ESCUELA POLITÉCNICA DEL EJÉRCITO

FACULTAD DE INGENIERÍA ELECTRÓNICA

PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO EN INGENIERÍA ELECTRÓNICA

DISEÑO E IMPLEMENTACIÓN DE UN EMULADOR DE PLANTAS SISO ANALÓGICAS

JOSÉ DAVID SANTANA COBO

QUITO – ECUADOR

2006

CERTIFICACIÓN

Certificamos que el presente proyecto de grado: "**Diseño e implementación de un emulador de plantas SISO analógicas**", fue desarrollado en su totalidad por el señor José David Santana Cobo, bajo nuestra dirección.

Atentamente,

Ing. Rodolfo Gordillo	Ing. Víctor Proaño
DIRECTOR	**CODIRECTOR**

AGRADECIMIENTO

A Dios por ser mi sustento y guía en todo.

A mis padres, Daniel y Cecilia, por haberme educado con amor, honradez y valores cristianos, mostrando su apoyo incondicional durante toda mi vida.

A mis hermanos y mi abuela Margarita por apoyarme y darme aliento durante toda mi carrera y especialmente durante la Tesis.

A mi tío Pepe y prima Nicolina que facilitaron la adquisición de los componentes para realizar este Proyecto.

A todos los amigos que durante la carrera fueron apoyo e inspiración para seguir adelante.

Al Director y Codirector de este proyecto de grado que, con apertura y disposición, brindaron sus conocimientos y guía para concretar mis ideas.

DEDICATORIA

A mis padres, Daniel y Cecilia.

PRÓLOGO

I- ANTECEDENTES

A lo largo de los años los educadores han reconocido la necesidad e importancia de realizar prácticas de control en laboratorio con el fin de reforzar los conocimientos teóricos adquiridos en clase respecto al diseño de controladores, identificación de plantas y aplicación de conceptos de control industrial. La realización de prácticas en laboratorio ayuda a los estudiantes a relacionar los conceptos teóricos, muchas veces complicados, con aplicaciones reales de tal forma que tengan una mayor motivación en el aprendizaje de técnicas de control. Adicionalmente se introduce al estudiante en la integración de sistemas de control al tener que afrontar problemas típicos no tomados en cuenta muchas veces en la teoría como no linealidades, ruido, entre otros y el acondicionamiento de señal para compatibilizar con uno u otro controlador.

La motivación original detrás de la construcción de modelos a escala de procesos industriales para laboratorios es la de ofrecer a los estudiantes un sistema completo para el diseño de ingeniería, empezando por el modelamiento y simulación hasta terminar con la validación experimental de un sinnúmero de estrategias de control [*1*].

La gran utilidad que se puede dar a estos equipos se ve muchas veces limitada por el costo de los mismos y el espacio que estos pueden llegar a ocupar, limitando su utilización a prácticas con pocos estudiantes. Adicionalmente cada módulo representa un proceso en particular, lo que los vuelve inflexibles si se quiere variar parámetros del proceso o involucra la necesidad de contar con un modelo a escala distinto dependiendo del proceso que se desea controlar.

Ante los problemas que tienen los modelos a escala, en lo últimos años se ha visto una mayor tendencia a desarrollar modelos virtuales que se ejecutan en tiempo real ya sea en un computador personal (PC) o en módulos especializados basados en DSPs ó incluso microcontroladores. A este hardware se le equipa adicionalmente con la electrónica necesaria que contiene conversores A/D, D/A, entradas y salidas digitales. La idea de reemplazar los modelos a escala (o incluso la planta real) con un equipo basado en software se denomina Hardware en el Lazo ó HIL, por sus siglas en inglés.

II- ALCANCE DEL PROYECTO

El presente proyecto consiste en realizar un hardware compacto y barato basado en un Controlador Digital de Señales de la Familia Microchip dsPIC30F que permita la programación y simulación en tiempo real de sistemas SISO analógicos (una sola entrada y una sola salida), que en este proyecto se lo denominará *emulador SISO*, el cual admita y entregue señales de control analógicas unipolares y PWM.

El emulador contendrá en su forma básica la capacidad de simular funciones de transferencia hasta de 4to orden, retardo de transporte, zona muerta, saturación, curvas no lineales y perturbación externa. Una interfaz gráfica (GUI) desarrollada en MATLAB permitirá la programación de los distintos componentes del sistema y la descarga de parámetros al hardware por medio del puerto serial del computador (Ver Figura i).

El dispositivo contará con ciertas características que lo hagan realizable en nuestro país, así como la utilización de elementos comunes y relativamente fáciles de conseguir.

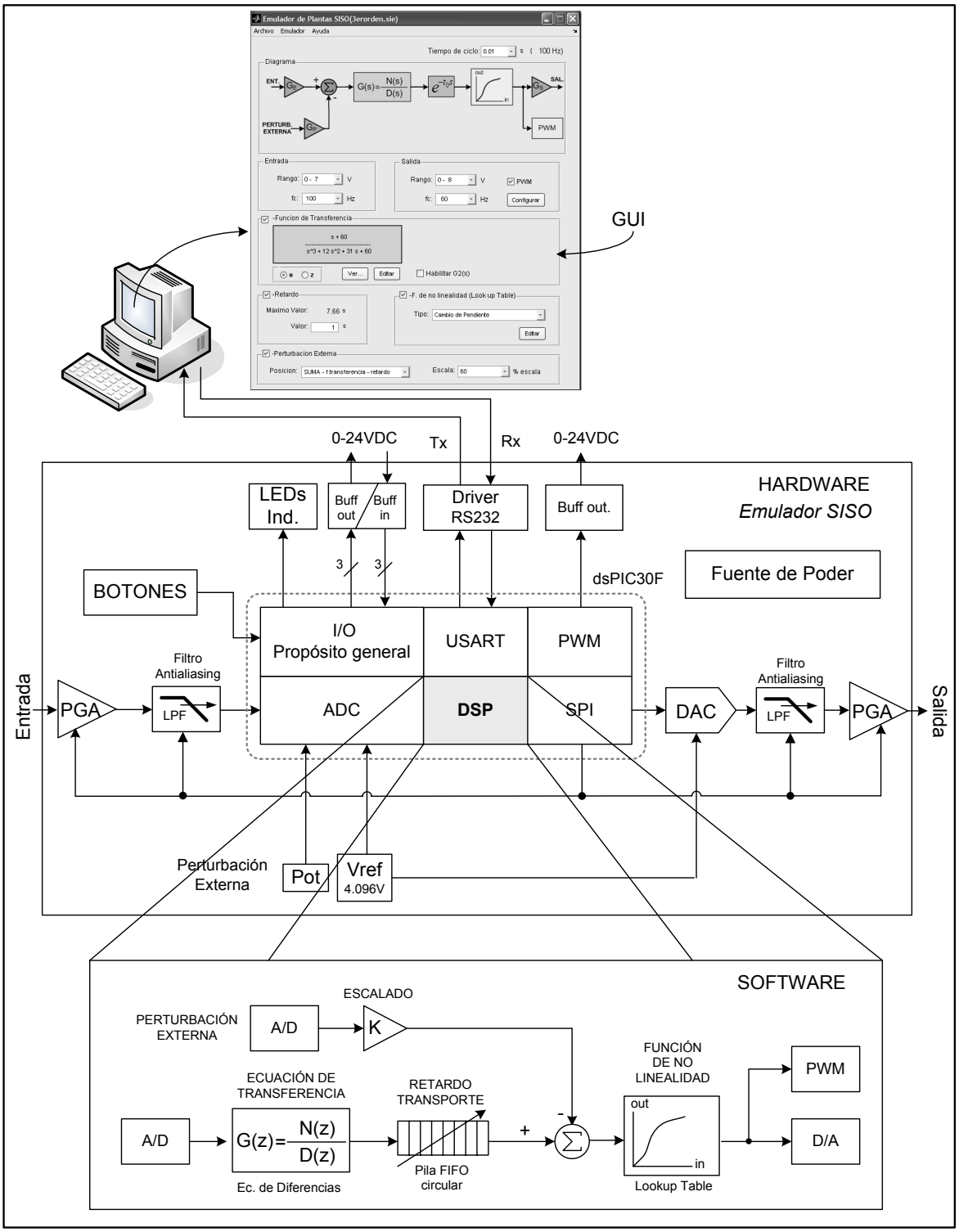


Figura. i. Componentes del Emulador SISO

III- ESTRUCTURA DEL PROYECTO

En el **Capítulo 1** se expone el nacimiento y el avance de los sistemas HIL, los conceptos y filosofía que subyacen al uso de estos, las aplicaciones y equipos disponibles.

El **Capítulo 2** está dedicado a la manera que se simulan las plantas físicas y se expone los métodos en que los distintos componentes que modelan una planta se los puede implementar en software. Adicionalmente se indica los aspectos relevantes a la adquisición de datos: especificaciones y determinación de la frecuencia de muestreo, filtros antialiasing y otros parámetros.

El **Capítulo 3** expone todos los aspectos relevantes al diseño de hardware del dispositivo: selección de los distintos componentes, diseño del circuito impreso y precios referenciales del proyecto.

En el **Capítulo 4** se desarrolla la estructura del software para el dispositivo emulador y el programa de interfase (GUI).

El **Capítulo 5** explica las pruebas al sistema y expone los resultados obtenidos.

El **Capítulo 6** contiene las conclusiones y recomendaciones referentes al presente trabajo.

En la sección de **Anexos** se incluye un manual de funcionamiento del emulador de plantas SISO con un ejemplo de aplicación conjuntamente con información adicional mencionada a lo largo del proyecto.

ÍNDICE DE CONTENIDO

CAPÍTULO 1

SISTEMAS HARDWARE EN EL LAZO

1.1 BREVE HISTORIA DE LOS SISTEMAS HARDWARE EN EL LAZO (HIL)

Desde la aparición de la teoría de control con retro alimentación, los ingenieros han visto la necesidad de probar los distintos mecanismos de control automático. Después de encontrar teóricamente los valores del mejor controlador se realizaba una serie de experimentos de prueba y error en los sistemas que se deseaban controlar con el fin de encontrar los parámetros que mejor desempeño mostraran en el mencionado proceso. Este mecanismo es válido hasta nuestros días pero con grandes diferencias.

Tras la aparición del transistor en 1947 y el desarrollo de las primeras computadoras analógicas, la ingeniería de control vio en estas herramientas una de las maneras más poderosas para implementar controladores. Hasta ese entonces, las pruebas se realizaban en los sistemas físicos o modelos físicos a escala que debían controlar o, en casos excepcionales, se implementaba un circuito analógico altamente complejo que simulaba el comportamiento de la planta deseada. Es aquí donde empieza el nacimiento de los sistemas HIL.

Sin embargo, el uso de emuladores resultaba altamente costoso y limitado a grandes proyectos; incluso la existencia de las computadoras digitales estaba circunscrita a pequeños grupos de investigación. Por muchos años la mayoría de estudiantes de ingeniería de control tenían que conformarse con controladores analógicos y modelos de plantas.

Se puede decir con seguridad que el advenimiento de la computadora personal a mediados de los 80 causó una proliferación de software encargado del diseño y simulación de sistemas y proveyó a los ingenieros de una herramienta poderosa para el análisis y diseño de controladores y modelamiento de sistemas. Pero tardaría varios años más hasta que la potencia de procesamiento aumente a un punto que se puedan realizar simulaciones de sistemas complejos en tiempo real [*2*].

En la década de los 90 se produjo la diversificación y aumento de poder computacional de procesadores, destacando la aparición en el mercado de los Procesadores Digitales de Señales (DSPs). Fue este factor sumado a la complejidad que iban adquiriendo muchos productos de consumo masivo que exigían menor tiempo de desarrollo, lo que catapultó el uso de hardware dedicado a la simulación del modelo de plantas para probar los distintos controladores [*3*].

1.2 FILOSOFÍA DE UN SISTEMA HIL

1.2.1 Generalidades

Como ya se ha mencionado con anterioridad, un simulador HIL consiste en reemplazar la planta o proceso por un hardware que simule en tiempo real el modelo de la planta (*Figura 1.1*). Una simulación en tiempo real involucra que la simulación sea efectuada de tal manera que las señales de entrada y de salida muestren los mismos valores dependientes en el tiempo de la misma manera que lo hace el componente real [*4*].

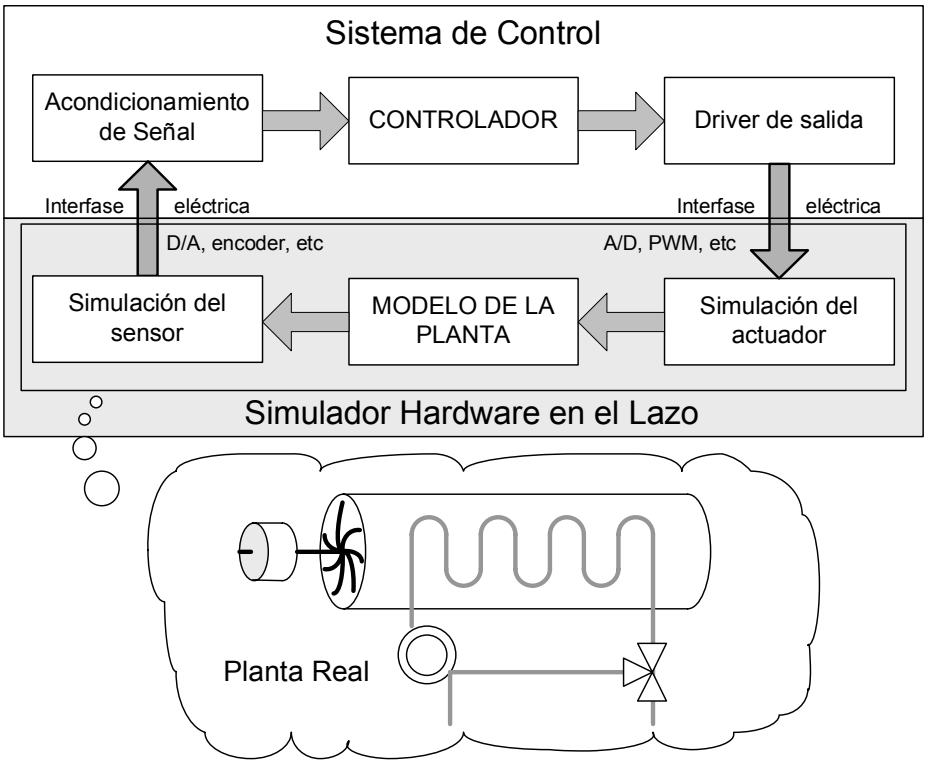


Figura. 1.1. Elementos de un sistema HIL y de control

Esto significa que la simulación debe tener un comportamiento tal, que el sistema de control no note la diferencia entre la planta real y el simulador HIL. Es por esta razón que un ciclo del simulador HIL debe ser algunas veces más rápido que el ciclo de control del sistema del controlador [*3*]; esto determina la potencia computacional que debe tener un simulador HIL dada la complejidad de la planta y el método de simulación que se utilice. Estos puntos serán explicados con más detalles en el Capítulo 2.

1.2.2 Ventajas

- Resulta especialmente útil al probar controladores para sistemas altamente complejos como vehículos, máquinas numéricas, etc. ó para procesos peligrosos como mezclas químicas o reactores nucleares. El costo de una falla en un sistema de estos podría ser, por algunos órdenes de magnitud, mayor al costo de un simulador HIL avanzado.

- Es posible someter y probar al controlador a condiciones extremas de funcionamiento, algo que con la planta real podría ser peligroso y costoso.
- Con un simulador HIL se cuenta con todas las señales generadas por un modelo y no simplemente por un generador de señales. El controlador se prueba ahora en lazo cerrado lo que recrea un escenario más parecido a la situación real.
- El diseño y pruebas del Software pueden ser movidos a una fase anterior del diseño ya que se pueden realizar sobre un simulador HIL antes de que esté concluido el primer prototipo físico/mecánico.
- Es posible variar los parámetros de la planta, ya sea el momento de modelarla o en tiempo real durante la simulación.
- El hardware puede ser reutilizado para simular distintos modelos sin requerir modificaciones físicas, o si las hay, son de fácil realización.

1.2.3 Desventajas

- Un simulador HIL no reemplaza las pruebas rigurosas que se debe hacer con la planta real, solo ayuda al desarrollo y depuración del controlador. El modelo de planta que uno simula puede ser muy detallado, pero siempre existirá algún parámetro que no se ha tomado en cuenta.
- Un simulador HIL no puede detenerse mientras se ejecuta, esto es, no es posible poner puntos de prueba (*breakpoints*) en el controlador mientras la simulación se lleva a cabo. Lo que realmente se desea es ver, en forma global, el comportamiento del controlador.
- El simulador HIL no puede saber lo que pasa dentro del controlador; no es un reemplazo para un analizador lógico o un depurador de software. Solo puede leer las salidas del controlador. Si ocurre alguna falla de software lo más seguro es que no haya suficiente información en las salidas del controlador como para saber qué porción del código se estaba ejecutando o el estado de las variables internas.

1.3 COMPONENTES

1.3.1 Hardware

1.3.1.1 Aspectos generales del Hardware

Existen tres componentes principales en un simulador HIL: un procesador y su memoria encargado de los cálculos del modelo de la planta, un conjunto de periféricos que forman la interfase eléctrica con el controlador y una interfaz de usuario que permite la configuración del simulador y modificación de parámetros en línea (*Figura 1.2*).

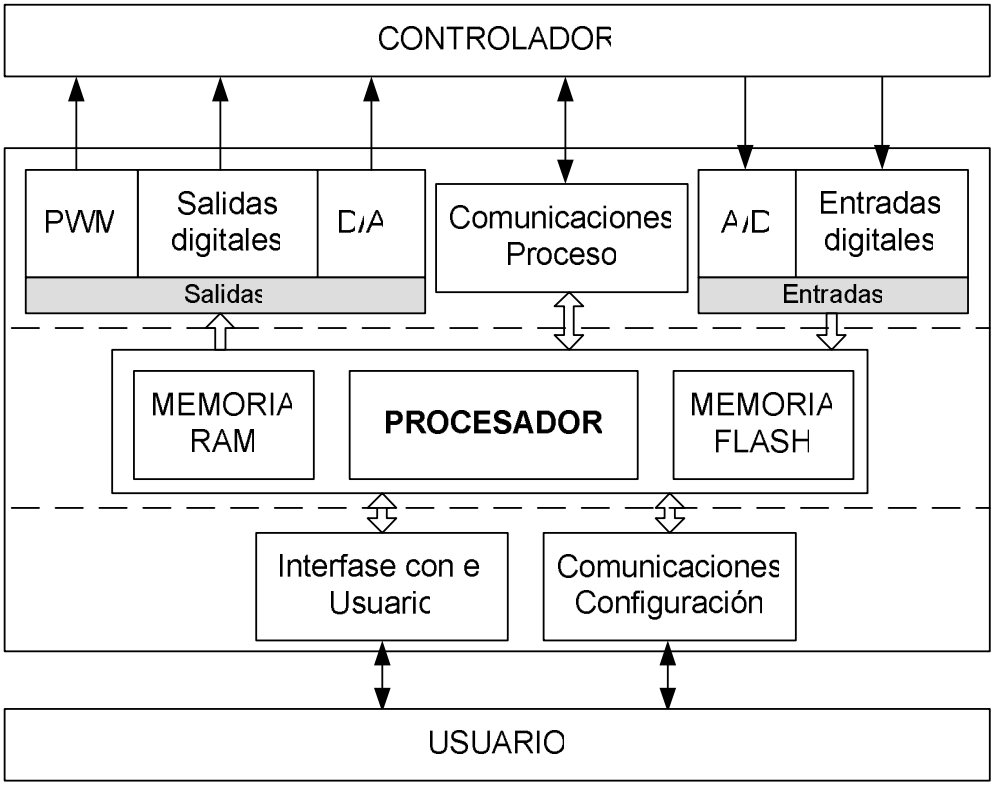


Figura. 1.2. Componentes internos de un simulador HIL

1.3.1.2 Procesador y memoria

Sea cual fuere el simulador HIL, este debe contar con por lo menos un procesador y memoria RAM. En la memoria (temporal o permanente) se guarda el código que representa el modelo de la planta, mientras que el procesador ejecuta los cálculos del código contenido. Un sinnúmero de variables determinan la eficacia de un procesador u otro para realizar de mejor o peor manera los cálculos necesarios para la simulación y estos van desde el tipo de instrucciones que maneja el procesador hasta la velocidad entre los distintos niveles de memoria.

Por ejemplo, si un simulador HIL se ejecuta sobre una plataforma típica de PC, se puede contar con un alto procesamiento de datos y una gran cantidad de memoria RAM, lo que no ocurre si hablamos de un sistema HIL basado en un microcontrolador. La elección de una u otra plataforma será dependiente de la aplicación en que vaya a ser usada.

1.3.1.3 Periféricos de Interfase con el controlador

Como consecuencia lógica de imitar un sistema real, un simulador HIL deberá contar necesariamente con la interfase eléctrica necesaria de las señales de entrada para interactuar con el controlador. Estas interfases pueden ser conversores A/D, D/A, entradas y salidas digitales discretas, encoders de pulsos e interfases de comunicación como RS232, LAN, CAN, LIN, entre otras.

1.3.1.4 Periféricos de Interfase con el usuario

Para complementar la funcionalidad de un simulador HIL es conveniente contar con una o varias interfases que permitan configurar y modificar al sistema que se desea simular. Una forma de interfase son controles directos como botones, perillas o teclados que modifican la planta según lo programado mientras que otra manera es usando un canal de comunicación (RS232, LAN, etc.) que, enlazado

por un programa de interfaz de usuario, modifica los parámetros de la planta a través de una PC. Este canal de comunicación es en la mayoría de casos indispensable ya que sirve para la descarga de la configuración del sistema.

1.3.2 Software

La principal característica de un simulador HIL es la de ejecutar el ciclo de simulación en tiempo real. Para lograr esto el código de simulación debe estar sincronizado con el reloj del hardware en el cual se ejecuta. Una de las formas es llamar al ciclo de simulación con la interrupción del reloj o encargar la invocación de esta a un sistema operativo [*5*]. La mayoría de simuladores HIL se basan en un sistema operativo en tiempo real (RTOS). El tiempo entre dos llamadas al código que representa la planta se denomina tiempo de ciclo de simulación.

1.3.3 Integración de los simuladores HIL

Como se expondrá más adelante en este capítulo, existen muchas maneras de realizar un simulador HIL, por esta razón la *Figura 1.2* no representa necesariamente un diagrama de Hardware sino más bien un diagrama conceptual del simulador HIL. Los nuevos microcontroladores o DSP contienen muchos de los periféricos, la memoria RAM, memoria Flash y procesador en una misma pieza de silicio lo que simplifica en gran medida el Hardware del simulador.

Otro punto importante de esta filosofía es la gran facilidad de un sistema HIL de convertirse de un simulador a un controlador. Lo anterior es cierto debido a que lo que define a un sistema HIL es básicamente su software, no su hardware. Basta un cambio de programa y el comportamiento del sistema puede ser redefinido; el lugar en el lazo de control es ahora como controlador y no como la planta. Los dos únicos puntos que hay que tener en cuenta son: primero, que un simulador HIL demandará mayor poder de cómputo que su respectivo controlador en el mismo lazo cerrado y segundo, que el número de entradas y salidas sea suficiente para realizar las conexiones respectivas.

1.4 DISPOSITIVOS COMERCIALES DE SIMULADORES HIL

Como se ha podido observar hasta este punto, es posible realizar un sistema HIL de múltiples formas y de una manera muy general se puede clasificar a los sistemas como se muestra en el árbol de la *Figura 1.3*:

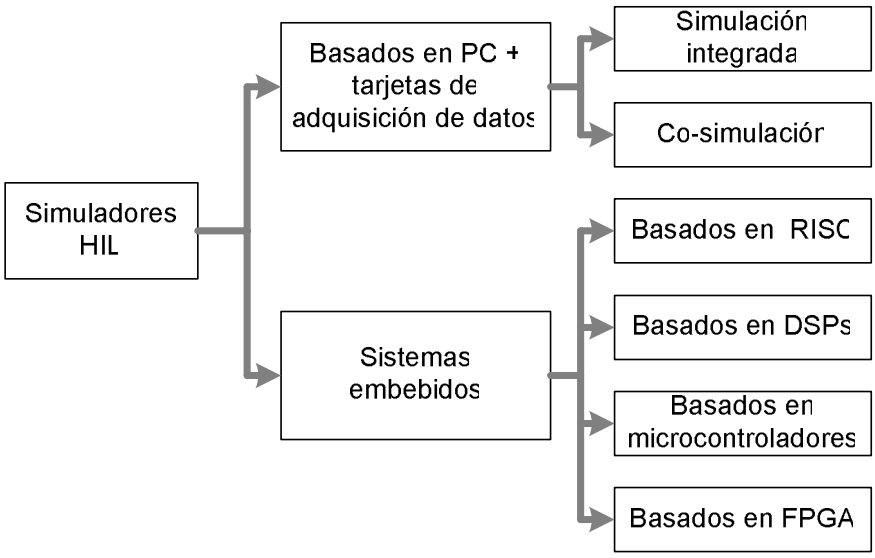


Figura. 1.3. Árbol de clasificación de simuladores HIL

Los dos grandes grupos son: sistemas basados en PC y Sistemas embebidos. En el primer grupo se utiliza el hardware común y barato de una computadora personal la cual es equipada con tarjetas de adquisición de datos. Este método es uno de los más usados por su economía y poder de cómputo [*1*]. El modelo corre generalmente sobre un RTOS proporcionado por el fabricante del producto. En sistemas más sofisticados se utiliza co-simulación que involucra el uso de procesadores complementarios para la ejecución de los modelos.

El segundo grupo pertenece a sistemas embebidos y se relaciona con todos los equipos que son capaces de funcionar independientemente. Su uso como simuladores HIL surge por la necesidad de contar con características especiales no incluidas en una PC, pueden ser más baratos y usados para realizar pruebas

de la tecnología; muchos de los fabricantes de DSPs, procesadores, microcontroladores y FPGA, proporcionan tarjetas de evaluación de sus productos, lo que los hace ampliamente disponibles. Adicionalmente estos productos son, en la mayoría de casos, usados como controladores embebidos, pero como se indicó anteriormente, esto también los vuelve susceptibles a ser simuladores HIL. En la Tabla 1.1 se resumen algunos de los fabricantes de soluciones para simulación HIL [*6*], [*7*], [*8*], [*9*], [*10*].

1.5 CONFIGURACIÓN DE SIMULADORES HIL

1.5.1 Trayectoria de diseño

Es común en la realización del diseño de un controlador el seguir varios pasos los cuales no involucran necesariamente una secuencia estricta y definida pero que se puede resumir en los siguientes pasos [*4*] (*Figura 1.4*):

- Modelar la planta y el controlador; verificarlos por simulación.
- Implementar el controlador y verificarlo por simulación.
- Realización final: prototipo y planta real; validación, mediciones y pruebas.

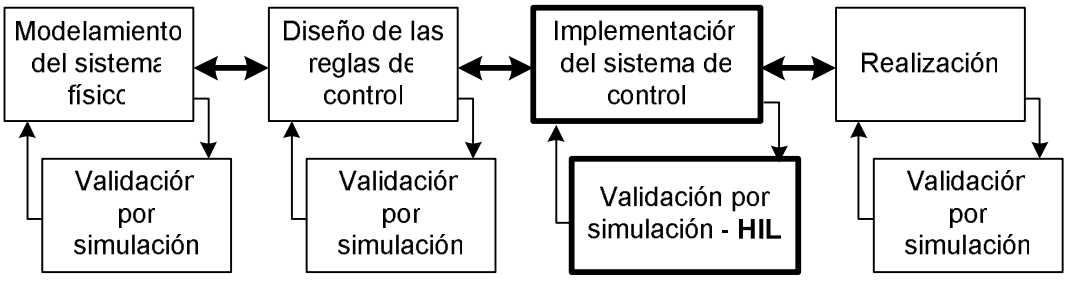


Figura. 1.4. Trayectoria de diseño

Producto / Fabricante	Campo de aplicación	Hardware de destino	Entorno de diseño	Comentarios
xPC Target *Mathworks*	Simulación de cualquier sistema	PC	Simulink, Stateflow	Permite a cualquier PC trabajar como simulador HIL. Trabaja con un pequeño RTOS. Gran cantidad de tarjetas de adquisición de datos para conectar. Altamente difundido.
LabVIEW Simulation Interface Toolkit + *National Instruments*	Simulación de cualquier sistema	PC, DSP, FPGA	Simulink LabVIEW	NI tiene una gran gama de dispositivos de adquisición de datos e interfases. El fabricante da la opción de trabajar en sistemas basados en PC o equipos embebidos.
VisSim *Visual Solution Incorporated*	Simulación en general, orientado a procesos industriales	PC, (DSP)	VisSim	VisSIm es un equivalente a Simulink de Matlab y es compatible con el mismo. También trabaja con MathCad. Ofrece su propia gama de tarjetas de adquisición.
dSPACE Simulator *dSPACE*	Especializado en automóviles y aviónica	DSP	Simulink, TargetLink	Es uno de los fabricantes más antiguos de simuladores HIL. Los equipos propietarios son necesarios para la simulación HIL.
RT-LAB *OPAL-RT*	Especializado en automóviles y aviónica	PC, FPGA	Simulink , RT-LAB, MATRIXx	Posee una serie de hardware y software especializado para una amplia gama de aplicaciones. Las simulaciones se pueden realizar con varias computadoras en red.
Real-Time Workshop Embedded Coder *Mathworks*	Cualquier sistema, modelos simples o complejos dependiendo del hardware	DSP, RISC, uC	Simulink	El poder de esta herramienta esta en convertir los diagramas de bloques de Simulink en código ANSI C eficiente para dispositivos con pocos recursos de memoria y procesador. En teoría, es posible acoplarlo a cualquier dispositivo con compilador en C.

Tabla. 1.1. Dispositivos comerciales de simuladores HIL

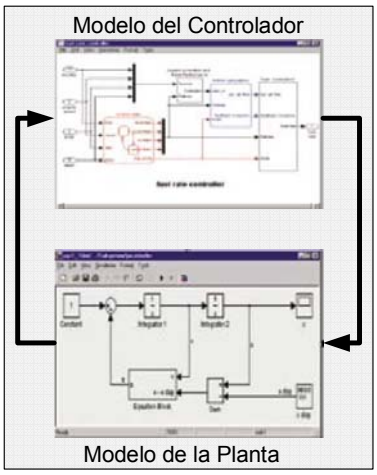
La cierta universalidad de la manera en que se diseña un sistema de control determina la forma en que se genera la configuración de un simulador HIL.

1.5.2 Creación de la configuración de un emulador HIL

1.5.2.1 Generación de código según un modelo en diagrama de bloques [*11*]

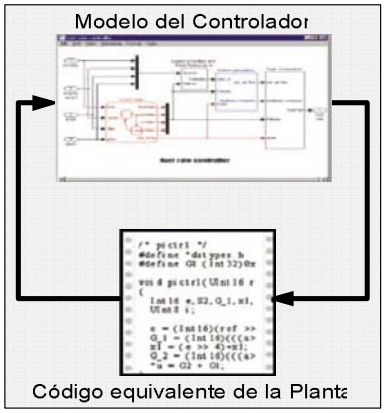
Este primer método que se presenta es el más difundido en la creación de simuladores HIL, y se divide en 4 pasos básicos: MIL, SIL, PIL y HIL. La razón de utilizar este método es que se puede generar el código del sistema que uno se plantee a partir de una representación de diagramas de bloques como es Simulink.

MIL – Modelo en el Lazo



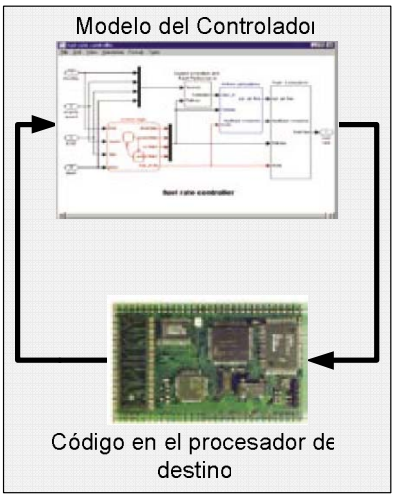
Esta es la etapa inicial del desarrollo de un controlador y consiste básicamente en realizar las simulaciones en un ambiente como Simulink de MATLAB. Tanto el modelo del controlador como de la planta se encuentran expresados en bloques. Se obtienen las gráficas y datos del comportamiento del sistema.

SIL – Software en el Lazo



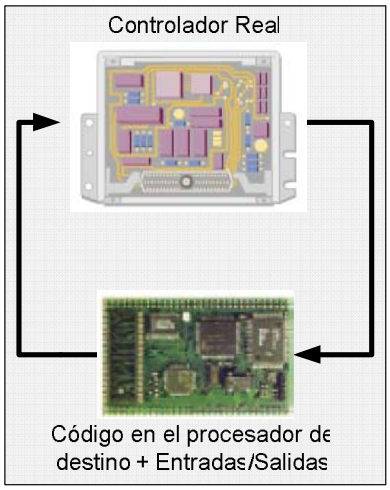
Con la ayuda de un codificador, se convierte el modelo en bloques a su equivalente en código, generalmente C. Este código es simulado como si se tratase de un bloque más, pero esta vez se lo ve como una caja negra a nivel del modelo global. Se verifica si los gráficos y el comportamiento son similares a los obtenidos con el MIL.

PIL – Procesador en el Lazo



Posteriormente se compila el código en C generado en SIL con la ayuda del correspondiente compilador del procesador y se lo descarga al procesador de destino. Usando mecanismos de comunicación se inyectan y se extraen datos del procesador en tiempo real para verificar que los datos son congruentes a los obtenidos en MIL y SIL.

HIL – Hardware en el Lazo



Finalmente, se integra al simulador HIL con el controlador real. En la mayoría de casos, el controlador real posee una interfase de comunicaciones que permite observar el estado de las variables de tal forma que se pueda comparar con los resultados obtenidos con MIL, SIL y PIL.

De los cuatro pasos mencionados, el primero y el último son necesarios, siendo los pasos segundo y tercero implícitos muchas veces.

1.5.2.2 Configuración con variación de parámetros – plantillas

Este método es el menos utilizado pero resulta útil cuando se desea realizar un simulador HIL a partir de un procesador o microcontrolador que no posee la compatibilidad con las herramientas de generación de código. Este es el caso de nuevos procesadores o microcontroladores con poca capacidad.

La ventaja de este mecanismo es que el diseñador crea todo el código para el procesador lo que puede llevar a una mayor optimización y acoplamiento con la arquitectura del procesador aunque esto también involucra un tiempo significativo de desarrollo. En este caso lo que se crea es un sistema embebido capaz de comunicarse con una computadora de tal manera que la Interfaz gráfica pase los parámetros que el usuario escoja. Básicamente la GUI es una plantilla que pasa los parámetros al procesador, el cual los usa como variables en el código que tiene predefinido (*Figura 1.5*).

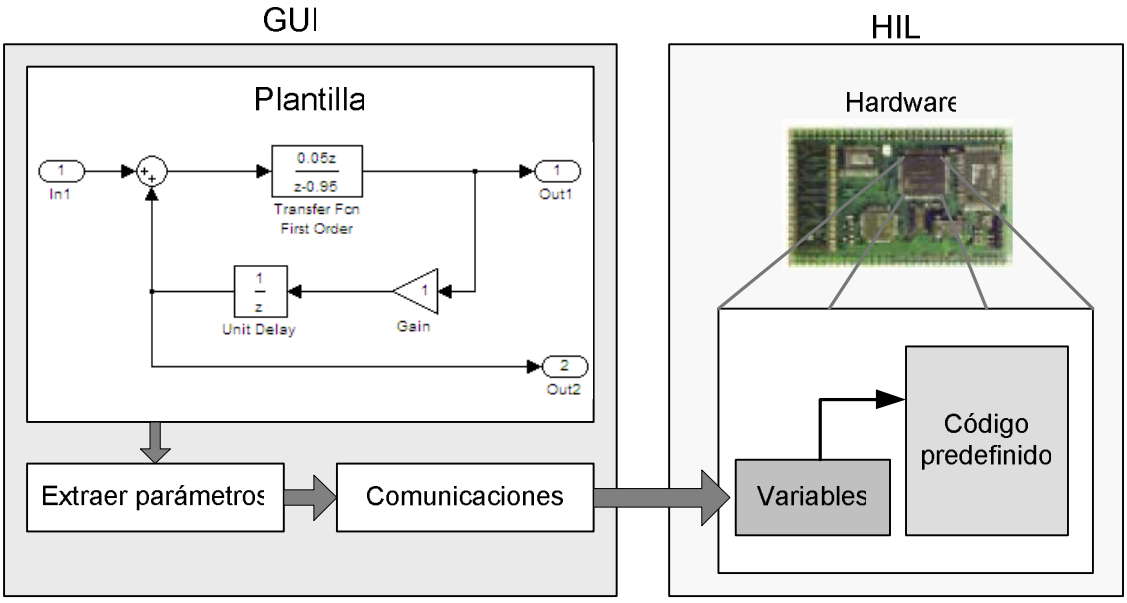


Figura. 1.5 Configuración tipo plantilla

Cabe destacar que puede haber muchas variaciones en la forma que se puede crear el software para un sistema HIL, sin embargo se han detallado dos maneras, la primera por ser la más utilizada y la segunda por ser la forma en que se realizó este proyecto.

CAPÍTULO 2

SIMULACIÓN DE PLANTAS

2.1 DESCRIPCIÓN DE LOS SISTEMAS LINEALES INVARIANTES EN EL TIEMPO

2.1.1 Origen de las ecuaciones diferenciales

Las dinámicas de muchos sistemas, sean mecánicos, eléctricos, térmicos, económicos, biológicos u otros, se pueden describir en términos de ecuaciones diferenciales. Esas ecuaciones diferenciales pueden obtenerse utilizando las leyes físicas que rigen un sistema particular. Pueden ser las leyes de Newton para sistemas mecánicos o las leyes de Kirchhoff para sistemas eléctricos. La respuesta de un sistema dinámico a una entrada puede obtenerse si se resuelven las ecuaciones diferenciales que modelan dicho sistema.

Una de las partes más importante en el diseño de sistemas de control es tener o encontrar un buen modelo matemático que represente el sistema que se desea controlar. Por lo general, mientras más exactitud se desea de un modelo, más complejidad matemática tendrá este. Sin embargo es común el tratar de simplificar el modelo para lograr una descripción del sistema por medio de ecuaciones diferenciales ordinarias. Esto es conveniente para el análisis pero a su vez limita la validez del modelo a ciertos rangos de operación como puede ser bajas frecuencias o temperaturas normales [*12*].

Las ecuaciones diferenciales, en la ingeniería de control, es la forma matemática que permite describir el comportamiento de un sistema con respecto

al tiempo. Estas pueden tener un sinnúmero de formas y métodos para resolverlas, pero en este trabajo solo se presentarán aquellas que son lineales e invariantes en el tiempo. La obtención de estas ecuaciones se las realiza por el modelamiento del sistema en cuestión y se basa en las respectivas leyes físicas que lo gobierna.

2.1.2 Sistemas lineales

Un sistema se denomina lineal si se aplica el principio de superposición (*Figura 2.1*A) y homogeneidad (*Figura 2.1B*). El primero establece que la respuesta producida por la aplicación simultánea de dos funciones excitadoras distintas, es la suma de las dos respuestas individuales. Esto significa que para sistemas lineales la respuesta a diversas entradas puede ser calculada tratando una entrada a la vez y sumando los resultados. Este principio permite elaborar soluciones complicadas de las ecuaciones diferenciales lineales partiendo de soluciones simples.

Otro requerimiento de los sistemas lineales es la homogeneidad, que significa que un cambio en la amplitud de la función excitadora resulta en un cambio de magnitud correspondiente en la respuesta [*13*].

2.1.3 Sistemas lineales invariantes en el tiempo y variables en el tiempo

Una ecuación diferencial es lineal si los coeficientes son constantes o funciones de la variable independiente exclusivamente. Los sistemas dinámicos que son lineales y están constituidos por componentes concentrados lineales e invariantes en el tiempo, pueden describirse por ecuaciones diferenciales lineales invariantes en el tiempo (de coeficientes constantes): A estos sistemas se los denomina sistemas lineales invariantes en el tiempo (LTI). Los sistemas representados por ecuaciones diferenciales cuyos coeficientes son funciones del tiempo, reciben el nombre de sistemas lineales variables en el tiempo.

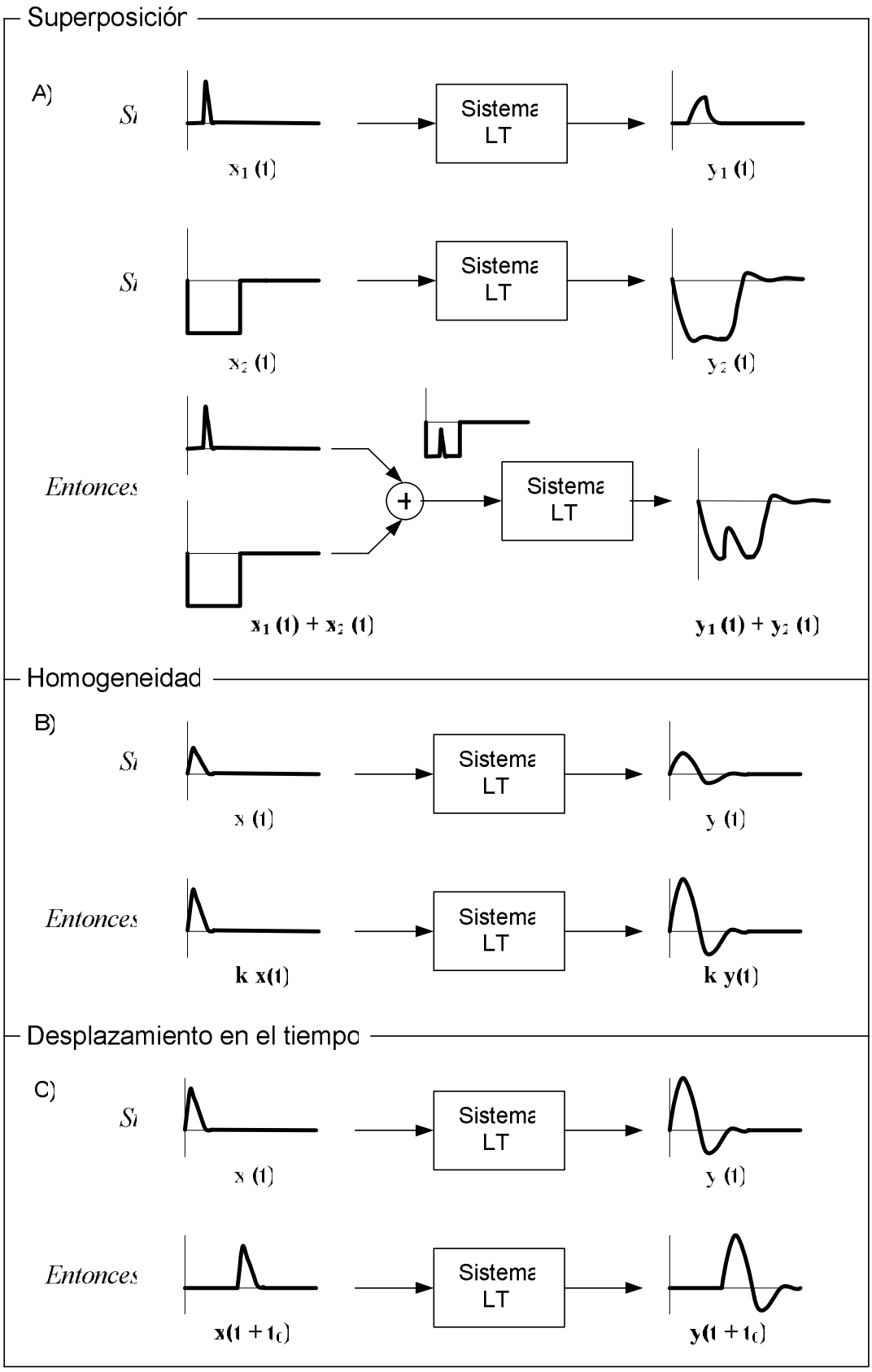


Figura. 2.1. Propiedades sistemas LTI

Para los sistemas LTI se cumple que son invariantes al desplazamiento en el tiempo (*Figura 2.1C*), lo que significa que un desplazamiento de la función de entrada resultará en un desplazamiento idéntico en la función de salida.

2.1.4 Función de transferencia

La denominada función de transferencia de un sistema se utiliza para caracterizar las relaciones entrada – salida de componentes o sistemas que pueden describirse por ecuaciones diferenciales lineales, invariantes en el tiempo. La manera más utilizada para resolver estas ecuaciones es el método de la transformada de Laplace que se puede encontrar en cualquier libro de control o ecuaciones diferenciales.

Por esta razón, la ecuación de transferencia se define como la relación entre la transformada de Laplace de la salida y la transformada de Laplace de la entrada, bajo la suposición de que todas las condiciones iniciales son cero.

Para la siguiente ecuación diferencial lineal invariante en el tiempo:

$$b_0 \frac{d^n y(t)}{dt^n} + b_1 \frac{d^{n-1} y(t)}{dt^{n-1}} + \cdots + b_n y(t) =$$

$$a_0 \frac{d^m x(t)}{dt^m} + a_1 \frac{d^{m-1} x(t)}{dt^{m-1}} + \cdots + a_m x(t) \quad (n \geq m) \qquad \text{(2-1)}$$

donde y es la salida del sistema y x es la entrada, la función de transferencia se obtiene tomando las transformadas de Laplace de ambos miembros de la ecuación (2-1), bajo la suposición de que todas las condiciones iniciales son cero, o sea

$$\textit{Función de transferencia} = G(s) = \left.\frac{L[salida]}{L[entrada]}\right| \text{ condiciones iniciales en cero}$$

$$= \frac{Y(s)}{X(s)} = \frac{a_0 s^m + a_1 s^{m-1} + \cdots + a_{m-1} s + a_m}{b_0 s^n + b_1 s^{n-1} + \cdots + b_{n-1} s + b_n} \qquad \text{(2-2)}$$

Con esta fórmula se puede expresar la dinámica de un sistema por ecuaciones algebraicas en s. Si la potencia más alta de s en el denominador de la función de transferencia es igual a n, se dice que el sistema es de orden n.

2.2 TRABAJO EN TIEMPO DISCRETO

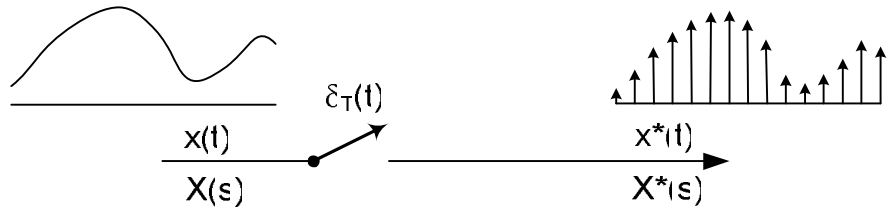
2.2.1 Discretización de sistemas continuos

Como se explicó en el Capítulo 1, los simuladores HIL trabajan en el dominio digital o discreto aunque sus interfaces sean analógicas. Lo mismo ocurre con el presente emulador SISO por lo que es lógico tener una buena idea de la forma en que se genera la función de transferencia en tiempo discreto.

Para poder realizar la conversión de un sistema continuo a uno discreto es necesario conocer el tipo de muestreo, o proceso, que se siguió en la conversión analógica – digital. La primera forma (*Figura 2.2A*) es obtener los valores de la señal por medio de un tren de impulsos equidistantes en el tiempo, los cuales marcan los puntos de la señal continua que se convierten en la secuencia discreta. Sin embargo es difícil para los dispositivos electrónicos lograr capturar la señal de una manera semejante a la planteada, por lo que se tiene el método de retención de datos.

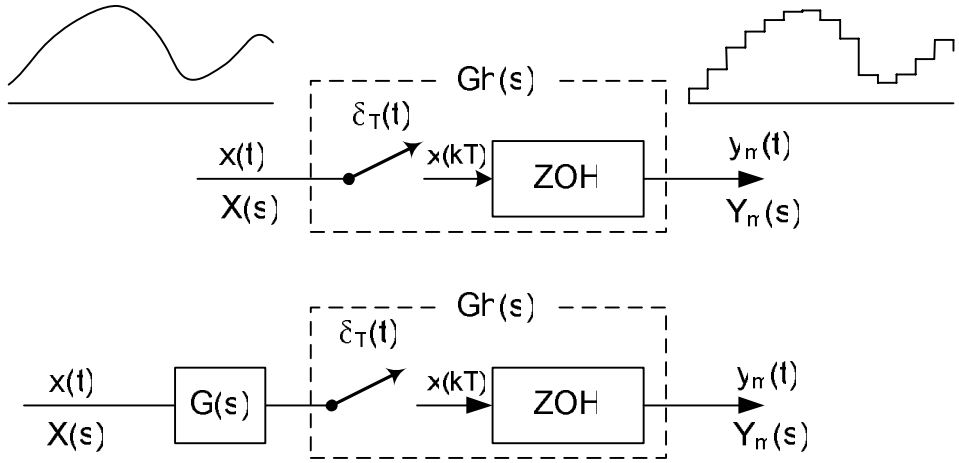
El más importante es el de orden cero (*Figura 2.2B*), debido a su practicidad a la hora de implementar. Este consiste en suavizar la señal muestreada para producir la señal de salida que se mantiene constante desde el último valor muestreado hasta que se puede disponer de la siguiente muestra [*14*].

A) Muestreo por impulsos



$$X^*(s) = \sum_{k=0}^{\infty} x(kT)e^{-kTs}$$

B) Retenedor de orden cero



$$Gh(s) = \frac{1-e^{-Ts}}{s} \; ; \qquad Ym(s) = \frac{1-e^{-Ts}}{s} \cdot G(s) \cdot X(s) \; ;$$

Figura. 2.2. Tipos de muestreos

Todas las conversiones de funciones continuas a discretas se realizarán utilizando un retenedor de orden cero (ZOH). El otro parámetro necesario y extremadamente importante es el tiempo de muestreo con el que se adquieren los datos. Este punto se tratará con detalle más adelante en este capítulo.

Una vez aclarado el proceso de conversión A/D, se precisa la formulación para pasar una función en el dominio s (continuo) al dominio z (discreto). Como se puede ver en la *Figura 2.2*, el retenedor de orden cero modifica la señal de entrada.

Tenemos por tanto:

$$Ym(s) = Gh(s) \cdot G(s) \cdot X(s) \quad ; \; Ym = \text{Señal de salida muestreada} \tag{2-3}$$

Para el caso del retenedor ZOH se tiene que:

$$Ym(s) = \frac{1 - e^{-Ts}}{s} \cdot G(s) \cdot X(s) \quad ; \; Ts = \text{tiempo de muestreo} \tag{2-4}$$

$$\frac{Ym(s)}{X(s)} = Gm(s) = \frac{1 - e^{-Ts}}{s} \cdot G(s) \quad ; \; Gm = \text{función de transferencia del sistema muestreado} \tag{2-5}$$

A la ecuación (2-5) se le aplica la transformada z ($\mathcal{Z}$) y se encuentra la forma general de obtener la salida discreta:

$$Gm[z] = \mathcal{Z}[Gm(s)] = \left(1 - z^{-1}\right) \mathcal{Z}\left[\frac{G(s)}{s}\right] \tag{2-6}$$

La ecuación resultante $Gm[z]$ resulta la ecuación de transferencia discreta:

$$Gm[z] = \frac{Y[z]}{X[z]} = \frac{a_{d0} + a_{d1} z^{-1} + \cdots + a_{d(j-1)} z^{-(j-1)} + a_{dj} z^{-j}}{1 - b_{d1} z^{-1} - \cdots - b_{d(i-1)} z^{-(i-1)} - b_{di} z^{-i}} \tag{2-7}$$

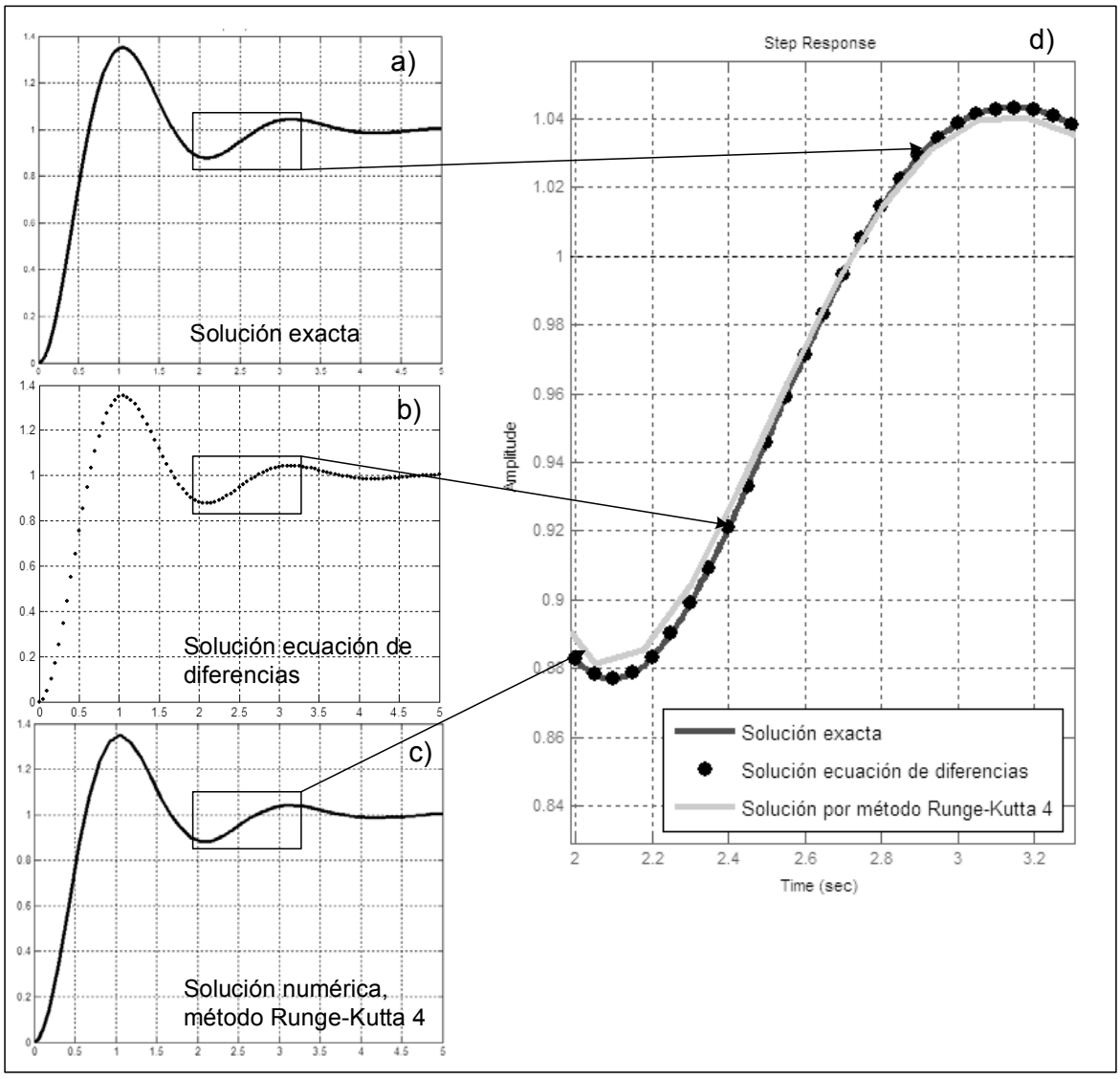
2.2.2 Ecuación de diferencias

A la ecuación (2-7) se le aplica la transformada Z inversa ($\mathcal{Z}^{-1}$) y se obtiene la ecuación de diferencias correspondiente que tiene la forma [*13*]:

$$y[n] = a_{d0}x[n] + a_{d1}x[n-1] + \cdots + a_{d(j-1)}x[n-(j-1)] + a_{dj}x[n-j] + b_{d1}y[n-1] + \cdots + b_{d(i-1)}y[n-(i-1)] + b_{di}y[n-i] \qquad (2\text{-}8)$$

La ecuación (2-8) resulta muy útil porque permite implementar la ecuación de transferencia en un algoritmo digital. El cómputo de ésta ecuación arroja datos exactos respecto a la solución (*Figura 2.3*) en cada punto discreto y no datos aproximados como los que producen los métodos de integración numérica usada para resolver ecuaciones diferenciales como son los de Euler y Runge-Kutta [*15*]. En la mayoría de simuladores HIL, se utiliza alguno de los métodos de integración numérica ya que ésta es la manera en que Simulink obtiene las soluciones a los diagramas de bloques. La ventaja al trabajar con integración numérica es la posibilidad de resolver ecuaciones diferenciales ordinarias y no limitarse a sistemas LTI [*16*]. *El emulador SISO de este trabajo solo trabajará con ecuaciones lineales invariantes en el tiempo, por lo que utiliza la ecuación de diferencias para implementar la función de transferencia.*

Es necesario aclarar que para lograr resultados satisfactorios al implementar filtros IIR (especialmente de orden mayor a 2), es necesario realizar los cálculos en punto flotante de doble precisión caso contrario el filtro no arrojará los resultados apropiados con considerables errores o incluso puede volverse inestable. Esto se debe a que en punto flotante de precisión simple o, peor aún, punto fijo, los errores de cada multiplicación se van acumulando exponencialmente debido a que se trabaja con muestras pasadas de la salida.



Comparación de las soluciones a la ecuación diferencial $y''+2y+10y=10x$ ante el estímulo de un escalón unitario. En la gráfica a) se tiene la solución exacta $y(t)=1-\frac{e^{-t}\left(3\cos(3t)+sen(3t)\right)}{3}$, en la gráfica b) la respuesta al escalón de la ecuación de diferencias equivalente con un Ts = 0.05 y en c) la solución a la ecuación diferencial usando integración numérica con el método Runge-Kutta 4. Hay que mencionar que intencionalmente se dio una tolerancia alta al método de integración numérica para poder notar la diferencia. Este método puede ser muy exacto si se dan tolerancias bajas de error.
Como se puede apreciar, la solución de la ecuación de diferencias arroja soluciones exactas a la ecuación diferencial en los puntos discretos correspondientes, cosa que no pasa con la integración numérica.

Figura. 2.3. Comparación de métodos de cálculo de ecuaciones diferenciales.

La ecuación (2-8) mencionada también es conocida en el campo del procesamiento digital de señales como un filtro IIR, ya que su respuesta a un impulso da una respuesta infinita. Esto se debe a que su valor actual de salida depende tanto de muestras pasadas de la entrada como muestras pasadas de la salida; es decir, es un filtro recursivo [*17*].

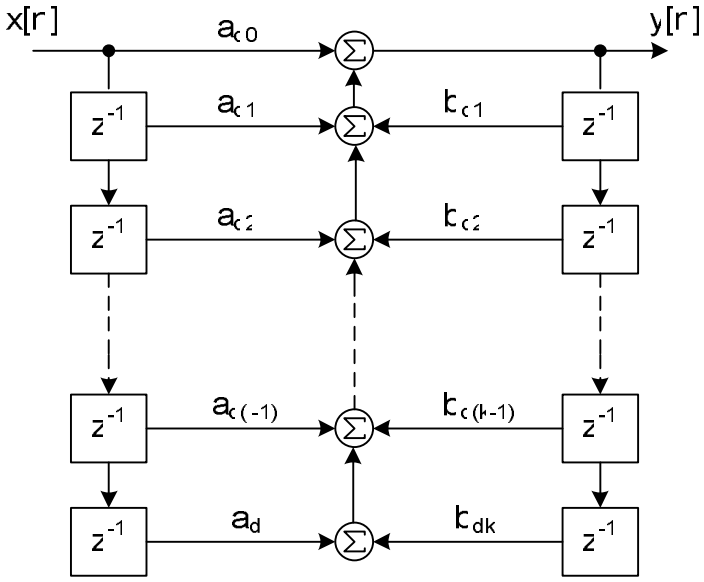


Figura. 2.4. Flujo de un filtro IIR

2.3 EMULACIÓN DE UN SISTEMA CONTINUO EN TIEMPO DISCRETO

2.3.1 Secuencia de adquisición de datos

En la *Figura 2.5* se muestra las etapas de entrada y salida de un simulador HIL. La señal de entrada al simulador tiene un contenido de frecuencia con ancho de banda *fi* (I). Debido a que no todo este espectro es información necesaria, se pasa esta señal por el filtro antialias (II) cuya principal característica es su frecuencia de corte *fc*, que a su vez es la frecuencia máxima que se estima generará el controlador. Por el teorema de Nyquist, se selecciona una frecuencia de muestreo (*fs*), tal que no se produzca traslape de frecuencias (o aliasing) de parte de la imagen de la señal muestreada a la banda de interés (III).

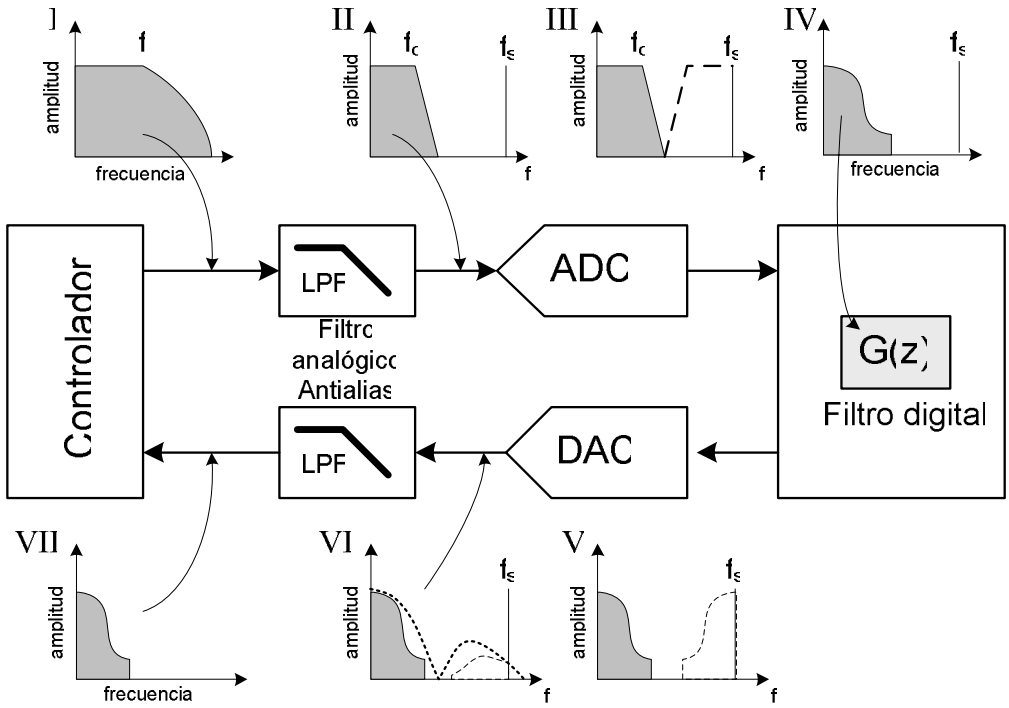


Figura. 2.5. Flujo de la señal en un simulador HIL

2.3.2 Secuencia de reproducción de la simulación

Una vez filtrada y digitalizada la señal de entrada, se la procesa digitalmente, que en este caso consiste en pasarla por un filtro IIR (la función de transferencia). Este proceso se caracteriza por tener una respuesta en frecuencia (IV), limitada por la frecuencia de simulación que supondremos es la misma que la de muestreo del ADC. La señal digital que se obtiene del filtro G(z) se transforma de nuevo a analógica con ayuda del DAC (V). El DAC genera componentes en alta frecuencia (VI) y componentes no deseadas por debajo de *fs/2*, por lo que se filtra analógicamente para suprimir estas componentes (VII).

2.3.3 Efectos de la conversión Analógica-Digital y Digital-Analógica

La conversión Analógica-Digital y Digital-Analógica es el proceso que permite a las computadoras digitales interactuar con las señales físicas, en la mayoría de casos voltajes y corrientes (que a su vez representan otras variables). La información digital es diferente de su contraparte analógica en dos aspectos importantes: se encuentra muestreada y cuantizada. Esto restringe cuánta información puede contener una señal digital. La cuantización se refiere a que un valor está representado por N bits que a su vez representa 2^N estados posibles, implicando que un DAC lineal de N bits puede tener únicamente 2^N posibles salidas analógicas y un ADC (lineal) solo 2^N valores que representan la entrada.

En la *Tabla 2.1* se encuentra la resolución del bit menos significativo para conversores de distinto número de bits (N).

RESOLUTION N	2^N	VOLTAGE (10V FS)	ppm FS	% FS	dB FS
2-bit	4	2.5 V	250,000	25	-12
4-bit	16	625 mV	62,500	6.25	-24
6-bit	64	156 mV	15,625	1.56	-36
8-bit	256	39.1 mV	3,906	0.39	-48
10-bit	1,024	9.77 mV (10 mV)	977	0.098	-60
12-bit	4,096	2.44 mV	244	0.024	-72
14-bit	16,384	610 µV	61	0.0061	-84
16-bit	65,536	153 µV	15	0.0015	-96
18-bit	262,144	38 µV	4	0.0004	-108
20-bit	1,048,576	9.54 µV (10 µV)	1	0.0001	-120
22-bit	4,194,304	2.38 µV	0.24	0.000024	-132
24-bit	16,777,216	596 nV*	0.06	0.000006	-144

*600nV is the Johnson Noise in a 10kHz BW of a 2.2kΩ Resistor @ 25°C

Tabla. 2.1. Tamaño del bit menos significativo

El rango dinámico se define como el peso del bit menos significativo expresado en db y es igual a $FS = 20 \cdot \log\left(\frac{1}{2^N}\right) = -6.02 \cdot N$.

En las siguientes secciones se explica el efecto del muestreo.

2.3.4 Parámetros para elegir la frecuencia de simulación

2.3.4.1 Efectos del retenedor de orden cero

La frecuencia de simulación es la inversa del tiempo entre ejecuciones seguidas del filtro digital IIR que representa la función de transferencia. En este punto hay que definir *fm* la cual es la frecuencia máxima que se desea simular con fidelidad. Por tanto la pregunta que se busca contestar es: ¿a qué frecuencia de simulación (del emulador HIL) el controlador lee una frecuencia simulada *fm* con buena fidelidad?

La tendencia sería aplicar el criterio de Nyquist y señalar que al simular al doble de la frecuencia *fm* se logra la fidelidad deseada, pero hay que estar concientes que tanto el ADC como el DAC son de orden cero y alteran la forma de onda [*18*] y que desconocemos la frecuencia de muestreo del controlador. Esto se puede observar en la *Figura 2.6*, a pesar de que la señal senoidal es generada a 3 veces más de la frecuencia *f* original, la onda que se tiene no es similar a la original especialmente si suponemos que el controlador lee (hipotéticamente) en tiempo continuo la onda. Es lógico pensar que debe existir otro criterio más para elegir la frecuencia de muestro o reproducción con el fin de lograr una buena fidelidad.

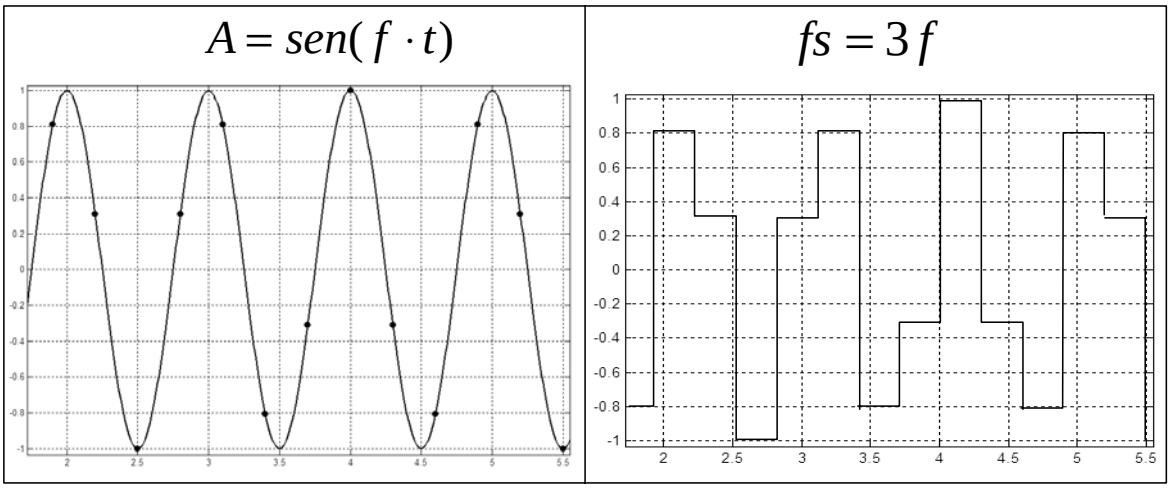


Figura. 2.6. Señal muestreada con fs = 3f

Como se puede apreciar en la *Figura 2.7a*, el espectro de una señal discretizada por un tren de impulsos tiene el espectro mostrado [*13*]. En esta el espectro no se pierde si se cumple el criterio de Nyquist. Por el contrario, el espectro de un retenedor ZOH es el valor absoluto de la funcion sinc (*Figura 2.7b*), que modifica la magnitud del espectro original, especialmente en las zonas cercanas a *fs/2*.

$$H(f) = \left| \frac{sen\left(\pi \, f / fs\right)}{\pi \, f / fs} \right| \qquad (2\text{-}9)$$

fm es la mayor frecuencia que se desea simular y como se aprecia ésta sufre un error dependiendo de su ubicación con respecto a *fs/2*. Mientras menor sea la relación *fm/fs*, menor error se tendrá por el retenedor de orden cero. En la *Tabla 2.2* se muestran varios valores de error según la relación *fm/fs*. Se considera que $f_s = 1$.

Relación f_m/f_s	Relación f_s/f_m	Amplitud $sinc(\pi f_m/f_s)$	Error %	Error dB
0.010	100	0.99984	0.016%	-0.00138
0.020	50	0.99934	0.066%	-0.00573
0.050	20	0.99589	0.411%	-0.03577
0.066	15	0.99271	0.729%	-0.06355
0.078	12.806	0.99000	1.000%	-0.08729
0.100	10	0.98363	1.637%	-0.14336
0.200	5	0.93549	6.451%	-0.57921
0.300	3.333	0.85839	14.161%	-1.32631
0.400	2.5	0.75683	24.317%	-2.42003
0.500	2	0.63662	36.338%	-3.93741

Tabla. 2.2. Valores de la función *sinc* y error respecto a 1 (0dB)

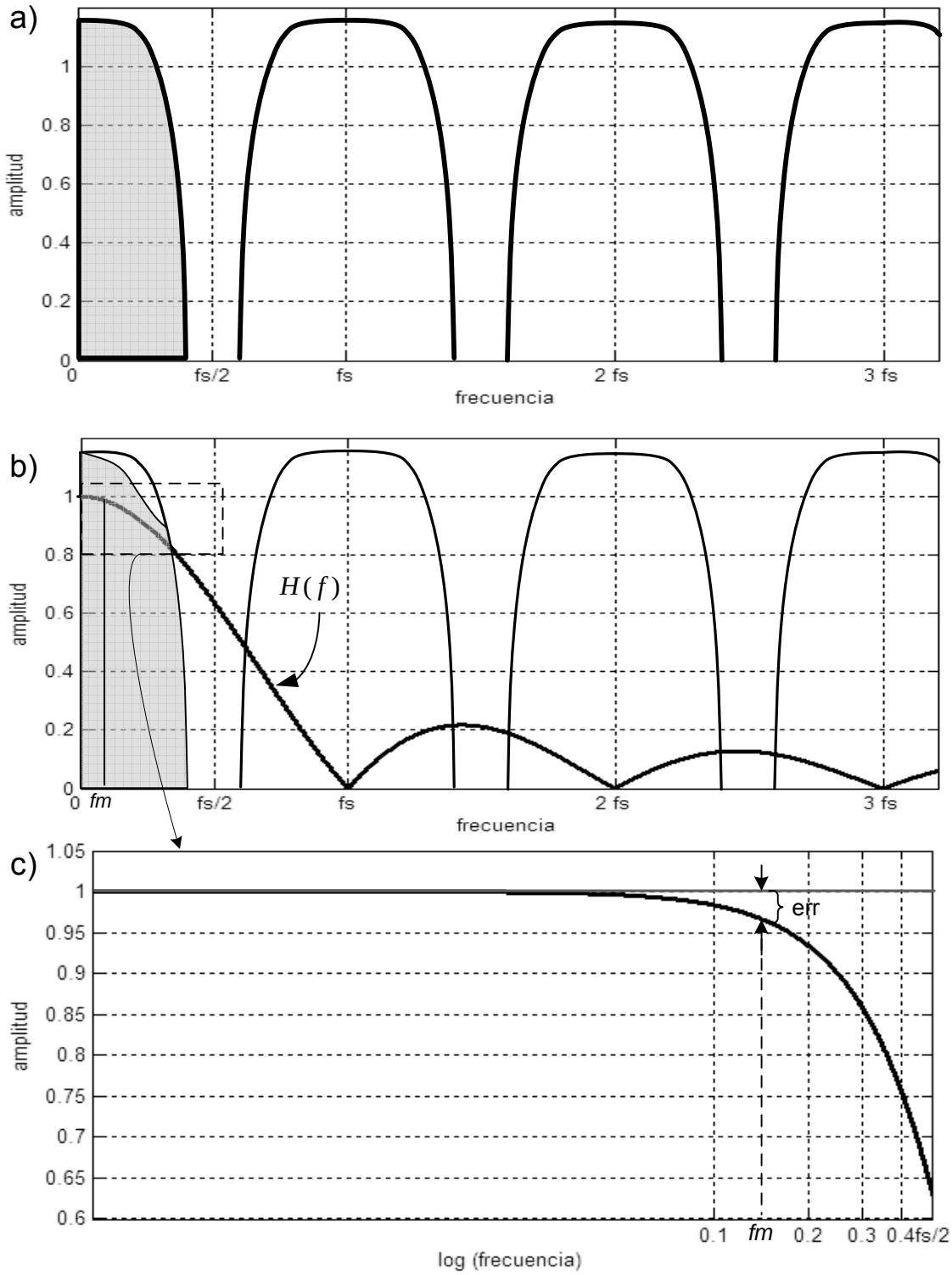


Figura 2.7. Respuesta en frecuencia del retenedor de orden cero

La *Figura 2.7c* es un acercamiento a la zona de interés ($0 < f < f_s/2$) en la cual se puede apreciar con mayor detalle los efectos del retenedor. Los valores de la *Tabla 2.2* indican que el mayor error se tiene cuando la simulación trabaja de 2 a 10 veces más que la frecuencia *fm*. Sin embargo es notable que con 12.8 veces *fm* se tenga un error de apenas 1%. A partir de este valor ($12.8fm$), la mejora no es notable e involucra valores altos de frecuencia de simulación. Con estos resultados se puede balancear el error tolerable con la potencia de cálculo del simulador; si se trabaja con frecuencias más altas el simulador deberá completar todos los cálculos referentes a la planta en menor tiempo, lo que involucra contar con un procesador de mayor potencia. Esta lógica está estrechamente ligada a la complejidad del sistema y por ende a los costos del hardware final del simulador HIL.

2.3.4.2 Error introducido por los filtros antialias

Como se puede ver en la *Figura 2.7b* es necesario suprimir las frecuencias que se encuentran sobre *fs/2*. Para esto se debe incluir un filtro analógico a la entrada y a la salida del ADC y el DAC. Por el momento se tomará en cuenta solo la frecuencia de corte del filtro (*fc*) que por definición es aquella en la cual se tiene -3db de ganancia respecto a 0db de ganancia en la banda de paso. En el capítulo 3 se tratará con más detalle la obtención del filtro, pero por ahora se supone que el filtro es uno tipo Bessel de segundo orden. La meta es saber a qué frecuencia de corte hay que fijar los filtros antialiasing para que cumplan efectivamente su trabajo. Aquí nos encontramos con la necesidad de balancear dos características: si la frecuencia de corte se encuentra muy desplazada hacia altas frecuencias, es probable que el filtro no alcance a suprimir las altas frecuencias antes de *fs/2*; por otro lado si *fc* se fija muy cerca de *fm*, introduce un error en la simulación. Según la aplicación se debe establecer cuánto afecta el filtro antialias a la frecuencia *fm*, sabiendo que mientras menos se afecte a *fm*, menos atenuación logrará el filtro antialias en *fs/2*. La capacidad de impedir aliasing por parte de estos filtros

dependerá del espectro de la señal de entrada y el espectro de la señal generada por el filtro recursivo IIR que representa la función de transferencia.

2.3.4.3 Selección de la máxima frecuencia de simulación (*fm*)

Es necesario saber cuál es la frecuencia *fm* para poder determinar la frecuencia de muestreo. Por lo general se dicta, mayormente por experiencia que por análisis teórico, que la frecuencia máxima que se simula sea de 10 a 25 veces la frecuencia que define el ancho de banda del sistema en lazo cerrado [*19*]. Esto implica la necesidad de conocer la respuesta en frecuencia de la función de transferencia. Se toma el ancho de banda (-3db) por ser el punto que marca las regiones donde se tiene el mayor contenido de energía de la señal.

En la *Figura 2.8* se puede observar dos representaciones típicas de respuesta en frecuencia de funciones de transferencia comunes a plantas reales. La gráfica a) corresponde a la mayoría de plantas físicas de temperatura, posición, flujo,etc. En la gráfica b) se tiene el comportamiento de ciertos sistemas que controlan aceleración u otro tipo de fenómenos. No es común encontrar plantas con energía en los espectros altos de frecuencia.

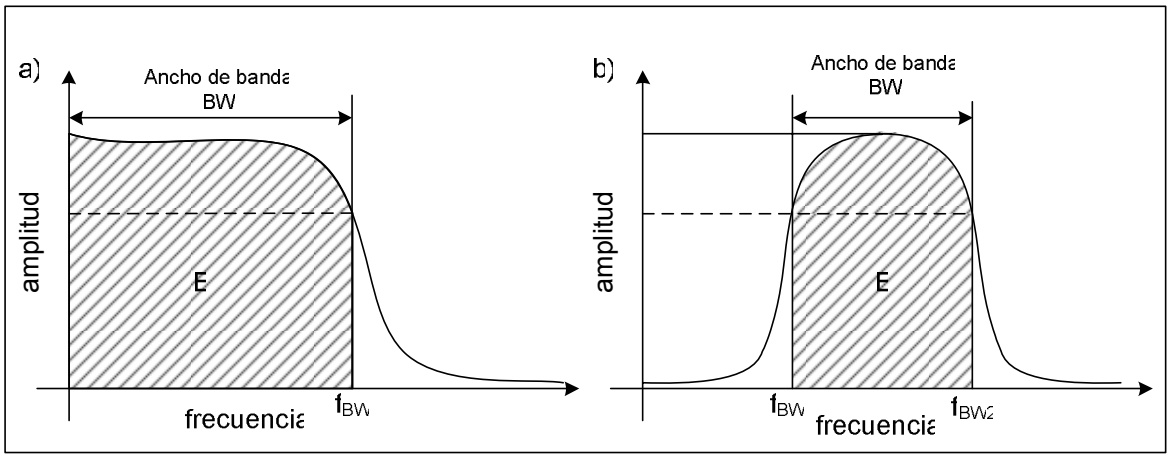


Figura 2.8 Ancho de banda de un sistema

Dependiendo de la respuesta en frecuencia de la función de transferencia unos sistemas pueden decaer con mayor o menor velocidad a la banda de menor energía. Los sistemas de primer orden tienen, en general, una pendiente baja (alrededor de -20dB por década), mientras que sistemas de 4to orden pueden decaer hasta -80dB por década.

2.3.4.4 Resumen de la selección de la frecuencia de muestro

El siguiente procedimiento es una guía para determinar los valores de las frecuencias de muestreo (*fs*), máxima de simulación (*fm*) y la del filtro antialias (*fc*) pero no representa un método exacto para obtenerlos. En el ejemplo del Anexo 1 se indica cómo obtener los valores de estas frecuencias.

1. Una vez conocida la función de transferencia, encontrar la expresión de la misma en lazo cerrado y obtener el diagrama de Bode del cual se obtiene la frecuencia f_{BW} de mayor frecuencia.
2. Buscar la pendiente con que decae la amplitud midiendo en una década la variación que se tiene en la amplitud.
3. Si el valor encontrado de la variación es pequeño (-15dB a -30dB), seleccionar una frecuencia *fm* de 25 a 20 veces f_{BW}, respectivamente.
4. Si el valor encontrado de la variación es grande (-30dB a -80dB), seleccionar una frecuencia *fm* de 20 a 10 veces f_{BW}.
5. De la *Tabla 2.2* se escoge el error tolerable para *fm* y se multiplica por la relación *fs/fm*, cuyo resultado es la frecuencia de muestreo.
6. Posteriormente se debe fijar la frecuencia del filtro antialias entre los valores $fm < fc < 1.7fm$. Si se desea que el filtro antialias modifique lo menos posible a *fm*, se seleccionará un valor de $fc = 1.7fm$, pero si se permite un error adicional de -3dB en *fm* se escogerá $fc = fm$. Hay que tomar en cuenta que mientras más cercana sea *fc* a *fm*, mejor trabajará el filtro antialias suprimiendo las frecuencias mayores a *fs/2*.

2.4 RETARDO DE TRANSPORTE

2.4.1 Generalidades

El retardo de transporte ó tiempo muerto es un fenómeno muy común en procesos industriales ya que está asociado al traslado de materiales desde un punto a otro, demoras en el cómputo de partes del proceso, velocidad de difusión de sustancias, entre otras. Es común ver retardo en sistemas de calentamiento por tuberías, laminado de planchas, procesos de destilación, etc. [*13*], [*20*]. Es por esta razón que es importante incluir este fenómeno en el *emulador SISO*.

En el dominio del tiempo el retardo de transporte no es más que un desplazamiento en el tiempo (*Figura 2.9*). La transformada de Laplace del retardo también se muestra en (2-10).

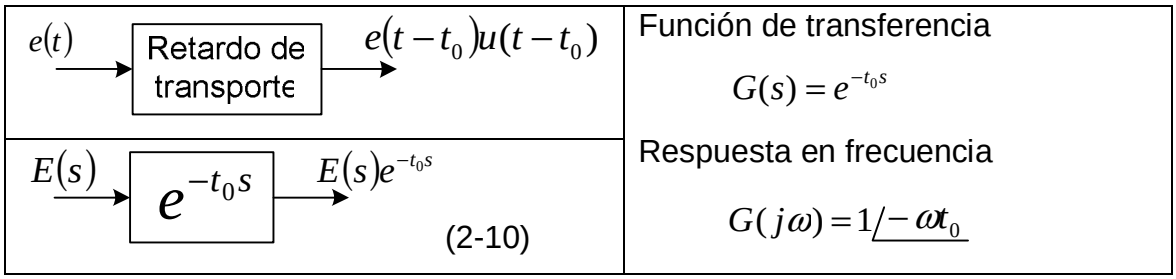


Figura. 2.9. Representación del retardo de transporte

Algunos de los problemas que puede ocasionar la existencia de retardo de transporte en un sistema son:

- Limita el máximo el ancho de banda alcanzable por un sistema en lazo cerrado.
- Para valores altos de ganancia el retardo de transporte puede producir inestabilidad incluso en sistemas de primer orden.
- Un controlador puede saturar los actuadores a falta de una respuesta inmediata ante la acción.

2.4.2 Modelado del retardo de transporte

Existen dos maneras en las que habría como implementar el retardo en un sistema digital. El primero es realizar una aproximación de Padé [*21*] del retardo en cuestión y luego multiplicar por la función de transferencia. La desventaja de este método es el hecho de que se aumenta el orden de la función e incluso puede llegar a crear inestabilidades.

El método más práctico es el usar un sistema de grabación [*22*], en el cual se registra el valor y se lo reproduce después de un tiempo (tiempo de retardo). La manera más eficiente de implementarlo en software es mediante una pila circular radicada en la memoria RAM. La limitación evidente de cuanto retardo se puede simular es la cantidad de memoria RAM disponible y la frecuencia a la que se trabaja. *Este será el método que se utilice en el emulador SISO*. En el Capítulo 4 (*4.3.5.2 Función de retardo de transporte*) se dan los detalles de implementación del retardo de transporte.

2.5 OTRAS NO LINEALIDADES

En este trabajo solo se presentarán e implementarán las no linealidades (estáticas) que son invariantes en el tiempo y no contienen efectos de histéresis. En todos los casos estas no linealidades pueden ser implementadas por software con la ayuda de una tabla de búsqueda, en la cual se genera la salida no lineal al relacionar la entrada con una curva que representa el fenómeno. En la *Tabla 2.3* se indica ciertos tipos comunes de estas no linealidades y los parámetros que los definen.

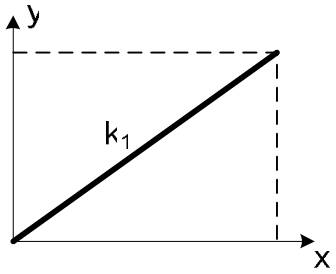
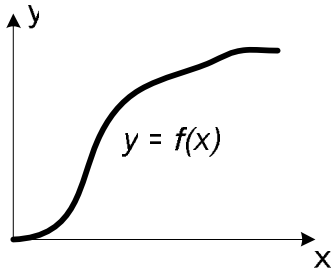
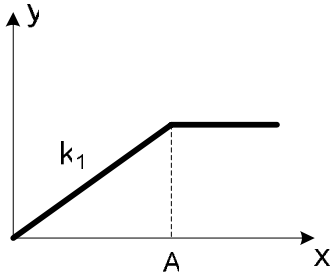
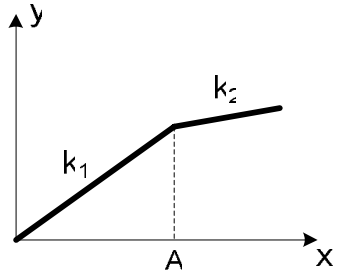
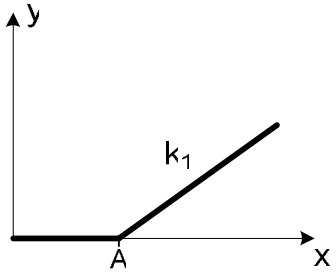
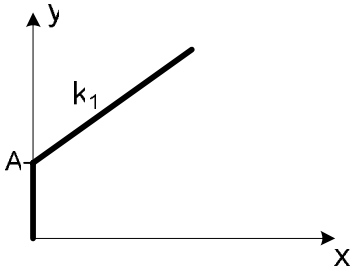
Función lineal sin discontinuidades	Función arbitraria
	
Saturación	**Cambio de pendiente**
	
Zona muerta	**Fricción de Coulomb**
	

Tabla. 2.3. No linealidades a ser implementadas en el *emulador SISO*

2.6 PERTURBACIONES EXTERNAS

Para este proyecto se trabajará únicamente con dos perturbaciones:

- perturbación aditiva al flujo principal de la señal y
- perturbación de ganancia a la función de transferencia

Solo una de las perturbaciones puede estar activa a la vez ya que se cuenta únicamente con un solo control externo que la modifique. Cada una de estas perturbaciones contará con los límites de variación necesarios. En la *Figura 2.10* se muestra las dos configuraciones que pueden tener las perturbaciones.

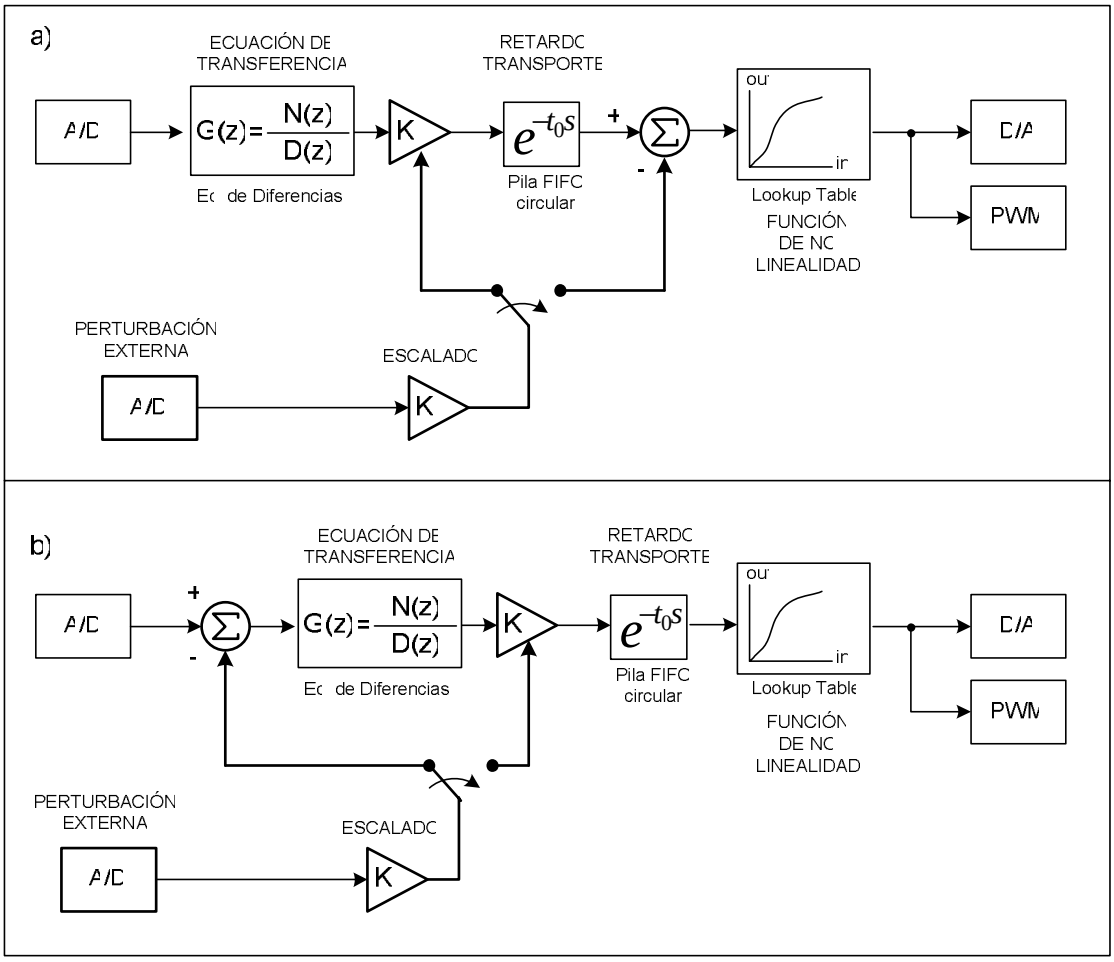


Figura. 2.10. Lugares donde se ubican las señales de perturbación

CAPÍTULO 3

DISEÑO DEL HARDWARE DEL SISTEMA

3.1 DESARROLLO DE ESPECIFICACIONES

En el siguiente capítulo se expone las especificaciones definitivas y detalladas que tendrá el hardware del *emulador SISO* y el diseño que le corresponde. Como se mencionó en el Capítulo 1, un simulador HIL debe cumplir con ciertos requerimientos de hardware para trabajar como tal. Desde el punto de vista de características funcionales, el emulador tendrá un hardware muy similar a un controlador embebido con adquisición y procesamiento de datos digitales ya que debe cumplir con lo requerimientos generales (Re1-Re6) y aquellos relacionados a la tecnología disponible en nuestro país y que definitivamente permiten la realización del proyecto (Re7-Re10).

REQUERIMIENTOS GENERALES	
Re1	Ser autosuficiente al cumplir la tarea que se le encomienda, esto es, todos los cálculos deben realizarse “a bordo”, sin ayuda de una PC u otro equipo. Se considera que algunos de los cálculos involucran operaciones en punto flotante.
Re2	La ejecución de la aplicación debe realizarse en tiempo real, lo que significa que para el resto de los equipos conectados una acción en la entrada del equipo emulador produce una salida “aparentemente” instantánea.
Re3	Debe incluirse al menos un canal de comunicación por medio del cual se pueda configurar el equipo o realizar otras funcionalidades.

Re4	Es necesario contar con al menos una entrada y una salida analógica (para garantizar un sistema SISO analógico) que puedan manejar rangos de señal típicos en la industria (en amplitud y frecuencia).
Re5	Debe tener entradas y salidas digitales capaces de soportar voltajes entre 5 y 24V.
Re6	El circuito debe acondicionar el voltaje de alimentación para que todos los componentes se energicen correctamente.
REQUERIMIENTOS TECNOLÓGICOS	
Re7	Se procurará utilizar la tecnología y componentes más modernos en la medida de las posibilidades.
Re8	La tecnología de placa debe ser máximo de doble lado y que todos los componentes sean "*through hole*" para permitir un ensamblaje fácil y la reparación de la placa a lo largo de su vida útil.
Re9	En lo posible se intentará realizar el proyecto con elementos disponibles en el mercado ecuatoriano, caso contrario se importarán las partes.
Re10	Los valores de los componentes pasivos deben ser nominales.

Tabla. 3.1. Requerimientos del Hardware

De los seis primeros puntos expuestos que definen el hardware se obtienen los 5 bloques básicos que componen al *emulador SISO*:

Requerimientos	*Bloque funcional encargado*		*Referencia Planos*
1 y 2	Procesador/Controlador e Interfase	B1	A2.2
3	Comunicaciones	B2	A2.2
4	I/O Analógicas	B3	A2.4 y A2.5
5	I/O Digitales	B4	A2.3
6	Fuente de poder	B5	A2.6

Tabla. 3.2. Bloques funcionales según requerimientos

La *Figura 3.1* es el diagrama general del hardware del emulador, la explicación y justificación de cada bloque se detalla conforme se avanza en el capítulo. En la *Tabla 3.2* se señala el lugar en los planos o esquemáticos (*Anexo A2*) del *emulador SISO* donde se encuentra cada bloque funcional.

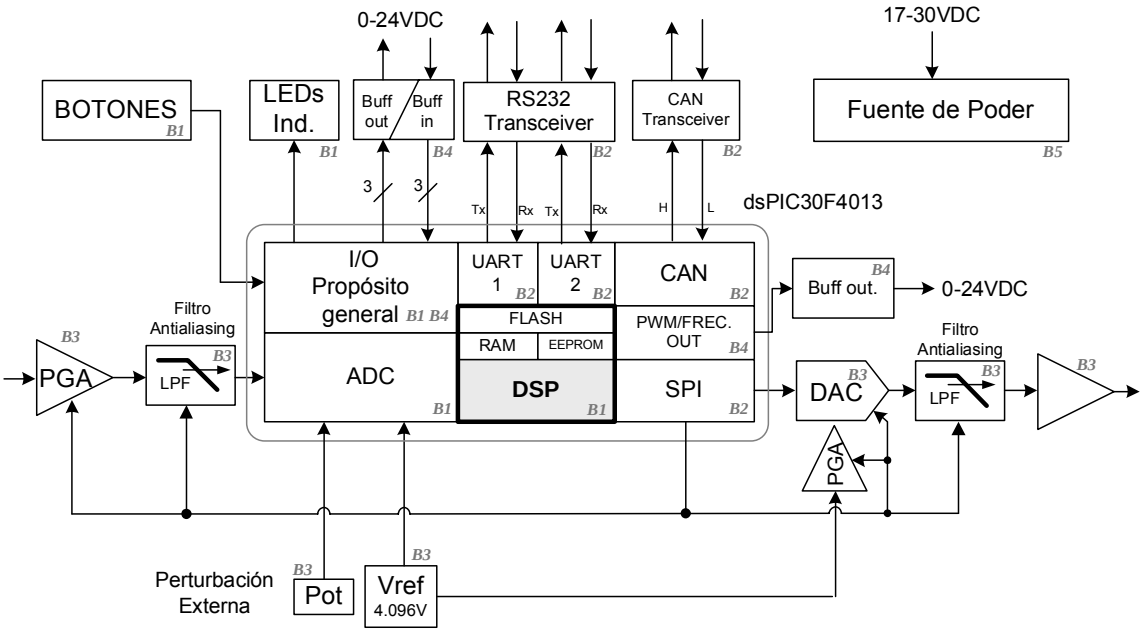


Figura. 3.1. Diagrama general del Hardware del *emulador SISO*

3.2 CONTROLADOR E INTERFASE

3.2.1 Selección de componentes

La selección del procesador/controlador es clave para un buen desarrollo del proyecto y, como se indicó en la *Tabla 3.1*, Re1 y Re2, se deben cumplir ciertas exigencias del mismo. Desde un principio se estableció que el emulador debería tener la mayor cantidad de funciones relacionadas al *emulador SISO* incorporadas en un solo chip, esto es, que al menos se encuentre incorporado el procesador, la memoria RAM y Flash, comunicaciones (Re3) y salidas/entradas digitales (Re5).

Se buscó entre distintos fabricantes de procesadores para aplicaciones embebidas que demandan cierto nivel de complejidad matemática, como Texas Instruments, Freescale y Microchip. Todos estos fabricantes tienen una línea de productos denominados controladores digitales de señales (DSC), los cuales incorporan la robustez e integración de un microcontrolador con la capacidad computacional de un DSP. Para la selección se tomaron en cuenta no solo aspectos de potencia computacional sino de facilidad en conseguir los elementos, forma de programarlos, disponibilidad de herramientas de desarrollo y empaques DIP.

La selección final fue por el DSC de Microchip de la familia dsPIC30F y, más concretamente, por el dispositivo dsPIC30F4013.

3.2.2 Descripción del controlador digital de señales (DSC)

3.2.2.1 Aspectos generales del DSC

El dsPIC30F es un procesador RISC de 16bits de alto desempeño que ofrece la funcionalidad de un DSP con las capacidades fundamentales de control de un microcontrolador. Posee una arquitectura Harvard modificada lo que permite tener un rendimiento de hasta 30 millones de operaciones por segundo (MIPS). Integra un buen grupo de periféricos y elementos que permiten una fácil implementación de aplicaciones como control de motores, reconocimiento de voz, medidores de energía, controladores avanzados entre otros.

En el anexo A3.1 [*23*] se incluye el resumen del fabricante en cuanto a las características del DSC y en el Manual de Referencia [*24*] una descripción completa de funcionalidades. A continuación se da un resumen de las características del chip dsPIC30F4013 que cumplen con los requerimientos planteados en el punto 3.1.

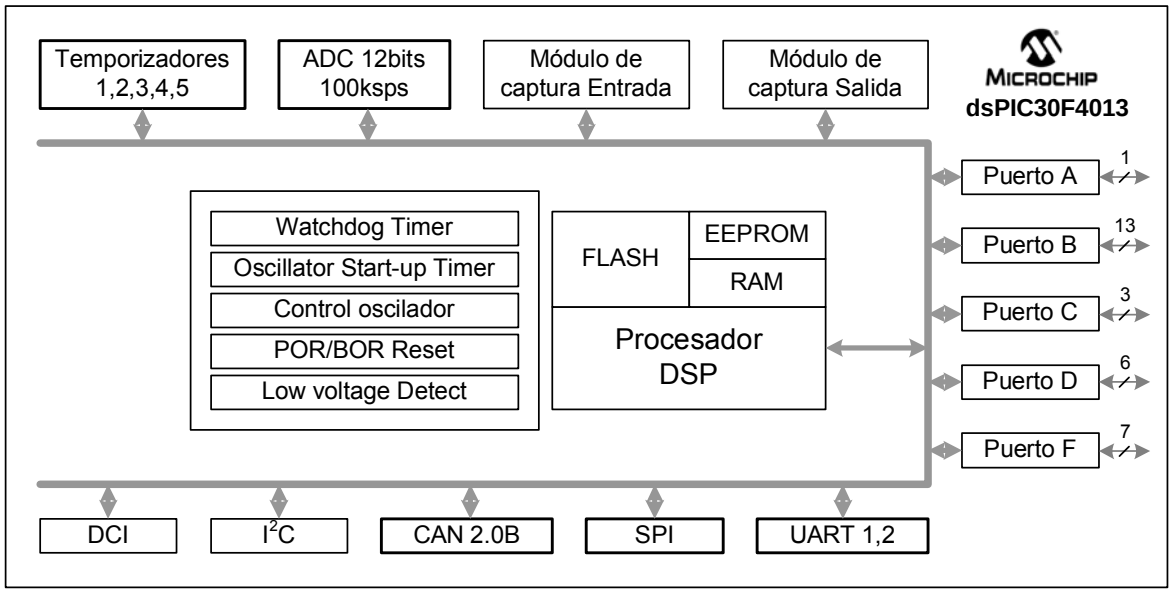


Figura. 3.2. Diagrama general del DSC dsPIC30F4013

3.2.2.2 Cumplimiento de requerimientos

Re1 - Aunque la arquitectura de 16 bits y con funcionalidad de DSP no permite el manejo “en hardware” de punto flotante, si permite el uso de este por software con un rendimiento muy superior al que tendría un microcontrolador de 8 bits. El 30F4013 posee 48kbytes de memoria flash incorporada que puede ser programada en tiempo de ejecución normal permitiendo la programación del dispositivo con la ayuda de un programa de carga al arranque (*bootloader*) a través de un puerto de comunicación. Los 2Kbytes de memoria RAM son suficientes para la aplicación del *emulador SISO*. El banco de memoria EEPROM de 1024 kbytes es útil para el almacenamiento de información de calibración y de los parámetros de la plantilla que definen el comportamiento del dispositivo. El DSC cuenta adicionalmente con un PLL configurable (4x, 8x, 16x) para trabajar con un cristal externo o con el oscilador RC interno de alto rendimiento y cuya frecuencia se puede calibrar por software.

Re2 - La capacidad de trabajar a 30 MIPS, tener una memoria RAM que trabaja a la velocidad del procesador y el contar con una arquitectura de DSP garantiza en gran medida una ejecución rápida del código. El manejo eficiente de interrupciones (4 ciclos) ayuda aún más en aprovechar al máximo el tiempo útil del procesador. Es posible lograr que un programa de mediana complejidad trabaje en tiempo real en este DSC.

Re3 - El 30F4013 cuenta con dos canales de comunicación UART independientes y adicionalmente tiene un módulo CAN2.0B. Para comunicación entre componentes cuenta con un módulo SPI uno I2C y uno DCI (soporta I^2S y AC'97). Como se puede apreciar se tiene suficiente número de canales de comunicación. Para el *emulador SISO* se utilizará únicamente un módulo UART para la interfase con el configurador, el módulo SPI para comunicación con otros elementos de la placa y se deja el módulo CAN listo para realizar cualquier otra aplicación en el futuro.

Re4 - El 30F4013 cuenta con un flexible módulo de conversión A/D que multiplexa hasta 13 canales a un conversor de 12 bits con una capacidad de 100.000 conversiones por segundo. Cuenta con módulos de comparación para la generación de señales PWM con precisión de hasta 16 bits. No cuenta con un módulo de conversión D/A (DAC) por lo que es necesario usar uno externo.

Re5 - Las funciones de los distintos módulos o periféricos del DSC están multiplexadas hacia los pines físicos del chip. Son pocos los pines que no se pueden configurar como entradas o salidas digitales. Todos estos pines trabajan máximo con 5V y pueden entregar o drenar hasta 25mA.

Re7 y Re8 - El componente dsPIC30F4013 salió en el 2004 y para el 2006 se considera todavía un producto moderno, es más, el momento de usar este componente es preciso debido a que ha tenido suficiente tiempo para madurar y se cuenta con una mejor documentación, más y depuradas herramientas de desarrollo y se tiene más notas de aplicación. El fabricante Microchip se encuentra

impulsando la arquitectura de esta familia con nuevos productos (familia dsPIC33F) pero que son totalmente compatibles entre si. En cuanto al empaque sin duda este jugó un papel importante en la selección de este dispositivo.

El autor está seguro que el 30F4013 es el microcontrolador (DSC) con mayor potencia de cómputo, memoria y periféricos que todavía se encuentra en empaques PDIP de 40 pines. La ventaja radica en que es posible armar los prototipos en *protoboard* y realizar un circuito impreso con tecnología "*through hole*". Adicionalmente puede trabajar con voltajes desde 3 a 5 voltios, algo que no es común en los nuevos procesadores que trabajan máximo con un voltaje de 3.3V.

3.2.3 Conexiones básicas[2]

Debido a la alta integración de funcionalidades que proporciona el DSC son pocas las condiciones que se necesita para ponerlo a trabajar y estas son:

- Una correcta polarización y fuente de alimentación,
- Un circuito adecuado de reset y
- Un cristal, una red RC ó una señal TTL para el reloj interno (opcional).

3.2.3.1 Polarización

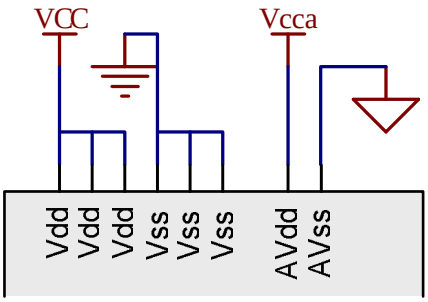


Figura. 3.3. (A2.2-B3)

El DSC provee 3 pines para la alimentación digital (Vcc = 5V), 3 pines para la conexión a la tierra digital y un pin de polarización para el ADC (Vcca) junto con un pin para la conexión a la tierra analógica. Los capacitores de desacoplo se incluyen en la fuente de alimentación.

[2] Al lado de cada diagrama se explica su respectiva ubicación en el anexo A2. Se entiende A2.x-(V,W)(r,s) que se encuentra en la página x de los esquemáticos, secciones verticales V, W y horizontales r,s.

3.2.3.2 Reset

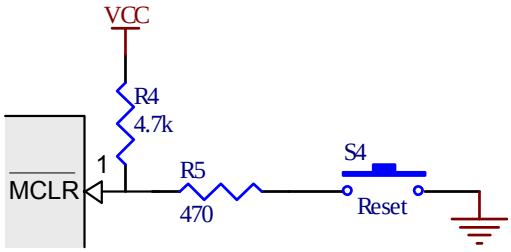


Figura. 3.4. (A2.2-A3)

El circuito de Reset incluye el divisor de tensión conformado por R4 y R5, previendo que R5 se añade para evitar un enclavamiento accidental del reset. Al presionar S4 el dispositivo se resetea.

3.2.3.3 Oscilador

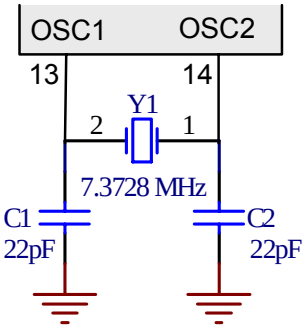


Figura. 3.5. (A2.2-D3)

Se decide trabajar con un cristal externo Y1 de 7.3728MHz (valor comercial). Este es el valor más cercano a 7.5MHz que es el máximo que permite el dispositivo si se habilita el PLL de 16x. Los capacitores C1 y C2 son necesarios para un correcto funcionamiento del cristal (Valores recomendados por el fabricante Microchip [*25*]). Con el cristal (HC49) se logra tener una precisión de ±50ppm. Adicionalmente se incluye R34 con el fin de tener en el pin RB5 una señal proporcional al reloj de manera que se pueda medir la frecuencia real y realizar la calibración necesaria.

Figura. 3.6. (A2.2-B2)

3.2.4 Interfase con el usuario

Para el caso del *emulador SISO* se proporciona una interfase muy sencilla y básica con el usuario: tres LEDs indicadores, tres botones y un potenciómetro, este último descrito en la sección de 3.4.10. La razón de esto es la enorme simplificación del circuito y del programa al obviar el uso de un LCD y teclado, aparte de que el uso de estos requiere muchas líneas de señal, insuficientes en el 30F4013.

3.2.4.1 Botones

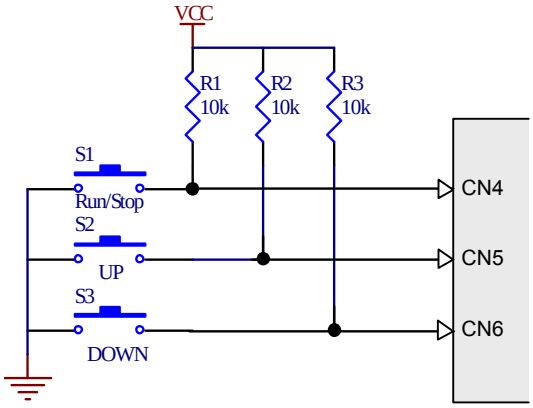


Figura. 3.7. (A2.2-A2)

Los tres botones marcados S1, S2 y S3 y nombrados respectivamente RUN/STOP, UP y DOWN. S1 permite empezar y parar una simulación HIL. Los otros dos se dejan de reserva para futuras ampliaciones en la aplicación. Las resistencia R1-R3 funcionan como "pull-up"s de los botones, aunque el DSC proporciona en los pines marcados CNx resistencias weak pull ups.

3.2.4.2 LEDs indicadores

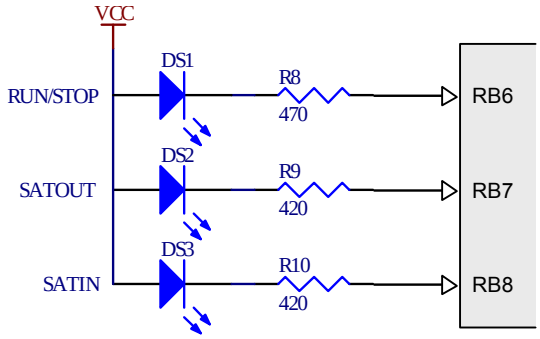


Figura. 3.8. (A2.2-D2)

Se tiene 3 LEDs indicadores configurados para que funcionen el momento de poner los pines RBx en bajo (trabajen como drenaje). RUN/STOP indica el estado del *emulador SISO* (encendido, parado, simulando). SATOUT indica cuando la señal de salida satura el DAC y SATIN indica cuando el ADC se satura.

3.3 COMUNICACIONES

3.3.1 Tipos de comunicaciones en el *emulador SISO*

El *emulador SISO* utiliza cuatro de los periféricos dedicados a comunicación que tiene el DSC y cumplen las siguientes funciones (*Tabla 3.3*):

Módulo de comunicación	Utilización en el *emulador SISO*
UART1 *Universal Asynchronous Receiver Transmitter*	Configurado para trabajar en los pines alternos es el canal de comunicación por el cual se descarga el programa con ayuda del bootloader y se realiza la configuración del *emulador SISO* y además es el que, con un cable cruzado convencional, se comunica con cualquier equipo CTU (*Control Terminal Unit*) con interfase RS232, por ejemplo una PC.
UART2	Se da la posibilidad de que con un cable especial se tenga un segundo canal asíncrono RS232. Con un cable cruzado se tienen las señales de control CTS y RTS.
SPI *Serial Peripheral Interface*	Este canal está dedicado para la comunicación con los potenciómetros digitales y el DAC. Se lo utiliza con tres señales: Entrada de datos, Salida de datos y el reloj. Logra altas velocidades de transmisión de datos y no necesita ningún driver para las señales. Se trabaja con 5 V. En el *emulador SISO* trabajará con un reloj a 5MHz.
CAN2.0B *Controller Area Network*	Aunque no se utilizará en el *emulador SISO*, es conveniente dejar la extensión y el Hardware para futuras aplicaciones. El CAN es un protocolo que cada vez tiene mayor aceptación en vehículos y maquinarias en general.

Tabla. 3.3. Designación de los canales de comunicación en el *emulado SISO.*

3.3.2 Interfase para UART1 y UART2

Para poder lograr los niveles de voltaje requeridos por la interfase RS232 es necesario incorporar al *emulador SISO* un transceiver para acoplar correctamente

las señales. Otra función importante que debe cumplir este elemento es la protección del circuito de sobrevoltajes y cortocircuitos.

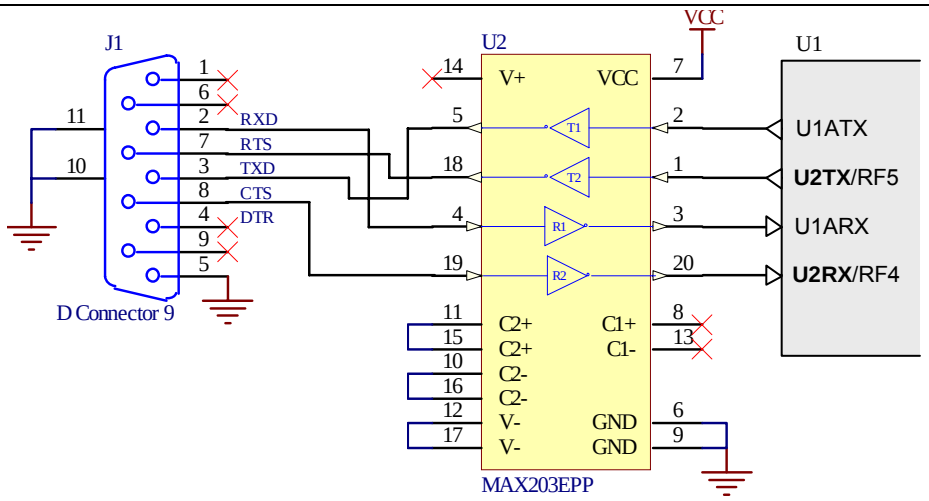
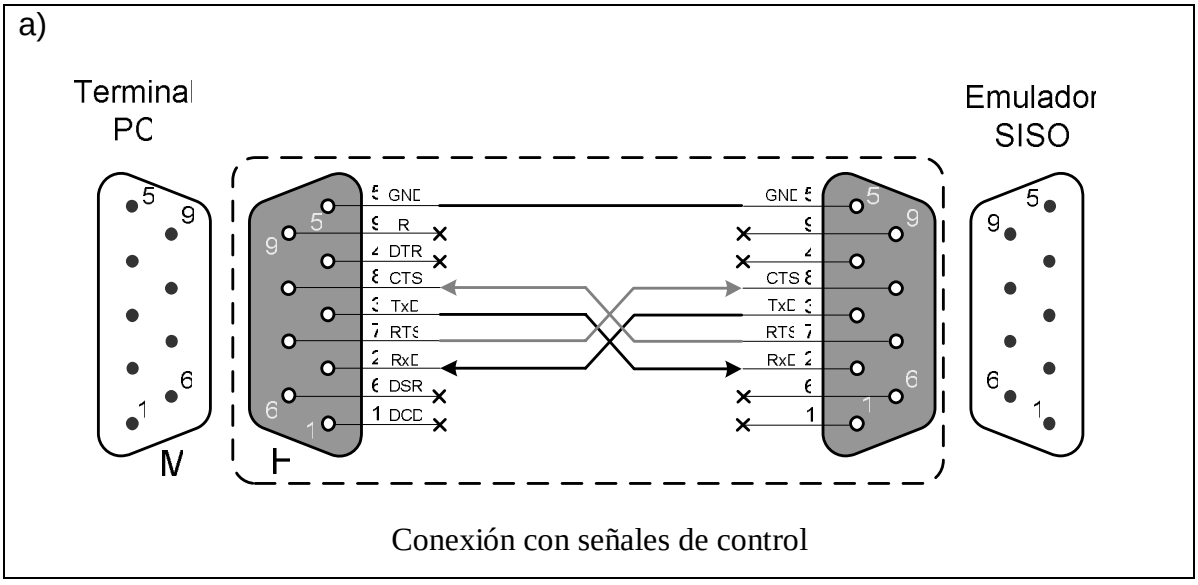


Figura. 3.9. (A2.2-C(1,2))

Para esta aplicación se seleccionó el transceiver MAX203ECPP (anexo A3.2), funcionalmente muy simular al conocido MAX232. Contiene 2 drivers de salida y 2 drivers de entrada que acoplan señales TTL (0-5V) con los voltajes de la interfase RS232 (±12V). Para generar estos voltajes altos, el MAX203, duplica el voltaje de alimentación (+2Vcc) y luego lo invierte (-2Vcc).



Conexión con señales de control

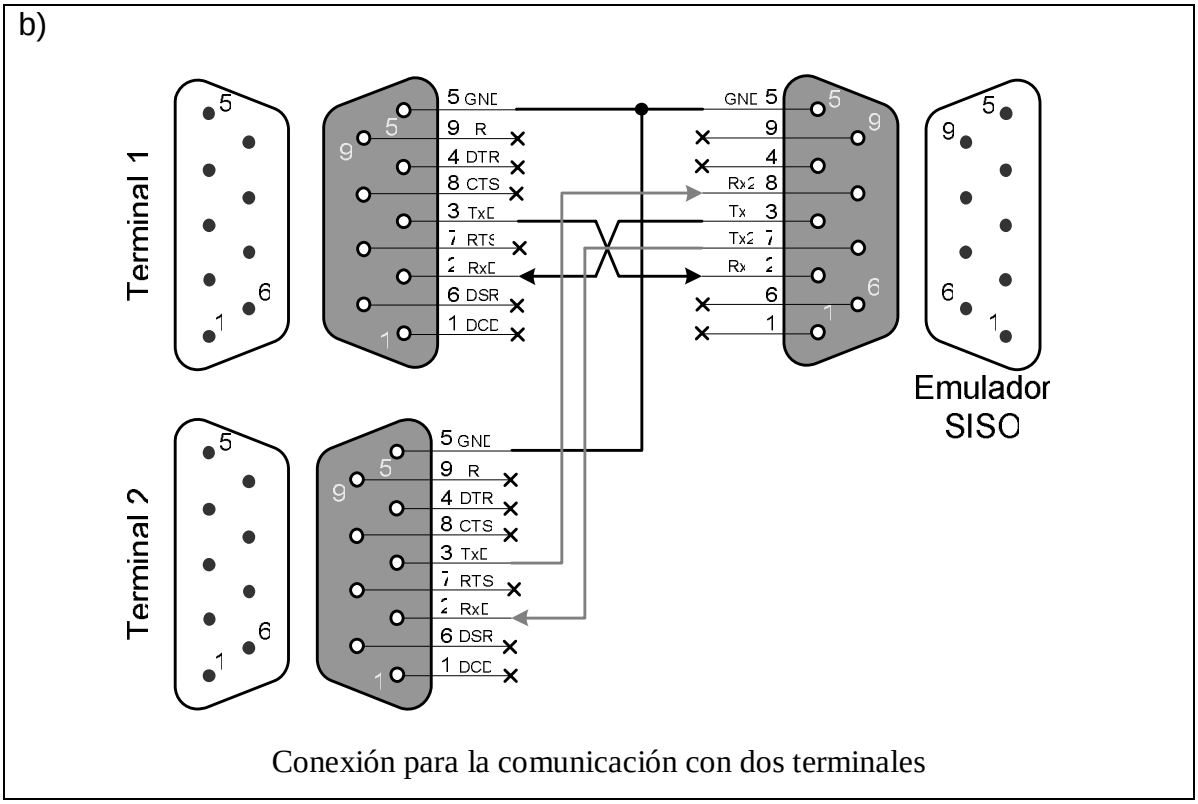


Figura. 3.10. Conexiones posibles por medio del puerto serial.

A diferencia del MAX232, el MAX203 no necesita los cuatro capacitores externos para la bomba de carga (*charge pump*) ya que vienen incorporados en el mismo empaque. Adicionalmente provee una protección ESD de ±15kV. El conector utilizado es un DB9 macho. Las posibles conexiones se muestran en la *Figura 3.10*.

La compatibilidad necesaria para un adaptador USB-Serial se garantiza con los niveles de voltaje presentado por el MAX203. Este garantiza una interfase EIA/TIA-232E y en los adaptadores USB-Serial por lo general cumplen con esta especificación.

3.3.3 Interfase para CAN2.0B

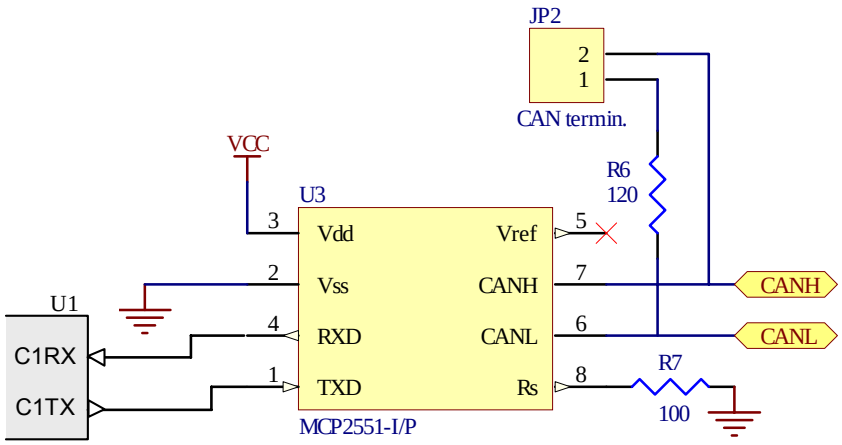


Figura. 3.11. (A2.2-(A,B)4)

La interfase CAN trabaja con un transceiver MCP2551-I/P de Microchip (anexo A3.3) y que esta configurado para trabajar a 5V. Adicionalmente se deja la resistencia R7 para controlar la pendiente de las señales y tener un mejor control de las emisiones electromagnéticas (EMI). R6 se puede conectar a través de JP2 a los dos conductores de la red CAN si se desea que esta terminal sea el último nodo y contenga la resistencia de terminación. El conector utilizado es uno tipo bornera.

3.4 ENTRADAS Y SALIDAS ANALÓGICAS

3.4.1 Aspectos generales del circuito analógico

El diseño del circuito analógico juega el papel más importante en la calidad del sistema de adquisición de datos. Para el *emulador SISO* solo se tendrá una entrada y una salida analógica, con rangos de señal unipolar de 0 a 12 V. Si bien se encuentra en la industria sistemas bipolares, se sacrifica esta característica en el *emulador SISO* en aras de lograr un circuito más sencillo, especialmente en la fuente de alimentación (sección 3.6) y en el circuito de adaptación de señal para el ADC y el DAC. Se supone que los voltajes de alimentación son: Vcca = 5V y Vah = 15V.

Para el circuito analógico se buscará cumplir con los siguientes requerimientos:

ReA1	El sistema de adquisición de datos es de 12 bits, por lo que el rango dinámico es de 72dB. Cualquier señal por debajo de este rango se considera despreciable. El voltaje de referencia para la conversión A/D y D/A es de 4.096V, por lo que se buscará que el contenido de ruido interno sea menor a 1mVpp. (Ver *Tabla. 2.2.*)
ReA2	El ancho de banda del sistema es de DC a 400 Hz, privilegiando en todo momento un mejor desempeño en el dominio del tiempo antes que en el dominio de la frecuencia. Para logar un buen desempeño en DC, el circuito debe introducir lo menos posible o compensar los *offsets* de voltaje que se presenten.
ReA3	El sistema analógico debe trabajar satisfactoriamente con voltajes unipolares de alimentación. Esto implica que los componentes y el circuito deben manejar rangos de señal cercanos al potencial nulo del sistema (tierra).
ReA4	Se buscará que el diseño tenga la mayor estabilidad respecto a variaciones de temperatura y envejecimiento. Se considerará un rango de funcionamiento entre -20 y 60 ºC.
ReA5	Siempre se tomará en cuenta el peor caso para encontrar los valores que cumplan los requisitos.
ReA6	Las etapas de entrada y salida deben estar protegidas para cortocircuitos y sobrevoltajes.

Tabla. 3.4. Requerimientos del circuito analógico

El circuito analógico está dividido en 8 componentes principales con funciones específicas cada uno y los esquemáticos completos se encuentran en las láminas A2.2, A2.4 y A2.5. De manera general las funciones que deben cumplir cada uno se muestran en la *Figura 3.12* y *Tabla 3.5*:

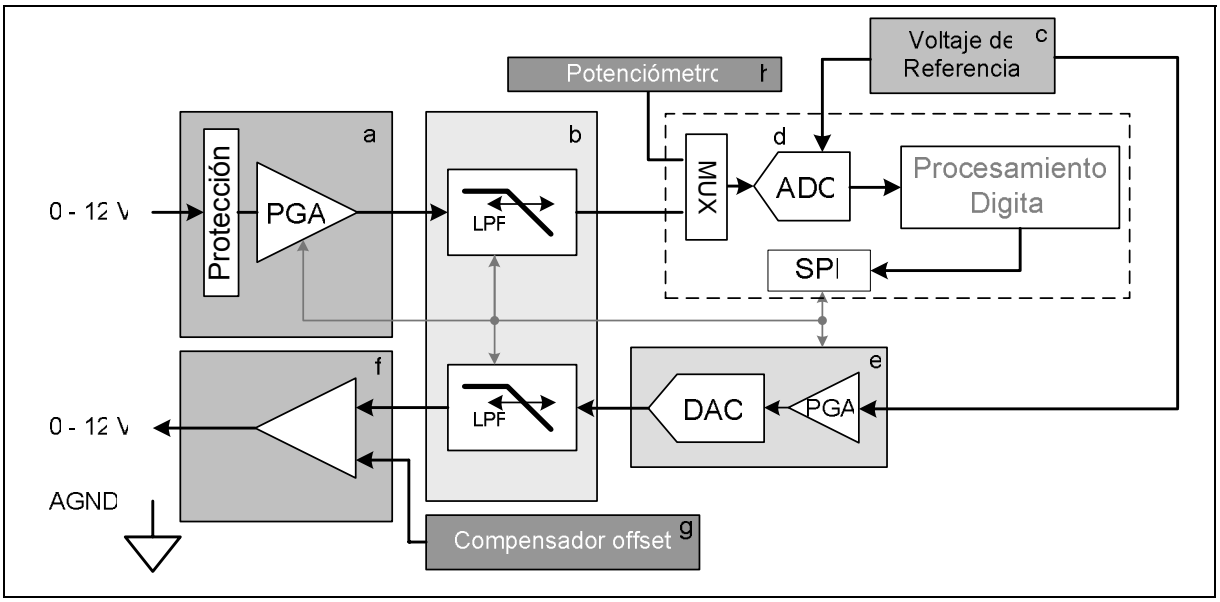


Figura. 3.12. Diagrama funcional del circuito analógico

a	*Buffer de entrada analógico* Tiene las protecciones contra sobrevoltajes y voltajes negativos. Adicionalmente tiene un Amplificador de ganancia programable (PGA) para adaptar los rangos de voltaje de la entrada (0-12) a los 4.096 que maneja el ADC.
b	*Filtros antialiasing de entrada y salida* Estos filtros suprimen las frecuencias altas que pueden producir aliasing tanto a la entrada como a la salida. Cada uno es tipo Bessel de segundo orden implementado con amplificadores operacionales y potenciómetros digitales programados via SPI que varían la frecuencia de corte.
c	*Referencia de voltaje* Genera un voltaje de 4.096V estable en temperatura y tiempo tanto para el ADC como para el DAC.
d	*Conversor Analógico – Digital* Incorporado en el DSC, muestrea alternadamente el potenciómetro y la señal de entrada ya acondicionada.
e	*Conversor Digital – Analógico* Compuesto por dos DACs que se programan mediante el canal SPI; el primero sirve para calibrar el rango de voltajes de salida del segundo que

	es el encargado de convertir los valores calculados.
f	*Buffer de salida analógico* Amplifica el voltaje de la etapa anterior y proporciona una salida de baja impedancia.
g	*Compensación de offset a la salida* Permite compensar el valor de offset a la salida.
h	*Potenciómetro* Es usado como un elemento de interfase con el usuario. Permite variar un parámetro de software, dependiendo del valor analógico que genere.

Tabla. 3.5. Explicación de bloques en el circuito analógico

3.4.2 Amplificadores Operacionales

Como se indicó en ReA2 y ReA3, las condiciones de funcionamiento demandan que los amplificadores operacionales (OPAMP) tengan ciertas características que permitan un buen desempeño en el *emulador SISO*. La primera condición es que tengan rango de entrada y salida riel a riel. Esto implica que los rangos de la señal de entrada y salida deben trabajar sin saturarse o distorsionarse al tener valores cercanos al voltaje de polarización del OPAMP. Por lo general este es un distintivo de operacionales dedicados para aplicaciones que trabajan con baterías y por ende sistemas unipolares, el parámetro en las hojas técnicas se lo obtiene del valor *Output voltage swing* (V_{OH} y V_{OL}).

Adicionalmente se busca que el OPAMP seleccionado tenga un bajo offset de entrada esto para lograr una buena precisión en DC. Este valor, conocido como *Voltage input offset* (V_{IO}) se refiere al voltaje que presenta a la salida el OPAMP cuando la entrada inversora (o no inversora) se conecta a tierra. Los requerimientos de desempeño en frecuencia no son particularmente especiales ya que la máxima frecuencia que se manejará estará en el orden de los 400Hz. Por razones de confiabilidad en la reproducción se buscará que la frecuencia en lazo cerrado sea mucho mayor, cosa que no es difícil con los nuevos operacionales.

Otro aspecto importante es el voltaje de operación. En la actualidad existe una gran gama de OPAMPS dedicados a aplicaciones portátiles que restringen su polarización de 3.3 a 5V. Como el *emulador SISO* necesita trabajar hasta con voltajes de 15 V de polarización, esto restringe mucho la búsqueda. Sin embargo un cuidadoso análisis del circuito lleva a la conclusión de que solo el operacional dedicado al buffer de salida debe ser capaz de trabajar con 15V. Para el resto de operacionales es posible una polarización de 5V. Por esta razón el circuito incluye jumpers (JP4) para seleccionar el voltaje de polarización de los distintos operacionales. En la *Tabla 3.6* aparecen algunos OPAMPS que pueden ser usados para el *emulador SISO*. El LM358 se usará en la aplicación a menos que se indique lo contrario en el diagrama definitivo (Anexo A3.8):

	V_S V	V_{OH} V	V_{OL} mV	V_{IO} mV	BW MHz
LM358	3 – 32	28	5 - 20	2	1.1
LMC6462	3 – 16	14.965	25	0.25	0.05
TC913B	6.5 – 16	15.7	3	0.03	1.5
MCP6022	2.5 –5.5	5.480	15	0.5	10

Tabla 3.6. Características de algunos operacionales para el *emulador SISO*

3.4.3 Potenciómetros digitales

Se incluye en el circuito potenciómetros digitales los cuales son elementos mixtos que cambian la resistencia por medio de un canal de comunicación SPI. Esto permite una gran flexibilidad el momento de configurar y, como se verá más adelante, compensa casi completamente el no contar con componentes pasivos de tolerancias bajas. En el *emulador SISO* serán usados como resistencias variables para programar los filtros antialiasing y variar la ganancia del amplificador de entrada. También trabajarán como potenciómetros en la compensación de offset.

Para esta aplicación se utilizarán el MCP42050 y MCP42100 (anexo A3.4), ambos del fabricante Microchip y funcionalmente idénticos, variando solo el valor de la resistencia. Cada uno de estos elementos posee dos potenciómetros, siendo cada uno un arreglo de resistencias que puede ser acomodado digitalmente para variar la resistencia entre cualquiera de los extremos (A o B) y el conector central (W), logrando un comportamiento similar a un potenciómetro mecánico. Para estos dispositivos se cuenta con 256 pasos de programación, entre el valor mínimo y el máximo.

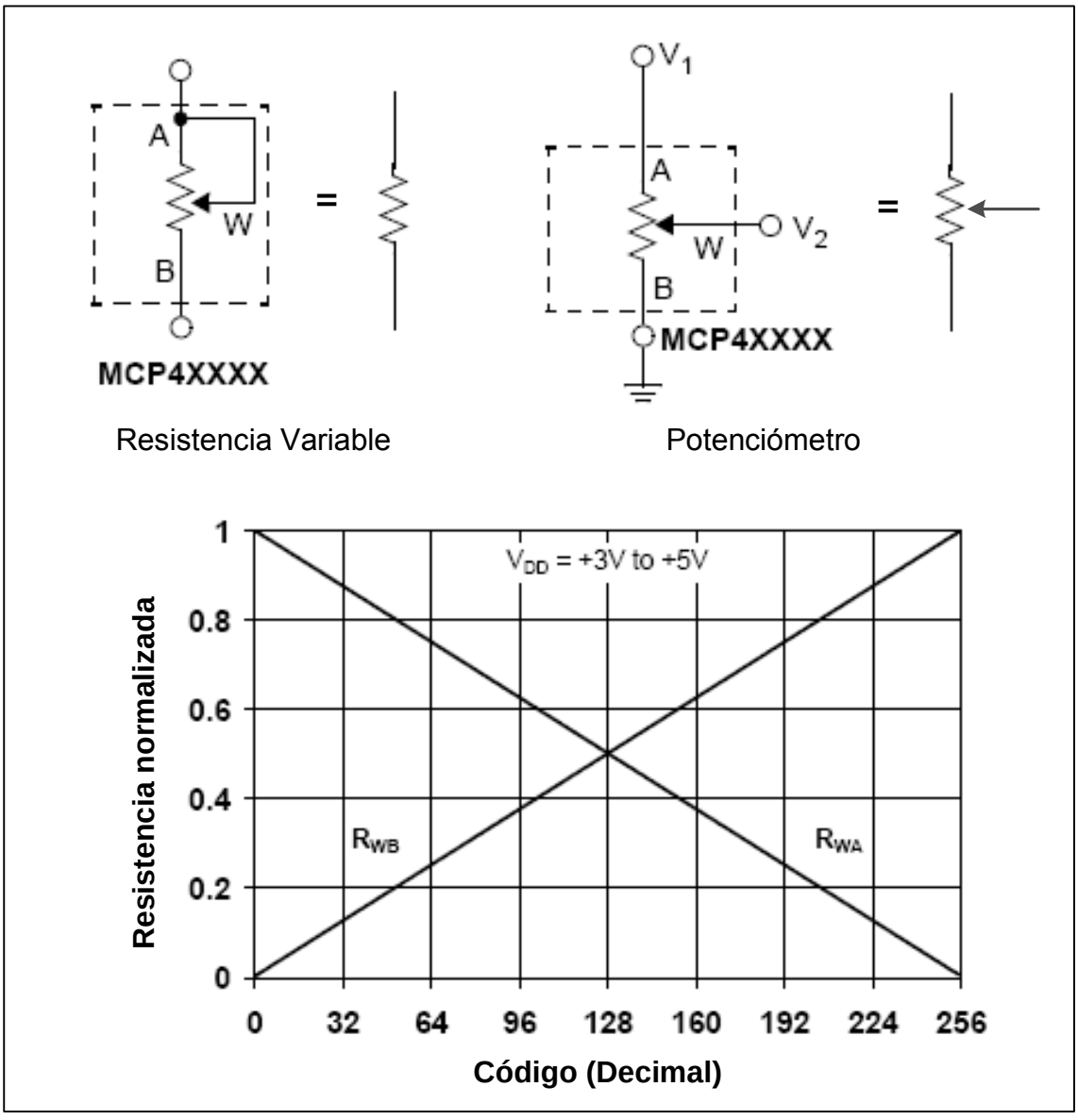


Figura 3.13 Usos y comportamiento de los potenciómetros digitales

Entre las ventajas que presentan los potenciómetros digitales están:

- Se pueden ajustar (programar) infinitamente (Los potenciómetros mecánicos se desgastan conforme se utilizan).
- En comparación con sus contrapartes mecánicas, los potenciómetros digitales no son sensibles a la humedad, polvo o suciedad.
- La linealidad entre pasos es alta y estable en temperatura.
- Aunque la tolerancia es alta entre partes (hasta 30%), la diferencia de resistencia entre dos potenciómetros en un mismo empaque es máximo del 1%.
- Por su naturaleza digital permite calibrar por software ciertas variables analógicas del circuito.

Algunas características de los potenciómetros digitales son distintas a los mecánicos y hay que tener en cuenta en el diseño:

- El voltaje entre cualquiera de las terminales del potenciómetro no puede exceder el voltaje de polarización del elemento.
- No están diseñados para manejar corrientes altas; corrientes superiores a 1 mA pueden dañar el dispositivo.
- La resistencia del elemento de conexión W es elevada y para valores bajos de código esta debe tomarse en cuenta.
- Poseen un ancho de banda limitado (alrededor de 1MHz).
- Es posible que se tenga componentes de la señal digital (*feed through*) en la señal que procesa el potenciómetro.

En la parte de control digital los MCP42xxx necesitan indispensablemente las líneas de salida de datos y reloj del puerto SPI. Pueden ser conectados “en cadena” de tal manera que con una sola señal de selección (CS) se maneje a todo un grupo de dispositivos. Esto resulta especialmente útil si se cuenta con pocas

salidas en el sistema. Los pines de reset ($\overline{RS}$) y apagado ($\overline{SHDN}$) se los conecta directamente a Vcca (quedan inhabilitados) por no ser particularmente útiles para el *emulador SISO* y se liberan 2 pines más para otros propósitos. La conexión se muestra en *Figura 3.14*.

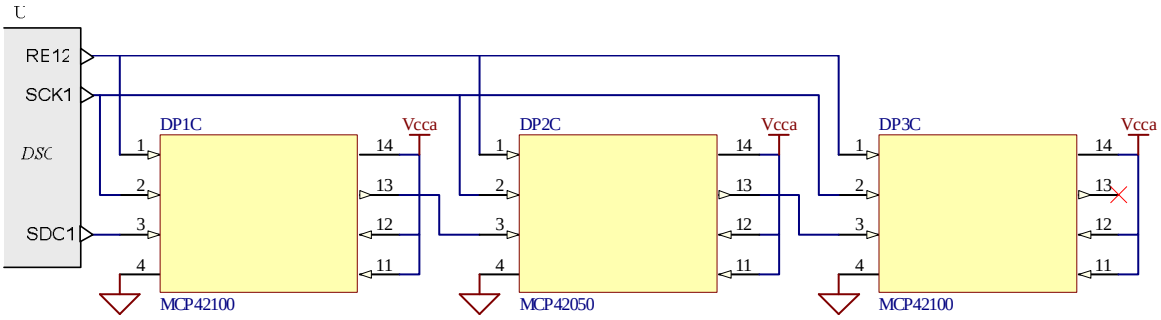


Figura. 3.14. (A2.4-D(1,2,3))

3.4.4 Buffer de entrada analógico

3.4.4.1 Protección de entrada y divisor de tensión

En el *Figura 3.15* los diodos D5 y D7 son diodos Schottky que permiten una rápida respuesta de protección. Vah es nominalmente de 15V, por lo que D5 conducirá cuando la señal de entrada supere los 15.5 V. Para el caso de polarizaciones negativas, D7 conduce cuando el voltaje de entrada es inferior a -0.5V. A continuación de la protección se encuentra el divisor de tensión conformado por R23 y R24. Los valores deben ser en lo posible de 1% de tolerancia con los valores indicados. Este divisor cumple dos funciones: da a la etapa de entrada una impedancia más baja (R23||R24 ≅ 10.8kΩ) y reduce el voltaje de entrada en una relación de 1/3 aproximadamente. El voltaje de trabajo máximo es de 12 V, lo que genera un voltaje en B de 4 V. Con esto se asegura que el potenciómetro DP1A no tenga voltajes mayores a Vcca (5V) en su terminal PB0.

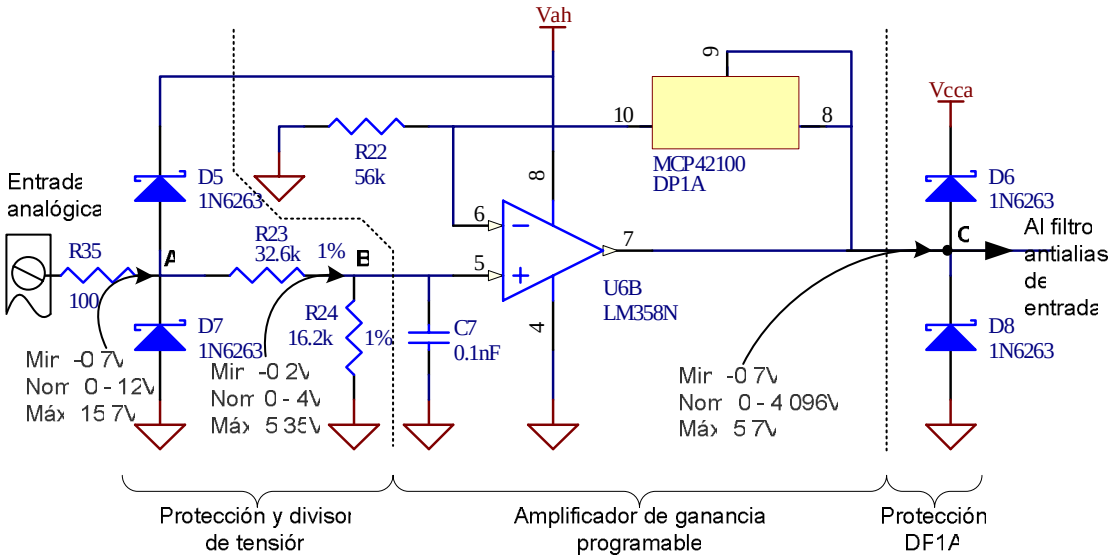
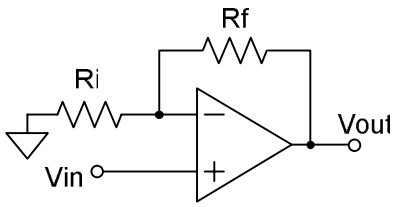


Figura. 3.15. (A2.4-(A,B)(1,2))

3.4.4.2 Amplificador de ganancia programable (PGA)

El amplificador programable se basa en una configuración de amplificador no inversor cuya ecuación de transferencia es (3-1):



$$Vout = Vin \cdot \left(1 + \frac{Rf}{Ri}\right) \quad (3\text{-}1)$$

$$G = 1 + \frac{Rf}{Ri} = \frac{Vout}{Vin} \quad (3\text{-}2)$$

Para encontrar el valor de R22 (Ri) se buscó la relación Rf/Ri que cumpla con los extremos y se le utilice en toda su escala al potenciómetro DP1A. Se conoce que independientemente del rango de entrada seleccionado, el valor en C debe estar comprendido entre 0 a 4.096 V.

Escala	V_A (max)	$V_B = \left(\frac{1}{3}V_A\right)$	V_C	G	$\frac{Rf}{Ri}$	$Ri\ con\ Rf = 100k$	$Ri\ con\ Rf = 1.2k$
0 – 5	5	1.6666	4.096	2.467	1.46747	**68.144 k**	-
0 – 12	12	4.0000	4.096	1.024	0.02400	-	**50.000k**

Tabla. 3.7. Cálculo de valores para R22

En la *Tabla 3.7* se tienen los casos extremos y se ve que para poder utilizar el máximo rango de Rf (DP1A), es necesario proporcionar una resistencia Ri (R22) entre 50kΩ y 68kΩ. Se escoge el valor de 56kΩ con una tolerancia de 5%.

En modo normal de funcionamiento y si todo está conectado correctamente, el voltaje en el punto C no debe ser mayor a 4.096 [V]. El peor caso sería si el usuario conecta 12 [V] a la entrada y el PGA está configurado para la escala de 0-5. En este caso el voltaje en B se multiplicaría por la ganancia (2.467) y se tendría en C 9.87 [V]. Por esta razón se añaden los diodos D6 y D8 que impiden que el voltaje en C suba por encima de 5.4 [V] en la terminal PA0, protegiendo de esta manera al DP1A que debe mantener un voltaje entre sus terminales no mayor a Vcca+0.3.

3.4.5 Filtros antialiasing de entrada y salida

3.4.5.1 Características y soluciones para filtros analógicos

Los filtros antialiasing son indispensables para lograr una buena adquisición de datos. Cumplen la función de suprimir las frecuencias que se encuentran sobre fs/2 de tal manera que estas no pasen a la banda de interés y modifiquen los valores digitalizados.

Para poder decidir el filtro que se debe usar hay que estudiar a un filtro analógico por sus dos facetas: la que se refiere a su comportamiento teórico (*Tabla 3.8*) y la que se refiere a su implementación (*Tabla 3.9*).

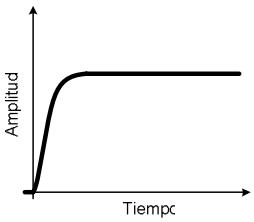
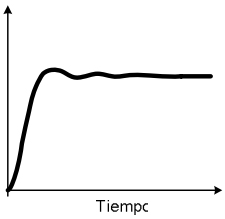
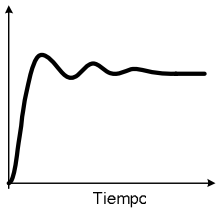
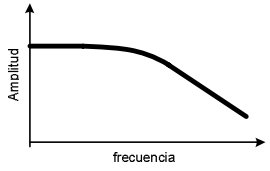
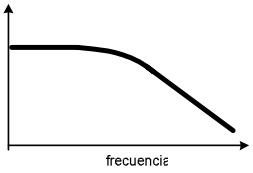
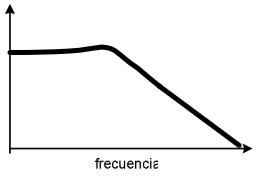
	Tipo de filtro (Todos los filtros mostrados son de segundo orden)		
Características	Bessel	Butterworth	Chebyshev
Función de Transferencia	$H(s) = \frac{\omega_0^{\,2}}{s^2 + \left(\frac{\omega_0}{Q}\right) + \omega_0^{\,2}}; \quad \omega_0 = 8 \cdot fc$ (3 - 3)		
Respuesta al escalón			
Comportamiento fase	La fase varía linealmente con la frecuencia.	Desplazamiento de fase con respecto a la frecuencia	Desplazamiento de fase con respecto a la frecuencia
Comportamiento en frecuencia			
Valores típicos de Q	0.57735	0.70711	1.3046

Tabla 3.8. Tipos de filtro según su función de transferencia

	Forma de implementar el filtro		
	Pasivo	Activo	Capacitores conmutados
Componentes	Realizado únicamente con componentes pasivos: resistencias, capacitores e inductores.	Basado en amplificadores operacionales, resistencias y capacitores.	Filtro implementado en silicio; simula el comportamiento de resistores al conmutar la carga entre capacitores.
Facilidad de Diseño	Sumamente complicado para órdenes altos.	Complicado para órdenes altos.	Muchos no necesitan diseño, depende del modelo utilizado.
Facilidad de Implementación	El buscar componentes de precisión o lograr valores no comerciales de R, C y L, puede llevar mucho tiempo o conducir a la irrealizabilidad	Hay que tener mucho cuidado en la selección de componentes R y C, el diseño de la placa y la selección del OPAMP.	Existen ICs que cumplen la función de un filtro avanzado sin necesidad de elementos externos. Otros ICs se programan con el uso de resistencias externas.
Distorsión	Distorsión no existente	Dependiendo del OPAMP se pude lograr distorsiones sumamente bajas. Este parámetro varía según el rango de variación de la señal.	Resulta difícil conseguir elementos con distorsión menor a -85dB.

Desviación en DC	No introduce offset en la señal	Se introduce cierto grado de offset.	El offset que produce es apreciable.
Ruido	No inserta ruido en la señal.	Cierto nivel de ruido	Introduce un apreciable nivel de ruido debido a su comportamiento conmutado.
Forma de fijar de la frecuencia de corte	Depende de los com - ponentes R, C y L	Depende de los valores de R y C.	Se controla con una frecuencia que es de 20 a 100 veces la frecuencia de corte.
Aplicaciones	-Filtros sencillos de primero o segundo orden. -Aplicaciones que no toleran ruido -Aplicaciones de alta frecuencia.	- Equipos de audio de alta fidelidad. -Filtros de 2do o más orden. -Aplicaciones que requieren buen rango dinámico.	-Sistemas de voz como teléfonos, etc. -Aplicaciones con rango dinámico bien definido.

Tabla 3.9. Tipos de filtros según el método de implementación

3.4.5.2 Selección del tipo de filtro

Según el requerimiento ReA2, se busca un buen comportamiento del filtro en el dominio del tiempo. A partir de las gráficas y de la literatura consultada [*13*], [*26*] sobre filtros y como se puede ver en la *Tabla 3.8*, aquellos de tipo Bessel logran desplazar la fase de manera lineal respecto a la frecuencia. Esto se refleja en el dominio del tiempo en que no distorsionan las señales que no son senoidales. Este fenómeno es más evidente cuando se aplica una entrada escalón a un filtro Bessel ya que la respuesta del mismo no presenta sobreimpulsos ni oscilaciones, cosa que sí ocurre con los filtros Butterworth y Chebyshev. El tener esta característica se penaliza en el dominio de la frecuencia haciendo que los filtros Bessel tengan un pobre desempeño (necesitan más espectro para suprimir las frecuencias) respecto a otras clases de filtros.

Debido a que en aplicaciones de control es indeseable que se presenten oscilaciones y peor si el emulador debe representar fielmente la función de transferencia simulada, se considera como única opción el filtro analógico Bessel.

Una de las características del *emulador SISO* es que debe ser capaz de simular sistemas lentos y sistemas rápidos. En los sistemas lentos, la frecuencia de simulación es baja con variaciones lentas de la señal y apreciables tiempos

muertos. En cambio, los sistemas rápidos generan señales más “detalladas”. Este gran espectro de sistemas complica en cierta medida la utilización de un sistema que trabaje con una sola velocidad de muestreo, especialmente en la simulación de retardos de transporte.

Por esta razón se buscó un tipo de filtro que permita variar la frecuencia de corte de manera sencilla en tiempo de ejecución. Los filtros conmutados son excelentes para aquello debido a que por medio de una frecuencia se controla radiométricamente la frecuencia de corte. Tras estudiar cuidadosamente su comportamiento se llegó a determinar que estos añaden demasiado offset a la señal y que adicionalmente sufren de grandes distorsiones cuando la señal se acerca a tierra o al voltaje de polarización del elemento (no trabajan riel a riel).

Este hecho llevó a consultar la implementación de un filtro activo que, por medio de potenciómetros digitales, pudiera variar su frecuencia de corte. La nota de aplicación [27] sugiere el diseño de un filtro de segundo orden con estas características. Este requiere un amplificador operacional, dos resistencias (digitales) y dos capacitores.

El hecho de usar potenciómetros digitales suprime la necesidad de contar con elementos (resistencias y capacitores) de bajas tolerancias (1%) para fijar de manera precisa la frecuencia de corte.

3.4.5.3 Implementación de los filtros antialiasing

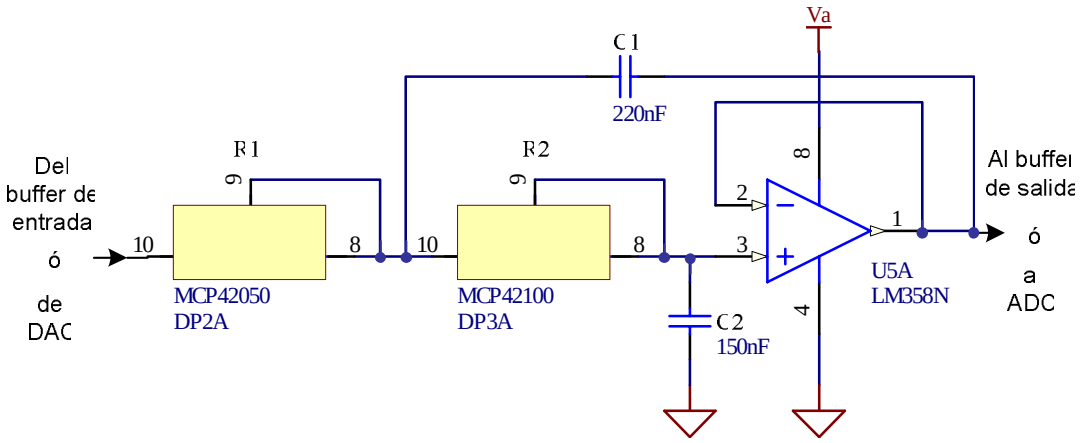


Figura. 3.16. (A2.4-A3)

Para el circuito mostrado en la *Figura 3.16* tenemos que la función de transferencia está dada por (3-4):

$$H(s)=\frac{1/R_1R_2C_1C_2}{s^2+\left(1/R_2C_1+1/R_1C_1\right)\cdot s+1/R_1R_2C_1C_2} \qquad (3\text{-}4)$$

Y según la ecuación (3-3) los parámetros ω_0 y Q están descritos por:

$\omega_0=\frac{1}{\sqrt{R_1R_2C_1C_2}}$ (3-5)	$Q=\frac{\omega_0R_1R_2C_1C_2}{(R_1+R_2)\cdot C_2}=\frac{\sqrt{R_1R_2C_1C_2}}{(R_1+R_2)\cdot C_2}$ (3-6)

Como se utilizan potenciómetros digitales para fijar los valores de R_1 y R_2, los parámetros que necesitamos conocer de antemano son C_1, C_2, $\omega_0=8\cdot fc$ y Q, que para que sea filtro Bessel $Q=1/\sqrt{3}$. Con la ayuda del programa FilterLab de Microchip se determina que, con capacitores $C_1=220nF$ y $C_2=150nF$ los valores de las resistencias R_1 y R_2 quedan definidas por:

$R_1 = \frac{503994}{f_c}[\Omega]$ (3-7)	$R_2 = \frac{938840}{f_c}[\Omega]$ (3-8)

En la *Tabla 3.10* se muestran los valores de R_1 y R_2 calculados para frecuencias entre 10 y 400 Hz que son de interés para el *emulador SISO*.

Frecuencia de corte	***Valor ideal R_1***	***Valor Ideal R_2***	***Valor en el potenciómetro digital R_1***	***Valor en el potenciómetro digital R_2***	***Código en el potenciómetro digital MCP42050(R_1)***	***Código en el potenciómetro digital MCP42100(R_2)***
10	50399.4	93884.0	50195.3	93359.4	257	239
11	45817.6	85349.1	45703.1	85156.3	234	218
15	33599.6	62589.3	33398.4	62500.0	171	160
20	25199.7	46942.0	25195.3	46875.0	129	120
30	16799.8	31294.7	16796.9	31250.0	86	80
40	12599.9	23471.0	12500.0	23437.5	64	60
50	10079.9	18776.8	9960.9	18750.0	51	48
100	5039.9	9388.4	5078.1	9375.0	26	24
200	2520.0	4694.2	2539.1	4687.5	13	12
300	1680.0	3129.5	1757.8	3125.0	9	8
400	1260.0	2347.1	1171.9	2343.8	6	6

Tabla. 3.10. Valores de las resistencias R1 y R2 para capacitores C1 = 220 nF y C2 = 150 nF

Como se puede apreciar en la *Tabla 3.10* R_1 varía de 0 a 50k aproximadamente y R_2 de 0 a 100k. Como es necesario dos de estos filtros (uno para la entrada y otro para la salida), R_1 para los filtros estará compuesta por el MCP42050 y R_2 por el MCP42100.

3.4.6 Referencia de voltaje

Contar con un sistema de referencia de voltaje permite obtener y generar muestras estables en temperatura, voltaje y tiempo. Cuando se utiliza el voltaje de alimentación como referencia en los convertidores A/D y D/A existe el riesgo de que este sufra fluctuaciones por variaciones en la carga o condiciones ambientales. Asimismo es muy común que la fuente de alimentación sufra cambios

en el nivel de tensión de un punto a otro de la placa. Por esta razón el *emulador SISO* utiliza una referencia de precisión, la MAX874 (anexo A3.5), que trabaja como una fuente voltaje de 4.096 con una precisión inicial de 0.2% altamente estable en temperatura, tiempo y voltaje de alimentación, así como silenciosa.

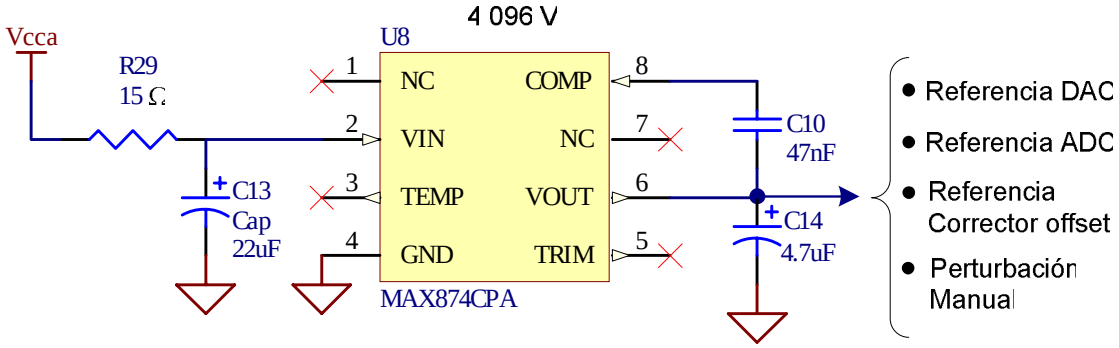


Figura. 3.17. (A2.5 – B(1,2))

El fabricante proporciona de manera explícita la manera en que hay que conectar la referencia de voltaje. R29 y C13 forman un filtro pasabajos de la fuente Vcca (5V). C14 es un capacitor de compensación de carga que se añade cuando la carga es superior a los 300 µA. Si el capacitor C14 es mayor a 1µF se añade C10, que es C14/100. Como se aprecia en la *Figura 3.17*, la referencia de voltaje se conecta a varias secciones del sistema. La suma de los consumos de corriente de estos elementos no debe ser mayor a 500 µA.

3.4.7 Conversor análogo-digital

Como se ha indicado con anterioridad, el DSC dsPIC30F4013 tiene un ADC de aproximaciones sucesivas con capacidad de realizar hasta 100000 conversiones por segundo. De los trece canales que pueden ser multiplexados hasta el ADC, solo se utilizarán dos como se muestra en la *Figura 3.18*. Adicionalmente este módulo permite la conexión de un voltaje de referencia alto (Vref+) y un voltaje de referencia bajo (Vref-). De esta manera si en la entrada de un canal del ADC se tiene un voltaje igual o mayor al voltaje de referencia alto, el valor digital

corresponderá al valor máximo (0xh0FFF, para 12 bits) y si el voltaje de entrada es igual o menor a Vref-, entonces el correspondiente valor digital será (0xh0000).

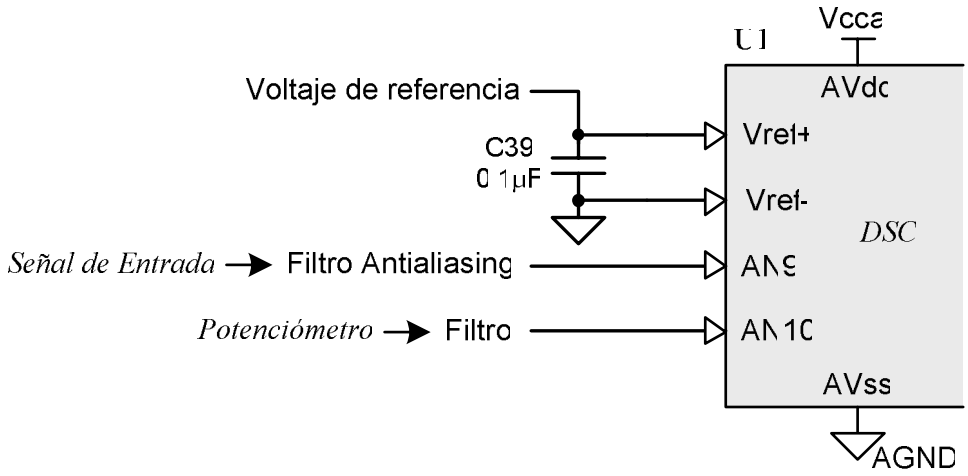


Figura. 3.18. (A2.2 – B3)

La referencia del ADC es 4.096 V proveniente del MAX874 y C39 es un capacitor de desacoplo de este voltaje. La referencia negativa (Vref-) se conecta a tierra. El DSC permite realizar una conexión interna entre Vref- y AVss, pero se dejó la conexión externa con el propósito de probar las dos configuraciones.

3.4.8 Conversor digital – análogo

Existen varias maneras de realizar una conversión D/A. La primera que se propuso para el *emulador SISO* fue la de generación de señales analógicas a partir de señales PWM filtradas. Esta opción es válida y utilizada en controladores pero cuando se requiere precisión y un buen desempeño en el dominio del tiempo y de la frecuencia no tiene un muy buen comportamiento. La primera razón es que la misma naturaleza de conmutación de una señal PWM genera un espectro complejo que necesita ser correctamente filtrado para tener una señal sin distorsión. Esto conlleva a la utilización de filtros de órdenes altos (y su inherente complicación en implementación) ó la generación PWM a altas frecuencias. Este último aspecto involucra una disminución en la resolución del generador PWM y es

difícil alcanzar la resolución impuesta de 12bits. Adicionalmente es difícil lograr que las señales trabajen riel a riel.

La segunda opción es buscar un DAC, entre los miles que existen en el mercado, que pueda trabajar con polarización entre 0 a 5 V, señales riel a riel y comunicación por puerto SPI. De hecho no resulta difícil encontrar DACs que cumplan con estos requisitos y la decisión final fue el MCP4922 (anexo A3.6), del fabricante Microchip, por facilidad en conseguir la parte y las buenas características que este ofrece: contiene dos DACs independientes (DAC_A y DAC_B) que comparten un solo canal SPI que puede trabajar hasta 20MHz, cada uno es de tipo de arreglo de resistencias de 12 bits y con un rendimiento de 200k muestras por segundo. Cada DAC contiene su propia referencia de voltaje y su salida posee un amplificador con ganancia programable de 1x ó 2x.

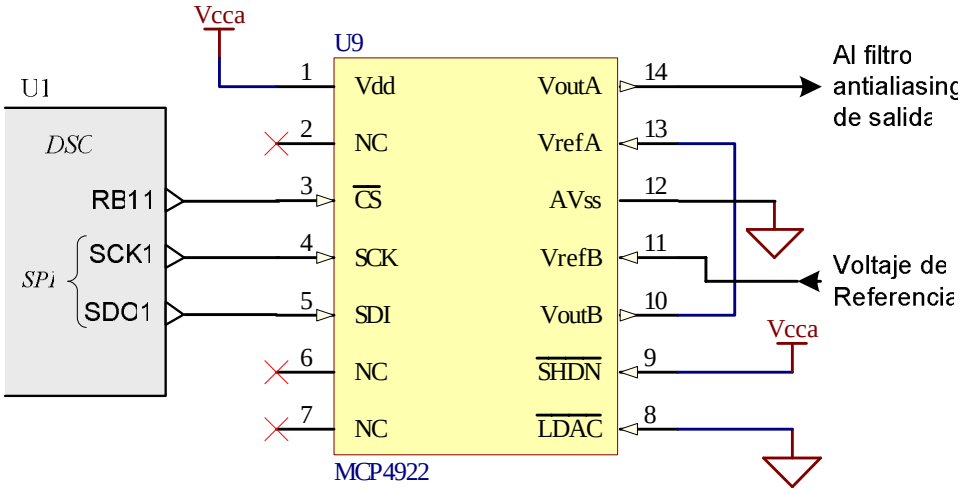


Figura. 3.19. (A2.5 – (B,C)(1,2))

Para que funcione el MCP4922 solo es necesario el reloj, los datos de salida del SPI y una señal de selección del chip (CS), en este caso correspondiente al pin RB11 del DSC. Cuando se conecta LDAC a tierra, los datos del registro digital se escriben en la correspondiente arreglo de resistencias en la transición de CS de bajo (Vss) a alto (Vdd).

Para poder establecer la escala de salida se utiliza el DAC_B como un generador de referencia de voltaje para el DAC_A. Como se puede observar en la *Figura 3.20*, la referencia del DACB es alimentada por la referencia de voltaje de precisión de 4.096. Según la escala de salida que se desee en A_3 (Salida analógica) se fija el voltaje de referencia VrefA. Por tanto se establece la ganancia de G = 3 de tal manera que para un voltaje de VrefA = 4 V, se tenga una salida en A_3 = 12V. El hecho de poder variar la referencia del DACA nos permite prescindir de elementos de precisión en el amplificador G (buffer de salida). Se entiende que los rangos de voltajes en A_1 y A_2 se mantienen sin cambios ya que el filtro pasabajos variable es de ganancia unitaria.

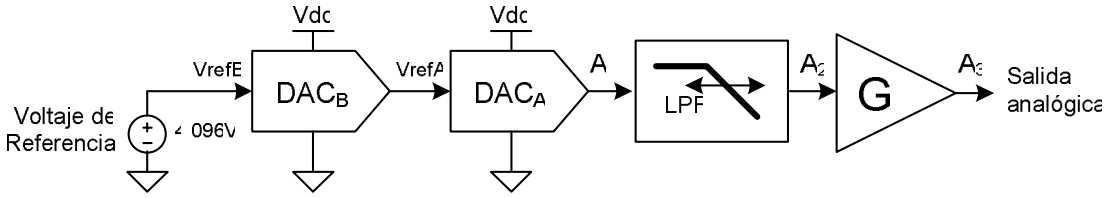


Figura. 3.20. Etapas de la salida analógica

3.4.9 Buffer de salida analógico y corrector de offset

El buffer de salida analógico contiene el amplificador de ganancia G=3 para generar los voltajes de 0 a 12V. Adicionalmente se añade un circuito capaz de compensar el offset que se haya añadido a la señal por el DAC y el filtro antialias. R28 proporciona una resistencia mínima al operacional con el fin de contar con mayor estabilidad.

El amplificador no inversor de la *Figura 3.21* cumple la función de adicionar el voltaje de corrección Vtrim que se resta a la señal amplificada. La ecuación [*28*] que define el comportamiento de este circuito están dadas por (3-9) si se supone que Vtrim = 0.

$$Vo = Vin \cdot \left(1 + \frac{R26}{R25 \parallel R27}\right) \qquad (3\text{-}9)$$

Si los valores de R25 = R26 = R27 = 68kΩ (5%) la ganancia del amplificador es 3. Para el caso de la señal Vtrim, esta también sufre una ganancia pero ésta corresponde a la de un amplificador inversor (Gtrim = 1). Por tanto la ecuación total del circuito de la *Figura 3.21* es:

$$Vo = Vin \cdot \left(1 + \frac{R26}{R25 \,||\, R27}\right) - Vtrim \cdot \frac{R26}{R25} \qquad (3\text{-}10)$$

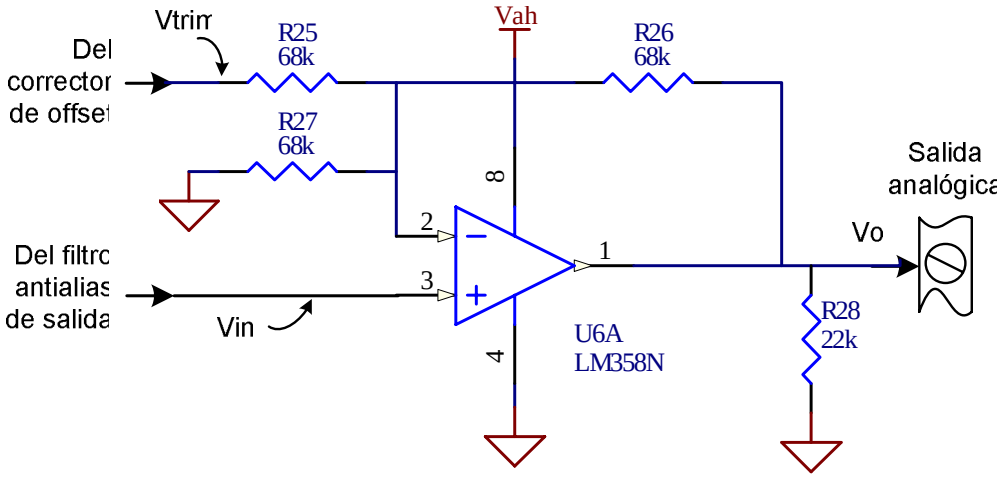


Figura. 3.21. (A2.4 – (B,C)4)

Si la salida analógica presenta un voltaje (offset) cuando a la salida del DACA se lo pone a cero, entonces aumentando Vtrim es posible encerar (poner en cero) el voltaje de salida. El circuito de compensación (*Figura 3.22*) está compuesto por el divisor de tensión R30 y R31 que generan el voltaje Ref2 que corresponde a 0.0625 Vref. Si Vref es 4.096, Vref2 = 256mV. El potenciómetro digital DP1B divide nuevamente el voltaje Vref2 de tal manera que en Vdiv se pueda tener voltajes de 0 a 255 mV en pasos de 1 mV. Se utiliza el operacional U7A como seguidor de tensión y poder de esta manera aislar impedancias entre el circuito compensador de offset y la etapa del buffer de salida. El voltaje a la salida de U7A es Vtrim que debido a la ganancia creada por R26/R25 = 1 define que se puede controlar el offset a la salida en pasos de 1mV.

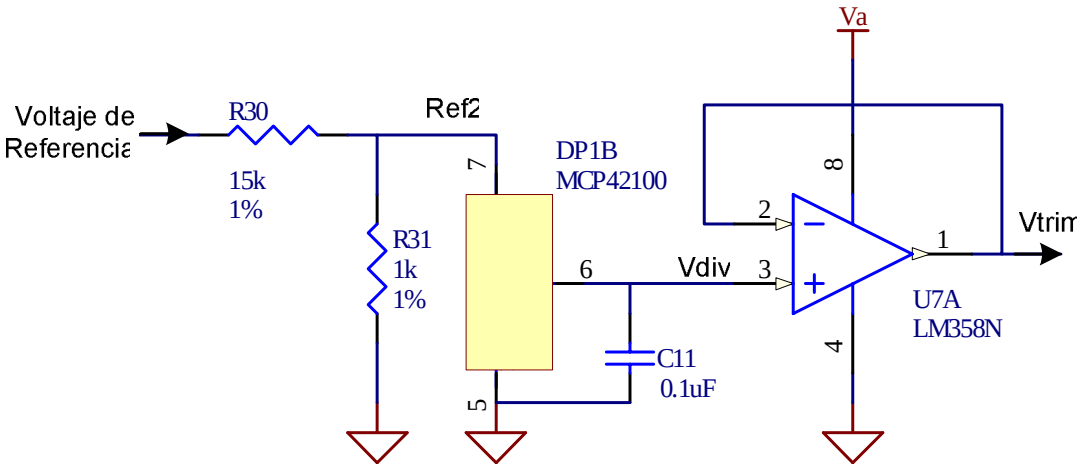


Figura. 3.22. (A2.5 – B(3,4))

3.4.10 Potenciómetro para perturbación manual

El Pot1 trabaja como un control de interfase con el usuario. Se alimenta con el voltaje de referencia con el fin de que el rango de variación sea de 0 - 4.096 V y correspondiente al máximo y mínimo del ADC. La red formada por R32 y C12 forman un filtro pasabajos con una fc = 10Hz. El U7B permite acoplar la impedancia del potenciómetro y del filtro con la entrada del ADC.

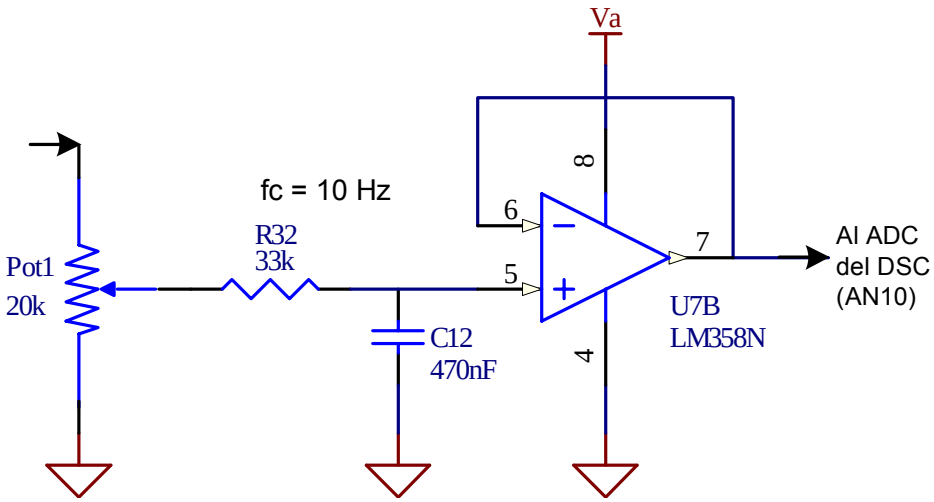


Figura. 3.23. (A2.5 – C3)

Si se comprueba que el circuito de salida no posee un offset considerable se puede sacar U7 y conectar Vtrim a tierra y unir las terminales 7 y 5 de U7.

3.5 ENTRADAS Y SALIDAS DIGITALES

3.5.1 Descripción general E/S digitales

Es necesario incluir en el *emulador SISO* entradas y salidas digitales con el fin de implementar las salidas en PWM, además de las posibles ampliaciones y aplicaciones que se le pueda dar a este Hardware en un futuro. Según el ReA4, estas entradas y salidas deben trabajar de 5 a 24 V. La razón de permitir un rango amplio de voltajes de funcionamiento es dar la flexibilidad de interactuar con equipos digitales de 5V y equipos industriales que trabajan generalmente con 24 V como son los PLC o lógica con relés electromecánicos.

Para el *emulador SISO* se disponen 3 entradas y 4 salidas digitales. El elemento central de interfase es el elemento DS2003 (anexo A3.7), el cual contiene 7 transistores darlington cada uno capaz de manejar voltajes emisor-colector de 50V y corrientes de hasta 500mA. La entrada del driver contiene una resistencia de 2.7kΩ con una corriente de base máxima de 25mA. Esto permite que voltajes hasta de 30V (máximo) polaricen la entrada. Para dar un rango de seguridad el voltaje máximo de entrada/salida será de 24V. Para el elemento DS2003 el voltaje de encendido empieza a partir de 2.4V (carga de 200mA).

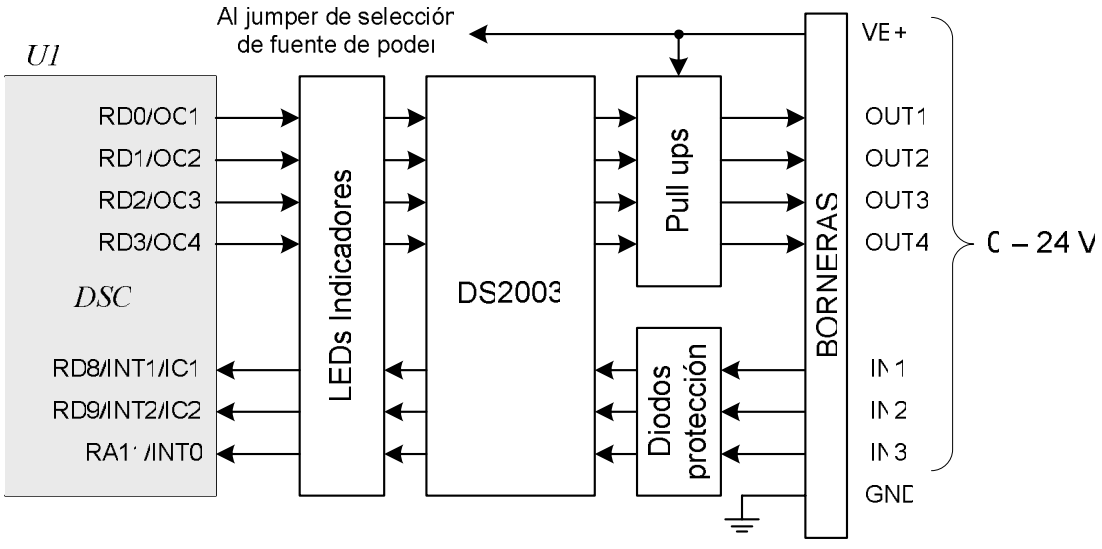


Figura. 3.24. Diagrama de bloques del circuito de Entradas/Salidas digitales

La *Figura 3.24* muestra los componentes del circuito de salidas y entradas digitales. Todas las salidas digitales del DSC pueden ser configuradas para trabajar con el respectivo módulo de comparación de salida (OCx) y de esta manera tener hasta cuatro canales generadores de PWM independientes. Todas las entradas pueden trabajar como fuentes de interrupción (INTx) y las entradas 1 y 2 conectarse a los módulos de comparación de entrada (ICx).

3.5.2 Salidas Digitales

El circuito de cada una de las salidas digitales se muestra en la *Figura 3.25* y puede tener uno de dos estados ya que cuando el pin del DSC está configurado como salida se puede drenar o suministrar corriente debido a su circuito tipo medio puente:

1. Si el puerto digital está en alto, el voltaje en A es +5V. La corriente que se inyecta a la base del driver Darlington es aprox. 2mA y suficiente para que se saturen los transistores, consecuentemente en B se tiene el voltaje de tierra. Es importante acotar que el DS2003 trabaja como colector abierto por lo que en este estado es posible “absorber” corriente por B, característica útil si se desea activar cargas como bobinas o pequeñas luminarias de hasta 250 mA de corriente. El LED indicador permanece apagado. La salida posee un voltaje a tierra.
2. El segundo estado ocurre si el puerto digital está en bajo. El pin del DSC trabaja como sumidero de corriente y en A se refleja el voltaje de tierra. No se tiene corriente de base en el driver Darlington y los transistores del DS2003 entran en la zona de corte. El voltaje de alimentación VE+ se refleja en B por medio de la resistencia “Pull-up”. El LED indicador se enciende y la corriente es absorbida por el DSC. La salida posee el voltaje de VE+. La ventaja de esta configuración es que el voltaje de trabajo de las salidas digitales se establece por medio de VE+, que puede estar entre 5 y 24V.

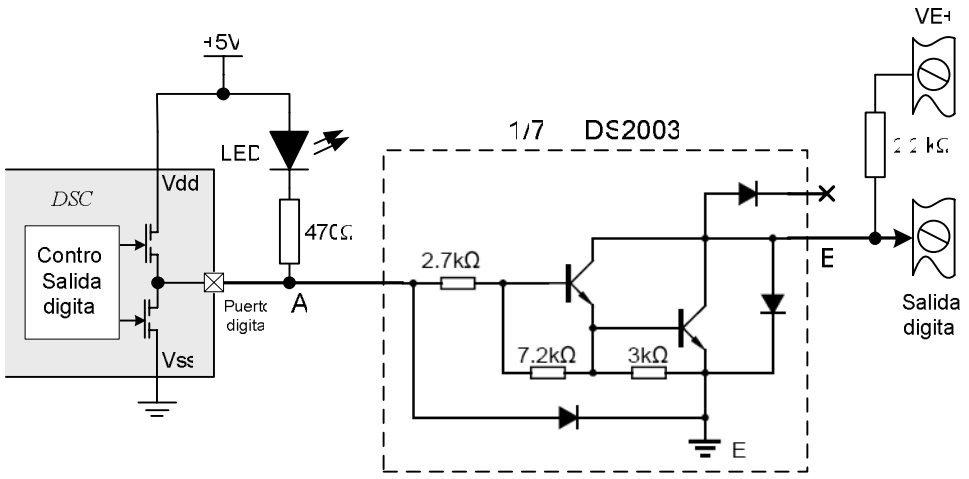


Figura. 3.25. Circuito de una salida digital

3.5.3 Entradas digitales

El circuito de la entrada digital (*Figura 3.26*) trabaja de manera similar al circuito de salida digital, con la diferencia que el equipo conectado a la entrada digital provee o no voltaje. Si el voltaje en C es mayor a 2.4V, el driver Darlington entra en saturación y D se conecta a tierra. El LED indicador se enciende indicando que en C hay voltaje presente. Si en C el voltaje es cero, el driver no drena corriente y el LED y la resistencia de 470Ω actúan como “pull-up” en la entrada del DSC.

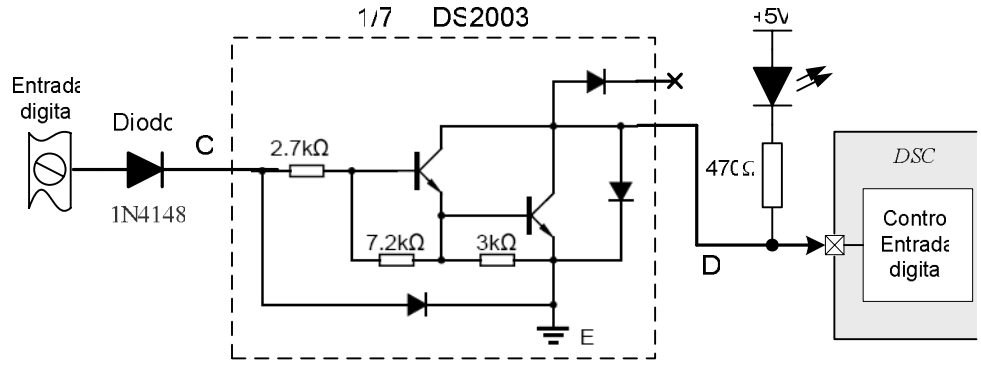


Figura. 3.26. Circuito de una entrada digital

En resumen, para tener un voltaje V+ en las salidas digitales el DSC debe poner la correspondiente salida en bajo. Si el DSC lee una entrada en bajo se entiende que la entrada en bornera posee un voltaje mayor a 2.4V.

3.6 FUENTE DE PODER

3.6.1 Especificaciones

La razón de realizar un sistema unipolar es la significativa simplificación del circuito y la facilidad de conseguir partes comerciales para el sistema. En el *emulador SISO* se deja la posibilidad de seleccionar entre un transformador o adaptador externo o la alimentación por medio de la bornera V+ (*Figura 3.27*). El tipo de fuente es lineal y aunque su eficiencia es baja, la baja generación de ruido es un atributo reconocido en el diseño de circuitos analógicos. La sencillez de diseño y la facilidad en conseguir las partes en el Ecuador son otros factores importantes por los que se seleccionó este tipo de fuente.

El diseño permite utilizar un adaptador externo (17-30VDC, no regulado) ó un transformador (16-20VAC). Si se selecciona trabajar con el voltaje V+, este no debe ser mayor a 24V. Se estima que el máximo consumo del *emulador SISO* es 300mA.

3.6.2 Componentes

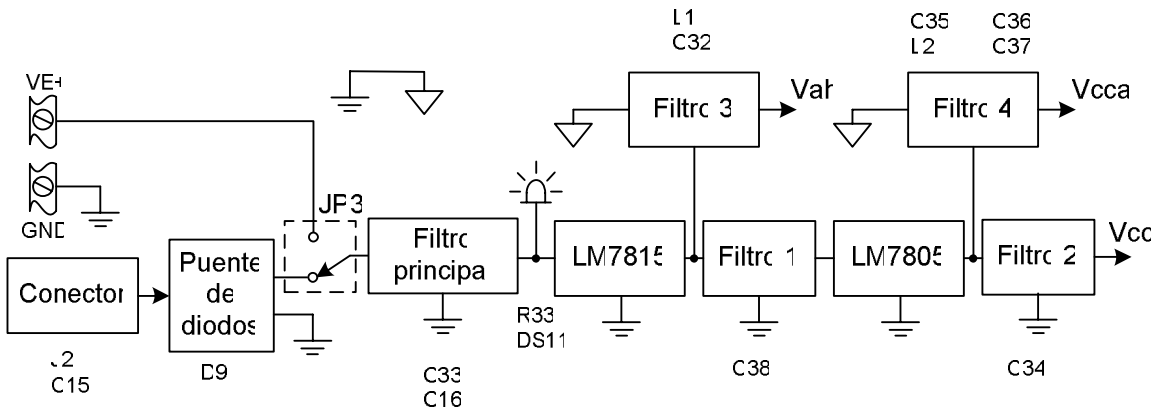


Figura. 3.27. Diagrama de bloques de la fuente de poder

Conector – Es un conector tipo jack hembra para la conexión del adaptador o transformador.

Puente de diodos – Rectifica si se conecta un transformador o protege de polarizaciones incorrectas si la fuente es DC.

JP3 – Permite seleccionar si se desea trabajar con adaptador ó con el voltaje de V+.

Filtro principal – Disminuye el rizado de entrada y acondiciona la señal para el LM7815.

Indicador – Permite saber si el *emulador SISO* se encuentra energizado y descarga el filtro principal ante la desconexión de cualquier fuente de alimentación.

LM7815 – Regulador lineal de 15[V]/1.5[A]. Entrega un voltaje regulado y con bajo contenido de ruido.

Filtro 1 – Estabiliza el voltaje de salida del LM7815.

LM7805 – Regulador lineal de 5[V]/1.5[A]. Disminuye el voltaje de 15 a 5V.

Filtro 2 – Estabiliza y regula el voltaje que se entrega a los componentes digitales (Vcc).

Filtro 3 – Filtro LC que proporciona el voltaje de 15V a los operacionales (Vah). El inductor es de núcleo de ferrita para filtrar las altas frecuencias generadas por el circuito digital y el capacitor estabiliza y filtra el voltaje. El capacitor de salida está conectado a la tierra analógica.

Filtro 4 – Filtro PI conformado por los dos capacitores y un inductor. Cumple la misma función que el Filtro 3 pero para la alimentación de 5V de los componentes analógicos (Vcca).

3.6.3 Capacitores de desacoplo

Estos capacitores van ubicados lo más cerca posible de los pines de alimentación de cada integrado (IC). Su función es importante en la supresión de ruido y el correcto funcionamiento de los componentes. Tienen la función de filtrar el ruido que hay en el sistema hacia el componente o evitar que el ruido que genera el componente se disperse por el sistema y afecte otros componentes. Para el caso del *emulador SISO*, todos los integrados tienen un capacitor de 0.1μF

en sus pines de alimentación y para los OPAMPs se incluye un capacitor de 10μF en paralelo a los de 0.1μF.

3.7 ELABORACIÓN DE LA PLACA PCB

3.7.1 Parámetros de diseño

Por facilidad en la elaboración de la placa se estableció de antemano que todos los elementos utilizados serían “through hole” y la placa utilizada de doble lado con recubrimiento de cobre de 1 oz y sustrato tipo FR4. El plano de la placa se encuentra en el anexo A4. Algunos de los lineamientos que se siguieron al hacer la placa PCB fueron [*29*], [*30*]:

1. Se trata de alejar lo más posible los elementos de comunicación y entradas/salidas digitales del circuito analógico con el fin de evitar interferencias y ruidos.

2. Los capacitores de desacoplo deben ir ubicados lo más cerca posible a los pines de alimentación de los componentes.

3. El tamaño de la placa debe ser lo más reducida posible con el fin de abaratar costos.

4. Se busca que el largo de los trazos sea lo más pequeño posible.

5. Se distinguen dos tierras: GND y AGND. La primera es la “tierra digital” del sistema y la segunda la “tierra analógica”. Estos dos planos se unen únicamente en un punto físico marcado como L3 en el diagrama.

6. La capa inferior se destina a tener amplias áreas de cobre a tierra, especialmente en la parte de los circuitos analógicos con el fin de proveer pantalla contra ruidos.

7. Alrededor del cristal oscilador y sus capacitores se provee un anillo a tierra como medida para controlar el ruido radiado de este.

8. En lo posible se trazan las líneas Tx y Rx paralelas para las comunicaciones USART y CAN. Para la comunicación SPI no es prioritario por ser un sistema de transmisión síncrono.

9. Los reguladores lineales no van a estar sometidos a grandes demandas de corriente, se calcula que la carga total está entre los 200 y 300 mA, por lo que se destina cierta área de cobre como disipador para ambos reguladores.

3.7.2 Lista de materiales y costos referenciales del proyecto

La lista de los componentes, cantidades, precios unitarios se encuentran en el anexo A5. Se estima que el costo de materiales del emulador es de 77.73 dólares y alrededor de 85 dólares incluyendo el ensamblaje.

CAPÍTULO 4

DISEÑO DEL SOFTWARE DEL EMULADOR

4.1 INTRODUCCIÓN AL DESARROLLO DEL SOFTWARE

4.1.1 Descripción General y especificaciones

Como se detalló en la sección 1.3.2 - *Software*, el comportamiento de un emulador HIL viene dado principalmente por el software que se ejecuta en su procesador. De la misma manera que se definió en el Capítulo 3 las características con las que debía contar el Hardware para poder ser un emulador, en este capítulo se imponen los requerimientos de Software y se detalla la manera en que se programó el controlador digital de señales (DSC) para trabajar correctamente en el emulador. Adicionalmente se describe el desarrollo de la interfaz gráfica de usuario (GUI) la cual crea el medio para que el usuario final configure correctamente el emulador SISO. Los principales requerimientos y funciones del programa embebido (PE) en el DSC se muestran en la *Tabla 4.1*, mientras que para la GUI se describen en la *sección 4.4*.

Funciones del programa embebido en el DSC	
FPE1	El PE debe administrar las comunicaciones por el puerto serial para que a través de este se descargue/cargue la configuración del emulador, los parámetros de calibración y se gestionen otros comandos para realizar pruebas del hardware.

FPE2	El PE debe guardar los datos de configuración y calibración en memoria no volátil y permitir recuperar los datos de la misma.
FPE3	El PE realizará todos los cálculos matemáticos correspondientes a las funciones de la planta SISO. Los flujos de estos cálculos y los parámetros de los mismos vendrán dados por los valores almacenados en la memoria no volátil del controlador y que a su vez fueron generados por el programa GUI.
FPE4	El PE es el encargado de gestionar los distintos controles e indicadores de usuario que posee la placa. Los LEDs indicadores deben brindar la información visual correspondiente respecto al contexto de la emulación así como de generar respuestas apropiadas ante el accionamiento de los botones.
FPE5	El PE tiene que configurar todos los periféricos del DSC y los componentes externos programables de manera correcta y oportuna. De la misma manera debe administrarlos y modificarlos eficientemente en el transcurso de la simulación.
Requerimientos del programa embebido en el DSC	
RPE1	El programa debe aprovechar al máximo la arquitectura y los periféricos del dsPIC30F4013 de manera que se obtenga el máximo rendimiento.
RPE2	El programa no debe tener código con funciones que realizan esperas por medio de la ejecución de código inútil.
RPE3	La sección de código que realiza el cómputo del modelo SISO debe ejecutarse con mayor velocidad que el mínimo ciclo de computo que se fije.
RPE4	El programa debe ser escrito de manera ordenada y comentada con el fin de facilitar la corrección, depuración y revisión de terceros.

Tabla. 4.1. Funciones y requerimientos generales del software embebido

4.1.2 Herramientas de desarrollo y documentación del DSC

4.1.2.1 Herramientas de desarrollo de software para el DSC

El desarrollo del software se apoya en varias herramientas computacionales que permiten la programación, depuración y análisis del código que se genera. Es importante enumerarlas (*Figura 4.2*) porque definen en gran manera el método de trabajo y la forma de programar. Adicionalmente es necesario mencionar los manuales y guías de programación y descripción del DSC proporcionados por el fabricante Microchip.

Microchip MPLAB IDE v7.1	Programa para desarrollar código y realizar depuraciones del mismo. Incluye el ensamblador, el *Linker* y el *Librarian*. No tiene costo alguno.
	• *MPLAB IDE® Users Guide (DS51519)* [*31*] • *MPLAB ASM30, MPLAB LINK30 and Utilities User's Guide (DS51317)* [*32*] • *MPLAB IDE Help*
dsPIC Language Suite	Se refiere al compilador en C MPLAB C30. Este permite la creación de código en ANSI C y su código base aprovecha al máximo la arquitectura del procesador. Para este proyecto se utilizó la licencia estudiantil que no realiza ciertas optimizaciones en el código. Este compilador incluye librerías estándar del lenguaje C, funciones para el manejo de los periféricos del DSC y funciones matemáticas.
	• *MPLAB® C30 C Compiler User's Guide (DS51284)* [*33*] • *dsPIC® Language Tools Libraries (DS51456)* [*34*]
MPLAB SIM Software Simulator	Simulador del dispositivo que se ejecuta en una PC y permite ver, modificar e inyectar valores a la memoria. Trabaja bien con C o ensamblador. Viene integrado al MPLAB IDE.

CMX-Scheduler for dsPIC	Librerías que permiten la implementación de un sistema operativo en tiempo real (RTOS) en el dsPIC30F. Es gratis y completamente compatible con las herramientas de Microchip.
	• *CMX – SHEDULER® User's Manual* [*35*]

Tabla 4.2 Herramientas de Software

Los documentos que contienen la información necesaria sobre la arquitectura, instrucciones y manejo de periféricos del DSC son:

- *dsPIC30F Family Reference Manual (DS70046)* [*24*]
- *dsPIC30F Programmer's Reference Manual (DS70030)* [*36*]
- *dsPIC30F Data Sheet General Purpose and Sensor Families (DS70083)* [*37*]
- *dsPIC30F3014, dsPIC30F4013 Data Sheet (DS70138)* [*38*]

4.1.2.2 Herramientas de descarga de código

Uno de los mayores impedimentos al probar un nuevo microcontrolador o DSC es la manera en que al dispositivo se lo programa, esto es, se le transfiere desde el computador el archivo con el código máquina (generalmente con extensión .hex) del programa que se ha compilado. Una de las grandes ventajas de los DSC de la familia Microchip es que se puede realizar este proceso con ayuda de un circuito sencillo que, por medio del puerto serial RS232, descarga el código al mismo. En el Anexo A6 se detalla el circuito (conocido como JDM Programmer). Para la descarga de los archivos .hex por medio de este método se utiliza el programa WinPic800 [*39*].

El proceso anterior permite grabar todos los registros de memoria, pero es necesario sacar y poner el dispositivo en el grabador o equipar al circuito impreso final con este circuito. Un método más conveniente es el usar un *Bootloader*

(cargador de arranque) que no es más que un pequeño código que se ejecuta justo después del arranque del dispositivo y controla la descarga del programa por medio de cualquier canal de comunicación del DSC. La ventaja radica en que se utiliza la misma infraestructura ya adecuada de la placa (en el emulador SISO el puerto serial y sus drivers) y elimina la necesidad de sacar el dispositivo cada vez que se desea programar. Para este proyecto se utilizó el *Ingenia dsPICbootloader* [*40*]. Los pasos para implementar el bootloader son:

1. Adecuar el programa en ensamblador que proporciona Ingenia del DSC para el modelo específico que se va a usar.
2. Con la ayuda de MPLAB IDE, compilar el archivo mencionado y generar el correspondiente archivo .hex.
3. Usando un grabador JDM (Anexo A6) y cargando el archivo .hex al programa WinPic800, descargarlo al DSC.
4. De la manera que se explica en el manual de operación del *bootloader* [*41*] Pág 7, modificar el archivo ibl_dspiclist.xml para incluir los datos del modelo específico de la aplicación.
5. Instalar el programa *Ingenia dsPICbootloader.*
6. Suponiendo que el circuito de aplicación tiene correctamente implementado la comunicación RS232, seguir los pasos que se describen en el manual de operación del bootloader [*41*], página 10.

4.2 CONFIGURACIÓN DEL HARDWARE DEL DSC

4.2.1 Configuración de Recursos y Periféricos del DSC

El dispositivo dsPIC30F4013 posee una serie de funcionalidades y periféricos que permiten cumplir con los requerimientos del emulador SISO. La configuración de estos recursos viene dada principalmente por el hardware que se diseñó y que se detalló en el capítulo anterior.

En la *Figura 4.1* se muestran los periféricos que se utilizan y a que pines del DSC están asociados (cuando es aplicable). A continuación se resume el uso y configuración de los periféricos[1].

4.2.1.1 Oscilador

Es la unidad encargada de generar el reloj del dispositivo. Para el emulador SISO se ha buscado la máxima velocidad por lo que se utiliza un cristal externo (XT) de 7.3728 MHz y adicionalmente se configura al PLL interno en aumento de 16x. Esto da un reloj interno de 117.9648 MHz. Cada ciclo de instrucción (Fcy) requiere 4 ciclos de reloj, por lo que la velocidad de procesamiento es de 29.4912 MIPS (Millones de instrucciones por segundo).

4.2.1.2 Funciones del sistema

Se tiene deshabilitada las funciones de POR (Reset al arrancar), BOR, LVD (Detección de bajo volyaje) y Watchdog Timer.

4.2.1.3 Interrupciones y Traps

Las interrupciones externas INTx se dejan deshabilitadas mientras que los Traps, o condiciones de error en la pila, en acceso a memoria, en operaciones matemáticas o en el oscilador se dejan habilitadas para indicar cualquier error de este tipo durante la ejecución del programa..

[1] El resumen de la configuración no representa una explicación de los periféricos. Corresponde al lector revisar el Manual de Referencia [*24*] del DSC.

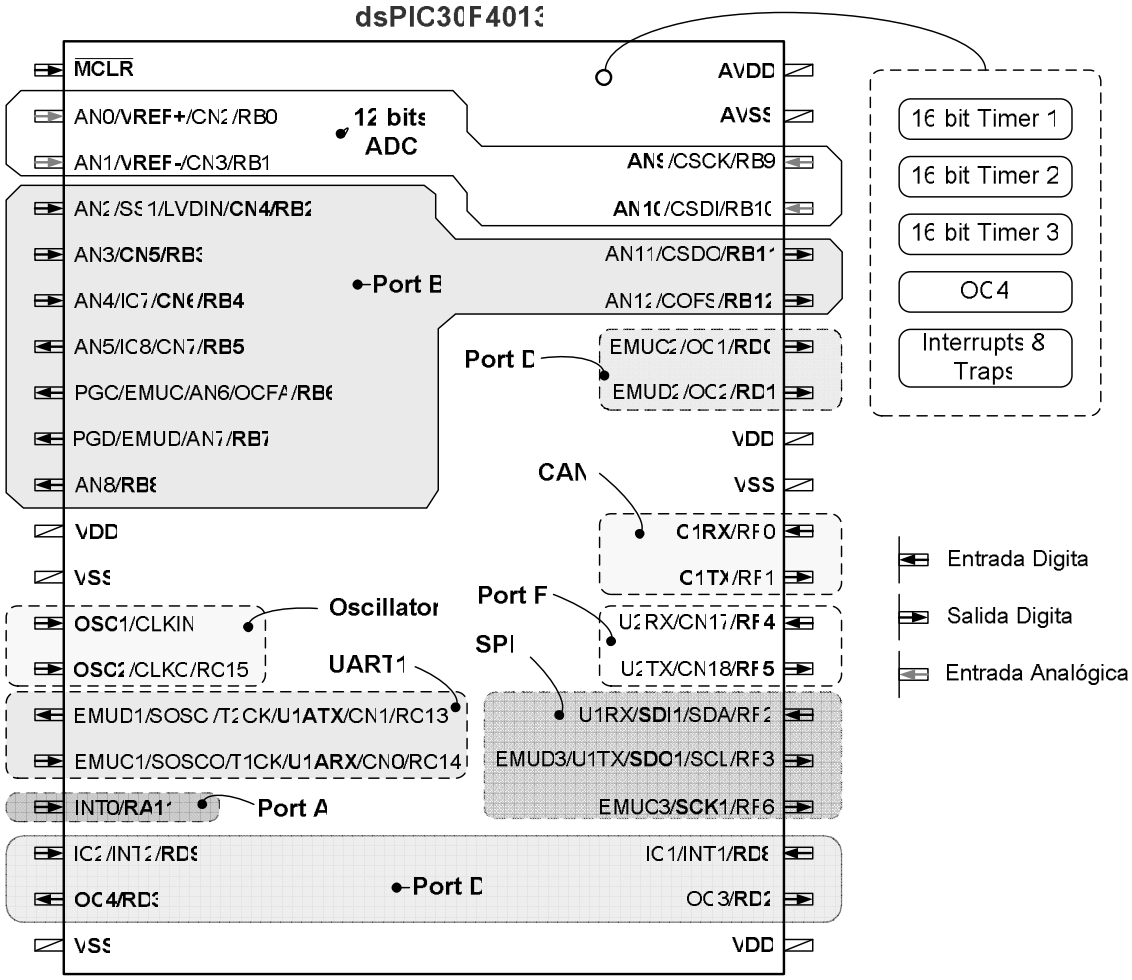


Figura. 4.1. Recursos utilizados en el emulador SISO

4.2.1.4 Conversor Análogo-Digital de 12 bits

Para este conversor se habilitan dos canales: AN9 (Señal externa acondicionada) y AN10 (Potenciómetro de variación). El voltaje de referencia alto (VREF+) corresponde al voltaje de 4.096V y el voltaje de referencia bajo (VREF-) corresponde a tierra. El reloj del módulo ADC corresponde al generador interno del mismo y trabaja a una velocidad de 1.03us (T_{AD}). Se deja configurado para que el módulo realice conversiones automáticas alternadas entre AN9 y AN10. El tiempo de muestreo se fija como 5 T_{AD} = 5.15 us que, sumado a los 14T_{AD} que se requiere en la conversión, resulta en un tiempo total de 19 T_{AD} = 19.57us para cada adquisición. Debido a que hay dos canales, el período de muestro para las

dos señales es 39.14 us ó 25.55kHz. El buffer de resultados se lo configura dividido, de esta forma se obtiene la secuencia de conversión: AN9→ADCBUF0, AN10→ADCBUF1, AN9→ADCBUF8, AN10→ADCBUF9, AN9→ADCBUF0,…. De esta manera antes de leer desde el programa principal el valor de AN9 o AN10 se verifica en qué sector del buffer (ADCBUF0-7 ó ADCBUF8-15) se realiza la conversión actual y se procede a leer el valor del otro sector del buffer. Es así que se asegura la lectura de un valor correcto del ADC sin tener que sincronizar la lectura con el fin de la conversión.

4.2.1.5 Puertos

Puerto A			
Pin	Sentido	Estado Inicial	Función
RA11	Entrada	-	Entrada digital externa I3

Puerto B			
Pin	Sentido	Estado Inicial	Función
RB0	Entrada	-	Configurados como entrada para que el módulo del ADC trabaje correctamente.
RB1	Entrada	-	
RB2	Entrada	-	Entradas de los pulsadores. Se deja la posibilidad de configurar los pull-up's internos para sacar las resistencias externas.
RB3	Entrada	-	
RB4	Entrada	-	
RB5	Salida	1	Salida que accionan los LEDs indicadores. Trabajan como drenadores de corriente.
RB6	Salida	1	
RB7	Salida	1	
RB8	Salida	0	Es la salida para realizar las pruebas de calibración del reloj.
RB9	Entrada	-	Configurados como entrada para que el módulo del ADC trabaje correctamente.
RB10	Entrada	-	
RB11	Salida	1	Activa/Desactiva el CS del DAC
RB12	Salida	1	Activa/Desactiva los CSs de los potenciómetros digitales.

Puerto D			
Pin	Sentido	Estado Inicial	Función
RD0	Salida	1	Salida digital externa O1
RD1	Salida	1	Salida digital externa O2
RD2	Salida	1	Salida digital externa O3
RD3	Salida	1	Salida digital externa O4/Salida PWM

RD8	Entrada	-	Entrada digital externa I1
RD9	Entrada	-	Entrada digital externa I2

Puerto F			
Pin	Sentido	Estado Inicial	Función
RF4	Entrada	-	Salida opcional para comunicación por RS232 usando handshake.(CTS) ó Rx del módulo UART2.
RF5	Salida	1	Entrada opcional para comunicación por RS232 usando handshake.(RTS) ó Tx del módulo UART2.

Tabla. 4.3. Configuración de los puertos

4.2.1.6 Módulo de comunicación SPI

Este módulo se configura como maestro sin sincronización de trama, en modo de transmisión de 16 bits, con estado de espera en alto, captura de dato a la mitad del reloj y transmisión en el flanco positivo del reloj. La frecuencia de operación se fija en 4.915 MHz para cumplir con los requerimientos de velocidad de los potenciómetros digitales. A esta velocidad del SPI es posible hacer que el DAC tenga un nuevo dato cada 3.26 us ó a 306.748 kHz.

4.2.1.7 Módulo UART1

Este módulo está configurado para trabajar en los pines alternos (U1ATX y U1ARX). Aunque este módulo tiene una gran cantidad de funcionalidades, para el emulador SISO este trabaja a 9600 kbps, sin paridad y con un bit de parada (8N1).

4.2.1.8 Módulo CAN

No se utiliza en la aplicación Emulador SISO pero se deja la posibilidad que para futuras aplicaciones se implemente este módulo ya que el hardware está listo.

4.2.1.9 Temporizadores

El Timer1 trabaja como generador de ticks del sistema. Su interrupción es la encargada de invocar a K_OS_Tick_Update() que es la que temporiza al sistema operativo. El Timer2 es ocupado por el módulo de comparación de salida y se utiliza cuando se configura al emulador para sacar una señal de PWM. El Timer3 es ocupado por la función de escritura al DAC y permite regular el tiempo que se mantiene en bajo el pin CS (Chip select) del DAC.

4.2.1.10 Módulo de comparación de salida OC4

El dsPIC30F4013 posee 4 módulos de comparación de salida que se los dejó intencionalmente accesibles a través de las salidas O1-O4. Para el Emulador SISO solo se utiliza el generador OC4. Este se activa según la configuración del usuario y produce una modulación al ancho de pulso proporcional a la salida analógica.

4.2.1.11 Funciones de configuración de los módulos

Función de entrada	void iniPeripherals(void)	Archivo
ADC de 12 bits	.global _configADC	drivers.h, config.s
Puerto A	.global _configPortA	
Puerto B	.global _configPortB	
Puerto D	.global _configPortD	
Puerto F	.global _configPortF	
SPI	.global _configSPI	
UART1	.global _configUART	
Timer1	.global _configTimer1	
Timer2	void iniPWMFunction(void)	plant.h, plantcomp.c
Timer3	.global _configTimer3	drivers.h, config.s
OC4	void iniPWMFunction(void)	plant.h, plantcomp.c

Tabla. 4.4. Funciones de configuración.

4.2.2 Drivers para el Hardware

Con el fin de proveer un cierto nivel de aislamiento del Hardware hacia el programa elaborado en C se desarrolló un conjunto de drivers escritos en ensamblador y otros en C que permiten un uso eficiente del DSC. El nombre de la función, los argumentos y el parámetro de devolución se encuentran descritos en la *Tabla 4.5*.

Nombre:	extern void escribirPOT(void);
Archivos:	driver.h, functions.s
Argumentos:	ninguno
Devuelve:	nada
Descripción:	Realiza una transferencia por el puerto SPI hacia los potenciómetros digitales de los valores contenidos en las variables Pot10, Pot11, Pot20, Pot21, Pot30, Pot31.

Nombre:	extern void AnalogWrite(unsigned int val, unsigned char channel);
Archivos:	driver.h, functions.s
Argumentos:	*val*: valor entre 0 a 4095 que se desea escribir en el DAC. *channel*: el canal del DAC que se cambia; CH_A(0) ó CH_B(1).
Devuelve:	nada
Descripción:	Esta función envía el dato *val* a través del puerto SPI para modificar el valor del canal especificado por *channel*. Con el fin de no esperar hasta que se trasmita toda la trama, esta función utiliza al timer3 para desactivar el CS (chip select). (RSwE2)

Nombre:	extern unsigned int AnalogRead(unsigned char *channel*);
Archivos:	driver.h, functions.s
Argumentos:	channel: valor entre 0 (CH_SIGNAL) y 1(CH_POT) que indica que canal del ADC se desea leer.
Devuelve:	La función retorna un valor entre 0 y 4095 correspondiente al valor analógico en la entrada del canal especificado.
Descripción:	El DSC muestrea las dos entradas analógicas automáticamente y las almacena alternadamente en 2 buffers separados. Por esta razón esta función averigua cuál fue la última conversión válida y en cuál de los dos buffers se encuentra, devolviendo este valor. De esta manera no es necesario que esta función espere a que el ADC termine la conversión de una de las entradas para tener el resultado. (RSwE2).

Nombre:	extern sysFlags KeypadScan(sysFlags sFlags);
Archivos:	driver.h, functions.s
Argumentos:	La clase que contiene los estados del sistema
Devuelve:	nada
Descripción:	Esta función escanea los pines del puerto B que están conectados a los pulsadores y realiza las siguientes tareas: Verifica si un pulsador ha sido presionado. En caso de serlo espera un tiempo antirebotes y vuelve a verificar el estado del pulsador. Si la pulsación es válida, modifica el valor de BUTTONSx en sFlags. Esta función está diseñada para ser llamada concurrentemente por un bucle de ciclo.

Nombre:	extern int ReadEE(int Page, int Offset, int* DataOut, int Size);
	extern int EraseEE(int Page, int Offset, int Size);
	extern int WriteEE(int* DataIn, int Page, int Offset, int Size);

Archivos	DataEEPROM.h, DataEEPROM.s
Argumentos:	*Page:* los 8 bits más significativos de la dirección de la memoria EEPROM. *Offset:* los 16 bits menos significativos de la dirección de la memoria EEPROM. **DataOut*: Puntero a la dirección de memoria donde se grabarán los datos de la EEPROM a la RAM. **DataIn*: Puntero a la dirección de memoria de donde se copiarán los valores de la RAM a la EEPROM. *Size*: número de bytes para leer/escribir en la EEPROM. Es un valor de 1 ó 16.
Devuelve:	-1 si hay un error en Size. Otro valor si se tuvo éxito.
Descripción:	Estas funciones son proporcionadas por Microchip bajo la librería DataEEPROM.h. Permiten Leer, Borrar o escribir la memoria EEPROM del dispositivo.

Nombre:	void WriteTxBuff(unsigned char data); void WriteHWTxBuff(void);
Archivos:	serial.h, serial.c
Argumentos:	data: el byte que se desea enviar por el serial.
Devuelve:	nada
Descripción:	WriteTxBuff() transfiere el valor de *data* a la pila FIFO de salida del UART. La pila de salida está definida por *UARTBUFFLENGTH*. WriteHWTxBuff() es invocada por la interrupción cuando el buffer de salida de 4 bytes que posee el DSC es completamente vaciado, de tal manera que realiza la transferencia de la pila FIFO al buffer físico.

Tabla. 4.5. Drivers para el manejo del hardware.

4.3 PROGRAMA DEL DSC

4.3.1 Breve descripción del sistema operativo multitarea

Para el desarrollo del software del Emulador SISO se escogió basar el programa en un sistema operativo en tiempo real (RTOS) perteneciente a CMX, el CMX-Scheduler. El CMX-Scheduler permite implementar un ambiente multitarea administrado con derecho a preferencia (*preemptive*) sobre los procesadores 24F/30F/33F de la familia Microchip. CMX provee las funciones necesarias y el sistema operativo para escribir aplicaciones multitarea de forma ordenada y bien diseñada. Esto evita que se tenga un código engorroso y lleno de bucles, condicionales y banderas lo que demora la detección de errores y la depuración en general. Adicionalmente ayuda a analizar y cumplir los tiempos necesarios para que el programa se comporte en tiempo real.

El sistema multitarea permite que en un solo procesador se ejecuten muchas tareas al conmutar concurrentemente las distintas tareas con el fin de obtener el efecto de que todas operan simultáneamente. Cada tarea representa un proceso en particular y puede ser coordinada con otras para definir su flujo. El CMX-Scheduler posee las funciones [*35*] descritas en la *Tabla 4.6*:

K_Task_Create	Crea una nueva tarea
K_Task_Start	Inicia una tarea
K_Task_Wake	Despierta una tarea
K_Task_Wait	Se mantiene en espera a una tarea
K_Task_Kill	Se elimina una tarea
K_Task_Coop_Sched	Realiza un cambio cooperativo
K_Event_Wait	Espera por un evento
K_Event_Signal	Se señala un evento desde una tarea a otra
K_Event_Signal	Se señala un evento desde una interrupción a una tarea
K_Event_Reset	Resetea un evento para una tarea en particular.

Tabla. 4.6. Funciones del RTOS CMX-Scheduler

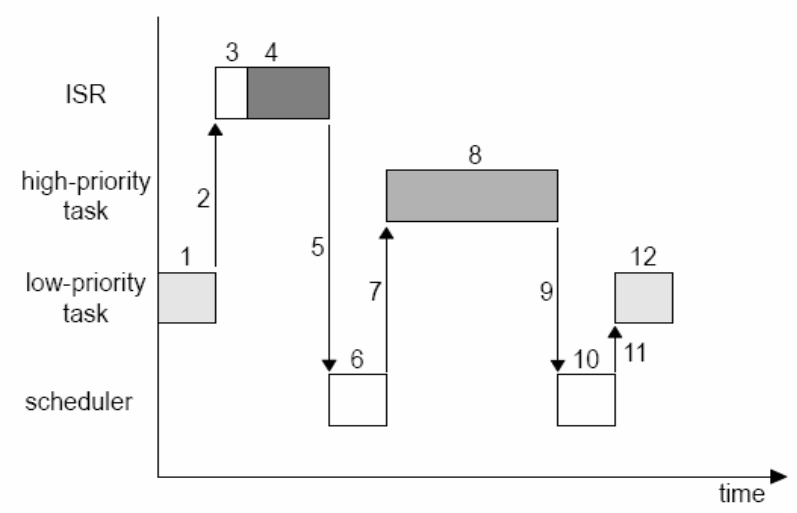


Figura. 4.2. Diagrama de tiempo de un RTOS con derecho de preferencia.

Una discusión completa sobre RTOS está fuera del alcance de este proyecto, por lo que se puede ampliar este tema con las referencias [*42*] y [*43*].

4.3.2 División funcional del programa

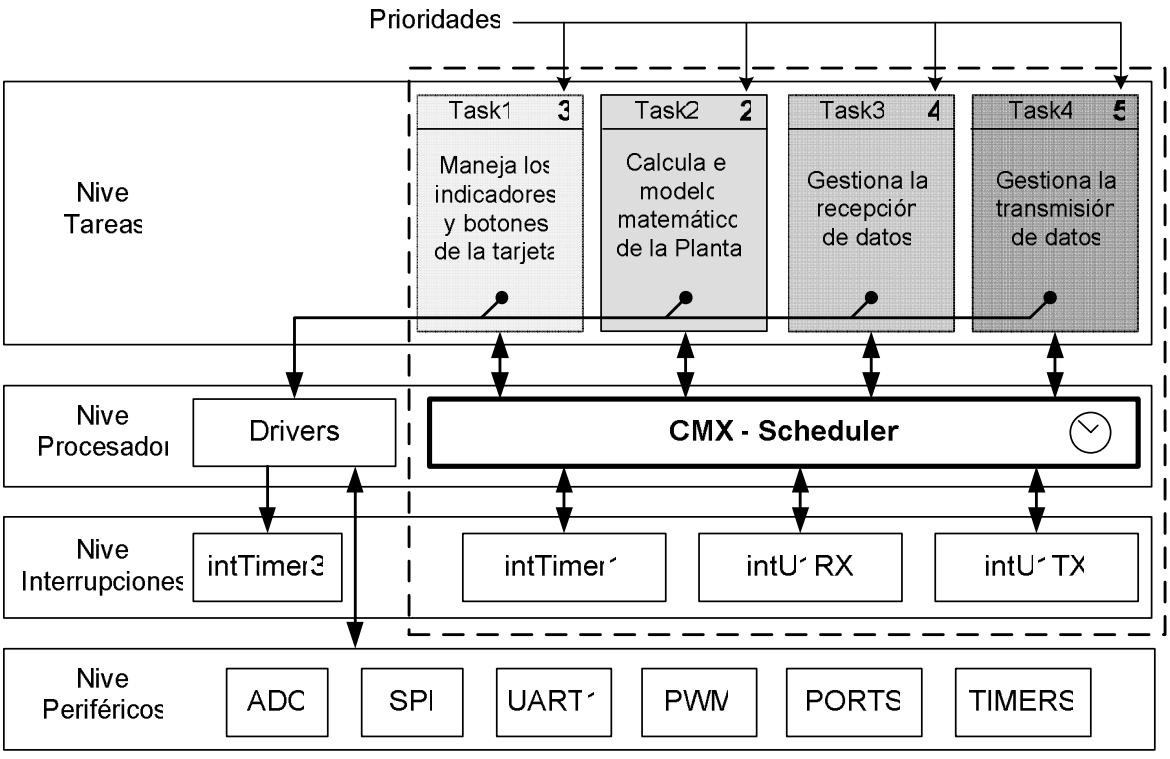


Figura. 4.3. Interacción entre los elementos de Software

La *Figura 4.3* permite tener una perspectiva de los distintos elementos que componen el software embebido en el Emulador SISO y las distintas interacciones entre si.

Los únicos procesos que se ejecutan realmente en paralelo son los de nivel de procesador y nivel de periféricos, éstos últimos permiten contar con procesos automáticos a nivel de hardware ya que alivian tareas como la adquisición de los datos analógicos (ADC), la temporización (Timers), el envío y recepción de datos por SPI ó UART y la generación de pulsos PWM. El tiempo de uso del procesador debe compartirse en los niveles de interrupción, procesador y tareas.

Desde un punto de vista de software, el DSC está dividido en 4 Niveles de ejecución que debido al uso del RTOS hace posible suponer que se ejecutan simultáneamente. Las interrupciones son generadas siempre por los periféricos, el nivel de procesador corresponde al código de los drivers y el RTOS, mientras que el código de las tareas es administrado por el RTOS.

Las prioridades de las 4 tareas se encuentran indicadas en la *Figura 4.3* (mientras menor el número, mayor es la prioridad). La ejecución de cada una de ellas es arbitrada por el CMX-Scheduler, el cual actualiza el estado de las variables de cada tarea, los temporizadores y eventos cada vez que se invoca a K_OS_Tick_Update() al producirse la interrupción del Timer1. Por esta razón la resolución de los timers del RTOS viene dada por la frecuencia de llamado a la función de actualización (tick del OS).

Mientras mayor número de llamadas a esta función se tenga en un segundo, más resolución en las tareas de espera se logra pero a su vez se pierde tiempo útil del procesador. Para el caso del Emulador SISO se estableció una frecuencia de tick de 10kHz en la cual el uso del RTOS “roba” aproximadamente el 1% del tiempo del procesador (trabajando a 29.4912 MIPS).

Cada tarea cumple las siguientes funciones:

- *Task1*: Es la encargada de realizar la revisión de los estados de los botones de interfase integrados al hardware además de encender/apagar los LEDs indicadores según el contexto. Actualiza las banderas indicadoras y, si las condiciones son correctas, inicia la Task2.

- *Task2*: Esta tarea se crea cuando el sistema empieza a emular la planta. Carga los datos desde la EEPROM que configura las funciones de retardo de transporte, función de transferencia, perturbación externa y no linealidad. Un timer del sistema operativo controla la frecuencia del bucle. Lee el ADC y escribe al DAC con el resultado del cálculo; también actualiza el ciclo de trabajo del generador PWM.

- *Task3*: Espera la interrupción debida a la recepción de un nuevo byte por el UART. Invoca a la función que procesa los datos del serial.

- *Task4*: Espera la interrupción generada por el UART cuando el buffer físico queda vacío con el fin de enviar los datos que faltan en la pila FIFO.

4.3.3 Componentes principales del programa embebido

A continuación (*Tabla 4.7 – 4.11*) se resumen y clasifican las principales funciones y variables del programa que trabajan con el RTOS en el emulador SISO.

Funciones de inicialización y del RTOS		
Librería	Nombre función/ explicación	Código fuente
csfunc.h	Contiene las definiciones de las funciones y variables del sistema operativo CMX-Scheduler.	cmx_inis.o csskv1a.o
drivers.h	void iniPeripherals(void) – Inicializa todos los periféricos	drivers.c

	void iniSys(void) – Arranca los timers y comunicaciones. void activateInterrupts(void) – Habilita las interrupciones	
	Estructura con los bits que controlan el flujo de la aplicación. typedef struct sysFlags{ unsigned RUN_STOP :1; Arranca(1)/Para(0) la emulación. unsigned BUTTON1 :1; (1) si el botón 1 ha sido presionado. unsigned BUTTON2 :1; (1) si el botón 2 ha sido presionado. unsigned BUTTON3 :1; (1) si el botón 3 ha sido presionado. unsigned SATIN :1; (1) si la entrada se ha saturado unsigned SATOUT :1; (1) si la salida se ha saturado. unsigned GO :1; *no usado con RTOS.* unsigned KILLP :1; *no usado con RTOS.* unsigned PINI :1; (1) si el proceso se ha iniciado unsigned PBLOCK :1; (1) para que no se pueda iniciar la emulación unsigned FILL :6; *relleno* }sysFlags; extern volatile sysFlags SFlags;	

Tabla. 4.7. Funciones de inicialización y del RTOS

Funciones de la Task1		
Librería	Nombre función/ explicación	Código
drivers.h	void cyclicDuties(void) – Realiza el escaneo de los botones, enciende/apaga los leds indicadores según el contexto y lleva a cabo el tiempo de espera de la comunicación serial.	drivers.c
	unsigned int SaturationCheck(unsigned int val,unsigned char where) - Verifica si val está dentro de los rangos superior/inferior de la señal correspondiente a where (entrada y salida). Retorna el valor delimitado según sea el caso.	

Tabla. 4.8. Funciones de la Tarea1

Funciones de la Task2		
Librería	Nombre función/ explicación	Código
plant.h	void iniBasicParams(void); Formatea y carga desde la EEPROM los valores que definen los rangos de E/S, los filtros analógicos, el flujo y la frecuencia de emulación	plantbas.c

	void iniTransferFunction(void); Formatea y carga desde la EEPROM las variables de la función de transferencia.	
	void iniDelayFunction(void); Formatea y carga desde la EEPROM las variables de la función de retardo de transporte.	
	void killPlantbas(void); Deshace el formato de las variables de la función de transferencia y retardo de transporte	
	unsigned int TransferFunction(unsigned int input);*	
	unsigned int DelayFunction(unsigned int input);*	
	void iniPerextFunction(void); Formatea y carga desde la EEPROM las variables de la función de perturbación externa.	plantcomp .c
	void iniPWMFunction(void); Formatea y carga desde la EEPROM las variables que definen el comportamiento del generador PWM (OC4).	
	void killPlantcomp(void); Deshace el formato de las variables de plantcomp	
	unsigned int PerextFunction(unsigned int input,unsigned int chpot,procFlow pFlow); *	
	unsigned int NolinFunction(unsigned int input); *	
	void PWMFunction(unsigned int input,procFlow pFlow) *	
	Estructura que define el flujo del modelo SISO de la planta typedef struct procFlow{ unsigned TFACT :1; (1) se ejecuta la función de transferencia unsigned RETACT :1; (1) se ejecuta la función de retardo unsigned NOLINACT :1; (1) se ejecuta la función de no linealidad unsigned PERTACT :1; (1) existe perturb. en algún punto del flujo unsigned PER1 :1; (1) perturbación en la posición 1 unsigned PER2 :1; (1) perturbación en la posición 2 unsigned PER3 :1; (1) perturbación en la posición 3 unsigned PER4 :1; (1) perturbación en la posición 4 unsigned PWMVFACT :1; (1) se activa el módulo OC4 unsigned PWMACT :1; (1) se activa el generador de PWM unsigned VFRECACT :1; no usado unsigned OUTINV :1; (1) el pulso PWM sale invertido unsigned baal :4; relleno }procFlow; extern volatile procFlow pFlow;	

Tabla. 4.9. Funciones de la Tarea2

Funciones de la Task3		
Librería	Nombre función/ explicación	Código
serial..h	void UartMain(void); - Función de entrada ante la recepción de un nuevo byte por el serial.	Serial.c
	Unsigned int uartCycle(void); - Realiza el filtrado del comando para llamar a la función respectiva.	
	void GroupACommands(void); - Grupo de comandos con mayor prioridad.	
	void GroupBCommands(void); - Grupo de comandos con prioridad intermedia.	
	void GroupCCommands(void); - Grupo de comandos que solo se ejecutan cuando el emulador está detenido. Carga/Descarga de la configuración hacia/de la EEPROM.	
	void GroupDCommands(void); - Grupo de comandos que solo se ejecuta cuando el emulador está detenido. Comandos de calibración.	
	void WriteTxBuff(unsigned char data); - Escritura de bytes al buffer FIFO de salida.	
DataEE PROM.h	void LoadValuestoEE(unsigned char *data,unsigned char EE_offset); - Graba los valores de *data* en el banco de la memoria EEPROM definido por *EE_offset*.	DataEE PROM.c
	void UpLoadValuesfromEE(unsigned char EE_offset); - Envía los datos del banco de la EEPROM definido por *EE_offset* por el serial.	
	void EraseEERow(unsigned char EE_offset); - Borra el banco de memoria de la EEPROM definido por *EE_offset*.	
	Estructura para el control de la recepción serial: typedef struct uartFlags{ unsigned char comd; guarda el comando del mensaje unsigned STATE :4; mantiene el estado de la recepción unsigned IN_COM :1; no usado unsigned WDON :1; (1) se activa el temporizador de tiempo fuera unsigned WDOF :1; (1) se ha vencido el tiempo de espera unsigned BUFFINI :1; no usado }uartFlags; extern volatile uartFlags uFlags;	

Tabla. 4.10. Funciones de la Tarea3

Funciones de la Task4		
Librería	Nombre función/ explicación	Código
serial.h	void WriteHWTxBuff(void); - Escribe la parte de la pila FIFO que le corresponde al buffer físico del módulo UART.	serial.c

Tabla. 4.11. Funciones de la Tarea4

4.3.4 Flujo de inicialización y tareas 1 y 2

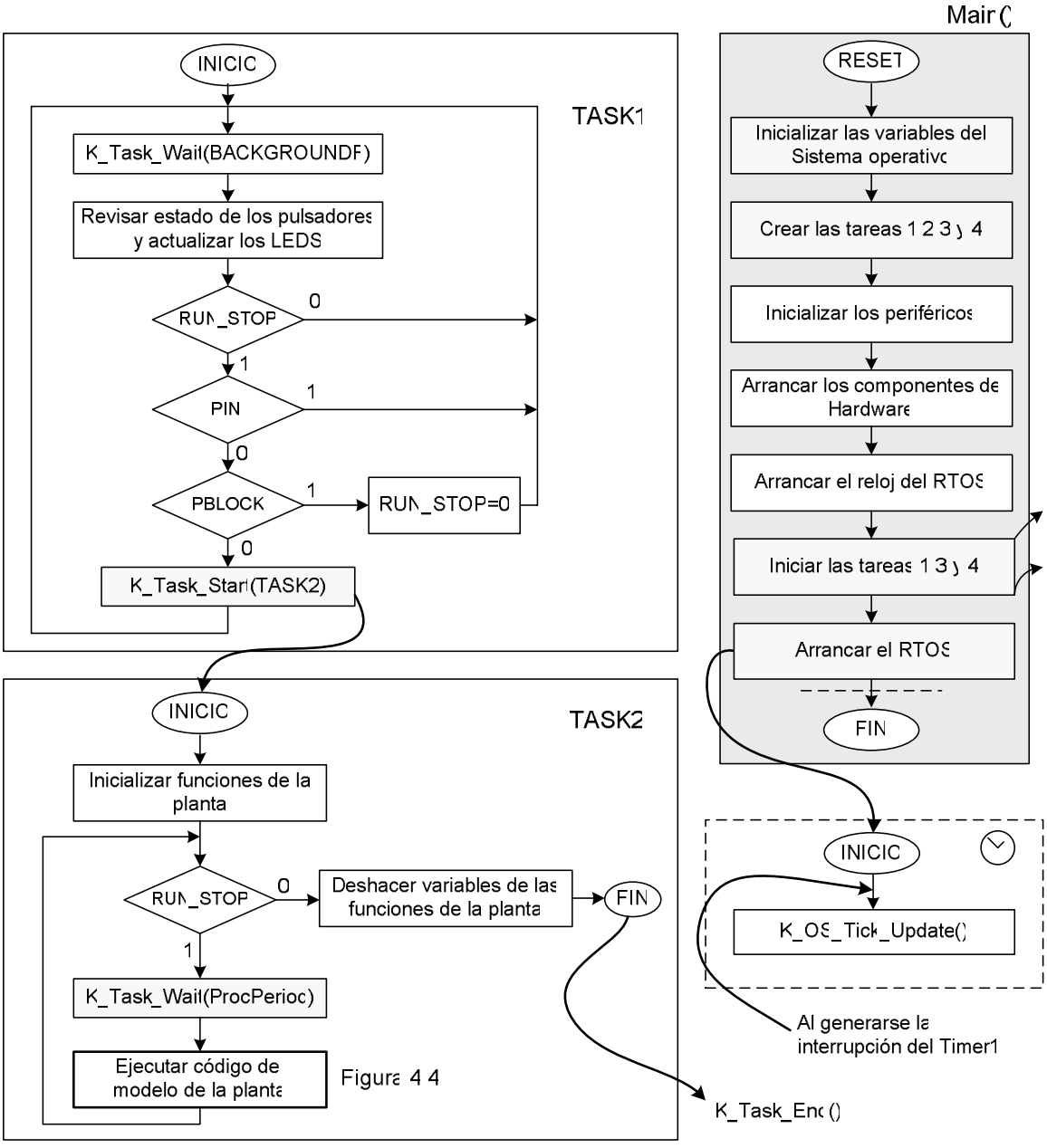


Figura. 4.4 Flujo de la inicialización, Task1 y Task2

El programa tiene su entrada por la función main() en SISOmain.c. Como se puede observar en el flujo de la *Figura 4.4,* después de la inicialización de todo el hardware se arranca el RTOS y las tareas 1,3 y 4. En este estado se queda el emulador esperando comandos por parte del usuario por medio de los botones o por medio de la GUI (serial).

Al arrancar la emulación de la planta se inicia la Task2, la cual inicializa los parámetros desde la EEPROM y entra en un bucle indefinido mientras RUN_STOP = 1. Usando la función del RTOS (K_Task_Wait()) se temporiza la ejecución del código del modelo de la planta, que tiene un período de simulación correspondiente a *ProcPeriod* número de ticks del RTOS. Cada vez que se invoca al código del modelo de la planta se ejecuta el flujo impuesto por la *Figura 4.5*.

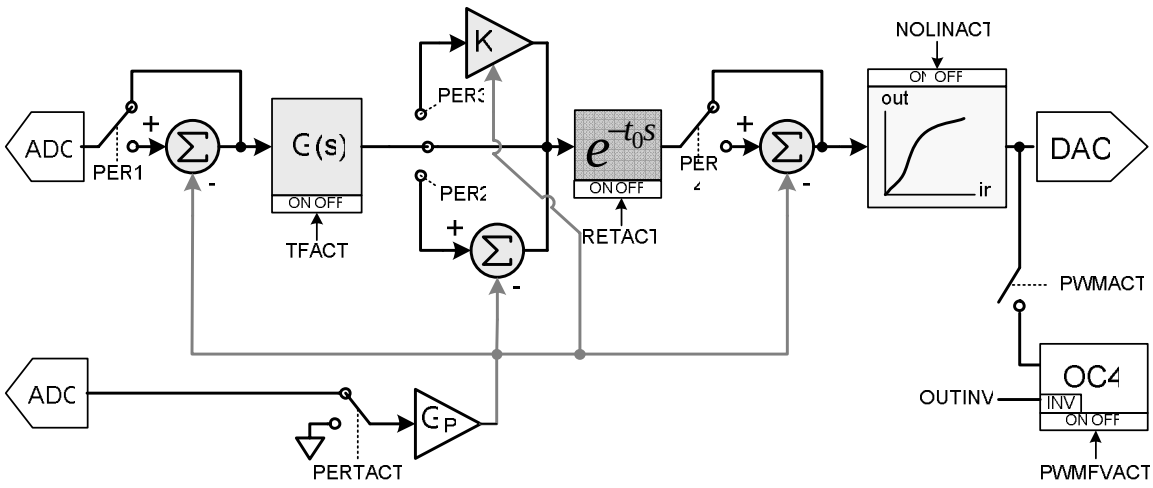


Figura. 4.5. Flujo del código de la planta

4.3.5 Funciones que componen el modelo de la planta

En el Anexo A7 se presenta el código de estas funciones apropiadamente comentado. En la *Tabla 4.12* se detallan los tiempos de ejecución de los distintos bloques de código en donde se puede observar que en el peor caso el ciclo de la planta toma 283 us. Es por esta razón que la máxima frecuencia de simulación

que se tiene es de 2000Hz (500us), dejando 210 us ó 6193 ciclos de máquina para el resto de tareas (más que suficiente).

4.3.5.1 Función de transferencia

Esta función implementa el filtro recursivo basándose en los coeficientes (obtenidos en la GUI utilizando la función de MATLAB *c2d*) que se envían y almacenan en la EEPROM, los datos actuales y anteriores de la entrada y salida. Cada vez que se invoca esta función actualiza la pila de retardo con la muestra de entrada y calcula la salida del filtro digital. Los cálculos se realizan en punto flotante (64bits) con el fin de evitar la acumulación de errores por los cálculos del filtro IIR (Ver *sección 2.2.2 - Ecuación de Diferencias*).

Se controla la saturación de la salida (de 0 a 4095) y si existen cálculos que salen de estos rangos se recorta la señal hasta estos límites. Sin embargo los cálculos internos almacenan los valores aún cuando estos pasen en ±20% (-819 a 4914) del rango real del DAC. Si se da el caso de que los cálculos arrojan datos más extremos (probable en plantas inestables), los cálculos se detienen hasta que las condiciones de los nuevos datos de entrada arrojen valores dentro de los límites del cálculo. En caso de que no haya dato de entrada que estabilice el sistema, la emulación deberá pararse y volver a iniciar.

4.3.5.2 Función de retardo de transporte

Esta función administra una pila FIFO en la cual se propaga el dato que se ingresa. Cada llamada a esta función desplaza a todos los datos una posición de tal manera que al cabo de n (largo de la pila) desplazamientos el dato se devuelve. El tiempo de retardo que se obtiene se define por el largo de la pila y el período entre invocaciones a esta función.

El emulador SISO dedica 766 bytes de memoria RAM para esta función. La ubicación del retardo se la hizo intencionalmente después de la función de

transferencia ya que se puede conocer analíticamente cuál es la pendiente máxima de la respuesta al escalón de dicha función de transferencia discreta. La GUI se encarga de obtener la mayor diferencia entre dos puntos contiguos de la función escalón unitario y calcula si la máxima es menor o mayor al 3.125% de la magnitud del impulso.

Si es mayor a 3.125% la pila FIFO trabaja en modo de 12bits lo que significa que cada “casillero” de la pila contiene un dato de 0 a 4095 (modo1). Si la mayor diferencia entre dos puntos contiguos es menor al 3.125% es posible trabajar en modo delta-sigma (modo2). Esto implica que lo que ingresa a la pila FIFO es solo la diferencia (+-127) entre muestras contiguas de manera que se reconstruye la señal a la salida; usando el modo2 se duplica efectivamente el tiempo de retardo para la misma cantidad de memoria.

4.3.5.3 Función de no linealidad

Esta función es implementada por medio de una tabla de búsqueda y un algoritmo de interpolación. El dato que recibe esta función varía entre 0 y 4095, pero la tabla de búsqueda solo tiene 256 puntos de conversión generados por la GUI. Por esta razón se entiende que la tabla de búsqueda está definida cada 16 puntos de la entrada. Este inconveniente es fácil de superar al hallar los dos puntos entre los cuales está comprendido el valor de entrada y obtener el valor interpolado lineal correspondiente respecto a estas dos referencias.

4.3.5.4 Función de perturbación externa

La función de perturbación externa escala el valor crudo (0-4095) al valor definido en la GUI. Posteriormente resta o multiplica este valor a la señal donde se ubicó la perturbación.

Función	Condiciones	Ciclos	Tiempo [us] @29.4912 MIPS
F. transferencia	Orden num.: 4 Orden den.: 4	Nom.: 6550 Máx.: 6660	222.10 225.83
	Orden num.: 1 Orden den.: 2	Nom.: 3420 Máx.: 3700	115.97 125.46
Retardo de transporte	modo1	35	1.19
	modo2	42	1.43
No linealidad		111	3.76
Perturbación externa	Aditiva Multiplicativa	496 873	16.82 29.60
Funciones adicionales			
Leer ADC	Signal+Pot	27	0.92
Escribir DAC		18	0.61
Ver saturación	Input+output	32	1.09
Actualización PWM		499	16.92
Sobrecarga adicional		65	2.20
Peor Caso: F.Transferencia (num:4,den:4) + Retardo(modo2) + NoLinealidad + Perturbación(Multiplicativa) + Todas las funciones adicionales =			
6660 + 42 +111 +873 + 27 +18 +32 + 499 + 65 = 8327 ciclos = 282.35us			

Tabla. 4.12. Rendimiento del código que modela la planta

4.3.6 Comunicaciones en el Emulador SISO

4.3.6.1 Recepción y transmisión de datos

Los diagramas de flujo de la transmisión y recepción de datos se muestran en las *Figuras 4.6 y 4.7*. Estos flujos son aplicables sea cual sea el protocolo de la aplicación. Para el caso de la recepción, el manejo lógico de la comunicación (protocolo) es manejada por la función UartMain(),

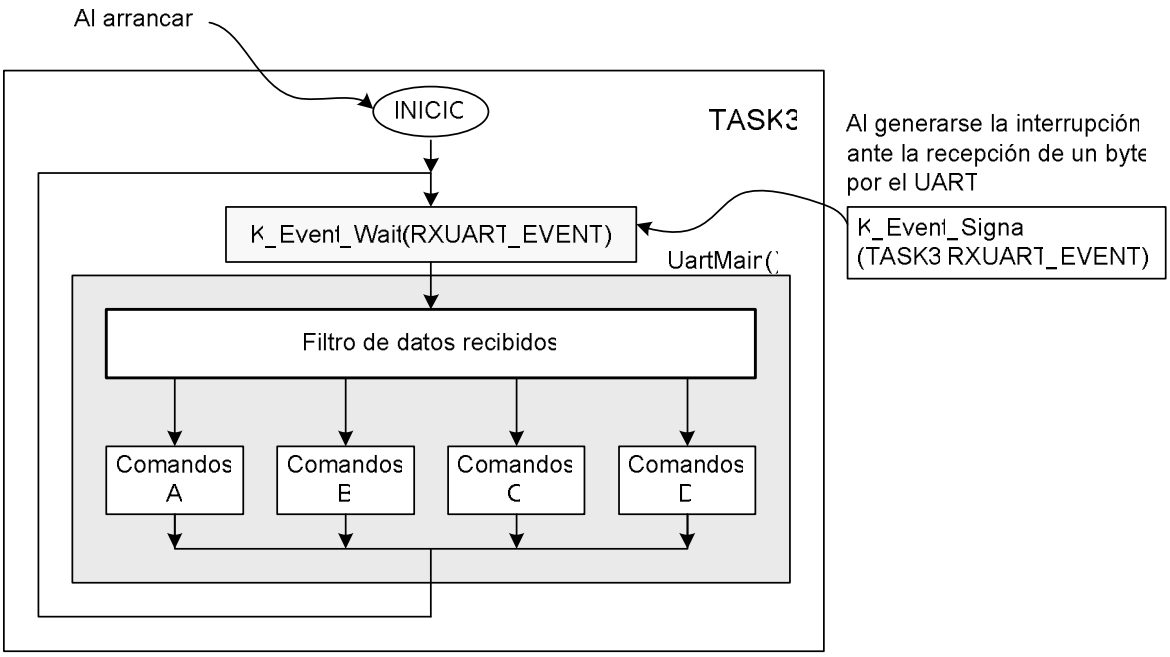


Figura. 4.6. Flujo de la recepción de datos por el UART

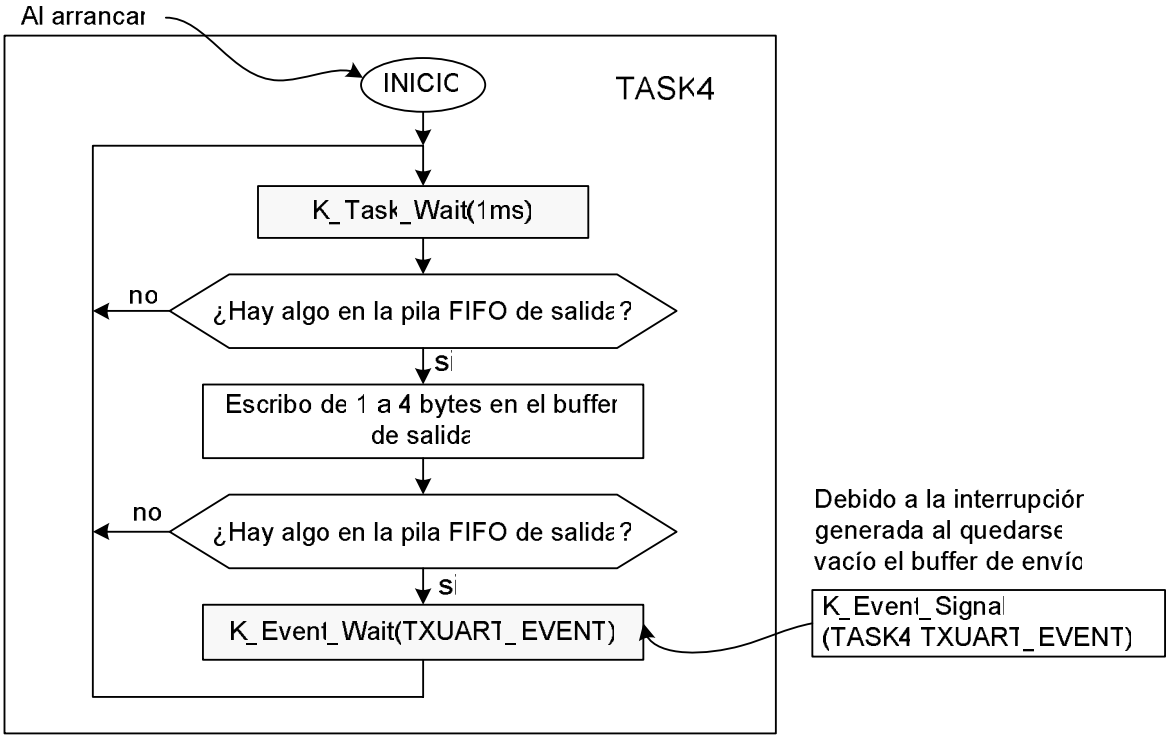


Figura. 4.7. Flujo de la transmisión de datos por el UART

4.3.6.2 Protocolo de comunicación

El protocolo de comunicación se basa en un sistema de 'polling' por parte de la GUI. Esto involucra que el emulador SISO siempre se comporta como esclavo, reaccionando o enviando la información que la GUI le indica. Se distingue dos tipos de mensajes maestro-esclavo: los que involucran solo dos bytes (PP1) y los que tienen una mayor longitud (PP2) en donde se envía el byte del menos al más significativo.

Los paquetes de respuesta (esclavo-maestro) son de tres tipos: (PR1) se envía cuando se ha ejecutado con éxito el comando enviado previamente y este no pedía información de vuelta; (PR2) es la respuesta cuando el comando previo pide valores (valores en la memoria, en los ADC's, etc.) y (PR3) si se detecta un paquete incompleto, inexistente o con error. La lista de comandos se detallan en la *Tabla 4.13*

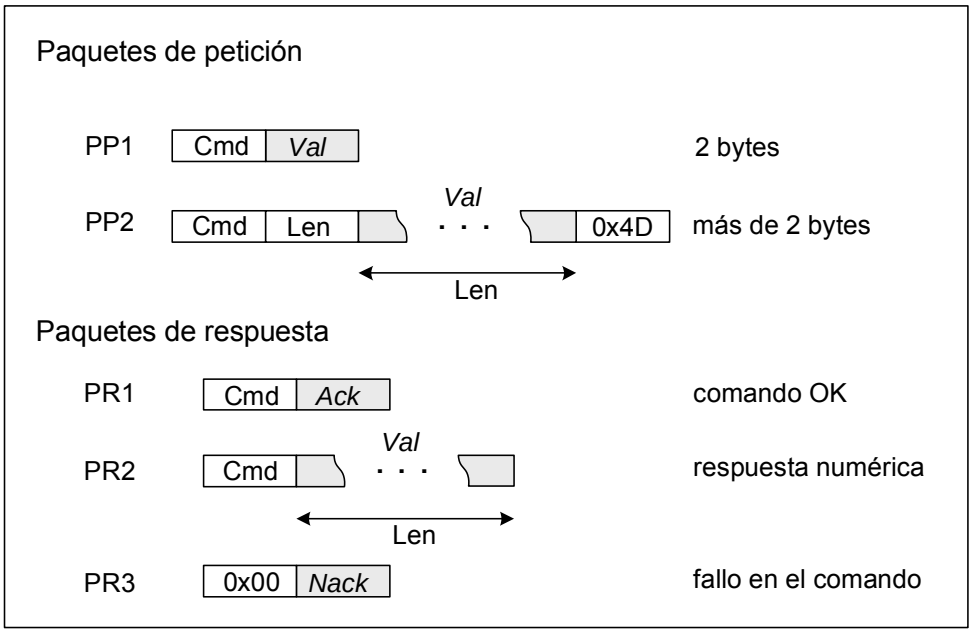


Figura 4.8. Formato de los paquetes de comunicación

Cmd	Mensaje		Resp.	Explicación
	tipo	Val		
Comandos D – No se ejecutan si el emulador está corriendo. Hay que detenerlo para que se puedan ejecutar. Se ejecutan funciones con fines de calibración.				
16	PP1	0	PR1,PR3	Realiza los cambios de valores a los potenciómetros digitales.
20	PP1	Pot10	PR1,PR3	Envía el nuevo valor para el Pot10
21	PP1	Pot11	PR1,PR3	Envía el nuevo valor para el Pot11
22	PP1	Pot20	PR1,PR3	Envía el nuevo valor para el Pot20
23	PP1	Pot21	PR1,PR3	Envía el nuevo valor para el Pot21
24	PP1	Pot30	PR1,PR3	Envía el nuevo valor para el Pot30
25	PP1	Pot31	PR1,PR3	Envía el nuevo valor para el Pot31
26	PP1	Channel	PR1,PR2	Realiza una lectura al canal especificado (1 – Señal, 2 - Potenciómetro). Devuelve [lsb msb] = 0-4095.
33	PP2	DAC_CHA	PR1,PR3	Se escribe el valor enviado val = [lsb msb] = 0-4095 en el canal A del DAC
34	PP2	DAC_CHB	PR1,PR3	Se escribe el valor enviado val = [lsb msb] = 0-4095 en el canal B del DAC.
36	PP2	*key*	PR1,PR3	El emulador entra en modo de prueba de reloj si key = [242 90]. Termina al enviar cualquier byte.
Comandos E – No se ejecutan si el emulador está corriendo. Hay que detenerlo para que se puedan ejecutar. Comandos para leer/escribir la configuración de/hacia la memoria EERPOM del DSC.				
65	PP1	*bank*	PR1,PR3	Borra el banco de memoria EEPROM especificado por *bank* (0-31).
66	PP1	*bank*	PR1,PR2	Retorna el contenido del banco de memoria *bank* (0-31). Envía 32 bytes, el primero es el lsb.
100	PP2	[*data* (32), *bank*]	PR1,PR3	Graba el contenido de *data* en el banco de memoria EEPROM especificado por *bank.*
Comandos B – Se pueden ejecutar en cualquier estado del emulador SISO. Controlan y averiguan el estado de la aplicación.				
129	PP1	0	PR1,PR2	Retorna [145, *est*], siendo est = 16 si el emulador está corriendo y est = 32 si está detenido.
131	PP1	0	PR1,PR3	Inicia la emulación (RUN_STOP = 1)
132	PP1	0	PR1,PR3	Para la emulación (RUN_STOP = 0)
133	PP1	0	PR1,PR3	Bloquea el inicio de una emulación.
134	PP1	0	PR1,PR3	Desbloquea el inicio de una emulación.
Comandos A – No implementados				

Tabla. 4.13. Lista de comandos manejados entre el emulador y la GUI

4.4 PROGRAMA DE INTERFASE DE USUARIO – GUI

4.4.1 Descripción general de la GUI

Las funciones y requerimientos de la GUI se detallan en la *Tabla 4.14*. El manejo de la interfase es descrita en el Anexo A1 – *Manual de operación*.

Funciones del programa de interfase GUI	
FGUI3	La GUI se encargará de formatear los datos de manera adecuada para el envío de los datos por el serial. Asimismo será función de la GUI el realizar las correcciones de los valores basándose en los parámetros de calibración recuperados de la memoria del DSC.
FGUI4	Los datos generados por la GUI se deben descargar a la memoria del DSC por el puerto serial y recuperarlos del mismo. Estos datos podrán ser grabados en medio magnético.
Requerimientos del programa de interfase GUI	
RGUI1	El programa debe correr en la mayor cantidad de plataformas posibles. Indispensable debe ser la compatibilidad con Windows XP.
RGUI2	La utilización de la GUI debe ser fácil e intuitiva, con las protecciones, mensajes, advertencias y guías gráficas necesarias para un correcto ingreso de datos.

Tabla. 4.14. Funciones y Requerimientos de la GUI

De acuerdo a las funciones y requerimientos planteados se pudo elegir el componente GUIDE de MATALB 7 como la herramienta para desarrollar la interfase el cuál tiene los siguientes puntos a favor:

- Matlab es una herramienta ampliamente difundida y conocida en ambientes educativos y experimentales, por lo que un programa desarrollado con esta aplicación facilita su difusión.

- La aplicación que se desarrolla en Matlab no necesita ser instalada. Si la instalación de Matlab y los componentes que requiere la GUI fueron instalados correctamente, entonces el programa se ejecutará sin problemas.

- Matlab puede ser instalado en varias versiones de Windows y en otros sistemas operativos. Por el punto anterior esto involucra que la aplicación GUI puede ser ejecutada en varias plataformas sin modificación o al menos con muy pocas.

- Matlab cuenta con un gran número de funciones y herramientas especializadas para el análisis de sistemas de control y manejo matemático. Esto evita el desarrollo de estos componentes si se desarrollara el programa en otra plataforma.

- Matlab provee la infraestructura necesaria para administrar la comunicación por medio del puerto serial.

4.4.2 Requerimientos de la instalación de MATLAB

Para que la aplicación se ejecute correctamente es necesario que se cumplan las siguientes condiciones:

- En la PC debe estar instalada la versión de Matlab 7.0 (R14) o superior.
- Tener instalado el Instrument Control Toolbox 2.0 o superior.
- Tener instalado el Control System Toolbox 6.0 o superior.
- Tener instalado el Fixed-Point Toolbox 1.0 o superior.
- El directorio actual debe ser la carpeta \sisoemu.
- Debe haber por lo menos un puerto serial RS232 en el equipo y el hardware del emulador debe estar conectado para poder acceder a ciertas funcionalidades.

- Si la PC no posee un puerto serial RS232 (común en computadoras portátiles) es posible trabajar con un adaptador serial-USB, ya que su funcionalidad ha sido probada para la GUI y el emulador SISO.

4.4.3 Componentes de la GUI

Al desarrollar una aplicación GUI en MATLAB es necesario dividir la aplicación en muchos archivos. Se distinguen dos principales: *.m y *.fig. Los primeros contienen el código de la GUI mientras que los segundos son la plantilla de la interfase gráfica. Los papeles que cumplen cada uno de los archivos se encuentran resumidos en la *Tabla 4.15*. El código posee los comentarios oportunos.

Componentes	Propósito
sisoemu.m sisoemu.fig sisoemudata.mat	Esta es la función de entrada a la GUI, adicionalmente contiene la lógica de la interfase global. Es desde aquí que se llama al resto de diálogos de configuración y comunicación y las funciones de descarga/carga de código. Sisoemudata contiene las variables de entrada para la GUI.
cargarDibujos.m dibDiagrama.m dibRetardo.m dibTransfer.m	cargarDibujos abre los archivos de imágenes que se encuentra dentro de la carpeta de sisoemu. dibDiagrama actualiza el gráfico que indica al usuario el modelo de la planta. dibTransfer actualiza los controles correspondientes a la función de transferencia y dibRetardo a los controles de la función de retardo. Realizan además el chequeo de valores.
comDlg.m comDlg.fig	Crean y manejan el cuadro de diálogo para seleccionar el puerto de comunicación y las propiedades que debe tener el mismo.
swrite.m	Esta es la función de interfase para la comunicación por puerto serial con el emulador SISO. Realiza el formateo de los datos y realiza las acciones correspondientes dependiendo de las respuestas que se obtienen.
calibconsole.m calibconsole.fig	Crean y manejan el cuadro de diálogo por medio del cual es posible leer y fijar los parámetros de calibración. Adicionalmente da acceso para que se realice la autocalibración de la entrada analógica y se de inicio/fin a la calibración del reloj.

utilDlg.m utilDlg.fig	Crean y manejan el cuadro de diálogo por el cual es posible variar a voluntad los parámetros de los potenciómetros digitales y salidas del DAC. Permite leer los valores de las entradas analógicas.
tfconsole.m tfconsole.fig	Crean y manejan el cuadro de diálogo que permite el acceso/visualización de los valores de la función de transferencia. Valida el orden, la validez de los elementos ingresados y devuelve los parámetros depurados.
nolinDlg.m nolinDlg.fig	Crean y manejan el cuadro de diálogo que permite modificar la tabla de búsqueda de linealidad. Permite seleccionar entre distintos tipos comunes de no linealidad o cargar desde el escritorio un vector arbitrario que lo describa.
pwmDlg.m pwmDlg.fig	Crean y manejan el cuadro de diálogo que permite ver y modificar la configuración de la salida de PWM. Valida los datos que introduce el usuario y los normaliza al máximo valor con el fin de generar la tabla de búsqueda definitiva.
cargarConfiguracion2.m descargarConfiguracion2.m	Estas dos funciones son las encargadas de formatear los datos que se van creando en la GUI. Por medio de la revisión de los estados de los controles y los valores ingresados por el usuario, descargarConfiguracion2 hace que los datos de la interfase se transformen en un arreglo de bytes de 32x32. Estos datos son los que se envían a la memoria del EEPROM del DSC. Estos datos son también los que se guardan en el archivo magnético. cargarConfiguracion toma los valores crudos de la memoria (EEPROM o del archivo magnético) y los convierte a los datos que definen la interfase GUI.
imageneeprom.XTRAINFO	Este archivo corresponde al arreglo de 32x32 bytes que se envía a la memoria EEPROM. Es usado para poder cargarlo en el MPLAB y de esta manera depurar el programa embebido del DSC.
readEEbank.m writeEEbank.m	Administran el envío/recepción de datos hacia/desde la memoria EEPROM del arreglo de 32x32bytes por medio del serial.

Tabla. 4.15. Organización y propósito de los archivos de la GUI

CAPÍTULO 5

PRUEBAS AL SISTEMA Y RESULTADOS

5.1 DESCRIPCIÓN DEL CAPÍTULO

En este capítulo se presentan las pruebas realizadas al emulador SISO y los resultados de las mismas. Se describen dos tipos de pruebas[3]: las primeras obtienen el comportamiento del sistema como equipo de adquisición de datos y las características del mismo. Las segundas pruebas consisten en comparar las simulaciones de MATLAB de un conjunto de plantas con diferentes funciones de transferencia y el comportamiento de las mismas pero configuradas en el *Emulador SISO* y obtenidas experimentalmente en laboratorio.

5.2 PRUEBAS AL SISTEMA DE ADQUISICIÓN DE DATOS

5.2.1 Ruido

Para conocer el contenido de ruido del sistema se aplicó la técnica que se describe en [*30*], la cual consiste en realizar un número grande de mediciones del resultado de la conversión análoga-digital ante la presencia de un valor constante en la entrada. A partir de esta colección de datos se realiza un histograma el cual indica gráficamente la distribución de los valores de las lecturas. Mientras mayor el número de distintos valores, peor el desempeño del sistema. Para implementar este experimento se conectó la salida analógica del emulador SISO a la entrada analógica; posteriormente se fijó un valor constante a la salida y se realizaron 1024 lecturas consecutivas de la conversión del ADC. Los resultados en la Tabla 5.1.

[3] Se usaron como OPAMPs en U5 y U7 el MCP6022; para U6 el TC913B.

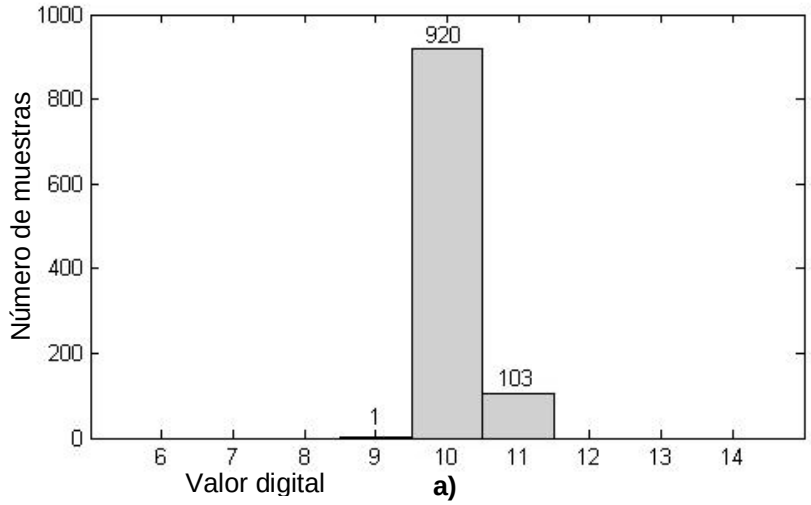
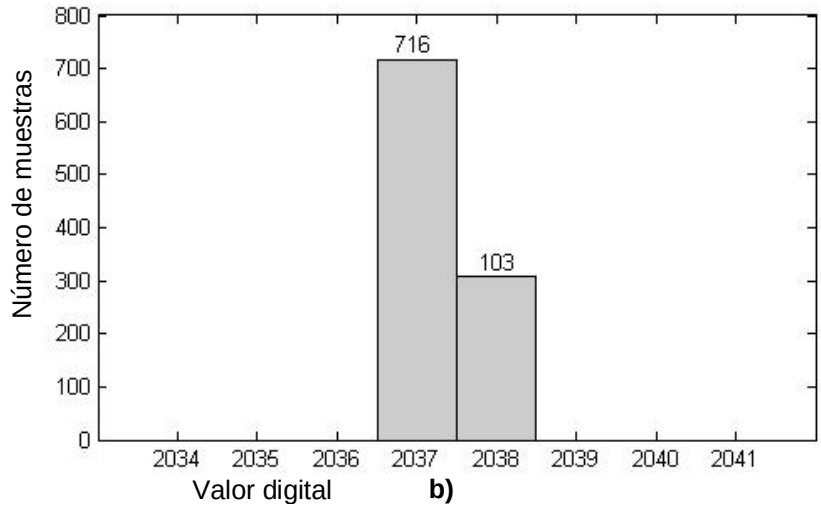
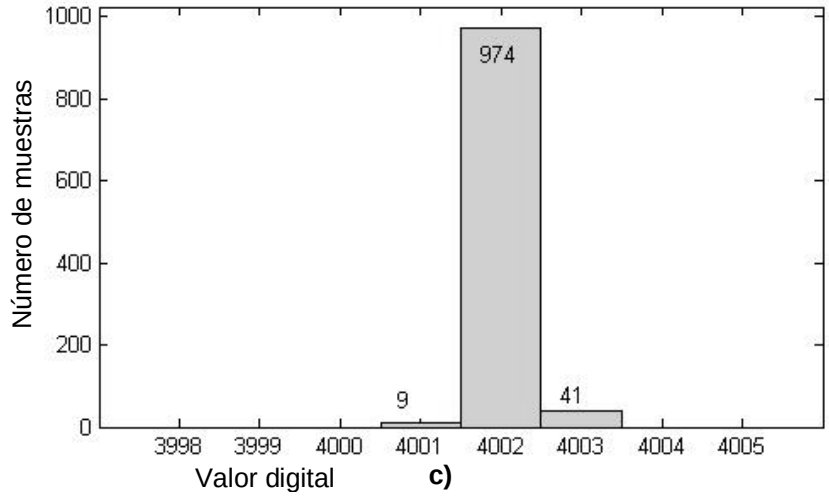
Condiciones	Histogramas
fs = 20Hz Entrada: 13mV Escala: 5V	
fs = 20Hz Entrada: 2.5V Escala: 5V	
fs=20Hz Entrada: 11.74V Escala: 12V	

Tabla. 5.1. Pruebas de ruido en el sistema

5.2.2 Linealidad

La linealidad indica la fidelidad con la que un sistema de adquisición adquiere/reproduce los datos. Para el caso del *emulador SISO* se sigue un procedimiento similar a la prueba de ruido (conectando la salida con la entrada) pero esta vez se varía el valor de la salida analógica desde su mínimo (0 digital) hasta su máximo (4095 digital). Esta prueba demuestra de una manera sencilla la linealidad de todo el sistema que incluye el DAC y el ADC, los filtros antialiasing de entrada y salida, el amplificador de ganancia programable en la entrada y el amplificador a la salida. Todos estos elementos añaden cierto grado de no linealidad.

Los valores de no linealidad se obtienen a partir de la resta de la recta que mejor ajusta la colección de datos obtenido con los respectivos datos [*44*] como se muestra en la Figura 5.1.

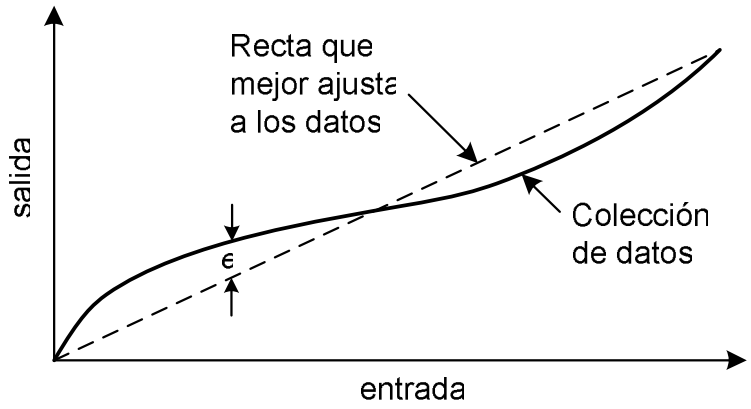
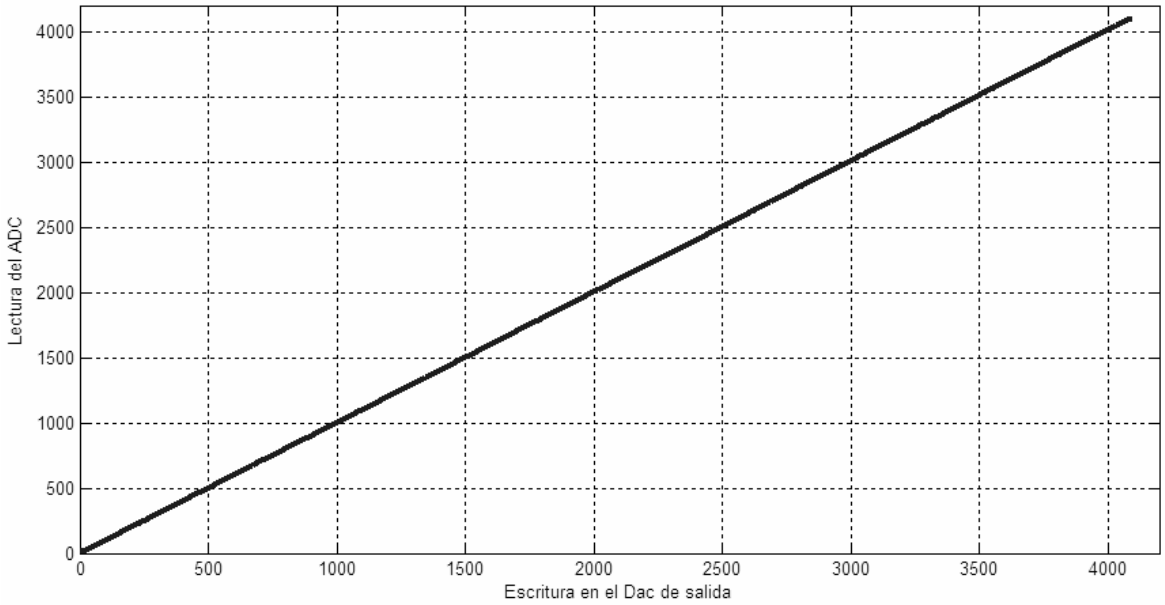


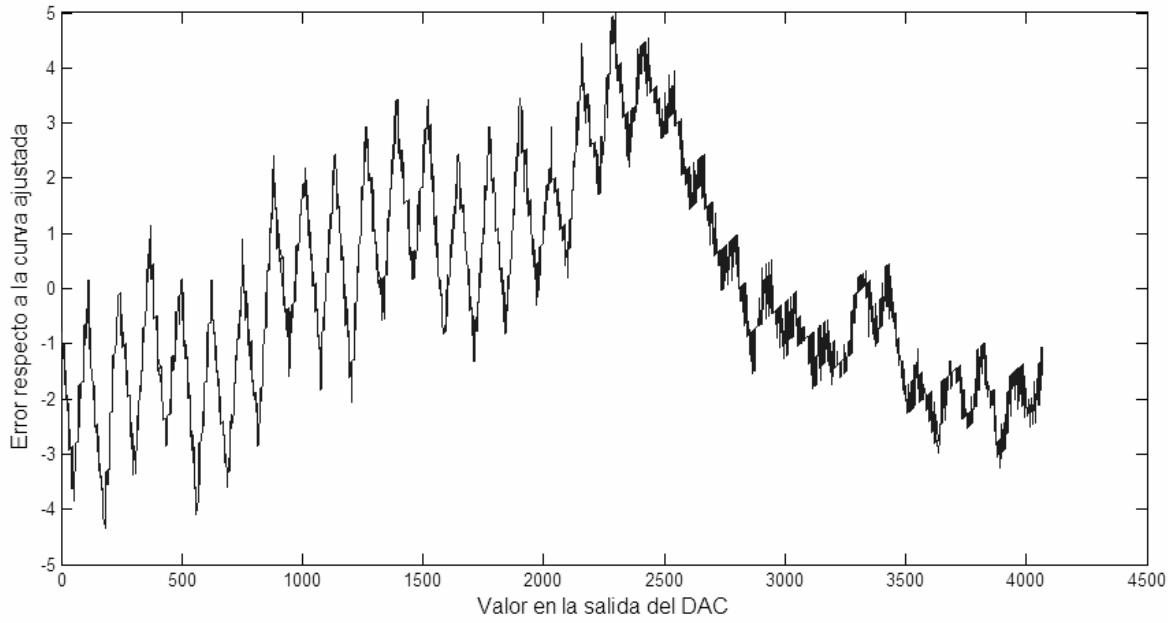
Figura 5.1 Descripción gráfica de no linealidad.

Se realizaron dos pruebas en los extremos de las escalas, esto es, la primera con escala de 5V (Tabla 5.2) y la segunda con escala de 12V (Tabla 5.3). Las escalas intermedias tienen un desempeño comprendido entre los dos extremos.

Escala del DAC_B: **1662** (0 - 5V)
Escala de la entrada (Pot10): **217** (0 - 5V)
Rango de los valores tomados en cuenta: **16 – 4090**
Ecuación de la recta que mejor ajusta: $y = 1.0039 \cdot x - 4.1083$
Mayor error (diferencia) positiva: **4.961**[bits] en el punto 2296
Mayor error (diferencia) negativa: **-4.361** [bits] en el punto 176
Error de linealidad: **+0.121% y -0.106%**.



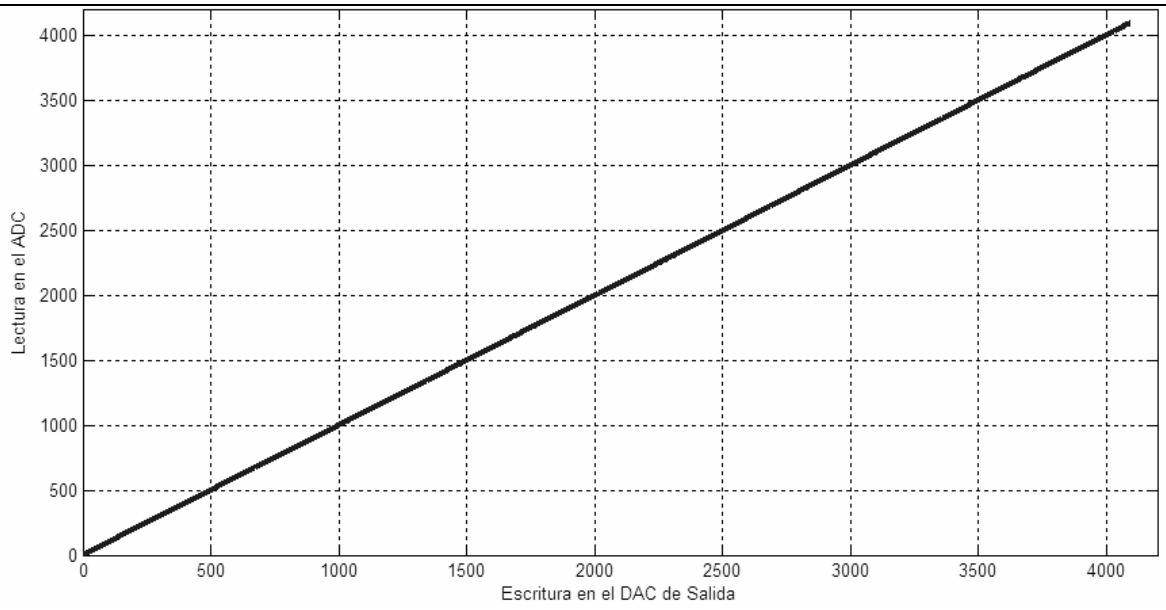
a) Relación de la salida con la entrada



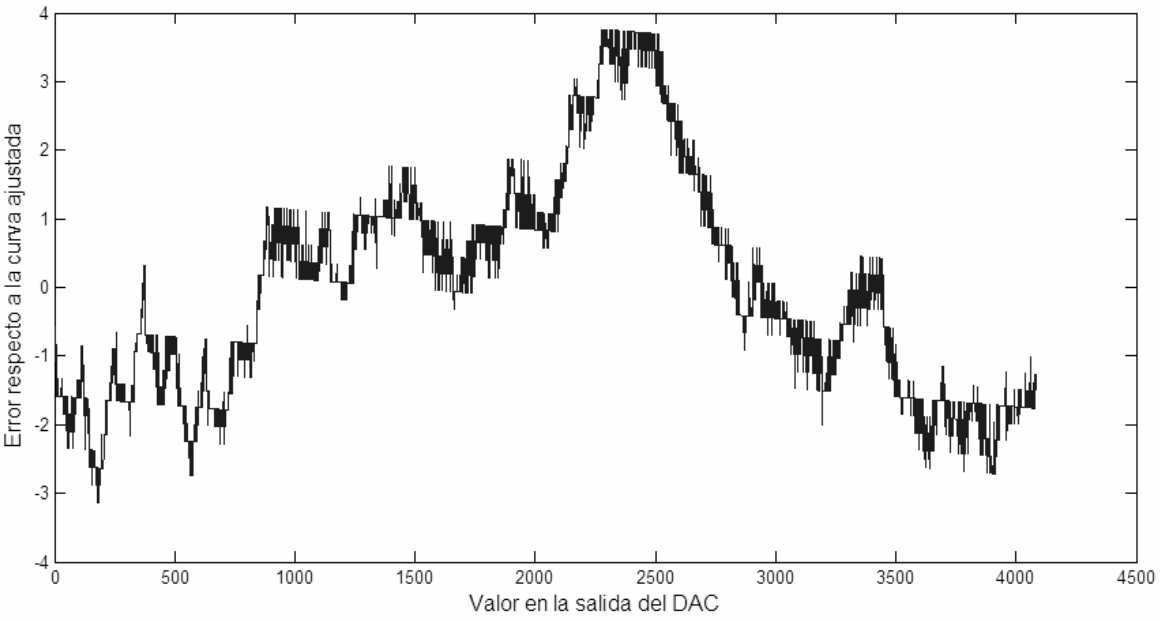
b) Diferencia entre cada punto de la prueba con la curva ajustada

Tabla. 5.2. Prueba de linealidad en escala de 5V

Escala del DAC_B: **3988** (0 - 12V)
Escala de la entrada (Pot10): **8** (0 - 12V)
Rango de los valores tomados en cuenta: **11 – 4095**
Ecuación de la recta que mejor ajusta: $y = 0.9997x - 2.5746$
Mayor error (diferencia) positiva: **3.762** [bits] en el punto 2277
Mayor error (diferencia) negativa: **-3.131** [bits] en el punto 186
Error de linealidad: **+0.092% y -0.076%**.



a) Relación de la salida con la entrada



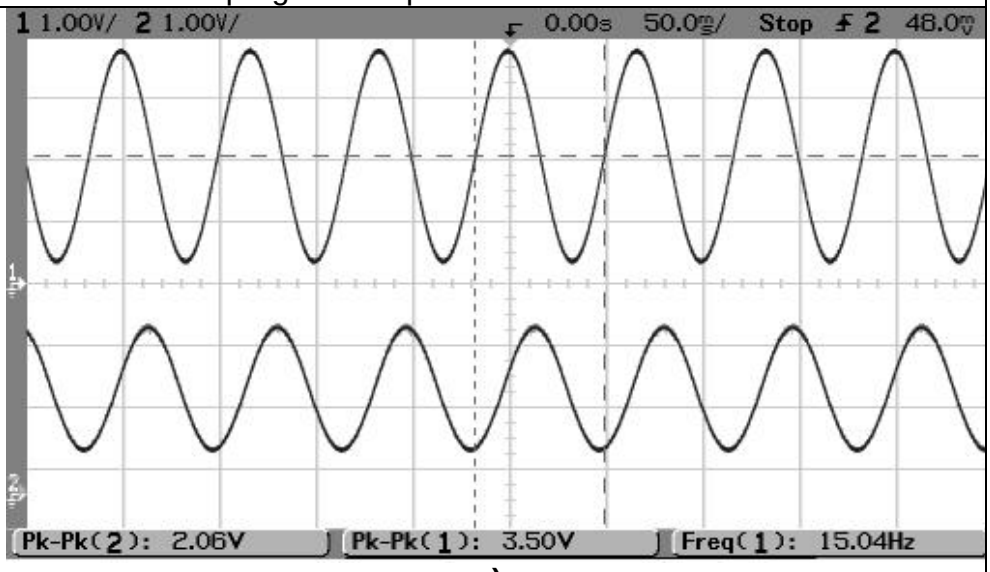
b) Diferencia entre cada punto de la prueba con la curva ajustada

Tabla. 5.3. Prueba de linealidad en escala de 12V

Cabe destacar que el promedio de los errores en ambas mediciones está en el orden de $2x10^{-12}$, lo que es técnicamente 0.

5.2.3 Filtros antialiasing

Para comprobar el desempeño de los filtros antialiasing se realizaron tres pruebas: la primera consiste en obtener la atenuación que produce el filtro ante la presencia de una señal senoidal correspondiente a la frecuencia de corte deseada con el fin de conocer el error del filtro. La segunda prueba realiza un barrido de frecuencias para conocer el desempeño general del filtro y la tercera es la obtención de la respuesta ante un escalón que describe el comportamiento en el dominio del tiempo. Se tomaron tres frecuencias de corte para las pruebas: 15Hz, 100Hz y 400Hz. El cálculo de la atenuación viene dada por la relación en dB entre el valor pico-pico de la señal con frecuencia en la zona de paso y el palor p-p de la señal con la frecuencia de corte. $Vfil_x$ se refiere al valor p-p de la señal con frecuencia x a la salida del filtro. Se considera que los filtros han sido previamente calibrados (Ver *Anexo 1* - Manual de Operación). Para todas las pruebas se usó una escala de entrada de 0-5 [V]. El Canal1 (arriba) es la señal de entrada al emulador; el Canal2 (abajo) es la señal a la entrada del ADC (señal filtrada).

Filtro programado para fc = 15 Hz	
Amplitud en la frecuencia de corte (fc = 15 Hz)	 **a)**

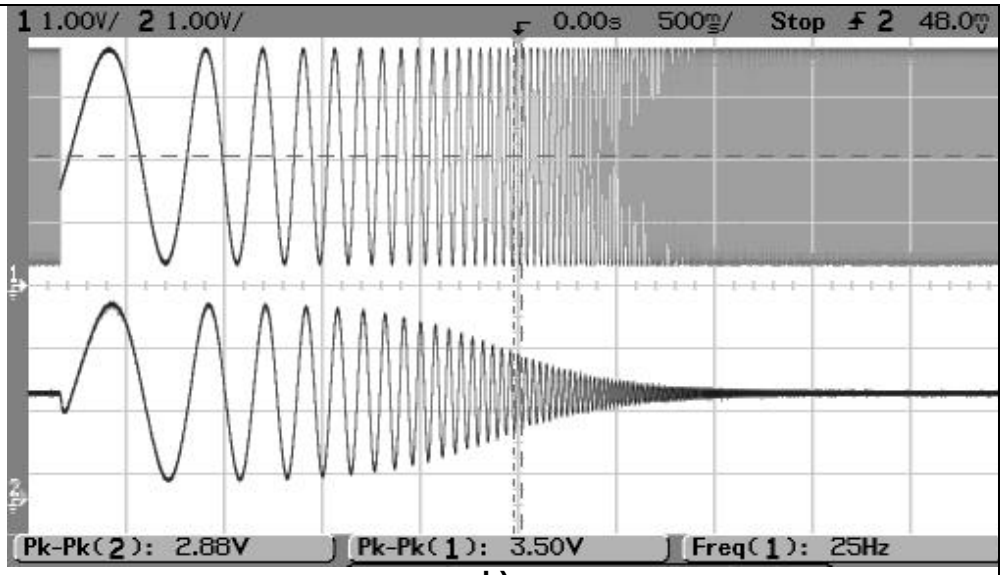
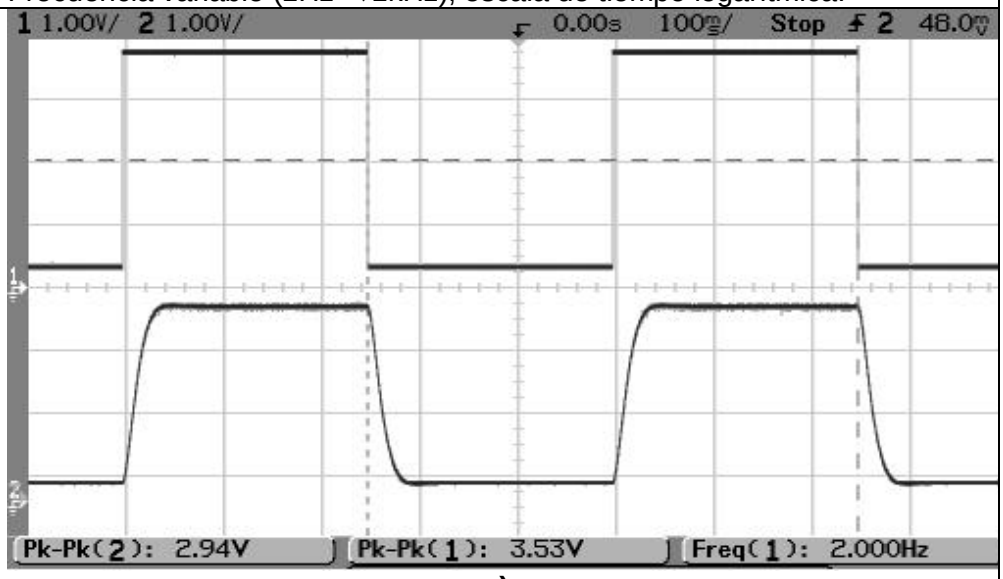
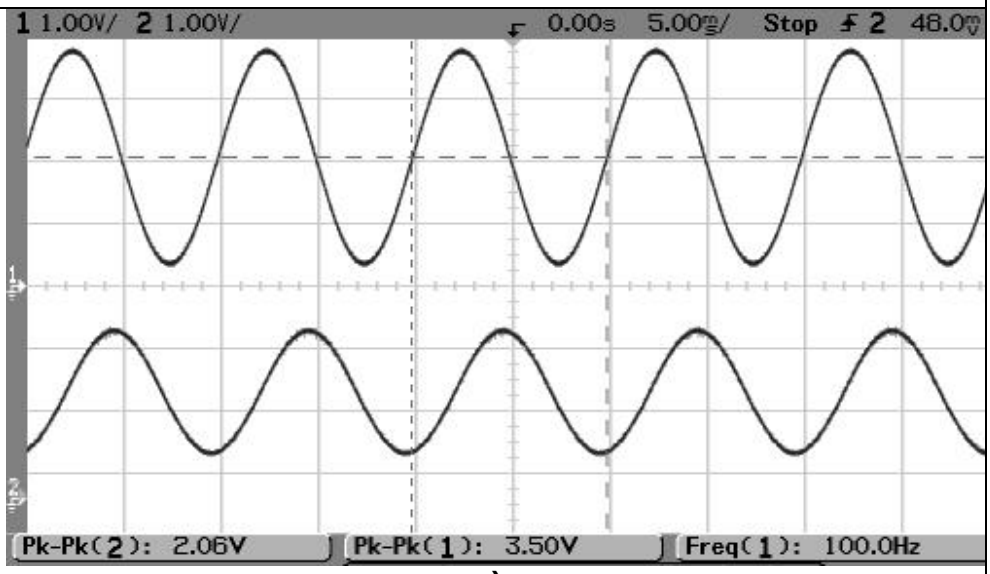
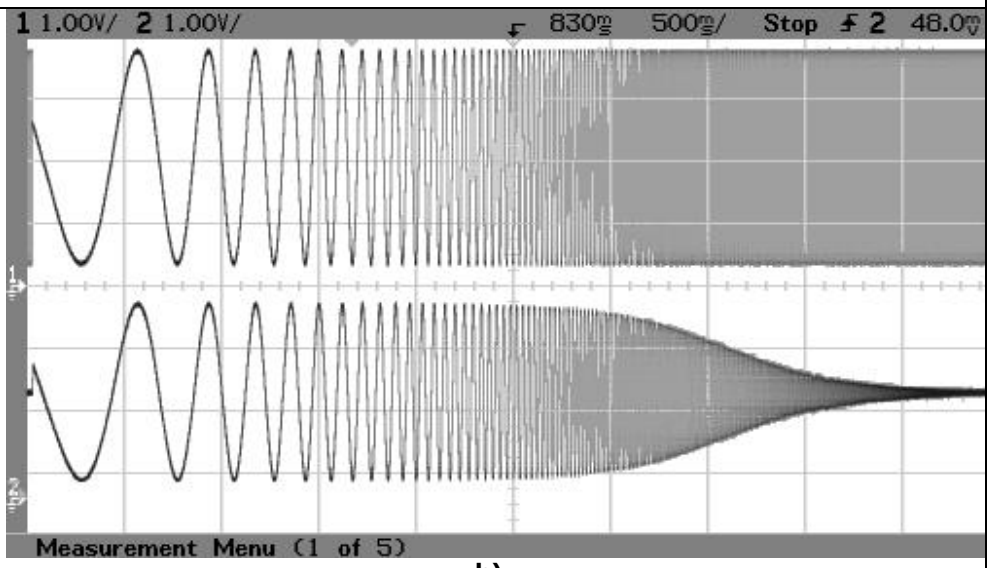
	Vfil$_1$ (f =1 Hz): 2.90 V	$20\log\left(\frac{Vfil_{15}}{Vfil_{DC}}\right) = 20\log\left(\frac{2.06}{2.90}\right) = -2.9706dB$
	Vfil$_{15}$ (f =15 Hz): 2.06 V	Error (respecto a -3dB) = -1%.
Barrido de frecuencias	 **b)**	
	Frecuencia variable (1Hz →1kHz); escala de tiempo logarítmica.	
Respuesta ante el escalón	 **c)**	
	Frecuencia de exitación: 2Hz. No hay sobreimpulso apreciable.	

Tabla. 5.4. Resultados para filtro antialiasing con fc = 15Hz

<table>
<tr><td colspan="3">Filtro programado para fc = 100 Hz</td></tr>
<tr><td rowspan="3">Amplitud en la frecuencia de corte (fc =100 Hz)</td><td colspan="2"> 

a)</td></tr>
<tr><td colspan="2"><table>
<tr><td>Vfil$_1$ (f =1 Hz): 2.90 V</td><td>$20\log\left(\frac{Vfil_{100}}{Vfil_{DC}}\right)=20\log\left(\frac{2.07}{2.90}\right)=-2.9286dB$</td></tr>
<tr><td>Vfil$_{100}$ (f =100 Hz): 2.07 V</td><td>Error (respecto a -3dB) = -2.4%.</td></tr>
</table></td></tr>
<tr><td>Barrido de frecuencias</td><td colspan="2"> 

b)</td></tr>
<tr><td colspan="2">Frecuencia variable (1Hz →1kHz); escala de tiempo logarítmica.</td></tr>
</table>

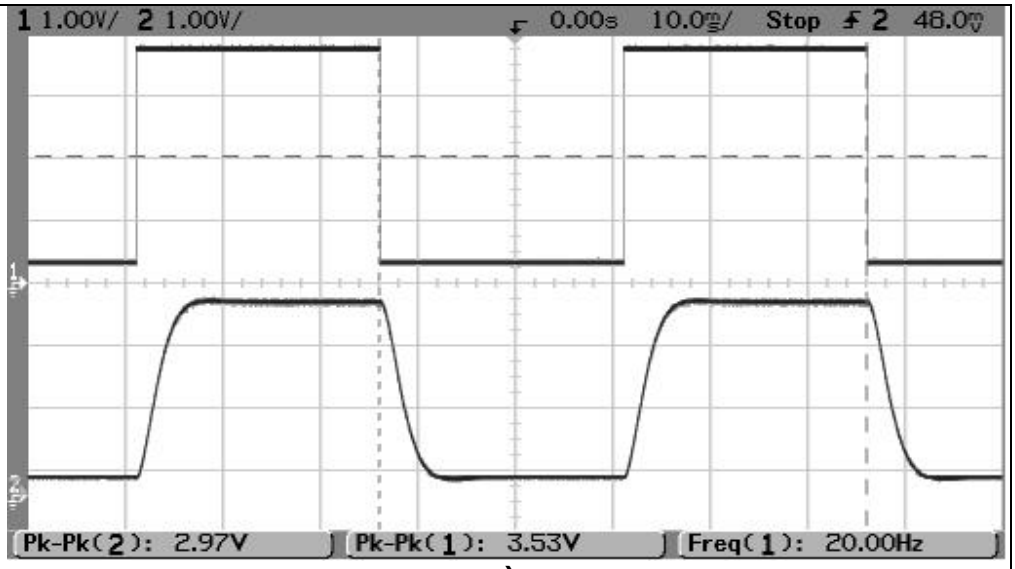
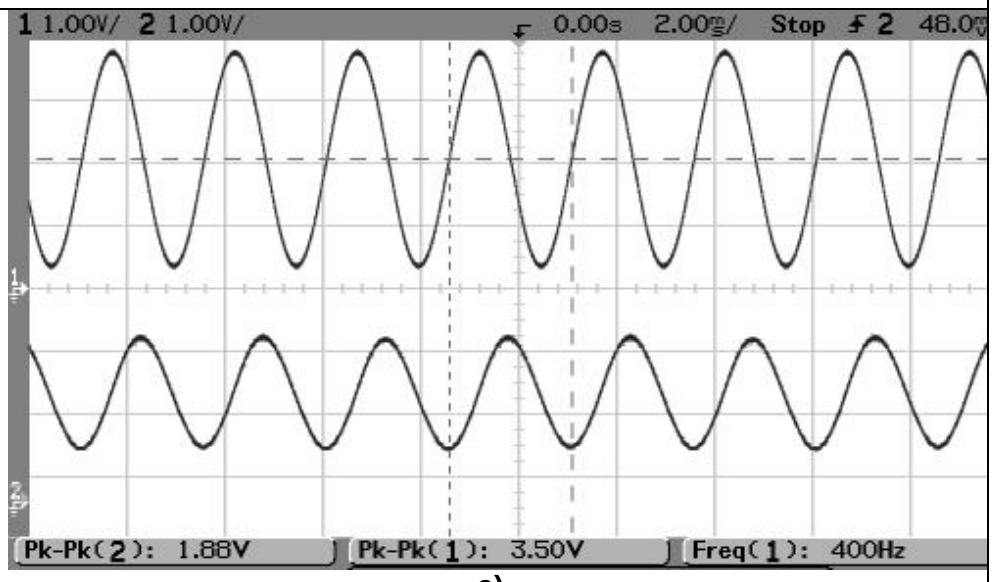
Respuesta ante el escalón	 **c)**
	Frecuencia de exitación: 20Hz. No hay sobreimpulso apreciable.

Tabla. 5.5. Resultados para filtro antialiasing con fc = 100Hz

Filtro programado para fc = 400 Hz		
Amplitud en la frecuencia de corte (fc =400 Hz)	 **a)**	
	$Vfil_1$ (f =1 Hz): 2.88 V	$20\log\left(\frac{Vfil_{400}}{Vfil_{DC}}\right) = 20\log\left(\frac{1.91}{2.88}\right) = -3.567dB$
	$Vfil_{400}$ (f =400 Hz): 1.91V	Error (respecto a -3dB) = -19%.

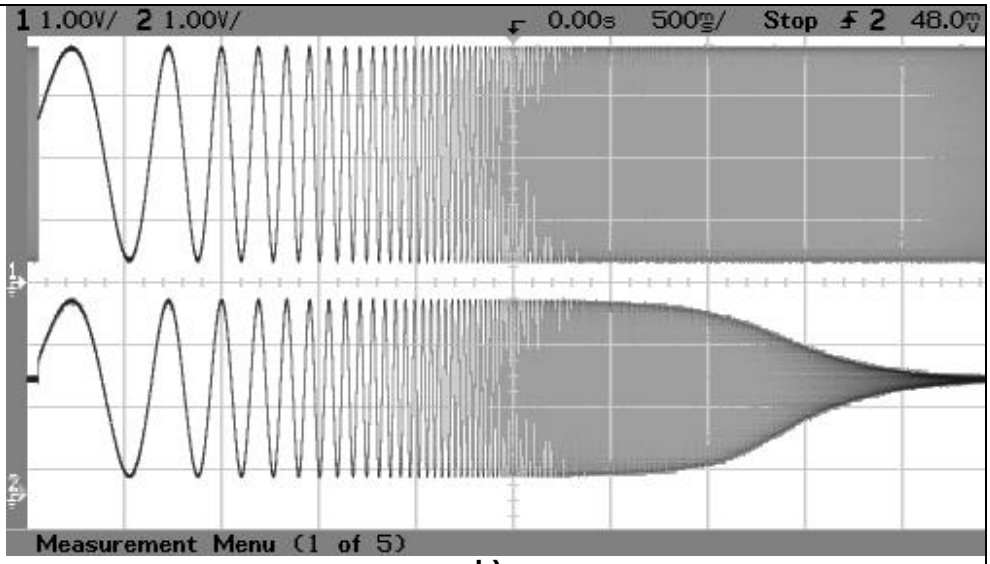
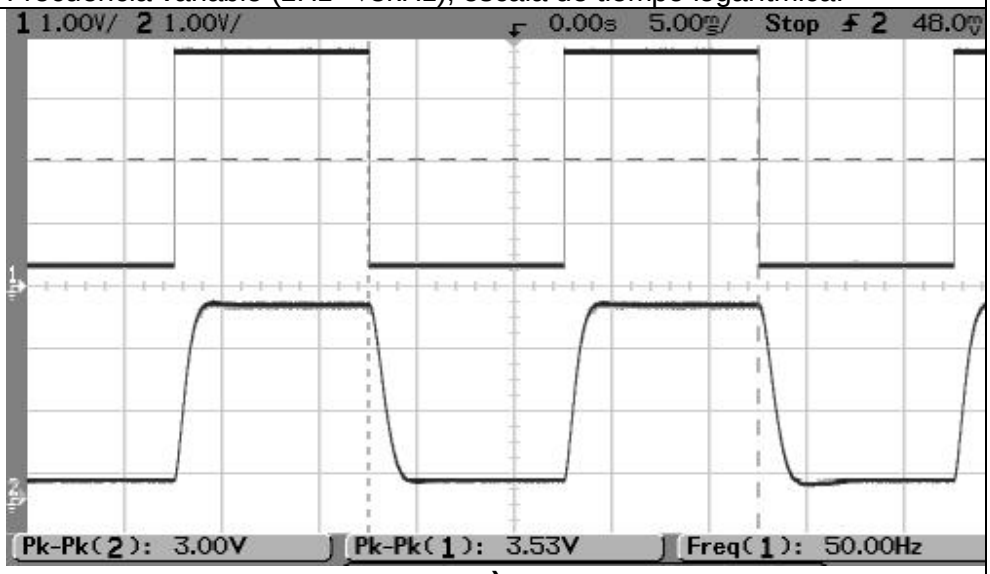
Barrido de frecuencias	 **b)**
	Frecuencia variable (1Hz →3kHz); escala de tiempo logarítmica.
Respuesta ante el escalón	 **c)**
	Frecuencia de exitación: 50Hz. El sobreimpulso observado no es particularmente relevante.

Tabla. 5.6. Resultados para filtro antialiasing con fc = 400Hz

5.3 PRUEBAS COMO EMULADOR

5.3.1 Descripción de las pruebas

Las pruebas del emulador consisten en comparar el comportamiento en laboratorio (inyectando señales) del emulador y las simulaciones en MATLAB. Al inicio de cada prueba se especifica la ecuación de transferencia y la configuración de Entrada/Salida del emulador. Para cada caso se realizan tres pruebas:

1. Desempeño sólo de la función de transferencia.
2. La función de transferencia con retardo de transporte.
3. La función de transferencia con una no-linealidad estática. En este caso se especifica la no linealidad por medio de su tabla de búsqueda respectiva.

Por medio de un generador de señales se inyecta una función escalón con el fin de observar la respuesta que genera el emulador. La conexión de los equipos para estas pruebas y el software utilizado se muestran en la Figura 5.2.

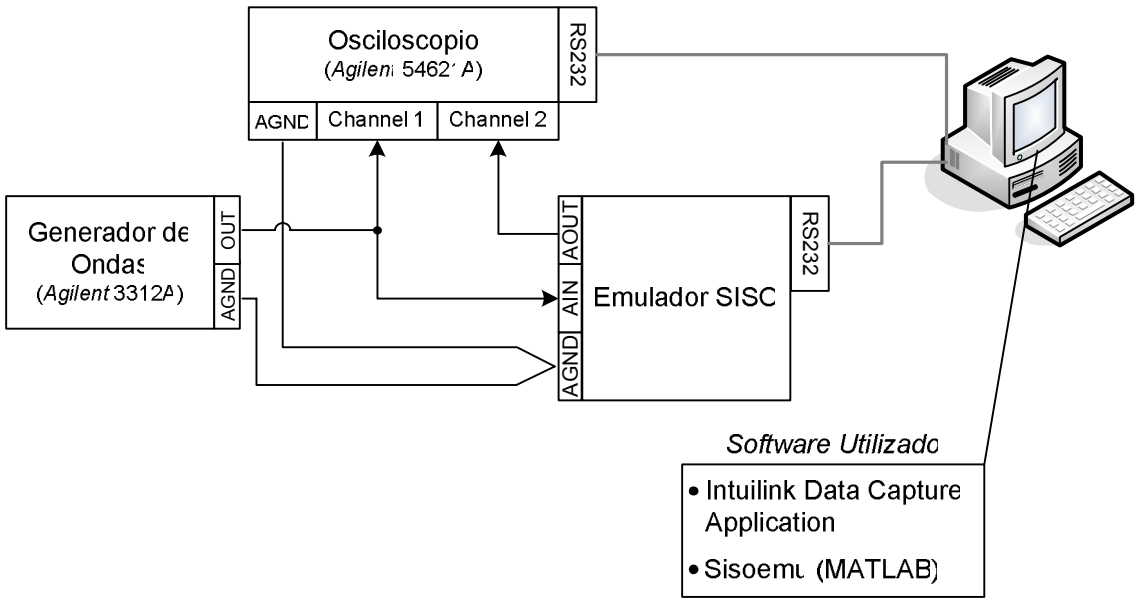


Figura. 5.2. Conexión de los equipos para las pruebas de emulador

5.3.2 Planta de primer orden

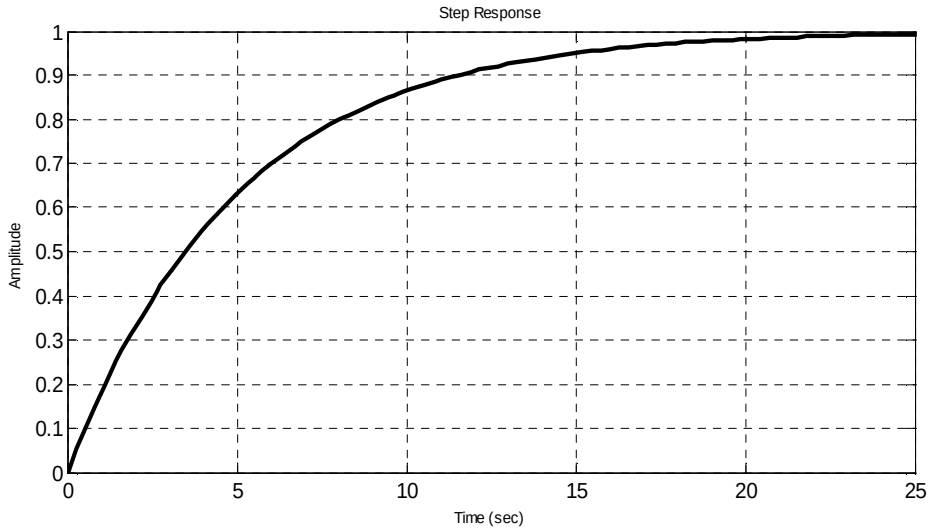
Ecuación de transferencia

$$G(s) = \frac{1}{5s+1} \quad (5\text{-}1)$$

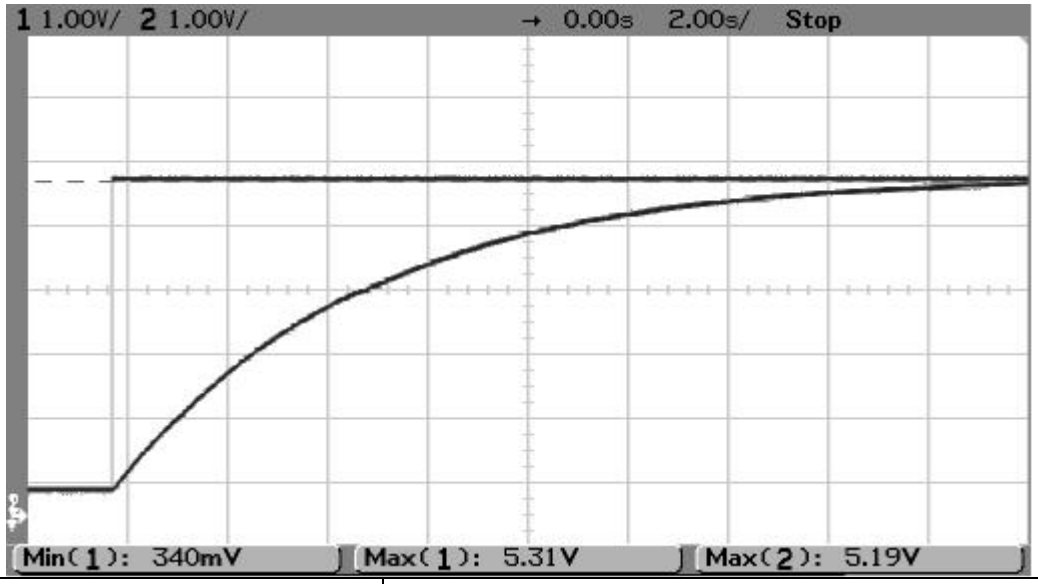
Configuración E/S emulador

Entrada:	0 – 6 V fc: 20Hz	Salida:	0 – 6 V fc: 20Hz
Tiempo de ciclo:		0.01 [s] (100 Hz)	

a) - Comportamiento teórico (simulado)



b) - Comportamiento experimental (amplitud escalón: 5[V])

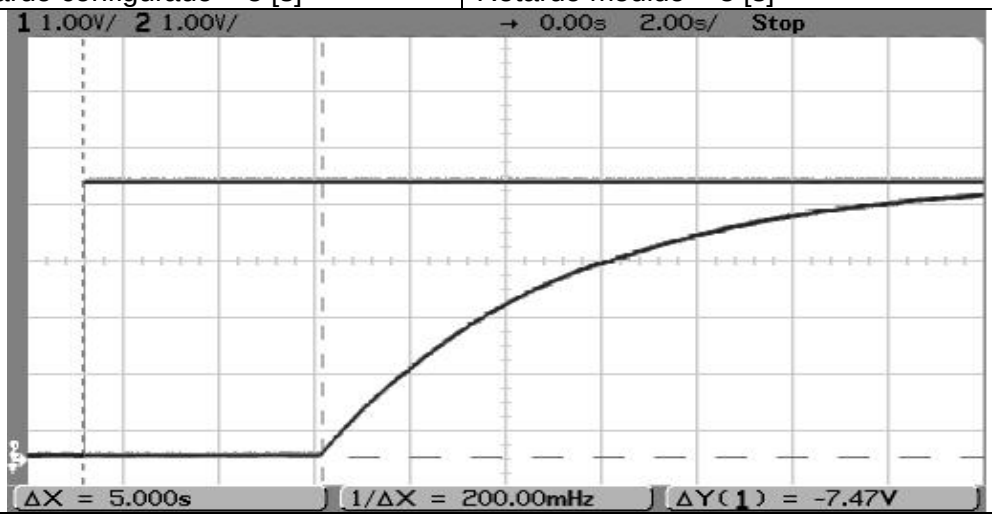


τ (medido) = 4.99 [s]	τ = 5 [s]

Tabla. 5.7. Comparación teórico-experimental planta primer orden

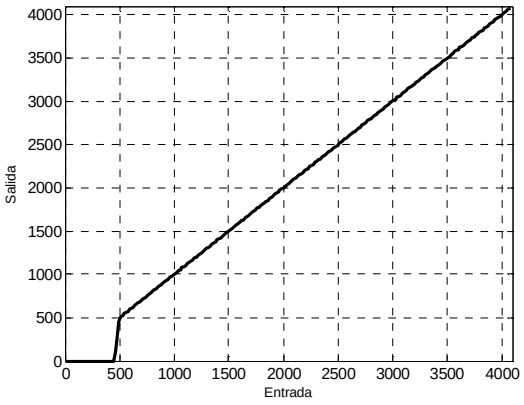
Función de transferencia con retardo	
Retardo configurado = 5 [s]	Retardo medido = 5 [s]

a)

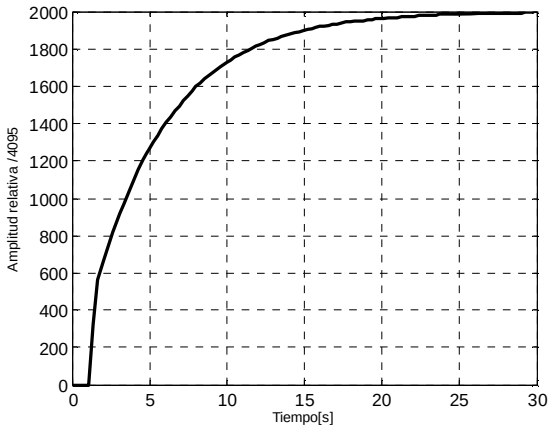


No linealidad estática

b) - Tabla de búsqueda



c) - Respuesta teórica (simulada)



d) - Comportamiento experimental (amplitud escalón: 5[V])

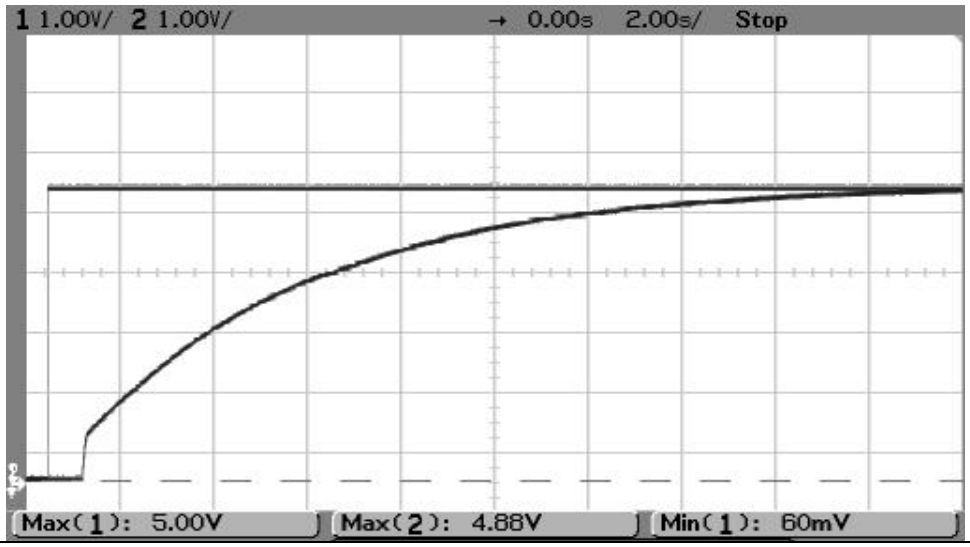


Tabla. 5.8. Retardo y no linealidad en planta de primer orden

5.3.3 Planta de segundo orden

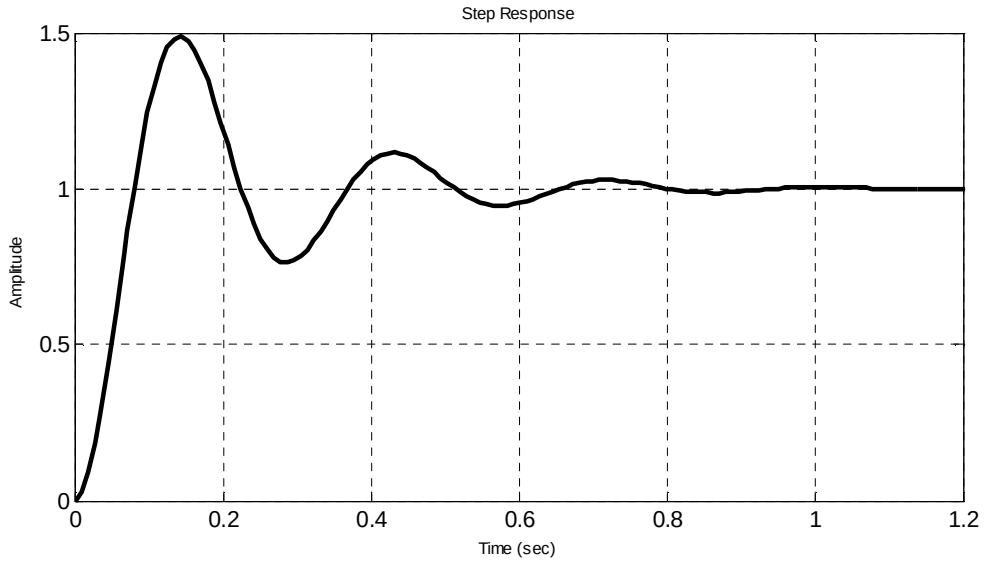
Ecuación de transferencia

$$G(s) = \frac{1+500}{s^2+10s+500}$$

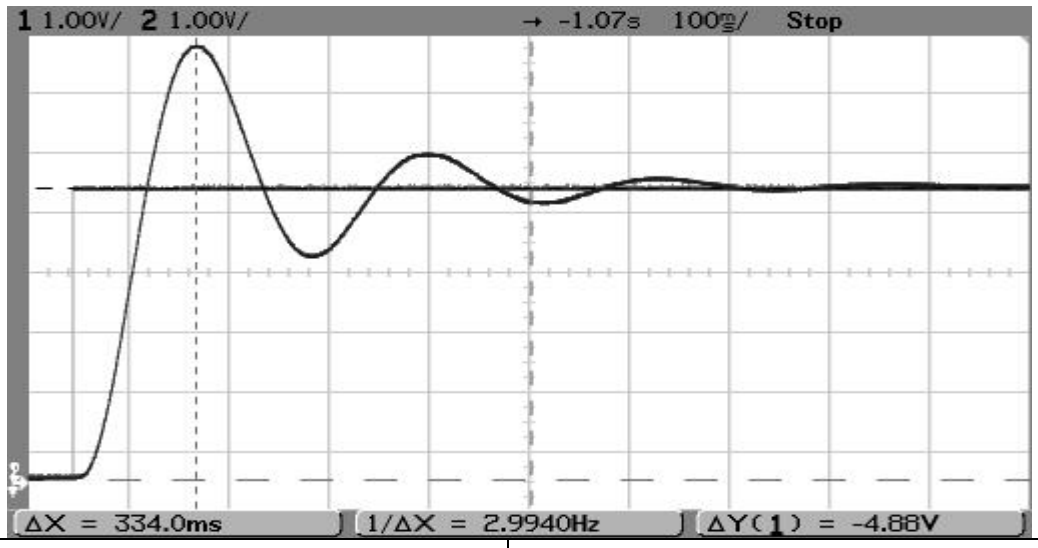
Configuración E/S emulador

Entrada:	0 – 10 V fc: 50Hz	Salida:	0 – 10 V fc: 50Hz
Tiempo de ciclo:		1 [ms] (1000 Hz)	

a) - Comportamiento teórico (simulado)



b) - Comportamiento experimental (amplitud escalón: 5[V])



amplitud vs tiempo - teórico	amplitud vs tiempo - experimental
Tiempo de subida: 0.0551 [s] Amplitud pico : 1.49 Ganancia en DC: 1 Sobreimpulso(%): 48.7 %	Tiempo de subida: 0.056 [s] Amplitud pico : 7.26 [V] Ganancia en DC: 1 [V/V] Sobreimpulso(%): 48.72 %

Tabla. 5.9. Comparación teórico-experimental planta segundo orden

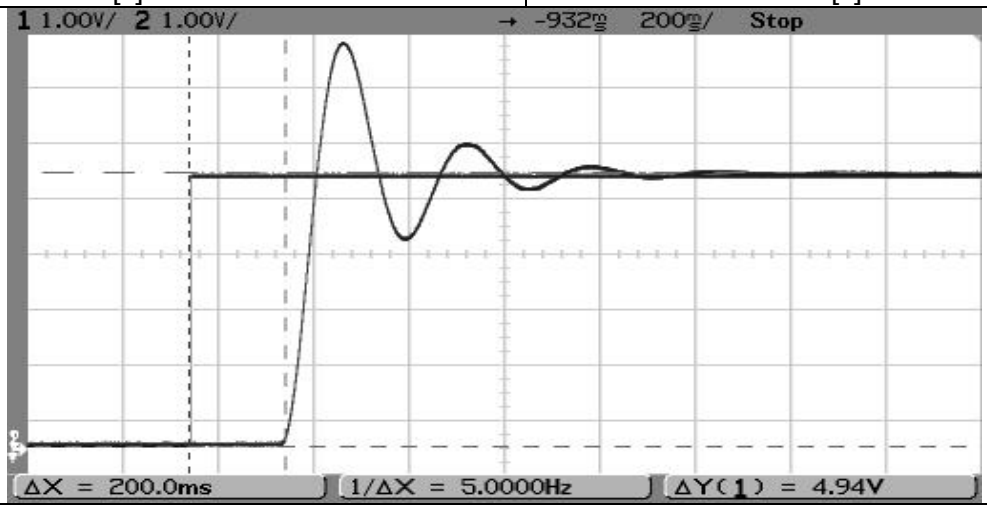
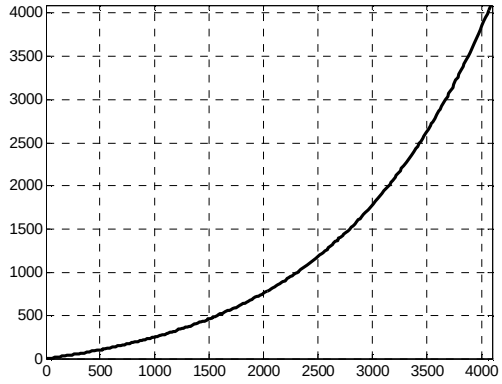
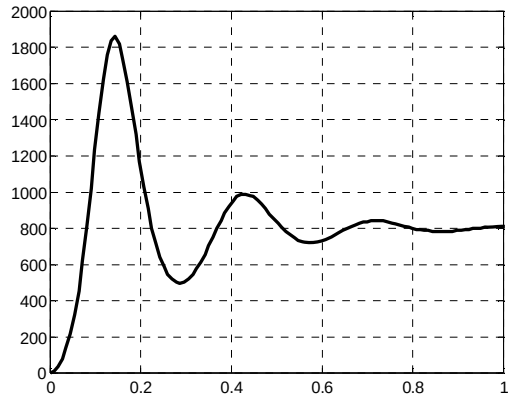
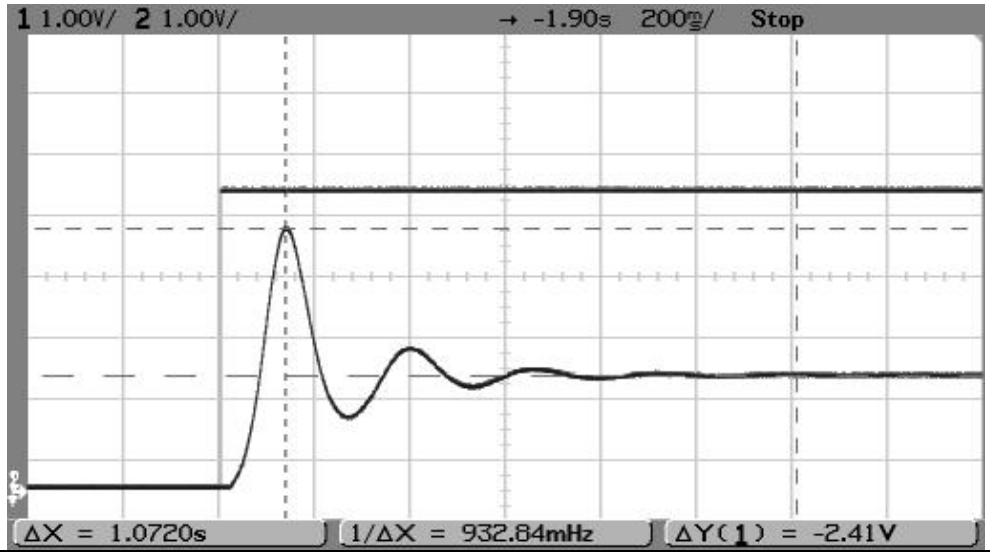
<table>
<tr><td colspan="2">Función de transferencia con retardo</td></tr>
<tr><td>Retardo = 0.2 [s]</td><td>Retardo medido = 0.2 [s]</td></tr>
<tr><td colspan="2">a)
 
</td></tr>
<tr><td colspan="2">No linealidad estática</td></tr>
<tr><td>b) - Tabla de búsqueda
 
</td><td>c) - Respuesta teórica (simulada)
 
</td></tr>
<tr><td colspan="2">d) - Comportamiento experimental (amplitud escalón: 5[V])
 
</td></tr>
</table>

Tabla. 5.10. Retardo y no linealidad en planta de segundo orden

5.3.4 Planta con doble integrador

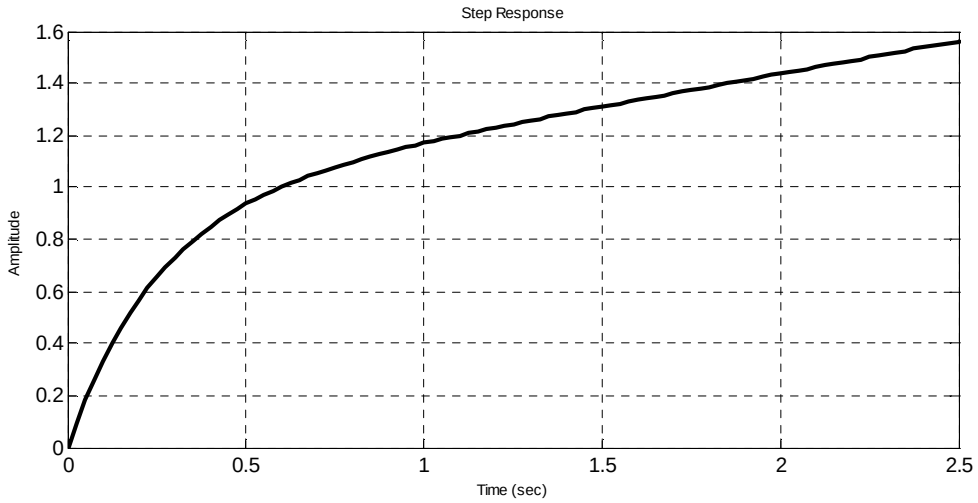
Ecuación de transferencia

$$G(s) = \frac{4s+1}{s(s+4)}$$

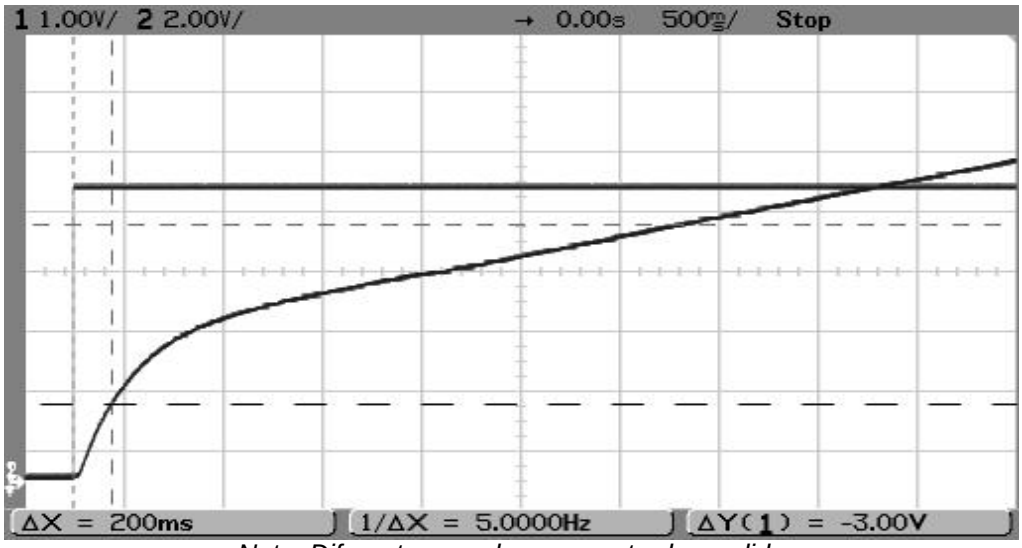
Configuración E/S emulador

Entrada:	0 – 12 V fc: 30Hz	Salida:	0 – 12 V fc: 15Hz
Tiempo de ciclo:		5 [ms] (200 Hz)	

a) - Comportamiento teórico (simulado)



b) - Comportamiento experimental (amplitud escalón: 4.80 [V])



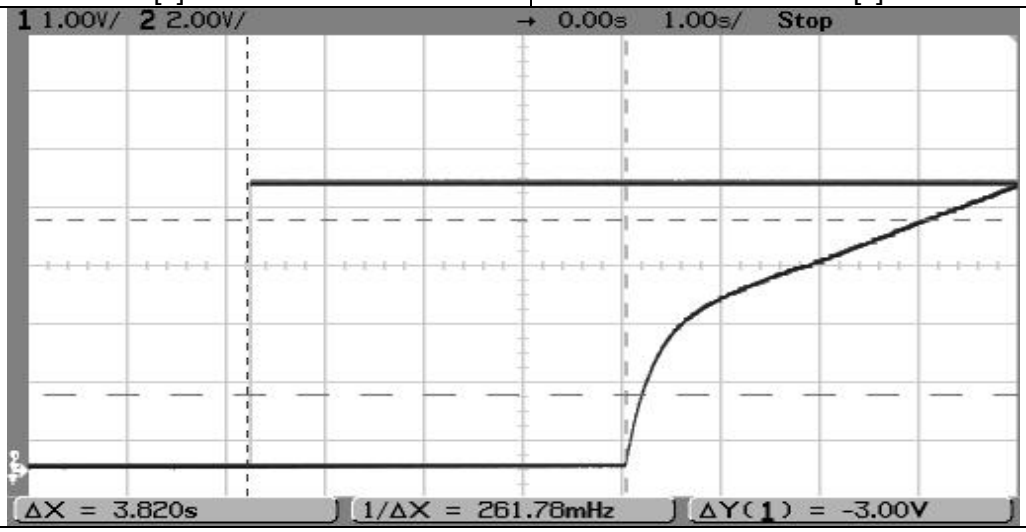
Nota: Diferentes escalas para entrada y salida

amplitud vs tiempo - teórico	amplitud vs tiempo - experimental
Amplitud a 0.2 s: 0.566	Amplitud a 0.2 s: 2.736 (Norm: 0.570)
Amplitud a 0.5 s: 0.936	Amplitud a 0.5 s: 4.502 (Norm: 0.938)
Amplitud a 1.0 s: 1.170	Amplitud a 1.0 s: 5.616 (Norm: 1.170)
Amplitud a 2.0 s: 1.440	Amplitud a 2.0 s: 6.960 (Norm: 1.450)

Tabla. 5.11. Comparación teórico-experimental planta con doble integrador

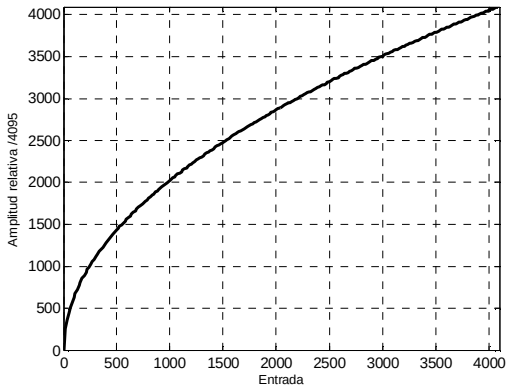
Función de transferencia con retardo	
Retardo = 3.83 [s]	Retardo medido = 3.82 [s]

a)

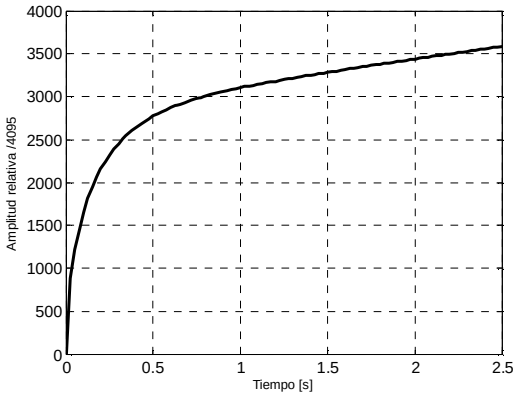


No linealidad estática

b) - Tabla de búsqueda



c) - Respuesta teórica (simulada)



d) - Comportamiento experimental (amplitud escalón: 4.8[V])

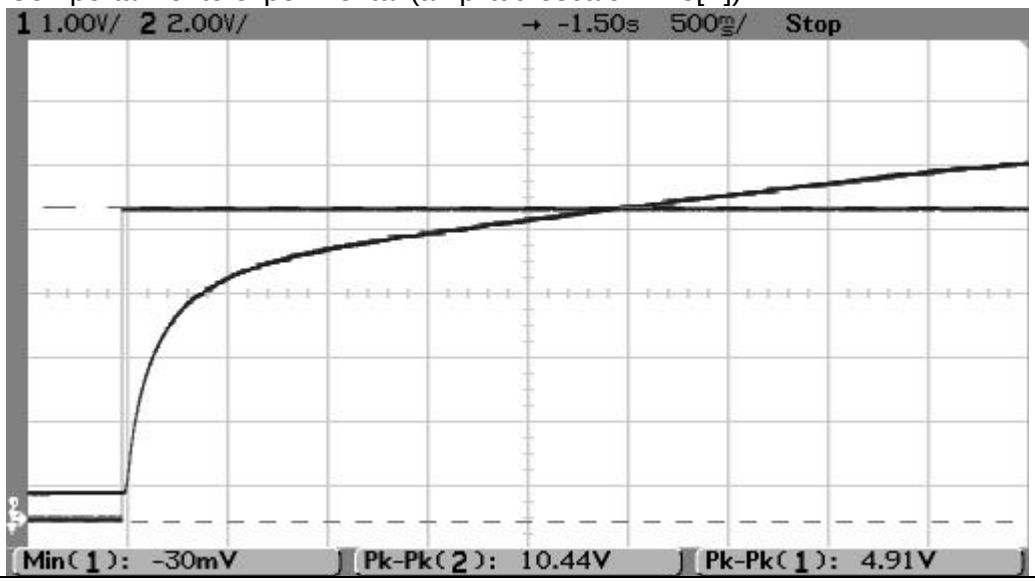


Tabla. 5.12. Retardo y no linealidad en planta con doble integrador

5.3.5 Planta de cuarto orden

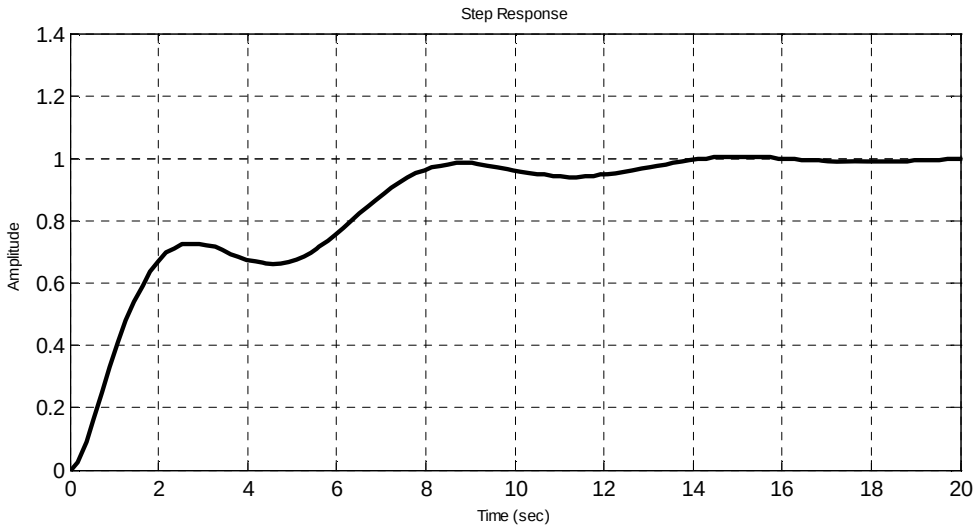
Ecuación de transferencia

$$G(s) = \frac{2s^2 + s + 1}{s^4 + 4s^3 + 3.5s^2 + 4s + 1}$$

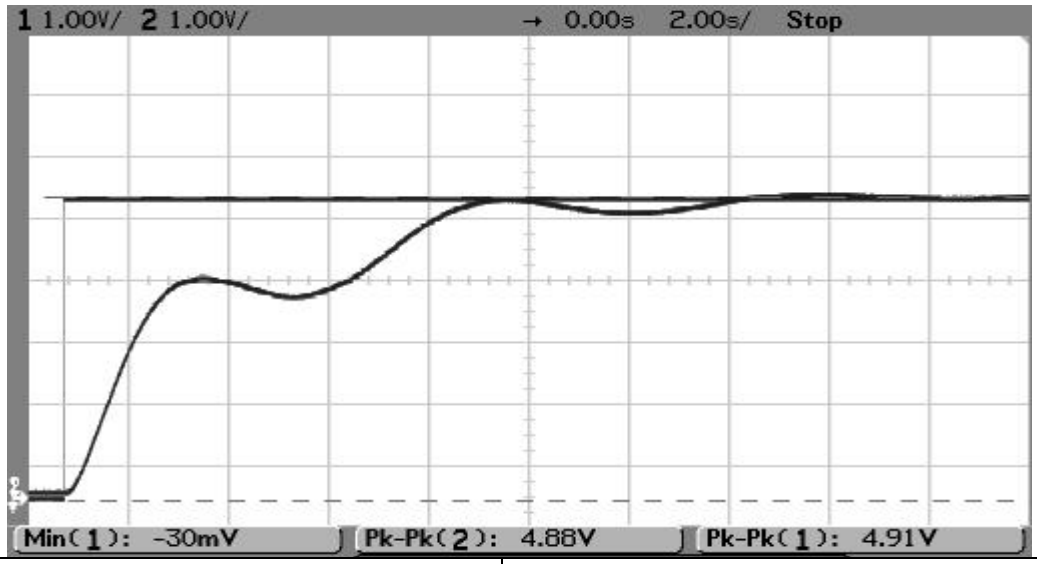
Configuración E/S emulador

Entrada:	0 – 6 V fc: 30Hz	Salida:	0 – 6 V fc: 15Hz
Tiempo de ciclo:		5 [ms] (200 Hz)	

a) - Comportamiento teórico (simulado)



b) - Comportamiento experimental (amplitud escalón: 4.88 [V])

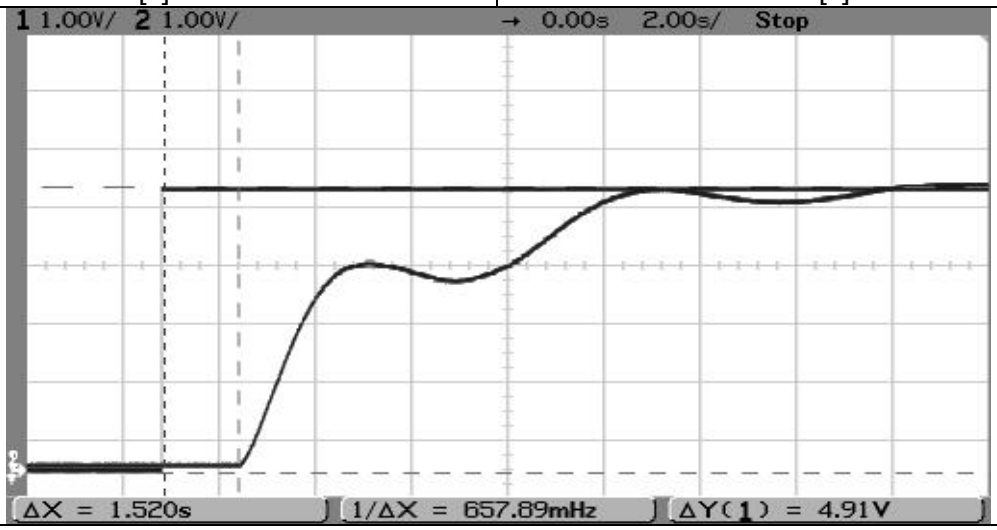


amplitud vs tiempo - teórico	amplitud vs tiempo - experimental
Amplitud a 2.7 s: 0.728 Amplitud a 4.5 s: 0.662 Amplitud a 8.7 s: 0.984 Amplitud a 26. s: 1	Amplitud a 2.7 s: 3.56 (Norm: 0.730) Amplitud a 4.5 s: 3.24 (Norm: 0.664) Amplitud a 8.7 s: 4.81 (Norm: 0.986) Amplitud a 26. s: 4.93 (Norm: 1.010)

Tabla. 5.13. Comparación teórico-experimental planta curto orden

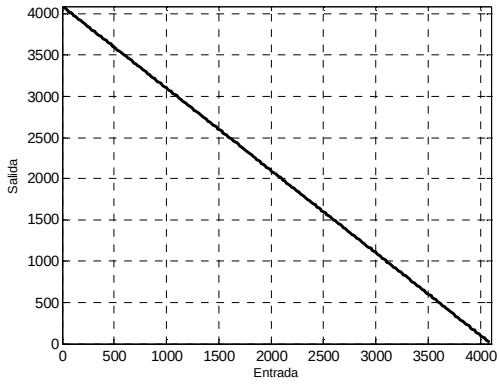
Función de transferencia con retardo	
Retardo = 1.52 [s]	Retardo medido = 1.52 [s]

a)

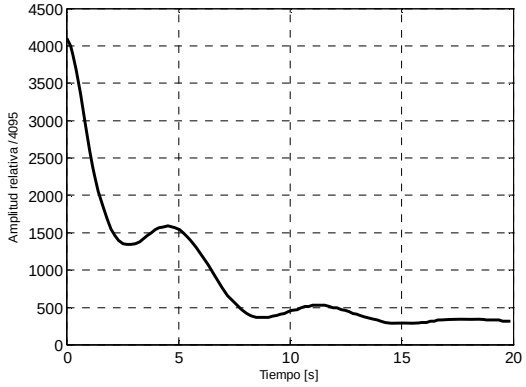


No linealidad estática

b) - Tabla de búsqueda



c) - Respuesta teórica (simulada)



d) - Comportamiento experimental (amplitud escalón: 4.88[V])

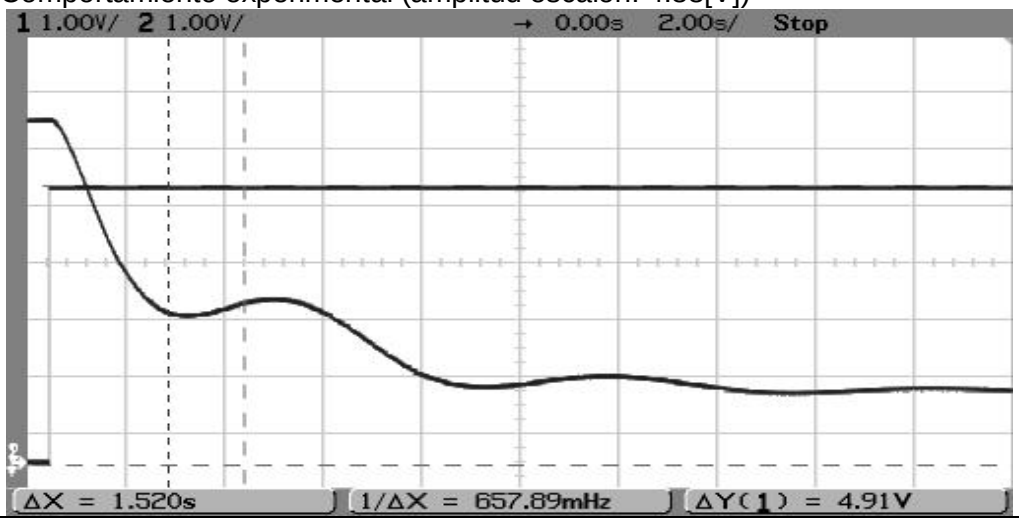


Tabla. 5.14. Retardo y no linealidad en planta de cuarto orden

5.4 RESULTADOS

5.4.1 Ruido

Los histogramas resultantes de la recolección de datos muestran un bajo contenido de ruido en la adquisición de datos del emulador. En el gráfico a) de la Tabla 5.1 se muestra que el 89.84% de las muestras están concentradas en una sola medición, considerando que la señal de entrada es muy baja (13mV). Para el gráfico b) se muestra el resultado de la inyección de un señal intermedia entre 2037 y 2038. El gráfico c) se produce cuando la ganancia a la entrada es menor y muestra una concentración de 95.18% en un solo valor. Por tanto el peor caso de ruido es ±1 bit en la adquisición (±1mV), sin embargo las concentraciones puntuales demuestran que la adquisición es bastante precisa

5.4.2 Linealidad

Las pruebas de linealidad muestran que en todo el rango del sistema (0 - 4.096 V) la linealidad se altera aproximadamente ±5mV (corresponde a ±5 bits) en el peor caso. Esto indica que el error a full escala es de +0.121% y -0.106%. Este error incluye todo el sistema analógico de entrada y salida por lo que se considera un buen desempeño de linealidad.

5.4.3 Filtros antialiasing

Los filtros antialiasing muestran en todos los casos (Tablas 5.4, 5.5 y 5.6 figura b)) un comportamiento pasabajos. La exactitud que se logra tener en la frecuencia de corte varía dependiendo del valor de la frecuencia de corte. El error se calcula usando (5-2) y se tiene que para frecuencias bajas (p.e. 15Hz) el error es bajo, -1%, mientras que en frecuencias de 100 Hz y 400Hz se tienen errores de -2.4% y -19% respectivamente. La razón es que la variación de resistencia en el potenciómetro es lineal mientras que los valores de resistencia para el filtro

varían inversamente a la frecuencia de corte (3-7 y 3-8), permitiendo tener mayor control en frecuencias bajas y menor control en frecuencias altas.

$$Ganancia\,\mathrm{Re}\,al = 20\log\left(\frac{Vfil_{fc}}{Vfil_{DC}}\right) \qquad (5\text{-}1)$$

$$e = \frac{-Ganancia\,\mathrm{Re}\,al + 3db}{3dB} \qquad (5\text{-}2)$$

Un punto extremadamente positivo de este filtro es la excelente respuesta al escalón. En sistemas de control es importante el comportamiento en el dominio del tiempo de los sistemas por lo que desde un principio se buscó una respuesta sin oscilaciones en el filtro antialiasing. Para los tres casos (Tablas 5.4, 5.5 y 5.6, Figura c)) se muestra un escalón sin sobreimpulso.

5.4.4 Emulación de plantas

Es preciso anotar que resultó complicado medir los errores que podía tener el sistema emulado con las simulaciones en MATLAB. Por medio de las gráficas se puede apreciar que la respuesta en todos los casos es visualmente idéntica. Esto concuerda con el hecho de que los cálculos que realiza internamente el DSC son los mismos que usa MATLAB para encontrar las soluciones en las simulaciones.

Si sumamos a la buena linealidad del circuito analógico se garantiza una excelente fidelidad en la reproducción del modelo deseado. Las pruebas de retardo indican un buen desempeño en la precisión del reloj del DSC y los experimentos con la función de no linealidad muestran un buen desempeño del algoritmo de tabla de búsqueda e interpolación.

CAPÍTULO 6

CONCLUSIONES Y RECOMENDACIONES

6.1 CONCLUSIONES

6.1.1 Conclusiones sobre simulación HIL

- El presente proyecto logró implementar con éxito un programa embebido en un controlador digital de señales dsPIC30F4013 que calcula funciones de transferencia hasta de 4^{to} orden en punto flotante de 64 bits, retardo de señal y tabla de búsqueda hasta una frecuencia de ciclo de 2kHz. Adicionalmente es posible inyectar perturbaciones por medio de un potenciómetro mecánico.

- La interfase de usuario (GUI) permite configurar lo parámetros de la planta de manera sencilla e intuitiva, integrando la generación de parámetros para el programa del DSC y la gestión de las comunicaciones necesarias para enlazarse con el mismo. Posee las herramientas que facilitan la calibración de los parámetros analógicos del circuito emulador.

- El método más sencillo de simular ecuaciones diferenciales lineales ordinarias es usando una ecuación de diferencias equivalente a la ecuación por resolverse; de esta manera se obtienen soluciones exactas en los puntos de cálculo.

- Al realizar los cálculos de la ecuación de diferencias hay que tener en cuenta la precisión con que se trabaja en cada operación. Debido a que la aritmética en la resolución involucra valores anteriores computados, los

errores en los cálculos se acumulan cada ciclo de iteración y pueden llevar a grandes desviaciones en los resultados; incluso llevando a la inestabilidad del sistema. Este criterio pesa más para ecuaciones de 3er ó mayor orden. Por esta razón se usa punto flotante de 64 bits.

- El uso del DSC es indispensable para alcanzar la precisión en los cómputos que se desea gracias a su velocidad de funcionamiento (30MIPS) y la arquitectura de DSP. Este desempeño computacional no es posible con un microcontrolador de 8 bits.

- El uso de la comunicación serial por RS232 resulta ser la mejor opción para conectar dos equipos debido a la difusión, disponibilidad de periféricos en los microcontroladores y drivers para el manejo de los mismos desde una PC.

- Debido a que las computadoras portátiles actuales no incluyen un puerto RS232 se incluyó compatibilidad para trabajar con adaptadores USB a RS232.

6.2.2 Conclusiones sobre el Hardware

- El circuito analógico del emulador SISO posee una linealidad de $\pm$0.12% en toda su escala, logra configurar el rango de entrada y salida entre 0-5V y 0-12V sin perder su característica de 12 bits y posee filtros antialiasing de segundo orden variables por software a la entrada y salida del ADC y DAC respectivamente. El contenido de ruido es de $\pm$1mVpp.

- El usar potenciómetros digitales permite tener mayor control en la calibración de los parámetros de las señales analógicas. La utilización de estos elementos puede eliminar casi completamente el uso de potenciómetros mecánicos para la calibración y regulación de parámetros analógicos.

- No fue necesario usar resistencias de precisión ya que es posible compensar estos errores al utilizar potenciómetros digitales.

- Los filtros antialiasing implementados con potenciómetros digitales muestran baja distorsión, bajo ruido y buena respuesta al escalón, mostrando un buen desempeño respecto al teórico definido en la Tabla 3.9.

- El uso de una referencia de voltaje garantiza la estabilidad y exactitud de las mediciones y valores de salida del emulador ante cambios de temperatura y envejecimiento.

- Un correcto uso de planos a tierra permite disminuir notablemente el ruido en el sistema. A este criterio se suma la correcta ubicación de los elementos.

6.1.3 Conclusiones sobre el Software

- A pesar de que el compilador de ANSI C Microchip C30 es muy completo, es necesario conocer bien el lenguaje ensamblador del DSC para realizar ciertas operaciones y aún más si se desea implementar rutinas que optimicen el tiempo de ejecución.

- Se estructuró de mejor manera el código y se simplificó el desarrollo del programa embebido gracias al uso del sistema operativo en tiempo real CMX-Scheduler, cuyo uso es sencillo y el esfuerzo en su aprendizaje se compensa por las ventajas que trae en la programación. Hay que tener especial cuidado en la asignación de memoria a las distintas tareas.

- La inclusión del firmware bootloader en el DSC simplifica y agilita el proceso de descarga de código al dispositivo, evitando el maltrato del empaque debido a la extracción e inserción constante desde el zócalo.

- El uso de MATLAB resulta complicado a la hora de realizar aplicaciones de interfaz de usuario de mediana o alta complejidad. Sin embargo las herramientas matemáticas que incluye permiten simplificar el desarrollo matemático de la aplicación.

- La interfaz gráfica o ambiente de configuración del emulador SISO juega un papel importante en la robustez del sistema ya que debe verificar estrictamente los valores que se descargan a la memoria EEPROM del hardware.

- Las funciones que facilita MATLAB para el manejo del puerto serial poseen cierta inestabilidad por lo que se producen fallas inesperadas al momento de usarlas.

6.2 RECOMENDACIONES

- Queda por explorar la implementación de un Embedded Target (Objetivo embebido) usando el Real-Time Workshop de MATLAB para los procesadores de la familia dsPIC30F y dsPIC33F de Microchip. De esta manera sería posible generar el código apropiado para el DSC a partir de un modelo en Simulink.

- El hardware implementado para el emulador SISO puede ser usado tranquilamente para varias aplicaciones de control. La incorporación de un controlador digital de señales da la opción de crear controladores avanzados de tipo adaptativo, difuso, neuronal, entre otros ó permite aumentar la funcionalidad de controladores actuales. También pone a disposición la infraestructura para compatibilizar con señales industriales analógicas y digitales.

- Sería importante incorporar en el plan de estudios de la Facultad de Electrónica el uso de controladores digitales de señales ya que sabiendo manejarlos es posible realizar proyectos más complejos y completos. Hay que notar que el manejo de estos dispositivos está estrechamente ligado al conocimiento del procesamiento digital se señales y sería un buen punto de inicio para demostrar el aspecto práctico de este campo de la electrónica.

- En este momento existen muchos sistemas operativos para procesadores de 8 y 16 bits. Los conceptos que involucran permiten crear aplicaciones embebidas más robustas y más sencillas de programar. Sería recomendable estudiar más detenidamente los conceptos y requerimientos para la implementación de un sistema operativo para microcontroladores.

- Cada vez es más difícil conseguir componentes nuevos que vengan en empaques tipo DIP. De hecho el DSC que se usó para este proyecto es el último que el fabricante lanzó con este empaque y las nuevas generaciones de estos dispositivos vienen en empaques de montaje superficial. Esto obliga a que la ESPE busque nuevas formas de realizar los prototipos a parte del clásico protoboard para que la enseñanza pueda moverse hacia los nuevos dispositivos comerciales.

- Los nuevos diseños de fuentes de poder deberían basarse en el uso de reguladores conmutados para lograr dispositivos más eficientes. La mayor complicación para realizar estos reguladores es la dificultad para conseguir inductores en nuestro medio.

REFERENCIAS BIBLIOGRÁFICAS

PRÓLOGO

[1] WOJCIECH Grega, **Hardware-in-the-loop simulation and its application in control education** – "Department of Automatics, University of Mining and Metallurgy – Krakow", San Juan, 1999.

CAPÍTULO 1

[2] www.theorem.net/theorem/lewis1.html, **A Brief History of Feedback Control**

[3] http://www.embedded.com/story/OEG20011129S0054 , **Hardware In The Loop Simulation.**

[4] VISSER Peter M., GROOTHUIS Marcel A. and BROENINK Jan F., "**FPGAs as versatile configurable I/O devices in Hardware-in-the-Loop Simulation"**, Work-In-Progress Proceedings, Lisbon, 5-8 December 2004. www.esi.nl/site/projects/ boderc/publications/129CE2004_Lisbon.pdf.

[5] MATHWORKS, **"Real Time Workshop User's Guide"**, version 6, 2006

[6] http://www.opal-rt.com/, *Página oficial de Opal-RT*

[7] http://www.ni.com/labview/control/hil_test.htm, Sección de simulación HIL de National Instruments.

[8] http://www.mathworks.com/products/xpctarget/, *Página principal del producto de Mathworks XPCTarget*.

[9] http://www.dspace.de/, *Página oficial de dSpace*.

[10] http://www.vissim.com/, *Página oficial de VisSim*.

[11] www.dspace.de/shared/data/pdf/catalog2006/TargetLink_dSPACE_Catalog_2006.pdf, **"Production Code Generation Software"**.

CAPÍTULO 2

[12] OGATA, Katsuhiko, **Ingeniería de Control Moderna**, Tomo 1, 2da edición, Prentice-Hall, México 1993, 1024p.

[13] SMITH, Steven W., **The Scientist and Engineer's Guide to Digital Signal Processing**, Tomo1, 2da edición, California Technical Publishing, San Diego 1999, 630p.

[14] OGATA, Katsuhiko, **Sistemas de Control en Tiempo Discreto**, Tomo 1, 2da edición, Prentice-Hall, México 1996, 745p.

[15] BLANCHARD Paul, DEVANEY Robert C. **Ecuaciones Diferenciales**, Tomo 1, 1ra edición, International Thomson Editores, México 1998, 850p

[16] PRESS William, TEUKOLSKY Saul, VETTERLING W., FLANNERY B., **Numerical Reciepes in C**, Tomo 1, 2da edición, Cambridge University Press, USA 2002, 993p.

[17] ANALOG DEVICES TECHNICAL REFERENCE BOOKS, **Mixed-Signal and DSP Design Techniques,** USA, 2001.

[18] www.informatics.bangor.ac.uk/~dewi/modules/dcs/sig_recon_and_plant_model.pdf **Modeling signal reconstruction.**

[19] www.personal.rdg.ac.uk/~shs99vmb/lab_scripts/Exp227.pdf, **Experiment 227: Computer aided design of a sampled data servo using MATLAB**

[20] PHILLIPS Charles, HARBOR Royce, **Feedback Control Systems**, Tomo 1, Tercera Edición, Editorial Prentice-Hall, USA 1995, 620p.

[21] MATHWORKS, **Control System Toolbox 6 Help**, Matlab 7.

[22] SMITH Carlos, CORRIPIO Armando. **Control Automático de Procesos**, Tomo 1, Primera Edición, Editorial Limusa, México 1997, 717p.

[23] MICROCHIP Technology Inc. **dsPIC30F3014, dsPIC30F4013 Data Sheet** DS70138C, USA, 2004.

CAPÍTULO 3

[24] MICROCHIP Technology Inc. **dsPIC30F Family Reference Manual**, DS70046C, USA, 2004.

[25] PECK Kim, MICROCHIP Technology Inc. **PICmicro™ Microcontroller Oscillator Design Guide**, AN588, USA, 1997.

[26] LACANETTE Kerry, NATIONAL Semiconductor, **A Basic Introduction to Filters – Active, Passive and Switched-Capacitor**, AN779, USA, 1991.

[27] BAKER Bonnie, MICROCHIP Technology Inc., **Using Digital Potentiometers to Design Low-Pass Adjustable Filters**, AN737, USA, 2004.

[28] http://www.ecircuitcenter.com/Circuits/op_voff/op_voff.htm#top, **Opamp Input Offset Voltage**.

[29] JONES David L., **PCB Design Tutorial**, http://alternatezone.com/electronics/files/PCBDesignTutorialRevA.pdf, 2004.

[30] BAKER Bonnie, MICROCHIP Technology Inc., **Layout Tips for 12-Bit A/D Converter Application**, AN688, USA, 2004.

CAPÍTULO 4

[31] MICROCHIP Technology Inc., **MPLAB IDE Users Guide**, DS51519, USA, 2005.

[32] MICROCHIP Technology Inc., **MPLAB ASM30, MPLAB LINK30 and Utilities User's Guide**, DS51317, USA, 2004.

[33] MICROCHIP Technology Inc., **MPLAB C30 C Compiler User's Guide**, DS51284, USA, 2005.

[34] MICROCHIP Technology Inc., **dsPIC Language Tools Libraries***, DS51456*, USA, 2004.

[35] CMX Systems Inc., **CMX – SHEDULER® User's Manual**, USA, 2002.

[36] MICROCHIP Technology Inc., **dsPIC30F Programmer's Reference Manual**, DS70030E, USA, 2003.

[37] MICROCHIP Technology Inc., **dsPIC30F Data Sheet General Purpose and Sensor Families**, DS70083F, USA, 2004.

[38] MICROCHIP Technology Inc., **dsPIC30F3014, dsPIC30F4013 Data Sheet**, DS70138, USA, 2004.

[39] http://perso.wanadoo.es/siscobf/winpic800.htm, **WinPic800, Free - Software programador de PIC - Windows 95/98/NT/2000/ME/XP compatible**

[40] http://www.ingenia-cat.com/soportecnic.php?tema=downl&lang=es, Ingenia, **Herramientas software de desarrollo**.

[41] INGENIA-CAT S.L., **Ingenia dsPIC bootloader User's Guide v1.00**, España, 2005.

[42] PUMPKIN Inc, **Salvo User's Manual**, http://www.pumpkininc.com/, USA, 2002.

[43] OSEK Group, **OSEK/VDX Operating System v2.1r1**, *N/D*, 2000.

CAPÍTULO 5

[44] Johnson Curtis D., **Process Control Instrumentation Technology**, Tomo 1, Sexta Edición, Editorial Prentice-Hall Inc., USA 2000, 678p.

EMULADOR SISO

A N E X O S

EMULADOR SISO

ANEXO A1

MANUAL DE OPERACIÓN DEL EMULADOR SISO

ÍNDICE
MANUAL DE OPERACIÓN

MANUAL DE OPERACIÓN DEL EMULADOR SISO

1- INTRODUCCIÓN

Descripción General

El emulador SISO es una solución de Hardware y Software que permite el reemplazo de plantas didácticas a escala para realizar prácticas de control. Por medio de la interfase de usuario (GUI) es posible configurar el rango de las señales de entrada/salida, fijar el período de simulación, definir la función de transferencia, el retardo de transporte, la no-linealidad y las perturbaciones al sistema los cuales son descargados por medio del puerto serial a la memoria del hardware, el cual una vez activado, se comporta como el sistema previamente definido. Al hardware se le pueden realizar actualizaciones de firmware con el fin de mejorar el programa de la aplicación o darle otro uso.

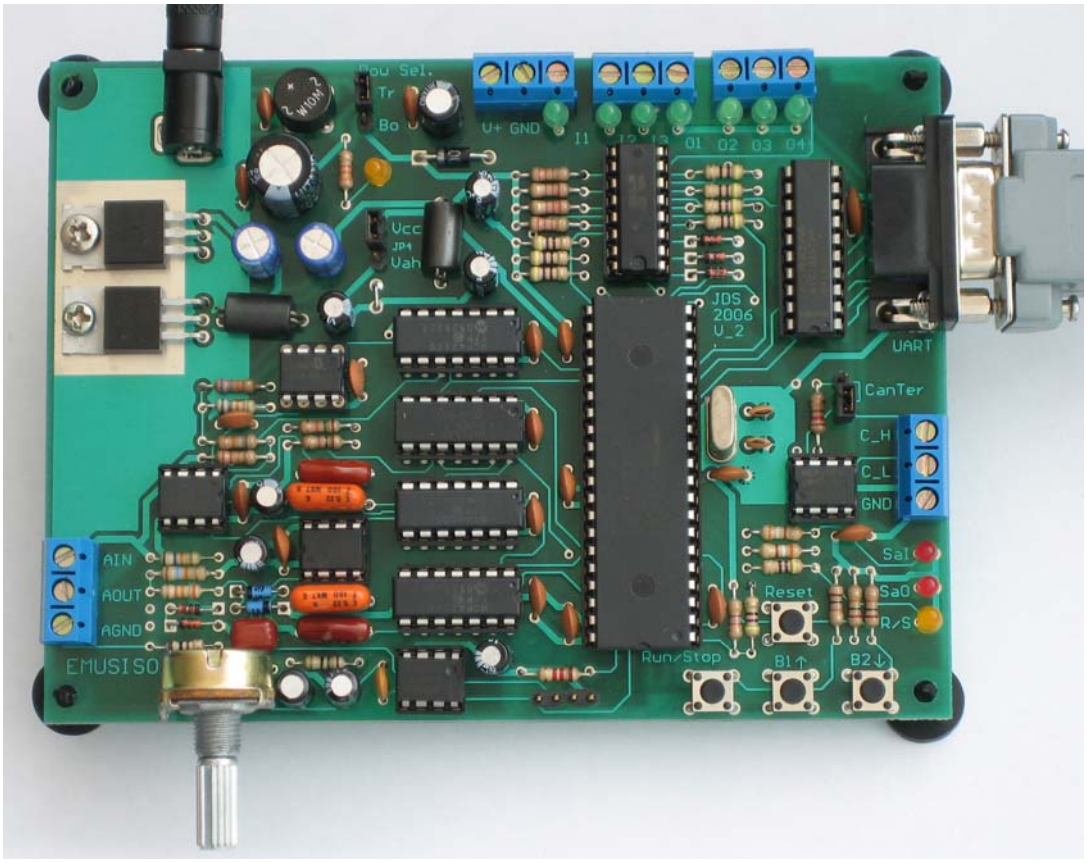


Figura. M1. Hardware del Emulador SISO con sus componentes

2- COMPONENTES DEL EMULADOR SISO

El emulador SISO viene con los siguientes componentes:

- Hardware del emulador SISO (Circuito impreso con sus componentes).
- Adaptador de 16VDC.
- Cable de interfase RS-232.
- Firmware del emulador SISO precargado.
- Firmware de carga al arranque (bootloader) precargado.
- Programa de interfase de usuario (GUI) para MATLAB 7.
- Programa de carga al arranque Ingenia dsPicbootloader.

3- CARACTERÍSTICAS DEL HARDWARE DEL EMULADOR SISO

3.1 Energización de la placa

- Reguladores incluidos para voltajes de placa de +5V y +15V por medio de un adaptador externo de 17 a 30 Vdc ó 16-20 Vac accesible por un jack estándar.

- Alimentación opcional de 24Vdc por bornera.

- LED indicador de circuito energizado.

- Consumo de 250mA máx.

3.2 Comunicaciones

- Dos puertos seriales RS-232 accesibles por un solo conector DB9 macho.

- Comunicación CAN 2.0B accesible por borneras.

3.3 Entradas/Salidas digitales

- Tres entradas y cuatro salidas digitales con rangos programables (por polarización externa) entre 5 y 24 V.

- Dos de las entradas pueden ser utilizadas como contadores. Las cuatro salidas pueden ser configuradas como salidas de PWM.

- Las salidas pueden absorber 200mA (colectores abiertos).

3.4 Entrada/Salida analógica

- Una entrada analógica unipolar con rango programable entre 5 - 12 V con filtro pasabajos bessel analógico programable de 2do orden entre 10Hz y 500Hz.

- Una salida sanalógica unipolar con rango programable entre 5 – 12 V con filtro pasabajos bessel analógico programable de 2do orden entre 10Hz y 500Hz. Incluye compensación de offset programable.

- Voltaje de referencia interno de alta precisión que garantiza estabilidad de las mediciones ante variaciones de temperatura y envejecimiento.

3.5 Controles e indicadores para el usuario

- Tres botones que controlan el hardware (RUN/STOP, UP, DOWN).

- Botón para resetear el dispositivo.

- Potenciómetro mecánico para ingresar perturbaciones.

- 10 LEDs indicadores de las entradas/salidas analógicas y digitales.

3.6 Adicionales

- Conector para acceder a las señales analógicas y voltaje de referencia con el fin de facilitar la calibración del hardware.

- Todos los chips están colocados en zócalos para facilitar la reparación o actualización de los componentes de la placa.

4- TUTORIAL

El tutorial presenta la manera de poner a trabajar al emulador SISO. Se describe primeramente el hardware y luego la GUI. La referencia completa del software se presenta en la sección 5 – Interfase de Usuario (GUI).

4.1 Descripción del Hardware del emulador SISO

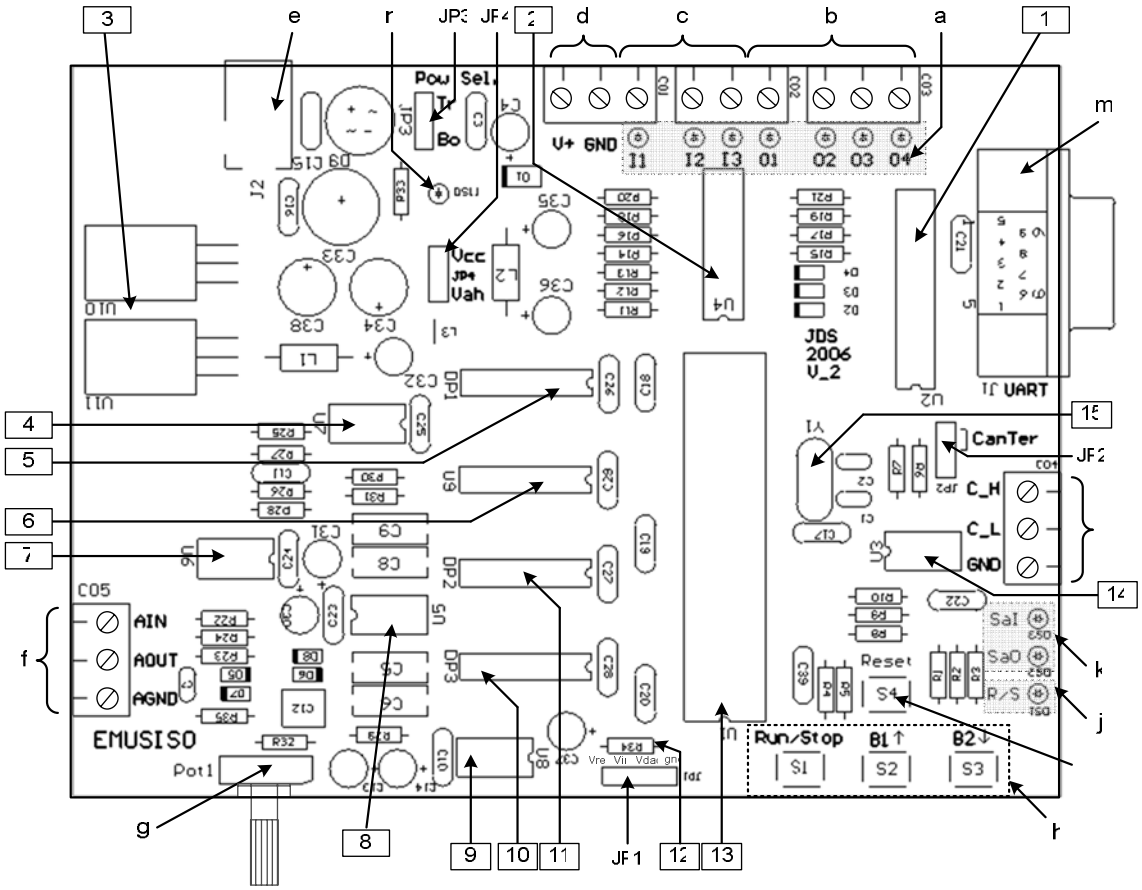


Figura M2 – Elementos del hardware del emulador SISO

Conectores e Interfase	
a	Indicadores de las entradas y salidas digitales.
b	Salidas digitales
c	Entradas digitales
d	Polarización de las entradas/salidas digitales (V+ = 5 – 24 Vdc). También se usa como terminales de alimentación si JP3 está en Bo y V+ = 17 – 24 Vdc.
e	Conector para el adaptador de 17 – 30 Vdc / 16 – 20 Vac

f	Entrada (AIN)/Salida (AOUT) analógica y tierra analógica (AGND)
g	Potenciómetro para el ingreso de perturbaciones al sistema
h	Botones de interfase. RUN/STOP se usa en el emulador SISO. B1 y B2 se dejan de reserva.
i	Botón para resetear el procesador.
j	Indicador del estado del emulador SISO.
k	Indicadores de la entrada/salida analógica.
l	Terminales para la conexión de una red CAN.
m	Conector DB9 – Macho para conexión con el PC u otros periféricos. El emulador SISO es un equipo CTU (se necesita un cable cruzado para comunicarse con una PC).
n	Indicador de que la placa está energizada.
1	Transceiver del puerto RS-232.
2	Driver de las entradas y salidas digitales.
3	Reguladores lineales de 5 y 15 V.
4	Amplificador operacional. Buffer del potenciómetro analógico y calibración offset.
5	Potenciómetro digital 100kΩ. Calibra la ganancia de entrada y la corrección de offset a la salida.
6	Conversor digital-analógico.
7	Amplificador operacional. Buffer de la entrada y salida analógica.
8	Amplificador operacional. Filtro pasa bajos de entrada y salida.
9	Generador de voltaje de referencia de 4.096V.
10	Potenciómetro digital 50 kΩ. Junto con [11] fijan la frecuencia de corte de los filtros de entrada y salida.
11	Potenciómetro digital 100 kΩ. Junto con [10] fijan la frecuencia de corte de los filtros de entrada y salida.
12	Toma para medir la salida de frecuencia del controlador con fines de calibración.
13	Controlador digital de señales (Procesador del sistema).
14	Transceiver del puerto de comunicación CAN.
15	Cristal oscilador para el procesador de 7.3728 MHz.
JP1	Conector para acceso a las señales de Vref, Vin(filtrada), Vout (no filtrada) y AGND
JP2	Permite conectar la resistencia de terminación de 120Ω entre CanH y CanL.
JP3	Permite seleccionar entre energizar el circuito con adaptador externo (Tr) o con la fuente conectada en la bornera [d] (Bo).
JP4	Selecciona el voltaje de polarización de los amplificadores operacionales en [4] y [8] según las características de los mismos. Vcc (5V) ó Vah (15V).

Tabla M1 – Descripción de componentes de la placa

4.2 Indicadores de la placa

La placa posee una serie de indicadores para conocer el estado de las entradas/salidas y de la simulación.

RUN/STOP	Según el estado de simulación este LED puede estar: Prendido: El emulador está listo para el trabajo pero la simulación se encuentra detenida. Titilando: El proceso de emulación se encuentra en progreso. Apagado: Ha ocurrido un error y el dispositivo debe reiniciarse.
SIN	Saturación en la entrada. Si la señal analógica de entrada supera el rango establecido en la configuración este indicador se prenderá por al menos medio segundo.
SOUT	Saturación en la entrada. Si la señal analógica de salida alcanza el máximo del rango establecido en la configuración este indicador se prenderá por al menos medio segundo.
I1	Si cualquiera de estos indicadores se prende indica que la señal de entrada correspondiente posee un valor lógico de 1.
I2	
I3	
O1	Si cualquiera de estos indicadores está encendido en la bornera se tiene un voltaje igual a V+. En caso de estar apagados el voltaje en la salida es GND.
O2	
O3	
O4	Es la salida de PWM para el emulador y, dependiendo de la frecuencia del PWM y su ciclo de trabajo, puede variar la luminosidad del indicador. Cuando no está activada la función de LED trabaja igual que O1-O3.

Tabla M2 – Descripción de indicadores en la placa

4.3 Prerrequisitos para la Interfase de usuario (GUI)

- En la PC debe estar instalada la versión de Matlab 7.0 (R14) o superior.
- Tener instalado el Instrument Control Toolbox 2.0 o superior.
- Tener instalado el Control System Toolbox 6.0 o superior.
- Tener instalado el Fixed-Point Toolbox 1.0 o superior.
- Debe haber por lo menos un puerto serial RS232 en el equipo y el hardware del emulador debe estar conectado para poder acceder a ciertas funcionalidades.
- Si la PC no posee un puerto serial RS232 (común en computadoras portátiles) es posible trabajar con un adaptador USB-serial, ya que su funcionalidad ha sido probada para la GUI y el emulador SISO.

4.4 Instalación de la GUI

Desde el CD de datos, copiar la carpeta \sisoemu en la carpeta de trabajo de Matlab (*Matlabroot*\work)

4.5 Ingreso a la GUI

- Abrir MATLAB 7.
- Fijar a sisoemu como el directorio actual.

- Escribir en la ventana de comandos sisoemu y presionar *enter*.

4.6 Conexión con el Hardware

- Conectar el Emulador SISO con la PC usando el cable RS-232 de interfase proporcionado.
- Abrir la aplicación sisoemu, como se indicó anteriormente.
- Ingresar al menú *Emulador*, *Conectar…*
- Se abre el cuadro de diálogo de configuración de comunicaciones.
- En *Puerto* se despliegan todos los puertos seriales disponibles para comunicarse. Seleccione el puerto correspondiente a la conexión física.
- Asegúrese que la velocidad de comunicación está fijada en 9600 bps.
- Presionar OK.
- Si la conexión se realiza con éxito, en el menú *Emulador* aparece *Desconectar* en vez de *Conectar…* y las demás opciones del menú se vuelven activas. En caso de que se presente un error al conectarse la interfase se encargará de informar oportunamente.

4.7 Carga y descarga de la configuración

- En la sección 5: *Interfase de usuario – GUI*, se detalla la manera de ingresar los datos para configurar la planta.

- Para descargar la configuración deseada seguir *Emulador* → *Descargar Planta*. Enseguida se abre el cuadro con la barra de avance de la descarga. La simulación en la planta debe estar detenida para que se pueda descargar.

- El Emulador SISO guarda la configuración en memoria no volátil por lo que es posible conservar la misma aún cuando se quite la energía. Para recuperar la configuración de la planta seguir *Emulador* → *Cargar Planta* y enseguida se abre el cuadro de diálogo que indica el progreso de la carga. Después de culminar la carga se refresca la interfase actualizando los valores según los datos residentes en el emulador.

4.8 Iniciar/Parar la emulación

- Siguiendo *Emulador* → *Iniciar/Parar* se puede arrancar la emulación de la placa desde la GUI. Este procedimiento tiene el mismo efecto que presionar el botón de RUN_STOP en la placa. Si el proceso inicia correctamente el LED de RUN/STOP empieza a titilar.

- Para parar el Emulador SISO seguir *Emulador* → *Iniciar/Parar* mientras en el emulador titilea el LED RUN/STOP. Un cuadro de diálogo pide confirmación de la acción. Este procedimiento tiene el mismo efecto que presionar el botón RUN_STOP en la placa. Si se detiene correctamente el LED deja de titilar y queda encendido.

5- INTERFASE DE USUARIO – GUI

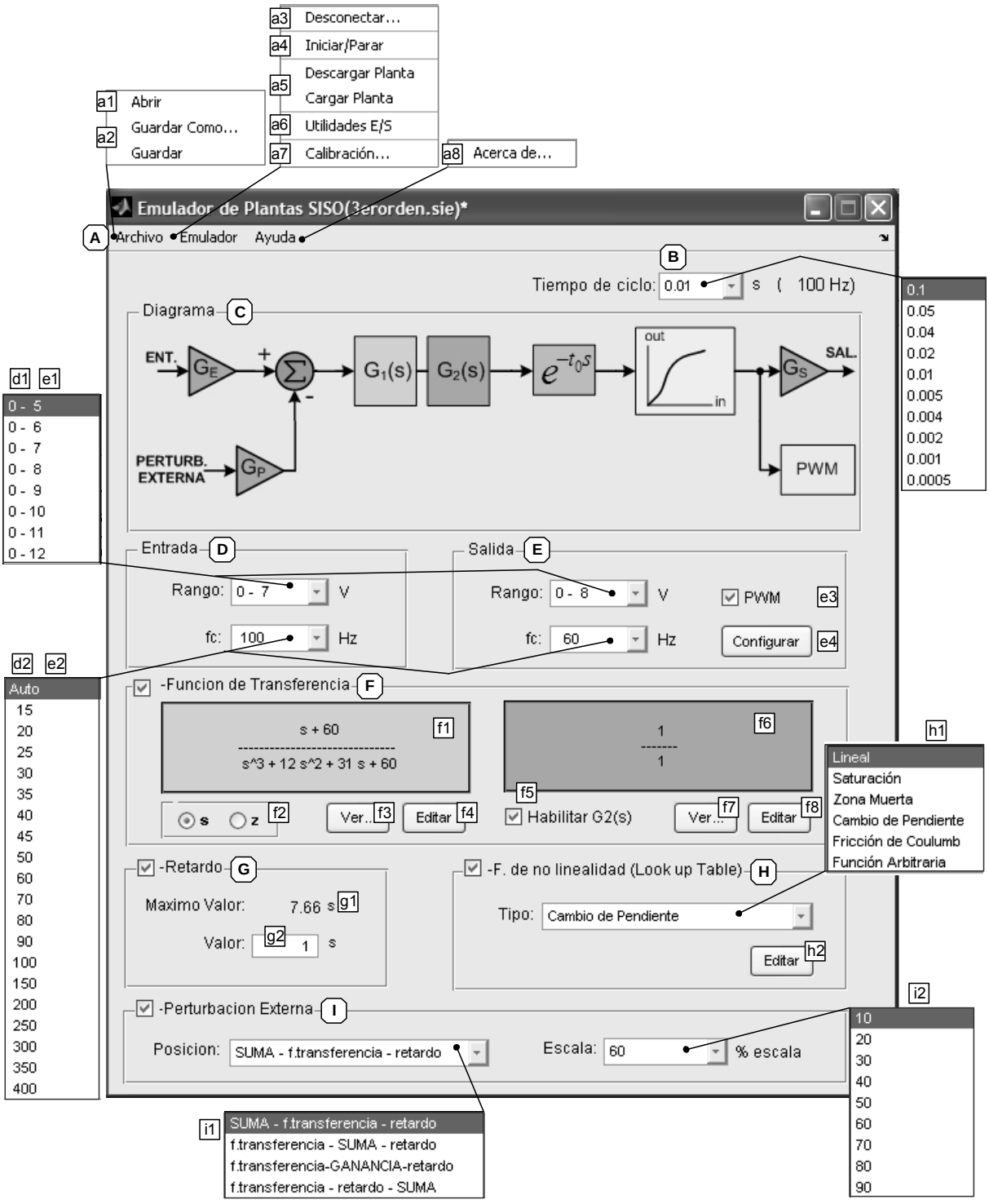


Figura S1 – Pantalla principal de la GUI del emulador SISO

A	Menú de opciones del emulador SISO
B	Menú para configurar el período de simulación.
C	Diagrama del flujo que se tiene para la emulación.
D	Controles para configurar la entrada
E	Controles para configurar la salida
F	Controles para configurar la(s) función(es) de transferencia
G	Controles para configurar el retardo de transporte
H	Controles para configurar la función de no linealidad
I	Controles para fijar la perturbación externa

5.1 (A) MENÚ DE OPCIONES

A1 – Permite abrir archivos *.sie pertenecientes a la aplicación sisoemu, de tal manera que se carga a la GUI la configuración del archivo.

A2 – Permite guardar una configuración en un archivo con extensión *.sie. Adicionalmente permite guardar los cambios desde la última modificación al archivo actual abierto.

A3 – Opción de Conexión/desconexión. El estado inicial de esta opción es [Conectar…] y permite acceder al cuadro de diálogo (Figura S2) de selección del puerto serial y configuración del mismo.

Figura S2 – Pantalla de configuración de comunicaciones

El menú desplegable de *Puerto* contiene los puertos seriales libres que posee el equipo y por medio de los cuales hay como comunicarse al emulador SISO. Adicionalmente permite configurar la velocidad de conexión con el emulador, aunque hasta la versión 2.0 del emulador la velocidad es fija en 9600 kbps. Cuando el equipo se ha conectado, en lugar de [Conectar…] aparece [Desconectar…] el cual al ser presionado desconecta el equipo y libera al puerto serial.
Si existen problemas al conectarse con el hardware del emulador SISO se pueden presentar los siguientes errores:

⊗	Falla al abrir el puerto COMx
⊗	No hay ningún dispositivo conectado al puerto. Vuelva a intentar…
⊗	No se detectó el dispositivo “Emulador de plantas SISO”

Estos mensajes pueden también presentarse si después de iniciar una sesión válida se produce una desconexión al hardware o se presenta algún otro tipo de problema.

A4 – Iniciar/Parar solo está habilitado cuando el hardware del emulador SISO se encuentra conectado y con una sesión valida en progreso (se logró comunicar correctamente el emulador SISO con la PC). Por medio este menú se puede iniciar o parar la emulación en el Hardware. Equivale a presionar el botón RUN_STOP en el hardware del emulador.

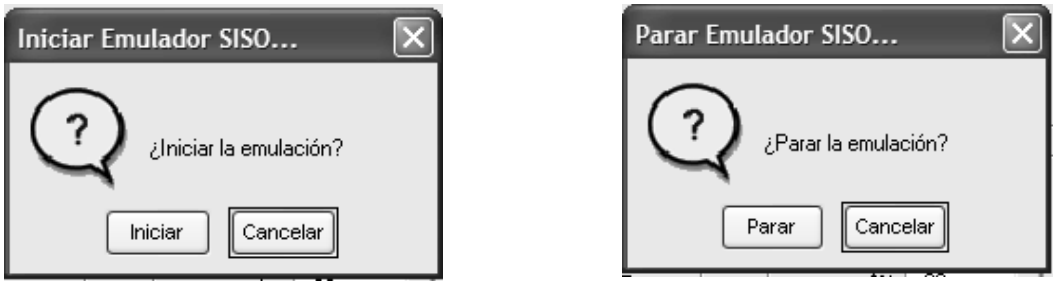


Figura S3 – Cuadros de diálogo para iniciar/parar la emulación.

A5 – Cargar/Descargar configuración (solo está habilitado cuando el hardware del emulador SISO se encuentra conectado y con una sesión valida en progreso, además la emulación debe estar detenida). Estas opciones permiten **descarga**r

el modelo de la planta que se tiene en la GUI hacia el hardware del emulador o **cargar** el modelo que se encuentra almacenado en la memoria del emulador hacia la interfase de usuario. Barras de avance indican el progreso de estas transferencias.

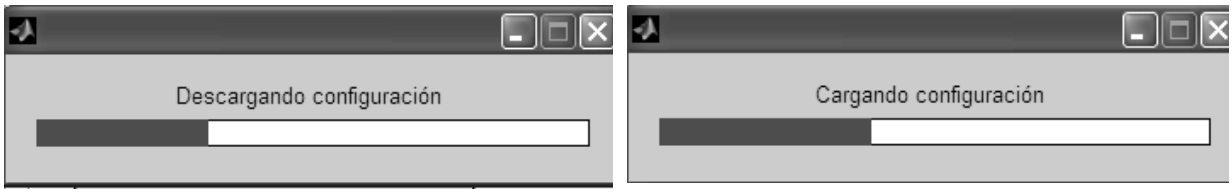


Figura S4 – Barras de descarga/carga de la configuración.

A6 – Utilidades de entrada/salida (solo está habilitado cuando el hardware del emulador SISO se encuentra conectado y con una sesión válida en progreso, además la emulación debe estar detenida).
Esta opción permite acceder al cuadro de diálogo que controla los distintos parámetros del hardware del emulador SISO.

Advertencia: *Esta opción puede modificar las ganancias de la entrada y la salida, por lo que se recomienda revisar los rangos de la señal conectada a la entrada y verificar si el equipo conectado a la salida soporta voltajes hasta de 13V.*

U1	Permite escribir un valor entre 0-4095 en el canal A del DAC. Corresponde a la salida analógica.
U2	Permite escribir un valor entre 0-4095 en el canal B del DAC. Corresponde a la ganancia de la salida analógica.
U3	Permite escribir un valor entre 0 y 255 al potenciómetro que determina la ganancia de entrada.
U4	Permite escribir un valor entre 0 y 255 al potenciómetro que corrige el offset de la salida
U5	Fija los valores de los potenciómetros de 50k y 100k que definen el comportamiento del filtro de primer orden de la entrada.
U6	
U7	Fija los valores de los potenciómetros de 50k y 100k que definen el comportamiento del filtro de primer orden de la salida.
U8	
U9	Hace efectivo los cambios para los potenciómetros digitales.
U10	Realiza una lectura de los dos canales analógicos del emulador SISO. Los valores obtenidos están entre 0 y 4095.
U11	Cierra este cuadro de diálogo.

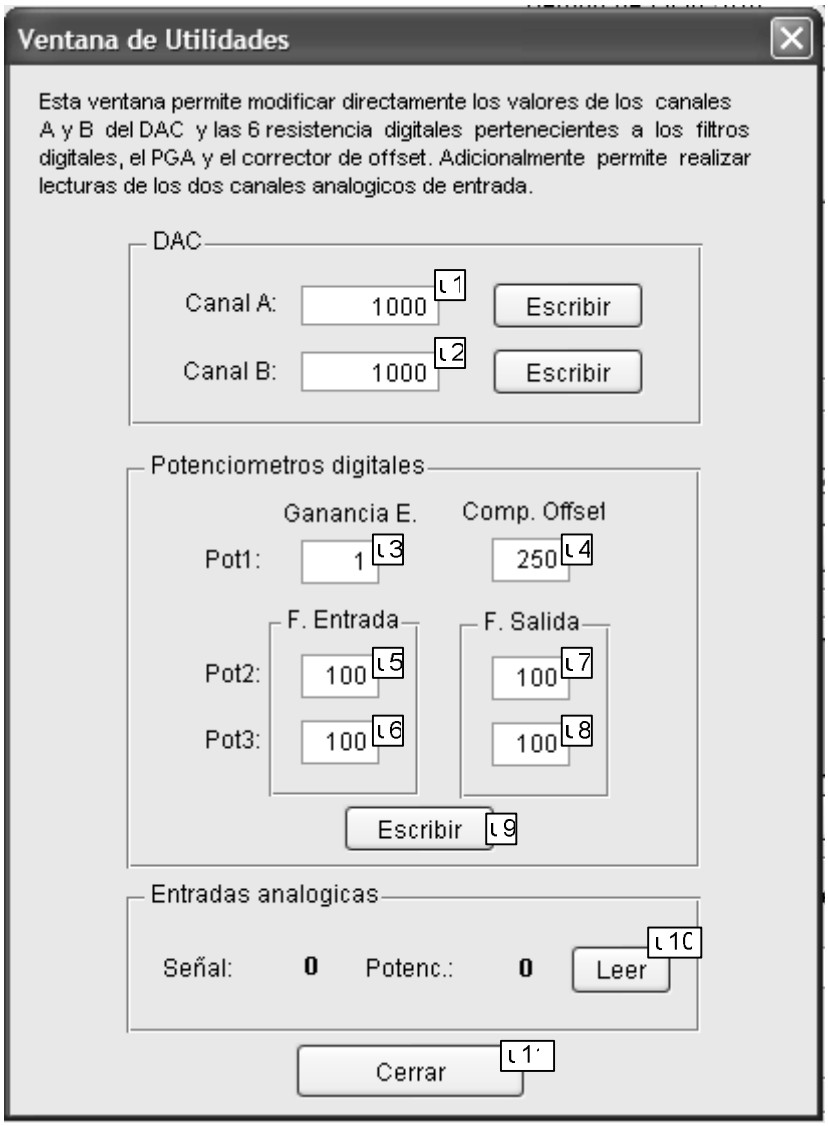


Figura S5 – Cuadro de diálogo de utilidades.

Nota. Mientras éste cuadro de diálogo está activo no es posible iniciar un proceso de emulación.

A7 – Calibración (solo está habilitado cuando el hardware del emulador SISO se encuentra conectado y con una sesión válida en progreso, además la emulación debe estar detenida).

Esta opción abre el cuadro de diálogo que permite ver/modificar los parámetros de calibración del hardware del emulador SISO.

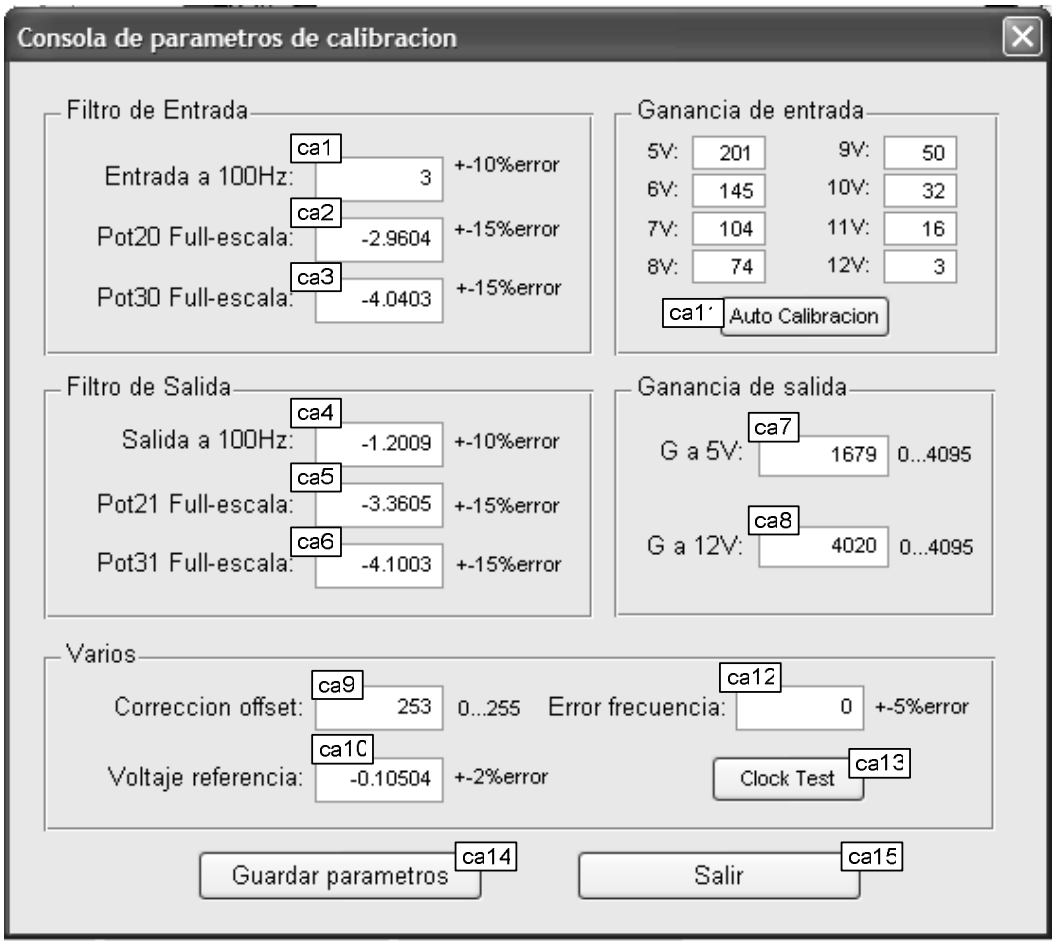
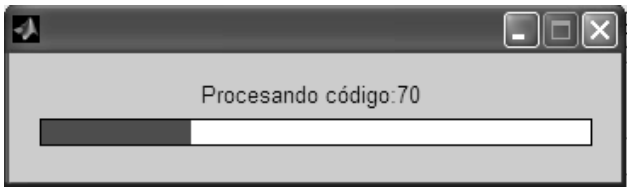
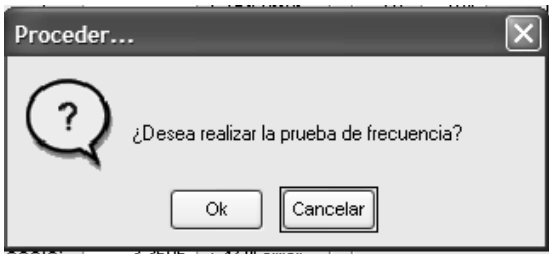
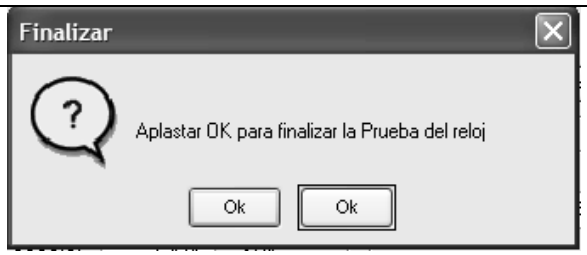


Figura S6 – Cuadro de diálogo para calibrar el sistema.

ca1	Error en la atenuación de -3db ante una señal senoidal de 100Hz por parte del filtro de entrada.*
ca2	Error del potenciómetro 20 fijado en un valor de 255 respecto a un valor de 50kΩ. *
ca3	Error del potenciómetro 30 fijado en un valor de 255 respecto a un valor de 100kΩ. *
ca4	Error en la atenuación de -3db ante una señal senoidal de 100Hz por parte del filtro de salida. En la mayoría de casos el valor es muy cercano a ca1. *
ca5	Error del potenciómetro 21 fijado en un valor de 255 respecto a un valor de 50kΩ. *
ca6	Error del potenciómetro 31 fijado en un valor de 255 respecto a un valor de 100kΩ. *
ca7	Valor digital en el canal B del DAC para el cual la salida entrega 5V. Se debe usar en conjunto con el diálogo de utilidades (A6).
ca8	Valor digital en el canal B del DAC para el cual la salida entrega 12V. Se debe usar en conjunto con el diálogo de utilidades (A6).
ca9	Valor en el potenciómetro 11 que permite tener 0V a la salida cuando el canal A del DAC está en cero (Corrige el offset).

ca10	Error en el voltaje de referencia respecto a un valor de 4.096V.
ca11	Permite encontrar automáticamente los valores del potenciómetro 10 para las distintas escalas. Se supone que los valores de ca7 y ca8 han sido correctamente calibrados. Los cuadros que se muestran a continuación se despliegan para guiar en el proceso de autocalibración.  *Figura S7 – Guía para la autocalibración.* Una vez que empieza la calibración se muestra la barra de avance del proceso.  *Figura S8 – Barra de progreso de la autocalibración.*
ca12	Error de la frecuencia interna del DSC respecto a la ideal de 29.4912 MHz *
ca13	Permite que el emulador SISO genere una señal cuadrada que corresponde a ¼ del ciclo de máquina (Nominalmente 7.3728MHz) con el fin de poder medir el error en la frecuencia. Antes de iniciar la prueba se pide confirmación:  El cuadro Finalizar se mantiene durante toda la prueba hasta que se aplasta OK.

	 *Figura S9 – Cuadros del test de frecuencia.* ***Nota**: Durante la prueba de reloj el emulador SISO no responde a ningún botón ni comando serial. Si existe algún problema se recomienda resetear el dispositivo.*
ca14	Guarda los parámetros que se hayan modificado
ca15	Sale del cuadro de diálogo sin que se registren las modificaciones hechas a menos que se haya presionado previamente ca14.
* Los errores de estos valores se registran en un registro de 16bits en el cual 0 corresponde al valor inferior y 65535 al superior. 32768 es el valor de la mitad (0% de error en todos los casos).	

A8 – Acerca de… -Muestra la versión del Emulador SISO.

Figura S10 – Cuadro de ayuda.

5.2 (B) PERÍODO DE SIMULACIÓN

Este menú muestra los distintos períodos de simulación para el emulador SISO. El usuario debe escoger el que más le convenga según el tipo de planta que desea simular.

5.3 (C) DIAGRAMA DE FLUJO

En esta sección de la interfase se muestra el flujo que va teniendo la planta según el usuario activa/desactiva y configura los distintos bloques que desea se emulen en el dispositivo.

5.4 (D) CONFIGURACIÓN ENTRADA

D1- Fija el rango de la señal de entrada unipolar. Para el rango que se fije se tendrá la escala completa. (pe. 0-5V entrada → 0 – 4095 digital).
D2- Fija la frecuencia de corte del filtro antialiasing analógico de entrada. En función "auto" el programa fija la frecuencia más conveniente según la frecuencia de emulación y la función de transferencia especificada.

5.5 (E) CONFIGURACIÓN DE LAS SALIDAS

E1 – Fija el rango de la señal de salida unipolar. Para el rango que se fije se tendrá la escala completa. (pe. 0-4095V digital → 0 – 5 V salida).

E2 – Fija la frecuencia de corte del filtro antialiasing analógico de salida (después del DAC). En función "auto" el programa fija la frecuencia más conveniente según la frecuencia de emulación y la función de transferencia especificada.

E3 – Activa/desactiva la salida de PWM por O4.

E4 – Abre el cuadro de diálogo que permite configurar la señal de PWM.

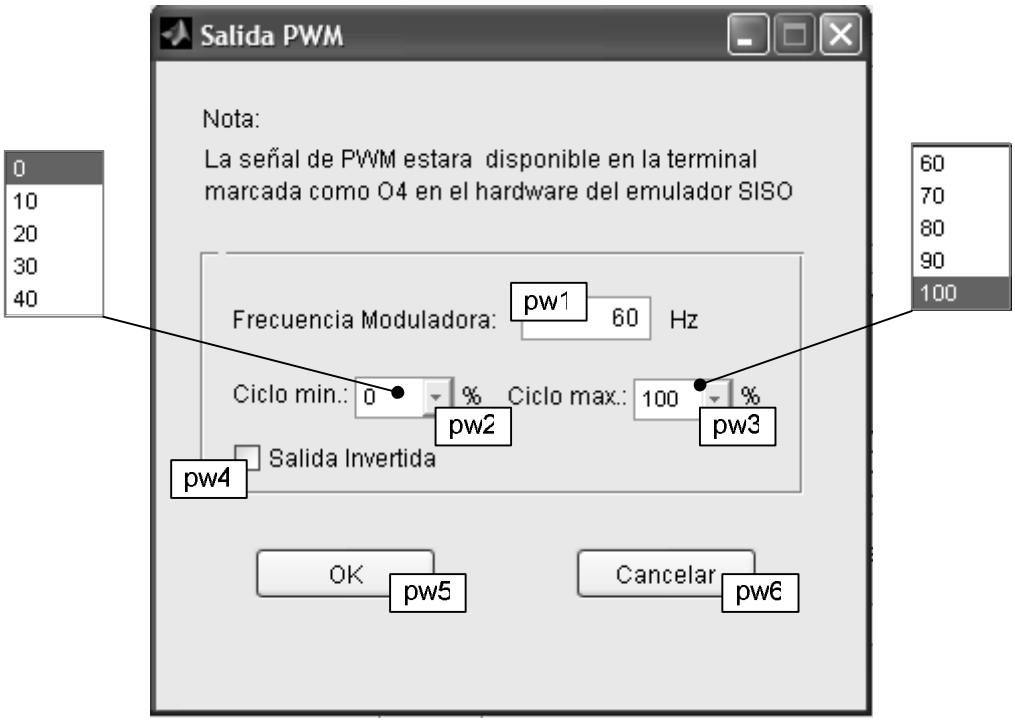


Figura S11 – Configuración de la salida PWM.

pw1	Fija la frecuencia de los pulsos de pwm de 10 a 30000 Hz.
pw2	Fija el ciclo mínimo de trabajo del pulso PWM (corresponde a 0V en la salida analógica).
pw3	Fija el ciclo máximo de trabajo del pulso PWM (corresponde al full-escala en la salida analógica).
pw4	Invierte la lógica de la señal a la salida según la conveniencia
pw5	Se confirman los cambios realizados y se cierra el diálogo.
pw6	Se ignoran los cambios realizados y se cierra el diálogo.

5.6 (F) FUNCIÓN DE TRANSFERENCIA

El cuadro de chequeo permite activar o desactivar la función de transferencia. En el caso que no esté activado, el resto de controles no serán accesibles.

F1 – Muestra la función de transferencia actual.

F2 – Permite que en F1 se muestre la función de transferencia ya sea en dominio continuo (s) ó discreto (z). En el caso de que esté en dominio discreto, los valores se modifican si se cambia el período de emulación.

F3 – Ver: Muestra la respuesta ante el escalón de la función de transferencia en F1 en el modo que se encuentre actualmente (continua o discreta).

F4 – Editar: Accede al cuadro de diálogo que permite ingresar los valores de los coeficientes de la función de transferencia en tiempo continuo.

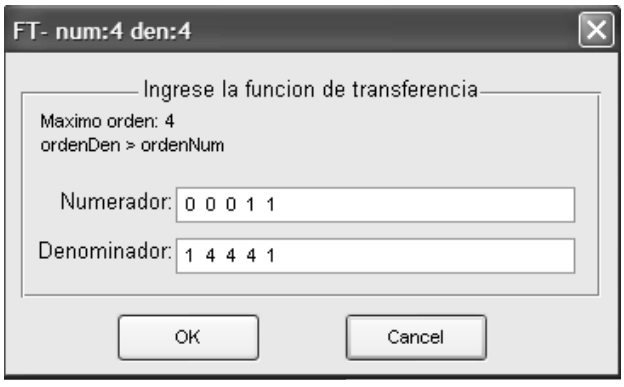


Figura S12 – Configuración de la función de transferencia.

El cuadro proporciona las protecciones para impedir el ingreso de valores incorrectos o que superen el orden máximo soportado según el contexto (si está activada la función de transferencia 2). Si se presiona OK se confirman los cambios, caso contrario (Cerrar o Cancelar) el cuadro se cierra sin que los cambios tengan efecto. Los errores y advertencias que entrega este diálogo son:

	¡Los elementos no están bien ingresados!
	Orden del numerador mayor al del denominador
	Denominador de orden mayor a_*x*
	Numerador de orden mayor a_*x*

F5 – Permite que la función de transferencia sea descrita por dos funciones de transferencia en cascada.

F6 – Muestra la segunda función de transferencia.

F7 – Ver: Muestra la respuesta ante el escalón de la función de transferencia en F6 en el modo que se encuentre actualmente (continua o discreta).

F8 – Editar: Accede al cuadro de diálogo que permite ingresar los valores de los coeficientes de la segunda función de transferencia en tiempo continuo.

5.7 (G) RETARDO

El cuadro de chequeo permite activar o desactivar la función de retardo de transporte. En el caso que no esté activado, el resto de controles no serán accesibles.

G1 – Muestra el valor máximo de retardo que se puede tener para las condiciones de contexto actuales. Su valor cambia ante una modificación en la frecuencia de emulación o un cambio en la función de transferencia.

G2 – Permite ingresar el valor de retardo deseado. Según el tiempo de emulación la GUI redondea el valor ingresado al correspondiente múltiplo del período de emulación. Impide que se ingresen caracteres no numéricos o fuera del rango permitido.

5.8 (H) NO LINEALIDAD

El cuadro de chequeo permite activar o desactivar la función de no linealidad. En el caso que no esté activado, el resto de controles no serán accesibles.

H1 – Este menú permite escoger entre los distintos tipos de no-linealidades que se desea implementar en el flujo de la planta.

H2 – Abre el cuadro de diálogo que permite configurar la no Linealidad. La Figura S13a muestra el diálogo para todas las perturbaciones menos la de Función Arbitraria (Figura S13b). Debido a que la tabla de búsqueda que define esta función está normaliza (la entrada varía de 0 a 1 y lo mismo la salida), el usuario tendrá que revisar el comportamiento de la misma respecto a las escalas que se han fijado en la entrada y en la salida.

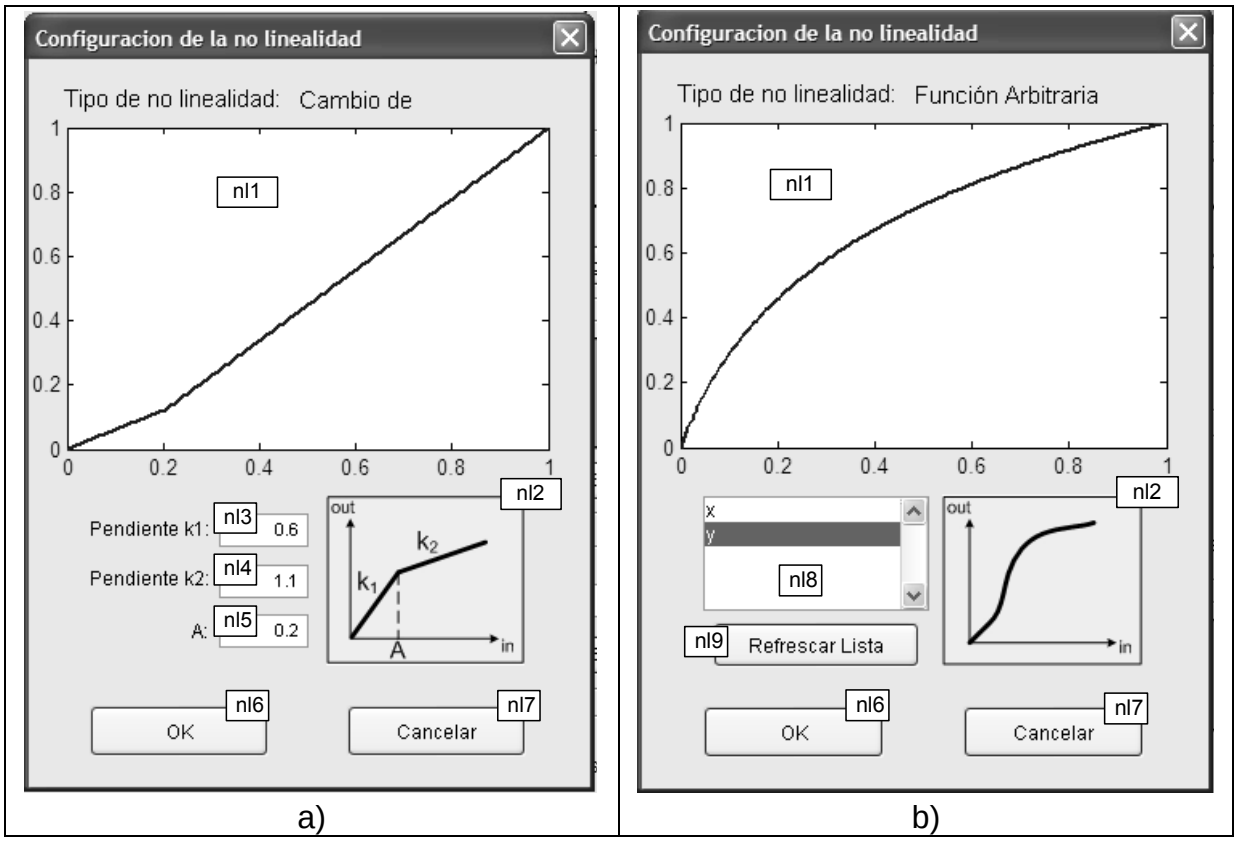


Figura S13 – Configuración de la no-linealidad

nl1	Muestra la forma en que se relaciona la salida con la entrada.
nl2	Dibujo que indica la función de cada parámetro.

nl3	Fija la pendiente k1 del gráfico.
nl4	Fija la pendiente k2 del gráfico (si es aplicable).
nl5	Fija el lugar de inflexión según el gráfico (si es aplicable). El valor debe estar entre 0 a 1.
nl6	Se registran los cambios realizados y se cierra el cuadro de diálogo.
nl7	Se ignoran los cambios realizados y se cierra el cuadro de diálogo.
nl8	Muestra los vectores del *Workspace* del MATLAB que pueden ser usados para definir la función de no linealidad. Al seleccionar uno de estos se dibuja el mismo en nl1.
nl9	Actualiza la lista de vectores utilizables existentes en el *Workspace* para definir la función arbitraria.

Los errores que se puede dar en esté diálogo están relacionados con el ingreso de datos:

	¡Ingresar solo números!
	¡A debe estar entre 0 y 1!

5.9 (I) PERTURBACIÓN EXTERNA

El cuadro de chequeo permite activar o desactivar las distintas perturbaciones externas. En el caso que no esté activado, el resto de controles no serán accesibles.

I1 – Menú que permite escoger el lugar donde va la perturbación externa y cual se trata (aditiva o multiplicativa). Dependiendo de la opción escogida se actualiza el Diagrama de flujo (C).

I2 – Permite escoger el rango que tendrá la perturbación externa. Por ejemplo, cuando se tiene una perturbación aditiva y se escoge la escala de 50%, la perturbación tendrá una variación desde 0 (potenciómetro completamente al tope en sentido antihorario) hasta 50% de la full-escala (potenciómetro completamente al tope en sentido horario). Si ésta full-escala es de 7V, entonces la perturbación aditiva tendrá una variación entre 0 y 3.5V. Cuando la perturbación es en ganancia es posible escoger entre escalas de (0.1, 0.2, 0.5, 1, 1.5, 2, 2.5, 3, 4 y 5). Respecto al mismo razonamiento anterior, si el potenciómetro se encuentra al tope anti-horario la ganancia es cero, mientras que en el tope opuesto se logra la ganancia indicada.

6- EJEMPLO DE APLICACIÓN

6.1 Definición del ejercicio

El presente ejercicio sirve para conocer las funcionalidades del emulador SISO. Se propondrá el control de una planta de intercambio de calor por medio de un PLC Modicon Micro 612 cuya interfase con el usuario estará implementada en Lookout 5.0.

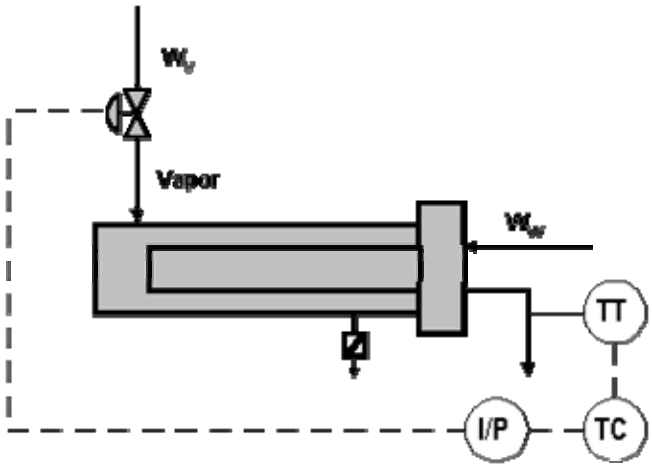


Figura E1 Planta emulada

La planta presenta tres componentes importantes: función de transferencia de 1er orden, retardo de transporte y no-linealidad estática. Se incluye una perturbación aditiva antes del retardo. De antemano se conocen las siguientes características de la planta:

Rango de entrada al actuador (W_V): 0 -10 V
Rango de salida del sensor de temperatura (TT) : 0 – 7 V
Rango de la perturbación: 70%.

Función de transferencia: $G(s) = \frac{1}{8s+1} \cdot e^{-1.1s} = \frac{TT(s)}{W_V(s)}$ **

Ecuación de relación de la no linelaidad: $y = 8.9425 \cdot \left(e^{0.025 \cdot x} - 1\right)$

***no se considera la escala. En el emulador SISO es recomendable que la función de transferencia de la planta tenga una ganancia en DC de 1 y que sean los rangos de salida los que definen esta ganancia.*

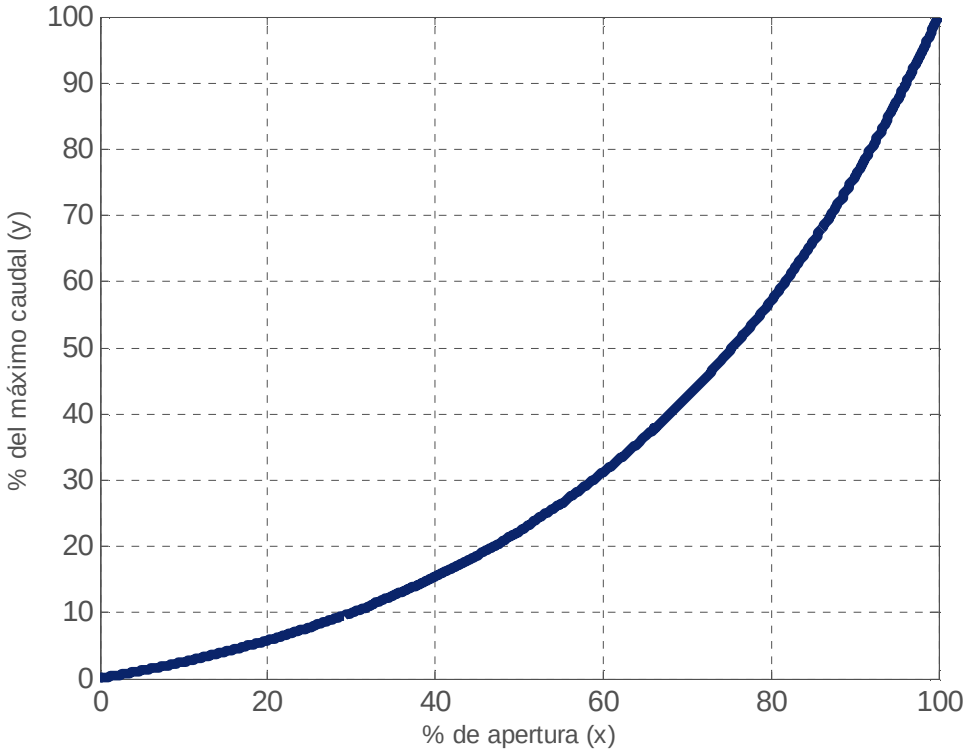


Figura E2 – Relación de no linealidad de la planta

6.2 Conexión de los equipos

Para asegurar un correcto funcionamiento de los equipos y evitar daños es necesario verificar lo siguiente:

- La polarización de la entrada y salida analógica debe ser correcta. El emulador SISO solo trabaja con voltajes unipolares.

- Verificar que la salida analógica del equipo conectado al emulador no produzca voltajes bipolares.

- Verificar que la fuente de alimentación es capaz de suministrar suficiente corriente para los dos equipos. En caso que no sea suficiente usar el adaptador externo del emulador SISO.

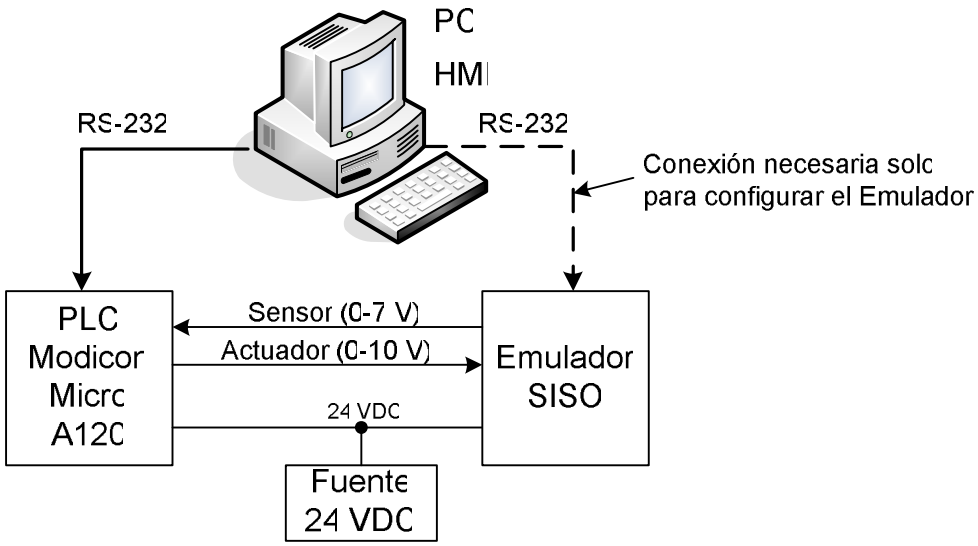


Figura E3 - Diagrama general de la conexión

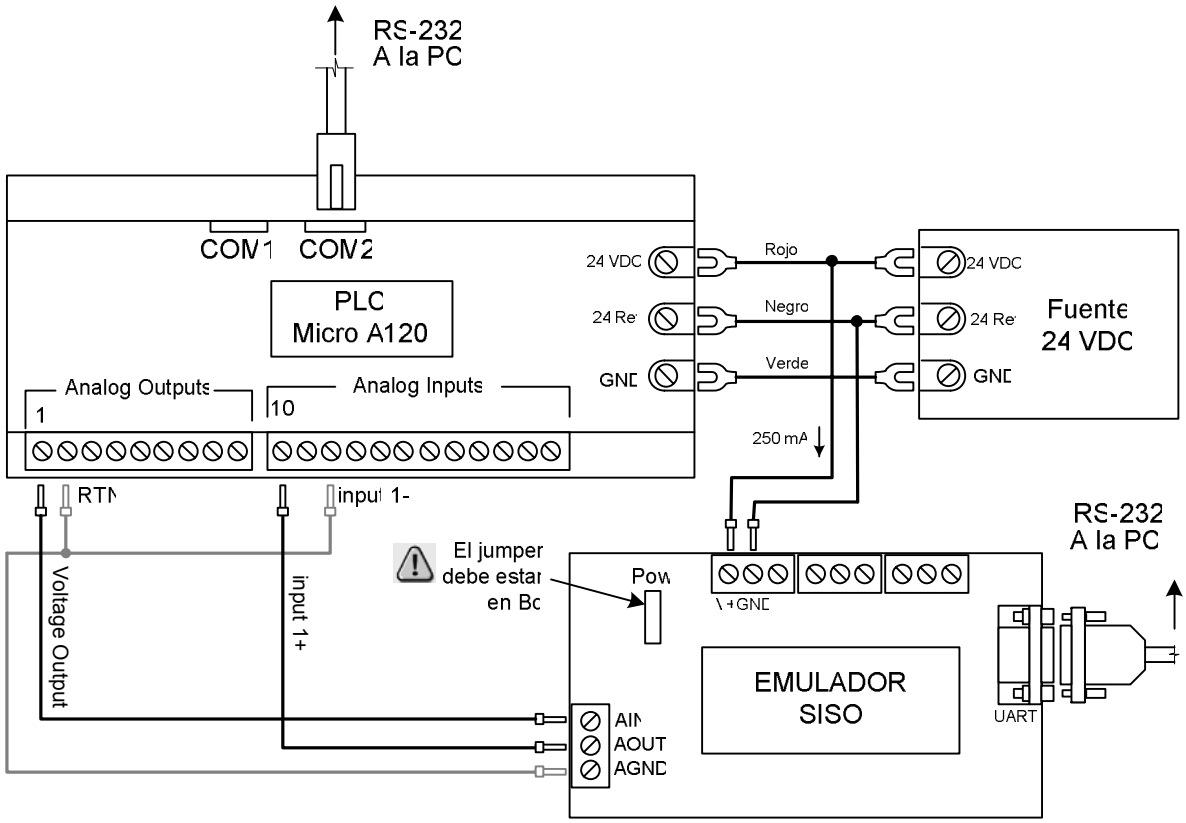


Figura E4 - Conexión de los equipos

6.3 Configuración de la interfase

I - Configurar la GUI del emulador SISO como se muestra en la Figura E5.

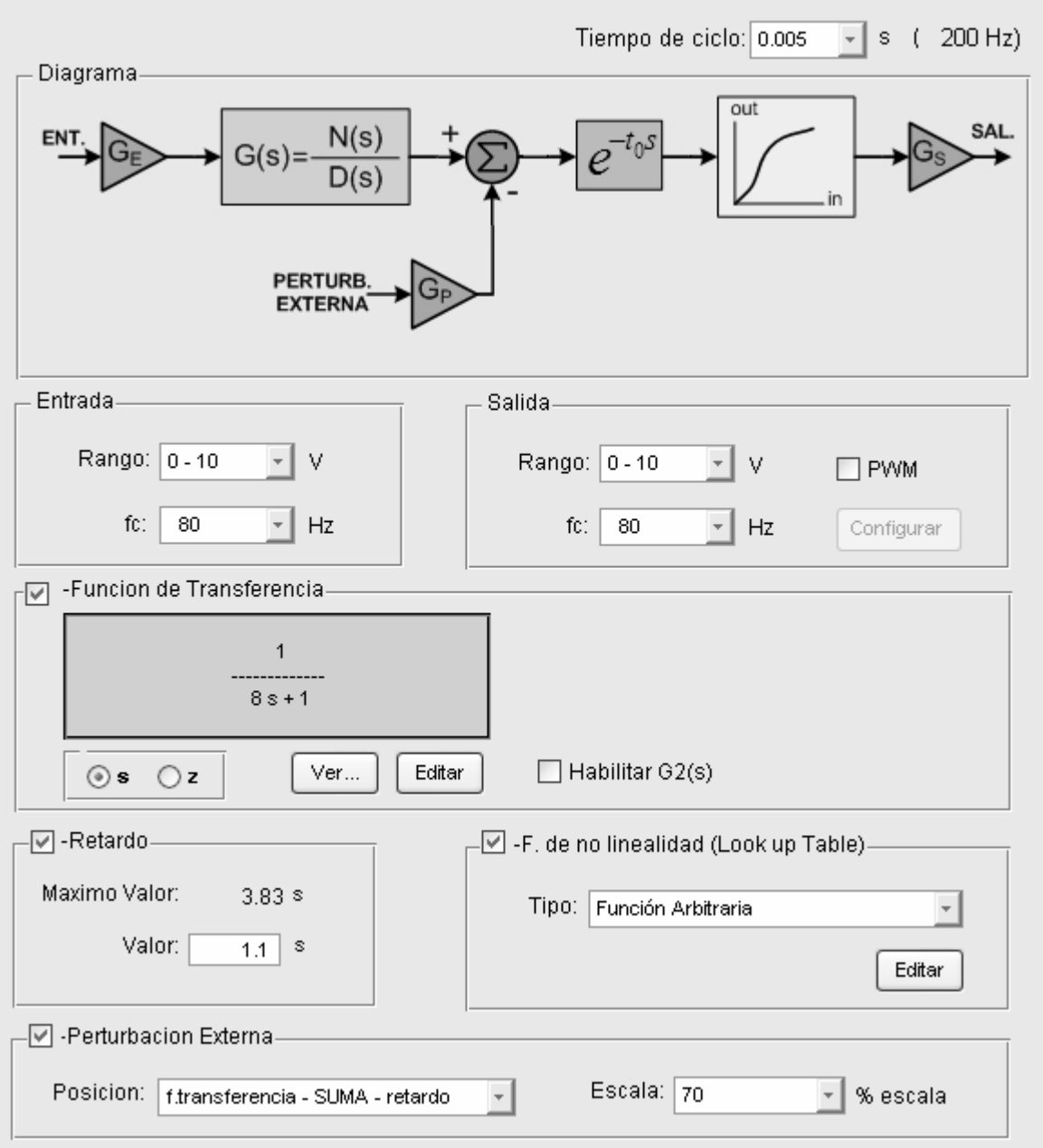


Figura E5 – Configuración de la Interfase para el ejemplo

La manera de encontrar los valores para el Tiempo de ciclo y las frecuencias de los filtros antialias se realiza de la siguiente manera (descrita también en el punto *2.3.4.4 Resumen de la selección de la frecuencia de muestro*):

1 - Se obtiene la función de transferencia en lazo cerrado:

```
>>g = tf(1,[8 1]);
>>glc = feedback(g,1);
>>bode(glc);
```

2 - Se obtienen los valores de ganancia en DC (G_{DC}), ancho de banda (f_{BW} @G_{DC}-3dB) y atenuación en una década (G@f_{BWx10}) a partir del gráfico de Bode:

- G_{DC} (del gráfico E6) = -6.03 dB;
- G_{BW} = GDC - 3dB = -9.03 dB; del gráfico E6 se busca f_{BW} = 0.25 rad/s;
- Se obtiene la ganancia de la función de transferencia en lazo cerrado a una década de la frecuencia de ancho de banda: G = ? en 10xf_{BW} $\rightarrow$ G = -26.1dB.
- La atenuación en una década es por tanto Gf_{BWx10} - Gf_{BW} = -26.1 + 9.03 = -17.07dB.
- Debido a que la variación en una década está comprendida entre -15dB y -30dB, se escoge una frecuencia máxima de simulación (*fm*) de 22 veces f_{BW} $\rightarrow$ *fm* = 5.5rad/s.

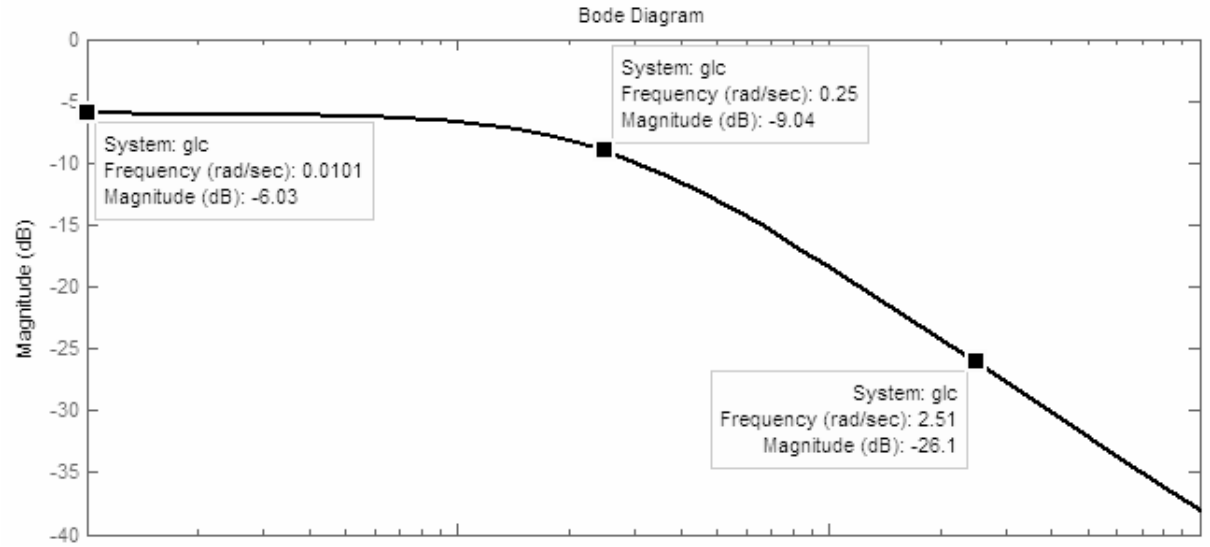


Figura E6 – Diagrama de Bode la F. de Transferencia en lazo cerrado.

3- Conociendo la frecuencia de simulación ahora es necesario determinar la frecuencia de muestreo. Se entiende que la frecuencia de simulación solo está relacionada con la respuesta de la planta en cuestión pero no toma en cuenta los efectos del ADC y el DAC. Por esta razón la frecuencia de muestro tiene que ser

mayor para compensar estos efectos. En la Tabla 2.2 se da el factor de la frecuencia de simulación por el cual se tiene un error determinado en la frecuencia tope. Para el caso de 1% de admisibilidad en el error es necesario que el factor citado sea 12.8 veces. Para este ejemplo si la frecuencia de simulación era 5.5rad/s, la frecuencia de muestro será 70.4 rad/s. Esta es la frecuencia que se debe fijar en Tiempo de ciclo. La frecuencia en rad/s es angular por lo que hay que dividirla para 2π y obtener su inversa para tener el valor de Tiempo de ciclo:

$$Tc = \frac{2\pi}{\omega} = \frac{2\pi}{70.4} = 0.0892 \approx 0.09[s].$$

Como se puede observar en los cálculos, la frecuencia de muestreo es pequeña (fs = 1/0.09 = 11.11Hz) para tener buenos resultados, sin embargo es posible aumentar considerablemente esta frecuencia ya que la restricción que tiene el emulador SISO en cuanto a la frecuencia de simulación viene dada por el retardo de transporte por lo cual se deja que el usuario escoja la frecuencia de simulación. La GUI indica el valor máximo de retardo que se puede tener para una función de transferencia y un tiempo de ciclo dado. Para el caso de ejemplo es posible simular hasta 200Hz pudiendo ingresar el retardo de 3.2s.

La frecuencia de los filtros antialias deberá ser siempre menor a la mitad de la frecuencia de simulación. En este caso se fija la frecuencia en 80Hz.

II- El ingreso de la función arbitraria es sencillo y los pasos para hacerlo son:

- Una vez abierto emusiso, ingresar los siguientes comandos en Matlab:

  ```
  >> x=0:0.1:100;
  >> y=(exp(x/40)-1)*8.9425;
  ```

- Luego, de nuevo en la GUI, ingresar a *Editar* correspondiente a la Función de No-Linealidad para que se abra el cuadro de diálogo *Configuración de no linealidad*. En la lista de selección deben aparecer las variables *x* e *y*. Al seleccionar y se dibuja la curva planteada. Confirmar el cambio y aplastar OK.

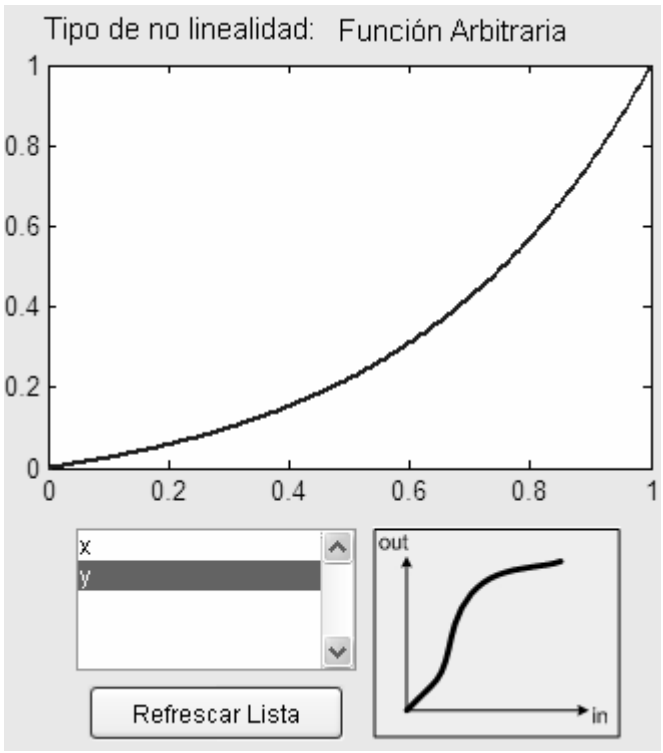


Figura E7 – Configuración de la función de No-linealidad

6.4 Inicio de la práctica

- Una vez terminada la configuración conectarse al emulador SISO siguiendo en el menú: Emulador →Conectar…. Seleccionar el puerto COM al cual está conectado, fijar la velocidad de transmisión en 9600 y presionar OK.

- Si la conexión fue exitosa, seguir Emulador→Descargar Planta.

- Iniciar la HMI que controla al PLC

- Una vez verificada la conexión de la HMI con el PLC, iniciar el proceso de emulación: presionar el botón de Run/Stop en el Emulador SISO o seguir en el menú *Emulador → Iniciar/Parar* y ante la confirmación presionar *Iniciar*.

- Manipular los valores del control PID para sintonizarlo hasta lograr el desempeño del controlador. Introducir perturbaciones a la planta por medio del potenciómetro.

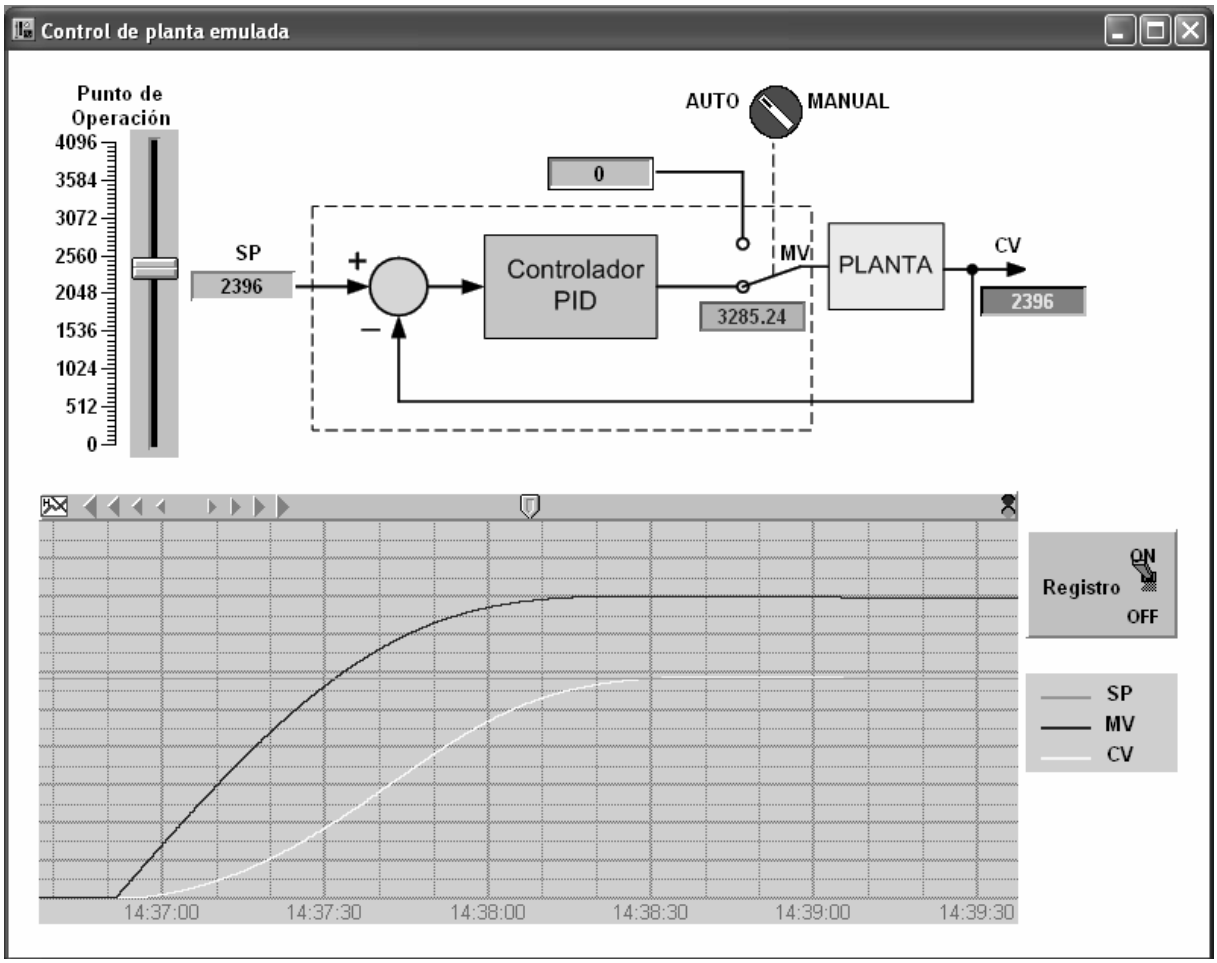


Figura E8 – HMI para el control del proceso

La configuración del control PID de la interfase HMI se fija con los valores de la figura E9.

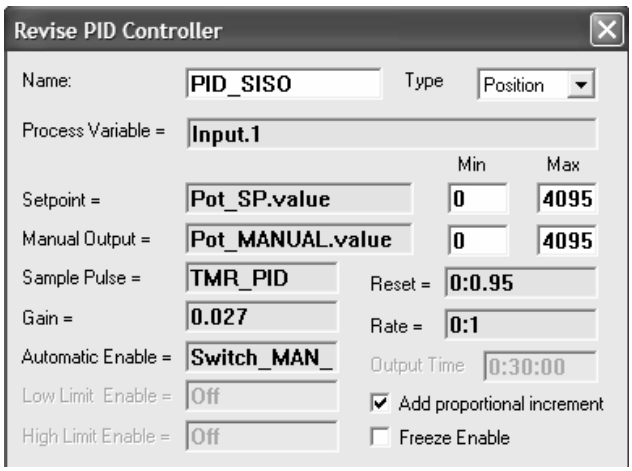
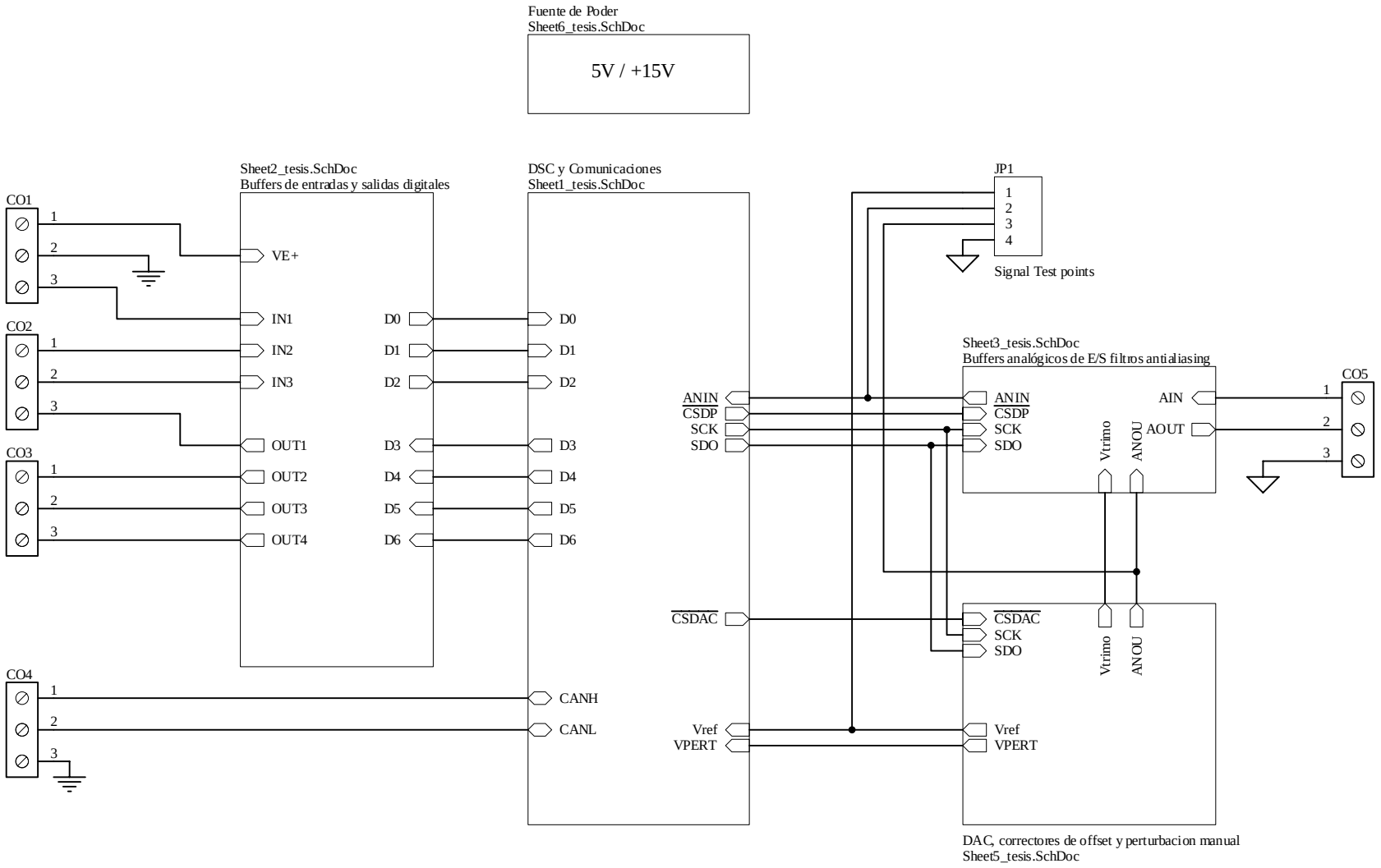


Figura E9 – Configuración del control PID

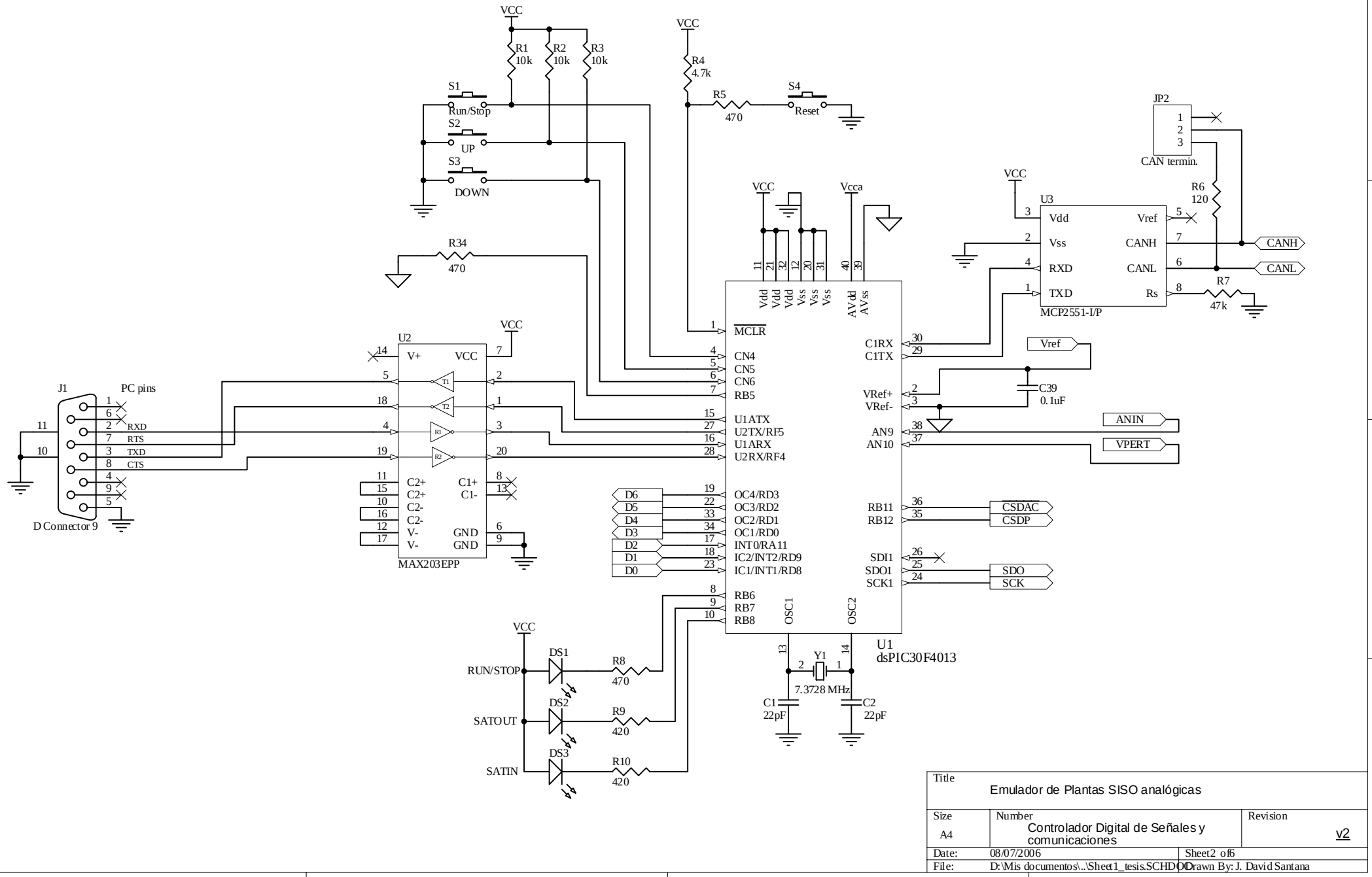
EMULADOR SISO

ANEXO A2

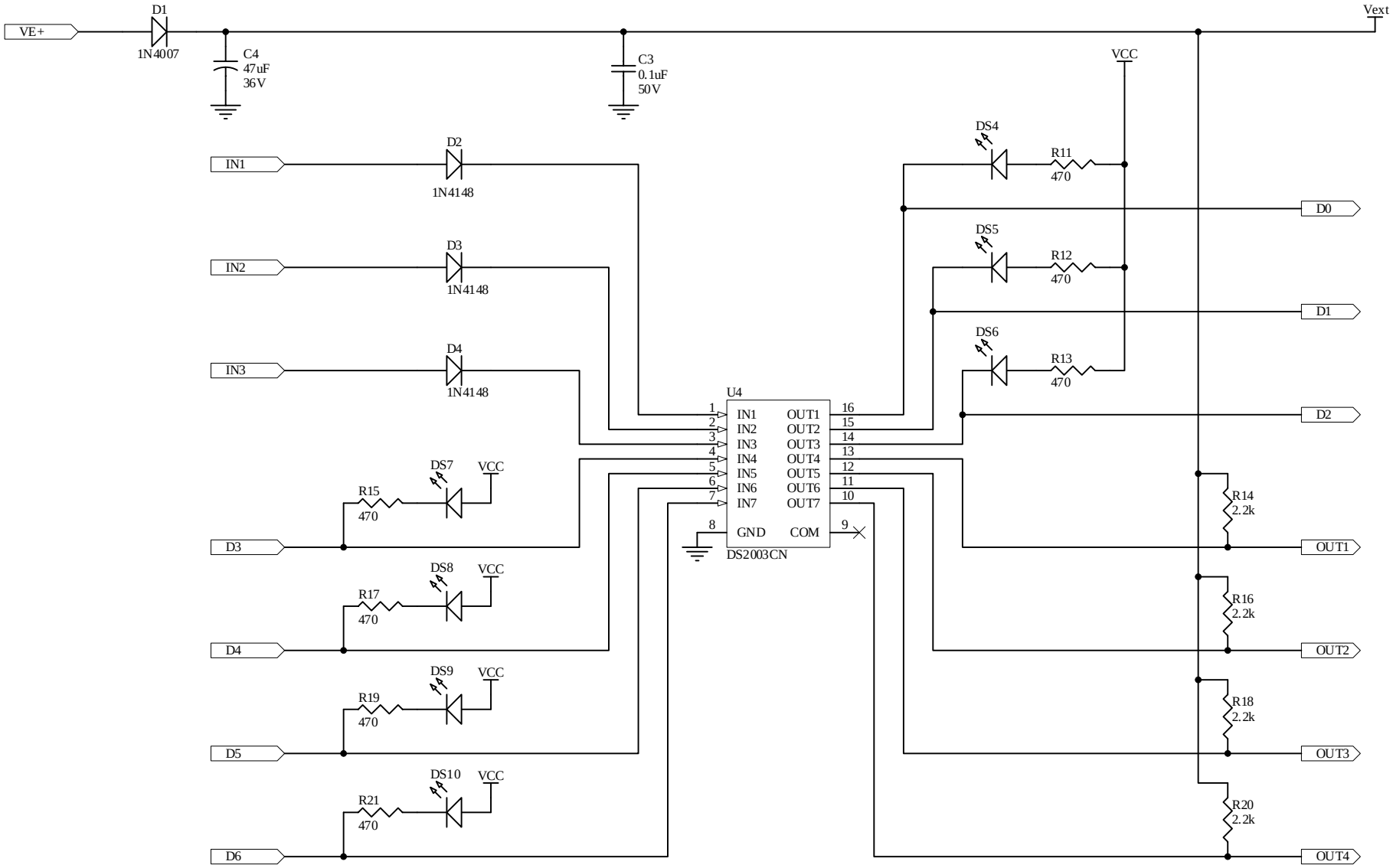
ESQUEMÁTICOS DEL CIRCUITO



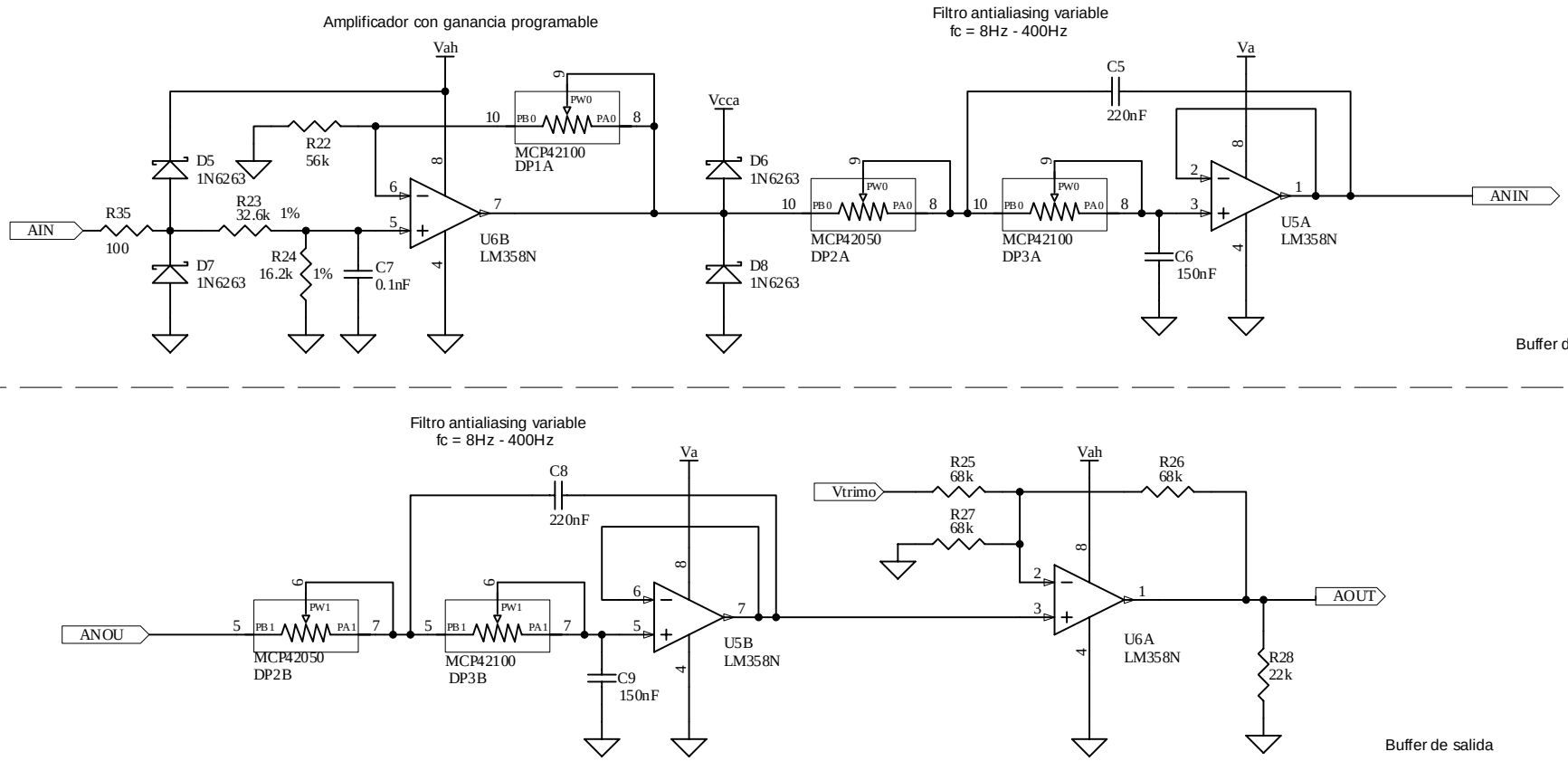
Title	Emulador de Plantas SISO analógicas	
Size: A4	Number: Conectores de salida	Revision:
Date: 08/07/2006		Sheet1 of6
File: D:\Mis documentos\..\Sheet7_tesis.SCHDOC		Drawn By: J. David Santana



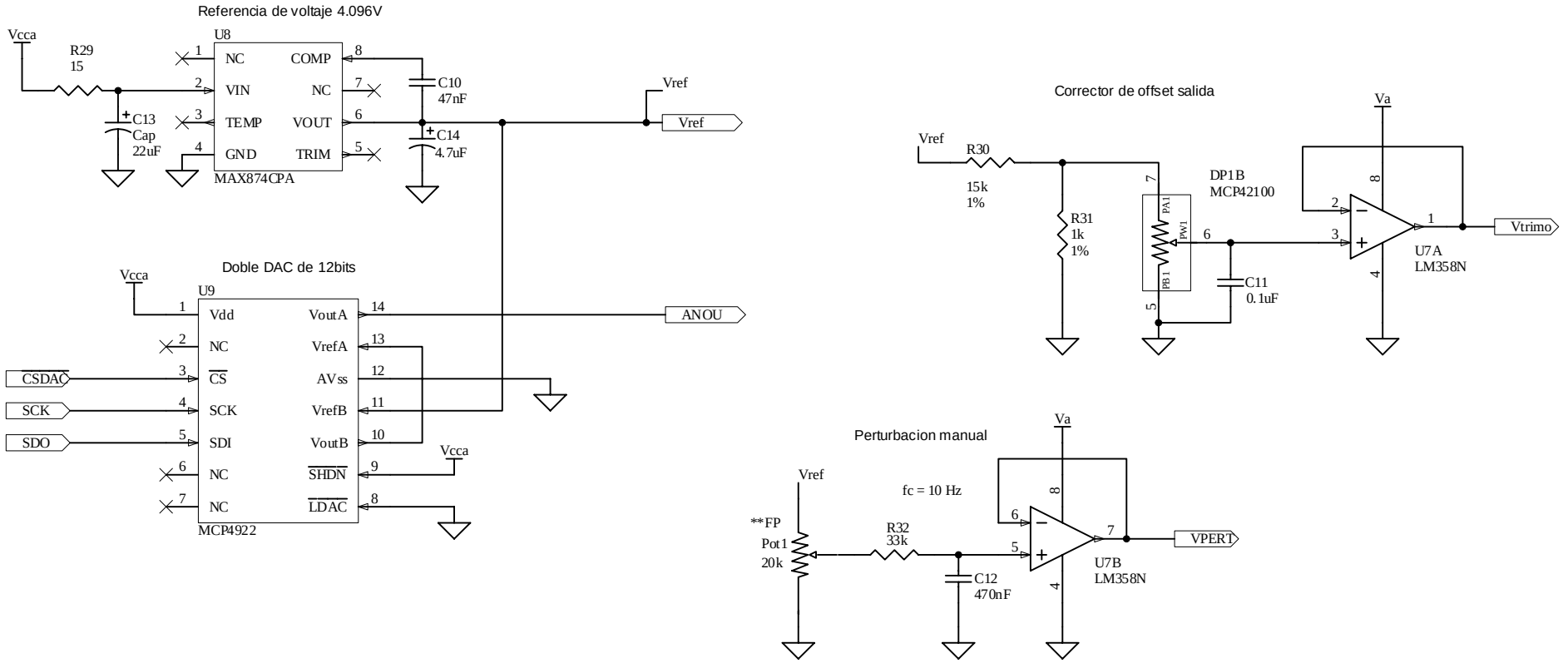
Title	Emulador de Plantas SISO analógicas	
Size: A4	Number: Controlador Digital de Señales y comunicaciones	Revision: v2
Date: 08/07/2006	Sheet2 of6	
File: D:\Mis documentos\..\Sheet1_tesis.SCHDOC	Drawn By: J. David Santana	



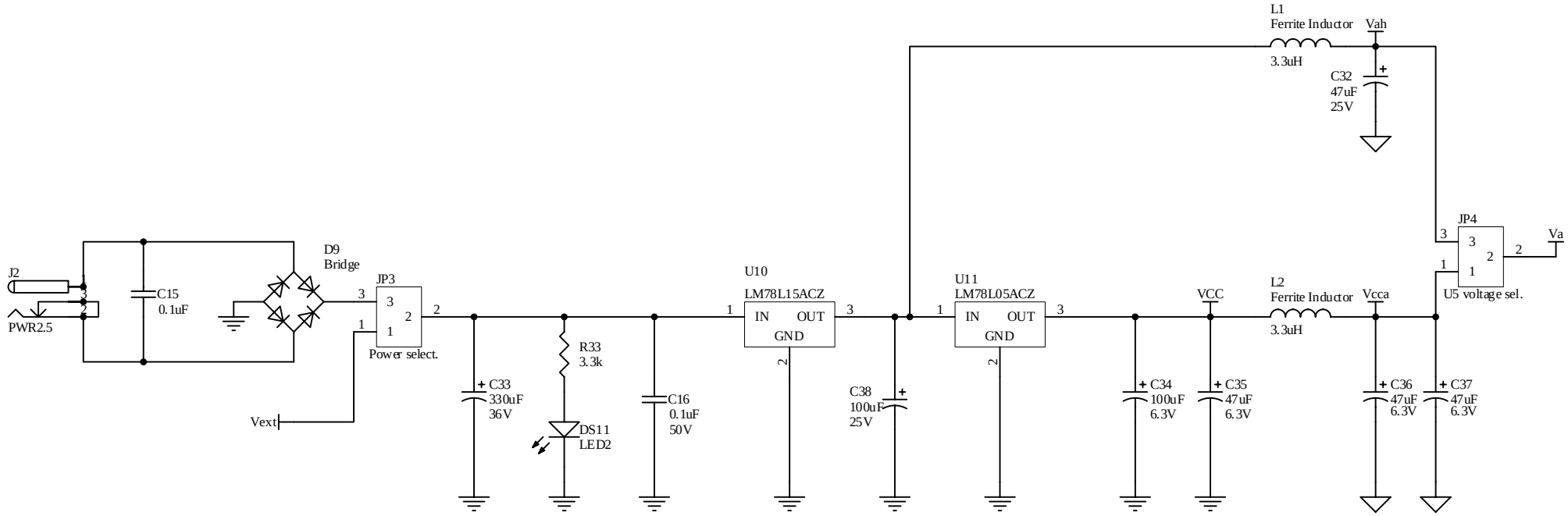
Title	Emulador de Plantas SISO analógicas	
Size: A4	Number: Buffers de entradas y salidas digitales	Revision:
Date: 08/07/2006		Sheet3 of 6
File: D:\Mis documentos\..\Sheet2_tesis.SCHDOC		Drawn By: J. David Santana



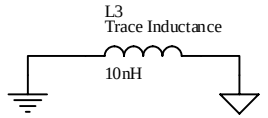
Title	Emulador de Plantas SISO analógicas	
Size: A4	Number: Buffers analógicos de E/S filtros antialiasing	Revision:
Date: 08/07/2006	Sheet 4of 6	
File: D:\Mis documentos\..\Sheet3_tesis.SCHDOC	Drawn By:J. David Santana	



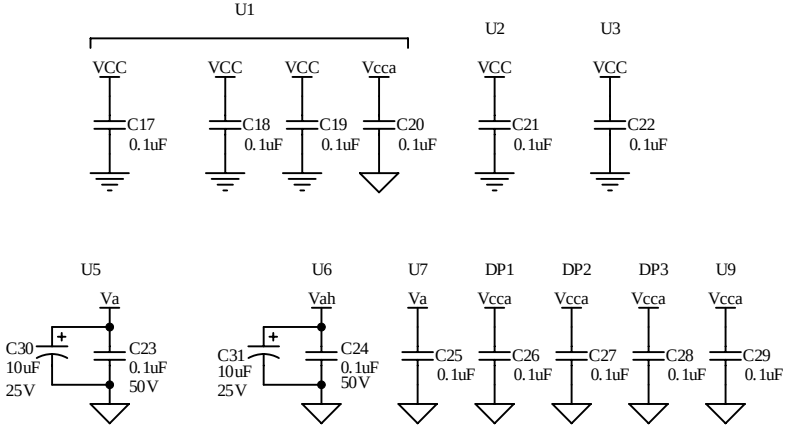
Title	Emulador de Plantas SISO analógicas	
Size: A4	Number: DAC, corrector de offset y perturbación manual	Revision:
Date: 08/07/2006		Sheet 5 of 6
File: D:\Mis documentos\..\Sheet5_tesis.SCHDOC		Drawn By: J. David Santana



Conexión en un solo punto



Bypass caps



Title: Emulador de Plantas SISO analógicas		
Size: A4	Number: Fuente de poder DC	Revision:
Date: 08/07/2006		Sheet6 of6
File: D:\Mis documentos\..\Sheet6_tesis.SCHDOC		Drawn By:J. David Santana

EMULADOR SISO

ANEXO A3

HOJAS TÉCNICAS DE LOS INTEGRADOS

dsPIC30F3014/4013

dsPIC30F3014/4013 High-Performance Digital Signal Controllers

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *dsPIC30F Family Reference Manual* (DS70046). For more information on the device instruction set and programming, refer to the *dsPIC30F Programmer's Reference Manual* (DS70030).

High-Performance Modified RISC CPU:

- Modified Harvard architecture
- C compiler optimized instruction set architecture
- Flexible addressing modes
- 84 base instructions
- 24-bit wide instructions, 16-bit wide data path
- Up to 48 Kbytes on-chip Flash program space
- 2 Kbytes of on-chip data RAM
- 1 Kbyte of non-volatile data EEPROM
- 16 x 16-bit working register array
- Up to 30 MIPs operation:
  - DC to 40 MHz external clock input
  - 4 MHz-10 MHz oscillator input with PLL active (4x, 8x, 16x)
- Up to 33 interrupt sources:
  - 8 user selectable priority levels
  - 3 external interrupt sources
  - 4 processor traps

DSP Features:

- Dual data fetch
- Modulo and Bit-reversed modes
- Two 40-bit wide accumulators with optional saturation logic
- 17-bit x 17-bit single cycle hardware fractional/ integer multiplier
- All DSP instructins are single cycle
  - Multiply-Accumulate (MAC) operation
- Single cycle ±16 shift

Peripheral Features:

- High current sink/source I/O pins: 25 mA/25 mA
- Up to five 16-bit timers/counters; optionally pair up 16-bit timers into 32-bit timer modules
- Up to four 16-bit Capture input functions
- Up to four 16-bit Compare/PWM output functions
- Data Converter Interface (DCI) supports common audio Codec protocols, including I^2S and AC'97
- 3-wire SPI™ module (supports 4 Frame modes)
- I^2C™ module supports Multi-Master/Slave mode and 7-bit/10-bit addressing
- Up to two addressable UART modules with FIFO buffers
- CAN bus module compliant with CAN 2.0B standard

Analog Features:

- 12-bit Analog-to-Digital Converter (A/D) with:
  - 100 Ksps conversion rate
  - Up to 13 input channels
  - Conversion available during Sleep and Idle
- Programmable Low Voltage Detection (PLVD)
- Programmable Brown-out Detection and Reset generation

Special Microcontroller Features:

- Enhanced Flash program memory:
  - 10,000 erase/write cycle (min.) for industrial temperature range, 100K (typical)
- Data EEPROM memory:
  - 100,000 erase/write cycle (min.) for industrial temperature range, 1M (typical)
- Self-reprogrammable under software control
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Flexible Watchdog Timer (WDT) with on-chip low power RC oscillator for reliable operation
- Fail-Safe Clock Monitor operation:
  - Detects clock failure and switches to on-chip low power RC oscillator

dsPIC30F3014/4013

Special Microcontroller Features (Cont.):

- Programmable code protection
- In-Circuit Serial Programming™ (ICSP™)
- Selectable Power Management modes:
  - Sleep, Idle and Alternate Clock modes

CMOS Technology:

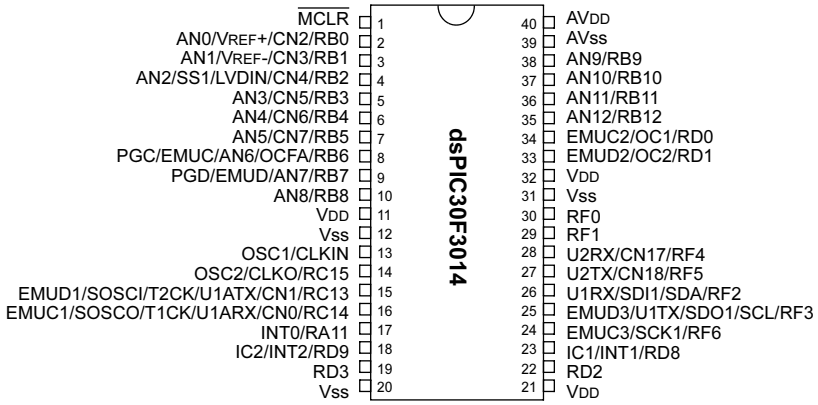
- Low power, high speed Flash technology
- Wide operating voltage range (2.5V to 5.5V)
- Industrial and Extended temperature ranges
- Low power consumption

dsPIC30F3014/4013 Controller Family

Device	Pins	Program Memory		SRAM Bytes	EEPROM Bytes	Timer 16-bit	Input Cap	Output Comp/Std PWM	Codec Interface	A/D 12-bit 100 Ksps	UART	SPI™	I²C™	CAN
		Bytes	Instructions											
dsPIC30F3014	40/44	24K	8K	2048	1024	3	2	2	-	13 ch	2	1	1	0
dsPIC30F4013	40/44	48K	16K	2048	1024	5	4	4	AC'97, I²S	13 ch	2	1	1	1

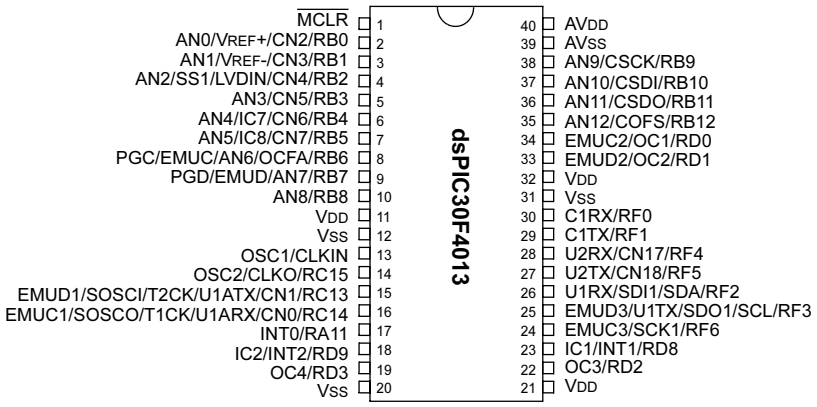
Pin Diagrams

40-Pin PDIP

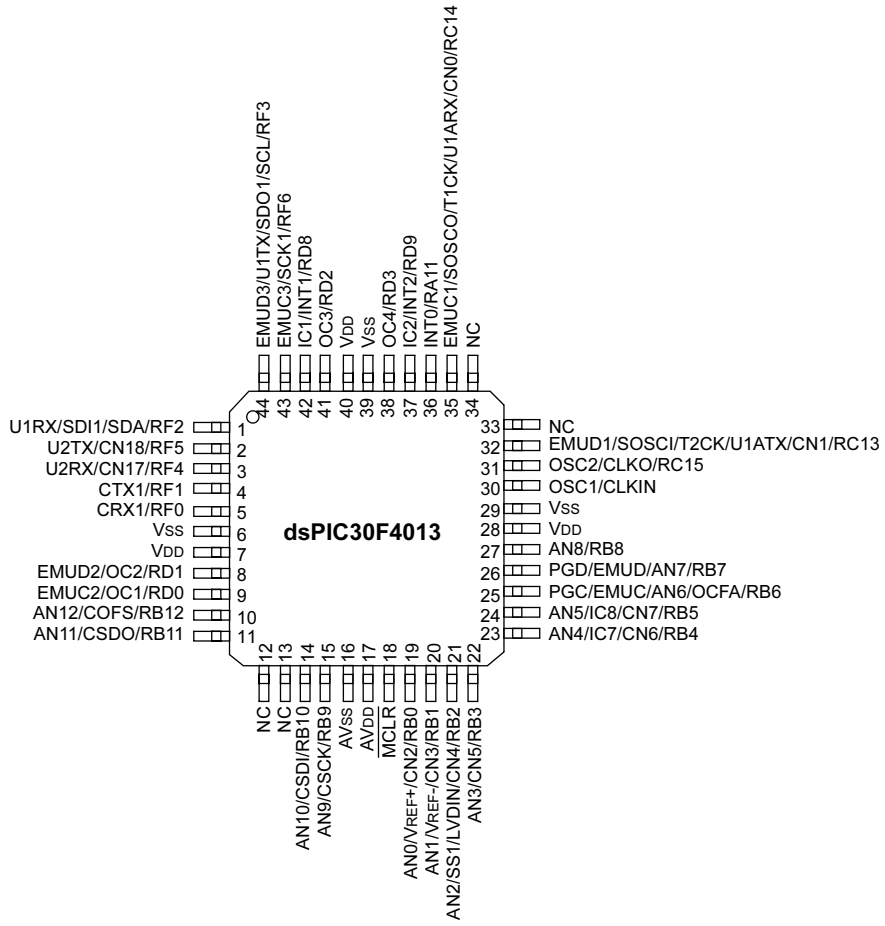


Pin Diagrams (Continued)

40-Pin PDIP



44-Pin TQFP



Note: For descriptions of individual pins, see Section 1.0.

19-0175; Rev 6; 3/05

MAXIM

±15kV ESD-Protected, +5V RS-232 Transceivers

General Description

The MAX202E–MAX213E, MAX232E/MAX241E line drivers/receivers are designed for RS-232 and V.28 communications in harsh environments. Each transmitter output and receiver input is protected against ±15kV electrostatic discharge (ESD) shocks, without latchup. The various combinations of features are outlined in the *Selector Guide.* The drivers and receivers for all ten devices meet all EIA/TIA-232E and CCITT V.28 specifications at data rates up to 120kbps, when loaded in accordance with the EIA/TIA-232E specification.

The MAX211E/MAX213E/MAX241E are available in 28-pin SO packages, as well as a 28-pin SSOP that uses 60% less board space. The MAX202E/MAX232E come in 16-pin TSSOP, narrow SO, wide SO, and DIP packages. The MAX203E comes in a 20-pin DIP/SO package, and needs no external charge-pump capacitors. The MAX205E comes in a 24-pin wide DIP package, and also eliminates external charge-pump capacitors. The MAX206E/MAX207E/MAX208E come in 24-pin SO, SSOP, and narrow DIP packages. The MAX232E/MAX241E operate with four 1µF capacitors, while the MAX202E/MAX206E/MAX207E/MAX208E/MAX211E/MAX213E operate with four 0.1µF capacitors, further reducing cost and board space.

Applications

Notebook, Subnotebook, and Palmtop Computers
Battery-Powered Equipment
Hand-Held Equipment

Next-Generation Device Features

- **For Low-Voltage Applications**
  MAX3222E/MAX3232E/MAX3237E/MAX3241E/MAX3246E: ±15kV ESD-Protected Down to 10nA, +3.0V to +5.5V, Up to 1Mbps, True RS-232 Transceivers (MAX3246E Available in a UCSP™ Package)
- **For Low-Power Applications**
  MAX3221/MAX3223/MAX3243: 1µA Supply Current, True +3V to +5.5V RS-232 Transceivers with Auto-Shutdown™
- **For Space-Constrained Applications**
  MAX3233E/MAX3235E: ±15kV ESD-Protected, 1µA, 250kbps, +3.0V/+5.5V, Dual RS-232 Transceivers with Internal Capacitors
- **For Low-Voltage or Data Cable Applications**
  MAX3380E/MAX3381E: +2.35V to +5.5V, 1µA, 2Tx/2Rx RS-232 Transceivers with ±15kV ESD-Protected I/O and Logic Pins

Ordering Information

PART	TEMP RANGE	PIN-PACKAGE
MAX202ECPE	0°C to +70°C	16 Plastic DIP
MAX202ECSE	0°C to +70°C	16 Narrow SO

Ordering Information continued at end of data sheet.

Pin Configurations and Typical Operating Circuits appear at end of data sheet.

AutoShutdown and UCSP are trademarks of Maxim Integrated Products, Inc.

Selector Guide

PART	NO. OF RS-232 DRIVERS	NO. OF RS-232 RECEIVERS	RECEIVERS ACTIVE IN SHUTDOWN	NO. OF EXTERNAL CAPACITORS (µF)	LOW-POWER SHUTDOWN	TTL TRI-STATE
MAX202E	2	2	0	4 (0.1)	No	No
MAX203E	2	2	0	None	No	No
MAX205E	5	5	0	None	Yes	Yes
MAX206E	4	3	0	4 (0.1)	Yes	Yes
MAX207E	5	3	0	4 (0.1)	No	No
MAX208E	4	4	0	4 (0.1)	No	No
MAX211E	4	5	0	4 (0.1)	Yes	Yes
MAX213E	4	5	2	4 (0.1)	Yes	Yes
MAX232E	2	2	0	4 (1)	No	No
MAX241E	4	5	0	4 (1)	Yes	Yes

MCP2551

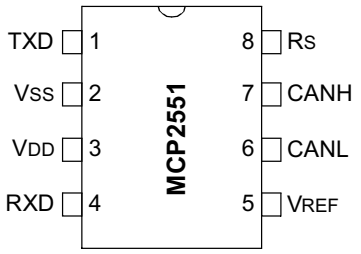
High-Speed CAN Transceiver

Features

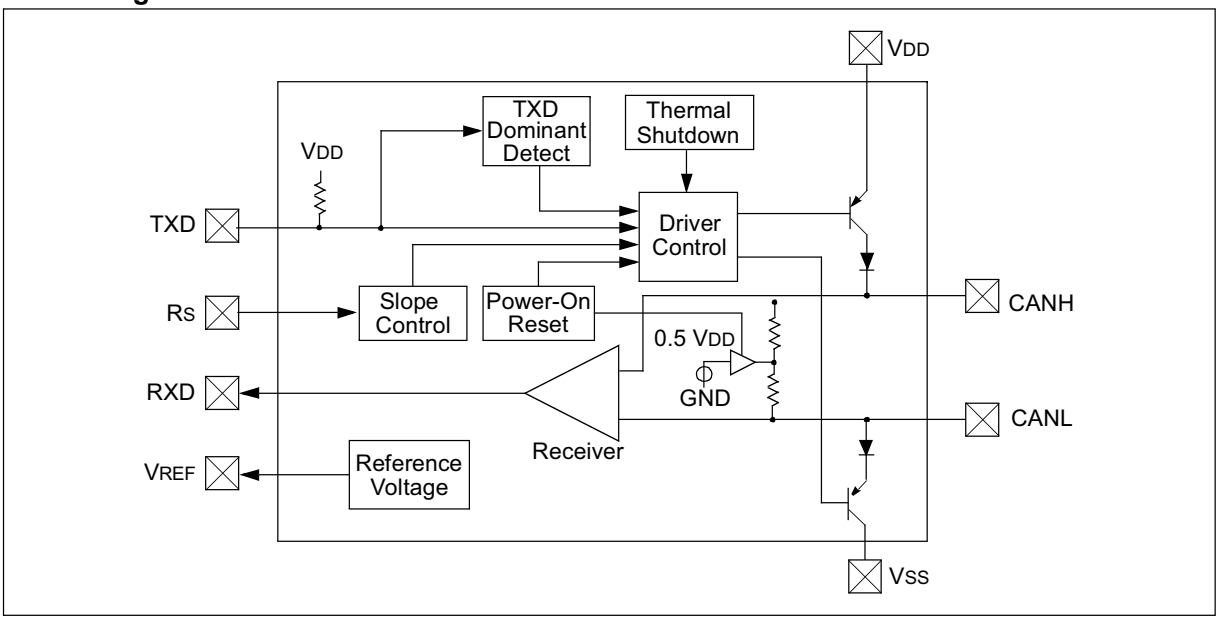
- Supports 1 Mb/s operation
- Implements ISO-11898 standard physical layer requirements
- Suitable for 12V and 24V systems
- Externally-controlled slope for reduced RFI emissions
- Detection of ground fault (permanent dominant) on TXD input
- Power-on reset and voltage brown-out protection
- An unpowered node or brown-out event will not disturb the CAN bus
- Low current standby operation
- Protection against damage due to short-circuit conditions (positive or negative battery voltage)
- Protection against high-voltage transients
- Automatic thermal shutdown protection
- Up to 112 nodes can be connected
- High noise immunity due to differential bus implementation
- Temperature ranges:
  - Industrial (I): -40°C to +85°C
  - Extended (E): -40°C to +125°C

Package Types

PDIP/SOIC



Block Diagram



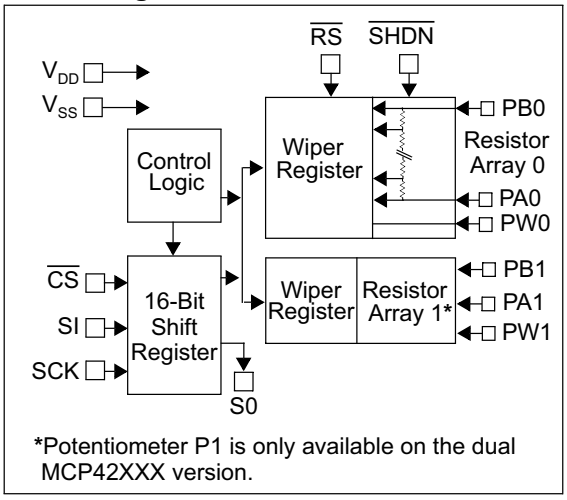
MCP41XXX/42XXX

Single/Dual Digital Potentiometer with SPI™ Interface

Features

- 256 taps for each potentiometer
- Potentiometer values for 10 kΩ, 50 kΩ and 100 kΩ
- Single and dual versions
- SPI™ serial interface (mode 0,0 and 1,1)
- ±1 LSB max INL & DNL
- Low power CMOS technology
- 1 µA maximum supply current in static operation
- Multiple devices can be daisy-chained together (MCP42XXX only)
- Shutdown feature open circuits of all resistors for maximum power savings
- Hardware shutdown pin available on MCP42XXX only
- Single supply operation (2.7V - 5.5V)
- Industrial temperature range: -40°C to +85°C
- Extended temperature range: -40°C to +125°C

Block Diagram



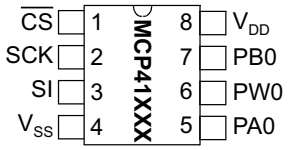
*Potentiometer P1 is only available on the dual MCP42XXX version.

Description

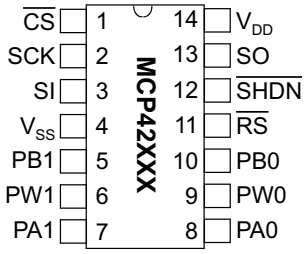
The MCP41XXX and MCP42XXX devices are 256-position, digital potentiometers available in 10 kΩ, 50 kΩ and 100 kΩ resistance versions. The MCP41XXX is a single-channel device and is offered in an 8-pin PDIP or SOIC package. The MCP42XXX contains two independent channels in a 14-pin PDIP, SOIC or TSSOP package. The wiper position of the MCP41XXX/42XXX varies linearly and is controlled via an industry-standard SPI interface. The devices consume <1 µA during static operation. A software shutdown feature is provided that disconnects the "A" terminal from the resistor stack and simultaneously connects the wiper to the "B" terminal. In addition, the dual MCP42XXX has a $\overline{SHDN}$ pin that performs the same function in hardware. During shutdown mode, the contents of the wiper register can be changed and the potentiometer returns from shutdown to the new value. The wiper is reset to the mid-scale position (80h) upon power-up. The $\overline{RS}$ (reset) pin implements a hardware reset and also returns the wiper to mid-scale. The MCP42XXX SPI interface includes both the SI and SO pins, allowing daisy-chaining of multiple devices. Channel-to-channel resistance matching on the MCP42XXX varies by less than 1%. These devices operate from a single 2.7 - 5.5V supply and are specified over the extended and industrial temperature ranges.

Package Types

PDIP/SOIC



PDIP/SOIC/TSSOP



19-0005; Rev 2; 6/97

MAXIM

10µA, Low-Dropout, Precision Voltage References

General Description

The MAX872/MAX874 precision 2.5V and 4.096V micropower voltage references consume a maximum of only 10µA and operate from supply voltages up to 20V. The combination of ultra-low quiescent current and low 200mV dropout makes them ideal for battery-powered equipment. They source and sink up to 500µA with only 200mV input voltage headroom, which makes the 2.5V MAX872 ideal for use with a 3V supply and the 4.096V MAX874 ideal for use with a 5V supply.

Initial accuracy of 0.2% at +25°C (±5mV for the MAX872, ±8mV for the MAX874) and low 40ppm/°C max drift make these references suitable for a wide range of precision applications.

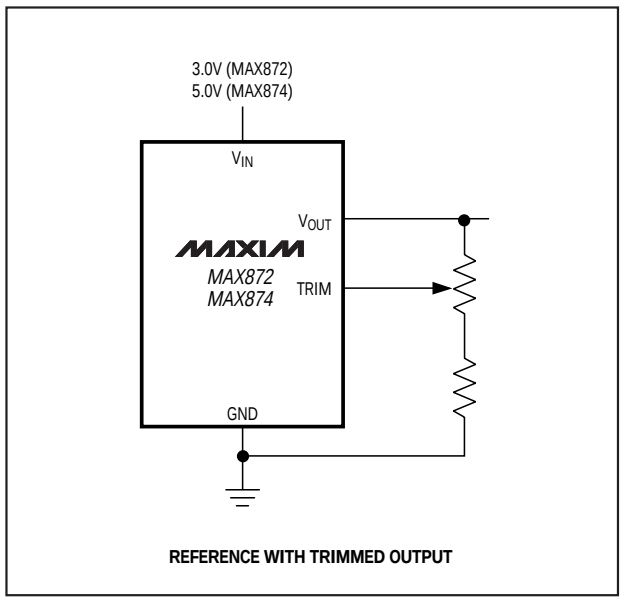
Applications

Hand-Held Instruments

Battery-Operated Equipment

Power Supplies

Typical Operating Circuit



REFERENCE WITH TRIMMED OUTPUT

Features

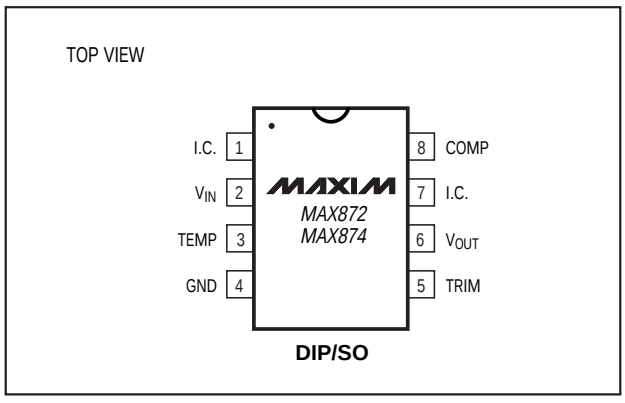
- **Output Voltage**
  2.500V ±0.2% (MAX872)
  4.096V ±0.2% (MAX874)
- **Wide Operating Voltage Range**
  2.7V to 20V (MAX872)
  4.3V to 20V (MAX874)
- **10µA Max Supply Current**
- **40ppm/°C Max Drift Over Extended Temp. Range**
- **Line Regulation Over Temp.**
  20µV/V (MAX872)
  75µV/V (MAX874)
- **Load Regulation Over Temp.**
  0.6mV/mA Max (MAX872)
  1.0mV/mA Max (MAX874)
- **±500µA Sink/Source Current**

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX872CPA	0°C to +70°C	8 Plastic DIP
MAX872CSA	0°C to +70°C	8 SO
MAX872C/D	0°C to +70°C	Dice*
MAX872EPA	-40°C to +85°C	8 Plastic DIP
MAX872ESA	-40°C to +85°C	8 SO
MAX874CPA	0°C to +70°C	8 Plastic DIP
MAX874CSA	0°C to +70°C	8 SO
MAX874C/D	0°C to +70°C	Dice*
MAX874EPA	-40°C to +85°C	8 Plastic DIP
MAX874ESA	-40°C to +85°C	8 SO

** Dice are specified at +25°C only.*

Pin Configuration



DIP/SO

MCP4921/4922

12-Bit DAC with SPI™ Interface

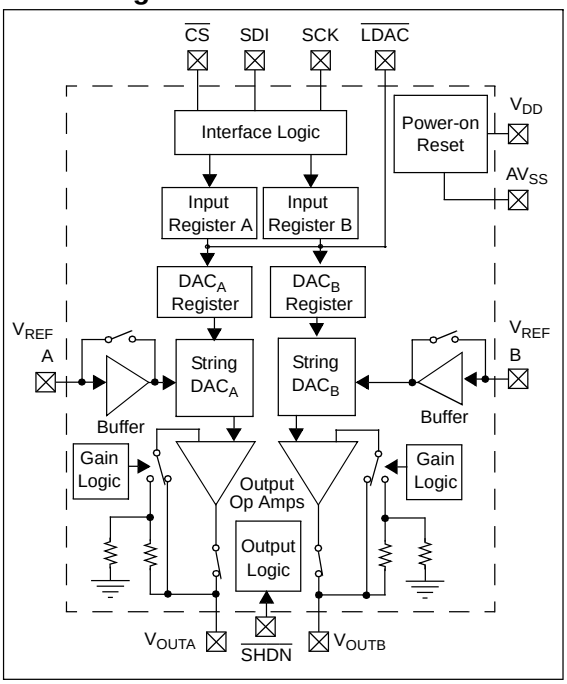
Features

- 12-Bit Resolution
- ±0.2 LSB DNL (typ)
- ±2 LSB INL (typ)
- Single or Dual Channel
- Rail-to-Rail Output
- SPI™ Interface with 20 MHz Clock Support
- Simultaneous Latching of the Dual DACs w/$\overline{\text{LDAC}}$
- Fast Settling Time of 4.5 µs
- Selectable Unity or 2x Gain Output
- 450 kHz Multiplier Mode
- External V_{REF} Input
- 2.7V to 5.5V Single-Supply Operation
- Extended Temperature Range: -40°C to +125°C

Applications

- Set Point or Offset Trimming
- Sensor Calibration
- Digitally-Controlled Multiplier/Divider
- Portable Instrumentation (Battery-Powered)
- Motor Feedback Loop Control

Block Diagram



Description

The Microchip Technology Inc. MCP492X are 2.7 – 5.5V, low-power, low DNL, 12-Bit Digital-to-Analog Converters (DACs) with optional 2x buffered output and SPI interface.

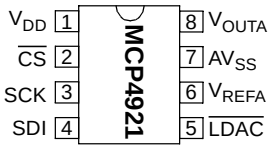
The MCP492X are DACs that provide high accuracy and low noise performance for industrial applications where calibration or compensation of signals (such as temperature, pressure and humidity) are required.

The MCP492X are available in the extended temperature range and PDIP, SOIC, MSOP and TSSOP packages.

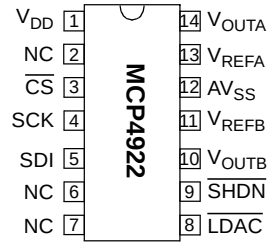
The MCP492X devices utilize a resistive string architecture, with its inherent advantages of low DNL error, low ratio metric temperature coefficient and fast settling time. These devices are specified over the extended temperature range. The MCP492X include double-buffered inputs, allowing simultaneous updates using the $\overline{\text{LDAC}}$ pin. These devices also incorporate a Power-On Reset (POR) circuit to ensure reliable power-up.

Package Types

8-Pin PDIP, SOIC, MSOP



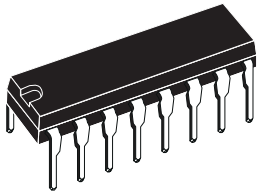
14-Pin PDIP, SOIC, TSSOP



ULN2001A-ULN2002A
ULN2003A-ULN2004A

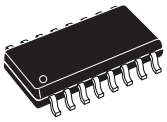
SEVEN DARLINGTON ARRAYS

- SEVEN DARLINGTONS PER PACKAGE
- OUTPUT CURRENT 500mA PER DRIVER (600mA PEAK)
- OUTPUT VOLTAGE 50V
- INTEGRATED SUPPRESSION DIODES FOR INDUCTIVE LOADS
- OUTPUTS CAN BE PARALLELED FOR HIGHER CURRENT
- TTL/CMOS/PMOS/DTL COMPATIBLE INPUTS
- INPUTS PINNED OPPOSITE OUTPUTS TO SIMPLIFY LAYOUT



DIP16

ORDERING NUMBERS: ULN2001A/2A/3A/4A



SO16

ORDERING NUMBERS: ULN2001D/2D/3D/4D

DESCRIPTION

The ULN2001A, ULN2002A, ULN2003 and ULN2004A are high voltage, high current darlington arrays each containing seven open collector darlington pairs with common emitters. Each channel rated at 500mA and can withstand peak currents of 600mA. Suppression diodes are included for inductive load driving and the inputs are pinned opposite the outputs to simplify board layout.

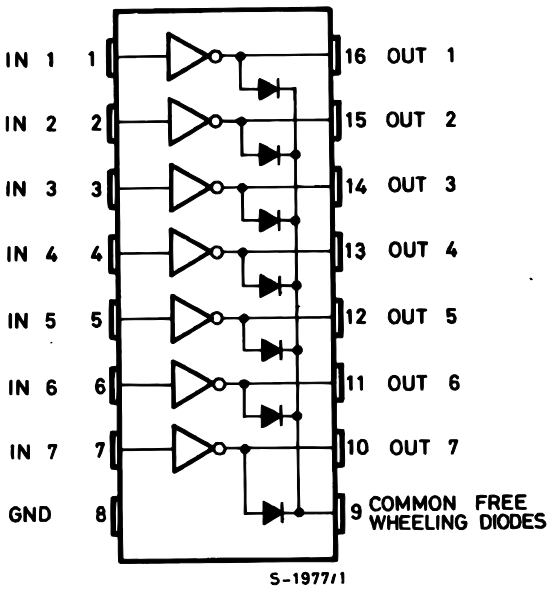
The four versions interface to all common logic families :

ULN2001A	General Purpose, DTL, TTL, PMOS, CMOS
ULN2002A	14-25V PMOS
ULN2003A	5V TTL, CMOS
ULN2004A	6–15V CMOS, PMOS

These versatile devices are useful for driving a wide range of loads including solenoids, relays DC motors, LED displays filament lamps, thermal printheads and high power buffers.

The ULN2001A/2002A/2003A and 2004A are supplied in 16 pin plastic DIP packages with a copper leadframe to reduce thermal resistance. They are available also in small outline package (SO-16) as ULN2001D/2002D/2003D/2004D.

PIN CONNECTION



LM158,A-LM258,A
LM358,A

LOW POWER DUAL OPERATIONAL AMPLIFIERS

- INTERNALLY FREQUENCY COMPENSATED
- LARGE DC VOLTAGE GAIN: 100dB
- WIDE BANDWIDTH (unity gain): 1.1MHz (temperature compensated)
- VERY LOW SUPPLY CURRENT/OP (500µA) ESSENTIALLY INDEPENDENT OF SUPPLY VOLTAGE
- LOW INPUT BIAS CURRENT: 20nA (temperature compensated)
- LOW INPUT OFFSET VOLTAGE: 2mV
- LOW INPUT OFFSET CURRENT: 2nA
- INPUT COMMON-MODE VOLTAGE RANGE INCLUDES GROUND
- DIFFERENTIAL INPUT VOLTAGE RANGE EQUAL TO THE POWER SUPPLY VOLTAGE
- LARGE OUTPUT VOLTAGE SWING 0V TO (Vcc - 1.5V)

N
DIP8
(Plastic Package**)**

D & S
SO8 & miniSO8
(Plastic Micropackage**)**

P
TSSOP8
(Thin Shrink Small Outline Package)

DESCRIPTION

These circuits consist of two independent, high gain, internally frequency compensated which were designed specifically to operate from a single power supply over a wide range of voltages. The low power supply drain is independent of the magnitude of the power supply voltage.

Application areas include transducer amplifiers, dc gain blocks and all the conventional op-amp circuits which now can be more easily implemented in single power supply systems. For example, these circuits can be directly supplied with the standard +5V which is used in logic systems and will easily provide the required interface electronics without requiring any additional power supply.

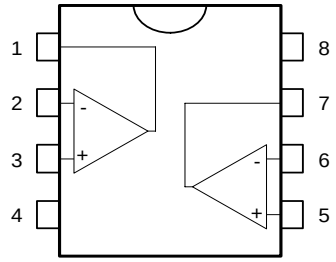
Inthe linear mode the input common-mode voltage range includes ground and the output voltage can also swing to ground, even though operated from only a single power supply voltage.

ORDER CODE

Part Number	Temperature Range	Package			
		N	S	D	P
LM158,A	-55°C, +125°C	•		•	•
LM258,A	-40°C, +105°C	•		•	•
LM358,A	0°C, +70°C	•	•	•	•
Example : LM258N					

N = Dual in Line Package (DIP)
D = Small Outline Package (SO) - also available in Tape & Reel (DT)
S = Small Outline Package (miniSO) only available in Tape & Reel (DT)
P = Thin Shrink Small Outline Package (TSSOP) - only available in Tape &Reel (PT)

PIN CONNECTIONS (top view)



1 - Output 1
2 - Inverting input
3 - Non-inverting input
4 - V_{CC}^-
5 - Non-inverting input 2
6 - Inverting input 2
7 - Output 2
8 - V_{CC}^+

MCP6021/2/3/4

Rail-to-Rail Input/Output, 10 MHz Op Amps

Features

- Rail-to-Rail Input/Output
- Wide Bandwidth: 10 MHz (typ.)
- Low Noise: 8.7 nV/√Hz, at 10 kHz (typ.)
- Low Offset Voltage:
  - Industrial Temperature: ±500 µV (max.)
  - Extended Temperature: ±250 µV (max.)
- Mid-Supply V_{REF}: MCP6021 and MCP6023
- Low Supply Current: 1 mA (typ.)
- Total Harmonic Distortion: 0.00053% (typ., G = 1)
- Unity Gain Stable
- Power Supply Range: 2.5V to 5.5V
- Temperature Range:
  - Industrial: -40°C to +85°C
  - Extended: -40°C to +125°C

Typical Applications

- Automotive
- Driving A/D Converters
- Multi-Pole Active Filters
- Barcode Scanners
- Audio Processing
- Communications
- DAC Buffer
- Test Equipment
- Medical Instrumentation

Available Tools

- SPICE Macro Model (at www.microchip.com)
- FilterLab® software (at www.microchip.com)

Description

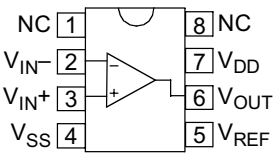
The MCP6021, MCP6022, MCP6023 and MCP6024 from Microchip Technology Inc. are rail-to-rail input and output op amps with high performance. Key specifications include: wide bandwidth (10 MHz), low noise (8.7 nV/√Hz), low input offset voltage and low distortion (0.00053% THD+N). These features make these op amps well suited for applications requiring high performance and bandwidth. The MCP6023 also offers a chip select pin ($\overline{CS}$) that gives power savings when the part is not in use.

The single MCP6021, single MCP6023 and dual MCP6022 are available in standard 8-lead PDIP, SOIC and TSSOP. The quad MCP6024 is offered in 14-lead PDIP, SOIC and TSSOP packages.

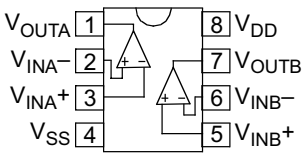
The MCP6021/2/3/4 family is available in the Industrial and Extended temperature ranges. It has a power supply range of 2.5V to 5.5V.

PACKAGE TYPES

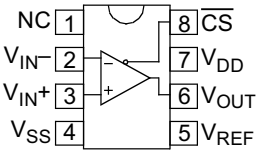
MCP6021
PDIP SOIC, TSSOP



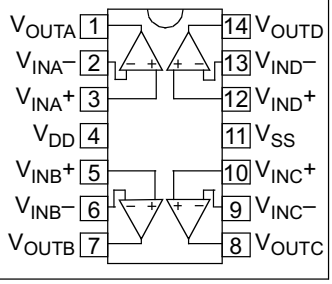
MCP6022
PDIP SOIC, TSSOP



MCP6023
PDIP SOIC, TSSOP



MCP6024
PDIP SOIC, TSSOP



TC913A/TC913B

Dual Auto-Zeroed Operational Amplifiers

Features

- First Monolithic Dual Auto-Zeroed Operational Amplifier
- Chopper Amplifier Performance Without External Capacitors
  - V_{OS}: 15µV Max
  - V_{OS}: Drift; 0.15µV/°C Max
  - Saves Cost of External Capacitors
- SOIC Packages Available
- High DC Gain; 120dB
- Low Supply Current; 650µA
- Low Input Voltage Noise
  - 0.65µV_{P-P} (0.1Hz to 10Hz)
- Wide Common Mode Voltage Range
  - V_{SS} to V_{DD} - 2V
- High Common Mode Rejection; 116dB
- Dual or Single Supply Operation:
  - ±3.3V to ±8.3V
  - +6.5V to +16V
- Excellent AC Operating Characteristics
  - Slew Rate; 2.5V/µsec
  - Unity-Gain Bandwidth; 1.5MHz
- Pin Compatible with LM358, OP-14, MC1458, ICL7621, TL082, TLC322

Applications

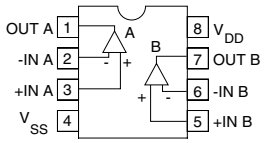
- Instrumentation
- Medical Instrumentation
- Embedded Control
- Temperature Sensor Amplifier
- Strain Gage Amplifier

Device Selection Table

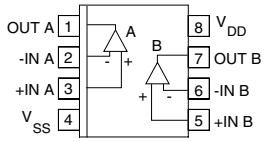
Part Number	Package	Temp. Range	Offset Voltage
TC913ACOA	8-Pin SOIC	0°C to +70°C	15µV
TC913ACPA	8-Pin PDIP	0°C to +70°C	15µV
TC913BCOA	8-Pin SOIC	0°C to +70°C	30µV
TC913BCPA	8-Pin PDIP	0°C to +70°C	30µV

Package Type

PDIP
TC913ACPA
TC913BCPA



SOIC
TC913ACOA
TC913BCOA



General Description

The TC913 is the world's first complete monolithic, dual auto-zeroed operational amplifier. The TC913 sets a new standard for low power, precision dual-operational amplifiers. Chopper-stabilized or auto-zeroed amplifiers offer low offset voltage errors by periodically sampling offset error, and storing correction voltages on capacitors. Previous single amplifier designs required two user-supplied, external 0.1µF error storage correction capacitors — much too large for on-chip integration. The unique TC913 architecture requires smaller capacitors, making on-chip integration possible. Microvolt offset levels are achieved and **external capacitors are not required.**

The TC913 system benefits are apparent when contrasted with a TC7650 chopper amplifier circuit implementation. A single TC913 replaces two TC7650's and four capacitors. Five components and assembly steps are eliminated.

The TC913 pinout matches many popular dual-operational amplifiers: OP-04, TLC322, LM358, and ICL7621 are typical examples. In many applications, operating from dual 5V power supplies or single supplies, the TC913 offers superior electrical performance, and can be a functional drop-in replacement; printed circuit board rework is not necessary. The TC913's low offset voltage error eliminates offset voltage trim potentiometers often needed with bipolar and low accuracy CMOS operational amplifiers.

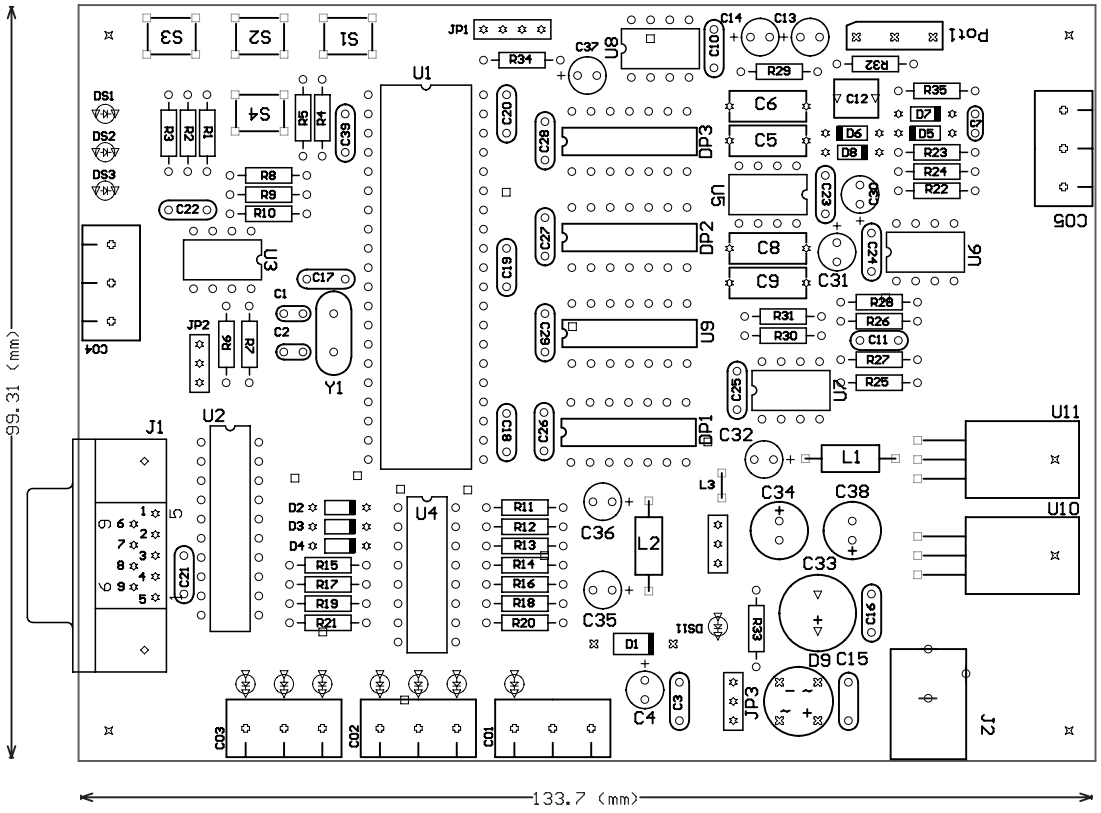
The TC913 takes full advantage of Microchip's proprietary CMOS technology. Unity gain bandwidth is 1.5MHz and slew rate is 2.5V/µsec.

EMULADOR SISO

ANEXO A4

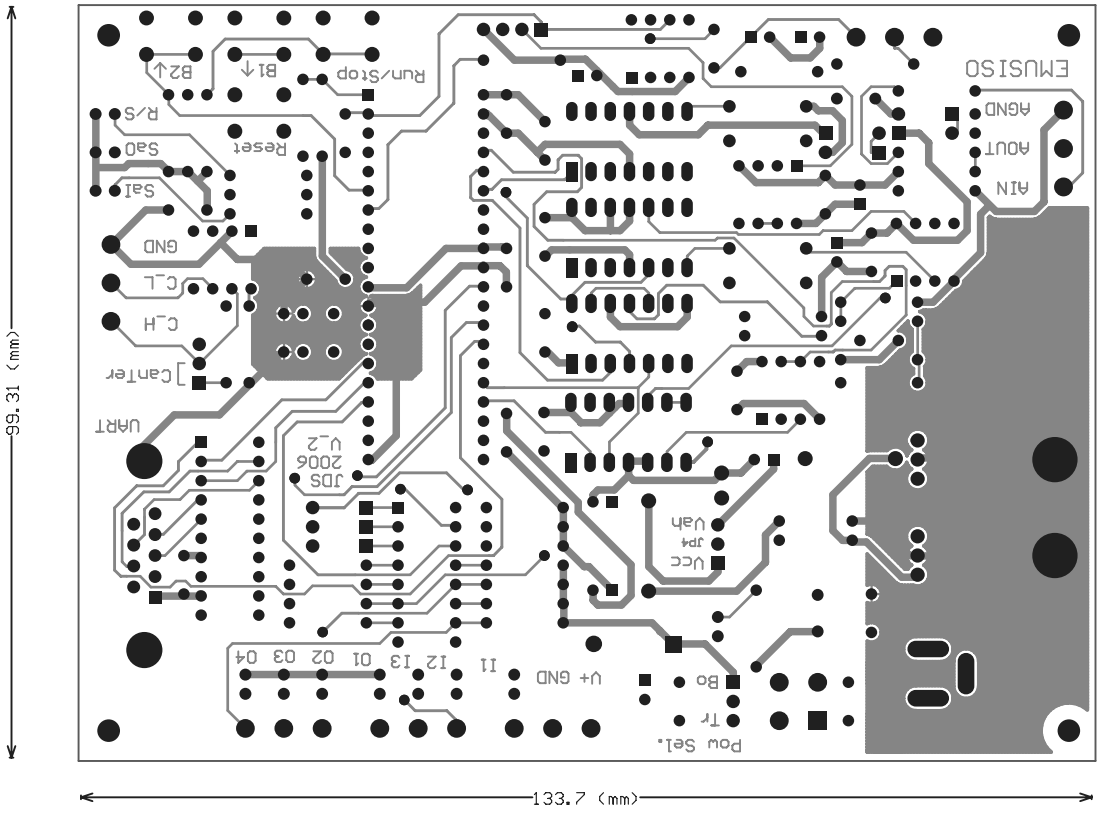
DIAGRAMAS DEL CIRCUITO IMPRESO
(PLACA PCB)

ANEXO A4.1



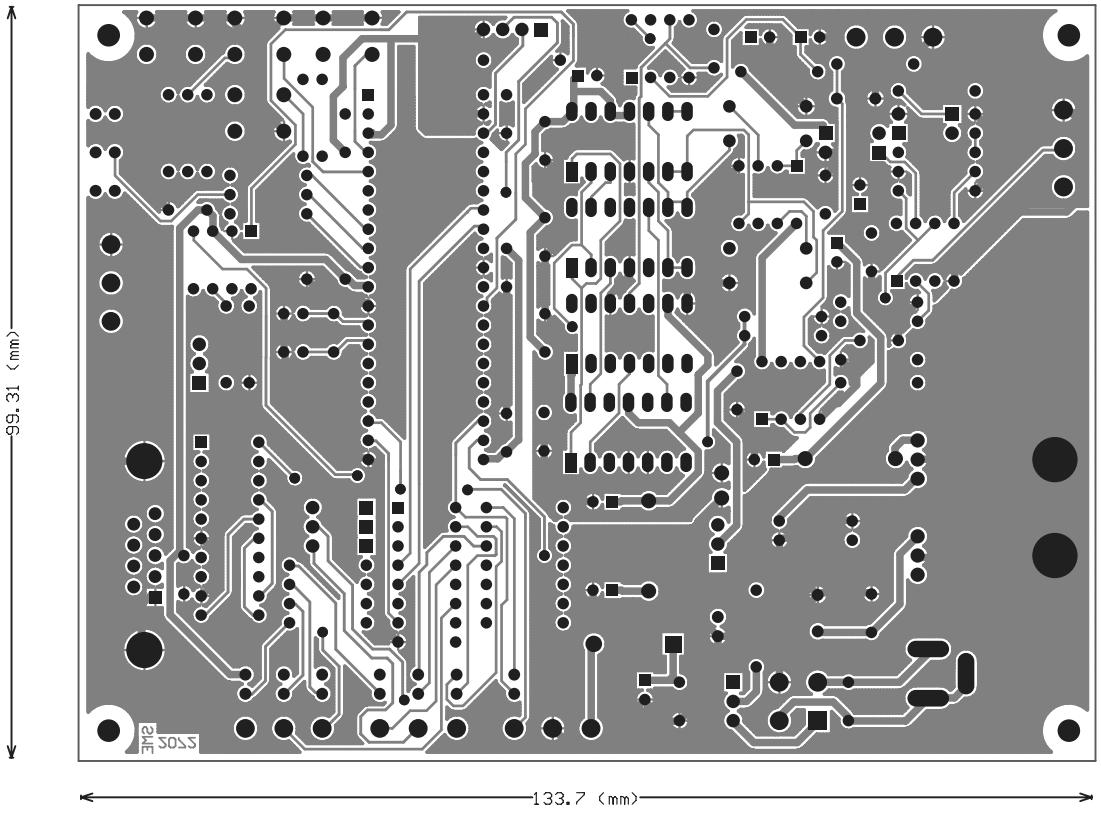
EMULADOR SISO
Ubicación de componentes

ANEXO A4.2



EMULADOR SISO
Lado Superior

ANEXO A4.3



EMULADOR SISO
Lado Inferior

EMULADOR SISO

ANEXO A5

LISTA DE MATERIALES, CANTIDADES Y PRECIOS

ANEXO A5

LISTA DE MATERIALES, CANTIDADES Y PRECIOS UNITARIOS

Descripción	Etiqueta	Designador	Valor	Tol.	Capacidad de voltaje	Cantidad	Precio Unitario	Costo	Comentari
1 Amp General Purpose Rectifier	1N4007	D1				1	0.08	0.08	C
3-Terminal Positive Regulator	LM78L05ACZ, LM78L15ACZ	U10, U11				2	0.50	1.00	C
4.096V Presicion Voltage Reference	MAX874CPA	U8				1	2.66	2.66	**
+5V RS-232 Transceiver with 0.1µF Internal Capacitor	MAX203EPP	U2				1	4.46	4.46	**
12 bit Dual DAC	MCP4922	U9				1	2.23	2.23	*
30MIPS Digital Signal Controller	dsPIC30F4013	U1				1	7.21	7.21	*
256 steps 50k Digital Potentiometer	MCP42050	DP2	50k			1	1.71	1.71	*
256 steps 100k Digital Potentiometer	MCP42100	DP1, DP3	100k			2	1.71	3.42	*
CAN Transceiver	MCP2551-I/P	U3				1	0.93	0.93	*
Capacitor	Cap	C1, C2	22pF			2	0.08	0.16	LT
Capacitor	Cap	C10	47nF			1	0.10	0.10	LT
Capacitor	Cap	C12	470nF			1	0.63	0.63	LT
Capacitor	Cap	C3, C11, C15, C16, C17, C18, C19, C20, C21, C22, C23, C24, C25, C26, C27, C28, C29, C39	0.1uF		50V	18	0.06	1.08	LT
Capacitor	Cap	C5, C8	220nF			2	0.15	0.30	LT
Capacitor	Cap	C6, C9	150nF			2	0.15	0.30	LT
Capacitor	Cap	C7	0.1nF		50V	1	0.20	0.20	LT
Capacitor	Elec	C4	47uF		36V	1	0.06	0.06	LT
Conector 3 polos	Conector	CO1, CO2, CO3, CO4, CO5				5	0.50	2.50	LT
Crystal Oscillator	7.3728 MHz	Y1				1	1.00	1.00	s
Fast Schottky Rectifier	1N6263	D5, D6, D7, D8				4	0.20	0.80	C-TP
Full Wave Diode Bridge	Bridge	D9				1	0.40	0.40	C-TP
Header, 2-Pin	CAN termin.	JP2				1	0.10	0.10	s
Header, 3-Pin	Power select., U5 voltage sel.	JP3, JP4				2	0.10	0.20	s
Header, 4-Pin	Signal Test points	JP1				1	0.10	0.10	s

Descripción	Etiqueta	Designador	Valor	Tol.	Capacidad de voltaje	Cantidad	Precio Unitario	Costo	Comentario
High Conductance Fast Diode	1N4148	D2, D3, D4				3	0.06	0.18	LT
High-Current/Voltage Darlington Driver	DS2003CN	U4				1	0.40	0.40	C-TP
Inductor	Ferrite Inductor	L1, L2	3.3uH			2	0.20	0.40	s
Inductor	Trace Inductance	L3	10nH			1	0.20	0.20	s
Low Voltage Power Supply Connector	PWR2.5	J2				1	0.25	0.25	C
Low-Power Dual Operational Amplifier	LM358N	U5, U6, U7				3	0.55	1.65	ES
Polarized Capacitor (Axial)	Cap	C13	22uF			1	0.08	0.08	LT
Polarized Capacitor (Axial)	Cap	C30, C31	10uF		25V	2	0.06	0.12	LT
Polarized Capacitor (Axial)	Elec	C14	4.7uF			1	0.06	0.06	LT
Polarized Capacitor (Axial)	Elec	C32, C35, C36, C37	47uF		6.3V, 25V	4	0.06	0.24	LT
Polarized Capacitor (Axial)	Elec	C33	330uF		36V	1	0.13	0.13	LT
Polarized Capacitor (Axial)	Elec	C34, C38	100uF		6.3V, 25V	2	0.10	0.20	LT
Potenciómetro	Plain potentiometer	Pot1	20k			1	0.28	0.28	LT
Push button	DOWN, Reset, Run/Stop, UP	S1, S2, S3, S4				4	0.20	0.80	C
Receptacle Assembly, 9 Position, Right Angle	D Connector 9	J1				1	0.36	0.36	LT
Resistor	Res3	R1, R2, R3	10k	5%		3	0.02	0.06	LT
Resistor	Res3	R14, R16, R18, R20	2.2k	5%		4	0.02	0.08	LT
Resistor	Res3	R22	56k	5%		1	0.02	0.02	LT
Resistor	Res3	R23	32.6k	1%		1	0.02	0.02	LT
Resistor	Res3	R24	16.2k	1%		1	0.02	0.02	LT
Resistor	Res3	R25, R26, R27	68k	5%		3	0.02	0.06	LT
Resistor	Res3	R28	22k	5%		1	0.02	0.02	LT
Resistor	Res3	R29	15	5%		1	0.02	0.02	LT
Resistor	Res3	R30	15k	1%		1	0.02	0.02	LT
Resistor	Res3	R31	1k	1%		1	0.02	0.02	LT
Resistor	Res3	R32	33k	5%		1	0.02	0.02	LT
Resistor	Res3	R33	3.3k	5%		1	0.02	0.02	LT
Resistor	Res3	R4	4.7k	5%		1	0.02	0.02	LT

ANEXO A5

Descripción	Etiqueta	Designador	Valor	Tol.	Capacidad de voltaje	Cantidad	Precio Unitario	Costo	Comentario
Resistor	Res3	R5, R8, R11, R12, R13, R15, R17, R19, R21, R34	470	5%		10	0.02	0.20	LT
Resistor	Res3	R6	120	5%		1	0.02	0.02	LT
Resistor	Res3	R7	100	5%		1	0.02	0.02	LT
Resistor	Res3	R9, R10	420	5%		2	0.02	0.04	LT
Resistor	Res3	R35	100	5%		1	0.02	0.02	LT
Typical GREEN GaAs LED	LED2	DS4, DS5, DS6, DS7, DS8, DS9, DS10				7	0.08	0.56	C-LT
Typical RED GaAs LED	LED2	DS11				1	0.08	0.08	C-LT
Typical YELLOW GaAs LED	RUN/STOP, SATIN, SATOUT	DS1, DS2, DS3				3	0.08	0.24	C-LT
Sócalo	40 pines					1	0.40	0.40	ES
Sócalo	20 pines					1	0.40	0.40	C-TP
Sócalo	16 pines					1	0.35	0.35	LT
Sócalo	14 pines					4	0.27	1.08	LT
Sócalo	8 pines					4	0.20	0.80	LT
Placa PCB 13.5x10cm	Doble Lado metalizado					1	36.16	36.16	cue
						COSTO TOTAL MATERIALES		**77.73**	**usd**

***** Muestras gratis de Microchip - Precio aproximado EEUU
****** Muestras gratis Maxim - Precio aproximado en EEUU
s Partes obtenidas de placas de repuesto
LT Almacén Laboratorio Técnico
TP Almacén TecoMPartes
ES Almacén Electro-Sonido
C Por confirmar
cue Ing. Raúl Pesantez C./SMELEKTRONIK (Cuenca)

Nota: Precio total no incluye mano de obra ni materiales de ensamblaje

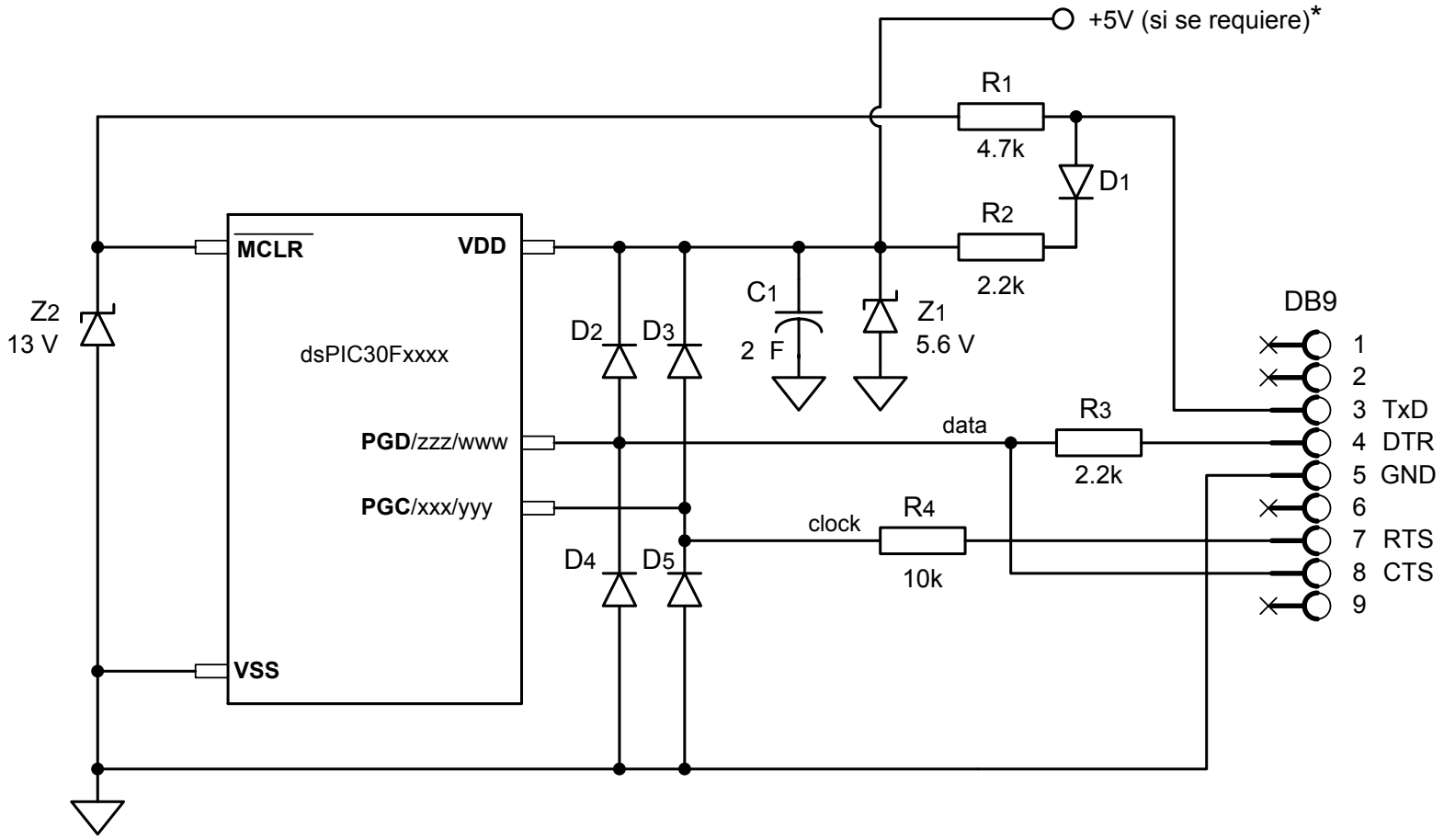
EMULADOR SISO

ANEXO A6

PROGRAMADOR JDM PARA LA FAMILIA
DsPIC30F

ANEXO A6

Programador JDM



D1 -- D5 1N4148

* Si no se logra grabar es probable que la corriente que puede entregar el puerto serial no sea suficiente para el DSC. Conectar una fuente externa de 5V.

EMULADOR SISO

ANEXO A7

CÓDIGO DE LAS FUNCIONES DE MODELADO DE LA PLANTA

ANEXO A7

Código de las funciones de modelado de la planta

TAREA2

```
void task2(void)
{
	byte status;

	//se cargan los parámetros de la planta desde la EEPROM
	iniBasicParams();
	iniTransferFunction();
	iniDelayFunction();
	iniPerextFunction();
	if (pFlow.PWMACT)
		iniPWMFunction();
	if(pFlow.VFRECACT)
		iniVFFunction();
	SFlags.PINI = 1;

	//repetir este bucle mientras esté activada la simulación
	while(SFlags.RUN_STOP)
	{
		//esperar para realizar un ciclo.
		status = K_Task_Wait(ProcPeriod);

		//leer los canales de señal y del potenciómetro.
		sig = AnalogRead(CH_SIGNAL);
		pot = AnalogRead(CH_POT);

		//se conmuta O3 con el fin de verificar el funcionamiento
		LATDbits.LATD2 ^= 1;

		//se comprueba si la señal de entrada está saturada o no.
		sig = SaturationCheck(sig,INPUT);

		//flujo de la planta. Determinado por las distintas banderas
		if (pFlow.PER1)
			sig = PerextFunction(sig,pot,pFlow);
		if (pFlow.TFACT)
			sig = TransferFunction(sig);
		if (pFlow.PER2||pFlow.PER3)
			sig = PerextFunction(sig,pot,pFlow);
		if (pFlow.RETACT)
			sig = DelayFunction(sig);
		if (pFlow.PER4)
			sig = PerextFunction(sig,pot,pFlow);
		if (pFlow.NOLINACT)
			sig = NolinFunction(sig);

		//se verifica si la señal de salida se satura
		sig = SaturationCheck(sig,OUTPUT);

		//se incluye/excluye la salida con PWM
		if (pFlow.PWMVFACT)
		{
			if(pFlow.PWMACT)
```

```
                        PWMFunction(sig,pFlow);

            }
            //se escribe el nuevo cómputo al DAC
            AnalogWrite(sig,CH_A);
      }

      //si RUN_STOP == 0 indica que se debe acabar la emulación.
      killPlantbas();
      killPlantcomp();
      AnalogWrite(0,CH_A);
      SFlags.KILLP = 0;
      SFlags.PINI = 0;
      K_Task_End();  //se termina la tarea2
}
```

FUNCIÓN DE LA ECUACIÓN DE TRANSFERENCIA

```
unsigned int TransferFunction(unsigned int input)
{

      unsigned int i;         //contador de bucles
      signed int sumi;        //suma temporal para conversiones
      unsigned char indxa[5]; //índice auxiliar de x
      unsigned char indya[5]; //índice auxiliar de y
      long double lastx;      //último valor válido de x
      long double in;         //valor de la entrada en formato double

      //convertir la entrada en doble
      in = (long double)(input);
      u = 0;

      //respaldar indx y indy
      for(i=0;i<5;i++)
      {
            indxa[i] = indx[i];
            indya[i] = indy[i];
      }

      //actualizar la linea de retardo de los datos de entrada
      for(i=0;i<ord+1;i++)
      {
            indx[i] = indx[i]+1;
            if (indx[i]>ord)
            {
                  indx[i] = 0;
                  u = i;
            }
      }

      //respaldar el último dato en la línea de retardo de entrada
      lastx = x[u];

      //registrar el nuevo valor de entrada
      x[u] = in;

      //algoritmo de la ecuación de diferencias

      sum = 0; // encerar la suma
```

```
if (zerocoef ==0) //ahorrar este paso si a[0]==0;
      { sum = a[0]*x[indx[0]];}

//suma de productos de entradas
for(i=1;i<ord+1;i++)
{
      sum = a[i]*x[indx[i]]+sum;
}
//suma de productos de las salidas
for(i=0;i<ord;i++)
{
      sum = b[i]*y[indy[i]]+sum;
}


//actualizar la línea de retardo de las salidas
for(i=0;i<ord+1;i++)
{
      indy[i] = indy[i]+1;
      if (indy[i]>ord)
      {
            indy[i] = 0;
            u = i;
      }
}

/*Protección de valores extremos:
Si el valor de salida (sum) está entre los valores -809 y 4904 que
corresponden a +-20% de sobreimpulso de la escala máxima, entonces
sum solo es limitada a los valores de 0 a 4095, pero sum sigue
tomándose en cuenta para los cálculos subsiguientes. Si sum se sale
de éstos valores el estado de la ecuación de dif. se devuelve al
estado anterior (cuando estaba en sus límites) de manera indefinida
hasta que una condición de entrada regrese a sum a sus límites
normales.*/


sumi = (signed int)(sum); //se convierte sum a unsigned int

//se revisa los límites de la señal
if( sumi <= 4904 && sumi >=-809)
      {y[u] = sum;}
else
{     //si los límites no se cumplen se regresa la ecuación de
      //diferencias a su estado anterior como si este ciclo nunca
      //hubiera existido.

      for(i=0;i<5;i++)
      {
            indx[i] = indxa[i];
            indy[i] = indya[i];
      }
      for(i=0;i<ord+1;i++)
      {
            if(indx[i]==0)
                  x[i] = lastx;
      }
}

//se recorta la señal apropiadamente
if(sumi > 4095)
```

```
            sumi = 4095;
      if(sumi < 0)
            sumi = 0;

      //se convierte la señal de salida correspondiente de flotante a
      //entera
      return(sumi);
}
```

FUNCIÓN DE RETARDO DE TRANSPORTE

```
unsigned int DelayFunction(unsigned int input)
{

      signed int dTemp;
      signed int dif;

      //modo sigma-delta
      if(buffmode == 1)
      {
            //se resta el valor actual del anterior
            dif = input - lastvalue;
            //se guarda en la pila FIFO la diferencia
            buff1[dIndex] = dif;
            //se registra la entrada actual para uso en el próximo ciclo
            lastvalue = input;
            //se incrementa el índice de la pila
            dTemp = dIndex + 1;
            //si el índice a alcanzado su máximo regresa al principio
            if (dTemp==length)
                  {dTemp = 0;}
            //se lee el valor de la diferencia que le toca salir de la
            //pila
            dif = buff1[dTemp];
            //se suma al último valor que salió de la función retardo
            output = dif + lastdelayvalue;
            //se actualiza lastdelayvalue para su uso en el siguiente
            //ciclo
            lastdelayvalue = output;
            //se aumenta el índice de la pila FIFO
            dIndex++;
            if(dIndex==length)
                  {dIndex =0;}
      }

      //modo de dato completo
      else
      {
            //Se ingresa la entrada a la pila
            //se incrementa el índice
            buff2[dIndex] = input;
            dTemp = dIndex + 1;
            if (dTemp==length)
                  {dTemp = 0;}
            //se lee el valor que le toca salir de la pila FIFO
            output = buff2[dTemp];
            //se aumenta el índice de la pila FIFO
            dIndex++;
            if(dIndex==length)
```

```
                {dIndex =0;}
        }
        return(output);
}
```

FUNCIÓN DE NO LINEALIDAD

```
unsigned int NolinFunction(unsigned int input)
{

        signed long lowy; //variables de apoyo
        signed long highy;
        signed long m;
        unsigned int lowyi;
        unsigned int highyi;
        unsigned int lowp;
        unsigned char um;

        lowp = input/16; //lowp contiene el punto inferior de la tabla
        //se lee el punto inferior
        um = ReadEE(0x007F,TABLE+2*lowp,&lowyi, WORD);
        //se lee el punto superior
        um = ReadEE(0x007F,TABLE+2*lowp+2,&highyi, WORD);

        //lowy y highy corresponden al valor de los puntos de la tabla
        //y varían de 0 a 65535
        lowy = (signed long)(lowyi);
        highy = (signed long)(highyi);

        //se calcula la pendiente entre los puntos
        m = highy-lowy;

        //se obtiene el valor entre 0 y 15 entre los dos puntos al que
        //corresponde la entrada
        highy = (signed long)(input - 16*lowp);

        //se calcula el valor correspondiente para la entrada
        m = m*highy/16;
        //se añade al valor del punto inferior
        highy = m + lowy;

        //se divide para 16 con el fin de compatibilizar con la salida
        //(0-4095)
        lowy = highy/16;
        m = highy-16*lowy;

        //se redondea correctamente la división
        if( m >= 8)
                lowy = lowy+1;
        else
                lowy = lowy;

        return((unsigned int)(lowy));
}
```

FUNCIÓN DE PERTURBACIÓN EXTERNA

```
unsigned int PerextFunction(unsigned int input,unsigned int
chpot,procFlow pFlow)
{
      unsigned long per1;
      unsigned long per2;
      unsigned int aux;

      if(!pFlow.PERTACT)  //si no hay perturbación, salir
            return(input);

      if(pFlow.PER3)  //si la perturbación es de ganancia
      {
            per1 = pesc*input; //se escalan los valores
            per1 = per1*chpot;
            per2 = per1/40950;
            aux = per1 - 40950*per2; //se redondea correctamente
            if(aux >= 20475)
                  aux = (unsigned int)(per2+1);
            else
                  aux = (unsigned int)(per2);
      }
      //si la perturbación es aditiva
      else
      {
            per1 = pesc*chpot; //se escala apropiadamente
            per2 = per1/10;
            aux = per1 - 10*per2; //se redondea correctamente
            if(aux >= 5)
                  per2 = per2+1;

            if(input >= per2)
                  aux = (unsigned int)(input - per2);
            else
                  return(0);
      }

      //se recorta si los valores salen de rango.
      if (aux > 4095)
            return(4095);
      else
            return(aux);
}
```

INDICE DE FIGURAS

INDICE DE TABLAS

GLOSARIO

ADC *Analog to Digital Converter*
Conversor análogo – digital

bootloader Cargador de arranque - Pequeño programa que se ejecuta después del arranque de un procesador y que tiene como función principal administrar la descarga de código y grabar el mismo en la memoria del dispositivo.

BOR *Brown-Out Reset*
Circuito que fuerza a un microprocesador o semejante a resetearse cuando el voltaje de alimentación cae por debajo de un límite preprogramado.

CAN *Controller Area Network*
Protocolo utilizado en la distribución de mensajes en ambientes distribuidos y que requieren alta fiabilidad y respuesta en tiempo real.

DAC *Digital to Analog Converter*
Conversor digital – analógico.

DSC *Digital Signal Controller*
Controlador digital de señales - Dispositivo que integra la capacidad y arquitectura de un DSP con la integración y fácil desarrollo de un microcontrolador.

DSP *Digital Signal Processor*
Procesador Digital de señales –es un procesador o microprocesador que posee un juego de instrucciones, un hardware y un software optimizados para aplicaciones que requieran operaciones numéricas a muy alta velocidad y son usado en aplicaciones que requieren comportamiento en tiempo real.

EEPROM *Electrically-Erasable Programmable Read-Only Memory*
Memoria sólo de lectura programable, borrable eléctricamente.

emulador Dispositivo o programa diseñado para hacer trabajar (imitar) a una computadora como otro dispositivo. Un emulador se refiere tanto a un equipo de hardware como a un software.

filtro antialiasing Circuito analógico que trabaja como filtro pasa bajos con el fin de que los datos digitalizados no se corrompan por la presencia de frecuencias superiores a la mitad de la frecuencia de muestro en la señal de entrada.

filtro IIR *Infinite Impulse Response*
Filtro digital cuya respuesta al impulso es infinita debido a la presencia en sus cálculos de resultados anteriores.

FLASH Tecnología de memoria no volátil mucho más barata que la EEPROM pero con la desventaja de que se graba/borra solo por bloques.

GUI *Graphical User Interface*
Interfaz gráfica de Usuario – Programa que sirve de nexo entre un equipo y un usuario para configurar y visualizar información del dispositivo.

GUIDE *Graphical User Interface Development Environment*
Programa incluido en MATLAB que permite la creación de interfases gráficas de usuario.

HIL *Hardware In the Loop*
Hardware en el lazo – Técnica por la cual se sustituye un equipo físico por un sistema digital que se comporta como el equipo físico real.

LVD *Low Voltage Detect*
Detección de bajo voltaje – Circuito que avisa a un microcontrolador o semejante si hay una caída de voltaje con el fin de respaldar información o tomar otras acciones.

modelado Técnicas que procuran encontrar con la mayor exactitud posible una descripción matemática para un sistema físico.

OPAMP *Operational Amplifier*
Amplificador Operacional

PCB *Printed Circuit Board*
Placa de circuito impreso

PGA *Programmable Gain Amplifier*
Amplificador de ganancia programable – Circuito discreto o integrado que por medios digitales permite fijar la ganancia de un amplificador analógico.

pila FIFO *First In – First Out*
Mecanismo lógico de almacenamiento de datos donde el primer elemento en ser ingresado en memoria es el primero en ser extraído.

planta Conjunto de sistemas mecánicos, eléctricos, químicos, etc., que realizan algún proceso.

POR Power-On Reset
Circuito que fuerza a un microcontrolador o semejante a estar en estado de reset por un tiempo después de encenderse hasta que el voltaje de alimentación se estabilice.

Potenció-metro digital Circuito Integrado que permite variar la resistencia entre dos o más de sus terminales por medio de un canal de comunicación digital.

preemptive Característica de un sistema operativo que permite escribir una tarea como si esta fuera la única que se ejecuta en el procesador. Esto indica que el sistema operativo está completamente a cargado de despachar la ejecución de las distintas tareas según su prioridad.

PWM *Pulse Width Modulation*
Modulación por ancho de pulso

RAM *Random Access Memory*
Memoria de acceso aleatorio

Riel a riel Característica de un dispositivo analógico de trabajar con señales cercanas a sus voltajes de polarización sin que esta sufra distorsiones.

RISC Reduced Instruction Set Computer
Computadora con Conjunto de Instrucciones Reducido - filosofía de diseño de CPU para computadora que esta a favor de conjuntos de instrucciones pequeños y simples que toman menor tiempo para ejecutarse

RS232 (también conocido como EIA RS-232C) es una interfaz que designa una norma para el intercambio serial de datos binarios.

RTOS *Real Time Operating System*
Sistema operativo en tiempo real - es un sistema operativo que ha sido desarrollado para aplicaciones de tiempo real. Como tal, se le exige corrección en sus respuestas bajo ciertas restricciones de tiempo. Si no las respeta, se dirá que el sistema ha fallado. Para garantizar el comportamiento correcto en el tiempo requerido se necesita que el sistema sea predecible (determinista).

scheduler Código de programa que es la base de un sistema operativo y es el encargado de administrar la creación, ejecución y destrucción de las distintas tareas definidas, juntamente con los mensajes que estas comparten.

SISO *Single Input – Single Output*
Una sola entrada, una sola salida - Descripción de un sistema por medio de una única ecuación cuya variable dependiente (salida) es función únicamente de una variable independiente (entrada).

sistemas embebidos Sistema de propósito específico en que una computadora está completamente encapsulada por el dispositivo que controla y que ejecuta tareas predefinidas con requerimientos bien delimitados y específicos.

SPI Serial Peripheral Interface
Interfase de periféricos serial - Estándar de comunicación serial síncrona de alta velocidad muy difundido para controlar periféricos, principalmente a nivel de placa.

through hole Se refiere al esquema utilizado por componentes electrónicos que involucran el uso de pines que se insertan dentro de agujeros taladrados en el circuito impreso y que se sueldan en el lado opuesto de la placa.

tiempo real Modo de trabajo de un programa en el que la reacción (cómputos respecto al contexto) ante una acción no demora más que el retardo permitido para una aplicación dada, de tal manera que se perciba “una reacción instantánea del dispositivo”.

UART *Universal Asynchronous Receiver Transmiter.*
Transmisor/Receptor Universal Asíncrono. Hardware que permite enviar datos de manera serial conteniendo la información de sincronización en la misma trama de datos.

Z_{OH} Abreviación para Retenedor de orden cero
Forma matemática de indicar el proceso de conversión análoga-digital en el que se retiene el valor de entrada hasta la próxima muestra.

ELABORADO POR:

_____

Sr. José David Santana Cobo

EL DECANO:

_____

Crnl. Xavier Martínez

EL SECRETARIO ACADÉMICO:

_____

Dr. Jorge Carvajal