19, por lo que el manejo de las señales también es correcto.

A partir de este trabajo, se podrán añadir otras estrategias de control alternativas como un control de fuerza, en la que se tenga en cuenta la resistencia que ofrece el paciente al movimiento. Esto sería posible gracias a los módulos que incorpora la Compact RIO que no se han utilizado en este proyecto.

Finalmente, con el fin de obtener una patente, se deberá validar el software asegurando que cumple con la normativa. Para ello, se pueden añadir funcionalidades de seguridad como un Watchdog, ya que el controlador Compact RIO permite esta función.

ANEXOS

A I. MANUAL DE USUARIO

Este manual explica la utilización de los sistemas que se han desarrollado en el proyecto, incluyendo los elementos hardware como el controlador y la placa de conexión, así como el uso de los programas.

UNIDAD DE CONTROL

De una manera esquemática, las conexiones a realizar para instalar el procesador, chasis y los módulos son las siguientes:

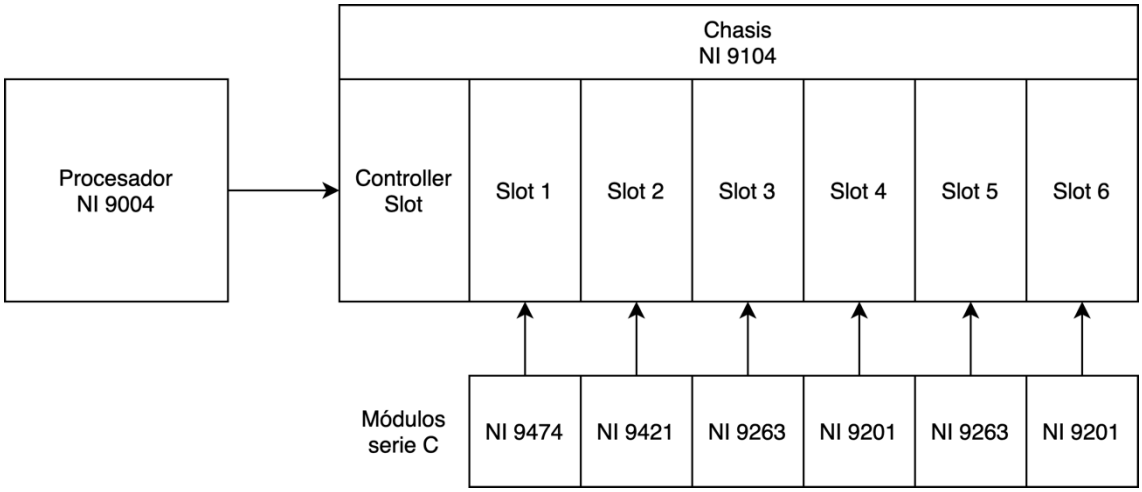


Figura 66: Diagrama de instalación del controlador

Para más información sobre la instalación física, consúltese el manual de usuario del chasis 9104 adjuntado en la documentación técnica.

El controlador debe conectarse a la red Ethernet a través del puerto RJ-45, que se ubica en el procesador NI cRIO-9004, si se necesita más información, consultar el manual de usuario de este dispositivo.

PLACA DE CONEXIÓN COMPACT RIO - CUADRO DE CONTROL

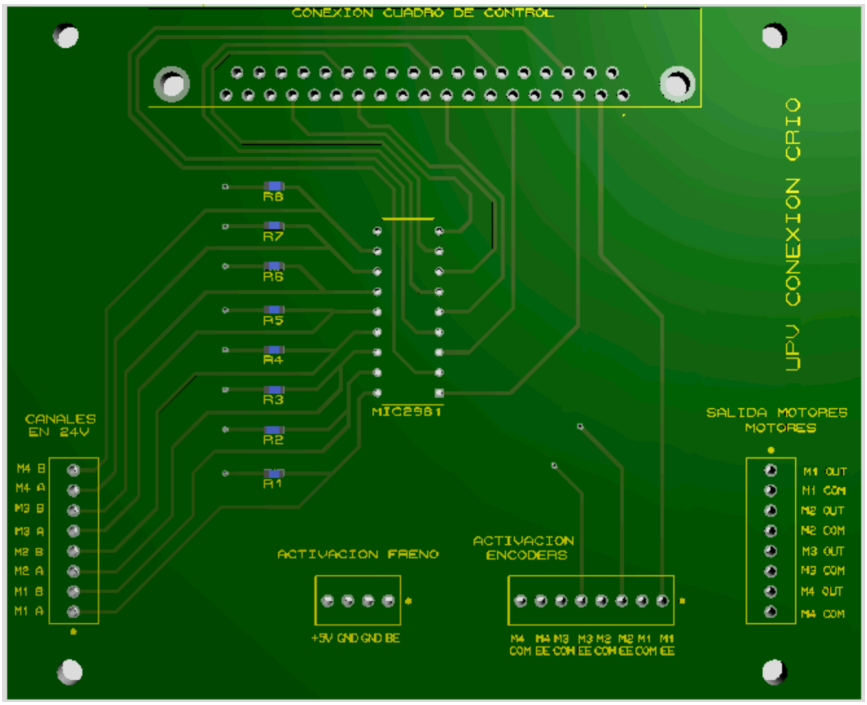


Figura 67: Vista en planta del modelo de la PCB

Esta placa simplifica el conexionado entre el controlador y los elementos del robot. Las conexiones que se deben realizar con respecto a los módulos de entradas y salidas del controlador son las siguientes:

NI - 9474	Activación freno y encoders	NI – 9421	Canales en 24 V	NI - 2963	Salida motores
DO0	M1 EE	DI0	M1 A	AO0	M1 OUT
DO1	M2 EE	DI1	M1 B	COM	M1 COM
DO2	M3 EE	DI2	M2 A	AO1	M2 OUT
DO3	M4 EE	DI3	M2 B	COM	M2 COM
DO4	BE	DI4	M3 A	AO2	M3 OUT
DO5	-	DI5	M3 B	COM	M3 COM
DO6	-	DI6	M4 A	AO3	M4 OUT
DO7	-	DI7	M4 B	COM	M4 COM
Vsup	+5 V	COM	GND		
COM	GND				

Tabla 8: Conexiones entre los módulos E/S y la placa

LISTA DE PROGRAMAS Y JERARQUÍA

La lista de programas para realizar experimentos son los siguientes, se incluyen también los modelos .mdl de Simulink:

Servidor	Controlador real	Controlador simulado	Arquitectura de control	Planta
Servidor_PC_v1.vi	Control_RT_v1.vi	Control_RT_v1_SIM.vi HIL_1Motor.mdl	Envío completo de las referencias cartesianas o de articulaciones	1 motor
Servidor_PC_v2.vi	Control_RT_v2.vi	Control_RT_v2_SIM.vi HIL_1Motor.mdl	Envío único de referencias cartesianas	1 motor
Servidor_PC_v3.vi	Control_RT_v3_PID.vi Control_RT_v3_PDG.vi	Control_RT_v3_SIM_PID.vi Control_RT_v3_SIM_PDG.vi HIL_Robot.mdl	Envío de múltiples referencias cartesianas	Robot

Tabla 9: Lista de programas

La jerarquía en el proyecto de LabVIEW es la que se muestra a continuación:

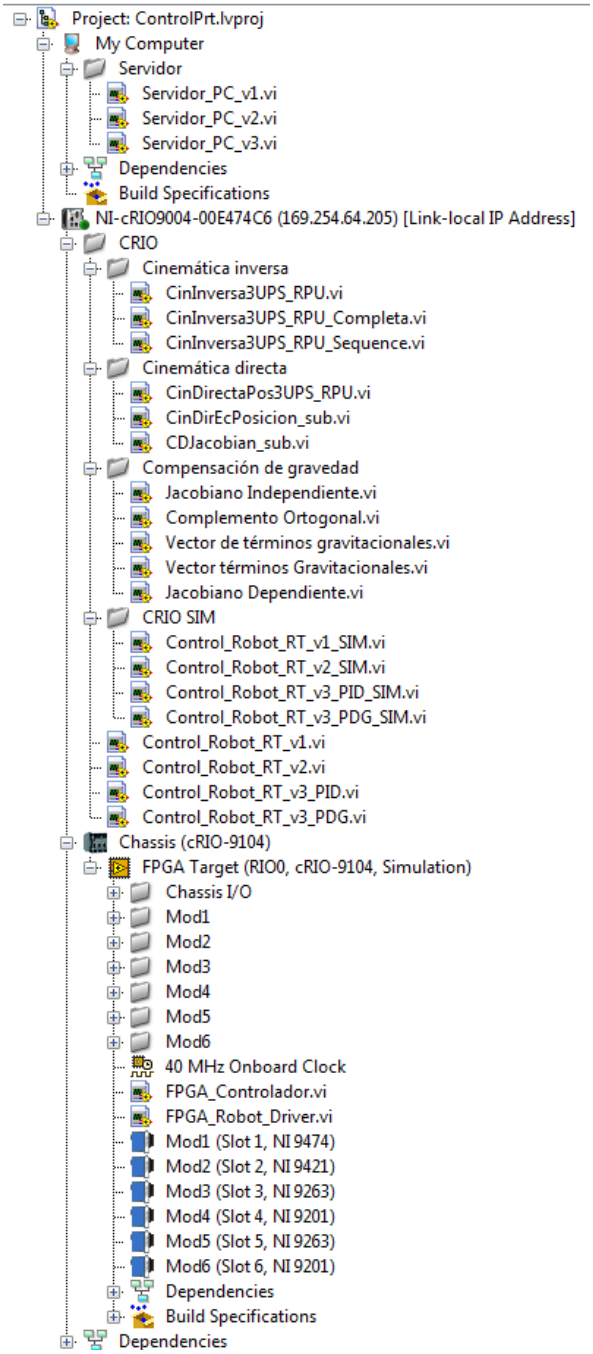


Figura 68: Jerarquía del proyecto

CONTROL DE UN MOTOR

Para realizar un experimento real con el envío de las referencias completas, se deben ejecutar los programas Servidor_PC_v1.vi en el ordenador y Control _v1.vi.

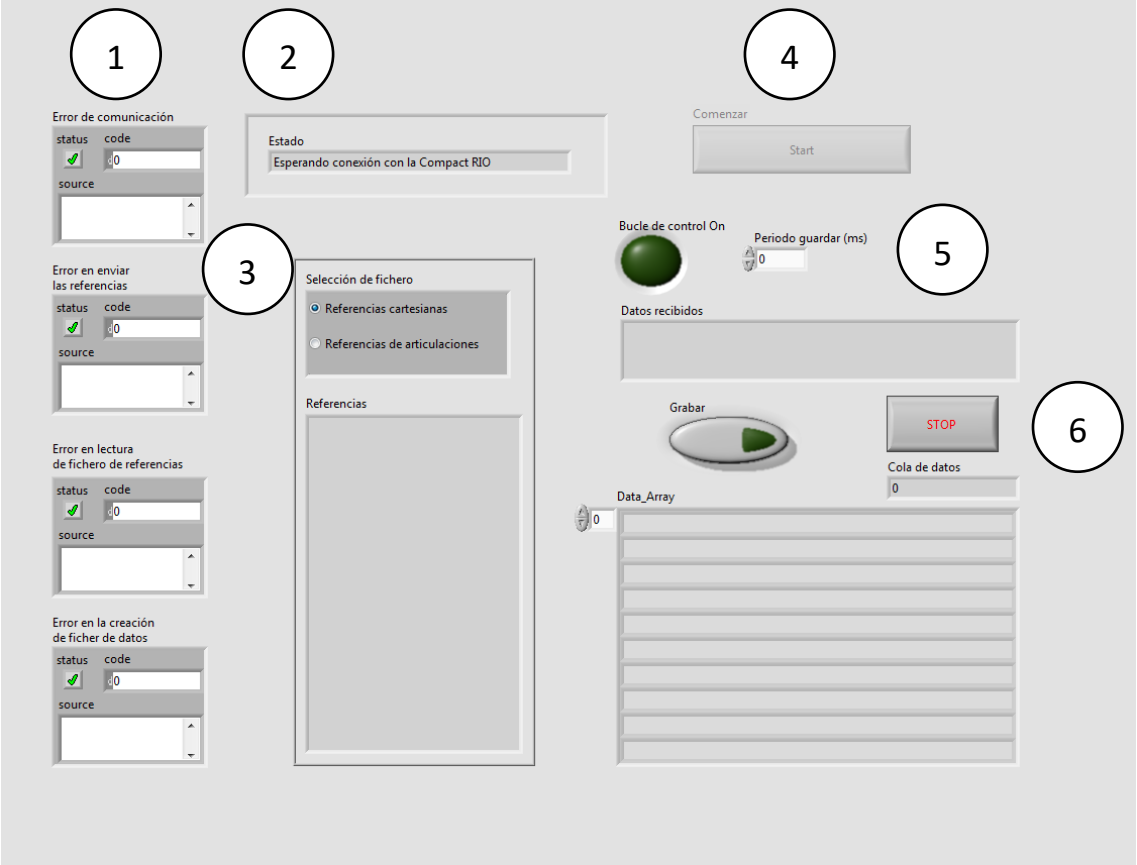


Figura 69: Panel frontal de Servidor_PC_v1.vi con indicadores de guía

1. **Indicadores de error:** Mediante estos indicadores se puede visualizar si ha habido algún error durante la comunicación con el controlador y a la hora de leer el fichero de referencias.
2. **Indicador de estado:** Aquí se indica si la conexión con la Compact RIO se ha efectuado o no.
3. **Controles de selección del tipo de referencia:** Utilizando este control, se indica si el fichero se encuentra en coordenadas cartesianas o en referencias de articulaciones.
4. **Botón de inicio:** Una vez se ha realizado la conexión con la CRIO, se podrá pulsar este botón para que comience el experimento.
5. **Selección de periodo de guardado:** En esta entrada se selecciona el periodo con el que se desea leer los datos de cada iteración del control.
6. **Botón de grabado y de paro:** Si se quieren guardar los datos del bucle de control en un fichero, se debe activar este botón.

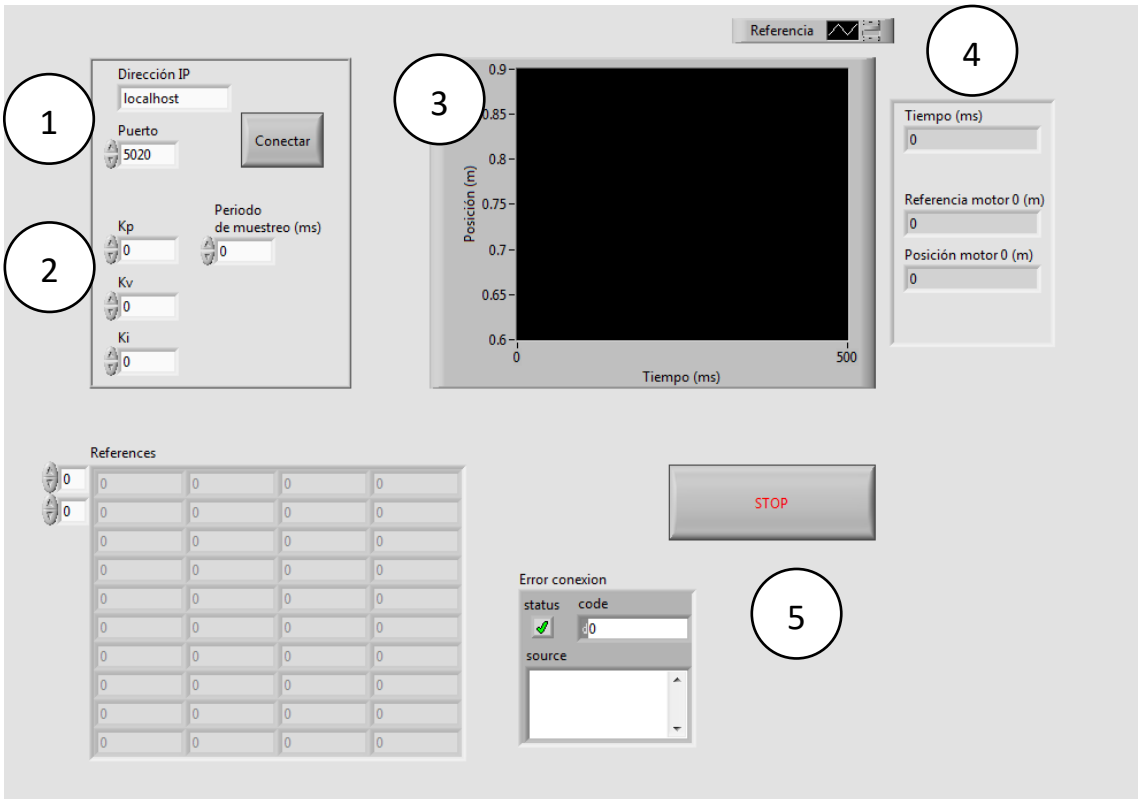


Figura 70: Panel frontal de Control_RT_v1.vi con indicadores de guía

1. **Controles de conexión con el ordenador:** A través de estos, se introduce la dirección IP del ordenador y el puerto que se utilice. Mediante el botón de conectar, se conecta la Compact RIO con el servidor.
2. **Selección de parámetros:** Aquí se pueden modificar los parámetros del controlador PID así como el periodo de muestreo.
3. **Visualización gráfica:** En la gráfica se pueden visualizar la posición del motor (blanco) y la referencia (rojo).
4. **Visualización numérica:** Se visualiza la misma información que en la gráfica, pero numéricamente.
5. **Botón de paro:** Se debe pulsar si se quiere acabar con el experimento encontrándose este a mitad. Se terminará el bucle de control y se escribirá una última acción de control de 0 V con el fin de parar el actuador.

En el caso de utilizar un único envío, se deben ejecutar los programas Servidor_PC_v2.vi junto con Control_RT_v2.vi cuyos controles son idénticos a los mencionados anteriormente.

CONTROL DEL ROBOT COMPLETO

Para controlar el robot completo hay que ejecutar los programas Servidor_PC_v3.vi y en el controlador Control_RT_v3_PID.vi ó Control_RT_v3_PDG.vi dependiendo de si se quiere un control PID (con el primero) o un control con compensación de gravedad (el segundo programa.

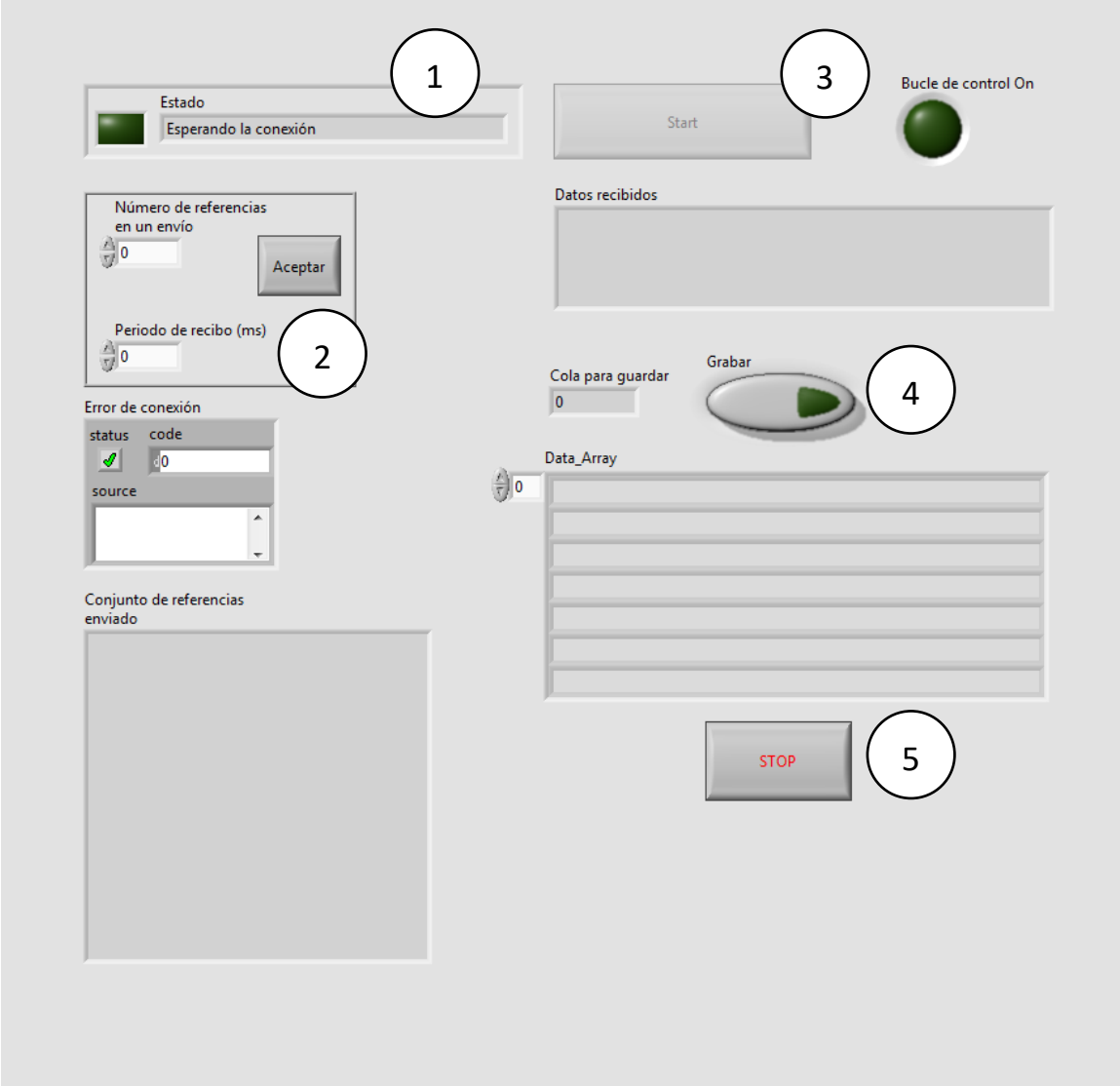


Figura 71: Panel frontal de Servidor_PC_v3.vi con indicadores numéricos

1. **Indicador de estado:** Se muestra si se ha realizado la conexión con el controlador o no.
2. **Configuración de parámetros de comunicación:** Mediante estos controles se puede elegir el número de referencias que hay en un envío además del periodo de muestreo y del periodo de recibir los datos. Cuando se haya decidido, se debe pulsar el botón de aceptar.
3. **Botón de inicio:** Cuando se haya efectuado la conexión y se haya pulsado el botón de aceptar anterior, se podrá activar este botón para que comience el control.

4. **Botón de grabar:** Del mismo modo que anteriormente, se debe pulsar este botón si se quieren monitorizar los datos de la prueba.
5. **Botón de paro:** Pulsar para que el programa termine.

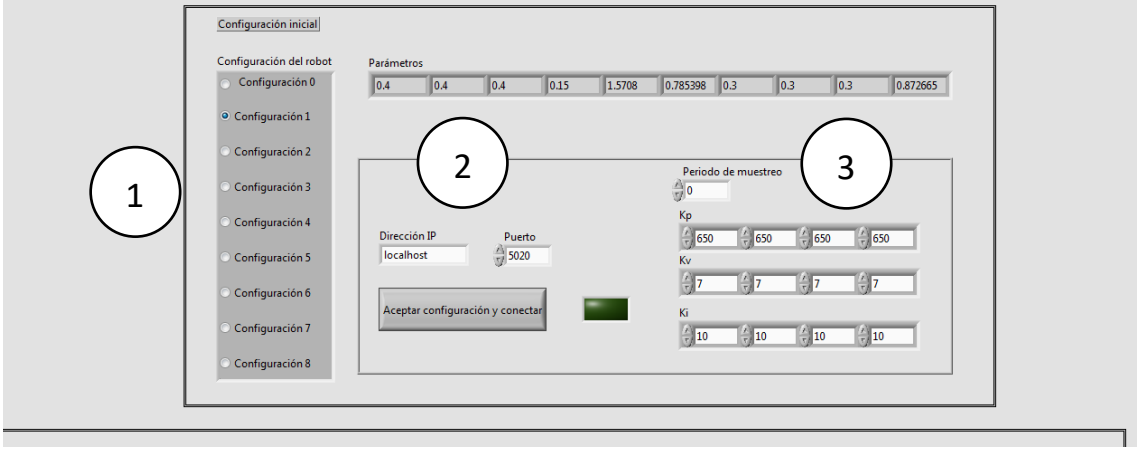


Figura 72: Panel frontal Control_RT_v3_PID.vi con indicadores numéricos 1

Figura 73: Panel frontal Control_RT_v3_PID.vi con indicadores numéricos 2

1. **Configuración de parámetros del robot:** Aquí se puede seleccionar una de las 8 configuraciones que tiene el robot, se debe hacer antes de conectar el controlador con el ordenador.
2. **Configuración de la comunicación:** Utilizando estos controles, se selecciona la dirección IP y puerto del servidor y se pulsa el botón para efectuar la conexión con este.

3. **Configuración de parámetros del controlador:** Aquí se pueden modificar los parámetros del controlador, cada columna corresponde con el control local de un motor. Además, se puede seleccionar el periodo de muestreo.
4. **Visualización gráfica y numérica:** Aquí se muestra una visualización gráfica y numérica de cada motor. Se representan las mismas señales que anteriormente y del mismo color.
5. **Botón de paro:** Este botón funciona como paro de emergencia en el caso de que haya un problema durante una prueba, si se activa, se termina con el bucle de control, se escribe un voltaje de 0 V en todos los actuadores y se activan los frenos.

FORMATOS DE FICHEROS DE ENTRADA Y DE SALIDA

El formato del fichero de referencias debe estar formado por cuatro columnas separadas por espacios y con el siguiente orden:

Articulaciones	Ref q1	Ref q2	Ref q3	Ref q4
Cartesianas	Ref X	Ref Y	Ref Theta	Ref Psi

Tabla 10: Formato del fichero de referencias

Por otro lado, en el fichero de datos, se escriben en columnas con el siguiente orden:

Tiempo	Refq1	Posq1	Refq2	Posq2	Refq3	Posq3	Refq4	Posq4	RefX	RefY	Ref Theta	Ref Psi	PosX	PosY	Pos Theta	Pos Psi

Tabla 11: Orden del fichero de datos

SIMULACIÓN

Para realizar la simulación, se deben seguir los pasos anteriores, con la diferencia de que hay que ejecutar los programas con la terminación "_SIM". Cuando se presione el botón de comenzar, el programa espera a que se ejecute el modelo de Simulink. Para ello, si se quiere controlar un motor, se debe ejecutar el modelo HIL_1motor.mdl y para el caso del robot completo, HIL_Robot.mdl.

Estos programas se deben ejecutar moviendo la carpeta virtual del proyecto "CRIO" debajo de "My Computer", para que los programas se ejecuten localmente en el ordenador.

A II. CÓDIGO DE PROGRAMACIÓN

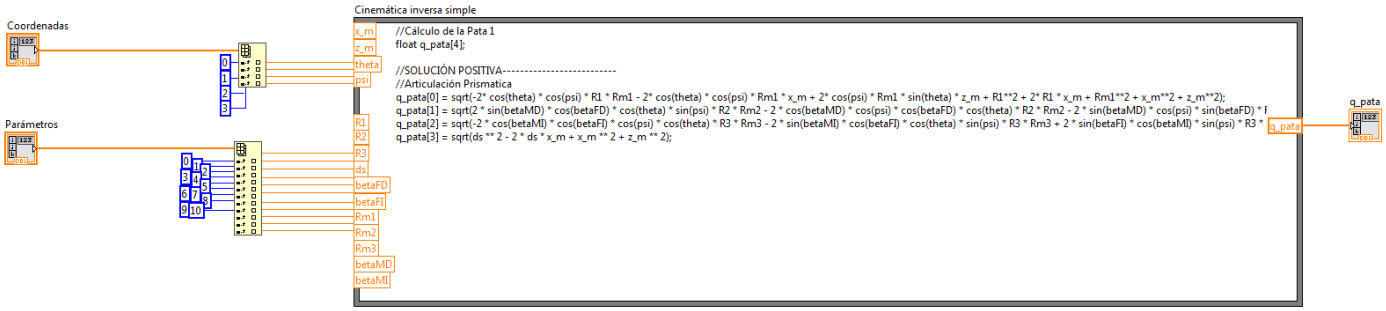


Figura 74: Cinemática inversa simple

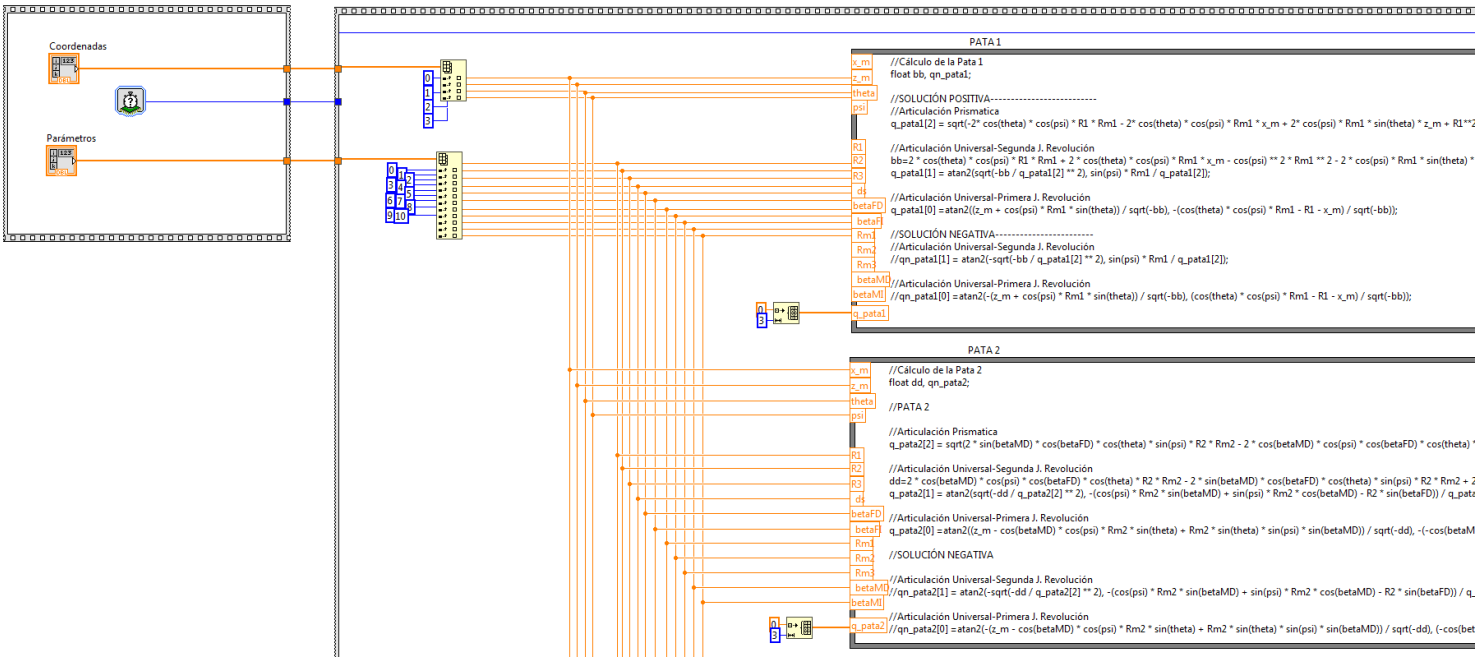


Figura 75: Cinemática inversa completa 1

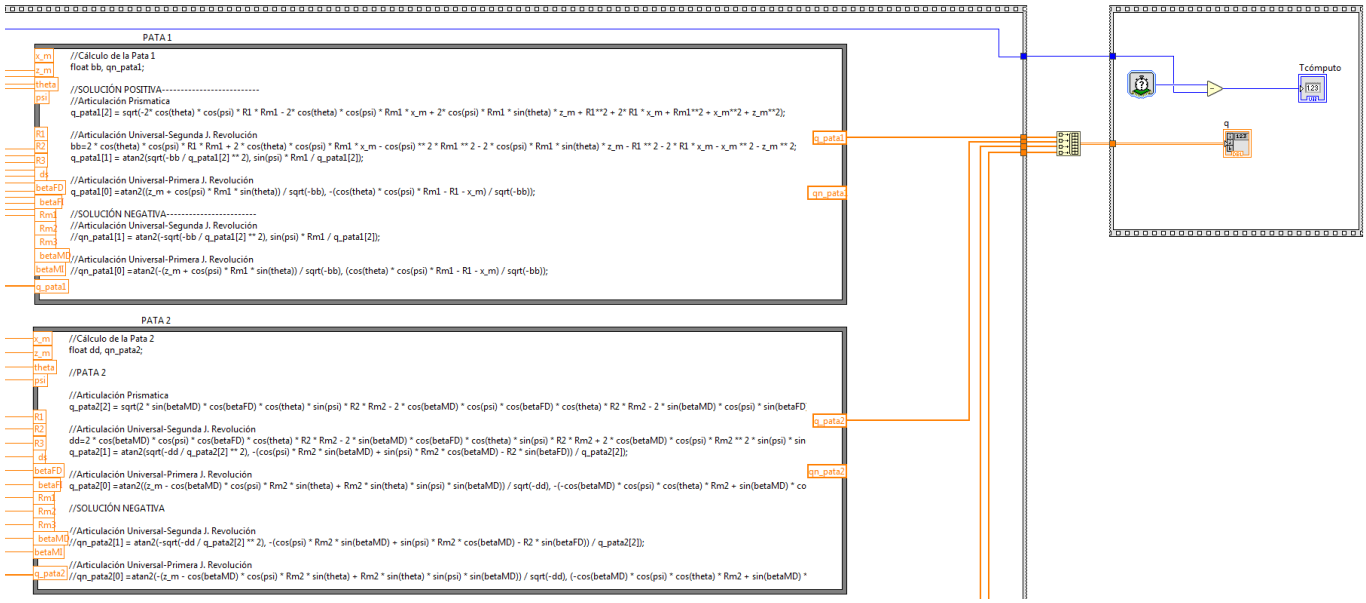


Figura 76: Cinemática inversa completa 2

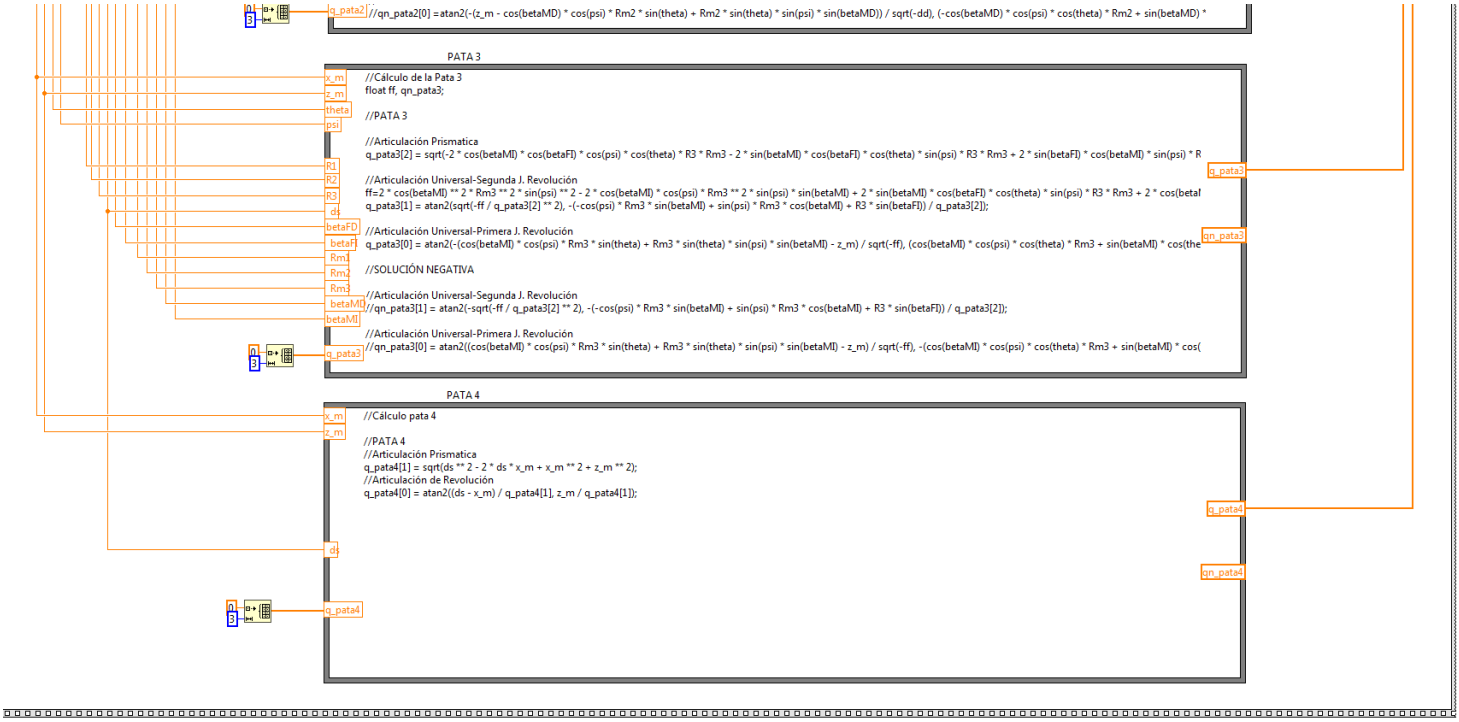


Figura 77: Cinemática inversa completa 3

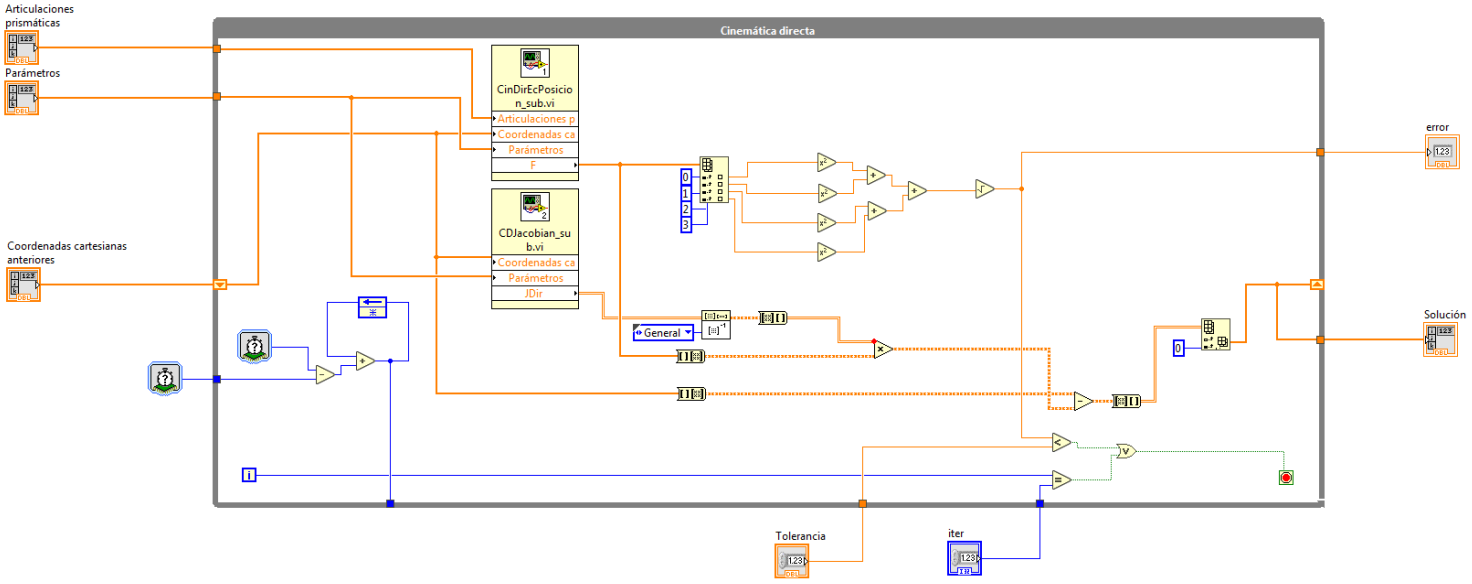


Figura 78: Cinemática directa

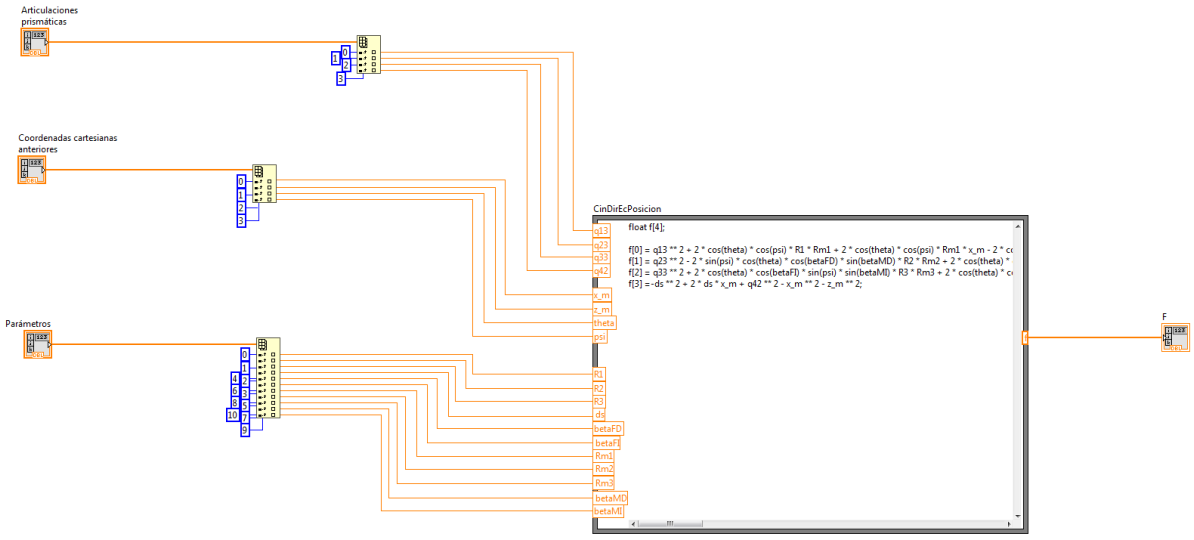


Figura 79: Cálculo de matriz de ecuaciones F(x)

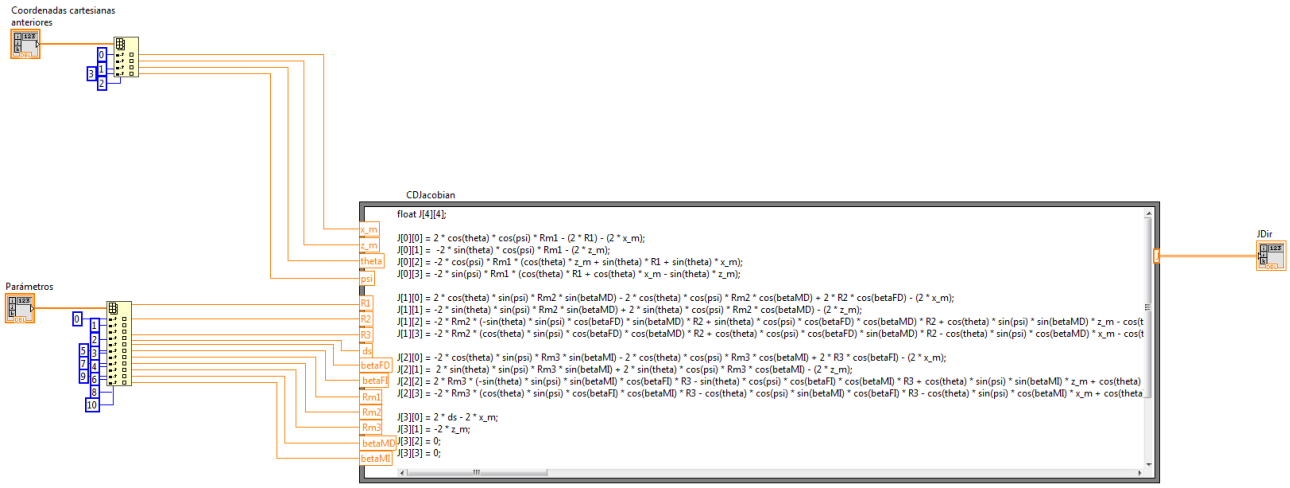


Figura 80: Cálculo del Jacobiano Directo

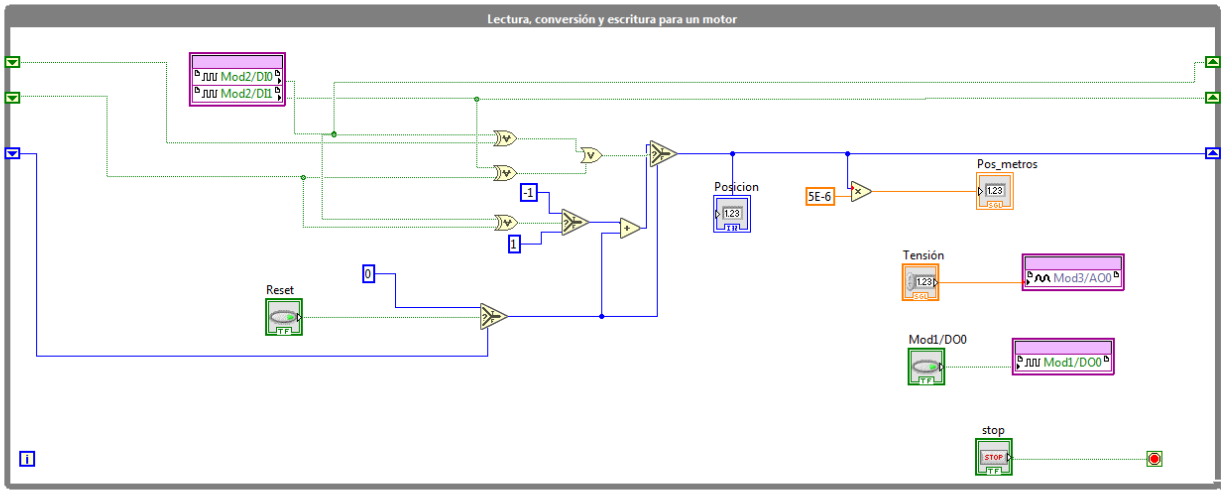


Figura 81: Lectura, conversión y escritura mediante FPGA para 1 motor

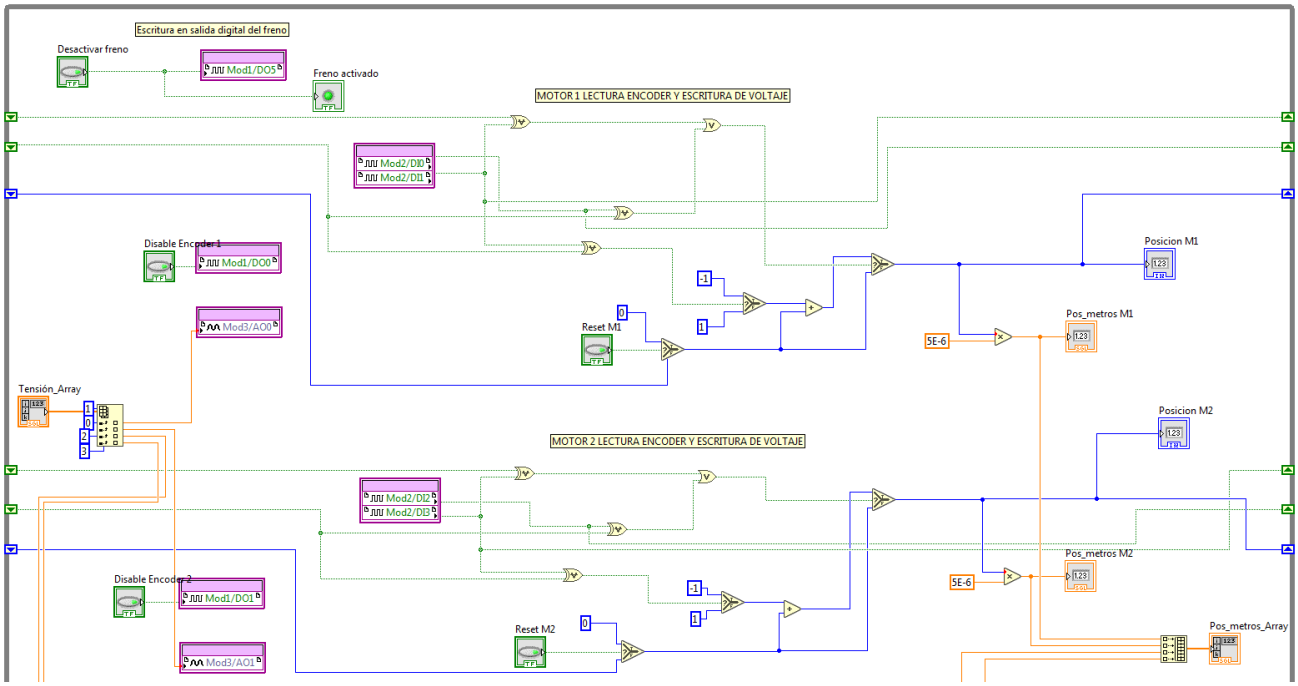


Figura 82: Lectura, conversión y escritura mediante FPGA para el robot completo 1

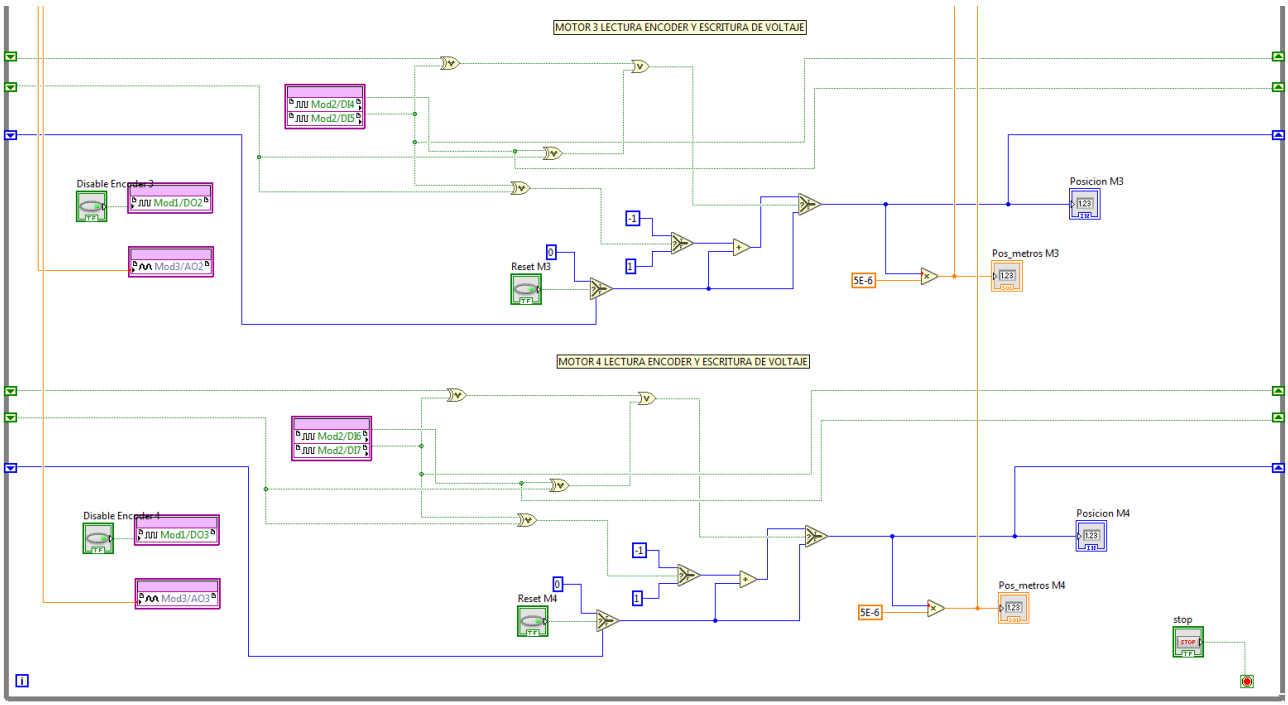


Figura 83: Lectura, conversión y escritura mediante FPGA para el robot completo 2

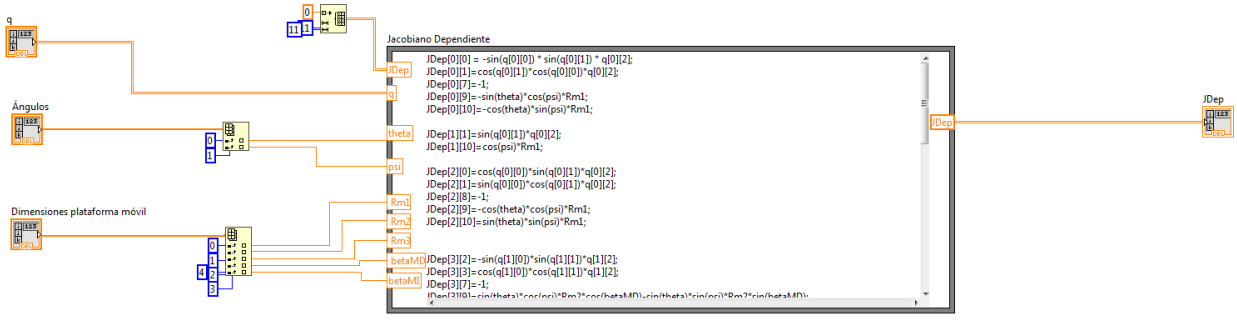


Figura 84: Compensación de gravedad, cálculo de Jacobiano Dependiente

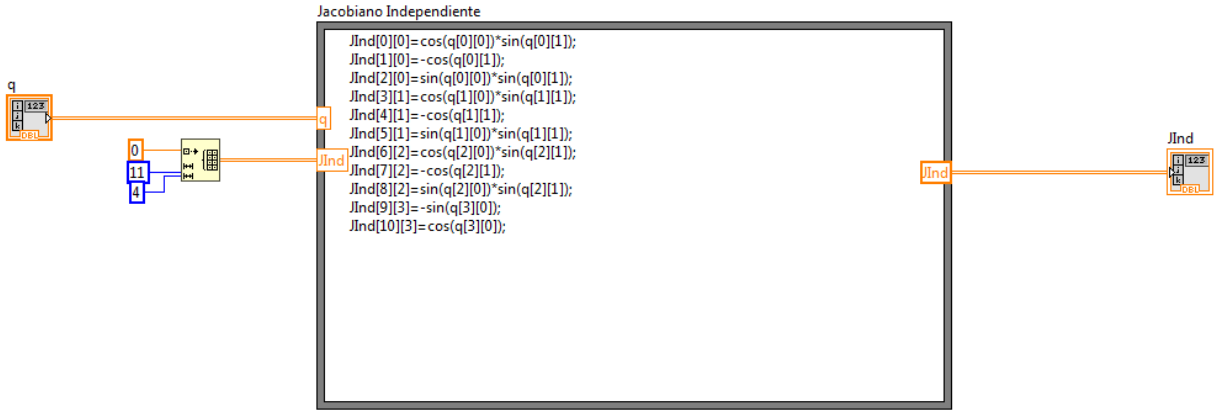


Figura 85: Compensación de gravedad, cálculo de Jacobiano Independiente

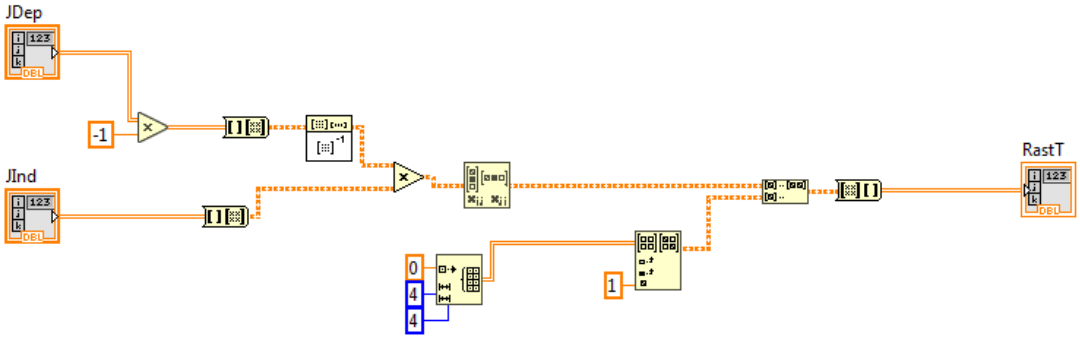


Figura 86: Compensación de gravedad, cálculo del complemento ortogonal

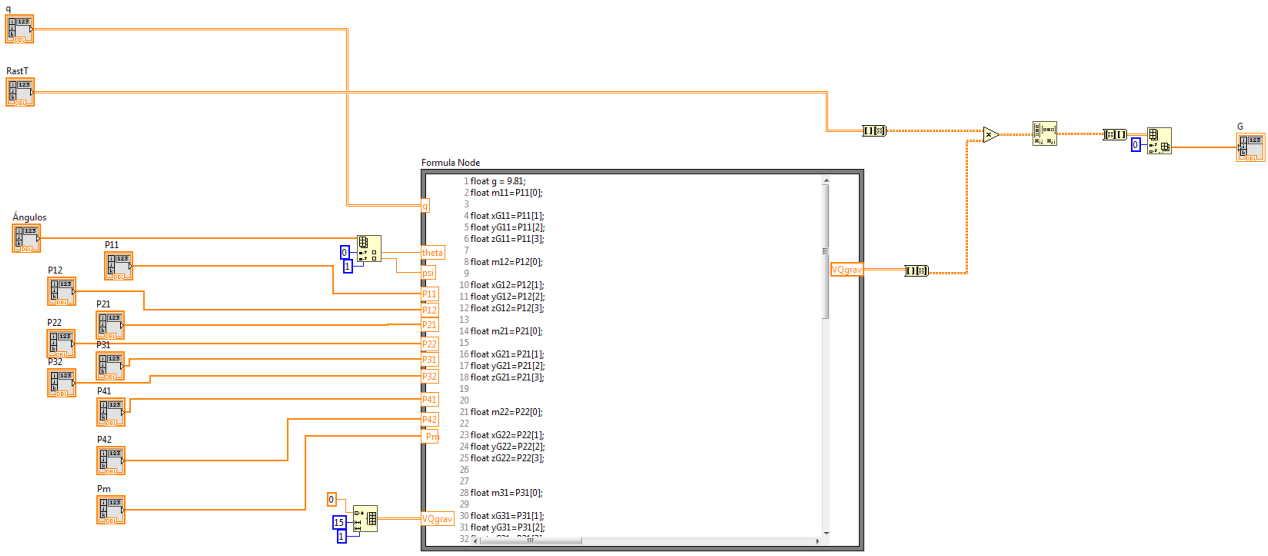


Figura 87: Compensación de gravedad, cálculo de vector de términos gravitatorios

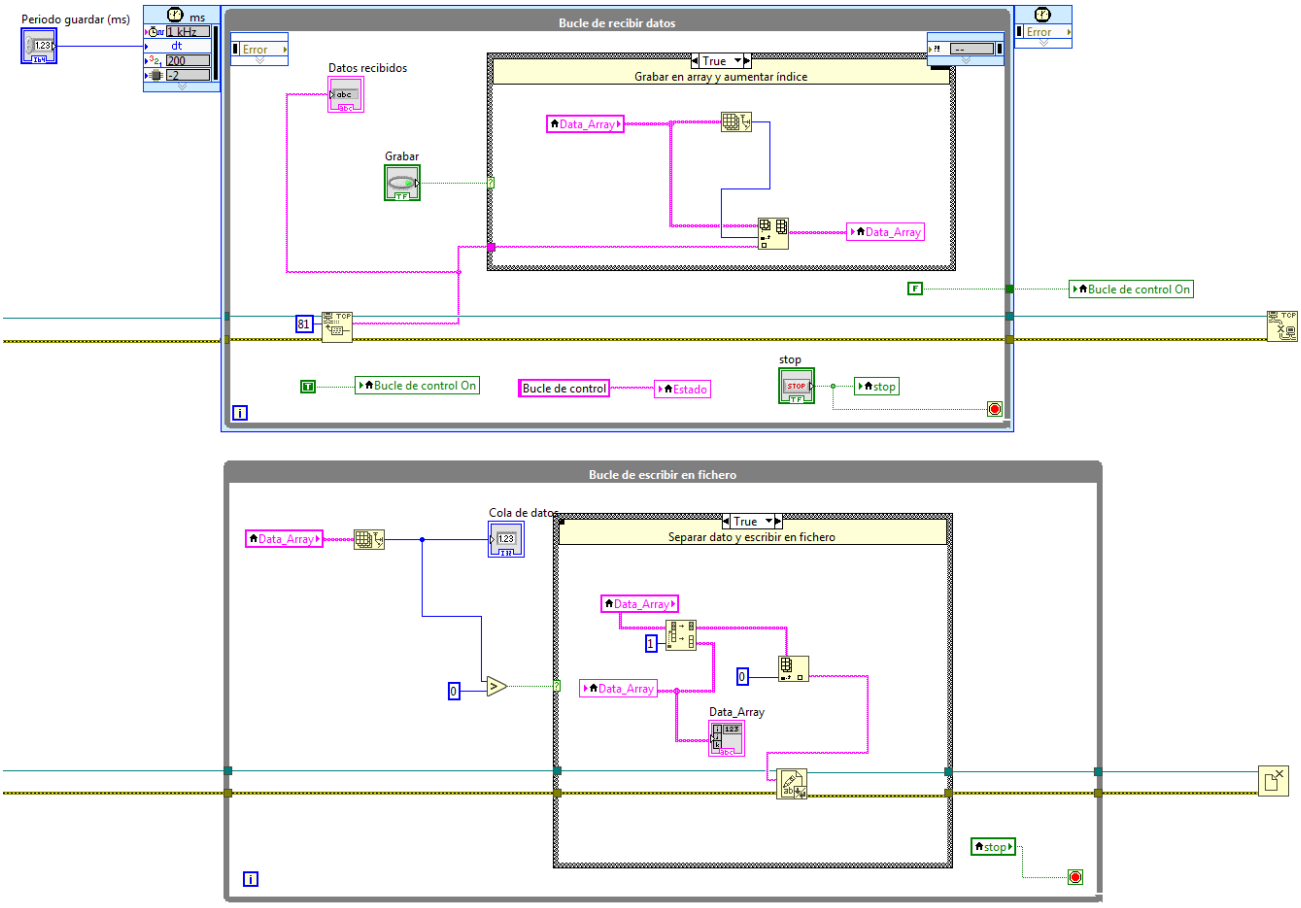


Figura 88: Búfer para recibir datos

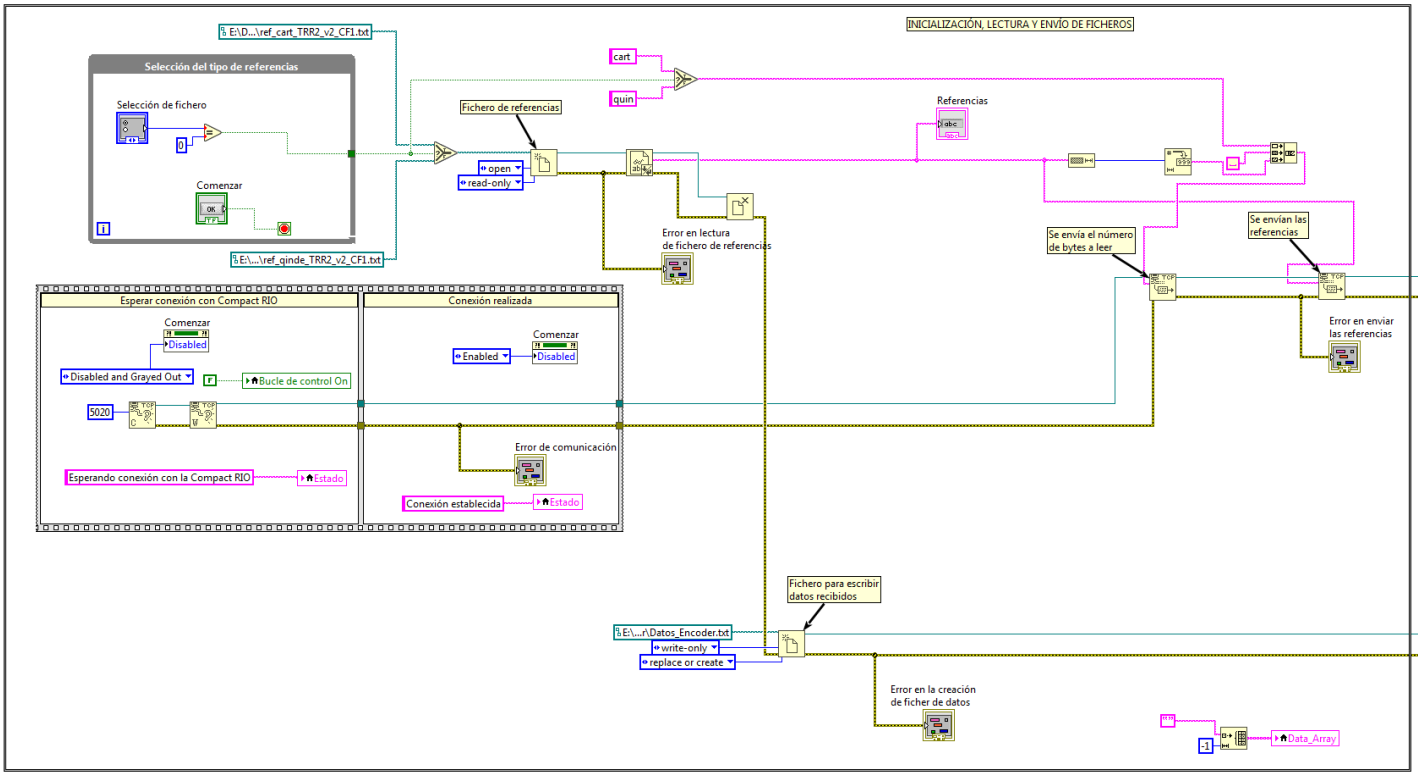


Figura 89: Inicialización programa para Servidor de envío de referencias completas de articulaciones y cartesianas

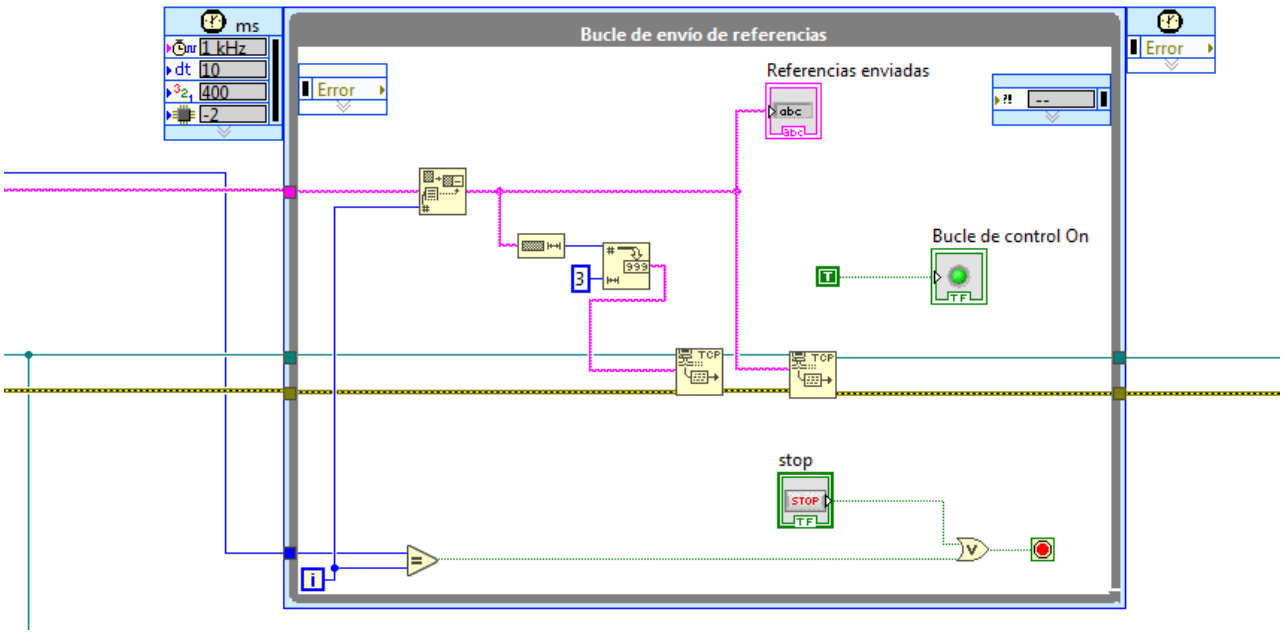


Figura 90: Envío de referencias completas de articulaciones o cartesianas

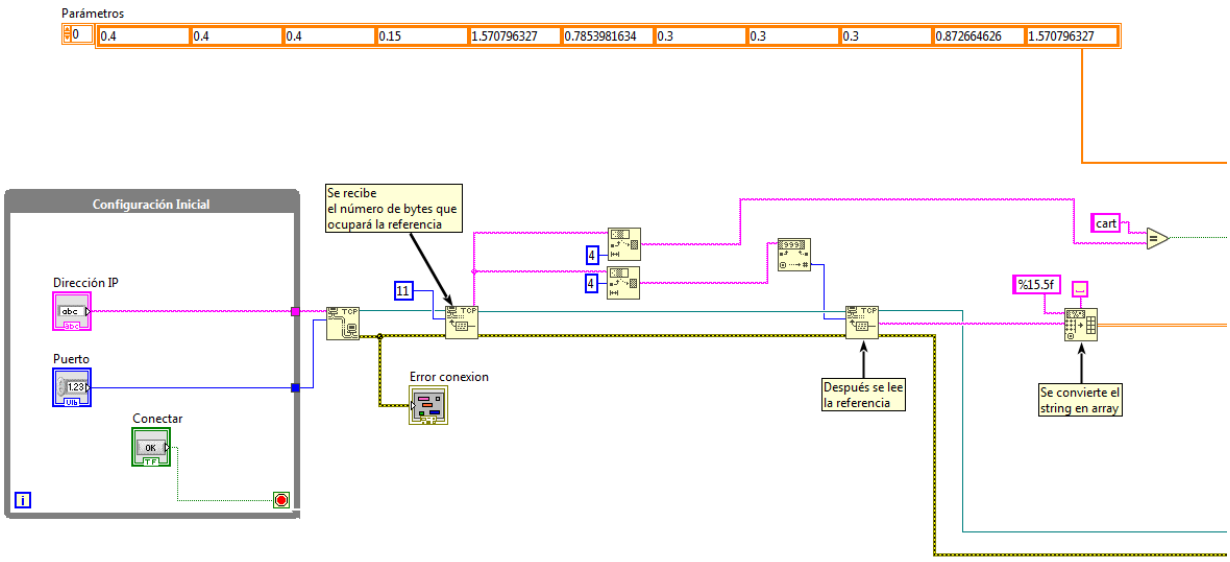


Figura 91: Controlador, recibo de referencias de articulaciones o cartesianas completas, inicio 1

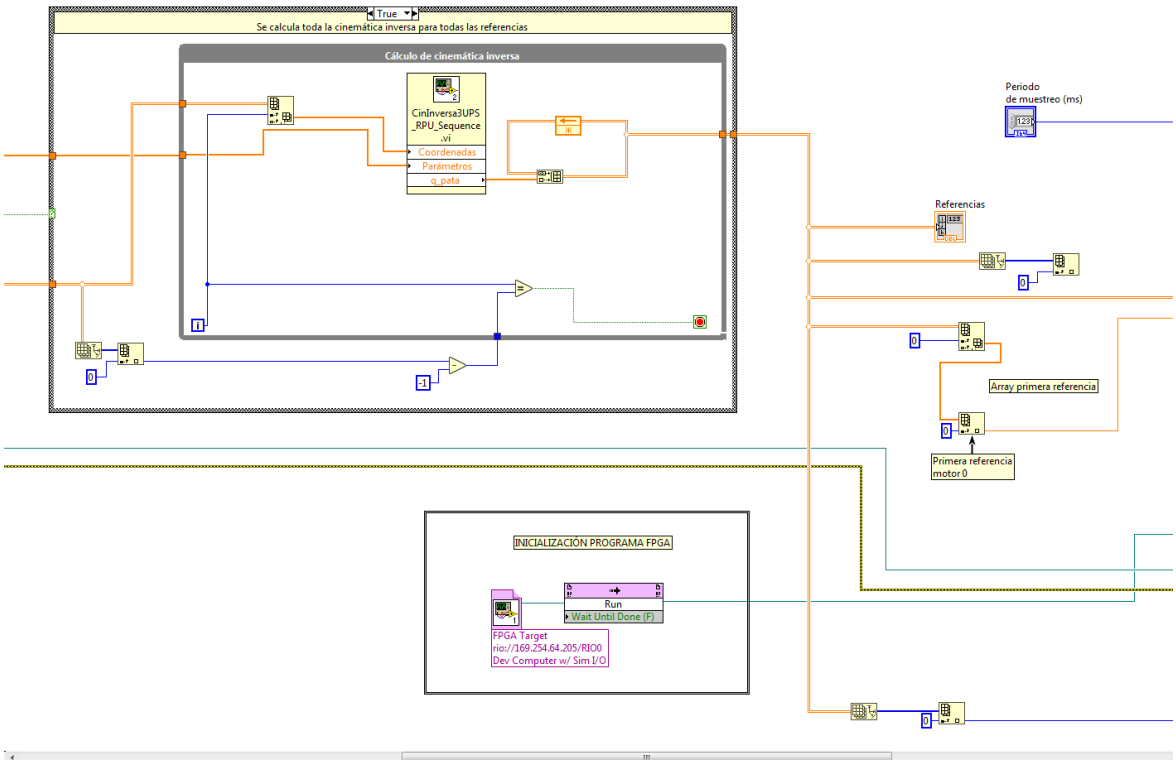


Figura 92: Controlador, recibo de referencias de articulaciones o cartesianas completas, inicio 2

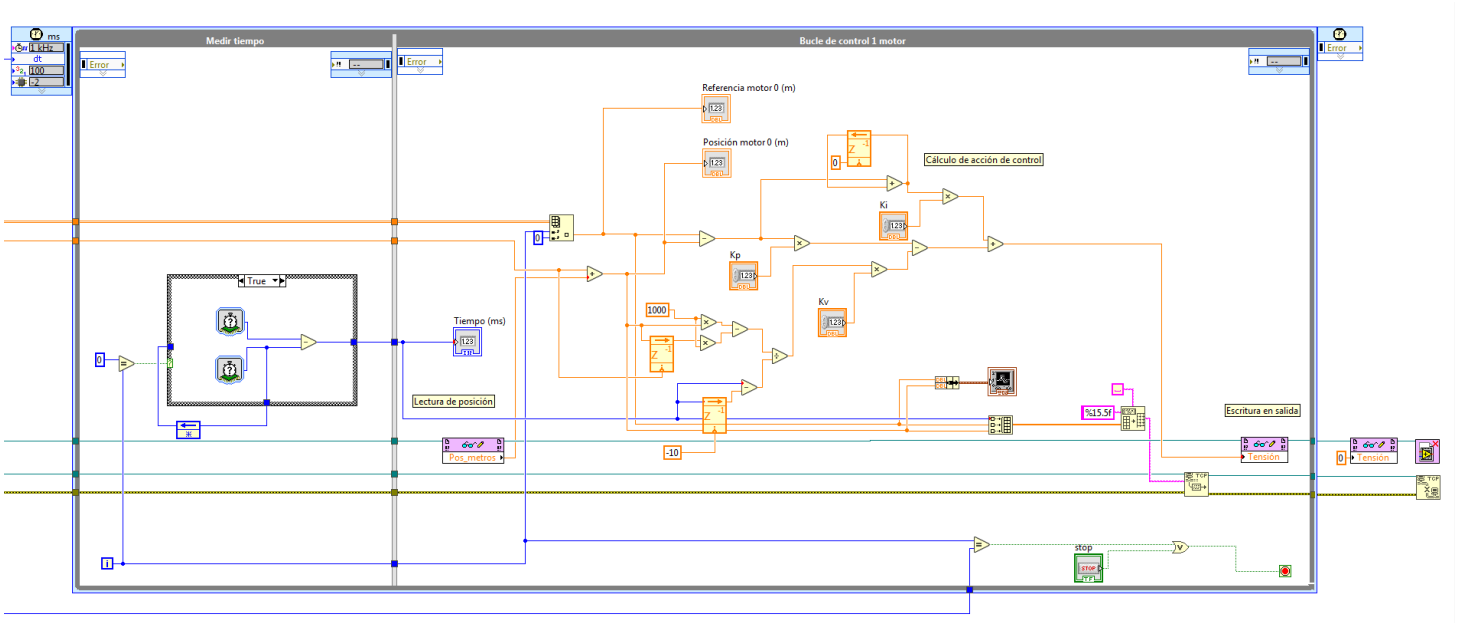


Figura 93: Controlador, recibo de referencias de articulaciones o cartesianas completas, bucle de control

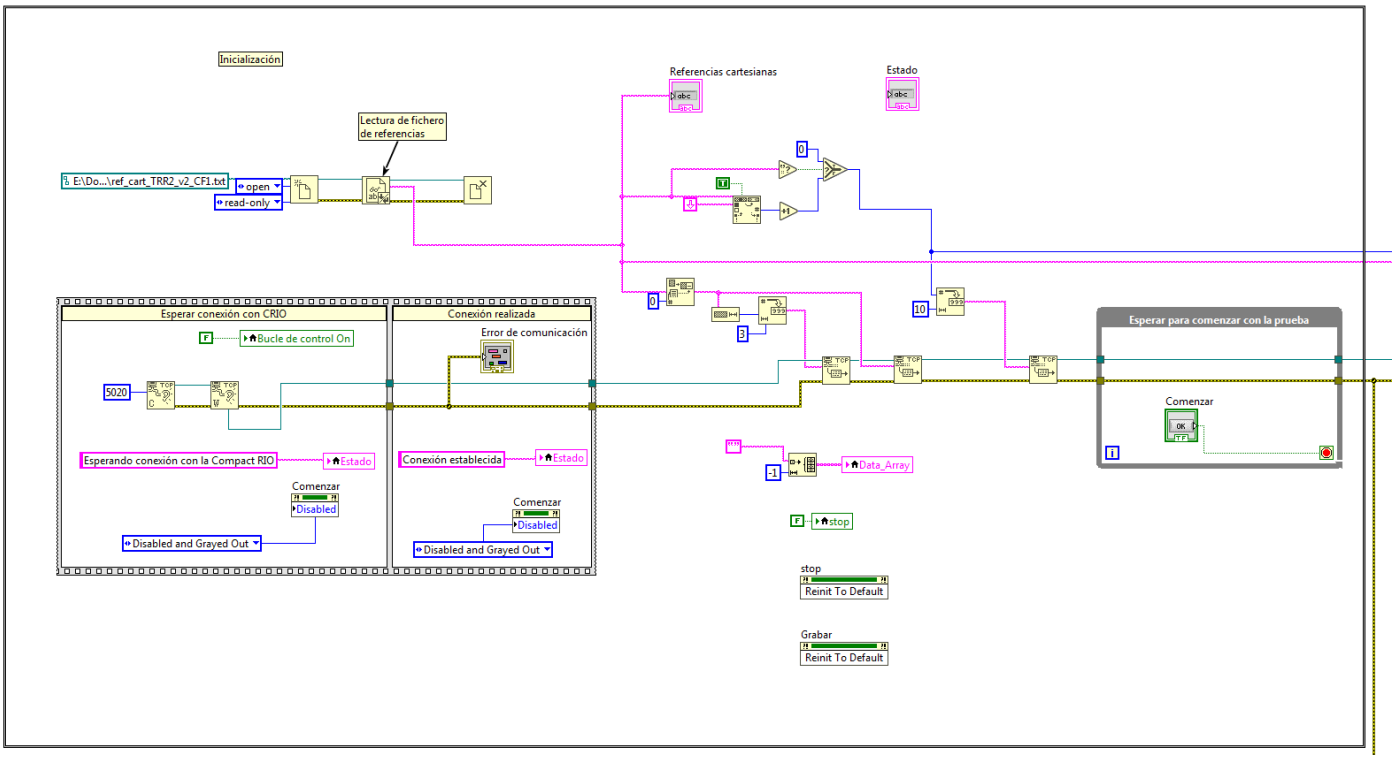


Figura 94: Inicialización del servidor de envío único de referencias cartesianas

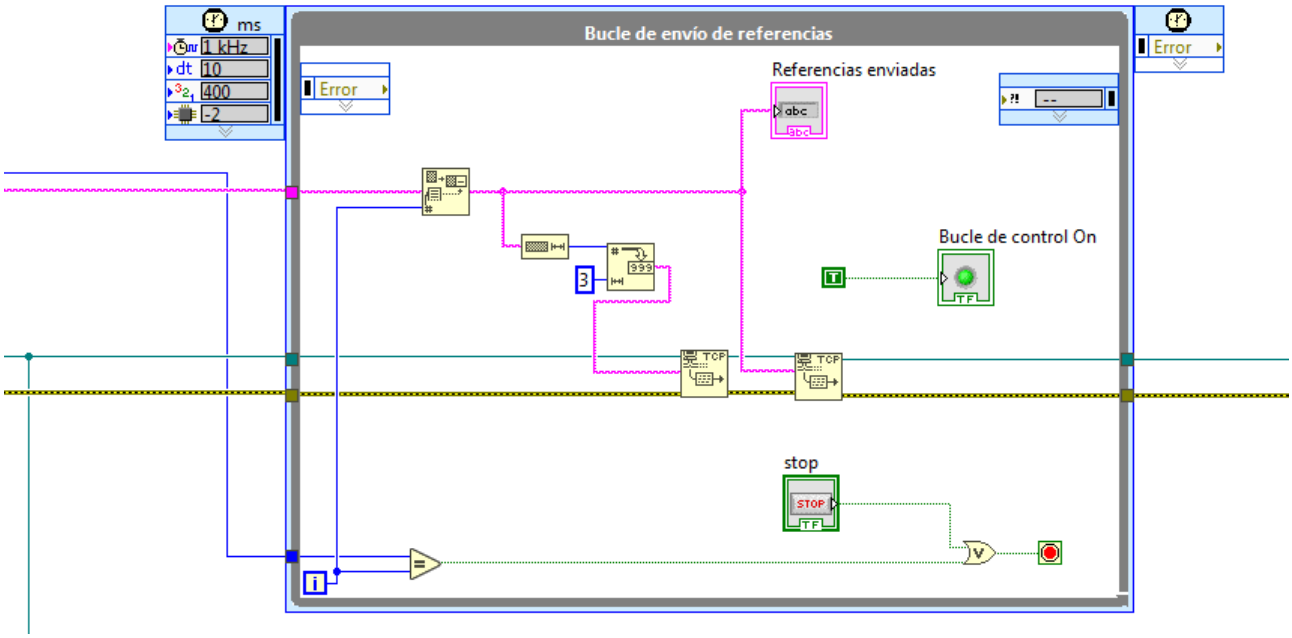


Figura 95: Envío único de referencias cartesianas

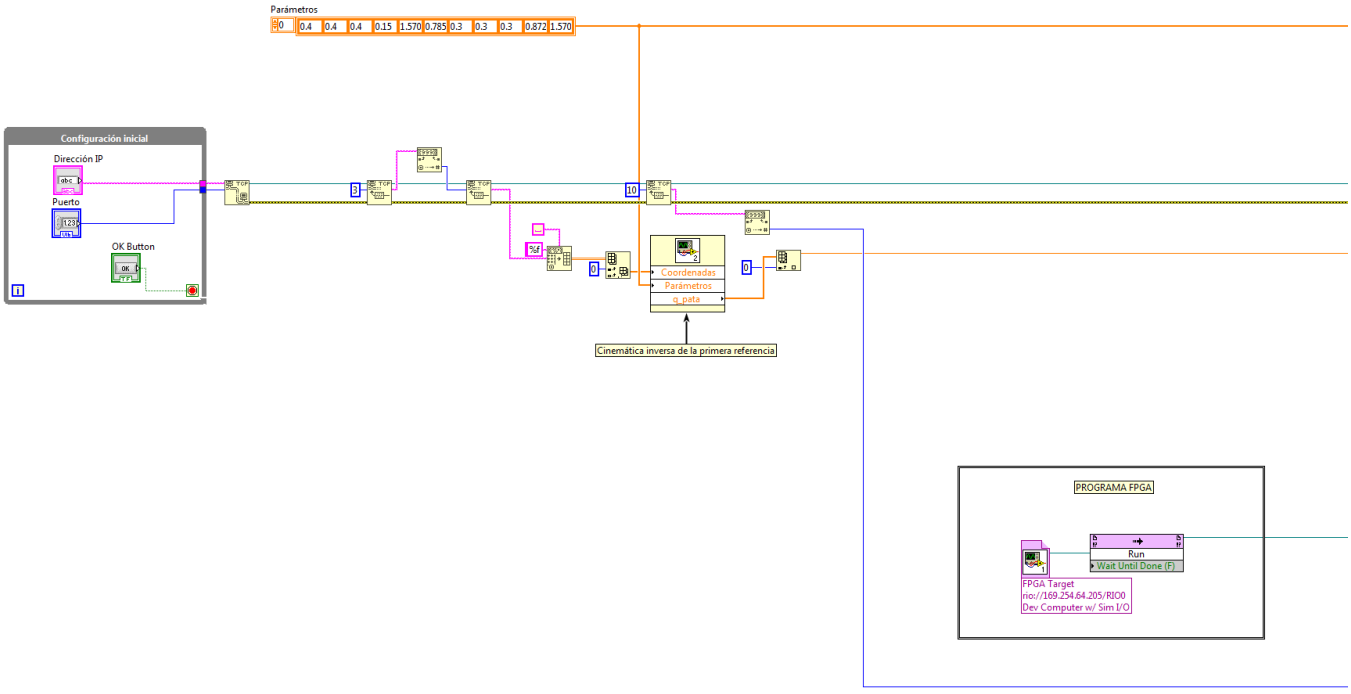


Figura 96: Controlador, recibo único de referencias cartesianas, inicio

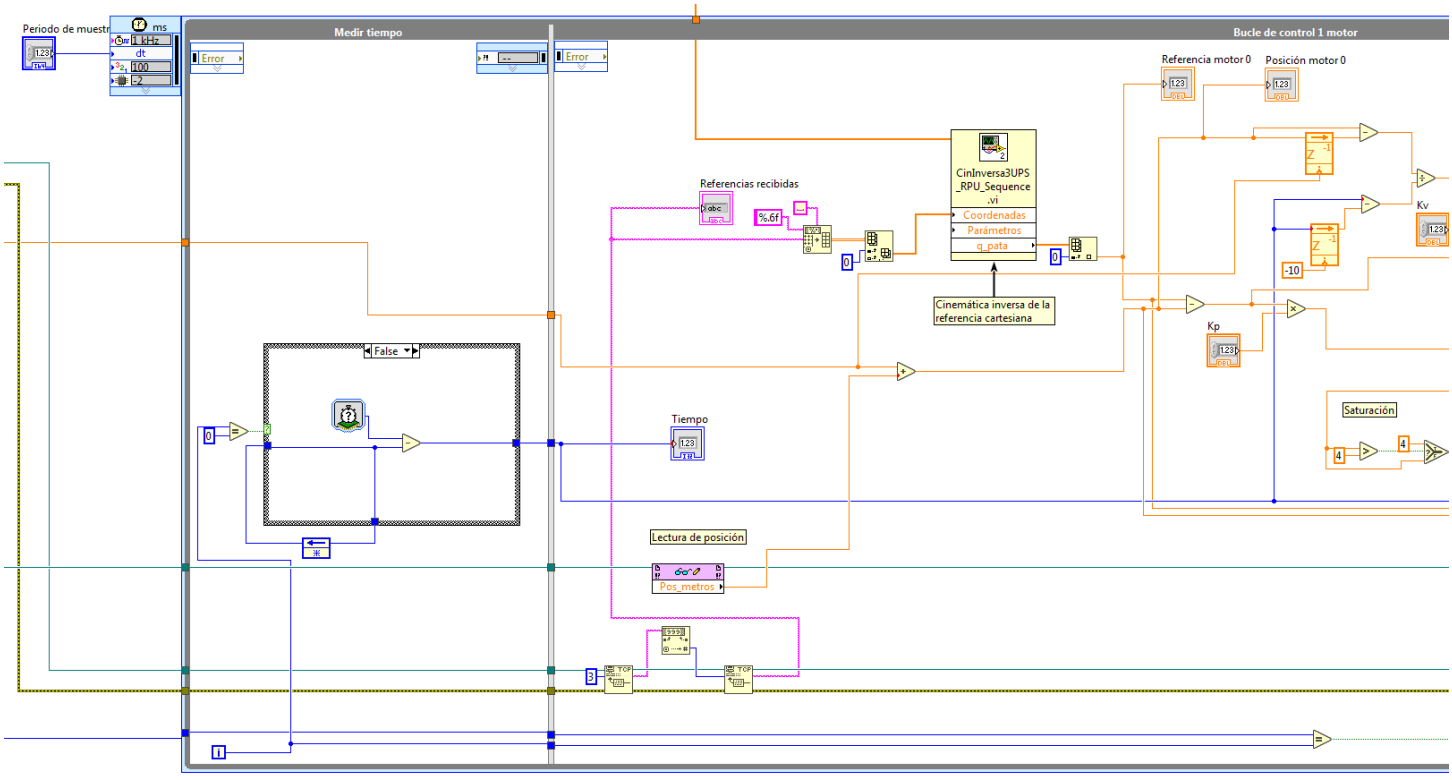


Figura 97: Controlador, recibo único de referencias cartesianas, bucle de control 1

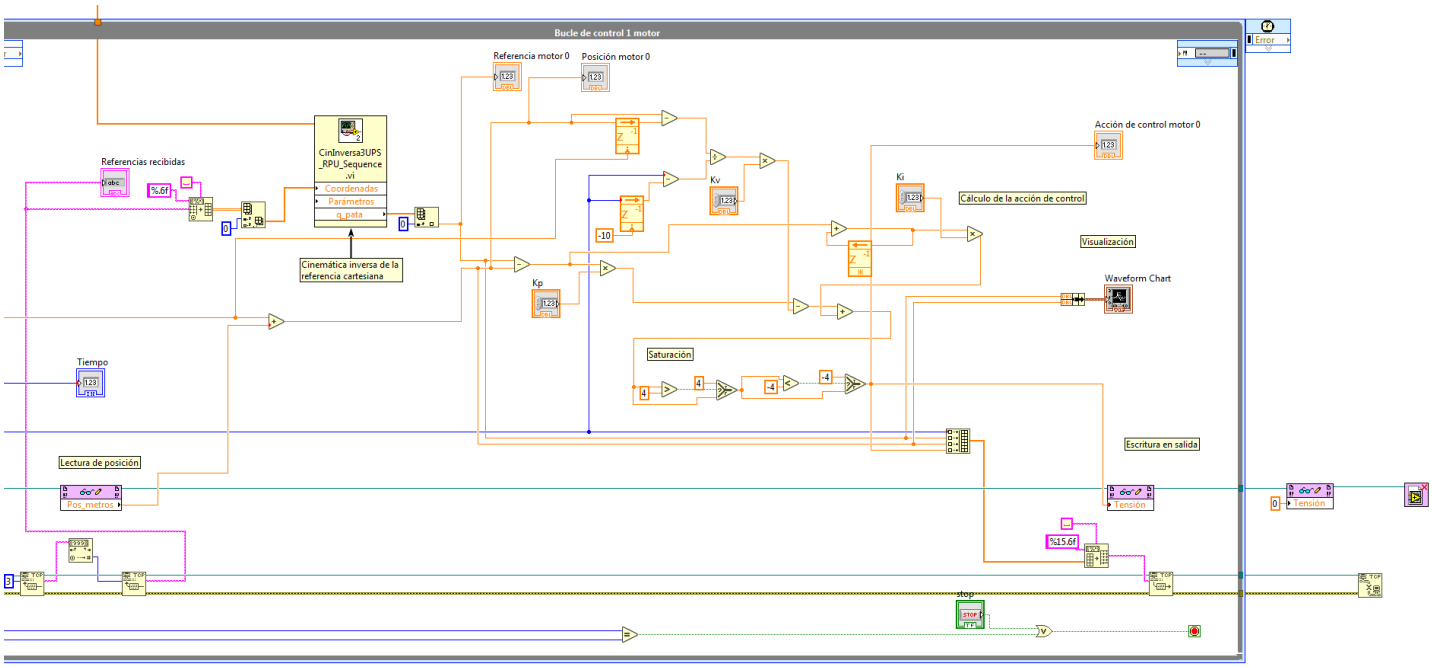


Figura 98: Controlador, recibo único de referencias cartesianas, bucle de control 2

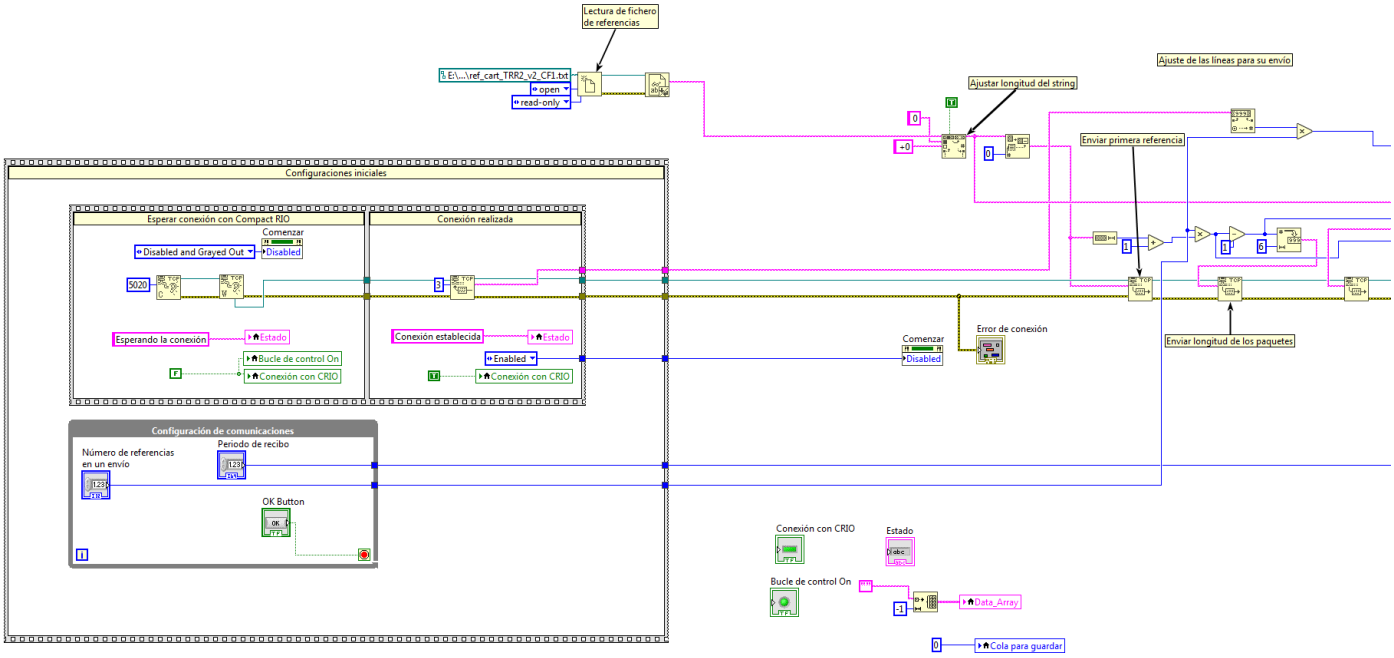


Figura 99: Inicialización servidor envío de múltiples líneas de referencias

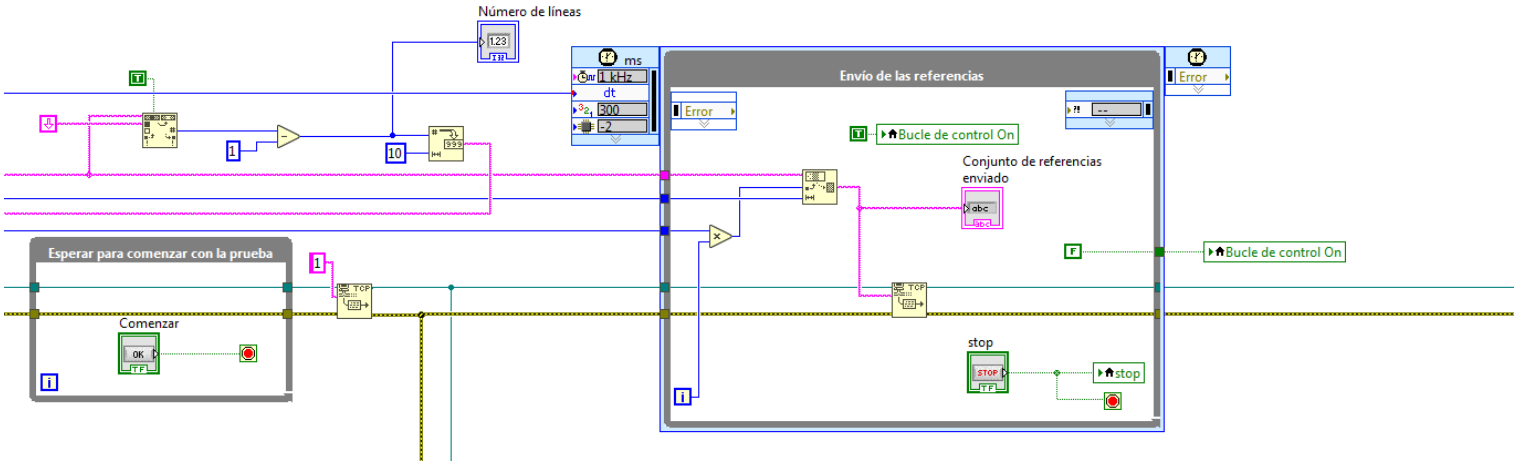


Figura 100: Servidor, envío de múltiples líneas de referencias

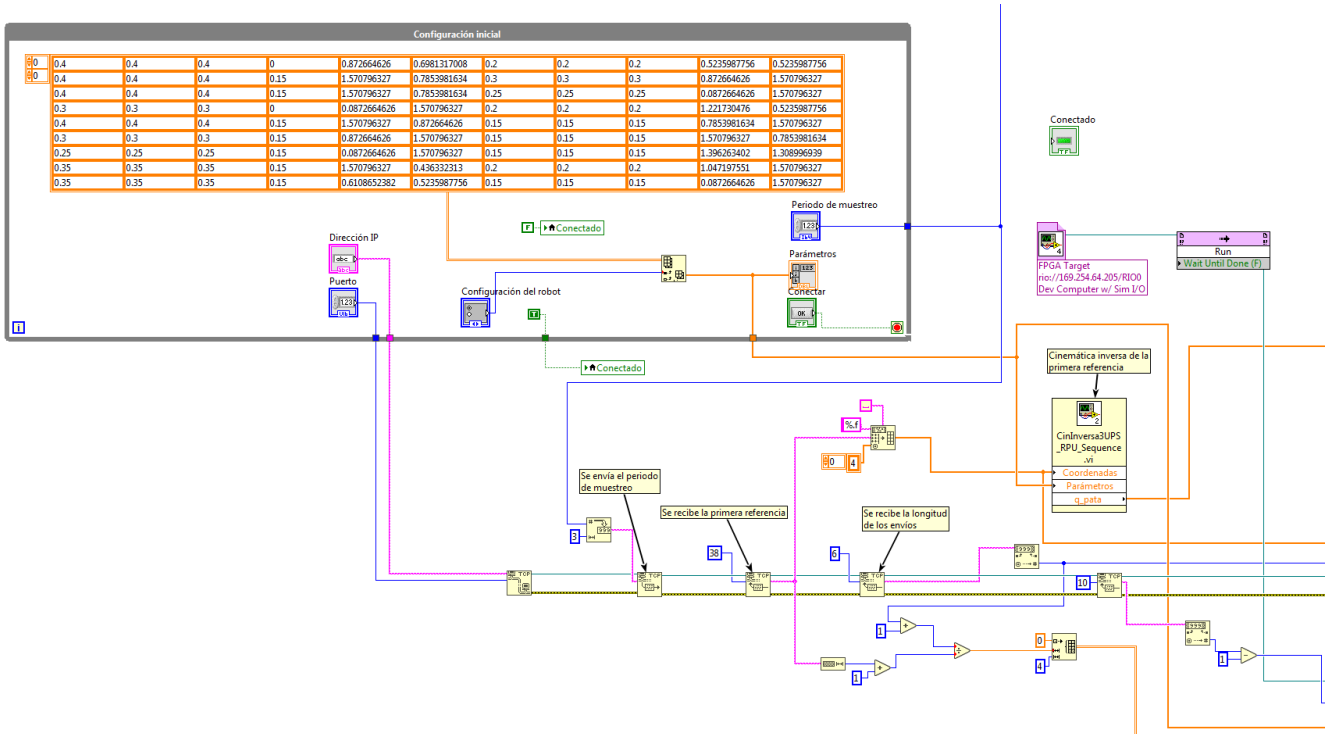


Figura 101: Controlador, múltiples líneas de referencias, inicio

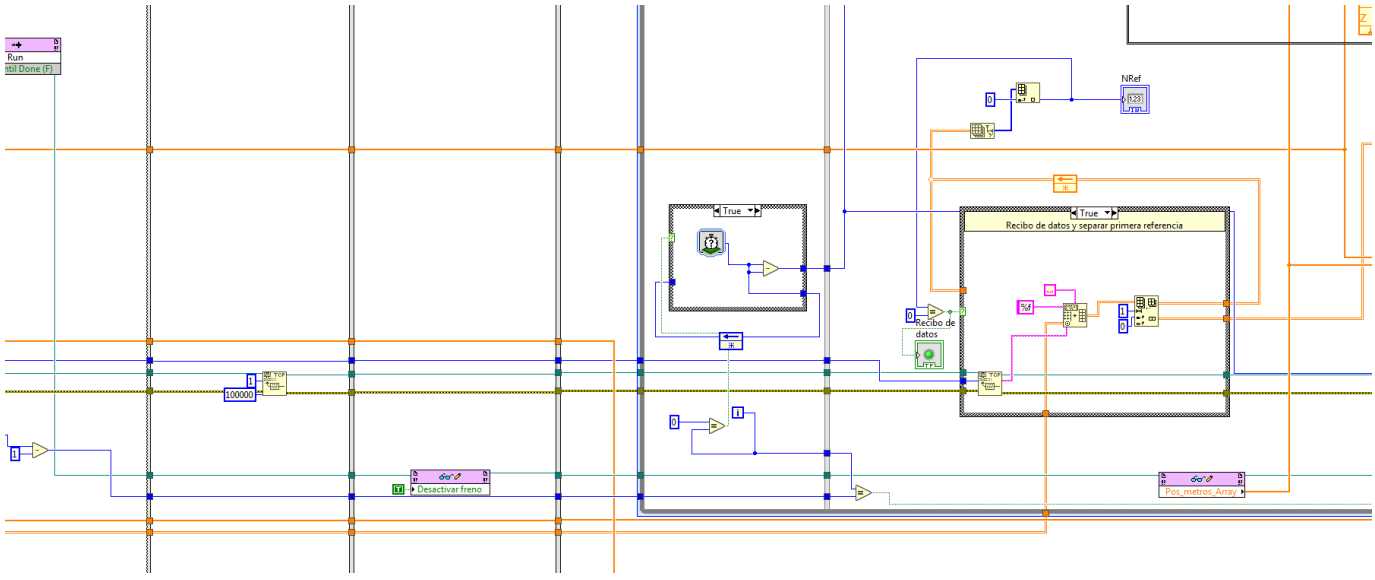


Figura 102: Controlador, múltiples líneas de referenicas, recibo de referencias

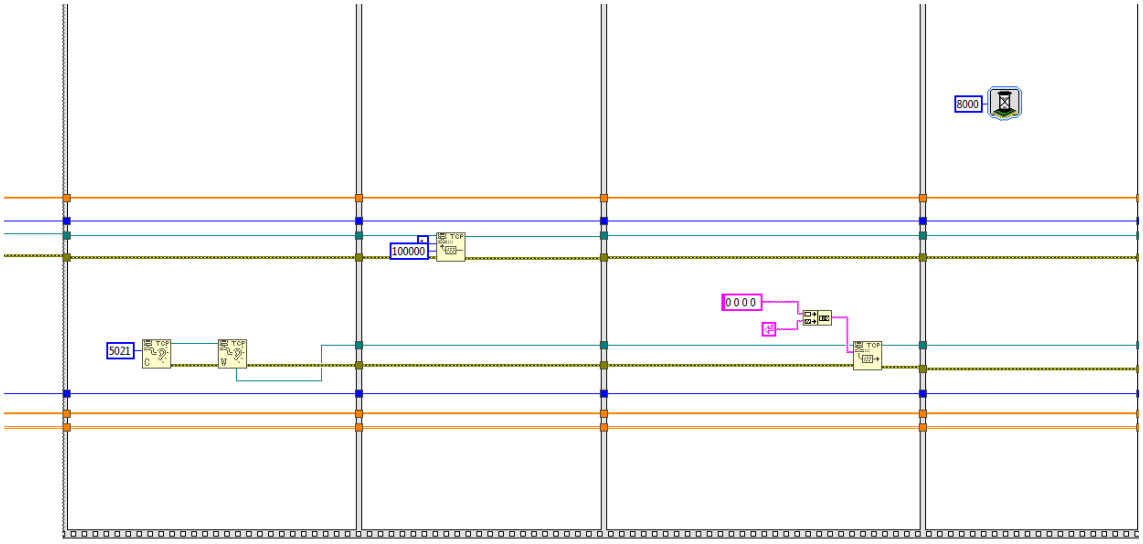


Figura 103: Controlador, múltiples líneas de referencias, conexión con Simulink

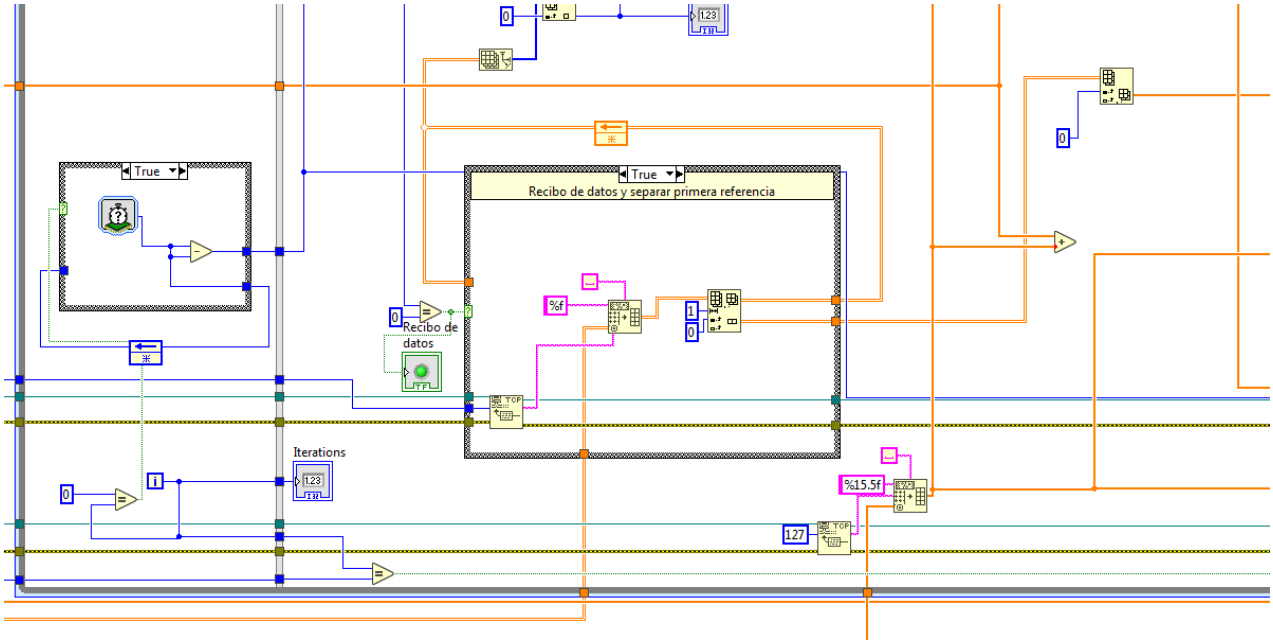


Figura 104: Controlador, múltiples líneas de referencias, recibo de referencias simulado

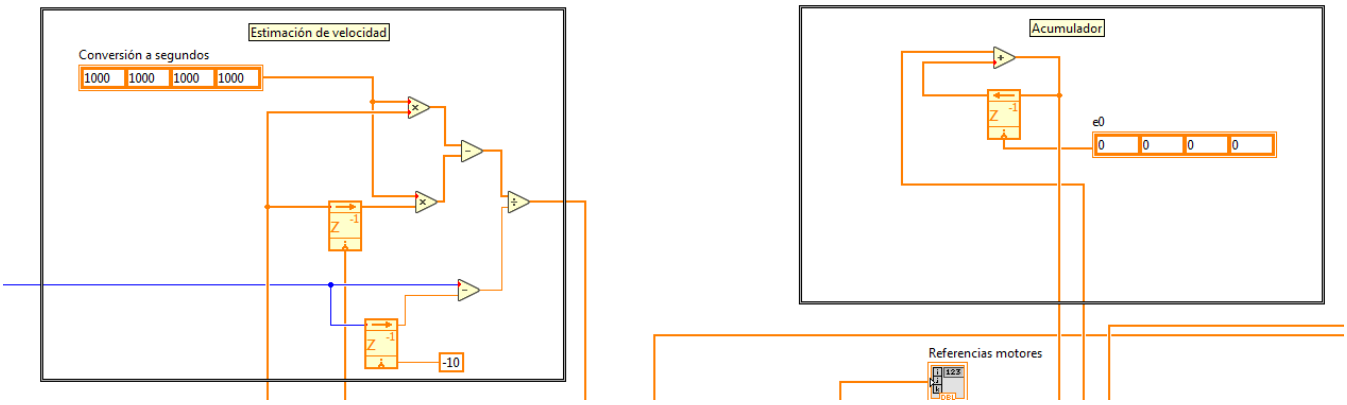


Figura 105: Controlador, múltiples líneas de referencias, estimación de velocidad y acumulador

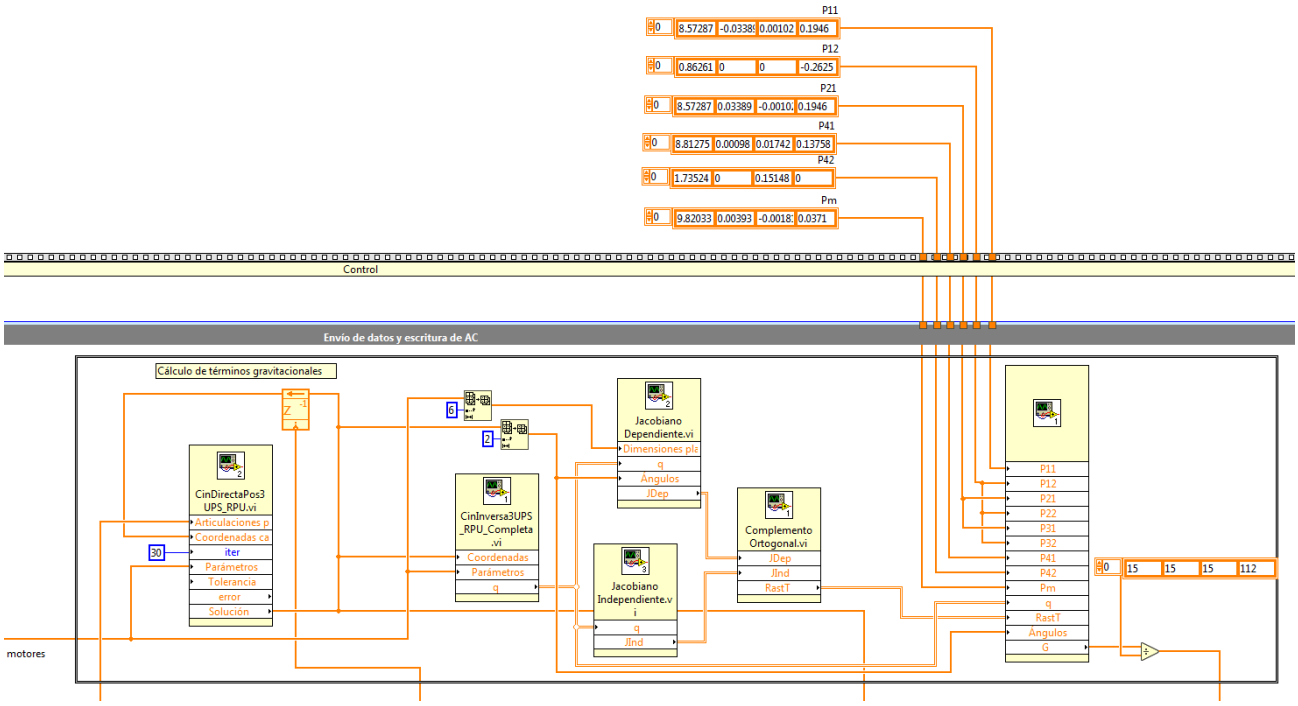


Figura 106: Controlador, múltiples líneas de referencias, cálculo de términos gravitatorios

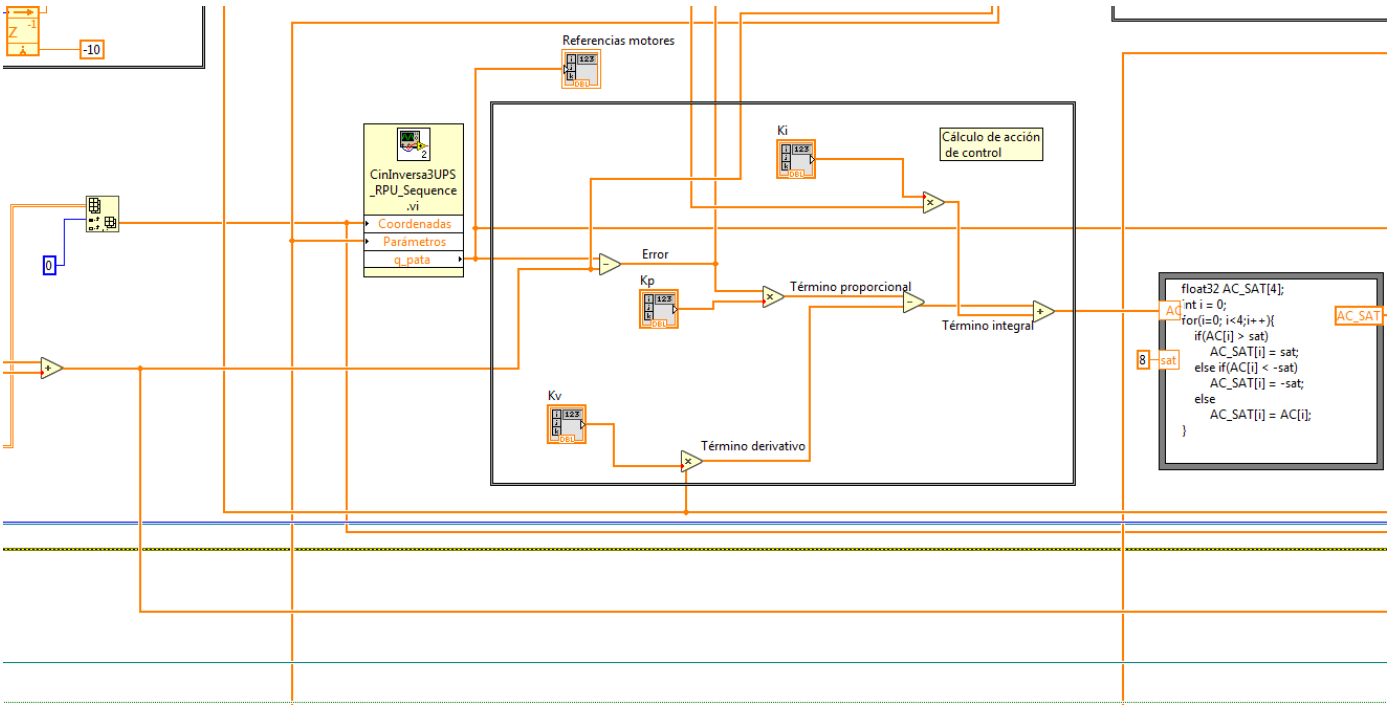


Figura 107: Controlador, múltiples líneas de referencias, cálculo de acción de control

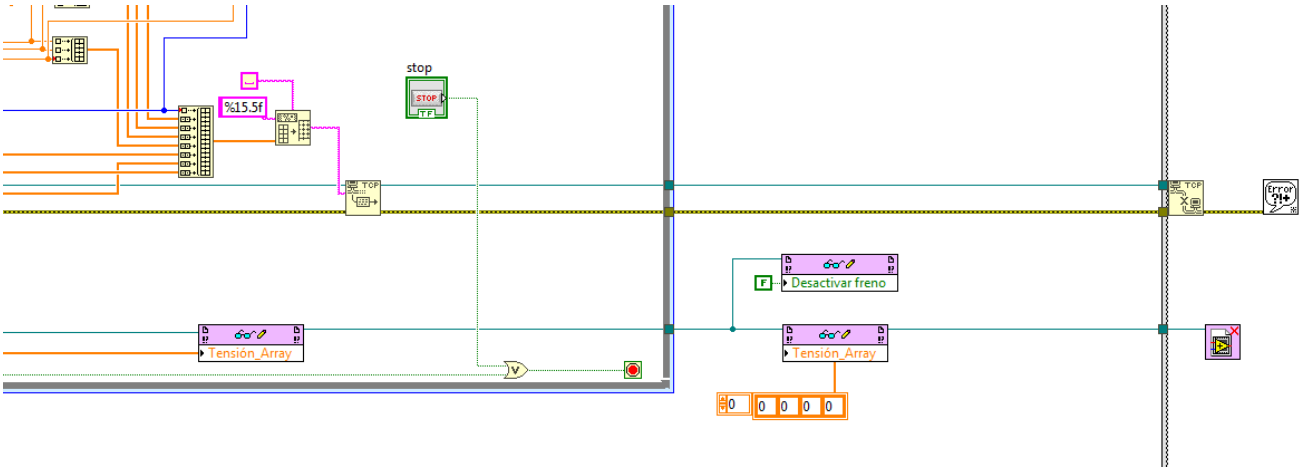


Figura 108: Controlador, múltiples líneas de referencias, escritura acción de control y paro

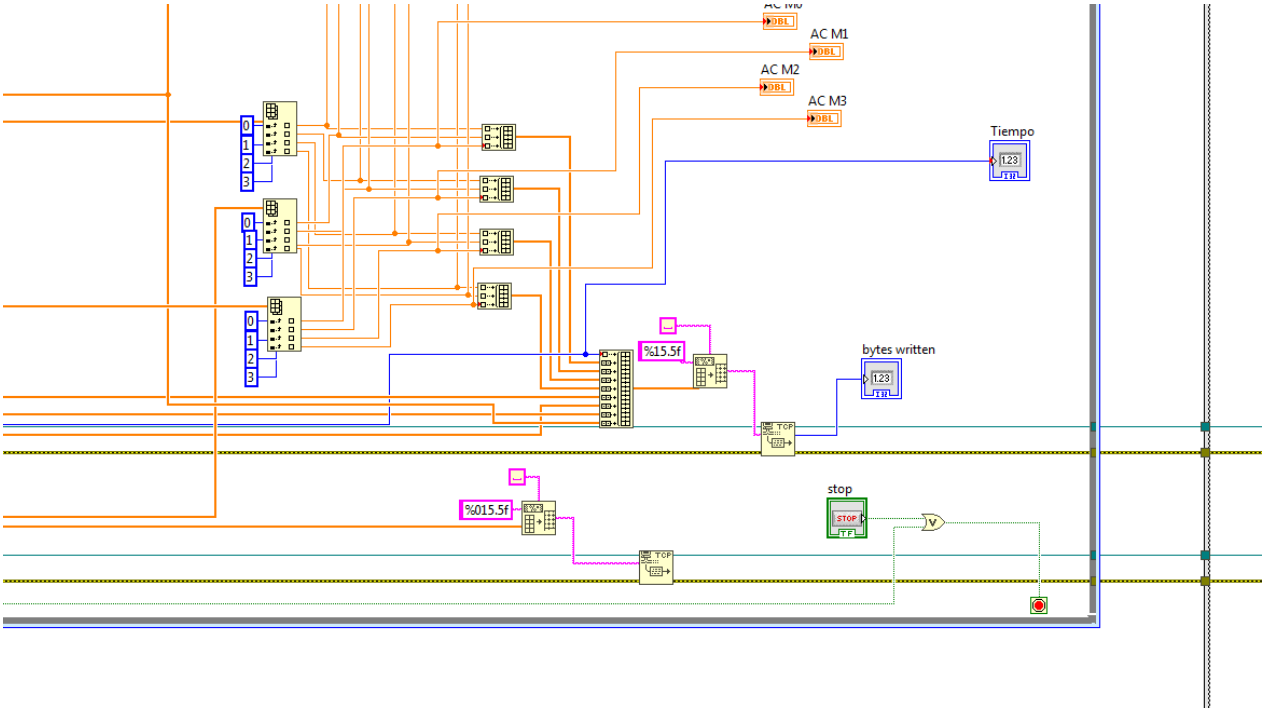


Figura 109: Controlador, múltiples líneas de referencias, envío de acción de control a Simulink

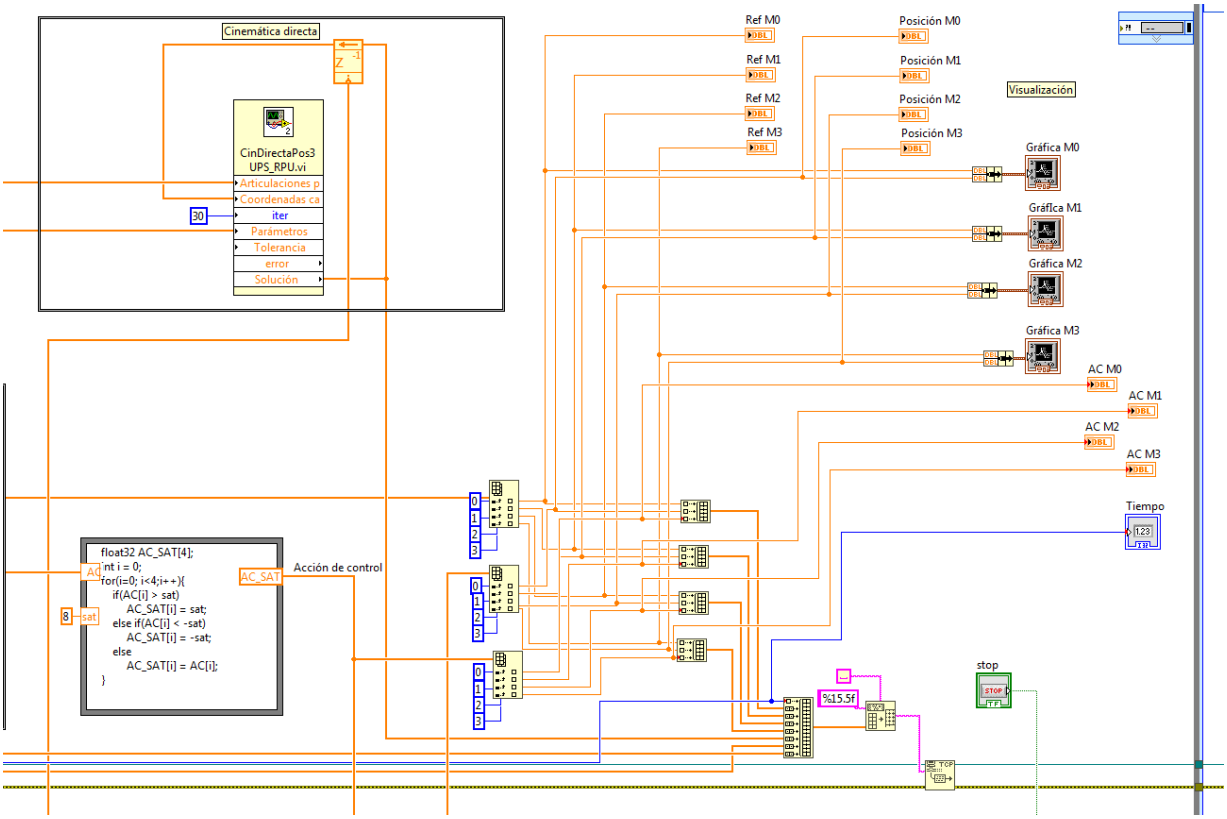


Figura 110: Controlador, múltiples líneas de referencias, visualización

A III. BIBLIOGRAFÍA

[1] Ingeniería de control moderna. Katsuhiko Ogata. Pearson, 2010.

[2] Parallel robots. J.-P. Merlet. Springer, 2006.

[3] Robots paralelos: máquinas con un pasado para una robótica del futuro. Universidad Politécnica de Madrid. Revista iberoamericana de automática e informática industrial, 2006.

[4] Implementación basada en el middleware OROCOS de controladores dinámicos pasivos para un robot paralelo. Universitat Politècnica de València. ScienceDirect, 2013.

[5] Sistemas de Tiempo Real y Lenguajes de Programación. Alan Burns, Andy Wellings. Addison Wesley, 2002.

[6] Diapositivas de la asignatura Dispositivos lógicos programables. García Berijo Eduardo. Universitat Politècnica de València.

[7] https://imbio3r.ai2.upv.es/galeria_de_fotos/control-architecture

[8] Sistemas de Tiempo Real Distribuidos Robots y Microcontroladores. Instituto de Investigación en Informática, Universidad Nacional de La Plata.

[9] Data brief NUCLEO-XXXXZX NUCLEO-XXXXZX-P. STMicroelectronics. https://www.st.com/resource/en/data_brief/nucleo-f439zi.pdf

[10] Información general de familia STM32 de 32 bits. https://www.digikey.es/es/product-highlight/s/stmicroelectronics/stm32-overview

[11] STM32F7 Series. https://www.st.com/en/microcontrollers-microprocessors/stm32f7-series.html

[12] User manual and specifications NI cRIO-9002/9004. National Instruments. http://www.ni.com/pdf/manuals/373561e.pdf

[13] User manual and specifications NI cRIO-9101/9102/9103/9104. National Instruments.

[14] Datasheet NI 9201. National Instruments. http://www.ni.com/pdf/manuals/373783a_02.pdf

[15] Datasheet NI 9263. National Instruments. http://www.ni.com/pdf/manuals/373781b_02.pdf

[16] Datasheet NI 9421. National Instruments. https://www.ni.com/pdf/manuals/373504a_02.pdf

[17] Datasheet NI 9474. National Instruments. https://www.ni.com/pdf/manuals/373974c_02.pdf

[18] Datasheet MIC2981/2982. MICREL. https://www.microchip.com/wwwproducts/en/MIC2981

[19] LabVIEW: Advanced Programming Techniques. Rick Bitter, Taqi Mohiuddin, Matt Nawrocki. Taylor & Francis Group, 2007.

[20] Learning with LabVIEW 8. Robert H. Bishop. Upper Saddler River: Prentice Hall, 2007.

[21] Diapositivas de la asignatura Instalaciones de control industrial. García-Nieto Rodríguez, Sergio. Universitat Politècnica de València.

2020

Planos: Placa de conexión entre cuadro de control y cRIO

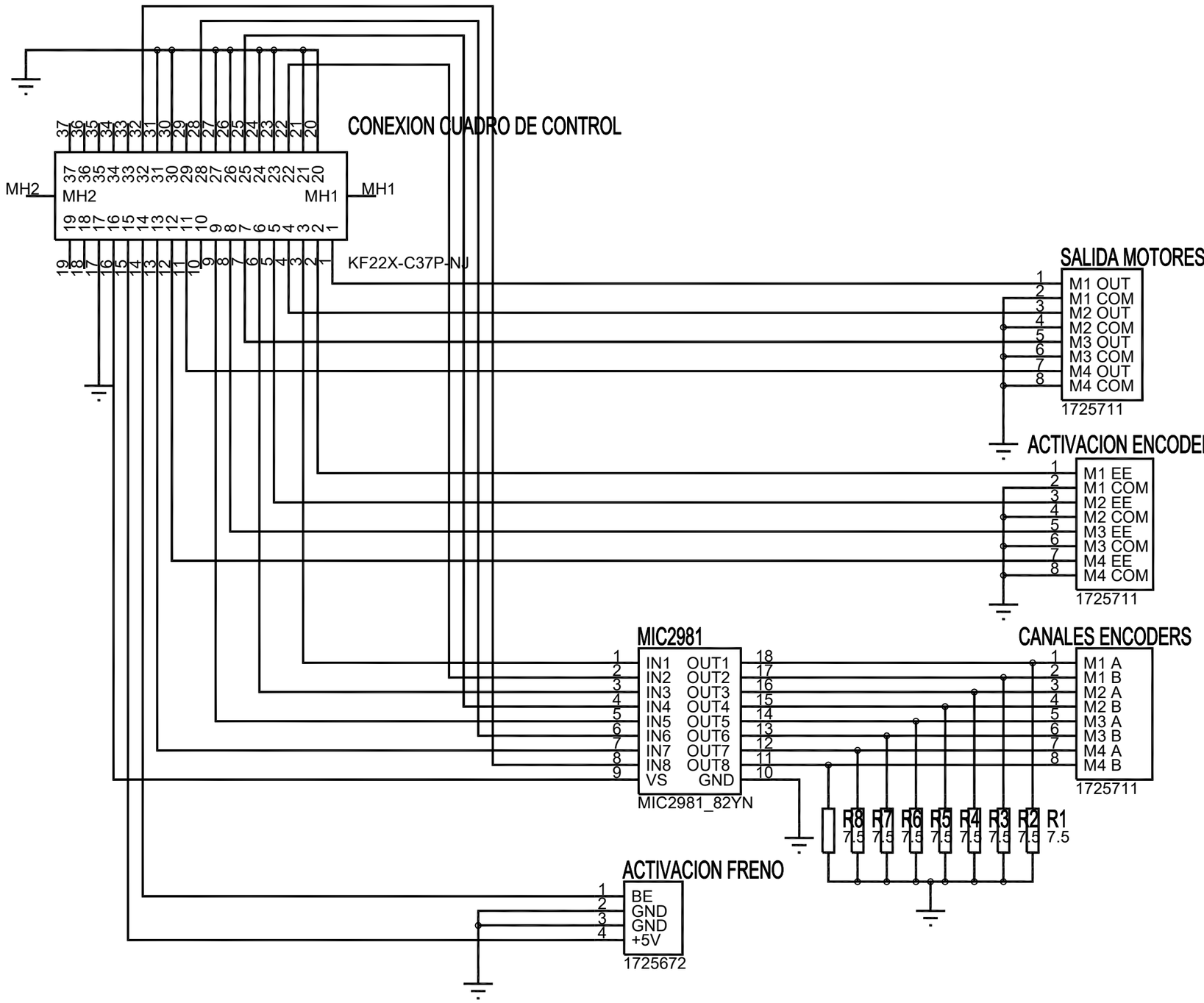
DOCUMENTO 2: PLANOS

AUTOR: FERRÁNDIZ ALARCÓN, JESÚS

TUTORA: MIQUEL VALLÉS, MARINA

COTUTOR: PULLOQUINGA ZAPATA, JOSE LUIS

TABLA DE CONTENIDO



NI-2963 SLOT 3	SALIDA MOTORES
AO0	M1 OUT
COM	M1 COM
AO1	M2 OUT
COM	M2 COM
AO2	M3 OUT
COM	M3 COM
AO3	M4 OUT
COM	M4 COM

NI-9421 SLOT 2	CANALES ENCODERS
DI0	M1 A
DI1	M1 B
DI2	M2 A
DI3	M2 B
DI4	M3 A
DI5	M3 B
DI6	M4 A
DI7	M4 B
COM	GND

NI-9474 SLOT 1	ACTIVACIÓN FRENO Y ENCODERS
DO0	M1 EE
DO1	M2 EE
DO2	M3 EE
DO3	M4 EE
DO4	BE
DO5	-
DO6	-
DO7	-
Vsup	+5 V
COM	GND

TRABAJO FINAL DE GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSID

Proyecto: PLACA DE CONEXIÓN ENTRE EL CUADRO DE CONTROL Y CRIO

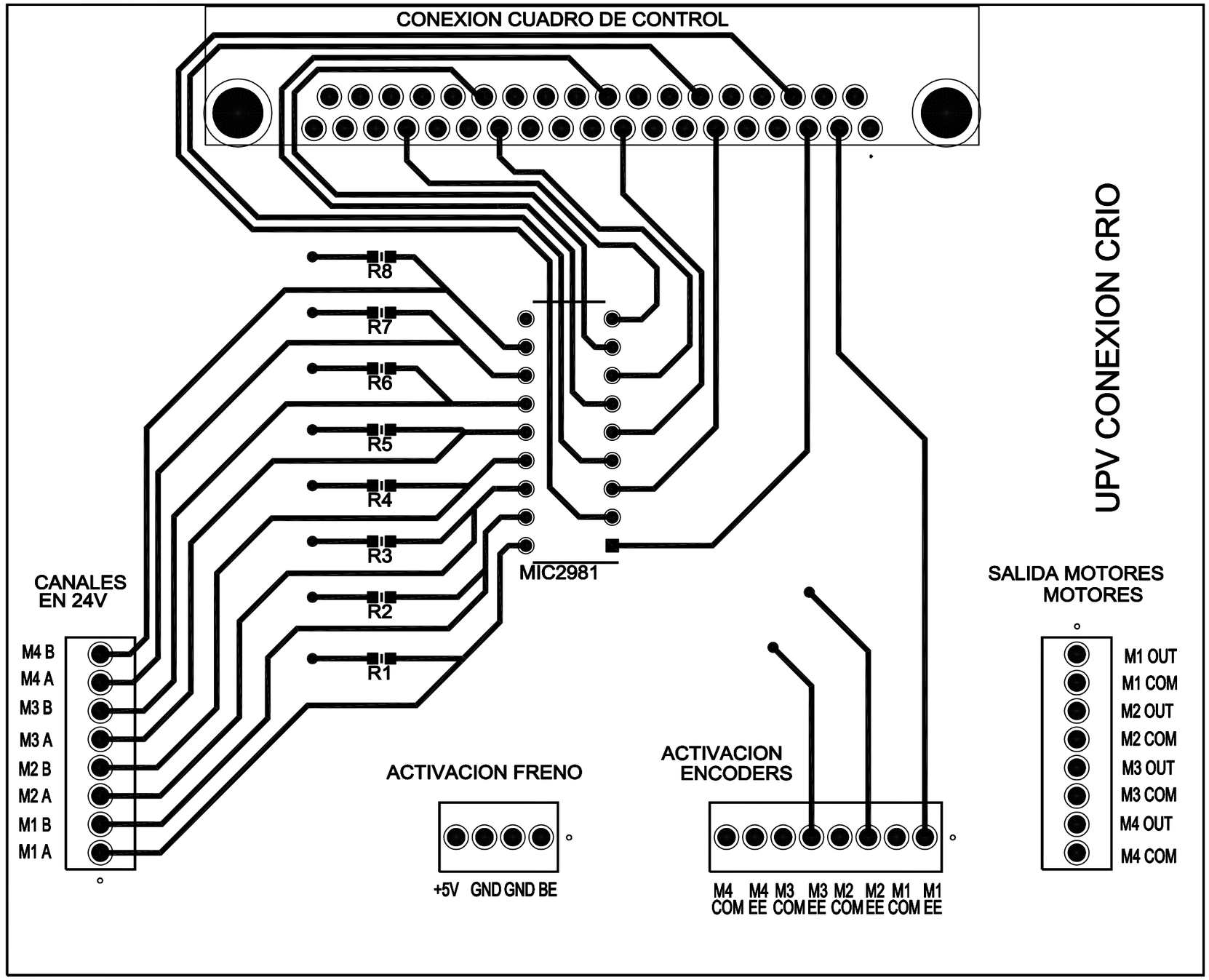
Plano: Esquema eléctrico y de conexiones

Fecha: Junio 2020

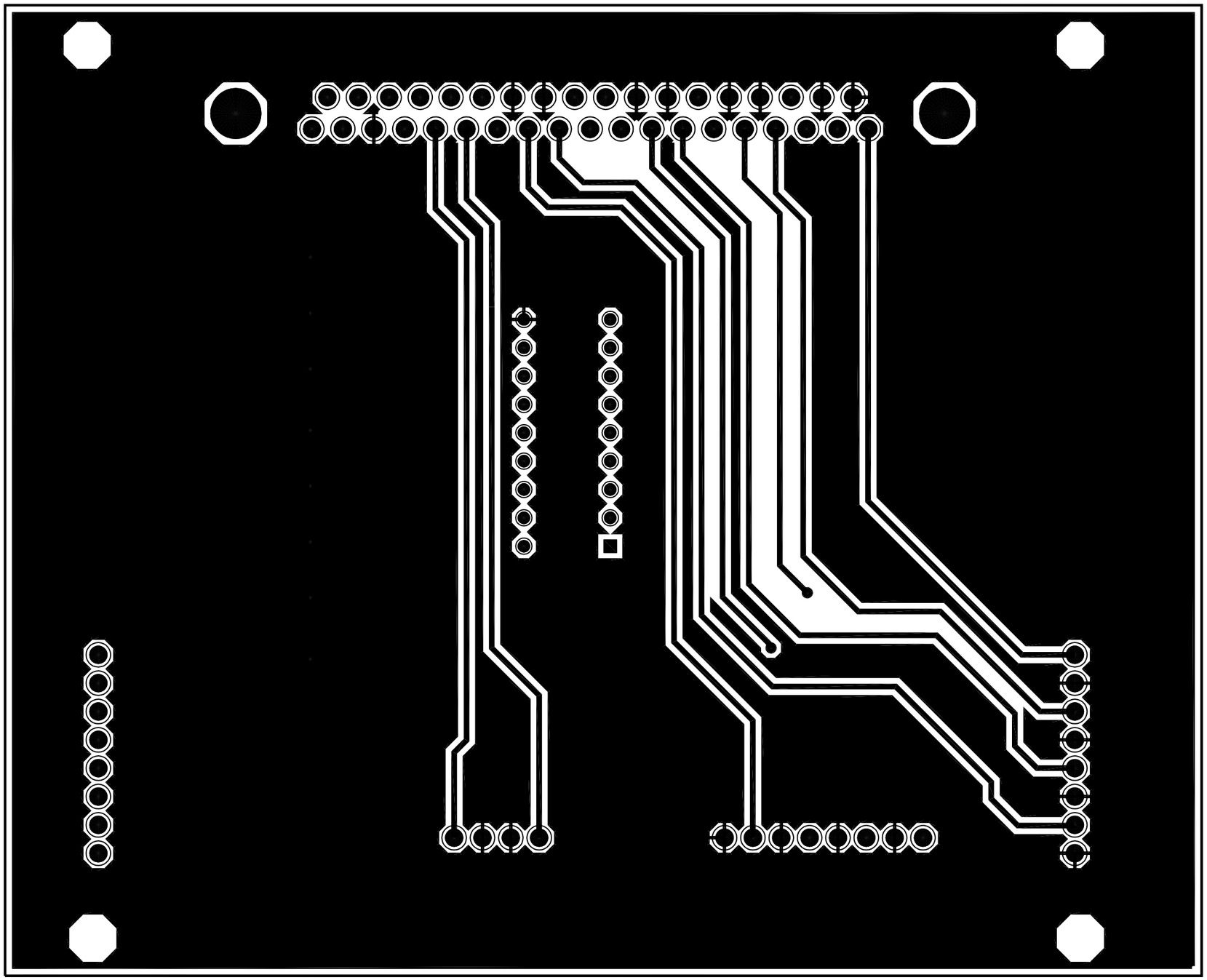
Nº Plano: 01

Autor: Ferrándiz Alarcón, Jesús

Escala: S/N



TRABAJO FINAL DE GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA UNIVERSITAT POLITÈCNICA DE VALÈNCIA ETSID	Proyecto: PLACA DE CONEXIÓN ENTRE EL CUADRO DE CONTROL Y CRIO	Plano: Vista de las capas superiores	Fecha: Junio 2020	Nº Plano: 02
		Autor: Ferrándiz Alarcón, Jesús	Escala: 2.5:1	



TRABAJO FINAL DE GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSID

Proyecto: PLACA DE CONEXIÓN ENTRE EL CUADRO DE CONTROL Y CRIO

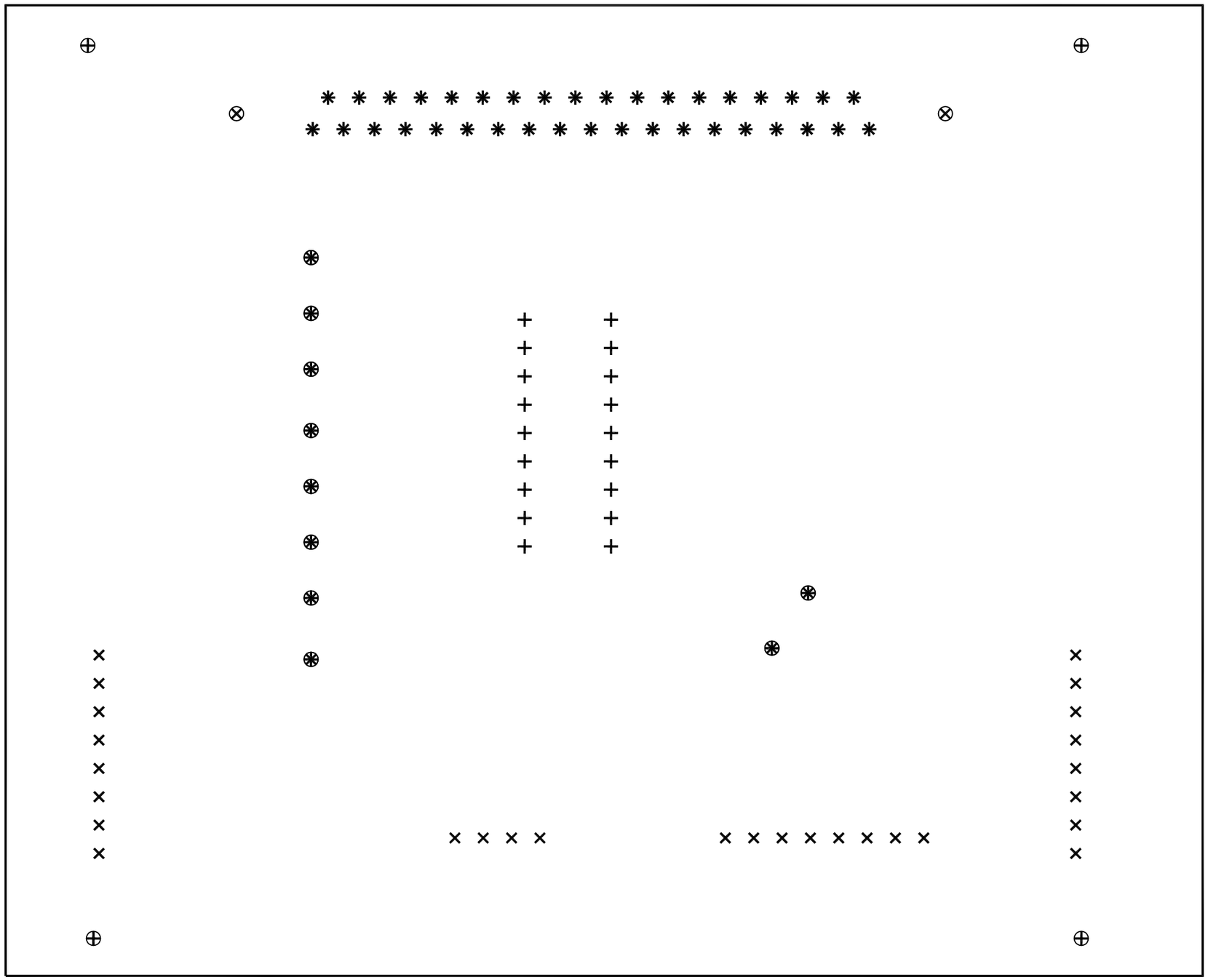
Plano: Vista de los trazados inferiores

Autor: Ferrándiz Alarcón, Jesús

Fecha: Junio 2020

Escala: 2.5:1

Nº Plano: 03



SYM	SIZE	PLATED	QTY
+	0.73mm	YES	18
×	1.1mm	YES	28
✱	1.09mm	YES	37
⊕	3.2mm	YES	4
⊗	3.05mm	YES	2
❋	15th	YES	10

TRABAJO FINAL DE GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSID

Proyecto: PLACA DE CONEXIÓN ENTRE EL CUADRO DE CONTROL Y CRIO

Plano: Vista de las perforaciones

Autor: Ferrándiz Alarcón, Jesús

Fecha: Junio 2020

Escala: 2.5:1

Nº Plano: 04

2020

Pliego de condiciones

DOCUMENTO 3

AUTOR: FERRÁNDIZ ALARCÓN, JESÚS
TUTORA: VALLÉS MIQUEL, MARINA
COTUTOR: PULLOQUINGA ZAPATA, JOSE LUIS

TABLA DE CONTENIDO

1. OBJETO

La presente especificación técnica se refiere al control de el robot paralelo de 4 grados de libertad del Instituto de automática e informática industrial de la Universidad Politécnica de Valencia, ubicado en el laboratorio de robótica del Departamento de ingeniería mecánica y de materiales.

Quedan excluidos los trabajos relacionados con los elementos mecánicos del robot, así como de las conexiones entre este y el cuadro de control. Esta especificación se centra en la unidad de control.

2. CONDICIONES DE LOS MATERIALES

2.1. PLACA DE CIRCUITO IMPRESO

En este apartado se explican todas las condiciones necesarias para el desarrollo de la placa de circuito impreso que conecta las señales procedentes del controlador con el cuadro de control.

2.1.1. PLACA

La placa será de cobre y baquelita, de una capa y de grosor 3 mm. El diseño tendrá que seguir lo especificado en los planos 2,3 y 4.

2.1.2. AMPLIFICADOR

Debe utilizarse el circuito integrado MIC2981.

2.1.3. RESISTENCIAS

Se utilizarán ocho resistencias SMD de 7,5 kΩ de una potencia mínima de 0,08 W. Antes de utilizarlas, se comprobará mediante un ohmímetro que el valor de resistencia es el correcto y se encuentra en el rango de la tolerancia empleada.

2.1.4. CONECTOR DE ENTRADA

Deberá emplearse el conector estándar del tipo D-sub con la siguiente referencia: KF22X-C37P-NJ.

2.1.5. TERMINALES DE SALIDA

Se tienen que emplear tres regletas de 8 contactos hembra de agujeros pasantes cuyo paso entre ellos sea de 2,54 mm. Se debe emplear otra regleta con las mismas características, pero en este caso de 4 contactos.

2.2. UNIDAD DE CONTROL

2.2.1. PROCESADOR

Se utilizará el controlador cRIO-9004 de la marca National Instruments.

2.2.2. CHASIS

El chasis a utilizar será el cRIO-9104 de la marca National Instruments.

2.2.3. MÓDULOS DE ENTRADA Y DE SALIDA

Se deben emplear los siguientes módulos de entradas y salidas, todos de la marca National Instruments:

- Dos módulos de salida de voltaje de la Serie C NI-9263.
- Módulo digital de la Serie C NI-9474.
- Módulo digital de la Serie C NI-9421.
- Dos módulos de entrada analógica de la Serie C NI-9201.

2.2.4. ORDENADOR

Debe utilizarse un ordenador que cumpla como mínimo con las siguientes características:

- 4 GB de RAM.
- 100 GB de disco duro.
- Core i3.
- Windows 7

2.2.5. CABLEADO

Para conectar el controlador CRIO con el conmutador Ethernet debe emplearse un conector RJ-45.

2.3. SOFTWARE

En el ordenador se debe tener instalado el siguiente software:

- LabVIEW 2016 de 32 bits.
- Módulo NI-RIO de 2016 y 32 bits.
- Módulo LabVIEW Real-Time de 2016 y 32 bits.
- Módulo LabVIEW FPGA de 2016 y 32 bits.

3. CONDICIONES DE LA EJECUCIÓN

3.1. PLACA DE CIRCUITO IMPRESO

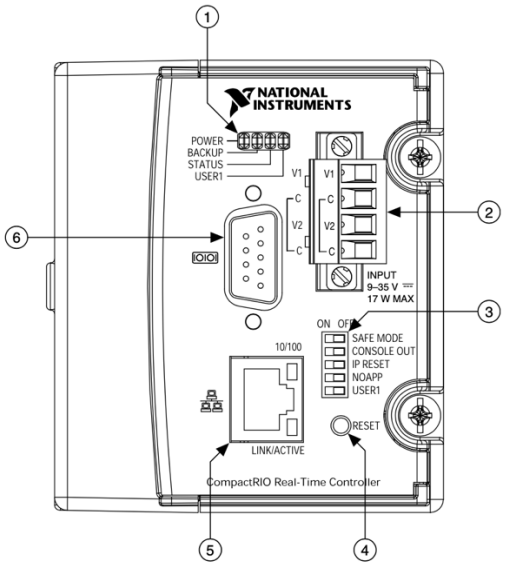
La placa se mandará fabricar al servicio de electrónica del Instituto de Automática e Informática Industrial de la Universitat Politècnica de València y se llevará a cabo la soldadura de los componentes situándolos en las posiciones que se indican en el plano 2.

3.2. UNIDAD DE CONTROL

3.2.1. CONEXIONADO

Las conexiones entre los módulos del controlador CRIO y la placa de conexión se realizarán conformes al plano 1. En este plano también se indica que el módulo NI-9474 debe conectarse en el slot 1 de la Compact RIO, el NI-9421 en el 2 y el NI-9263 en el 3.

Se realizará la conexión entre el controlador y el conmutador Ethernet conectando un terminal del conector RJ-45 en el puerto Ethernet número 5 de la figura 1 y el segundo terminal en un puerto Ethernet del conmutador.



1. LEDs	4. Reset Button
2. Power Connector	5. RJ-45 Ethernet Port
3. DIP Switches	6. RS-232 Serial Port

Figura 1: Controlador crio-9004 conexiones. fuente: manual de usuario ni crio-9002/9004

4. PRUEBAS Y AJUSTE FINALES

4.1. PRUEBA HIL

Antes de realizar experimentos con el robot, se desarrollará una prueba con el controlador a modo de Hardware in the loop. Se utilizará el programa del servidor que envía múltiples referencias ejecutado en el ordenador, y en este caso, el programa del controlador se ejecutará en la Compact RIO. Se empleará el diagrama de Simulink del robot y las trayectorias utilizadas en las simulaciones de la memoria.

Se monitorizarán los datos de cada periodo de muestreo y en el caso de que la respuesta sea igual que en las simulaciones de la memoria, la prueba es correcta.

4.2. PRUEBA CON EL ROBOT

Una vez se haya realizado con éxito la prueba anterior, se realizará la misma con el robot real. Se empleará la misma trayectoria y se cambiará el programa de la Compact RIO por la del control real. Se monitorizarán los datos de nuevo de cada periodo de muestreo y se compararán las respuestas reales con las simuladas, si son similares, la prueba es correcta.

En el caso de que haya un comportamiento no esperado del robot, se deberá pulsar la seta de emergencia para desconectar la alimentación.

2020

Presupuesto

DOCUMENTO 4

AUTOR: FERRÁNDIZ ALARCÓN, JESÚS

TUTORA: MIQUEL VALLÉS, MARINA

COTUTOR: PULLOQUINGA ZAPATA, JOSE LUIS

Documento 4: Presupuesto

Ref	Ud	Descripción	Precio	Cantidad	Total
Materiales					
PCB1	U	806-KF22X-C37P-NJ, Conector D-Sub	11,640 €	1	11,640 €
PCB2	U	MIC2981/82YN	1,930 €	1	1,930 €
PCB3	U	Regleta de terminales PCB, paso 2,54mm, 8 contactos	6,578 €	3	19,734 €
PCB4	U	Regleta de terminales PCB, paso 2,54mm, 4 contactos	2,292 €	1	2,292 €
PCB5	U	Resistencia SMD RS PRO, 7,5 kΩ	0,004 €	8	0,032 €
PCB7	U	Placa de baquelita	0,500 €	1	0,500 €
CR1	U	cRIO-9004 National Instruments	1.563,000 €	1	1.563,000 €
CR2	U	cRIO-9104 National Instruments	621,000 €	1	621,000 €
CR3	U	Módulo serie C NI-9263	493,000 €	2	986,000 €
CR4	U	Módulo serie C NI-9474	367,000 €	1	367,000 €
CR5	U	Módulo serie C NI-9421	119,000 €	1	119,000 €
CR6	U	Módulo serie C NI-9201	493,000 €	2	986,000 €
CR7	U	Cable RJ-45 de 5 metros.	5,990 €	1	5,990 €
CR8	U	Cables para conexión CRIO PLACA	5,450 €	1	5,450 €
PC1	U	Ordenador industrial	700,000 €	1	700,000 €
PC2	U	Licencia de LabVIEW	3.513,000 €	1	3.513,000 €
PC3	U	Módulo LabVIEW FPGA	3.135,000 €	1	3.135,000 €
				Subtotal	12.037,568 €
Mano de obra					
MO1	h	Ingeniero electrónico	20,000 €	30	600,000 €
MO2	h	Ingeniero de control	20,000 €	170	3.400,000 €
MO3	h	Operario	15,000 €	20	300,000 €
				Subtotal	4.300,000 €
Maquinaria					
MQ1	h	Máquina para fabricar PCB	4,000 €	1	4,000 €
MQ2	h	Soldador	2,000 €	1	2,000 €
				Subtotal	6,000 €
Medios auxilares					
%			4%	16.343,568 €	653,743 €

TOTAL PRESUPUESTO DE EJECUCIÓN MATERIAL	16.997,311 €
Beneficio industrial (6%)	1.019,839 €
TOTAL PRESUPUESTO DE EJECUCIÓN POR CONTRATA	18.017,149 €
IVA (21%)	3.783,601 €
TOTAL PRESUPUESTO BASE DE LICITACIÓN	21.800,751 €