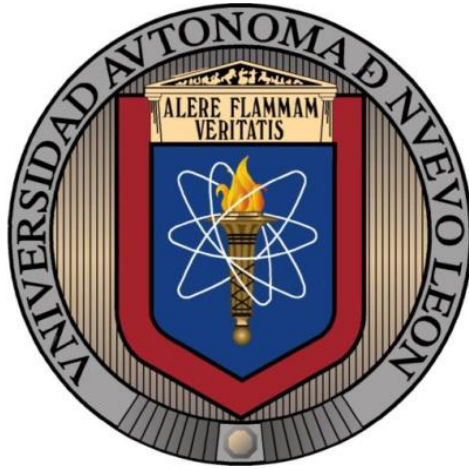


**UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA**



**ANÁLISIS E IMPLEMENTACIÓN DE ESTRATEGIAS DE
CONTROL TOLERANTE A FALLAS**

**POR
ING. ALFONSO BANDA URBINA**

**COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE
MAESTRÍA EN CIENCIAS DE LA INGENIERÍA ELÉCTRICA**

NOVIEMBRE, 2016

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA
SUBDIRECCIÓN DE ESTUDIOS DE POSGRADO



**ANÁLISIS E IMPLEMENTACIÓN DE ESTRATEGIAS DE
CONTROL TOLERANTE A FALLAS**

**POR
ING. ALFONSO BANDA URBINA**

**COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE
MAESTRÍA EN CIENCIAS DE LA INGENIERÍA ELÉCTRICA**

NOVIEMBRE, 2016

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica
Subdirección de Estudios de Posgrado

Los miembros del comité de tesis recomendamos que la tesis “Análisis e Implementación de Estrategias de Control Tolerante a Fallas”, realizada por el alumno Ing. Alfonso Banda Urbina, con número de matrícula 1535068, sea aceptada para su defensa como requisito parcial para obtener el grado de Maestría en Ciencias de la Ingeniería Eléctrica.

El Comité de Tesis




Dr. Efraín Alcorta García

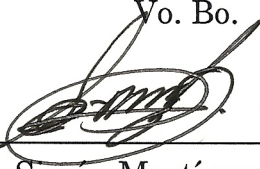
Asesor



Dr. David Alejandro Díaz Romero
Revisor



Dr. Alejandro Rodríguez Solís
Revisor

Yo. Bo.


Dr. Simón Martínez Martínez
Subdirección de Estudios de Posgrado

San Nicolás de los Garza, Nuevo León, a 15 de noviembre del 2016

ÍNDICE GENERAL

Agradecimientos	VII
Resumen	VIII
1. Introducción	1
2. Preliminares	3
2.1 Descripción de la Terminología.....	3
2.2 Propiedades de los Sistemas Sujetos a Fallas	5
2.3 Sistemas de Monitoreo y Supervisión.....	6
2.3 Modelado de las Fallas.....	8
3. Estrategias de Control Tolerante a Fallas	10
3.1 Reconfiguración Mediante Desacoplo de Perturbaciones	13
3.2 Seguimiento de Trayectoria Nominal Mediante Diseño Lineal Cuadrático....	18
3.3 Seguimiento de Trayectoria Nominal Mediante Enfoque Cascada	21
4. Descripción del Laboratorio de Pruebas	23
4.1 Descripción de la Plataforma de Implementación.....	23
4.1.1 NI miRIO–1900	23
4.1.1.1 Descripción General del Hardware	24

4.1.1.2	Arreglos de Compuertas Programables en Campo (FPGA).....	29
4.1.2	Software de Desarrollo de Sistemas NI LabVIEW.....	30
4.1.2.1	Módulo LabVIEW FPGA.....	32
4.1.2.1.1	Interfaz de Entrada/Salida.....	34
4.1.2.1.2	Comunicación de Datos	35
4.1.2.1.3	Temporización	38
4.1.2.1.4	Control.....	39
4.1.2.2	Módulo LabVIEW Real-Time	43
4.2	Descripción del Sistema	45
4.2.1	Descripción Matemática del Sistema	46
4.2.1.1	Modelo del Motor de Corriente Directa	47
4.2.1.2	Configuración del Lazo de Control de Corriente.....	48
4.2.2	Análisis Experimental del Sistema.....	50
4.2.2.1	Identificación del Sistema	52
4.2.3	Diseño del Control Nominal	56
4.2.3.1	Implementación del Sistema de Control Nominal	57
5.	Implementación de las Estrategias de Control Tolerante a Fallas	60
5.1	Escenario de Falla.....	60
5.2	Propiedades de la Ejecución en Pruebas de Laboratorio	61

5.3 Implementación de los Algoritmos de Control Tolerante a Fallas.....	63
5.3.1 Implementación del Actuador Virtual.....	63
5.3.2 Implementación del Diseño Lineal Cuadrático.....	65
5.3.3 Implementación del Enfoque Cascada	67
6. Conclusiones y Trabajo a Futuro.....	70
Bibliografía.....	74

AGRADECIMIENTOS

Agradezco:

A Dios, por darme la sabiduría necesaria para llegar hasta aquí y por llenar mi vida de dicha y bendiciones.

A mi asesor, el Dr. Efraín Alcorta García, mi más amplio agradecimiento por haber creído en mí y por su disposición para dirigir en todo momento el presente trabajo de tesis. Cuya experiencia y educación han sido mi fuente de motivación.

A mis padres, Alfonso Banda Torres y María Urbina Aguilar, les agradezco de todo corazón su apoyo incondicional, su guía y su confianza en la realización de mis sueños. Soy afortunado por contar siempre con su amor, comprensión y ejemplo. Este logro es para ustedes.

A Aurora Banda, Eloy Banda, Alejandro Banda y Daisy Banda, con mucho cariño, por la compañía y el apoyo que me brindan y por los sueños que hemos compartido.

A mis amigos, Karen, Alejandro, Oscar e Ivon, que siempre estuvieron a un lado mío para ayudarme, escucharme, aconsejarme y en muchas ocasiones guiarme.

Y a todas aquellas personas que de una u otra forma, colaboraron o participaron en la realización de esta investigación, hago extensivo mi más sincero agradecimiento.

RESUMEN

Ing. Alfonso Banda Urbina.

Candidato para obtener el grado de Maestría en Ciencias de la Ingeniería Eléctrica.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Título del estudio: ANÁLISIS E IMPLEMENTACIÓN DE ESTRATEGIAS DE CONTROL
TOLERANTE A FALLAS.

Número de páginas:

OBJETIVOS Y MÉTODO DE ESTUDIO: La presente tesis tiene como objetivo general establecer un panorama general y práctico, para el análisis e implementación de tres de las estrategias de control tolerante activo más representativas, utilizando la plataforma NI LabVIEW y el NI myRIO – 1900.

Como metodología se tiene que realizar una revisión de la bibliografía referente a la teoría de control tolerante a fallas y posteriormente establecer una forma para implementar las tres estrategias en un escenario de falla en común.

OBSERVACIONES Y CONCLUSIONES: Un punto importante en este trabajo fue el desarrollo de la implementación del control tolerante con el equipo y las herramientas disponibles en el laboratorio de pruebas. Aunque existen algunas

limitaciones para la implementación, se realizaron experimentos en donde cada estrategia pudo ser probada y analizada.

Un resultado importante, se basa en la idea inicial de tratar de realizar un análisis comparativo entre las tres estrategias, pero aunque es posible realizar este análisis, destaca mayormente el enfoque propio con el que cada estrategia trata un específico grupo de escenarios de falla.

Firma del asesor: _____
Dr. Efraín Alcorta García

CAPÍTULO 1

INTRODUCCIÓN

A lo largo de la historia, el ser humano ha buscado satisfacer sus necesidades a través de la creación de nuevo conocimiento e innovación. Esto ha provocado una revolución tecnológica que actualmente sigue creciendo, logrando expandir el desarrollo de las diferentes áreas de la ingeniería.

Desde la revolución industrial, los sistemas y procesos se han hecho cada vez más complejos en operación y en construcción. Alrededor de 1960 la influencia de la automatización en la operación y el diseño de procesos técnicos, ha tenido un incremento progresivo. Este desarrollo fue causado por una creciente demanda en el rendimiento de los procesos, la calidad de los productos, la independencia en la operación peligrosa del proceso de la presencia de operadores humanos y la posibilidad de relevar a los operadores de tareas monótonas reduciendo los costos de producción.

El uso de sistemas de automatización tuvo un impulso drástico alrededor de 1975 cuando los equipos computacionales relativamente confiables y baratos empezaban a estar disponibles, teniendo la capacidad de resolver muchos de los problemas de automatización con un solo dispositivo. Esto fue acompañado de un mayor progreso en las áreas de sensores, actuadores, sistemas de comunicación por bus e interfaces humano-maquina.

La mejora en la comprensión teórica de los procesos y funciones de automatización también jugó un papel importante en el desarrollo técnico; y en consecuencia hizo posible concebir y manejar procesos más complicados y de mayor dimensión. Se desarrollaron sistemas formados por muchos dispositivos interconectados y controlados por computadoras, que rigen su comportamiento a

través de interrelaciones de un gran número de variables, diseñados para operar en diversos ambientes y condiciones difíciles de predecir [6].

Eventualmente surgió la preocupación de hacer estos sistemas más seguros y confiables; suscitados en gran medida por catástrofes industriales de gran impacto ambiental y costo humano. Por lo que sistemas de monitoreo y supervisión comenzaron a desarrollarse e implementarse para detectar irregularidades en el funcionamiento general del sistema; intentando anticipar una acción preventiva basado principalmente en un conocimiento empírico.

Las principales alternativas para mantener el funcionamiento e integrar seguridad a los sistemas, eran el utilizar redundancia física o dispositivos específicos poco flexibles [6]. Esto suponía complicaciones debido a la disposición de los sistemas y al alto costo de implementación. Posteriormente, con el creciente interés de la comunidad de control automático y el desarrollo del diagnóstico de fallas, se estableció una alternativa para lidiar con los efectos de ciertas fallas a través de la reconfiguración del sistema de control, conocida como control tolerante a fallas (FTC).

El análisis de algunas de las estrategias más representativas del control tolerante a fallas será tratadas en esta tesis, así como el desarrollo de un sistema de implementación para la ejecución de pruebas en un entorno de laboratorio sencillo que permita mostrar las cualidades y complicaciones de los diferentes métodos.

PRELIMINARES

2.1 DESCRIPCIÓN DE LA TERMINOLOGÍA

Al desarrollar la terminología en algún campo, normalmente se toman palabras del vocabulario común que se aproximan al concepto que los especialistas tienen en mente. El término “falla” es utilizado comúnmente en un sentido práctico en la vida cotidiana y eventualmente ha sido asimilada en la terminología de diferentes áreas de la ingeniería y otras ciencias conservando la esencia práctica de su significado, pero adaptándose al contexto del marco teórico en el que se utiliza.

Al crecer el interés en el tratamiento de las fallas en las diferentes áreas de la tecnología, surgió, dentro de la comunidad de ingeniería de control, la intención de establecer un vocabulario común [6]. Las acciones coordinadas dentro del Comité Técnico de SAFEPROCESS de la IFAC (International Federation of Automatic Control) lograron establecer la mayoría de las definiciones utilizadas comúnmente por la comunidad [5], al ser publicadas en los trabajos de Isermann y Ballé (1997). Posteriormente la traducción de la terminología usada en español fue publicada por Puig et al. (2004), de donde la mayoría de las definiciones mencionadas en este trabajo son extraídas.

Las definiciones fundamentales usadas en el contexto de seguridad son las que describen los conceptos de falla, avería y disfunción. Considerando el reemplazo de la palabra fallo por el léxico latinoamericano falla [6], las definiciones son las siguientes:

- **Falla:** desviación no permitida de, al menos, una propiedad característica o parámetro de un sistema de su condición aceptable, usual o estándar.

- **Avería:** interrupción permanente de la capacidad de un sistema para realizar una función requerida bajo las condiciones de operación especificadas.
- **Disfunción:** Irregularidad intermitente en el cumplimiento de una función deseada del sistema.

La definición del término “falla” ayuda a aclarar el concepto, pero no queda tan claro cuando se intenta distinguir de las perturbaciones o las incertidumbres del modelo al realizar un análisis más profundo. En principio, las perturbaciones e incertidumbres del modelo tienen efectos similares en el sistema [2]. Las perturbaciones suelen ser representadas por señales de entrada desconocidas que se añaden a la salida del sistema. Las incertidumbres del modelo cambian los parámetros del modelo de forma similar que las fallas. Sin embargo, tal vez, la diferencia más importante entre perturbaciones, incertidumbres del modelo y fallas se puede ver en el hecho de que las perturbaciones e incertidumbres modelo están siempre presentes, mientras que las fallas pueden estar presentes o no. Las perturbaciones representan la acción del medio ambiente con el sistema, mientras que las incertidumbres son el resultado de las actividades de interpretación y análisis que terminan con un modelo como una representación aproximada del comportamiento del sistema. Ambos fenómenos son inconvenientes que se presentan en mayor y menor medida, y sus efectos sobre el desempeño del sistema son manejados por medidas apropiadas como el filtrado, el control de retroalimentación o el diseño robusto.

Existen otras consideraciones importantes al describir las fallas. Al modelarse las fallas suelen estar representados como señales externas adicionales o como desviaciones de los parámetros nominales. También, se clasifican según su dependencia temporal en [6]:

- **Falla abrupta.** Donde la falla se manifiesta con un cambio repentino y puede ser modelado mediante una función tipo escalón.
- **Falla incipiente.** Esta falla se manifiesta como un cambio en magnitud en las variables del sistema que va aumentando paulatinamente con respecto al tiempo y puede modelarse mediante una función tipo rampa con un tiempo de ocurrencia desconocido.

- **Falla intermitente.** Este tipo de falla se considera que no tiene una evolución determinada en el tiempo y frecuentemente se presenta solamente en ciclos de trabajo de manera aleatoria y desaparece también de forma aleatoria.

2.2 PROPIEDADES DE LOS SISTEMAS SUJETOS A FALLAS

Debido a las consecuencias que pueden ocasionar las fallas, como daños al medio ambiente y el riesgo para la vida humana, los ingenieros han investigado su aparición y sus efectos durante décadas. Diferentes nociones como la seguridad, la confiabilidad y la disponibilidad se han definido e investigado [6].

La **Seguridad** describe la ausencia de peligro. Un sistema de seguridad es una parte del equipo de control que protege un sistema tecnológico de daño permanente. Permite un apagado controlado, el cual lleva al proceso con a un estado seguro. Para ello, evalúa la información para señales críticas y activa los actuadores dedicados para detener el proceso si se cumplen determinadas condiciones. El sistema en su conjunto se denomina entonces un sistema a prueba de fallos.

La **Confiabilidad** es la probabilidad de que un sistema lleve a cabo su función prevista para un período de tiempo especificado en condiciones normales. Los estudios de confiabilidad evalúan la frecuencia con la que el sistema entra en estado de falla, pero no pueden decir nada sobre el estado de falla actual.

La **Disponibilidad** es la probabilidad de que un sistema opere cuando sea necesario. Al igual que la confiabilidad, la disponibilidad también depende de las políticas de mantenimiento que se aplican a los componentes del sistema.

La extensión natural para aumentar estas propiedades en los sistemas de control, es añadiendo actuadores o leyes de control suplementarias que permitan que el sistema de control ejerza sus funciones sobre el proceso aun ante una falla o uso

incorrecto del sistema. Los sistemas de control enfocados en el manejo de las fallas, se les denota como **control tolerante a fallas o FTC** (Fault Tolerant Control).

2.3 Sistemas de Monitoreo y Supervisión

Habitualmente un proceso bien automatizado integra varios niveles con el propósito de mantener un funcionamiento global del proceso, no solo con un desempeño satisfactorio cuando éste opera en condiciones nominales, sino también en los diversos modos de funcionamiento incluyendo los modos de falla. El nivel más cercano a la planta, está conformado por el sistema de control nominal, comúnmente formado de un control secuencial, un control retroalimentado o un compensador. Los niveles más altos comprenden tareas de acción más global como coordinación, optimización y administración. Por lo que el nivel intermedio correspondería al de la **supervisión del sistema**.

La **supervisión** se encarga de mostrar el estado presente del sistema, indicando estados no deseados o no permitidos, y tomando acciones apropiadas para evitar daños o accidentes. El supervisor es la entidad humana o artificial que realiza las tareas de supervisión de un proceso mediante el diagnóstico de fallas, determinación y ejecución de las acciones para corregir las fallas.

La **supervisión** está compuesta por un conjunto de tareas que pueden ejecutarse de manera asíncrona sobre el proceso, en función de su situación actual y las cuales están basadas en reglas de operación establecidas desde que se conceptualiza el sistema. La implementación de un sistema de supervisión supone recorrer tres etapas fundamentales: el monitoreo o vigilancia, la detección y el diagnóstico de fallas, y la reconfiguración del sistema.

- **Monitoreo:** es la determinación continua en tiempo real del estado de operación de un sistema mediante el registro y análisis de información significativa e indicación de sus anomalías de comportamiento.

- **Detección de Fallas:** es la determinación de la presencia de fallas en el sistema así como el instante de su aparición.
- **Diagnóstico de Fallas:** es la determinación del tipo, tamaño, localización e instante de aparición de una falla. Incluye la detección, el aislamiento y la estimación de la falla.
- **Reconfiguración del Sistema:** es el cambio en la estructura original del sistema para mantener su operación al afrontar una situación de falla.

Normalmente la supervisión automática se realiza por comprobación de límites (o comprobación de umbrales) de algunas variables importantes del proceso [6]. Usualmente las alarmas se activan cuando el límite de los valores de las variables se sobrepasa y los operadores o los sistemas de protección y/o recuperación deben actuar. El **control tolerante a fallas** ofrece una alternativa para la reconfiguración del sistema en las situaciones donde es posible recuperar la funcionalidad del sistema al presentarse fallas.

Ya que la función del sistema está determinada por el control nominal, el objetivo del control tolerante fallas depende del objetivo de control del controlador nominal. Los objetivos de control más comunes al implementar un sistema de control son estabilizar el sistema, alcanzar un cierto valor de referencia y seguir una trayectoria dada. El esquema de control clásico mostrado en la Figura 2.1 ejemplifica un sistema de control enfocado en alcanzar una referencia. Este esquema servirá posteriormente como base para mostrar las estructuras de control tolerante a fallas.

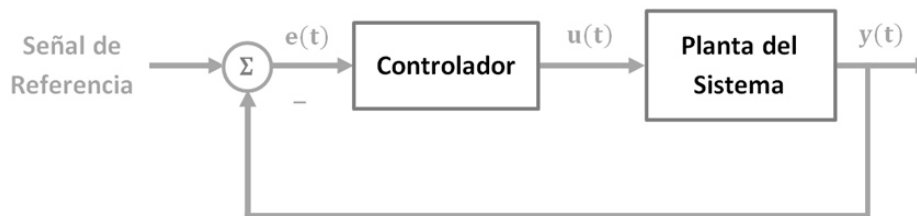


Figura 2.1: Esquema de Control Clásico.

2.4 Modelado de las Fallas

Durante la operación del sistema, la ocurrencia de las fallas puede afectar los sensores, los actuadores, o los componentes del sistema. Estas fallas pueden ocurrir como fallas aditivas o multiplicativas debido a una disfunción o al envejecimiento del equipo.

Para el sistema de diagnóstico de fallas, una distinción es usualmente hecha entre fallas aditivas y multiplicativas. Sin embargo, en el sistema de control tolerante a fallas, el objetivo es compensar el efecto de la falla en el sistema, independientemente de la naturaleza de la falla [9].

Las fallas que afectan a un sistema, a menudo son representadas por una variación de los parámetros del sistema. Por lo que en presencia de una falla, el modelo del sistema puede ser escrito como

$$\begin{aligned}\dot{x}_f(t) &= A_f x_f(t) + B_f u_f(t) \\ y_f(t) &= C_f x_f(t)\end{aligned}\tag{2.1}$$

donde las nuevas matrices del sistema con falla están definidas como

$$A_f = A + \delta A; \quad B_f = B + \delta B; \quad C_f = C + \delta C\tag{2.2}$$

y δA , δB , y δC corresponden a la desviación de los parámetros del sistema con respecto a los valores nominales.

Diferentes tipos de fallas pueden presentarse, desde fallas típicas que cambian solo una pequeña parte de la planta o hasta una la perdida completa de la operación de alguno o varios componentes. Por lo que del modelo con falla se distinguen tres tipos de fallas a ser tratadas.

- **Fallas en los Sensores.** Son representadas en el modelo con falla por la matriz de salida C_f y su diferencia con respecto a la matriz de salida nominal C caracteriza el tamaño de la falla.

- **Fallas en los Actuadores.** Al igual que las fallas en los sensores, las fallas en los actuadores tienen una representación dual en la matriz de entrada B_f . Por ejemplo, una pérdida completa de un actuador se modelado sustituyendo su correspondiente columna en B_f con ceros, de esta manera la variable de entrada puede tomar cualquier valor sin afectar el sistema. Dado este tipo de falla, puede afectar la controlabilidad de la planta. Con el fin de hacer posible la reconfiguración, se asume que los polos no controlables del par (A, B_f) son estables.
- **Fallas Internas.** Dado que estas fallas afectan los acoplamientos internos de la planta, son llamadas “fallas internas”.

ESTRATEGIAS DE CONTROL TOLERANTE A FALLAS

El control tolerante a fallas se puede ver en última instancia como una herramienta (junto con el diagnóstico de fallas) a integrar en los sistemas de supervisión y control para aumentar la confiabilidad a la presencia de fallas previamente estudiadas.

Las diferentes estrategias para la implementación del control tolerante a fallas se pueden clasificar en dos enfoques generales:

- **Control Pasivo.** El cual se enfoca en utilizar la propiedad que tienen los sistemas retroalimentados de hacer frente a perturbaciones o cambios en la dinámica del sistema para atenuar los efectos de las fallas. Prácticamente consiste en un diseño robusto del sistema de control retroalimentado para hacerlo inmune a determinadas fallas.

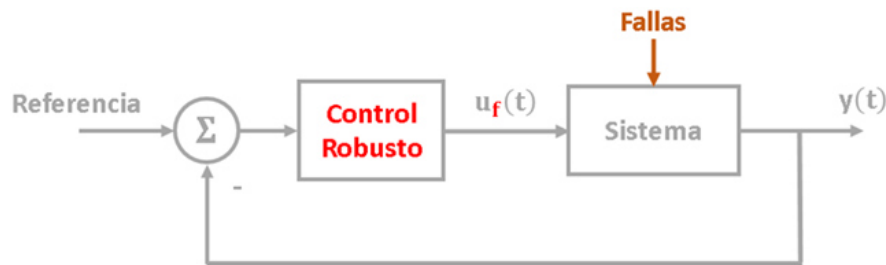


Figura 3.1: Esquema de Control Pasivo Tolerante a Fallas.

- **Control Activo.** Este enfoque consiste en utilizar la información del sistema de diagnóstico de fallas, para activar el mecanismo de reconfiguración del control, diseñado para compensar el efecto de las fallas.

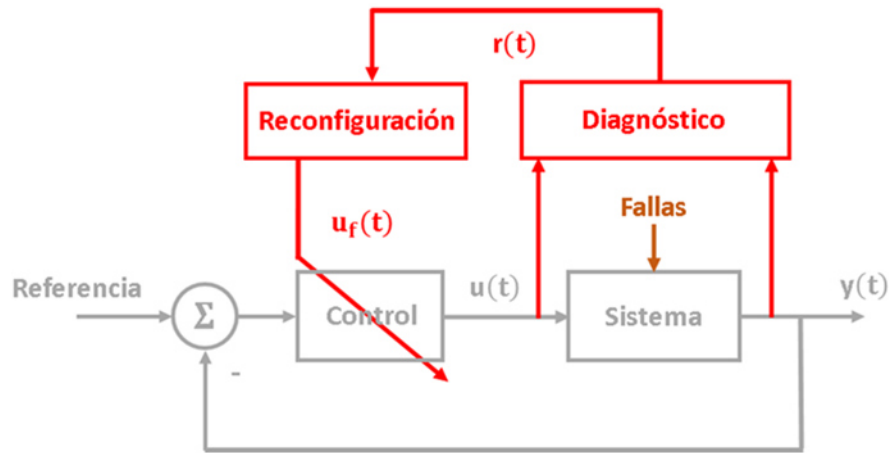


Figura 3.2: Esquema de Control Activo Tolerante a Fallas.

El control activo supone un mayor costo de implementación al necesitar capacidades de supervisión y diagnóstico, pero ofrece una mejor solución para fallas específicas. La reconfiguración del control es la respuesta planteada para manejar la ocurrencia de fallas y es la base del control tolerante activo. Durante la última década, las diferentes comunidades han tratado este tema, y desafortunadamente se ha utilizado terminología que es incompatible. Varias definiciones diferentes han sido usadas para el término “reconfiguración”, dando lugar a enfoques que no pueden ser razonablemente comparados entre sí. Para este trabajo, se considera la definición de Isermann y Ballé [5].

Reconfiguración es el cambio en las entradas y/o salidas del controlador a través de un cambio en la estructura del controlador y sus parámetros. Para esto, se pretende alcanzar los objetivos de control, aunque las prestaciones se pueden degradar.

La eficacia con la que el objetivo de control se alcanza (en este caso el recuperar la respuesta de salida nominal del sistema original) al aplicar la reconfiguración del control, deriva en dos objetivos principales de reconfiguración muy utilizados en la literatura. Las siguientes definiciones formales fueron extraídas de [1].

- **Reconfiguración Débil.**

El objetivo de la reconfiguración débil requiere que el lazo de control reconfigurado tenga el mismo comportamiento estacionario que el del lazo nominal.

Definición:

Un lazo de control reconfigurado satisface el objetivo de reconfiguración débil, si y solo si la salida $y_f(t)$ de la planta con falla converge a la salida $y_n(t)$ tal como está definido por la planta nominal en el lazo de control nominal.

$$\lim_{t \rightarrow \infty} y_n(t) - y_f(t) = 0 \quad (3.1)$$

para todas las señales de entrada constantes y todos los estados iniciales x_0 .

- **Reconfiguración Fuerte.**

El objetivo de reconfiguración fuerte va más allá, se requiere que el comportamiento dinámico sea idéntico también.

Definición:

Un lazo de control reconfigurado satisface el objetivo de reconfiguración fuerte, si y solo si la salida $y_f(t)$ de la planta con falla sigue la trayectoria definida por la salida $y_n(t)$ de la planta nominal en el lazo de control nominal.

$$\forall t \in \mathbb{R}_+: y(t) = y_f(t) \quad (3.2)$$

para todas las trayectorias de entrada y todos los estados iniciales x_0 .

La formulación de varias de las estrategias de control tolerante a fallas, se basan en ideas tomadas de las soluciones a diversos problemas de control. Tres de estas estrategias más representativas serán tratadas en esta tesis, desde su análisis previo, hasta su posterior implementación física en un prototipo de laboratorio.

3.1 RECONFIGURACIÓN MEDIANTE DESACOPLO DE PERTURBACIONES (ACTUADOR VIRTUAL)

Para esta estrategia de control tolerante se añade un bloque entre el controlador nominal y la planta con falla, llamado “actuador virtual”, cuya función principal es compensar el efecto de la falla, sin modificar el controlador nominal (Fig.3.3). Esta estrategia se enfoca en situaciones de falla en donde se pierde la funcionalidad de uno o más actuadores del sistema, como por ejemplo, al sufrir una avería en algún actuador o al no contar con un sistema de diagnóstico que puede estimar la falla.



Figura 3.3: Esquema del Actuador Virtual.

El actuador virtual restablece la funcionalidad del sistema, estableciendo una configuración alterna para la acción de control, diseñada para cumplir el objetivo de la reconfiguración fuerte y ser estable, mientras el controlador nominal sigue siendo parte del lazo de control reconfigurado. Esto conlleva a condiciones de existencia estrictas, debido a que las cualidades de algunos sistemas no permiten la reconfiguración del lazo de control.

Para construir el actuador virtual, se empieza comparando la salida del sistema con falla con la salida nominal del modelo (Fig.3.4).

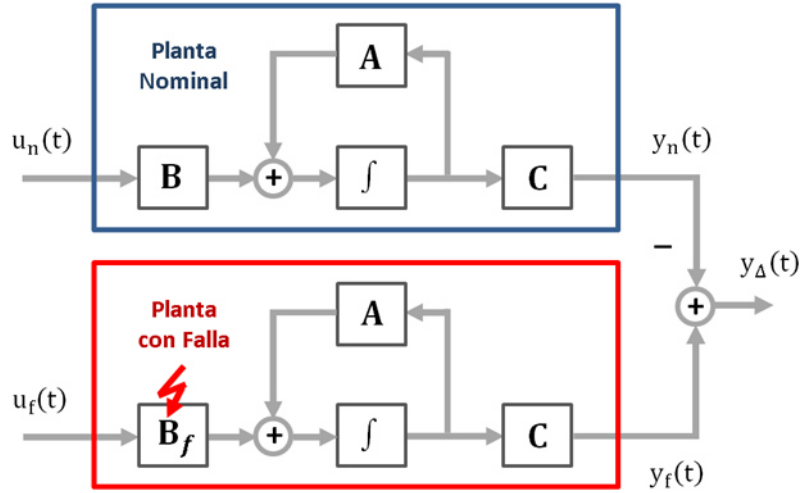


Figura 3.4: Comparación Salida Nominal contra Salida con Falla.

La planta nominal y con falla se describen con las ecuaciones:

$$\dot{x}_n(t) = Ax_n(t) + Bu_n(t)$$

$$\dot{x}_f(t) = Ax_f(t) + B_f u_f(t)$$

(3.3)

$$y_\Delta(t) = y_n(t) - y_f(t) = C(x_n(t) - x_f(t))$$

$$x_f(t_f) = x_n(t_f)$$

Para lograr el objetivo de la reconfiguración fuerte la salida de desviación $y_\Delta(t)$, deberá permanecer en 0. Al introducir el estado de desviación $x_\Delta(t) = x_n(t) - x_f(t)$, se obtiene la ecuación de estado:

$$\dot{x}_\Delta(t) = Ax_\Delta(t) + Bu_n(t) - B_f u_f(t)$$

(3.4)

$$x_\Delta(t_f) = x_n(t_f) - x_f(t_f) = 0$$

En donde la salida del controlador nominal $u_n(t)$ puede ser interpretada como una perturbación, y se busca la manera de desacoplarla de la salida de desviación $y_\Delta(t)$. La manera que se plantea para resolver este problema es utilizando el enfoque geométrico.

El enfoque geométrico fue desarrollado por Wonham (1985) y Basile y Marro (1992); y proporciona una forma natural para aplicarse en modelos de espacio de estados [1]. El enfoque geométrico está centrado alrededor de la noción de un subespacio condicionalmente invariante del espacio de estados \mathbb{R}^n . Un subespacio $\mathcal{V} \subseteq \mathbb{R}^n$ es llamado invariante, si el estado del sistema nunca deja el subespacio una vez que ha entrado en él. Y es llamado (A, B_f) –invariante si existe una matriz de retroalimentación de estado que genere el subespacio invariante.

Al dividir el sistema x_Δ en dos partes, donde un subsistema representa el subespacio (A, B_f) –invariante, y el otro subsistema contiene el resto, se puede lograr mantener $y_\Delta(t) = 0$. Una condición importante de esta división es que no exista conexión entre ambos subsistemas. Por lo que, la ley de control $u_f(t)$ se diseña con el objetivo de hacer que la perturbación $u_n(t)$ solo afecte la parte invariante del sistema, y la salida $y_\Delta(t)$ dependa solo de la otra parte del sistema. Para esto el subespacio invariante \mathcal{V} debe satisfacer concretamente tres condiciones, que se pueden formular de la siguiente manera:

$$\begin{aligned}
 \textit{invarianza: } & A\mathcal{V} \subseteq \mathcal{V} + \text{im } B_f \\
 \textit{inobservable: } & \mathcal{V} \subseteq \ker C \\
 \textit{imperturbable: } & \text{im } B \subseteq \mathcal{V} + \text{im } B_f
 \end{aligned} \tag{3.5}$$

En esta notación, “im” denota la imagen de la transformación (el conjunto de posibles valores de salida) y “ker” denota el kernel de la transformación (el conjunto de entradas sin efecto). La multiplicación $A\mathcal{V}$ denota el conjunto de soluciones de la transformación A basadas en los valores dentro de \mathcal{V} , y la adición $\mathcal{V} + \text{im } B_f$ denota la combinación lineal de los dos subespacios.

Si las tres condiciones para el subespacio invariante \mathcal{V} se cumple, una solución

$$\begin{aligned}
 \exists M : & (A + B_f M)\mathcal{V} \subseteq \mathcal{V} \\
 \exists N : & \text{im}(B + B_f N) \subseteq \mathcal{V}
 \end{aligned} \tag{3.6}$$

puede ser encontrada. Vale la pena señalar que la solución por lo general no es única. Cualquier $u_f(t)$ con $B_f u_f(t) \in \mathcal{V}$ se puede añadir a la entrada sin afectar la propiedad de desacoplamiento. Por lo tanto, la ley de control tendría la forma

$$u_f(t) = Mx_\Delta(t) + Nu_n(t) + Q\tilde{u}(t) \quad (3.7)$$

donde

$$\text{im } Q = B_f^{-1}\mathcal{V}$$

denota el subespacio de entrada indefinido y $\tilde{u}(t)$ puede tomar cualquier valor sin perturbar la segunda parte del sistema, y ofrece una alternativa para solucionar el problema de estabilidad.

El sistema que genera el estado $x_\Delta(t)$, junto con la ley de control $u_f(t)$, componen el bloque de reconfiguración conocido como “actuador virtual”. La matriz M hace que el subespacio \mathcal{V} sea invariante para el sistema del actuador virtual, mientras la matriz S hace que el efecto de la perturbación se mueva dentro de este subespacio.

Con el fin de encontrar parámetros adecuados, \mathcal{V} tiene que ser construido primero. El algoritmo clásico para encontrar la solución está basado en el hecho que \mathcal{V} es un subespacio máximo.

La serie

$$\mathcal{V}_0 = \ker C \quad (3.8)$$

$$\mathcal{V}_{k+1} = \mathcal{V}_k \cap A^{-1}(\mathcal{V}_k + \text{im } B_f)$$

se vuelve estacionario después de al menos n pasos y se obtiene el máximo subespacio invariante $\mathcal{V} = \mathcal{V}_n$. Al comprobarse la existencia del subespacio \mathcal{V} , se obtiene la matriz de rango completo V_K con $\ker V_K = \mathcal{V}$. Al tener V_K , los requerimientos para una solución se pueden escribir como:

$$V_K B_f M = V_K A \quad (3.9)$$

$$V_K B_f N = V_K B$$

Y si una solución existe, se puede escribir como

$$\begin{aligned} M &= (V_K B_f)^+ V_K A \\ N &= (V_K B_f)^+ V_K B \end{aligned} \tag{3.10}$$

donde $(V_K B_f)^+$ denota la pseudoinversa del lado izquierdo de $V_K B_f$, el cual resuelve

$$V_K B_f = V_K B_f (V_K B_f)^+ V_K B_f \tag{3.11}$$

Mientras esta solución desacopla la salida $y_\Delta(t)$, usualmente no estabiliza el sistema del actuador virtual. La entrada libre $\tilde{u}(t)$ se puede usar para estabilizar el subespacio invariante sin afectar el desacoplamiento. En la literatura este problema es conocido como desacoplo de perturbaciones conocidas con estabilidad (DDPS). Con el fin de encontrar una matriz de retroalimentación apropiada, el sistema tiene que ser reducido al subespacio invariante. Esto se logra mediante la reducción de estado

$$x'(t) = T x_\Delta(t) \tag{3.12}$$

el cual mapea el subespacio invariante \mathcal{V} dentro de un nuevo espacio de estado $x'(t) \in \mathbb{R}^m$, donde m es la dimensión de \mathcal{V} . La transformación inversa está definida por la pseudoinversa T^+ con $TT^+ = I_m$ y $\text{im } T^+ = \mathcal{V}$.

La parte invariante del sistema es

$$\begin{aligned} \dot{x}'(t) &= A' x'(t) + B' u_c(t) - B'_f u_f(t) \\ u_f(t) &= M' x'(t) + N u_c(t) + Q \tilde{u}(t) \\ y_\Delta(t) &= C' x'(t) \end{aligned} \tag{3.13}$$

$$x'(0) = 0$$

con

$$A' = T A T^+$$

$$B' = TB$$

$$B'_f = TB_f \quad (3.14)$$

$$C' = CT^+$$

$$M' = MT^*$$

Este sistema reducido tiene que ser estabilizado sin afectar la propiedad de desacoplamiento. El grado de elección se hace explícito en la entrada libre $\tilde{u}(t)$. Por lo que, un controlador estabilizante de la forma

$$\tilde{u}(t) = Kx'(t) \quad (3.15)$$

puede ser aplicado sin afectar la propiedad de desacoplamiento. La matriz de retroalimentación K tiene que estabilizar el par $(A' - B'_f M', B'_f Q)$.

Varios métodos de diseño de controladores (como asignación de polos o diseño lineal cuadrático) pueden ser usados para encontrar K . Para el objetivo de estabilización, se requiere que los polos del actuador virtual sean estables.

3.2 SEGUIMIENTO DE TRAYECTORIA NOMINAL MEDIANTE DISEÑO LINEAL CUADRÁTICO

Dentro de las estrategias de control tolerante activo, se pueden diferenciar estrategias, que como en el caso del actuador virtual, buscan minimizar las diferencias entre la dinámica de los sistemas nominales y los sistemas con fallas. En el caso de esta y la siguiente estrategias de control tolerante activo, se busca que el sistema con falla siga la trayectoria del sistema nominal.

Para esta estrategia en particular, la idea básica es hacer que el sistema con falla siga la trayectoria del sistema nominal mediante el uso de una función de costo cuadrático, la cual pondera la diferencia entre la trayectoria actual $y_f(t)$ y la

trayectoria nominal $y_n(t)$. La existencia de la solución se garantiza con la estabilidad del sistema con falla, es decir, este método solo se puede aplicar a la clase de fallas que no afecten la estabilidad. Esta estrategia permite compensar fallas internas y en los actuadores.

Esta estrategia fue desarrollada en el trabajo de (Staroswiecki y Cazaurang, 2008)[4]. En él se plantea el problema de seguimiento de trayectoria nominal, como un problema de control óptimo de regulación al origen. Se considera un análisis de admisibilidad para las fallas y se propone un método para el cálculo de los parámetros del control tolerante, para una acomodación progresiva de la falla.

En trabajos posteriores [16], se propone un diseño diferente del control tolerante. Se considera un índice de desempeño que solo considera el error de seguimiento de trayectoria y la influencia del control de reconfiguración. También el problema se plantea como un problema de seguimiento a una referencia variante en el tiempo, en vez del de regulación al origen.

Para este trabajo, se plantea un diseño de control como el mencionado anteriormente, considerando la solución en un horizonte infinito.

Se establece un índice de desempeño

$$J = \frac{1}{2} \int_{t_f}^{\infty} \left((x_f(t) - x_n(t))^T Q (x_f(t) - x_n(t)) + u_f(t)^T R u_f(t) \right) dt \quad (3.16)$$

que se busca minimizar, con $Q = C^T C$ para ponderar el seguimiento de la trayectoria nominal, y R para ponderar la ley de control de reconfiguración. Se considera la ocurrencia de una falla en el actuador y se obtiene el conjunto de condiciones óptimas necesarias

$$\dot{x}_f(t) = Ax_f(t) + B_f u_f(t)$$

$$\dot{p}_f(t) = -Q (x_f(t) - x_n(t)) - A^T p_f(t) \quad (3.17)$$

$$0 = R u_f(t) + B_f^T p_f(t)$$

Donde el superíndice T denota transpuesta y $p_f(t)$ es el vector de estado adjunto. Siendo la ley de control de reconfiguración

$$u_f(t) = -R^{-1}B_f^T p_f(t) \quad (3.18)$$

Siguiendo el problema clásico de seguimiento de trayectoria y horizonte infinito, el estado adjunto toma la forma

$$p_f(t) = Kx_f(t) + S \quad (3.19)$$

con K y S obtenidas de

$$0 = KA + A^T K - KB_f R^{-1} B_f^T K + Q \quad (3.20)$$

$$0 = -(A^T - KB_f R^{-1} B_f^T)S + Qx_n(t)$$

Al implementar el control tolerante a fallas, el esquema del lazo de control reconfigurado estaría representado en la Fig 3.4, donde al presentarse la falla se reconfigura el lazo sustituyendo el controlador nominal por el control tolerante. En algunos casos no es necesario sustituir la ley de control nominal, sino más bien acomodar sus parámetros a los del control tolerante.

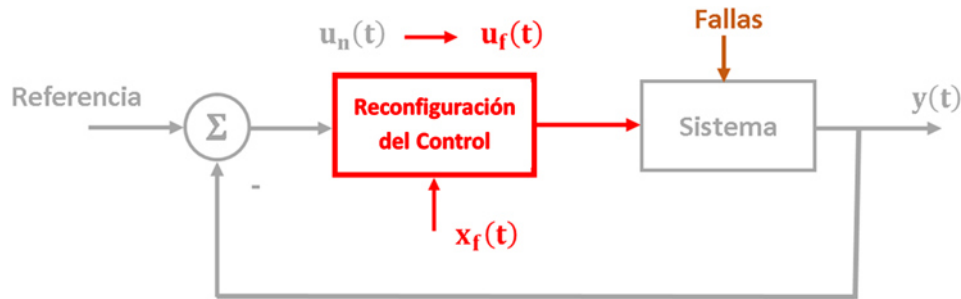


Figura 3.4: Esquema del Control LQ.

Como se muestra en la Fig 3.4, es necesario tener disponible la información del estado nominal y la retroalimentación constante del estado con falla, aunque que si la salida del sistema $y(t)$ originalmente no depende del estado completo y $Q = C^T C$ es seleccionado, solo sería necesario comparar las trayectorias, en lugar de todo el estado.

3.3 SEGUIMIENTO DE TRAYECTORIA NOMINAL MEDIANTE ENFOQUE CASCADA

Al igual que el enfoque anterior la idea básica es utilizar la salida del sistema nominal, obteniéndola de un modelo, como referencia a seguir cuando ocurre una falla. La gran distinción con la cuenta este enfoque es que utiliza un esquema de control en cascada como el que se muestra en la Fig 3.5.

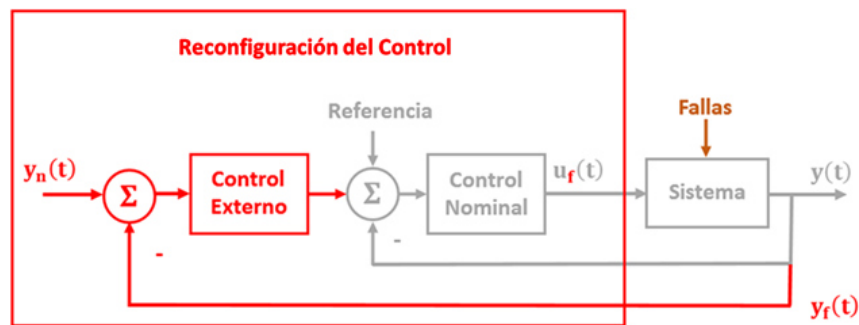


Figura 3.5: Esquema del Control Cascada.

Una ventaja de este enfoque es que el lazo de control nominal se mantiene intacto después de que la falla aparece. Sin embargo, se genera un nuevo lazo exterior que incluye el lazo del sistema de control nominal como un lazo interno. Este arreglo forma una estructura cascada con un lazo interno formado por el control nominal que no cambia y un lazo externo formado por control de reconfiguración. La solución se reduce a diseñar un control que garantice estabilidad y desempeño del sistema cuando ocurre la falla [8].

El error de estimación, el cual se obtiene de la diferencia entre la medición y el valor de salida nominal, se utiliza como base para la corrección. El objetivo de control es el de seguimiento de la trayectoria nominal y al igual que las demás estrategias de control tolerante activo basadas en este enfoque, se busca cumplir con el objetivo de reconfiguración débil.

Diferentes tipos de controladores pueden ser candidatos, para resolver el problema indicado con la estructura considerada. En [8], por razones de simplicidad se

considera y analiza una realimentación de salida estática, en donde la ganancia del controlador se define por regiones de estabilidad con LMI.

Al seleccionar un controlador más complejo, podría dificultarse el análisis de estabilidad necesario para su implementación. Además, que el conocimiento del control nominal es requerido para el diseño. Un controlador PI, es la herramienta más común para el seguimiento de una referencia y la estabilidad del sistema reconfigurado se puede determinar por simulación.

DESCRIPCIÓN DEL LABORATORIO DE PRUEBAS

4.1 DESCRIPCIÓN DE LA PLATAFORMA DE IMPLEMENTACIÓN

Las herramientas de implementación y diseño con las que dispone el laboratorio de pruebas, se compone básicamente de una combinación de Hardware y Software distribuidos por la empresa National Instruments (NI).

National Instruments ofrece una plataforma con un desarrollado enfoque de control, que permite la realización de las pruebas requeridas para la realización de esta tesis. Dentro de las opciones de hardware y software que NI ofrece, existen dispositivos y licencias enfocados en el uso estudiantil que cuentan con facilidades de adquisición para las instituciones educativas.

El laboratorio cuenta con el software NI LabVIEW que es la piedra angular de la plataforma de desarrollo de sistemas de NI, además de varios módulos que extienden las herramientas básicas del NI LabVIEW y permiten diseñar aplicaciones en diferentes niveles de la estructura de hardware.

El hardware utilizado para la implementación física en el laboratorio es el NI myRIO – 1900, siendo un dispositivo con recursos modestos y de uso estudiantil, pero enfocado para su utilización en aplicaciones de control.

4.1.1 NI MYRIO – 1900

El National Instruments myRIO – 1900 es un dispositivo portátil de entradas y salidas reconfigurables (RIO) dirigido para uso estudiantil y aplicaciones básicas. El NI myRIO usa la tecnología Zynq de Xilinx, que ofrece un FPGA integrado con un procesador ARM® Cortex™-A9 dual-core que ejecuta un Sistema Operativo (OS) en tiempo real. Debido a sus componentes internos, acceso a software transparente y biblioteca de recursos y tutoriales, NI myRIO es una herramienta accesible con la capacidad suficiente para desempeñar las tareas de control requeridas.

4.1.1.1 DESCRIPCIÓN GENERAL DEL HARDWARE

El NI myRIO-1900 proporciona entradas analógicas (AI), salidas analógicas (AO), entradas y salidas digitales (DIO), audio, y salidas de alimentación en un dispositivo integrado compacto. El NI myRIO-1900 se conecta a una computadora (host) a través de un cable USB y a través de conexión inalámbrica (wireless) 802.11b,g,n [12].

La Figura 4.1 muestra la disposición y las funciones de los componentes del NI myRIO-1900.

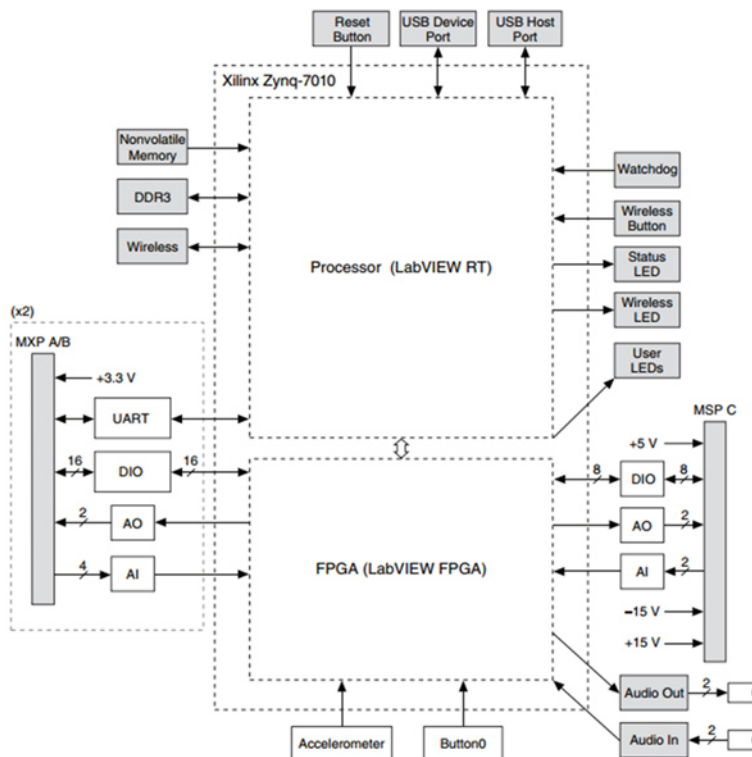


Figura 4.1: Diagrama de Bloques del Hardware NI myRIO-1900

• **DISPOSICIÓN DE LOS PINES DE CONEXIÓN.**

Los conectores A y B del Puerto de Expansión del NI myRIO-1900 (MXP) tienen grupos idénticos de señales. Las señales se distinguen en el software por el nombre del conector, como en ConnectorA/DIO1 y ConnectorB/DIO1. La Figura 4.2 y Tabla 4.1 muestran las señales en los conectores A y B del MXP.

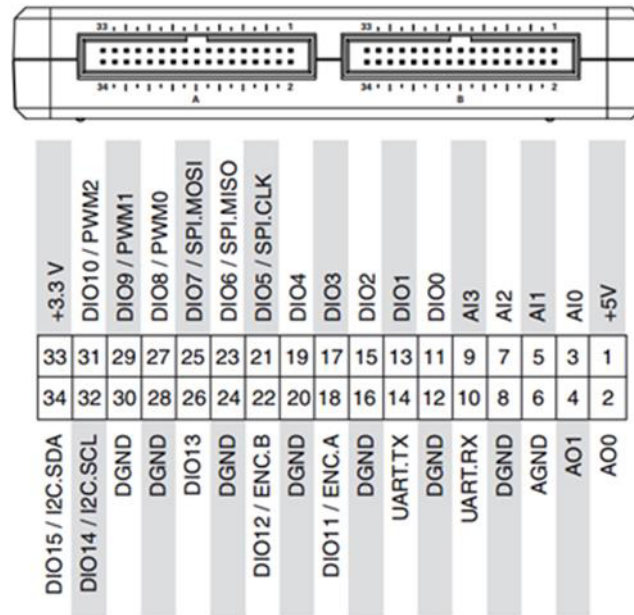


Figura 4.2: Señales Primarias/Secundarias en los Conectores A y B del MXP.

Nombre de la Señal	Referencia	Dirección	Descripción
+5V	DGND	Salida	+5 V salida de alimentación.
AI < 0..3 >	AGND	Entrada	0–5 V, referenciados, canales de entrada analógica de una sola terminal.
AO < 0..1 >	AGND	Salida	0–5 V, referenciados, canales de salida analógica de una sola terminal.
AGND	N/A	N/A	Referencia para la entrada y salida analógica.
+3.3 V	DGND	Salida	+3.3 V salida de alimentación.
DIO < 0..15 >	DGND	Entrada o Salida	Lineas digitales de propósito general con 3.3 V de salida, 3.3 V/5 V de entrada compatible.

UART. RX	DGND	Entrada	Entrada de recepción UART. Las líneas UART son eléctricamente idénticas a las líneas DIO.
UART. TX	DGND	Salida	Salida de transmisión UART. Las líneas UART son eléctricamente idénticas a las líneas DIO.
DGND	N/A	N/A	Referencia para las señales digitales, +5V, y + 3.3V.

Tabla 4.1: Descripción de las Señales en los Conectores A y B del MXP.

La siguiente Figura 4.3 y Tabla 4.2 muestran las señales en el conector C del Puerto del Mini Sistema (MSP).

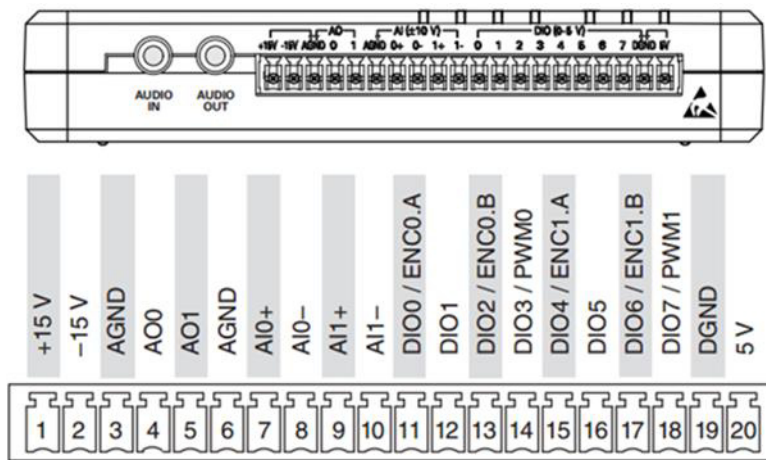


Figura 4.3: Señales Primarias/Secundarias en el Conector C del MSP.

Nombre de la Señal	Referencia	Dirección	Descripción
+15 V/−15 V	AGND	Salida	Salida de alimentación +15 V/−15 V.
AI0 +/AI0 − ; AI1 +/AI1 −	AGND	Entrada	Canales de entrada analógica diferencial, ±10 V.
AO < 0..1 >	AGND	Salida	Canales de salida analógica de una sola terminal, ±10 V, referenciados.
AGND	N/A	N/A	Referencia para la entrada y salida analógica y la salida de alimentación +15 V/−15 V.
+5 V	DGND	Salida	Salida de alimentación + 5 V.
DIO < 0..7 >	DGND	Entrada o Salida	Lineas digitales de propósito general con 3.3 V de salida, 3.3 V/5 V de entrada compatible.

DGND	N/A	N/A	Referencia para las líneas digitales y la salida de alimentación + 5V.
------	-----	-----	--

Tabla 4.2: Descripción de las Señales en el Conector C del MSP.

- **ESPECIFICACIONES TÉCNICAS.**

Las siguientes especificaciones son típicas para el rango de temperatura de funcionamiento 0 a 40 para el NI myRIO – 1900.

Procesador	
Tipo	Xilinx Z – 7010
Velocidad	667 MHz
Núcleos	2

Memoria	
Memoria No Volátil	256 MB
Memoria de la DDR3	512 MB
Frecuencia del Reloj de la DDR3	533 MHz
Ancho de Bus de Datos de la DDR3	16 bits

Entrada Analógica	
Taza de Muestreo Global	500 kS/s
Resolución	12 bits
Protección de Sobretensión	± 16 V
Conectores MXP	
Configuración	Cuatro canales de una sola terminal por conexión
Impedancia de Entrada	> 500 k Ω adquiriendo a 500 kS/s 1 M Ω encendido e inactivo 4.7 k Ω apagado
Impedancia de la Fuente Recomendada	3 k Ω o menor
Rango Nominal	0 V a +5 V
Precisión Absoluta	± 50 mV
Ancho de Banda	> 300 kHz

Conector MSP

Configuración	Dos canales diferenciales
Impedancia de Entrada	Hasta 100 nA de fuga encendido; 4.7 k Ω apagado
Rango Nominal	± 10 V
Voltaje de Trabajo (Señal + Modo Común)	± 10 V de AGND
Precisión Absoluta	± 200 mV
Ancho de Banda	20 kHz mínimo, > 50 kHz típicamente

Salida Analógica

Tazas Máximas de Actualización en Conjunto	500 kS/s
Todos los Canales de salidas analógicas en los conectores MXP	345 kS/s
Todos los canales de salidas analógicas en el conector MSP	345 kS/s
Resolución	12 bits
Protección de Sobrecarga	± 16 V
Voltaje de Inicio	0 V después de la inicialización del FPGA
Conectores MXP	
Configuración	Dos canales de una sola terminal por conexión
Rango	0 V a +5 V
Precisión Absoluta	50 mV
Corriente del Controlador	3 mA
Tasa de Cambio en el Voltaje	0.3 V/ μ s
Conector MSP	
Configuración	Dos canales de una sola terminal

Rango	± 10 V
Precisión Absoluta	± 200 mV
Corriente del Controlador	2 mA
Tasa de Cambio en el Voltaje	2 V/ μ s

Tabla 4.3: Especificaciones Técnicas del Hardware NI myRIO-1900.

4.1.1.2 ARREGLOS DE COMPUERTAS PROGRAMABLES EN CAMPO (FPGA)

Los arreglos de compuertas programables en campo (FPGAs) son chips de silicio reprogramables. Al utilizar bloques de lógica pre-construidos y recursos para ruteo programables, estos chips se configuran para implementar funcionalidades personalizadas en hardware. Ross Freeman, el cofundador de Xilinx, inventó el primer FPGA en 1985. El gran auge del chip FPGA esta guiado por el hecho que los FPGAs combinan lo mejor de los circuitos integrados de aplicación específica (ASICs) y sistemas basados en procesador. Los FPGAs ofrecen velocidades temporizadas por hardware y fiabilidad, pero sin requerir altos volúmenes de recursos para compensar el gran gasto que genera un diseño personalizado de ASIC [11].

El silicio reprogramable también tiene la misma flexibilidad que un software que se ejecuta en un sistema basado en procesador, pero no está limitado por el número de núcleos de procesamiento disponibles. A diferencia de los procesadores, los FPGAs son verdaderamente paralelos por naturaleza, así las diferentes operaciones de procesamiento no tienen que competir por los mismos recursos. Cada tarea de procesamiento independiente es asignada a una sección del chip y puede ejecutarse de manera autónoma sin ser afectada por otros bloques de lógica. Como resultado, el rendimiento de una parte de la aplicación no se ve afectado cuando se agregan otros procesos.

Las FPGAs se utilizan en aplicaciones similares a los ASICs sin embargo son más lentas, tienen un mayor consumo de potencia y no pueden abarcar sistemas tan complejos como ellos.

Las especificaciones de un chip FPGA generalmente se dividen en bloques de lógica configurables como segmentos o células de lógica, funciones DSP, y recursos de memoria como RAM en bloque. El chip FPGA tiene otros componentes, pero éstos son generalmente los más importantes cuando se seleccionan y comparan FPGAs para una aplicación en particular. Las especificaciones del chip FPGA Xilinx Z-7010 integrado en el NI myRIO-1900 se muestran en la Tabla 4.4.

FPGA Xilinx Z-7010	
Bloques de Lógica Configurables	28K
Flip-Flops	35200
Tablas de Consulta (LUTs)	17600
RAM en Bloque	2.1 Mb
Número de Bloques de 36 Kb	60
Segmentos DSP (18 × 25 MACCs)	80

Tabla 4.4: Especificaciones Técnicas del FPGA Xilinx Z-7010.

4.1.2 SOFTWARE DE DESARROLLO DE SISTEMAS NI LABVIEW

LabVIEW (Laboratory Virtual Instrumentation Engineering Workbench) es una plataforma y entorno de desarrollo para diseñar sistemas, con un lenguaje de programación visual gráfico. Recomendado para sistemas hardware y software de pruebas, control y diseño, simulado o real y embebido, pues acelera la productividad. Con un lenguaje de programación gráfico nativo, IP integrado para análisis de datos y procesamiento de señales y una arquitectura abierta que permite la integración de

cualquier dispositivo de hardware y cualquier enfoque de software, LabVIEW brinda una amplia gama de soluciones para la implementación de sistemas.

El lenguaje de programación de LabVIEW es uno de alto nivel al abstraer la complejidad de bajo nivel, esto puede significar una pérdida en el rendimiento de las aplicaciones pero genera un aumento en la productividad al no tener que lidiar con un diseño demasiado complicado. También integra la mayoría de las funciones en un solo entorno de desarrollo unificado, a diferencia de cualquier otra alternativa basada en programación convencional. Programar en un entorno unificado significa que no se requiere tener experiencia en una variedad de herramientas para lograr obtener una solución a un problema con varios requerimientos.

Algunos de los elementos más fundamentales de la plataforma de programación de LabVIEW se describen a continuación:

- **Instrumento Virtual (VI).**- Es el programa desarrollado en LABVIEW y su nombre se debe a que su apariencia y funcionamiento imita al de un instrumento real, sin embargo, es análogo a las funciones creadas con los lenguajes de programación convencionales. Los VIs tienen una parte interactiva con el usuario (Panel Frontal) y otra parte de código fuente (Diagrama de Bloques), y aceptan parámetros procedentes de otros VIs.
- **Panel Frontal.**- Es la interfaz gráfica del VI con el usuario. Esta interfaz recoge las entradas procedentes del usuario y representa las salidas proporcionadas por el programa. Un panel frontal está formado por controles e indicadores. Los primeros sirven para introducir parámetros al VI, mientras que los indicadores se emplean para mostrar los resultados producidos, ya sean datos adquiridos o resultados de alguna operación.
- **Diagrama de Bloques.**- Constituye el código fuente del VI. En el diagrama de bloques es donde se realiza la implementación del programa del VI para controlar o realizar cualquier procesamiento de las entradas y salidas que se materializan como terminales al colocar controles e indicadores en el panel frontal y viceversa. El diagrama de bloques incluye funciones y estructuras integradas en las librerías que incorpora LabVIEW. En el lenguaje gráfico las funciones y las estructuras son

nodos elementales. Son análogas a los operadores o librerías de funciones de los lenguajes convencionales. El diagrama de bloques se construye conectando los distintos objetos entre sí, como si de un circuito se tratara.

Los principales objetos que se encuentran en el diagrama de bloques son:

- **Nodos.-** son o son objetos en el diagrama de bloques que tienen entradas y/o salidas y realizan operaciones cuando el VI se ejecuta. Son análogos a instrucciones, operaciones, funciones y sub rutinas en lenguajes de programación convencionales. Los nodos pueden ser:
 - **Funciones.-** son los elementos de operación fundamentales de LabVIEW. Las funciones no tienen ventanas del panel frontal o ventanas del diagrama de bloques pero tienen terminales de conexión. Una función tiene un fondo amarillo pálido en su ícono.
 - **Estructuras.-** son similares a las declaraciones causales y a los bucles en lenguajes convencionales, ejecutan el código que contienen de forma condicional o repetitiva. Las estructuras son elementos de control de procesos, como Estructuras de Casos, Ciclos For o Ciclos While.
 - **SubVIs.-** son VIs llamados desde el diagrama de bloques de otro VI. Un nodo de subVI corresponde a una llamada de subrutina en lenguajes de programación convencionales.
- **Terminales.-** Las terminales son puertos de entrada y salida que intercambian información entre el panel frontal y diagrama de bloques. Son análogos a parámetros y constantes en los lenguajes de programación convencionales. Los tipos de terminales incluyen terminales de control o indicador y terminales de nodo.
- **Cables.-** Son las trayectorias que siguen los datos desde su origen hasta su destino, ya sea una función, una estructura, un terminal, etc. Cada cable tiene un color o un estilo diferente, lo que diferencia unos tipos de datos de otros.

4.1.2.1 MÓDULO LABVIEW FPGA

El Módulo NI LabVIEW FPGA extiende la plataforma de desarrollo gráfico de LabVIEW para programar FPGAs en hardware de E/S reconfigurables (RIO) de NI. LabVIEW FPGA brinda herramientas para diseñar de manera más eficiente y efectiva sistemas complejos al proporcionar un entorno de desarrollo altamente integrado, un gran ecosistema de bibliotecas de IP (propiedad intelectual), un simulador de alta fidelidad y características de depuración [11].

Atraves del módulo de LabVIEW FPGA, es posible implementar una amplia variedad de rutinas de adquisición y procesamiento de datos que se ejecutan dentro de la FPGA de los dispositivos de hardware. La ejecución del hardware proporciona un mayor desempeño y determinismo que la mayoría de las soluciones de software basadas en procesador. Una vez que el código se compila y se ejecuta en el FPGA se ejecutará sin la fluctuación asociada a la ejecución de software y la priorización típica de los sistemas operativos, e incluso en un grado mucho menor que en los sistemas operativos en tiempo real.

LabVIEW FPGA ofrece un enfoque de programación gráfica que simplifica la tarea de conectar a E/S y comunicar datos, mejorando enormemente la productividad del diseño y reduciendo el tiempo de realización.

Para el diseño de los algoritmos de control a implementar en las posteriores pruebas de laboratorio, se pueden establecer cuatro elementos básicos para su desarrollo.



Figura 4.4: Elementos Básicos de LabVIEW FPGA.

4.1.2.1.1 INTERFAZ DE ENTRADA/SALIDA

Las entradas y salidas (E/S) en la FPGA permiten la conexión a otros dispositivos y se utilizan para transferir datos entre las diferentes partes del sistema. En el entorno de programación de LabVIEW las E/S del FPGA son nodos dentro del FPGA que se conectan a la parte de la FPGA diseñada por National Instruments con la parte de la FPGA a diseñar a través del VI de la FPGA.

Los recursos dentro del FPGA traducen cantidades físicas a un valor digital que se manipula en el software del Módulo FPGA. Una E/S tiene uno o más terminales para recibir o generar una cantidad física. Muchas de las E/S en la FPGA tienen terminales físicas que se pueden conectar directamente a los elementos del sistema.

Cuando el IV FPGA se ejecuta dentro del myRIO, este desempeña operaciones E/S en el hardware. Debido a que el VI se ejecuta en el FPGA, el IV puede reaccionar a la entrada con la velocidad y el determinismo disponible en el hardware del FPGA.

Es posible poner las entradas y salidas analógicas y digitales en un mismo nodo en el diagrama de bloques. También es posible utilizar propiedades y métodos específicos en los elementos de E/S con las funciones FPGA I/O Property Node y FPGA I/O Method Node, respectivamente.

- **ENTRADA ANALÓGICAS.**

Al configurar el FPGA I/O Node para leer una entrada analógica, el FPGA I/O Node inicia una conversión, espera por el resultado y entonces devuelve la representación binaria del voltaje como un entero con signo o número de punto fijo. El proceso de entrada analógica y el tipo de datos resultante varía según la FPGA. En la FPGA del myRIO, se crea el VI para utilizar los datos devueltos por la entrada analógica del FPGA I/O Node y realizar operaciones dentro del VI.

- **SALIDAS ANALÓGICAS.**

Al configurar el FPGA I/O Node para escribir una salida analógica, el FPGA I/O Node escribe la representación binaria del voltaje al convertidor digital-analógico (DAC), que establece el voltaje de salida analógica. El tipo de datos varía según la FPGA. Al recibir información de voltaje de VI externos u ubicados en otro nivel de hardware, el VI externo convierte el voltaje a una representación binaria apropiada antes de escribir el valor en el VI del FPGA.

4.1.2.1.2 COMUNICACIÓN DE DATOS

LabVIEW dispone de varios métodos de comunicación de datos, cada uno adaptado para un determinado caso de uso. La mayoría de ellas se pueden clasificar por establecer interfaces de comunicación de buffer o de variables.

- **Interfaces de Buffer.-** Las interfaces de memoria de almacenamiento temporal (buffer) se utilizan para mandar datos de un lugar a otro y no se desea sobrescribir o perder algún valor. Una interfaz de buffer se representa a menudo como un FIFO simple o un buffer multi-elemento donde uno o más elementos de escritura añaden datos en un extremo del buffer y un elemento de lectura recupera datos desde el otro extremo.
- **Interfaces de Variables.-** Las interfaces de variables se utilizan cuando se debe almacenar un valor en la memoria que los elementos de lectura y escritura puedan acceder con un mínimo o nulo control de flujo. En estos casos el último valor es de interés primordial y el sobrescribir los valores anteriores es una condición aceptable.

Para la visualización de los resultados de los experimentos de prueba es necesario que se realice en un nivel superior de la FPGA en la estructura de hardware, ya que la FPGA no cuenta con funciones de visualización de datos. Por lo que es necesario examinar las opciones para la transferencia y almacenamiento de datos con las que cuenta el Modulo FPGA.

Tres de las formas más comunes para almacenar y transferir datos en una FPGA son utilizando:

- **FIFOs.**- para almacenar múltiples muestras de datos secuenciales.
- **Elementos de Memoria.**- para acceder a los datos sin importar la secuencia en la que los datos se escriben en la memoria.
- **Registros FPGA.**- para mantener sólo una unidad del tamaño de los datos especificados en un momento.

La mejor forma para adquirir y mostrar los datos al terminar la ejecución en tiempo real de los experimentos es utilizando los FIFOs.

Un FIFO (First In, First Out) es una estructura de datos que contiene los elementos en el orden en que se reciben y proporciona acceso a esos elementos utilizando la regla del primero en entrar es el primero en salir.

La Figura 4.5 muestra el comportamiento de los elementos moviéndose a través de un FIFO.

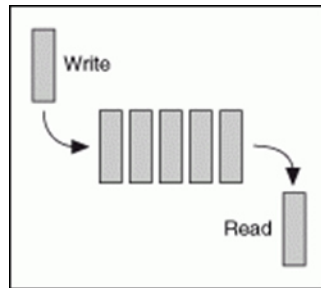


Figura 4.5: Transferencia de Datos a través de un FIFO.

Existen diferentes tipos de FIFOs que se pueden implementar dependiendo de la forma que se necesite transferir los datos. Para transferir grandes cantidades de datos entre la computadora y el FPGA es recomendable utilizar el FIFO de Acceso Directo a Memoria (DMA).

Este tipo de FIFO accede directamente a la memoria para transferir datos del VI FPGA al VI de la computadora y viceversa. Un DMA FIFO asigna memoria tanto en la computadora como en el FPGA, sin embargo, actúa como un único FIFO. Los FIFOs DMA proporcionan ventajas de rendimiento sobre el uso de controles e indicadores del panel frontal para la comunicación entre la computadora y el FPGA.

Otras ventajas que proporciona la comunicación DMA son:

- Una separación eficaz de las tareas de procesamiento entre la FPGA y el procesador de la computadora, además de que el procesador de la computadora queda libre para realizar operaciones durante la transferencia de datos.
- Ahorro de recursos FPGA al transferir matrices de datos.
- Sincronización automática de las transferencias de datos entre la computadora y el FPGA.

Un DMA FIFO asigna memoria tanto en la computadora como en el objetivo FPGA, pero actúa como un único FIFO. El VI FPGA escribe en el FIFO un elemento a la vez con el método de escritura del bloque FIFO Method Node o lee desde el FIFO un elemento a la vez con el método de lectura. El VI de la computadora lee o escribe en el FIFO uno o más elementos a la vez con la función Invoke Method.

- **TRASFERENCIA DE DATOS UTILIZANDO CONTROLES E INDICADORES DEL PANEL FRONTAL.**

La función Read/Write Control se utiliza en un VI que se ejecuta en un nivel de superior de la estructura de hardware para acceder a los controles e indicadores del panel frontal del VI FPGA, como se muestra en la Figura 4.6.

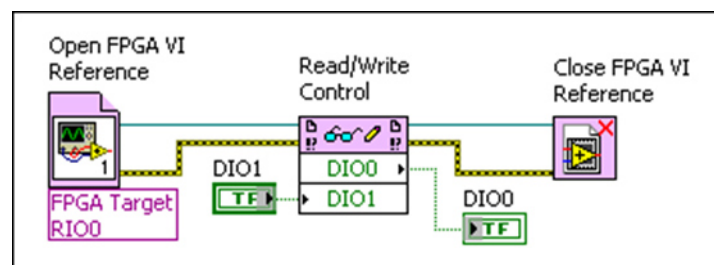


Figura 4.6: Diagrama de Bloques Básico para Transferencia.

Una ventaja de la comunicación a través de la programación del panel frontal en relación con otros métodos de transferencia de datos entre el FPGA y la computadora es su bajo costo operativo. Aunque no se puede alcanzar un alto rendimiento con la comunicación por programación del panel frontal, cada llamado a la función Read/Write Control inicia la transferencia de datos con el menor retraso

posible. Por lo tanto, la comunicación por programación del panel frontal es ideal para las transferencias de datos pequeños y frecuentes.

Una desventaja de la comunicación por programación del panel frontal es que este método transfiere sólo los datos más actuales almacenados en el control o indicador del VI del FPGA. Por ejemplo, los datos podrían perderse si el VI del FPGA escribe datos en un indicador más rápido que el VI de la computadora puede leerlos. Además, cada control o indicador en el VI del FPGA utiliza los recursos del FPGA. Mejores prácticas en la programación del FPGA recomiendan limitar el número de objetos en el panel frontal del VI del FPGA. Finalmente, la transferencia de datos entre el VI del FPGA y el VI de la computadora usando controles e indicadores del panel frontal requiere el uso del procesador de la computadora. Como resultado, la velocidad de transferencia de datos es altamente dependiente de la velocidad y la disponibilidad del procesador de la computadora. Un procesador lento o una falta de disponibilidad resultan en una lenta transferencia de datos desde el FPGA hacia la computadora.

4.1.2.1.3 TEMPORIZACIÓN

Las aplicaciones a menudo requieren que las E/S se ejecuten a una frecuencia específica. Por ejemplo, los algoritmos usados en los lazos de control requieren normalmente que las entradas sean muestreadas a una velocidad conocida. Para esto se emplea el Loop Timer Express VI (Temporizador de Lazo) en un Ciclo While para controlar la velocidad de ejecución de las E/S, como se muestra en el siguiente diagrama.

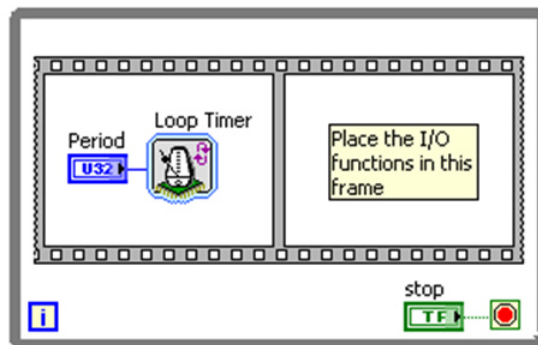


Figura 4.7: Diagrama de Bloques Básico para Temporización.

Para usar el Loop Timer Express VI para controlar la velocidad de ejecución de las E/S, se coloca una estructura secuencial dentro de un Ciclo While. El Loop Timer Express VI se coloca en el primer segmento de la estructura secuencial. Se configura las Unidades del Contador y el Tamaño del Contador interno en el cuadro de diálogo Configure Loop Timer que aparece. El código LabVIEW para las E/S se coloca en los segmentos subsiguientes de la estructura secuencial.

La primera llamada del Loop Timer Express VI no da lugar a ninguna espera o retraso, ya que establece una referencia de tiempo para las llamadas posteriores. Después de la primera llamada del Loop Timer Express VI, las subsiguientes llamadas no ocurren hasta que haya transcurrido el tiempo especificado por el parámetro Count (Conteo) desde la llamada anterior. Si el tiempo especificado por el parámetro Count es menor que el tiempo que tarda el FPGA en ejecutar el código en el Ciclo While, el Loop Timer Express VI no afecta la temporización del Ciclo While.

El tiempo de ejecución se selecciona 1ms para los segmentos de código diseñados para los algoritmos de control en general; ya que permite un margen de tiempo suficiente para la ejecución del código y facilitan el cálculo e implementación de las funciones de transferencia discretas.

4.1.2.1.4 CONTROL

El módulo LabVIEW FPGA dispone de varias funciones y subVIs para la realización de aplicaciones de control. Aunque su utilización puede facilitar la implementación de las funciones de transferencia discretas necesarias, no es complicado realizar las funciones utilizando elementos básicos como sumadores, multiplicadores y funciones de retardo, para obtener un mayor desempeño en la ejecución de la aplicación.

Para mejorar u optimizar una aplicación de control comúnmente se busca disminuir la latencia. La latencia es simplemente el retardo entre dos puntos en la ejecución de código. Normalmente se asocia con las aplicaciones de control debido a que el desempeño del sistema depende del tiempo de retraso entre una medición de la señal de entrada y la correspondiente señal de salida de control después de que algún

procesamiento ha sido aplicado a los datos de entrada. La disminución de la latencia proporciona un control más estricto sobre el proceso.

- **DATOS DE PUNTO FIJO.**

La mayoría de las plataformas FPGA no son compatibles de forma nativa al procesamiento de punto flotante; por lo que, las implementaciones de punto fijo a menudo son necesarias para lograr la precisión y rendimiento requerido del algoritmo. Punto Fijo es un formato para la representación de números en los dispositivos de procesamiento digital. Es un tipo de dato utilizado por un lenguaje de programación o lenguaje de descripción de hardware (HDL) para determinar cómo interpretar los bits en una ubicación de memoria.

Los datos de punto fijo se utilizan para representar un conjunto de números racionales utilizando dígitos binarios, o bits. A diferencia de los datos de punto flotante, donde la precisión y el rango de los datos varía, se pueden configurar los números de punto fijo para que utilicen siempre un número específico de bits para la parte entera y fraccionaria.

Para esto, se debe especificar la codificación binaria de un número de punto fijo cuando se desea que el número se ajuste a un determinado tamaño de bits.

- **Signo o Sin signo.-** esta opción determina si el dato de punto fijo tiene signo o no. Si se selecciona con signo, el bit del signo es siempre el primer bit de la cadena de bits que representa los datos.
- **Tamaño de palabra.-** es el número total de bits en la cadena de bits que LabVIEW utiliza para representar todos los valores posibles de los datos de punto fijo. LabVIEW acepta un máximo de tamaño de palabra de 64 bits.
- **Tamaño de entero de palabra.-** es el número de bits que corresponden a la parte entera en la cadena de bits que LabVIEW utiliza para representar todos los valores posibles de los datos de punto fijo, o dada una posición inicial a la izquierda o a la derecha del bit más significativo, el número de bits a desplazar para que el punto binario alcance el bit más significativo. El tamaño de entero de

palabra puede ser mayor que el tamaño de palabra, y puede ser positivo o negativo.

El Módulo LabVIEW FPGA incluye en la mayoría de sus funciones el trabajar con datos de punto fijo. Algunas de estas funciones utilizan específicamente este tipo de datos para lograr un alto rendimiento en su ejecución.

- **USO DE LAS FUNCIONES MATEMÁTICAS DE ALTO RENDIMIENTO.**

El módulo LabVIEW FPGA dispone de un conjunto de funciones matemáticas de alto rendimiento, que se implementan con números de punto fijo. Estas funciones son similares a las funciones numéricas, pero cuentan con mayores tasas de rendimiento, registros de entrada/salida, y segmentación automática.

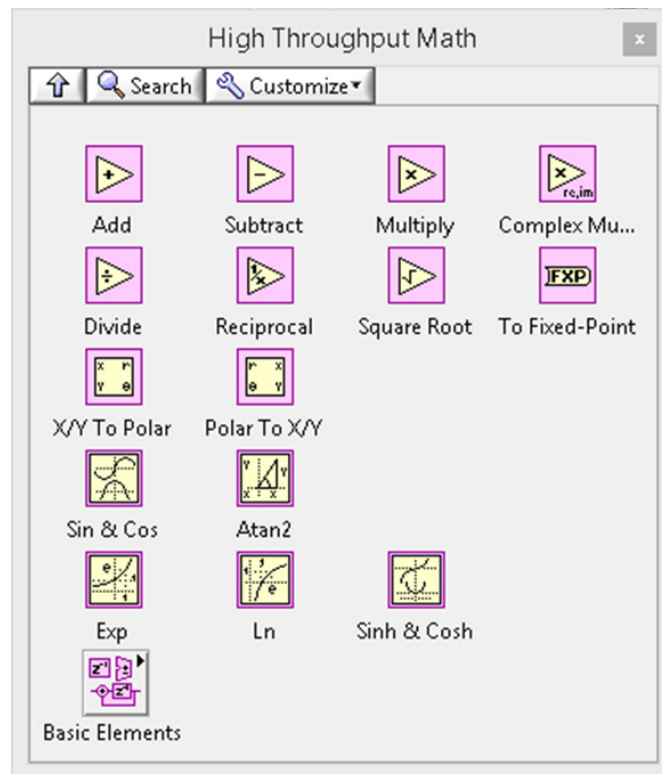


Figura 4.8: Funciones Matemáticas de Alto Rendimiento.

Como las funciones matemáticas de alto rendimiento trabajan con números de punto fijo, es necesario configurar el tamaño de palabra y el tamaño de entero de

palabra de las terminales de entrada, para evitar problemas con las configuraciones de punto fijo no compatibles.

- **REALIZACIÓN DE FUNCIONES DE TRASFERENCIA DISCRETAS UTILIZANDO FUNCIONES BÁSICAS.**

Las FPGAs modernas ofrecen considerables recursos para la aplicación de algoritmos de procesamiento de señales digitales en tiempo real (DSP), y el módulo FPGA de LabVIEW, con su entorno de desarrollo gráfico, ofrece ventajas significativas para el diseño FPGA basado en DSP sobre otros flujos de diseño.

En general, la “realización” de funciones de transferencia discretas significa determinar la configuración física para la combinación apropiada de operaciones aritméticas y de almacenamiento. Las funciones de transferencia discretas se construyen a base de sumadores, multiplicadores y elementos de retardo en un diagrama de bloques convencional. Gracias a que el entorno de programación de LabVIEW utiliza una sintaxis gráfica de diagrama de bloques, es sencillo visualizar la estructura del código al realizar funciones de transferencia discretas.

Diferentes métodos de programación para la realización de funciones de transferencia pueden ser planteados. Los más usuales son los que obtienen la realización del numerador y el denominador de la función de transferencia discreta mediante conjuntos de elementos de retraso por separado. De esta manera el número de elementos de retraso total que se utilizan es la suma del número de elementos de retraso que se utilizan para la realización de la función del numerador y denominador.

El método de programación más utilizado en la práctica es la programación estándar, ya que utiliza el número mínimo posible de elementos de retraso, ofreciendo una gran eficiencia en la ejecución del código. La configuración del diagrama de bloques de la programación estándar permite el utilizar solo el número de elementos de retraso más grande necesario para la realización de las funciones del numerador y denominador, como se muestra en la Figura 4.9.

$$\frac{Y(z)}{U(z)} = \frac{b_0 + b_1z^{-1} + b_2z^{-2} + b_3z^{-3} + b_4z^{-4}}{1 + a_1z^{-1} + a_2z^{-2} + a_3z^{-3} + a_4z^{-4}} \quad (4.1)$$

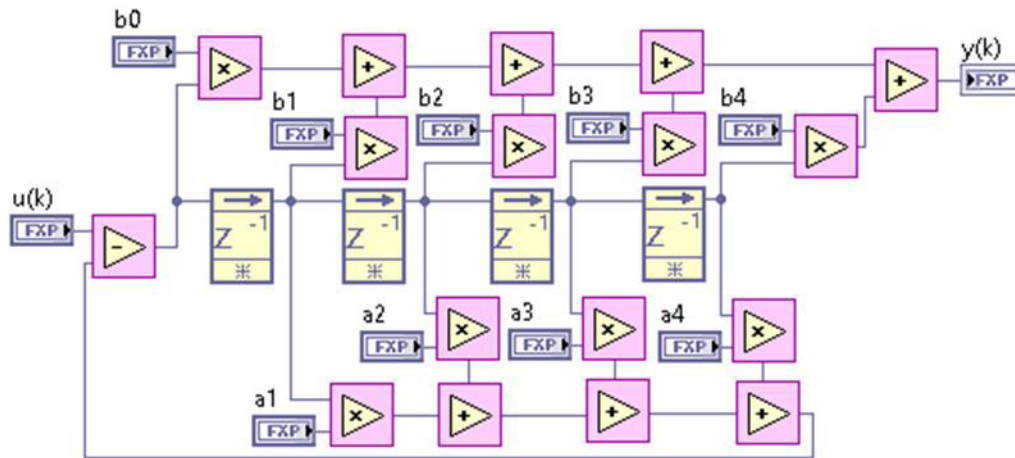


Figura 4.9: Programación Estándar.

4.1.2.2 MÓDULO LABVIEW REAL-TIME

El Módulo LabVIEW Real-Time es un componente adicional para el sistema de desarrollo LabVIEW que se puede utilizar para crear y depurar aplicaciones determinísticas y confiables, que se ejecutan en un Sistema Operativo en Tiempo Real (RTOS). Para la realización de las pruebas; el almacenamiento y visualización de los datos es la principal función que se desempeña en este nivel [11].

- **ASIGNACIÓN DE PARÁMETROS DE INICIO.**

Algunos de los parámetros en la programación del FPGA deben ser establecidos para controlar la ejecución de la aplicación. Ya que solo es necesario transferir los valores de los parámetros al VI FPGA una sola vez al comienzo, la función Read/Write Control es la opción indicada para hacerlo.

- **ADQUISICIÓN DE DATOS.**

Como se mencionó en la descripción de los elementos de la FPGA, el DMA FIFO es la mejor forma para transferir gran cantidad de datos desde el VI FPGA a

un VI anfitrión en un nivel superior en la estructura de hardware. Al implementar el DMA FIFO y adquirir datos en la FPGA, es necesario utilizar la función Invoke Method para disponer de los datos en el VI anfitrión.

- **Invoke Method.-** Invoca un método de interfaz FPGA o dicho de otra forma, establece una forma de interacción de un VI en un nivel superior con el VI FPGA. Dependiendo del FPGA, la función Invoke Method dispone de varios métodos de interfaz FPGA. El método FIFO.Read, lee los elementos de la DMA FIFO desde la parte de memoria del FIFO ubicada en el nivel del VI anfitrión y devuelve los datos cuando el número de elementos esté disponible.
- **CONVERSIÓN DE UNIDADES.**

Como se mencionó en la descripción de las entradas analógicas, el FPGA I/O Node devuelve la representación binaria del voltaje de entrada como un entero con signo. Para el procesamiento de datos realizado en la FPGA no es necesaria la conversión de datos, debido a que la escala se mantiene para los valores de las salidas analógicas. Sin embargo es poco práctico mostrar los resultados de las pruebas utilizando estos tipos de datos. A continuación se muestra las ecuaciones a utilizar para convertir datos sin procesar a valores de voltaje [12].

$$V = \text{Valor de Dato sin Procesar} * \text{Peso del LSB}$$

$$\text{Peso del LSB} = \text{Rango Nominal} / 2^{\text{Resolución del CAD}}$$

donde

Valor de Dato sin Procesar.- es el valor devuelto por el FPGA I/O Node.

Peso del LSB (Bit Menos Significativo).- es el valor en volts del incremento entre los valores de datos.

Rango Nominal.- es el valor absoluto en volts del rango nominal completo (pico a pico) del canal.

Resolución del CAD.- es la resolución del CAD en bits. (Resolución del CAD = 12)

Para canales de Entrada y Salida Analógica en los conectores del MSP,

$$\text{Peso del LSB} = 20 \text{ V} / 2^{12} = 4.883 \text{ mV}$$

$$\text{Lectura Positiva Maxima} = +2047 * 4.883 \text{ mV} = 9.995 \text{ V}$$

$$\text{Lectura Negativa Maxima} = -2048 * 4.883 \text{ mV} = -10.000 \text{ V}$$

4.2 DESCRIPCIÓN DEL SISTEMA

El sistema es un laboratorio de pruebas comercial distribuido por la empresa alemana amira (Figura 4.10). Cuenta con la realización técnica de un sistema no lineal de una entrada y una salida (SISO) con actuadores, sensores y mediciones de salida apropiadas y la posibilidad de conectar diferentes controladores.



Figura 4.10: Laboratorio Experimental – amira – DR300 – Control de Velocidad con Carga Variable.

La planta está representada por un motor de CD de excitación permanente (M1) donde la señal de entrada (corriente de armadura) es provista por un lazo de control de corriente. Los sensores para la señal de salida (velocidad) son un tacogenerador (T) y un encoder incremental (I). El extremo libre del eje del motor está firmemente acoplado (K) al eje de un segundo motor idéntico (M2). Cada motor está montado en la parte superior de un perfil de aluminio. El sistema está listo para funcionar y está conectado a la electrónica apropiada usando sólo un conector.

En la versión estándar el sistema DR300 consiste del sistema mecánico y una caja de 19 pulgadas con diferentes módulos de fuentes de alimentación, servoamplificadores operando como controladores de corriente, unidades de adaptación de señal y un módulo para la medición de las salidas (Figura 4.11). También cuenta con la opción de utilizar un control PI analógico ya integrado en la caja de módulos o de conectar un controlador externo.

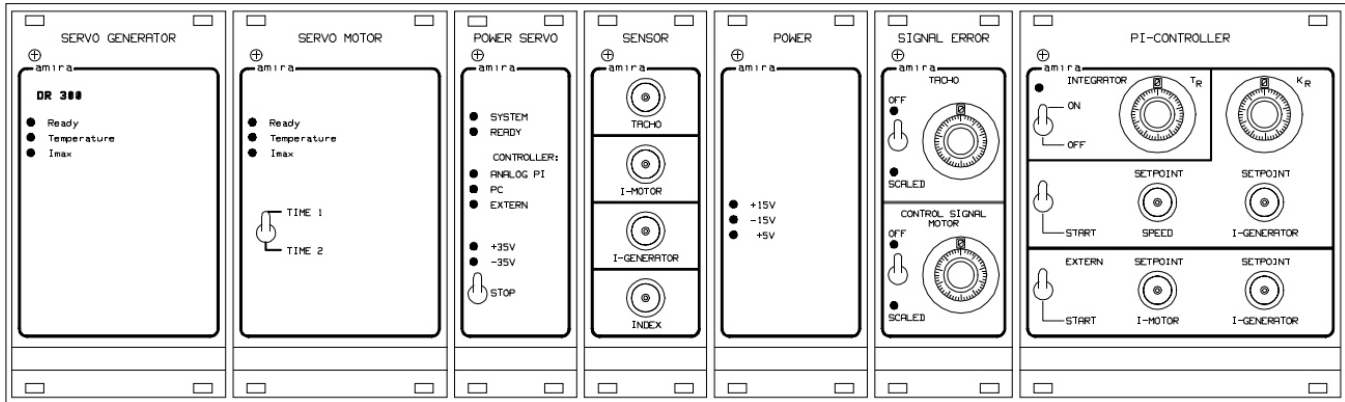


Figura 4.11: Caja de Módulos.

La planta está representada por un motor CD de excitación permanente donde la señal de entrada ($I_r(t)$) es provista por un lazo de control de corriente. Los sensores para las señales de salida son un tacómetro ($\omega_m(t)$) y un encoder incremental ($i_a(t)$). El eje del motor está acoplado fijamente al eje de un segundo motor que es usado como generador, cuyo comportamiento es aproximadamente como el de un elemento proporcional ($M_L(t)$).

4.2.1 DESCRIPCIÓN MATEMÁTICA DEL SISTEMA

Aunque en el manual del equipo se describe al sistema, como un sistema SISO con una entrada de perturbación controlada y una salida de corriente de armadura disponible, el sistema puede adoptar la forma de un sistema MIMO con dos entradas y dos salidas, del cual es necesario identificar su modelo matemático.

4.2.1.1 MODELO DEL MOTOR DE CORRIENTE DIRECTA

El motor de corriente directa convierte la energía eléctrica en energía mecánica. En la Figura 4.12 se muestra una descripción esquemática del motor de CD, el cual consiste de un parte eléctrica y una parte mecánica [13].

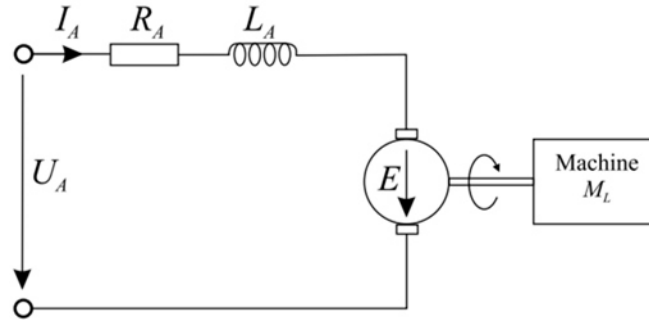


Figura 4.12: Diagrama del Circuito Eléctrico del Motor.

Al determinar la terminal de voltaje \$U_A(t)\$ como entrada, la carga \$M_L(t)\$ como perturbación y la velocidad del eje del motor \$\omega(t)\$ como salida, se obtiene el diagrama de bloques de la Figura 4.13.

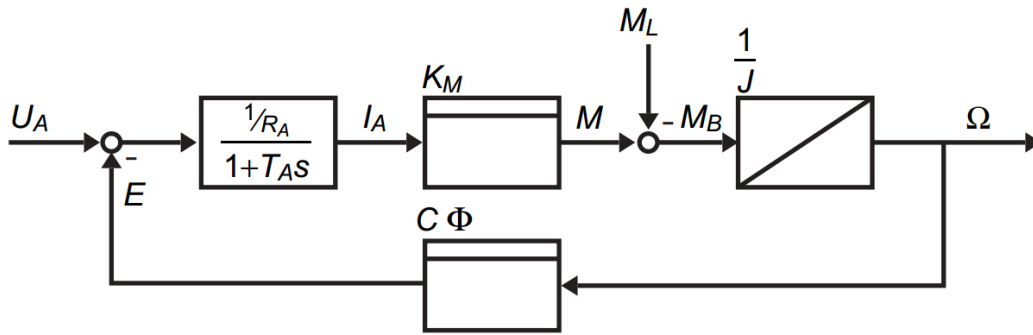


Figura 4.13: Diagrama de Bloques del Motor de CD.

En donde, si se define la corriente de lazo \$I_A(t)\$ y la velocidad del motor \$\omega\$ como variables de estado, se tendrá la siguiente descripción en espacio de estados.

$$\begin{bmatrix} \dot{I}_A(t) \\ \dot{\Omega}(t) \end{bmatrix} = \begin{bmatrix} -\frac{R_A}{L_A} & -\frac{C\Phi}{L_A} \\ \frac{K_M}{J} & 0 \end{bmatrix} \begin{bmatrix} I_A(t) \\ \Omega(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L_A} \\ 0 \end{bmatrix} U_A(t) + \begin{bmatrix} 0 \\ -\frac{1}{J} \end{bmatrix} M_L(t) \quad (4.2)$$

así como su función de transferencia

$$\Omega(s) = \frac{1}{C\Phi \left(1 + \frac{J R_A}{K_M C\Phi} s + \frac{J T_A R_A}{K_M C\Phi} s^2\right)} U_A(s) - \frac{R_A (1 + T_A s)}{K_M C\Phi \left(1 + \frac{J R_A}{K_M C\Phi} s + \frac{J T_A R_A}{K_M C\Phi} s^2\right)} M_L(s) \quad (4.3)$$

donde

$T_A = \frac{L_A}{R_A}$, es la constante de tiempo de la armadura

$T_M = \frac{J R_A}{K_M C\Phi}$, es la contante de tiempo del motor

Y los valores de los parámetros están resumidos en la Tabla 4.4.

Parámetro	Símbolo	Valor	Unidad
Inercia Total	J	$80.45 \cdot 10^{-6}$	$\text{kg} \cdot \text{m}^2$
Constante de Voltaje	$C\Phi$	$6.27 \cdot 10^{-3}$	V/Rpm
Constante del Motor	K_M	0.06	Nm/A
Inductancia de Armadura	L_A	0.003	H
Resistencia	R_A	3.13	Ohm
Salida de Voltaje del Tacómetro	K_T	$2.5 \cdot 10^{-3}$	V/Rpm
Constante de Tiempo del Filtro del Tacómetro	T_T	5	ms

Tabla 4.4: Especificaciones Técnicas del FPGA Xilinx Z-7010.

4.2.1.2 CONFIGURACIÓN DEL LAZO DE CONTROL DE CORRIENTE

La planta del sistema está compuesta del motor de CD con un control de corriente en cascada. Como se muestra en la Figura 4.14.

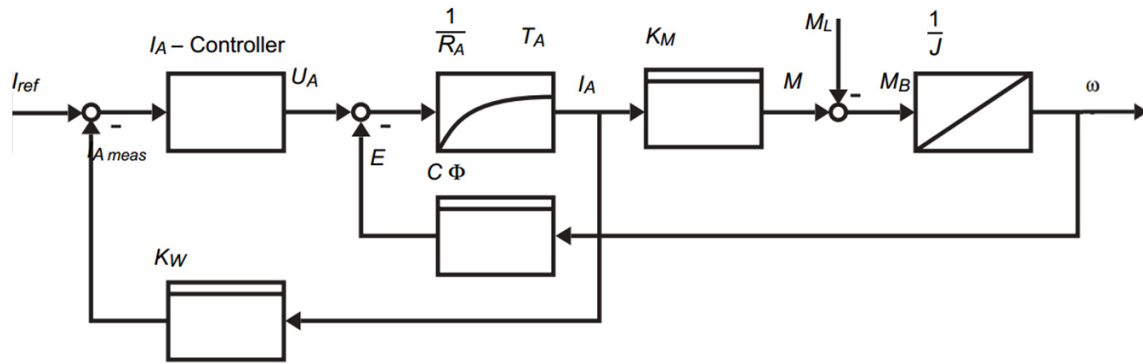


Figura 4.14: Lazo de Control de Corriente.

El diagrama anterior muestra que la variable de control $I_A(t)$ esta acoplada dentro del lazo de retroalimentación de $\omega(t)$, por lo que está siendo afectada por $M_L(t)$. Para separar los dos lazos se asume que el momento de carga no cambia ($M_L = \text{const.}$). Se diseña para el seguimiento de la referencia y se espera que la perturbación sea aceptable. Con estas asunciones la influencia de $M_L(t)$ se omite y el diagrama de bloques anterior se convierte en una estructura más clara después del reordenamiento del integrador (Figura 4.15).

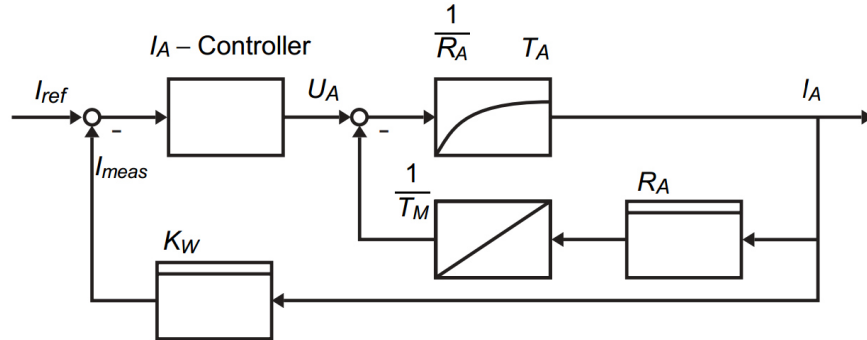


Figura 4.15: Lazo de Control de Corriente.

Debido a que usualmente $T_M > T_A$, la integración de la retroalimentación responderá con un gran retraso en comparación con el rápido lazo de control de la corriente de armadura. Por lo que podemos prescindir de esta retroalimentación en nuestra evaluación del comportamiento dinámico (Figura 4.16).

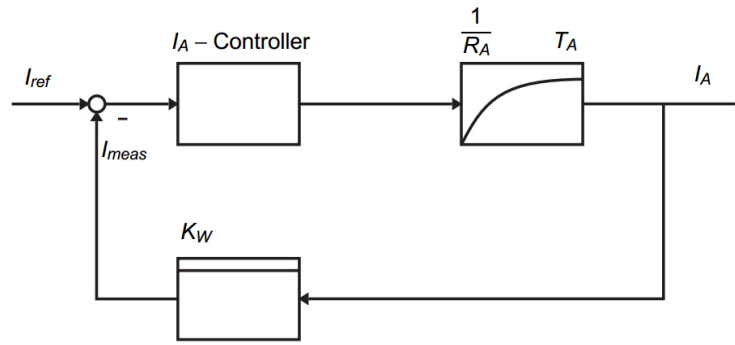


Figura 4.16: Lazo de Control de Corriente Reducido.

Ya que la planta no incluye una parte integral, un controlador PI se escoge para alcanzar precisión en estado estable. Ahora la función de transferencia de lazo cerrado es descrita como una función de primer orden.

$$G_{ir}(s) = \frac{K_i}{1 + T_i s} \quad (4.4)$$

Utilizando el interruptor localizado en el panel frontal de la caja de módulos, la constante de tiempo T_i se ajusta a un Tiempo1 = 0.03 s o Tiempo2 = 0.005 s.

4.2.2 ANÁLISIS EXPERIMENTAL DEL SISTEMA

Las mayores características no lineales que posee el sistema son la zona muerta que se presenta al suministrar señales de entrada inferiores a ± 1.2 V y la zona de saturación que se presenta con señales de entrada mayores a ± 3 V, como se muestra en la Figura 4.17.

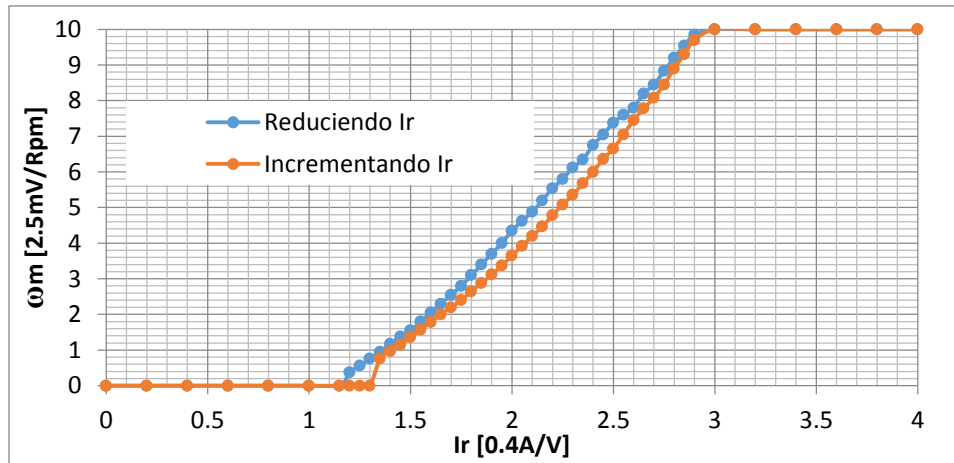


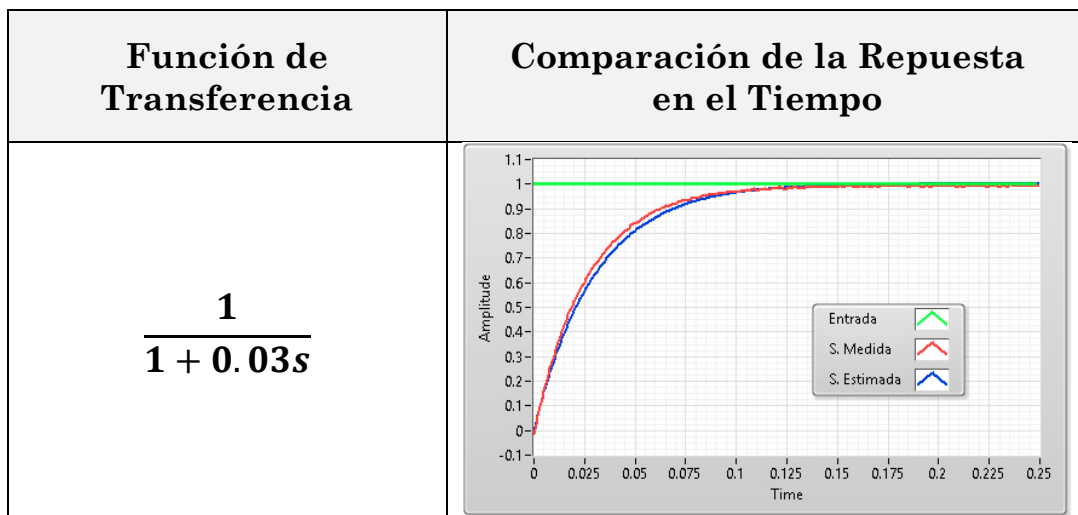
Figura 4.17: Caracterización Estática del Sistema.

Como se muestra en la figura, el sistema muestra un aceptable comportamiento lineal dentro de una región considerablemente amplia. El punto de operación

$$I_{ro} = 2 V \quad , \quad \omega_{mo} = 4.15 V \tag{4.5}$$

se seleccionó con el fin de realizar pruebas para establecer el comportamiento dinámico del sistema describiéndolo como un sistema LTI.

Es sencillo el comprobar el comportamiento del lazo de control de corriente, obteniendo la respuesta en el tiempo de la corriente de armadura $I_A(t)$ para cada una de las dos opciones de la constante de tiempo en el panel frontal (Figura 4.18).



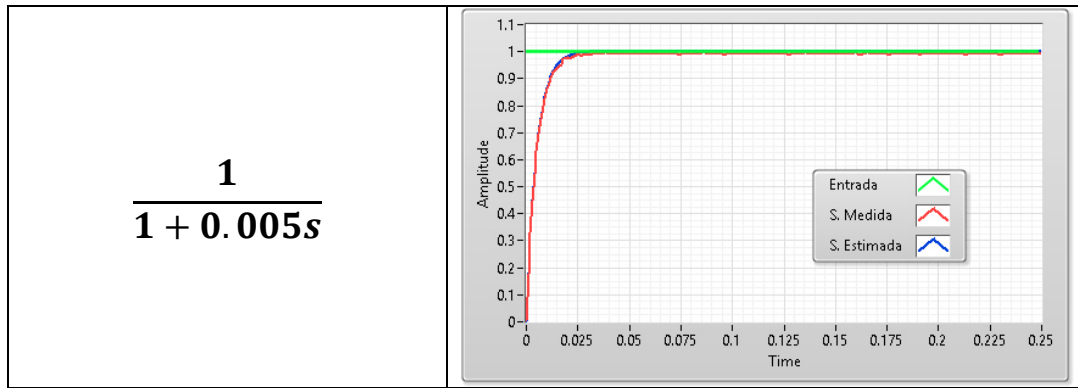


Figura 4.18: Descripción Dinámica del Lazo de Control de Corriente.

En el caso de determinar el comportamiento total entre la entrada $I_r(t)$ y la salida $\omega_f(t)$, el manual de instrucciones prácticas propone realizar experimentos cuyos resultados no son muy exactos. Una alternativa a emplear, más exacta, es posible utilizando las herramientas de identificación de sistemas del NI LabVIEW implementadas a través del NI myRIO.

4.2.2.1 IDENTIFICACIÓN DEL SISTEMA

Considerando la información anterior del sistema, es sencillo el desarrollar pruebas de laboratorio para obtener la descripción matemática general del modelo. Aunque el sistema cuenta con un filtro para reducir el rizo de la señal del tacómetro, no se toma en cuenta para describir el modelo, ya que su respuesta es bastante rápida y aumentaría el orden del sistema.

La identificación consiste básicamente de la obtención de datos de entrada y salida, que posteriormente se procesan para extraer un modelo basado en la información que contienen. Para la implementación y desarrollo de las pruebas de identificación se establece en el esquema de programación de LabVIEW las diferentes tareas del proceso.

- **NIVEL FPGA: GENERACIÓN DE SEÑALES DE ENTRADA Y ADQUISICIÓN DE SEÑALES DE SALIDA.**

La selección de la señal de entrada está determinada por el conocimiento previo que se tiene del sistema. Considerando que el sistema se pretende describir con un modelo LTI de segundo orden, la señal de onda cuadrada es un estímulo suficiente y adecuado para obtener los parámetros del sistema.

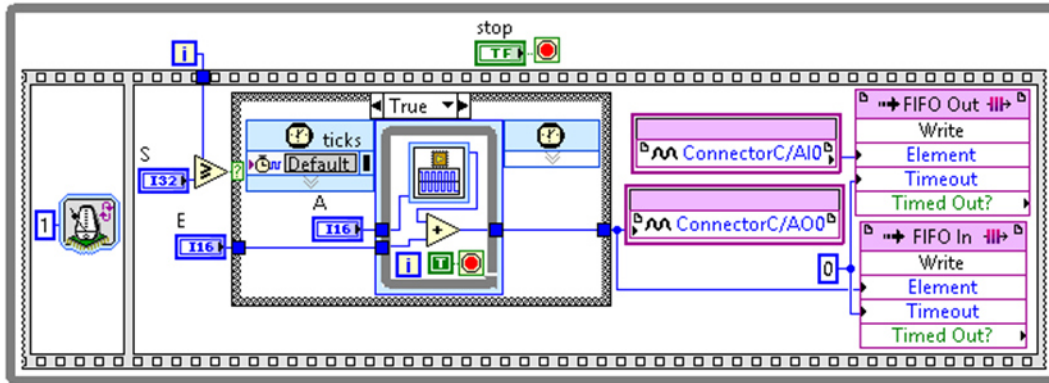


Figura 4.19: Diagrama de Bloques del VI FPGA.

La adquisición de datos se realiza con los DMA FIFO, que establecen un canal de transferencia de datos directo con el nivel Real-Time para su posterior procesamiento.

Los parámetros de ejecución se seleccionan en base a que el sistema alcance el punto de operación, siendo:

- **Entrada Inicial (E).**- es el valor de la señal de entrada I_{r0} , para la cual es sistema alcanza el punto de operación dentro de la región de comportamiento lineal.
- **Tiempo de Llegada al Punto de Operación (S).**- es el tiempo que tarda el sistema en alcanzar el punto de operación estacionario.
- **Amplitud de la Señal de Onda (A).**- para la generación de señales de onda básicas en el VI FPGA, es necesario fijar este valor y mantenerlo dentro del rango de operación lineal del sistema.
- **NIVEL REAL-TIME: IDENTIFICACIÓN Y COMPROBACIÓN DEL MODELO DEL SISTEMA.**

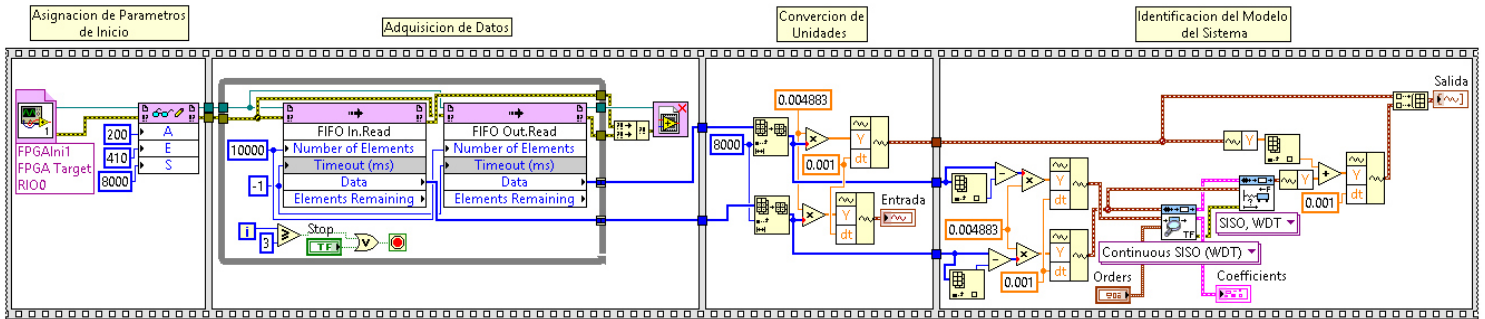


Figura 4.20: Diagrama de Bloques del VI Real-Time.

Para la identificación de los parámetros del sistema se utiliza la función SI Estimate Transfer Function Model VI que estima los parámetros de un modelo en función de transferencia continuo o discreto para un sistema desconocido. Este VI estima solo la función de transferencia entre el estímulo y la respuesta, y es necesario especificar el orden del polinomio del numerador y denominador.

Al tener la información del modelo se simula con la función SI Model Simulation VI para obtener su respuesta para el mismo estímulo de entrada y posteriormente se compara con la señal de salida medida. La correspondencia entre ambas gráficas puede servir como una evidencia para validar el modelo.

- **RESULTADOS DE LA IDENTIFICACIÓN DEL SISTEMA.**

Para la relación entre la entrada $I_r(t)$ y la salida $\omega(t)$ se obtuvo la siguiente función de transferencia promedio al realizar varias pruebas.

$$\frac{\Omega(s)}{I_r(s)} = \frac{6}{0.021s^2 + 0.73s + 1} \quad (4.6)$$

La correspondencia entre las gráficas indica una aceptable aproximación del comportamiento del sistema con el modelo estimado, marcándose más este hecho en los valores de entrada más grandes.

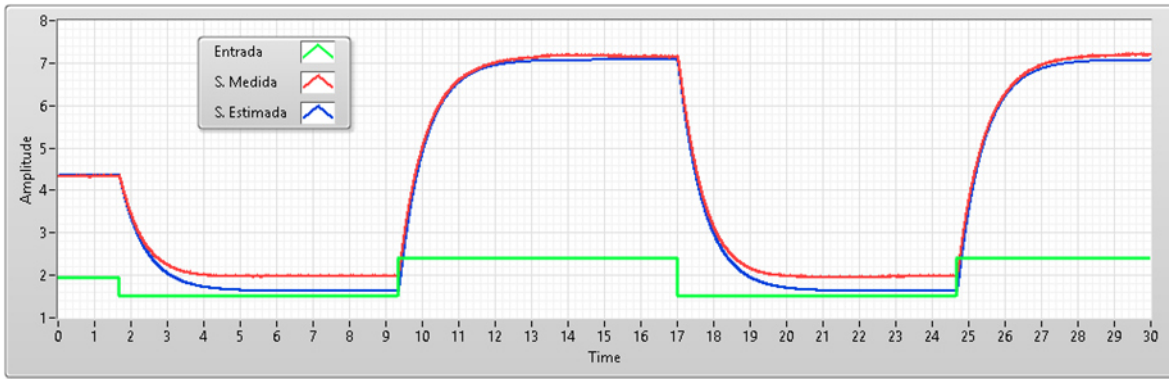


Figura 4.21: Comparación $\omega(t)/I_r(t)$ Medida contra Estimada.

Para la relación entre la entrada $M_L(t)$ y la salida $\omega(t)$, se toma en cuenta que al estimular la entrada $M_L(t)$ no genera ninguna respuesta en la corriente de armadura medida $i_a(t)$, por lo que se asume un subsistema desacoplado y de primer orden, debido a que su comportamiento depende solo de la dinámica de la parte mecánica.

La función de transferencia promedio al realizar varias pruebas, es la siguiente.

$$\frac{M_L(s)}{I_r(s)} = -\frac{3.38}{0.7s + 1} \quad (4.7)$$

Y la siguiente figura muestra las gráficas de la respuesta del sistema y modelo estimado, con una aceptable correspondencia entre sí.

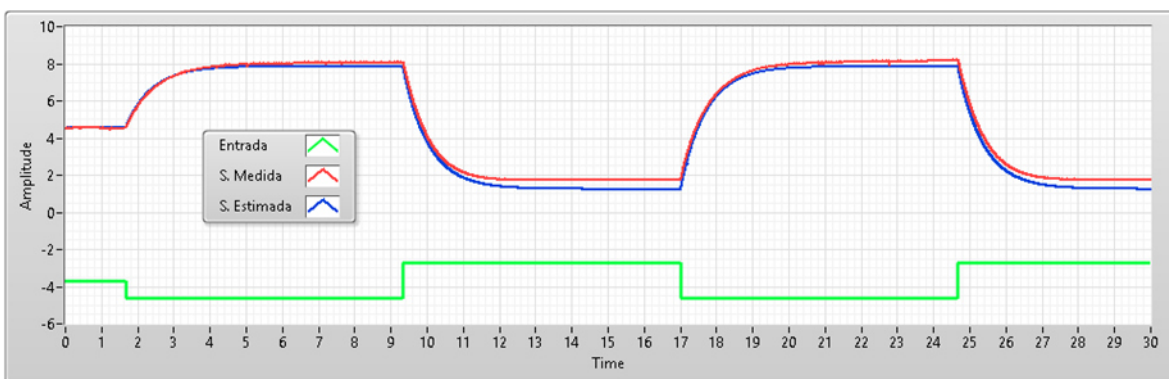


Figura 4.22: Comparación $\omega(t)/M_L(t)$ Medida contra Estimada.

Con estos resultados se obtiene el siguiente modelo completo de la planta a considerar en las pruebas de los algoritmos de control de fallas.

$$\frac{d}{dt} \begin{bmatrix} i_a(t) \\ \omega(t) \end{bmatrix} = \begin{bmatrix} -33.3333 & 0 \\ 8.57143 & -1.42857 \end{bmatrix} \begin{bmatrix} i_a(t) \\ \omega(t) \end{bmatrix} + \begin{bmatrix} 33.3333 & 0 \\ 0 & -4.82857 \end{bmatrix} \begin{bmatrix} I_r(t) \\ M_L(t) \end{bmatrix} \quad (4.8)$$

$$y(t) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} i_a(t) \\ \omega(t) \end{bmatrix}$$

4.2.3 DISEÑO DEL CONTROL NOMINAL

Para la realización de los experimentos que prueban las cualidades de los tres enfoques de control, es necesario que el sistema esté en un estado de operación, realizando alguna tarea específica que el sistema de control tolerante debe mantener al presentarse la ocurrencia de la falla.

Una tarea sencilla para operar el sistema es la de seguir una referencia. Existen varias alternativas de control para cumplir con este objetivo. Entre ellas está la retroalimentación de estado con una precompensación de la señal de referencia de entrada, cuyo algoritmo de implementación es sencillo de desarrollar.

El control nominal implementado en el sistema esta descrito por la siguiente ley de control:

$$I_r(t) = -[0.57714 \quad 0.43714] \begin{bmatrix} i_a(t) \\ \omega(t) \end{bmatrix} + 0.7\omega_r(t) \quad (4.9)$$

con la señal de referencia a seguir:

$$\omega_r(t) = \begin{cases} 0 & \text{si } 0 \leq t \leq 1 \text{ y } t \geq 10 \\ \frac{5}{3}t - \frac{5}{3} & \text{si } 1 < t \leq 4 \\ 5 & \text{si } 4 < t \leq 7 \\ -\frac{5}{3}t + \frac{50}{3} & \text{si } 7 < t < 10 \end{cases}$$

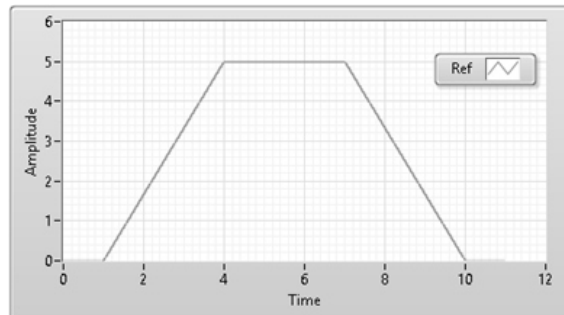


Figura 4.20: Señal de Referencia.

Tomando en cuenta las descripciones del sistema y las pruebas realizadas en este capítulo, se establece el lazo de control nominal como se muestra en la siguiente figura.

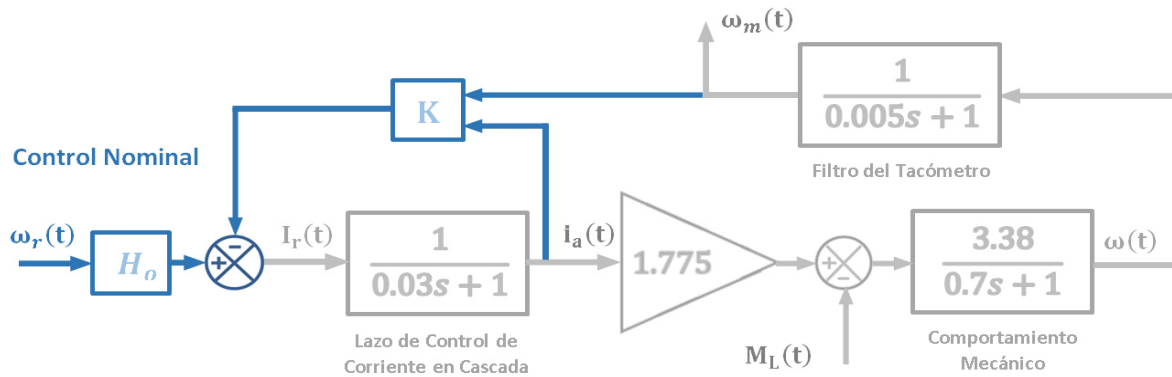


Figura 4.21: Lazo de Control Nominal.

4.2.3.1 IMPLEMENTACIÓN DEL SISTEMA DE CONTROL NOMINAL

Para la implementación del sistema de control nominal, se utilizan algunas funciones nuevas de conversión y agrupación de datos, para un mejor diseño en el desarrollo del código. Los nuevos parámetros a considerar en el control nominal son:

- **Amplitud de la Señal de Referencia (r).**- permite cambiar el valor de la parte superior de la referencia, conservando su forma geométrica de trapecio.
- **Salida Inicial (P).**- es el valor de la señal de salida ω_{m0} , correspondiente al valor de entrada inicial I_{r0} .

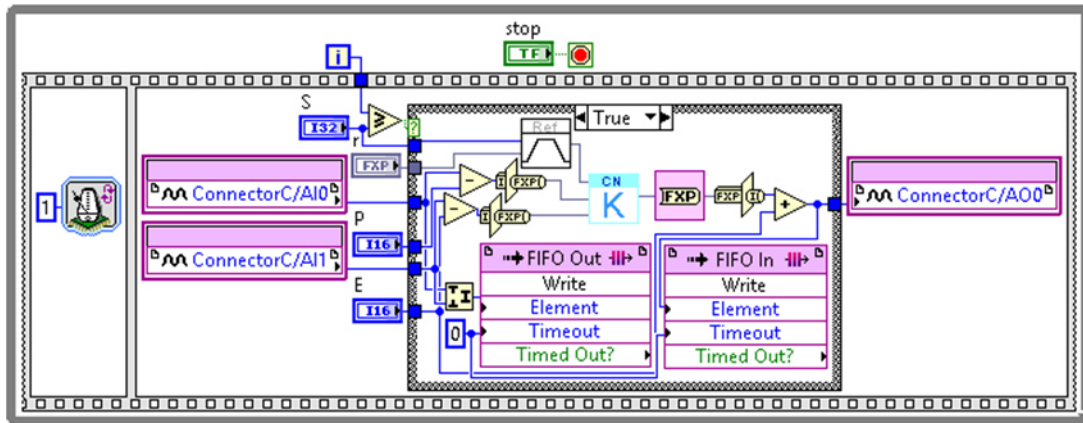


Figura 4.22: Diagrama de Bloques del VI FPGA para el Control Nominal.

El código grafico para la generación de la señal de referencia y la implementación ley de control de retroalimentación se realizan cada uno dentro de un subVI para una mejor distribución del código.

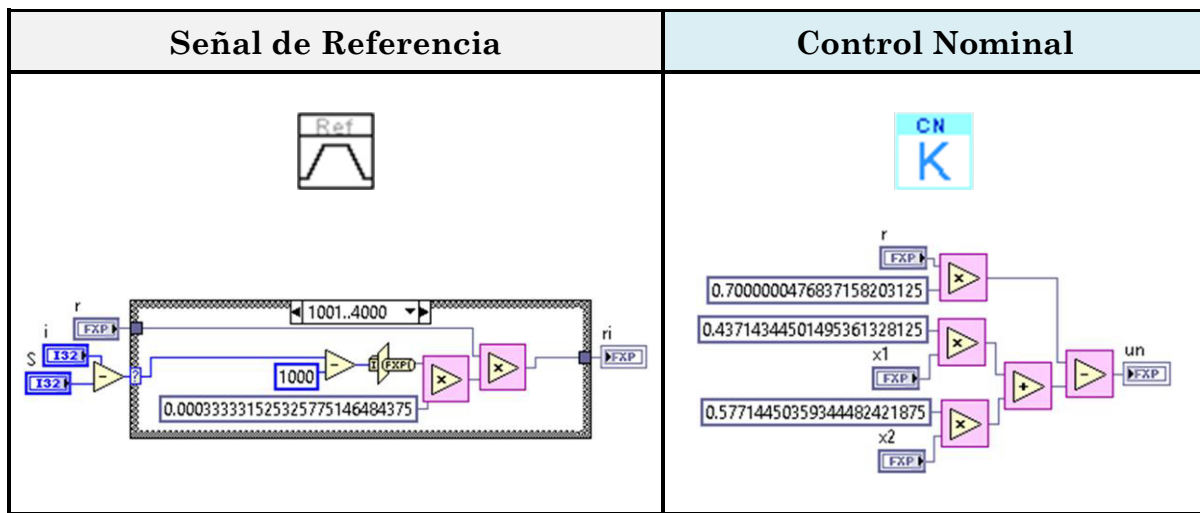


Figura 4.23: Diagrama de Bloques de SubVIs Auxiliares.

La respuesta de la implementación del sistema de control nominal se muestra en la Figura 4.24. Observándose una correcta ejecución y desempeño en los resultados, esta prueba establece una base en el diseño de implementación para las posteriores pruebas en los algoritmos de control tolerante a fallas.

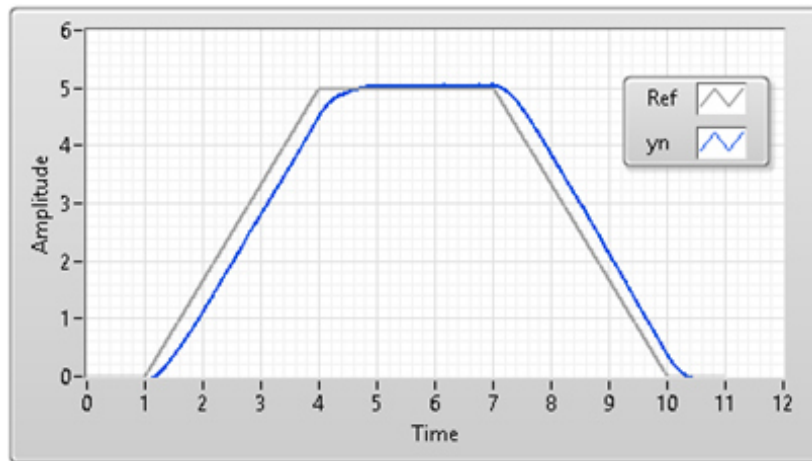


Figura 4.24: Ejecución del Control Nominal.

IMPLEMENTACIÓN DE LAS ESTRATEGIAS DE CONTROL TOLERANTE A FALLAS

5.1 ESCENARIO DE FALLA

El escenario de falla es la representación analítica de una situación de falla conocida, la cual se pretende manejar o contener mediante un estudio de tratamiento de fallas. Una buena selección del escenario de falla, al probar los métodos para la tolerancia a fallas, permite establecer un mejor panorama para el análisis de sus cualidades y deficiencias.

Las tres estrategias de control tolerante a fallas presentan, en comparación con las demás, una mejor disposición al tratar con determinadas situaciones de falla. Por tal motivo es complicado determinar un escenario de falla general. Sin embargo, el actuador virtual requiere ser aplicado a situaciones de falla, donde se opta por cambiar la entrada original del sistema. Esto condiciona el planteamiento del escenario de falla, orientado a ser aplicado en las tres estrategias.

El escenario de falla se plantea como una avería en el actuador principal, donde para la reconfiguración del control se considera una pérdida total de este. Esta avería provoca una ruptura en lazo de control, lo que conlleva a una pérdida temporal de la controlabilidad del sistema, como se muestra en la siguiente Figura 5.1.

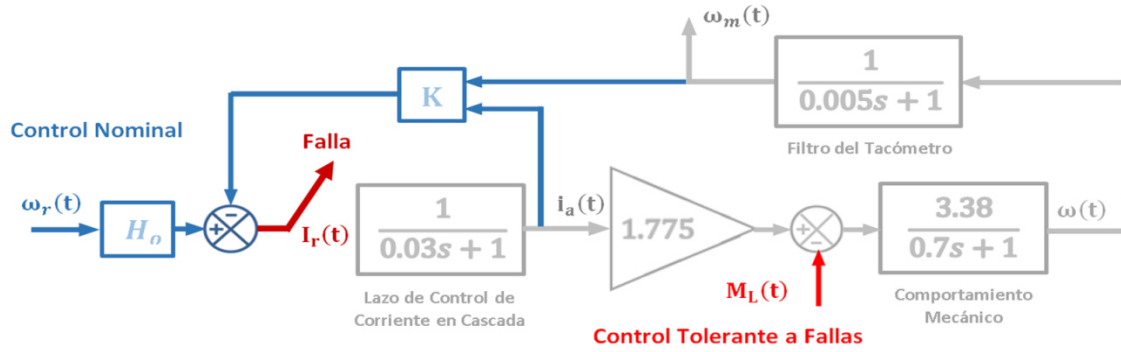


Figura 5.1: Lazo de Control después de la Falla.

Para poder hacer frente a la condición de falla, se utiliza la entrada suplementaria $M_L(t)$ para ejercer la acción de control tolerante a fallas. Como resultado el sistema se reduce, ya que la entrada $M_L(t)$ no afecta el lazo de control de corriente en cascada. El modelo matemático resultante para describir al sistema después de la ocurrencia de la falla es:

$$\frac{d}{dt}\omega_f(t) = -1.42857\omega_f(t) - 4.82857M_L(t) \quad (5.1)$$

$$y(t) = \omega_f(t)$$

5.2 PROPIEDADES DE LA EJECUCIÓN EN PRUEBAS DE LABORATORIO

Aunque en este trabajo no se realizó ninguna tarea de diagnóstico de fallas, debido a que no es de interés en esta tesis y la situación de falla es controlada a través parámetros de inicio en LabVIEW, es importante considerar el consecuente retardo producido por los cálculos en los algoritmos.

Además, debido a que el objetivo de control nominal es la de seguir una referencia que mantiene un cierto comportamiento en un diferentes rangos de tiempo, se plantea establecer diferentes instantes de tiempo en la ejecución de las pruebas para la ocurrencia de la falla.

Para la realización de las pruebas de laboratorio, se consideró el analizar la respuesta del sistema en nueve eventos, que son una combinación de tres tiempos seleccionados de ocurrencia de falla y de retardo en la aplicación de la reconfiguración del control (Tabla 5.1).

Retardo	Ocurrencia de Falla		
0 seg			
0.25 seg	3 seg	6 seg	8 seg
0.5 seg			

Tabla 5.1: Tiempos de Ocurrencia de Falla y Retardo de la Reconfiguración del Control.

Los tiempos de ocurrencia de falla, se seleccionaron en base de examinar la compensación de la falla por el control reconfigurante en las diferentes secciones, donde el sistema de control nominal muestra una trayectoria de seguimiento distinta. Los tiempos de retardo fueron seleccionados de manera más aleatoria, aunque se considera que no deberían ser valores muy grandes.

Para la incorporación de estas propiedades en la ejecución de las pruebas de laboratorio, se añaden dos parámetros de inicio en el código grafico en el nivel de ejecución de la FPGA.

- **Tiempo de Ocurrencia de Falla (F).**- establece el tiempo en el que el lazo de control nominal se rompe y la entrada $I_r(t)$ toma un valor 0 permanente, hasta el final de la prueba.
- **Tiempo de Retardo (D).**- establece el tiempo en que el lazo de control se restablece con la nueva configuración del control.

El tiempo de muestreo $T = 1 ms$, que fue usado en las pruebas para la identificación del sistema, muestra en las gráficas, una buena respuesta para la ejecución de los algoritmos de control. Por lo que se decide mantener este valor para la

ejecución y la discretización de las funciones necesarias para la implementación en las posteriores pruebas.

5.3 IMPLEMENTACIÓN DE LOS ALGORITMOS DE CONTROL TOLERANTE A FALLAS

Las pruebas se realizan con las características y parámetros de ejecución previamente establecidos anteriormente. Los resultados se muestran en las gráficas obtenidas, donde se pueden observar la capacidad que tiene cada estrategia para compensar el efecto de la falla, y se pone en perspectiva los requerimientos y la complejidad para implementar cada método.

5.3.1 IMPLEMENTACIÓN DEL ACTUADOR VIRTUAL

Como anteriormente se mencionó, la situación de falla se planteó para que correspondiera con este enfoque, para satisfacer las condiciones de existencia y facilitar el cálculo de la correspondiente ley de reconfiguración del control.

Al seguir cada paso en el cálculo de la ley de control para la reconfiguración y después de la discretización, se obtiene

$$u_f(k) = -1.77515x_{\Delta}(k) \quad (5.2)$$

con el estado diferenciado $x_{\Delta}(k)$, constantemente generado por el sistema simulado del actuador virtual

$$\dot{x}_{\Delta}(k+1) = 0.948786x_{\Delta}(k) + 0.0324723u_f(k) \quad (5.3)$$

El código grafico desarrollado en el VI FPGA para la ejecución del algoritmo para la reconfiguración del control se muestra en la Figura 5.3.

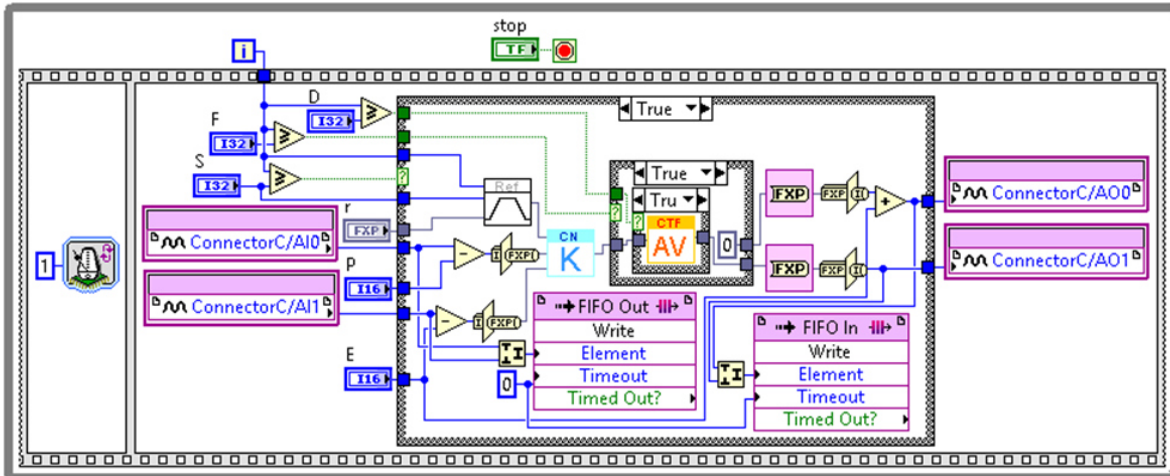


Figura 5.3: Diagrama de Bloques del VI FPGA para el Actuador Virtual.

El código grafico del subVI del actuador virtual se muestre en la Figura 5.4.

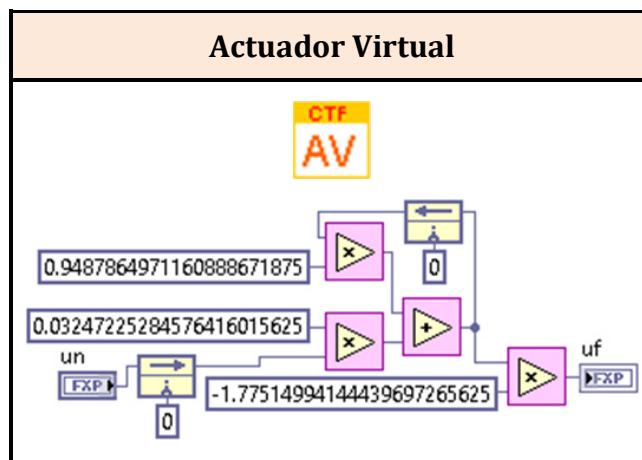


Figura 5.4: Diagrama de Bloques del SubVI del Actuador Virtual.

Los resultados de esta prueba se muestran en la Figura 5.5. En las gráficas se puede notar un error pequeño al alcanzar la referencia de salida $y_n(t)$, esto debido a que el actuador virtual se ve afectado por la incertidumbre del modelo. También el retardo en la aplicación del control tolerante, desajusta la condición inicial $x_\Delta(t_f) = 0$ necesaria para la reconfiguración fuerte.

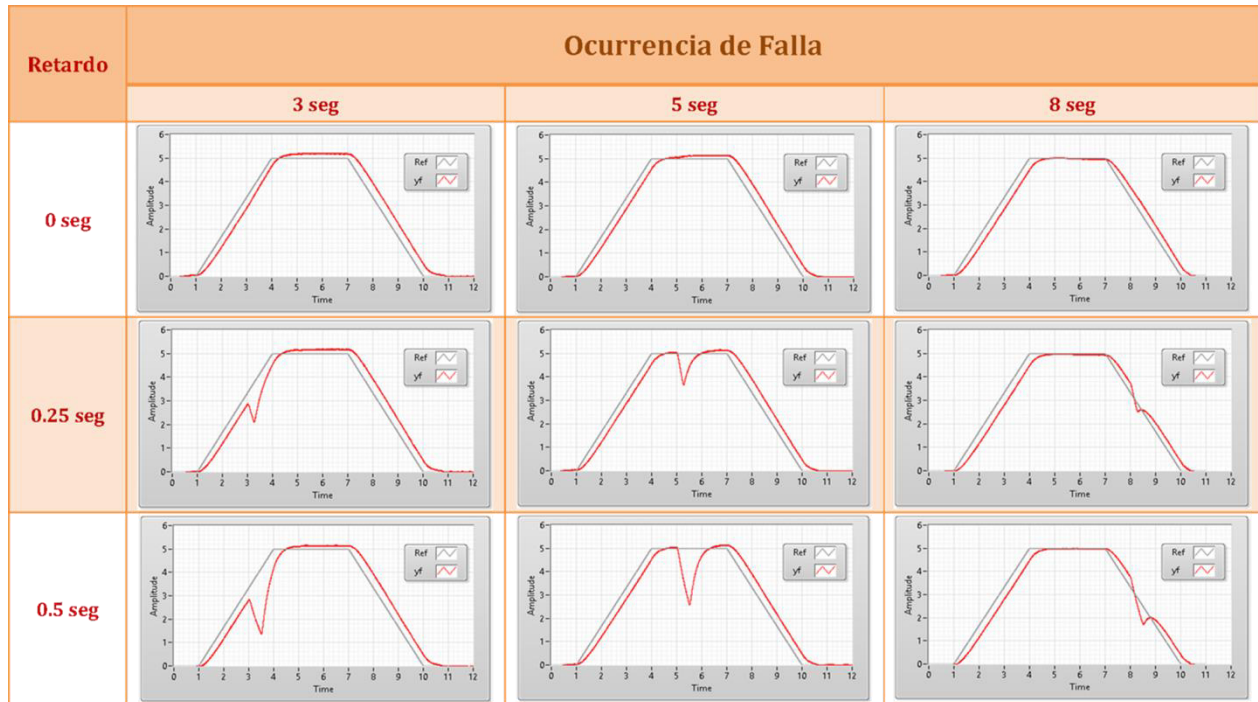


Figura 5.5: Resultados de las Pruebas del Actuador Virtual.

5.3.2 IMPLEMENTACIÓN DEL DISEÑO LINEAL CUADRÁTICO

Debido a que el enfoque de control está basado en el seguimiento de la trayectoria nominal, es necesario recrear la respuesta del sistema nominal para utilizarla en la reconfiguración del control. La ley de control de reconfiguración resultante sería:

$$u_f(k) = 0.74699x_f(k) - 0.958913x_n(k) \quad (5.4)$$

También es necesario el establecer los parámetros de desempeño marcados en la teoría del control lineal cuadrático. Para evitar complicaciones en el diseño y la implementación se utilizan los parámetros estandarizados para el problema convencional de optimización.

$$R = 1, \quad Q = C^T C, \quad \text{Horizonte Infinito} \quad (5.5)$$

El código grafico desarrollado en el VI FPGA para la ejecución del algoritmo de reconfiguración del control se muestra en la Figura 5.6.

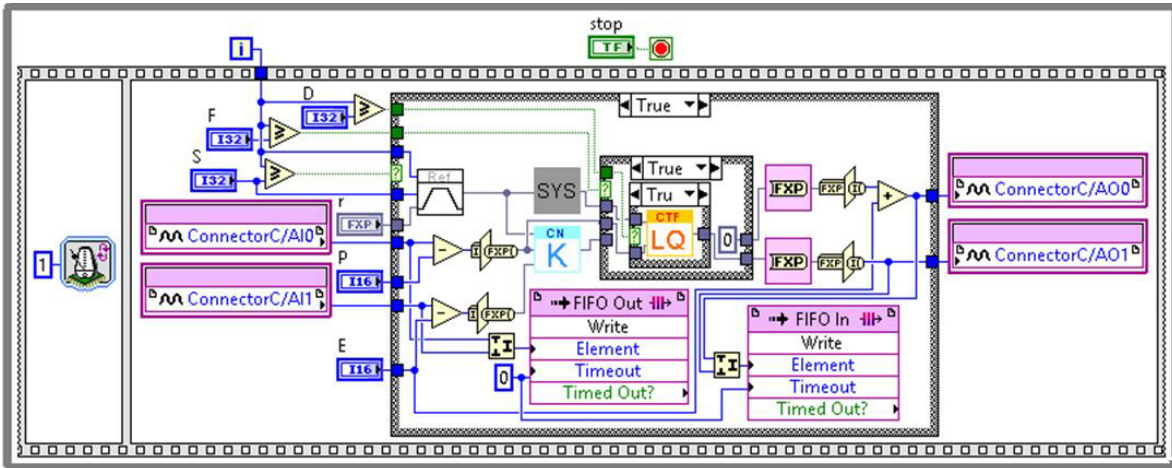


Figura 5.6: Diagrama de Bloques del VI FPGA para el Diseño LQ.

El código grafico para la generación de la trayectoria del sistema nominal y la implementación de la ley de control de reconfiguración, se realizan cada uno dentro de un subVI para una mejor distribución del código (Figura 5.7).

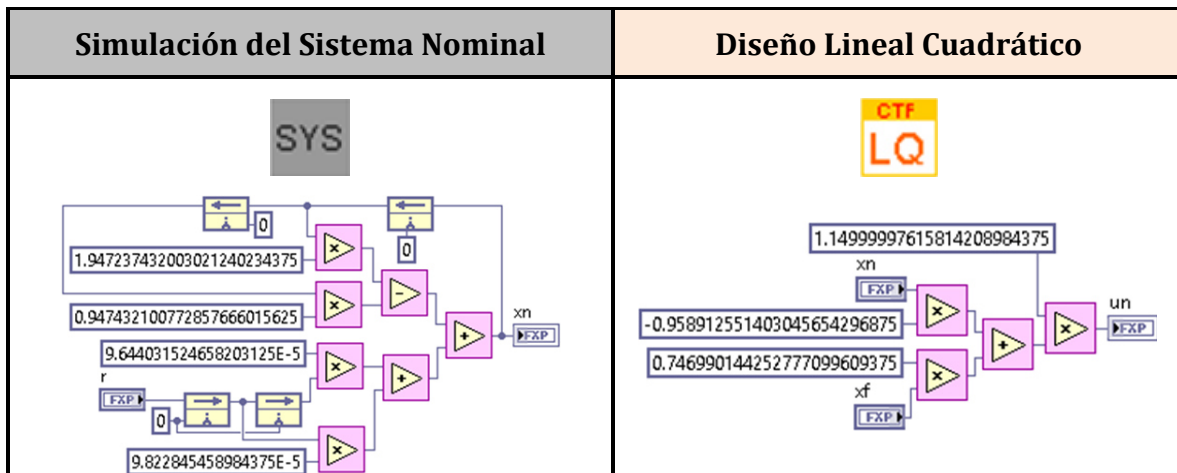


Figura 5.7: Diagramas de Bloques de los SubVIs del Controlador LQ y el Sistema Nominal.

Los resultados de esta prueba se muestran en la Figura 5.8. Los resultados muestran un error constante más pronunciado en el seguimiento que el del actuador virtual, esto debido a la poca rigurosidad al seleccionar los parámetros de diseño del control reconfigurante.

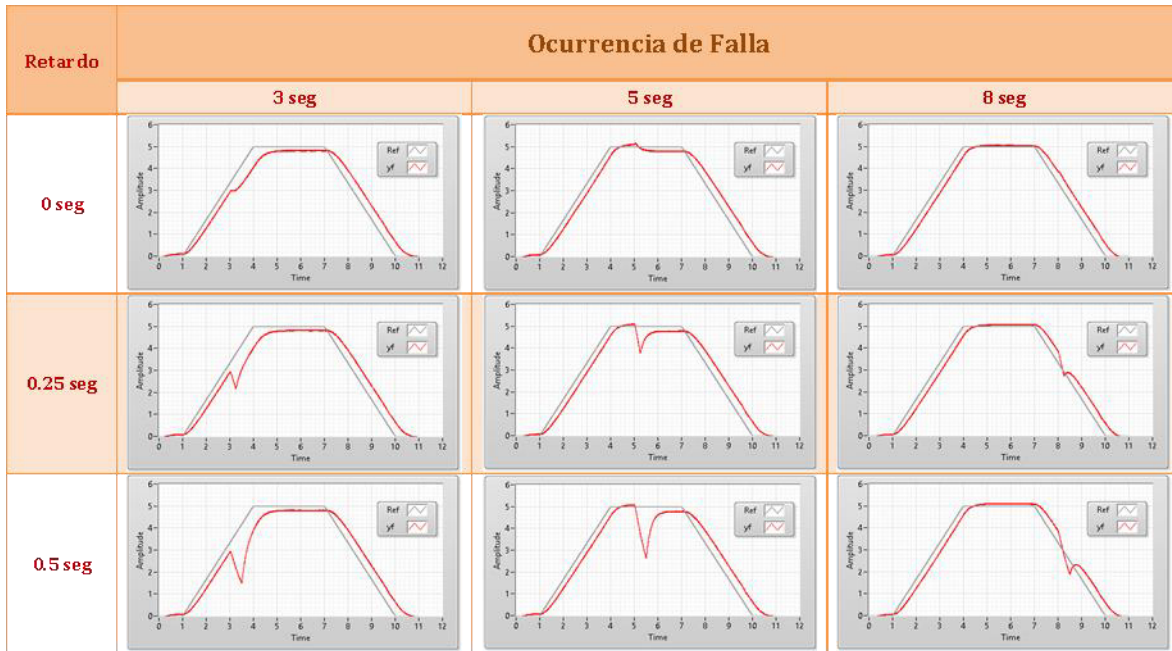


Figura 5.8: Resultados de las Pruebas del Diseño LQ.

5.3.3 IMPLEMENTACIÓN DEL ENFOQUE CASCADA

Al igual que el diseño LQ, el enfoque cascada está basado en el seguimiento de la trayectoria nominal y es necesario recrear la respuesta del sistema nominal. Ya que el sistema da la libertad de seleccionar cualquier controlador para el seguimiento de la trayectoria nominal, se seleccionó un controlador PI, el cual es ampliamente usado en la industria. Siendo la ley de control de reconfiguración resultante:

$$u_f(k) = 10(y_f(k) - y_n(k)) - 9.985(y_f(k-1) - y_n(k-1)) + u_f(k-1) \quad (5.6)$$

El código grafico desarrollado en el VI FPGA para la ejecución del algoritmo de reconfiguración del control se muestra en la Figura 5.9.

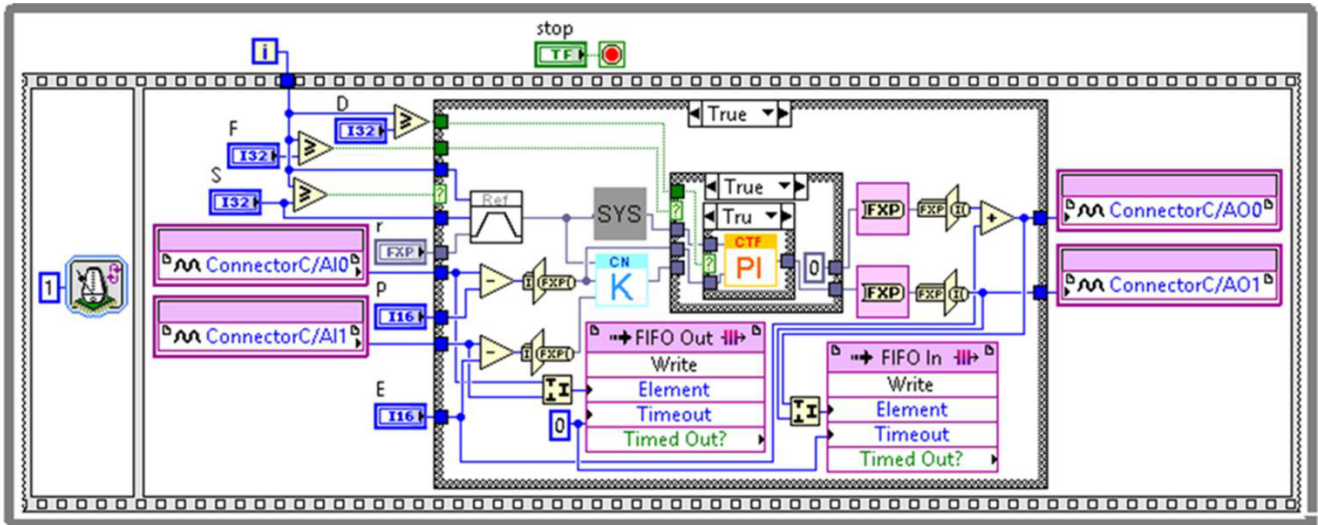


Figura 5.9: Diagrama de Bloques del VI FPGA para el Enfoque Cascada.

El código grafico para la generación de la trayectoria del sistema nominal y la implementación ley de control de reconfiguración se realizan cada uno dentro de un subVI para una mejor distribución del código (Figura 5.10).

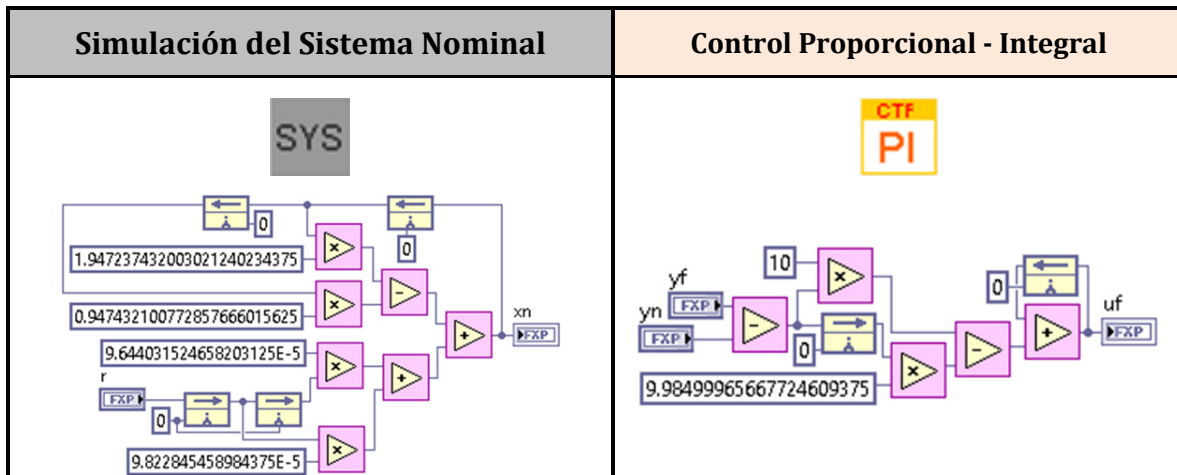


Figura 5.10: Diagramas de Bloques de los SubVIs del Controlador PI y el Sistema Nominal.

Los resultados de esta prueba se muestran en la Figura 5.11. Los resultados muestran una recuperación rápida y satisfactoria del sistema, debido a la selección del control PI para ser implementado en este enfoque.

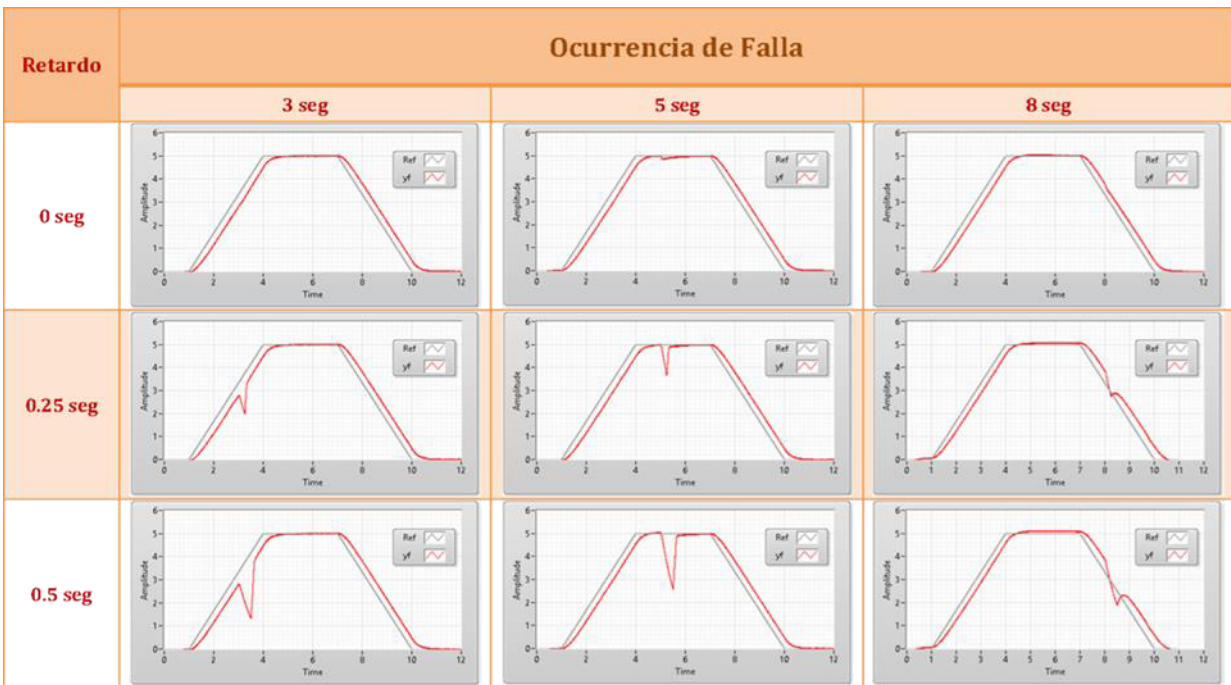


Figura 5.11: Resultados de las Pruebas del Enfoque Cascada.

CONCLUSIONES Y TRABAJO A FUTURO

Las estrategias de control tolerante a fallas basadas en seguimiento de trayectoria nominal, cuentan en general y en la mayoría de las situaciones de fallas recuperables con una solución que dependiendo de las características de diseño, restablecen relativamente la funcionalidad y desempeño del sistema.

El actuador virtual, en cambio, depende de condiciones muy estrictas de existencia, pero la solución que ofrece está muy enfocada para sistemas que cuentan con varios actuadores o actuadores suplementarios, reconstruyendo con una ruta de la señal de control alternativa la función de transferencia original.

La limitación de los escenarios de falla considerados en el desarrollo del actuador virtual, presenta una ventaja al no requerir la estimación del valor de la falla en su diseño, disminuyendo en gran medida el tiempo de retardo en la aplicación de la acción de control tolerante, pero en contraposición es el método que se ve más afectado por la evolución dinámica no controlada ocasionada por un tiempo prolongado de condición de falla.

De lo más importante a resaltar entre las estrategias de control presentadas en esta tesis, es sus características distintivas de diseño. El actuador virtual debido a su formulación basada en la teoría de desacoplo de perturbaciones y su restringido potencial de acción, dispone de una metodología bien establecida que formula una reconfiguración completa del sistema de control en estado de falla. En las estrategias basadas en seguimiento de trayectoria nominal, se diseña ponderando los parámetros del control en el cual se busca satisfacer el objetivo principal de control. El diseño LQ permite considerar cualidades básicas del enfoque óptimo como evaluar la acción de control y el seguimiento de la trayectoria nominal, mientras el enfoque cascada

adquiere las cualidades del control con el que se diseña. Ninguna de las estrategias de seguimiento logra una restauración tan completa del sistema de control nominal, como si lo hace el actuador virtual, pero a cambio ofrecen variantes de diseño de interés general y una mejor disposición para diferentes escenarios de falla.

Al implementar cada una de las estrategias, algunas de sus cualidades se vuelven más evidentes. Se puede notar en los esquemas de control, la estructura particular que forman las diferentes estrategias. En los esquemas del actuador virtual y el diseño lineal cuadrático, la integración del control reconfigurante es muy abrupta, teniendo que romper el lazo de control original para después reconstruirlo con el control reconfigurante. Aunque en ciertas circunstancias podría ser conveniente que el actuador virtual no afecte el control nominal; pudiendo ser una ventaja frente al diseño lineal cuadrático que modifica o simplemente reemplaza al control nominal. En este caso es donde el enfoque cascada resalta su mejor cualidad al no alterar la estructura original del sistema, siendo la mejor opción en situaciones donde no se tiene mucha accesibilidad en el sistema.

Los requerimientos propios de cada una de las estrategias resaltan una distintiva forma de implementación. La creciente complejidad al incrementar las especificaciones en el control (diseño lineal cuadrático) u optar por utilizar alternativas de control más sofisticadas (enfoque cascada) devienen en programación más extensiva, que no suele significar mucho para el objetivo principal de eliminar el efecto de las fallas en la respuesta del sistema. En esta característica el actuador virtual tiene ventaja aparente al ofrecer una implementación relativamente sencilla con requerimientos y resultados bien establecidos. Además de no tener la necesidad de disponer de la trayectoria del sistema nominal, por lo que su programación es más sencilla. El inconveniente más notorio es la afectación que causa las incertidumbres, como se menciona en el capítulo anterior, por lo que toma más importancia tener un buen modelo del sistema.

La Figura 6.1 muestra un resumen simplificado de las características más resaltantes de las conclusiones expuestas.

Estrategias de Control	Condiciones de Existencia	Fallas a Considerar	Sistema de Diagnostico	Características de Diseño	Integración en el Sistema de Control Nominal	Implementación	
Actuador Virtual	Estrictas	Averías en los Actuadores	Simplificado	Metodología bien establecida, que formula una reconfiguración completa	Abrupta	Relativamente Sencilla	
Diseño LQ	Relajadas	Fallas Internas y en Actuadores	Completo	Enfoque Óptimo, que permite evaluar la acción de control y el seguimiento de la trayectoria	Abrupta	Es necesaria la disposición de las trayectorias nominales	Su complejidad depende de los requerimientos del diseño
Enfoque Cascada	Relajadas			Permite utilizar diferentes estrategias de control para seguimiento	No altera la estructura original del Sistema		Su complejidad depende de la estrategia de control usada

Tabla 6.1: Conclusiones Comparativas de las tres Estrategias FTC.

Algunas partes de la teoría desarrolla en estas estrategias de control tolerante no fueron tomadas en cuenta para ser consideradas en la implementación, debido a que se extienden a tratar diferentes objetivos (actuador virtual)] o a considerar técnicas para el cálculo de la ley de control de tolerante mientras se alcanza una estimación adecuada de la falla (diseño lineal cuadrático)].

Otros escenarios de falla también fueron descartados, como un escenario con fallas en los sensores o fallas internas. El principal inconveniente para realizar pruebas con fallas internas, es la limitación del sistema del laboratorio, que vuelve complicado cambiar los parámetros de la matriz de estados. Para tratar fallas en sensores existe un método complementario al del actuador virtual, llamado “sensor virtual”, que básicamente se trata de un observador de Luenberger, que por sí solo no causa mucho interés, pero que junto con el actuador virtual ofrecen una solución para las fallas internas.

El enfoque cascada es una estrategia relativamente nueva, que aún está siendo analizada para diferentes controladores. Un análisis comparativo de diferentes controladores para utilizar en esta estrategia, podría resaltar sus cualidades y expandir sus formas de uso.

En general, en trabajos futuros se podría intentar aclarar las capacidades de cada estrategia y su correspondencia con las diferentes situaciones de falla, al igual que integrar la parte de diagnóstico necesaria para cada método y realizar un estudio completo de tratamiento de fallas, en un laboratorio de pruebas que permita mayores posibilidades de experimentación.

BIBLIOGRAFÍA

- [1] T. Steffen, *Control Reconfiguration of Dynamical Systems: Linear Approaches and Structural Tests*. Berlin, Germany: Springer-Verlag, 2005.
- [2] M. Blanke, M. Kinnaert, and J. L. M. Staroswiecki, *Diagnosis and Fault-Tolerant Control*, 2nd ed. Berlin Germany: Springer, 2006.
- [3] J. Lunze and T. Steffen, "Control reconfiguration after actuator failures using disturbance decoupling methods," *IEEE Transactions on Automatic Control*, vol. 51, no. 10, pp. 1590–1601, October 2006.
- [4] JM. Staroswiecki and F. Cazaurang, "Fault recovery by nominal trajectory tracking," in *American Control Conference*, Seattle, Washington, USA, June 11-13 2008, pp. 1070–1075.
- [5] R. Isermann, *Fault-diagnosis systems: An introduction from fault detection to fault tolerance*, 1st ed. Springer, November 2006.
- [6] C. Verde, S. Gentil y R. Morales-Menéndez, *Monitoreo y Diagnóstico Automático de Fallas en Sistemas Dinámicos*, Ed. Trillas, 2013.
- [7] Kirk, D. E. (1970). *Optimal Control Theory, An introduction*. Prentice Hall. New Jersey.
- [8] P. Acosta-Santana, L.H. Rodriguez-Alfaro, E. Alcorta-Garcia, "A cascade structure approach to control reconfiguration", *Proceedings of the 2nd*

- International Conference on Control and Fault-Tolerant Systems SysTol'13*, pp. 347-352, 2013.
- [9] H. Noura, D. Theilliol, J.-C. Ponsart, A. Chamseddine, “*Fault-tolerant Control Systems: Design and Practical Applications*”, Springer Verlag London, 2009. Advances in industrial control.
- [10] Steven X. Ding, “*Model-based fault diagnosis techniques—design schemes, algorithms and tools*”, 2nd edn. Springer, London, 2013.
- [11] Manuales de Productos – National Instruments (2016)
<http://www.ni.com/manuals/esa/> (URL).
- [12] National Instruments. (2013). User Guide and Specifications NI myRIO – 1900, United States: National Instruments Trademarks.
- [13] Amira: DR300 Laboratory Setup Speed Control with Variable Load, AMIRA GmbH, Duisburg, 2000.
- [14] Zhang, Y. y J. Jiang. (2008). Bibliographical review on reconfigurable fault-tolerant control systems. *Annual Reviews in Control* **32**(2), 229–252.
- [15] Staroswieki, M., H. Yang y B. Jiang (2007). Progressive accommodation of parametric faults in linear quadratic control. *Automatica* 43, 2070–2076.
- [16] E. Alcorta-Garcia, D. San Roman Silva, D. A. Diaz-Romero, Reconfiguración del Control Mediante Seguimiento de Trayectoria, CNCA 2013, p.g. 496–501.
- [17] Korbicz, J., Kościelny, J.M.: Modeling, Diagnostics and Process Control. Implementation in the DiaSter System. Springer, Heidelberg (2010).