UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SIMULADOR DE SISTEMAS LINEALES E INVARIANTES CON INTERFAZ DE CONFIGURACIÓN EN AMBIENTE WINDOWS

TESIS

PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

PRESENTAN:

DAN JACOB BAÑOS TOLEDO

LUIS GUILLERMO NEUMANN CASTILLO

INGENIERO ELÉCTRICO - ELECTRÓNICO

PRESENTA:

ENRIQUE CONTRERAS MARTÍNEZ

ASESOR: M.I. ANTONIO SALVÁ CALLEJA

CIUDAD UNIVERSITARIA, MÉXICO, D.F. Junio del 2005

Sin lugar a dudas, el camino que he recorrido para llegar a ser alguien en la vida, ha sido largo y arduo, desde mi infancia y hasta alcanzar la etapa adulta.

Hace varios años me encontraba frente al último camino que debía recorrer; hace algunos pocos y durante el trayecto, me encontré ante una difícil situación que pudo haber dado pie a que no lograra llegar al final. Sin embargo, ahora estoy justo a un paso de cruzar la línea final. Y para llegar a este punto, existen en mi vida muchas personas a quienes quiero agradecer de manera muy especial:

Primeramente, a mis padre, Ricardo Baños y Magnolia Toledo, por su gran esfuerzo, sacrificio e incansable apoyo. Les agradezco de todo corazón y los amo.

A mi hermanita Zudhizadaith Baños, por su cariño y paciencia.

A mi abuelita Virginia González, por alentarme, por darme ánimos en todo momento y sobre todo, por la enorme fé que me tiene. Muchas gracias Abue, te quiero mucho.

A mis más grandes amigos, Luis Neumann y David Blanco, quienes me apoyaron en todo sentido y en todo momento durante tiempos difíciles, y quienes también, hicieron posible que yo lograra alcanzar la meta final. Mil gracias queridos amigos.

Y finalmente a mis amigas Cecilia Vázquez, Paola González, Arley De la Rosa, Emilia Caldera y Karla Anguiano, por su cariño, apoyo y compresión. Muchas, en verdad muchas gracias.

Dan Jacob Baños.

Ment'se dijo: "Cuando el Cielo está a punto de conferir una importante misión a un hombre, primero, amarga su corazón en su propósito; le obliga a ejercitar sus huesos y sus nervios; hace que su cuerpo sufra hambre, infringe sobre él la carencia y la confusión de espíritu. Así, estimula su voluntad, fortalece su naturaleza y le hace capaz de realizar una obra, que jamás hubiera podido llevar a cabo."

GICHIN FUNAKOSHI

Ha sido un largo y difícil camino con el que se cierra un ciclo e inicia otro…

Quisiera agradecer, en primer lugar, a mis maravillosos padres: Luis y Virginia, por darlo todo a cambio de mi felicidad, por confiar en mí y sobre todo, por regalarme el amor que necesito para continuar. Gracias por ayudarme a elegir un camino de bien. Los AMO.

También, quisiera agradecer a mi hermana Violeta por su apoyo, cariño y compañía a lo largo de este duro camino. Te AMO hermanita.

A mi adorada novia Sofía, por eliminar la soledad en mi vida, por regalarme el amor, que también necesito para continuar. Y como si fuera poco, por su sacrificio y paciencia en este camino. Te AMO Sofi.

A todos mis amigos, a quienes temo enlistar pues no me gustaría olvidar a nadie. Un millón de gracias por estar a mi lado en las buenas y en las malas. Además, quiero darles las gracias por todas esas enseñanzas y lecciones de vida, que me han hecho madurar y crecer. No olviden que LOS QUIERO.

En particular, quiero darle las gracias a David, a Jacob y a sus respectivas familias, por ayudarme a levantar siempre que tropecé. LOS QUIERO MUCHO.

Nunca olvidaré a todos aquellos profesores que, con mucho esfuerzo, han tratado de ayudarme, enseñarme y guiarme a través de los distintos caminos que hay para resolver un problema. Gracias por su paciencia.

En especial, quiero darle gracias al M.I. Antonio Salvá, por enseñarme el “know how”, de la tecnología que ha desarrollado. Mil gracias por la ayuda, la paciencia, el apoyo y sobre todo, por escucharme y tratar de entenderme. En verdad, mil gracias por todo este tiempo.

También quiero agradecer a todos los colaboradores del departamento de Control de la D.I.E., por recibirme siempre con una sonrisa.

No quisiera terminar, sin antes dar un especial agradecimiento a los profesores: M.I. Ricardo Garibay, M. I. Antonio Salvá, Ing. Gloria Mata, Ing. Laura Sandoval y M. I. Jorge Valeriano, por ayudarnos a realizar la revisión de este trabajo y por hacernos el favor de evaluarlo. Su opinión es muy importante para nuestro trabajo. Gracias su paciencia y valioso tiempo.

La Universidad Nacional Autónoma de México, ha sido y seguirá siendo el lugar donde me formé. Estoy orgulloso de pertenecer a esta institución. Gracias UNAM, vales oro.

Finalmente, quiero agradecer a Dios, por darme vida, tiempo y la ayuda necesaria para llegar a este importante momento. Me seguiré esforzando, para ser tan buen Ingeniero como lo eres tú. Gracias AMIGO.

Luis Neumann.

A mi padres, Rosa María Martínez Castillo y Enrique Contreras López, quienes con su esfuerzo, dedicación y tenacidad han sabido superar la adversidad, y que con su ejemplo me enseñaron la razón de vivir, el significado del amor, y la importancia de la educación en este teatro que llamamos vida. Por lo que son y el amor que me han dado, gracias.

A mi hermana, con quien a lo largo de los años que duró mi licenciatura compartí momentos inolvidables, por ser quien es y por estar a mi lado en las situaciones dulce y amargas, gracias.

A mi asesor de tesis: M.I. Antonio Salvá Calleja por su interés en este proyecto, por el tiempo que nos dedicó, y por sus comentarios que sin duda ayudaron a la exitosa culminación de este trabajo.

A mis compañeros de tesis quienes siempre se mostraron dispuestos a compartir sus conocimientos y a trabajar en equipo. En especial, les agradezco la paciencia y comprensión que tuvieron cuando no pude estar directamente trabajando con ellos.

A todas aquellos seres, cuyos nombres y acciones permanecerán en mi memoria, y que a lo largo de la parte de mi vida que transcurrió durante la licenciatura me permitieron percatarme de mi estado de ser, y gracias a lo cuales encontré respuestas a algunas de las interrogantes de mi vida.

A todos aquellos que han trabajado y trabajan en favor de la UNAM y la F.I, sobre todo a quienes han defendido los ideales que permiten que la UNAM siga siendo la cuna del espíritu de México.

¡MI ETERNO AGRADECIMIENTO A TODOS POR SU PARTICIPACIÓN EN ESTA PARTE DE LA OBRA DE MI VIDA!

Enrique Contreras Martínez.

TABLA DE CONTENIDO

LISTA DE TABLAS Y FIGURAS

INTRODUCCIÓN

Con el advenimiento de la computadora digital electrónica comercial a principios de la década de los cincuentas, se han desarrollado una gran cantidad de herramientas analíticas que han tenido un profundo impacto en el campo científico. Una de estas herramientas es precisamente la simulación, cuyos usos y aplicaciones se han extendido significativamente en los últimos años. Así pues, es muy común encontrar en la actualidad, aplicaciones de simulación en áreas tales como: economía, finanzas, sistemas de inventarios, análisis y evaluación de inversiones, sistemas de colas, etcétera. Sin embargo, es necesario señalar que este trabajo está dedicado al diseño, análisis y validación de sistemas dinámicos lineales en invariantes en el tiempo.

La simulación y las computadoras son desarrolladas con variados enfoques, de acuerdo a los objetivos que guían cada texto, las orientaciones de los autores y sus ideas acerca de la importancia relativa de los diversos tópicos.

Particularmente, en teoría de circuitos y control un concepto fundamental es el modelado y simulación de sistemas dinámicos lineales e invariables. En la enseñanza de estos tópicos, es importante contar con bloques funcionales que simulen el comportamiento de un sistema dinámico real. Estos bloques pueden ser utilizados para prácticas estudiantiles que ayuden a la asimilación de conceptos matemáticos alrededor de la dinámica y control de los mismos.

Para implementar tales simuladores, como se sabe, existen dos vertientes:

1. Simuladores analógicos, los cuales emplean bloques funcionales tales como integradores, sumadores y potenciómetros que interconectados de una forma determinada, pueden llevar a cabo la simulación de un sistema dinámico en particular.
2. Simuladores digitales, los cuales se implementan mediante el uso de una computadora digital, la cual realiza en tiempo real, la función de

transferencia $H[z]$ asociada con la discretización de la función de transferencia $G(s)$ correspondiente al sistema dinámico analógico que se pretende simular.

Este trabajo reporta los resultados obtenidos en la realización de un Simulador Digital de Sistemas Dinámicos Lineales e Invariables (SDSLI). El dispositivo se desarrolló alrededor del microcontrolador 68HC11F1 montado en la arquitectura FACIL_11B. Para el desarrollo del software asociado, se utilizó principalmente un compilador cruzado del lenguaje C de uso comercial; no obstante, otros bloques funcionales de software se desarrollaron en lenguaje ensamblador para el 68HC11. Es también de importancia, dejar claro que se desarrolló una interfaz de usuario para la plataforma Windows, con la finalidad de facilitar al usuario el control, configuración y uso del hardware del SDSLI.

El capítulo 1 de este trabajo, introduce los conceptos básicos, los principios de la formación de modelos y la técnica de simulación de sistemas, todo esto, con la finalidad de que el lector comprenda lo expuesto en este trabajo de tesis.

El capítulo 2, se refiere al estado del arte en la simulación de sistemas lineales e invariantes con el objetivo de mostrar al lector, las soluciones actuales al problema de la simulación de sistemas lineales, sus ventajas y desventajas y sobre todo, observar los aspectos no resueltos.

El capítulo 3, expone el problema del Simulador Digital de Sistemas Lineales e Invariantes (SDSLI), en donde se propone la implementación del SDSLI, dando solución a diversas cuestiones no resueltas en la simulación de los sistemas lineales y además, como una herramienta complementaria a las que posee la Facultad de Ingeniería de la UNAM, para apoyar a los laboratorios en los cuales se realiza simulación de sistemas lineales.

A continuación, el capítulo 4, trata sobre el desarrollo de hardware que da origen al SDSLI, en el que se encuentra la circuitería de conversión Digital a Analógica y de acondicionamiento de señales eléctricas para adaptarlas a la tarjeta de desarrollo FACIL_11B basada en el microcontrolador 68HC11F1. Además, se expone el circuito que forma la fuente de poder con la cual se alimenta todo el hardware del SDSLI. Este capítulo, presenta también, en forma breve, el diseño del software de bajo y alto nivel, para simular sistemas lineales en invariantes en el tiempo con el hardware del SDSLI, así como la presentación del software que conforma la interfaz para Windows del SDSLI, y que permite al usuario aprovechar con facilidad, las capacidades del simulador.

Con el capítulo 5, se proporciona un ejemplo práctico en el cual se demuestra el uso y validez de los resultados obtenidos por el simulador.

Finalmente, el apartado 6, correspondiente a las conclusiones, proporciona en declaraciones cortas y concisas las inferencias derivadas de este trabajo. También en esta sección, se presentan las contribuciones nuevas al conocimiento generado por esta tesis. La última subsección de este apartado, se incluye con la finalidad de que investigadores que se involucren con este trabajo en el futuro, reciban el beneficio de las nuevas ideas generadas por éste.

CAPÍTULO 1

TEORÍA BÁSICA

En este capítulo encontrará:

1. TEORÍA BÁSICA

1.1. CONCEPTOS BÁSICOS

En este capítulo se explicarán y definirán brevemente los conceptos básicos necesarios, que establecen los fundamentos teóricos que originan el tema de este trabajo.

1.1.1. Concepto de señal

Se conoce como señal a todo estímulo que contiene información acerca del comportamiento o la naturaleza de algún fenómeno.

Desde un punto de vista matemático, las señales se representan como funciones con una o más variables independientes. Estas señales pueden ser expresadas en función de dos factores:

- En función del tiempo $f(t)$, donde t representa al tiempo.
- En función de la frecuencia $F(s)$, donde s representa literalmente a la variable de Laplace.

1.1.2. Definición de señales continuas y señales discretas

Existen dos tipos básicos de señales: *señales continuas* y *señales discretas*.

Las *señales continuas* son aquellas que se definen para una sucesión continua de valores, es decir, toman valores reales o complejos para todo valor de t.

Las *señales discretas* por su parte, están definidas para un conjunto de valores discretos, esto significa que toman valores solamente en ciertos instantes de tiempo.

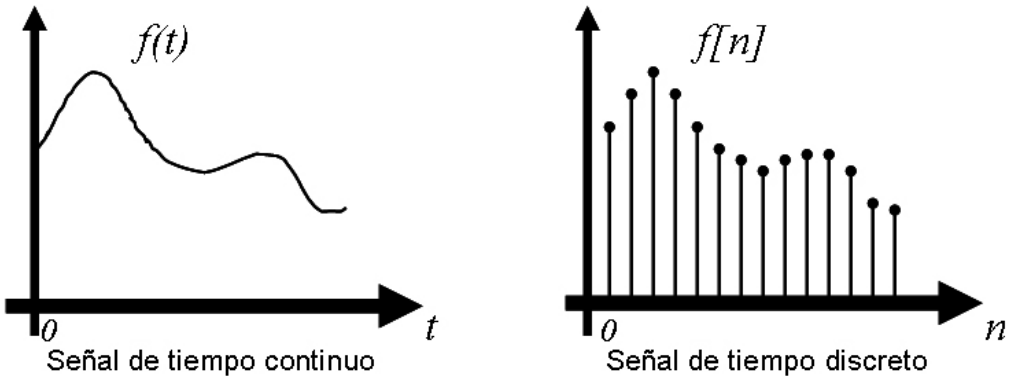


Figura 1.1. Ejemplo de señal continua y señal discreta

1.1.3. Concepto de sistema

El término *sistema* lo utilizamos en nuestra vida diaria en tal diversidad de maneras, que resulta difícil llegar a una definición lo suficientemente extensa y concisa para tratar de explicarlo.

A diario, hablamos de sistemas políticos, sistemas eléctricos, sistemas bancarios, sistemas de comunicaciones, sistemas económicos, etc. Cada uno de ellos son distintos, pero tomándolos en términos generales como sistemas, todos ellos tienen características en común, ya que están formados por un conjunto de partes elementales, que en combinación, actúan juntas presentando un determinado comportamiento y entregando resultados, con base a cierta información que han recibido como entrada.

Así pues, de acuerdo con lo anterior, podemos definir a un *sistema como una combinación de componentes elementales, que actúan juntos para alcanzar un fin común, presentando un comportamiento fundamentado en cierta información de entrada y generando resultados como salida*. En forma gráfica, esto lo podemos representar de la siguiente manera:

Figura 1.2. Representación general de un sistema

Desde un punto de vista formal, podemos estudiar a un sistema con un modelo matemático el cual, relaciona entradas (señales de entrada) y salidas (señales de salida) de acuerdo a una regla preestablecida. A un sistema también lo podemos ver como un proceso que produce una transformación de señales.

Se utilizará "el término *entidad* para denotar un objeto de interés en un sistema; el término *atributo* denota una propiedad de una entidad. Desde luego, pueden haber muchos atributos de una entidad dada. Todo proceso que provoque cambios en el sistema se conocerá como *actividad*. Se utilizará el término *estado del sistema* para indicar una descripción de todas las entidades, atributos y actividades de acuerdo con su existencia en algún punto del tiempo. El progreso del sistema se estudia siguiendo los cambios en el estado del sistema" **[Gordon 1982]**.

1.1.4. Concepto de sistemas dinámicos

Los *sistemas dinámicos* son aquellos en los cuales, uno o más aspectos de ellos son dependientes del tiempo, es decir, sus características en un tiempo determinado están interrelacionadas con otras en otro instante de tiempo. Dicho de otra manera, en un sistema dinámico, la salida o respuesta de éste no permanece constante, sino que varía con respecto al tiempo. Cabe mencionar que los sistemas dinámicos están formados por algunos elementos que son capaces de almacenar energía.

1.1.5. Definición de sistemas continuos y sistemas discretos

Los *sistemas continuos* son aquellos que procesan señales continuas o analógicas; en otras palabras, la señal de entrada en tiempo continuo, es transformada en una señal de salida también en tiempo continuo.

Un s*istema discreto* es aquel que procesa señales discretas o digitales; es decir, que toma como entrada una señal discreta para transformarla y producir una señal de salida en tiempo discreto.

Lo anterior se ilustra como sigue:

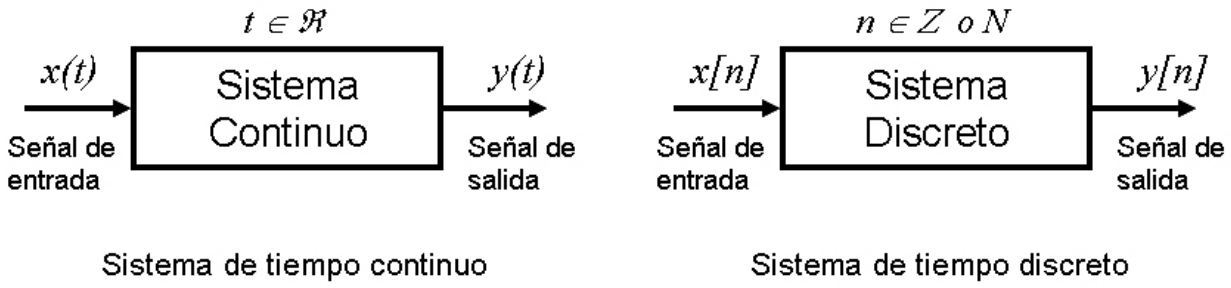


Figura 1.3. Representación general de los sistemas continuos y discretos

1.1.6. Definición de sistema lineal e invariante

Un sistema es *lineal, si* cumple con dos principios fundamentales: *superposición* y *homogeneidad*.

- *Superposición*: este principio, nos dice que la respuesta total de un sistema es la suma de las repuestas debidas a cada una de las entradas. Es decir que se puede considerar una entrada a la vez, obtener su respuesta y posteriormente sumar todas las respuestas para obtener la respuesta total del sistema.

- *Homogeneidad*: este segundo principio, indica que un cambio en la amplitud de la señal de entrada, reflejará el mismo cambio en la amplitud de la señal de salida, es decir, que el sistema es proporcional, ya que la causa y el efecto son proporcionales.

Esto lo podemos explicar de manera matemática como sigue:

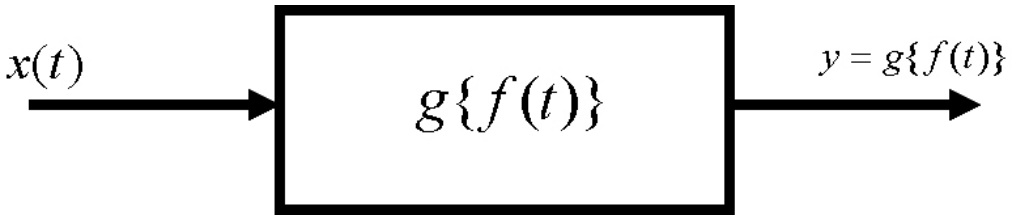


Figura 1.4. Modelo matemático de un sistema

Considerando por separado las entradas:

$$\begin{aligned} x_1(t) &\xrightarrow{g\{f(t)\}} y_1(t) = g\{x_1(t)\} \\ x_2(t) &\xrightarrow{g\{f(t)\}} y_2(t) = g\{x_2(t)\} \end{aligned} \tag{1.1}$$

Tomando los escalares a y b tal que:

$$x_3(t) = ax_1(t) + bx_2(t) \Rightarrow y_3(t) = g\{x_3(t)\} \tag{1.2}$$

Entonces, el sistema es lineal si cumple con el principio de superposición:

$$y_3(t) = g\{ax_1(t) + bx_2(t)\} = g\{ax_1(t)\} + g\{bx_2(t)\} \tag{1.3}$$

y también con el de homogeneidad:

$$\begin{aligned} y_3(t) &= ag\{x_1(t)\} + bg\{x_2(t)\} \\ y_3(t) &= g\{ax_1(t) + bx_2(t)\} = ay_1(t) + by_2(t) \end{aligned} \tag{1.4}$$

Un sistema es *invariante* si su respuesta a una misma entrada, es igual en cualquier instante de tiempo, es decir, las características del sistema permanecen constantes mientras transcurre el tiempo. Matemáticamente hablando esto es:

Si:

$$x_1(t) \xrightarrow{g\{f(t)\}} y_1(t) = g\{x_1(t)\} \tag{1.5}$$

y si consideramos:

$$\begin{aligned} x_2(t) &= x_1(t \pm t_0) \\ y_2(t) &= g\{x_2(t)\} = g\{x_1(t \pm t_0)\} \end{aligned} \tag{1.6}$$

Entonces, el sistema es invariante si:

$$y_2(t) = y_1(t \pm t_0) \tag{1.7}$$

Así pues, *un* sistema es *lineal* e *invariante* si cumple con los principios de superposición y homogeneidad, y además, si sus características permanecen constantes al transcurrir el tiempo. Esto por supuesto, es aplicable tanto para sistemas continuos como para sistemas discretos.

Además de estas dos propiedades de los sistemas, existen otras propiedades de importancia, tales como la *estabilidad* y la *causalidad*.

Un sistema es *causal* si su salida en cualquier instante de tiempo depende sólo de los valores de entrada en el momento presente y en el pasado, es decir, la salida del sistema no anticipa valores futuros de la entrada.

Un sistema *estable*, es aquel en el que entradas pequeñas conducen a repuestas que convergen.

1.2. MODELADO DE SISTEMAS

Para estudiar un sistema es posible experimentar con él. Sin embargo, el objetivo de muchos estudios de sistemas es predecir la manera como se comportará el sistema antes de que sea construido. “Es claro que no es factible experimentar con un sistema mientras está todavía en su forma hipotética. Una alternativa que se utiliza a veces, es construir una cantidad de prototipos y probarlos, lo que puede ser muy costoso y tardado, incluso con un sistema existente, es seguro que sea imposible o impráctico experimentar con el sistema real” **[Gordon 1982]**.

En consecuencia, por lo general los estudios de sistemas se realizan con un modelo. Para fines de casi todos los estudios, no es necesario tener en cuenta todos los detalles de un sistema; por consiguiente, un modelo no sólo es el sustituto de un sistema, sino también una simplificación del mismo.

Definimos *modelo, como una representación de las partes esenciales de un sistema existente (o sistema a ser construido), la cual proporciona información de este sistema en una forma útil y manejable*. Ya que el propósito del estudio determina la naturaleza de la información que se reúne, no hay un modelo único de un sistema. Los distintos analistas interesados en diferentes aspectos del sistema o el mismo analista, producirán distintos modelos del sistema según cambie su comprensión.

“La tarea de obtener un modelo de un sistema, se dividirá en forma genérica en dos subtareas: *la determinación de la estructura del modelo* y *proporcionar los datos”*

[Gordon 1982]. La *determinación de la estructura*, fija la frontera del sistema e identifica las entidades, atributos y actividades de éste. Los *datos*, suministran los valores de los atributos que pueden tener y definen las relaciones involucradas en las actividades. “Las dos tareas se definen como partes de una tarea más que como dos tareas por separado, debido a que por lo general, están tan íntimamente relacionados, que no se puede hacer una sin la otra” **[Gordon 1982]**. Las suposiciones relativas al sistema orientan la recolección de datos, y el análisis de éstos, confirma o refuta las suposiciones. Es común que los datos recolectados revelen una relación no sospechada que cambie la estructura del modelo.

1.2.1. Tipos de modelos

Para fines de esta tesis, se considera a los modelos como *modelos físicos* o *modelos matemáticos*.

Una segunda distinción la constituyen los *modelos estáticos* y los *modelos dinámicos*. En el caso de los modelos matemáticos, una tercera distinción es la técnica que se emplea para resolver el modelo. Se establece una distinción entre los métodos *analítico* y *numérico*.

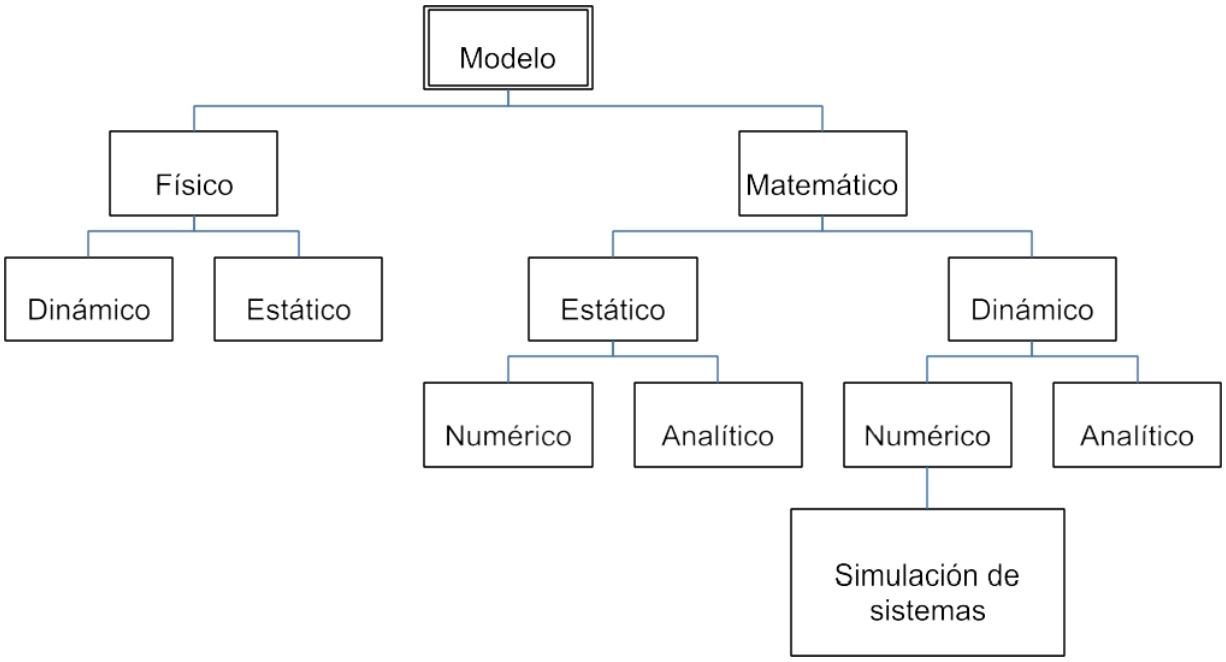


Figura 1.5. Tipos de modelos

1.2.1.1. Modelos físicos

Al explicar los modelos de sistemas en la sección anterior, no pretendimos implicar que un modelo es necesariamente una descripción matemática. Es posible formar sistemas en modelos físicos, cuyo comportamiento representa el sistema que se estudia. Los atributos de las entidades del sistema se representan mediante medidas físicas. Las actividades de sistemas se reflejan en las leyes físicas que subyacen al modelo.

Los ejemplos mejor conocidos de los modelos físicos son los modelos a escala que se utilizan en los túneles de viento y tanques de agua para estudiar el diseño de aeronaves y naves acuáticas. Las leyes bien establecidas de la similitud permiten realizar deducciones exactas relativas al comportamiento de un sistema a escala natural a partir del modelo a escala. Otros tipos de modelos físicos son los modelos icónicos, es decir, modelos que “semejan” al sistema que modelan; por ejemplo, los modelos de las estructuras moleculares formados a partir de esferas que representan a los átomos, con varillas que representan los enlaces atómicos. Tanto los modelos a escala como los modelos icónicos son ejemplos de modelos *físicos estáticos*.

Los modelos *físicos dinámicos* se apoyan en una analogía entre el sistema que se estudia y algún otro sistema de alguna naturaleza distinta, en que por lo general la analogía depende de una similitud subyacente en las fuerzas que gobiernan el comportamiento de los sistemas.

1.2.1.2. Modelos matemáticos

En este tipo de modelos, “las entidades de un sistema y sus atributos se representan mediante variables matemáticas. Las actividades se describen mediante funciones matemáticas que interrelacionan las variables” **[Gordon 1982]**.

Un modelo *estático* “despliega las relaciones entre los atributos del sistema cuando éste está equilibrado. Si se cambia el punto de equilibrio alterando uno o más de los atributos, el modelo permite deducir los nuevos valores de todos los atributos, pero no muestra la manera en que cambiaron a sus nuevos valores” **[Gordon 1982]**.

Dependiendo de la naturaleza del modelo, es posible resolverlo analíticamente o puede ser necesario resolverlo numéricamente.

Un modelo *matemático dinámico*, permite deducir los cambios de los atributos del sistema en función del tiempo. Dependiendo de la complejidad del modelo, la deducción puede hacerse con una solución analítica o de cómputo numérico.

Los modelos matemáticos dinámicos que se pueden resolver analíticamente y que dan resultados prácticos, no son muy comunes. “Es más frecuente que tenga que resolverse el modelo mediante métodos numéricos, la simulación es uno de esos métodos” **[Gordon 1982]**.

1.2.2. Principios utilizados en el modelado

No es posible suministrar reglas según las cuales se construyan modelos matemáticos, aunque sí se pueden expresar una diversidad de principios de guía. No describen los pasos claros que se realizan en la construcción de un modelo, sino que describen los distintos puntos de vista desde los cuales se puede juzgar la información a incluir en el modelo.

1.2.2.1. Formación de bloques

La descripción del sistema se debe organizar en una serie de bloques o subsistemas. El propósito de formar los bloques es simplificar la especificación de las interacciones dentro del sistema. Cada bloque describe parte del sistema que depende de pocas, preferiblemente una, variables de entrada y produce unas pocas variables de salida. Luego, puede describirse al sistema como un todo en términos de las interconexiones entre los bloques. En forma correspondiente, se puede representar gráficamente al sistema como un diagrama simple de bloques.

1.2.2.2. Relevancia

El modelo sólo debe de incluir los aspectos del sistema más importantes y de interés a los objetivos del estudio. Aunque la información irrelevante en el modelo no perjudica, se debe de excluir debido a que aumenta el grado de complejidad del modelo y genera más trabajo en la solución de éste.

1.2.2.3. Exactitud

Debe tenerse en cuenta la validez de la información que se recabe.

1.2.2.4. Agregación

Un factor adicional que debe considerarse es el grado con que pueden agruparse las distintas entidades individuales en entidades más grandes.

En algunos estudios, puede ser necesario construir entidades artificiales mediante el proceso de agregación.

A la representación de actividades se debe de dar consideraciones semejantes de agregación.

1.3. SIMULACIÓN DE SISTEMAS

1.3.1. Definición de la simulación de sistemas

Dado un *modelo matemático* de un sistema, a veces es posible obtener información relativa al mismo por medios analíticos. Cuando no es posible, es necesario utilizar métodos de cómputo numérico para resolver las ecuaciones. Se ha desarrollado una rica diversidad de métodos de cómputo numérico para resolver las ecuaciones de modelos matemáticos. En el caso particular de los modelos matemáticos dinámicos, una técnica específica que se ha llegado a identificar como simulación de sistemas es aquella en que se resuelven simultáneamente todas las ecuaciones del modelo con valores continuamente crecientes del tiempo.

Por tanto “definimos la *simulación de sistemas* como la técnica de resolver problemas siguiendo los cambios en el tiempo de un modelo dinámico de un sistema” **[Gordon 1982]**. La definición es suficientemente amplia para que incluya el uso de modelos físicos dinámicos, en cuyo caso se evalúan las variables del modelo a través de mediciones físicas en lugar de cómputos numéricos.

Ya que la técnica de simulación no pretende resolver analíticamente las ecuaciones de un modelo, por lo general un modelo matemático construido para fines de simulación, es de naturaleza distinta a uno formado para técnicas analíticas. Al formar un modelo para la solución analítica, es necesario tener presentes las restricciones impuestas por la técnica analítica y evitar complicar el modelo global. Se tienen que hacer muchas suposiciones generales para satisfacer estas restricciones.

Sin embargo, se puede construir con mayor libertad un modelo de simulación. Típicamente, se forma en una serie de secciones que corresponden al método de diagrama de bloques recomendado en la sección 1.2.2.1. Se puede describir matemáticamente a cada sección en forma directa y natural sin dar demasiada consideración a la complejidad que se introduce por tener muchas de esas secciones. Sin embargo, es necesario formar y organizar las ecuaciones de tal manera, que se pueda utilizar un procedimiento rutinario para resolverlas simultáneamente.

“En los sistemas continuos, en que el interés primordial son los cambios suaves, generalmente se utilizan conjuntos de ecuaciones diferenciales para describirlos. Se dice que las simulaciones basadas en esos modelos son *simulaciones continuas.* Las computadoras analógicas, pueden resolver conjuntos de ecuaciones diferenciales lineales en forma simultánea, y se utilizan extensamente para la simulación continua. Las computadoras digitales, pueden realizar la misma función utilizando pequeños incrementos de intervalos para integrar las ecuaciones” **[Gordon 1982]**.

“Para los sistemas discretos en que el interés primario está en los eventos, las ecuaciones son esencialmente ecuaciones lógicas que expresan las condiciones para que ocurra un evento. La simulación consiste en seguir cambios en el estado del sistema, resultado de la sucesión de eventos. Se dice que esas simulaciones son *discretas.* Es posible avanzar el tiempo en pequeños incrementos y verificar en cada paso, si ya es necesario ejecutar cualquiera de los eventos. Sin embargo, por regla

general, la simulación discreta se realiza decidiendo una secuencia de eventos y avanzando el tiempo al evento siguiente más inminente“ **[Gordon 1982]**.

1.3.2. Naturaleza experimental de la simulación

La técnica de la simulación no intenta específicamente, aislar las relaciones entre determinadas variables; en vez de ello, observa la manera en que cambian todas las variables del modelo con el tiempo. Las relaciones entre las variables deben deducirse de esas observaciones. Se tienen que realizar muchas corridas de simulación para comprender las relaciones que participan en el sistema, por lo que debe planearse la simulación en un estudio como una serie de experimentos.

La manera en que se desarrollan los experimentos de simulación depende de la naturaleza del estudio Por lo general, los estudios de sistemas son de tres tipos principales: a*nálisis de sistemas*, *diseño de sistemas* y lo que llamaremos *postulación de sistemas*.

En realidad, muchos estudios combinan dos o tres de estos aspectos o los alternan según avanza el estudio. Con frecuencia se utiliza el término *ingeniería de sistemas* para describir estudios de sistemas, en los casos en que se pretende que una combinación del análisis y el diseño, comprenda primero la manera como trabaja un sistema existente y luego prepare modificaciones al sistema para cambiar el comportamiento del mismo.

El *análisis de sistemas* pretende comprender la manera en que opera un sistema existente o propuesto. La situación ideal sería que el investigador pudiera experimentar con el propio sistema, pero lo que realmente se hace, es construir un modelo del sistema y mediante simulación, se investiga el comportamiento del modelo. Los resultados obtenidos se interpretan en términos del comportamiento del sistema.

En los estudios del *diseño de sistemas,* el propósito es producir un sistema que satisfaga algunas especificaciones. El diseñador puede elegir o planear determinados sistemas de componentes, y conceptualmente elige una combinación determinada de componentes para construir un sistema. El sistema propuesto se modela y se predice su comportamiento a partir del conocimiento del comportamiento del modelo. Si el comportamiento predicho se compara favorablemente con el comportamiento deseado, se acepta el diseño. En caso contrario, se rediseña el sistema y se repite el proceso.

La *postulación del sistema* es característica de la manera en que se emplea la simulación en estudios sociales, económicos, políticos y médicos en que se conoce el comportamiento del sistema pero no así con los procesos que producen dicho comportamiento. Se establecen hipótesis de un conjunto probable de entidades y actividades que pueden explicar el comportamiento. El estudio compara la respuesta del modelo con base en esas hipótesis contra el comportamiento conocido. Una comparación razonablemente buena, conduce en forma natural a la suposición de que la estructura del modelo semeja el sistema real, y permite postular una estructura del sistema. Con mucha seguridad, el comportamiento del modelo da una mejor percepción del sistema, que posiblemente ayude a formular un conjunto refinado de hipótesis.

1.3.3. Pasos involucrados en los estudios de simulación

La aplicación de la simulación a muchos tipos de sistemas junto con los distintos tipos de estudios, producen muchas variaciones en la forma como se desarrolla un estudio de simulación. Sin embargo, se pueden identificar determinados pasos básicos en el proceso. Los principales que deben de considerarse son:

1. Definición del problema.
2. Plan del estudio.
3. Formulación de un modelo matemático.

4. Construcción de un programa de computadora para el modelo.
5. Validación del modelo.
6. Diseño de experimentos.
7. Ejecución de la corrida de simulación y análisis de resultados.

Los dos primeros pasos son definir el problema y planear el estudio. Aunque estos pasos pueden parecer obvios, no dejan de ser importantes. No debe desarrollarse ningún estudio ni simulación sino hasta que se enuncien claramente el problema y los objetivos del estudio. Luego se pueden hacer las estimaciones del trabajo por realizar y del tiempo requerido. La utilidad del plan tampoco concluye cuando se inicia el estudio; el plan puede controlar el desarrollo del trabajo e impedir que el estudio se salga de balance, concentrándose en un aspecto del problema a costa de otro. Un fracaso común en los estudios de simulación, es que se concentra tanto en ésta, que de la simulación se extraen más datos de los necesarios o de los que pueden validarse con los datos disponibles.

El tercer paso consiste en construir un modelo, tarea que se puede considerar que cae dentro de dos subtareas. Es necesario establecer la estructura del modelo decidiendo los aspectos del comportamiento del sistema que son significativos para el problema de que se trata, y es necesario reunir los datos para proporcionar parámetros correctos para el modelo.

Dado un modelo matemático, la construcción de un programa de computadora para el modelo, el cuarto paso, es una tarea relativamente bien definida. No es una tarea fácil necesariamente, y puede ser larga, pero el modelo establece las especificaciones de lo que debe de programarse.

Es probable que las tareas de producir un modelo y programa de computadora se realicen en paralelo más que en serie.

El quinto paso, la validación del modelo, es un área que requiere buena cantidad de juicio. En gran medida, el problema es el complemento de la formulación del modelo. Las inferencias que se hacen al determinar el modelo se comprueban observando si éste se comporta como se esperó. Desde luego, pueden ocurrir errores al programar el modelo. Idealmente, los errores del modelo y los de programación, se separan validando el modelo matemático antes de iniciar la programación. Sin embargo, no es fácil hacerlo debido a que antes de todo, la razón de simular, generalmente, es que el modelo matemático no es manejable de manera analítica. Puede ser factible resolver casos especiales, por ejemplo, quitando todo carácter aleatorio, pero por regla general, la validación se desarrolla examinando la versión de computadora del modelo.

El sexto paso es el diseño de un conjunto de experimentos que satisfagan los objetivos del estudio. Un factor que debe de considerarse, es el costo de correr el modelo de computadora, ya que ello puede limitar el número de corridas que puedan hacerse. Y aunque no exista esta limitación, se debe de ponderar cuidadosamente el número de corridas que se necesitan. Una falla común en los estudios de simulación es que el analista de sistemas queda abrumado por una masa de resultados de computadora que se recaban sin plan determinado. La presencia de eventos aleatorios en una simulación complica el diseño de los experimentos, ya que debe de considerarse el significado estadístico de los resultados.

El último paso en el estudio de un sistema, es ejecutar las corridas de simulación e interpretar los resultados. En un estudio bien planeado se habrá planteado un conjunto bien definido de preguntas y el análisis tratará de responderlas.

1.4. SIMULACIÓN DE SISTEMAS CONTINUOS

1.4.1. Modelos de sistemas continuos

Como ya se mencionó anteriormente, un sistema continuo es aquel en que las actividades predominantes del sistema provocan cambios suaves en los atributos de las entidades del mismo. Cuando se modela matemáticamente al sistema, las variables del modelo que representan los atributos, se controlan mediante funciones continuas. De manera más general, en los sistemas continuos las relaciones describen las *tasas* con las que cambian los atributos, de tal manera que el modelo consiste en ecuaciones diferenciales.

"Los modelos más simples de ecuaciones diferenciales tienen una o más ecuaciones diferenciales lineales con coeficientes constantes. Entonces, con frecuencia, es posible resolver el modelo sin utilizar simulación. Aun así, el trabajo involucrado puede ser tan extenso que sea preferible utilizar técnicas de simulación. Sin embargo, cuando se introducen no linealidades al modelo, con frecuencia es imposible o al menos muy difícil, resolver los modelos. Los métodos de simulación para resolver los modelos no cambian fundamentalmente cuando ocurren no linealidades. El método de aplicar la simulación a los modelos continuos puede entonces desarrollarse mostrando su aplicación a los modelos en que las ecuaciones diferenciales son lineales y tienen coeficientes constantes, y luego generalizar a ecuaciones más complejas" **[Gordon 1982]**.

1.5. SIMULACIÓN MEDIANTE FUNCIONES DE TRANSFERENCIA

La simulación de sistemas a través de la función de transferencia, representa un enfoque útil para investigar el comportamiento dinámico de los sistemas.

1.5.1. Función de transferencia del sistema

Los sistemas dinámicos se pueden entender y analizar sin la ayuda de las matemáticas, incluso en nuestra vida diaria nos encontramos con situaciones dinámicas simples. Sin embargo, para los casos complejos, es necesario proceder sistemáticamente y con herramientas que permitan tratar el problema. Es aquí donde el uso de las matemáticas, nos dan la posibilidad de interpretación del problema o el análisis de los sistemas dinámicos.

Comúnmente como ya se mencionó, los sistemas dinámicos se representan en términos de ecuaciones diferenciales o de ecuaciones en diferencias, para representar el comportamiento dinámico de los sistemas. Si el comportamiento que se desea analizar ocurre en tiempo continuo, se hace uso de las ecuaciones diferenciales y, si el comportamiento por analizar es en tiempo discreto se utilizan las ecuaciones en diferencias.

A partir de dichas ecuaciones diferenciales o ecuaciones en diferencias, es posible obtener la función de transferencia del sistema.

La función de transferencia de un componente o sistema, establece una relación directa entre su respuesta (señal de salida) y excitación (señal de entrada).

1.5.2. Determinación de la función de transferencia basándose en la ecuación diferencial del sistema

Un sistema continuo lineal e invariante de orden n esta representado en términos de una ecuación diferencial de orden n y coeficientes constantes, de la siguiente manera:

$$\begin{aligned} &y^n(t) + a_1 y^{n-1}(t) + \ldots + a_{n-1} y^1(t) + a_n y(t) = \\ &b_0 x^m(t) + b_1 x^{m-1}(t) + \ldots + b_{m-1} x^1(t) + b_m x(t) \end{aligned} \tag{1.8}$$

Considerando condiciones iniciales nulas, al aplicar la transformada de Laplace a la ecuación (1.8), tenemos:

$$\begin{aligned} &s^n Y(s) + a_1 s^{n-1} Y(s) + \ldots + a_{n-1} s Y(s) + a_n Y(s) = \\ &b_0 s^m X(s) + b_1 s^{m-1} X(s) + \ldots + b_{m-1} s X(s) + b_m X(s) \end{aligned} \tag{1.9}$$

Trabajando con la expresión transformada, factorizando y despejando llegamos a:

$$Y(s)\left[s^n + a_1 s^{n-1} + \ldots + a_{n-1} s + a_n \right] = X(s)\left[b_0 s^m + b_1 s^{m-1} + \ldots + b_{m-1} s + b_m \right]$$

$$\frac{Y(s)}{X(s)} = \frac{b_0 s^m + b_1 s^{m-1} + \ldots + b_{m-1} s + b_m}{s^n + a_1 s^{n-1} + \ldots + a_{n-1} s + a_n} = G(s) \tag{1.10}$$

$$G(s) = \frac{Y(s)}{X(s)}$$

Donde $G(s)$ es la función de transferencia del sistema continuo.

Esta función de transferencia del sistema, $G(s)$, tiene una importancia enorme para sistemas continuos, ya que a través de ella, se puede llevar a cabo un análisis más sencillo de los sistemas, permitiendo contemplarlos como un operador matemático en el dominio de la frecuencia. Por medio de ella podemos conocer su comportamiento, es decir, patrón de polos y ceros, estabilidad y función de peso o respuesta al impulso.

El grado del polinomio del denominador, define el orden de la función de transferencia (y del sistema): primero, segundo, tercero, etc.

Las raíces del polinomio del denominador son conocidas como “polos” de la función de transferencia; mientras que a las raíces del numerador, se les conoce como “ceros” de la función de transferencia.

A través de los polos, es posible determinar la respuesta en el tiempo que tiene la función de transferencia cuando se le aplica una señal de entrada o excitación.

Por otro lado, para el caso de un sistema discreto lineal e invariante de orden n, la ecuación en diferencias se enuncia como sigue:

$$\begin{aligned} &y[n]+\alpha_1 y[n-1]+\alpha_2 y[n-2]+\ldots+\alpha_q y[n-q]= \\ &\beta_0 x[n]+\beta_1 x[n-1]+\beta_2 x[n-2]+\ldots+\beta_m x[n-m] \end{aligned} \tag{1.11}$$

La función de transferencia se obtiene al aplicar la transformada Z a la anterior ecuación en diferencias, considerando condiciones iniciales nulas. Así:

$$\begin{aligned} &Y[z]+\alpha_1 z^{-1}Y[z]+\alpha_2 z^{-2}Y[z]+\ldots+\alpha_q z^{-q}Y[z]= \\ &\beta_0 X[z]+\beta_1 z^{-1}X[z]+\beta 2 z^{-2}X[z]+\ldots+\beta_m z^{-m}X[z] \end{aligned}$$

$$\begin{aligned} &Y[z]\left[1+\alpha_1 z^{-1}+\alpha_2 z^{-2}+\ldots+\alpha_q z^{-q}\right]= \\ &X[z]\left[\beta_0+\beta_1 z^{-1}+\beta_2 z^{-2}+\ldots+\beta_m z^{-m}\right] \end{aligned} \tag{1.12}$$

$$\frac{Y[z]}{X[z]}=\frac{\beta_0+\beta_1 z^{-1}+\beta_2 z^{-2}+\ldots+\beta_m z^{-m}}{1+\alpha_1 z^{-1}+\alpha_2 z^{-2}+\ldots+\alpha_q z^{-q}}=H[z]$$

Multiplicando y dividiendo por z^q:

$$\begin{aligned} &H[z]=\frac{Y[z]}{X[z]}\frac{z^q}{z^q} \\ &H[z]=\frac{\beta_0 z^q+\beta_1 z^{q-1}+\beta_2 z^{q-2}+\ldots+\beta_m z^{q-m}}{z^q+\alpha_1 z^{q-1}+\alpha_2 z^{q-2}+\ldots+\alpha_q} \end{aligned} \tag{1.13}$$

Donde $H[z]$ es la función de transferencia del sistema discreto, que tiene la misma importancia para el análisis de sistemas discretos, así como la $G(s)$ para los sistemas continuos.

En este trabajo se utilizan funciones de transferencia $H[z]$ que cumplan con la siguiente desigualdad:

$$m \leq q \tag{1.14}$$

“La simulación mediante las funciones de transferencia representa un enfoque útil para investigar el comportamiento dinámico de los sistemas” **[Percuoco 1986]**. Es típicamente utilizada en los problemas de ingeniería y tiene aplicaciones interesantes en otros campos.

Con estos métodos se pueden representar máquinas, plantas, procesos, etcétera, mediante conjuntos de bloques. Cada bloque describe el funcionamiento de una parte del sistema y es realizado en una computadora analógica mediante una sección distinta del circuito de simulación o en una computadora digital mediante software.

De esta manera, se hace más directo el estudio del comportamiento individual de los distintos componentes del sistema a simular. Se simplifica también el ajuste de los parámetros del sistema en basándose en datos de carácter práctico.

Frecuentemente, el arreglo de los circuitos de computación o de rutinas de software, como su escalamiento en tiempo y magnitud, operaciones de revisión y corrección, etcétera, resulta más fácil de realizarse que con los métodos de simulación mediante ecuaciones diferenciales.

1.6. TEORÍA BÁSICA DE DISCRETIZACIÓN

Las computadoras digitales desempeñan una importante función en el análisis y el diseño de sistemas. No sólo se utilizan para calcular y simular el desempeño del sistema; también sirven como controladores en línea de procesos. Puesto que los sistemas continuos son los más comunes, y muchos sistemas contienen componentes de datos continuos, una práctica habitual es simularlos en una computadora digital.

"El procedimiento para hallar un equivalente digital del sistema continuo se conoce como *rediseño digital* o *discretización"* **[Kuo 2000]**. Si bien, es posible diseñar un sistema digital de manera independiente, si ya hay un sistema continuo con un desempeño satisfactorio entonces, lo más factible es encontrar un equivalente digital de él.

La simulación de un sistema continuo en una computadora digital, puede realizarse una vez que se aproxima a la dinámica del sistema con una función de transferencia en el dominio z, o por ecuaciones en diferencias. En general, el análisis se efectúa en dos pasos:

1. Representación del sistema continuo por un modelo digital.
2. Simulación del modelo digital en una computadora digital.

Es evidente, que el modelo digital también puede estudiarse analíticamente. En general, el modelado digital puede realizarse con los métodos siguientes:

1. Inserción de dispositivos de muestreo y retención en el sistema continuo.
2. Aproximación numérica de la integración continua.
3. Ecuaciones de estado discretas.

Para la finalidad de este trabajo, nos concentramos en el método de aproximación numérica de la integración continua y de éste método, desarrollamos solamente la transformación bilineal.

1.6.1. Retenedor poligonal

Parece ser que un sistema sencillo de extrapolación, desde el punto de vista de la reconstrucción de una señal continua a partir de un conjunto de muestras, es la unión de todos los puntos muestra con líneas rectas. La figura 1.6 ilustra este esquema de aproximación. Dado que la salida entre dos instantes de muestreo de un retenedor de datos con estas características tiene la forma de un polígono, este se conoce como *retenedor poligonal* o *retenedor de orden uno*. Es sencillo advertir que la salida $h(t)$ de la figura 1.6 puede considerarse como la suma de las señales rectangulares, $h_1(t)$ y $h_2(t)$, de la figura 1.7 en consecuencia, la respuesta al impulso del retenedor poligonal es la respuesta triangular de la figura 1.8 es obvio que el retenedor poligonal no es causal y, por tanto, no es posible construir físicamente un dispositivo de este tipo, ya que la salida anticipa a la entrada. Tal y como se indica en la figura 1.6, para unir los puntos muestra con líneas rectas, es necesario conocer en $t = kT$ el valor de la muestra $f[(k+1)T]$.

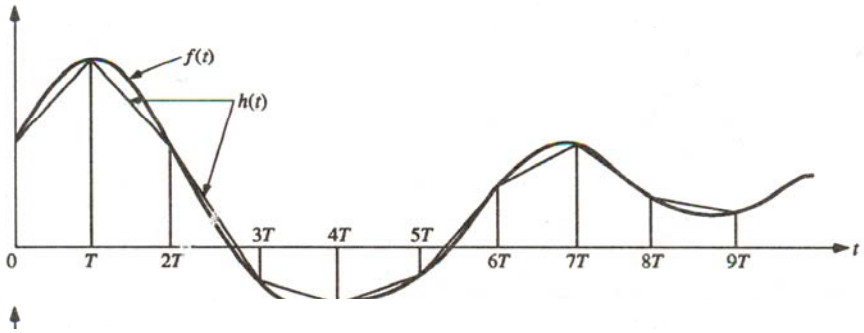


Figura 1.6. Señales de entrada y salida del retenedor poligonal

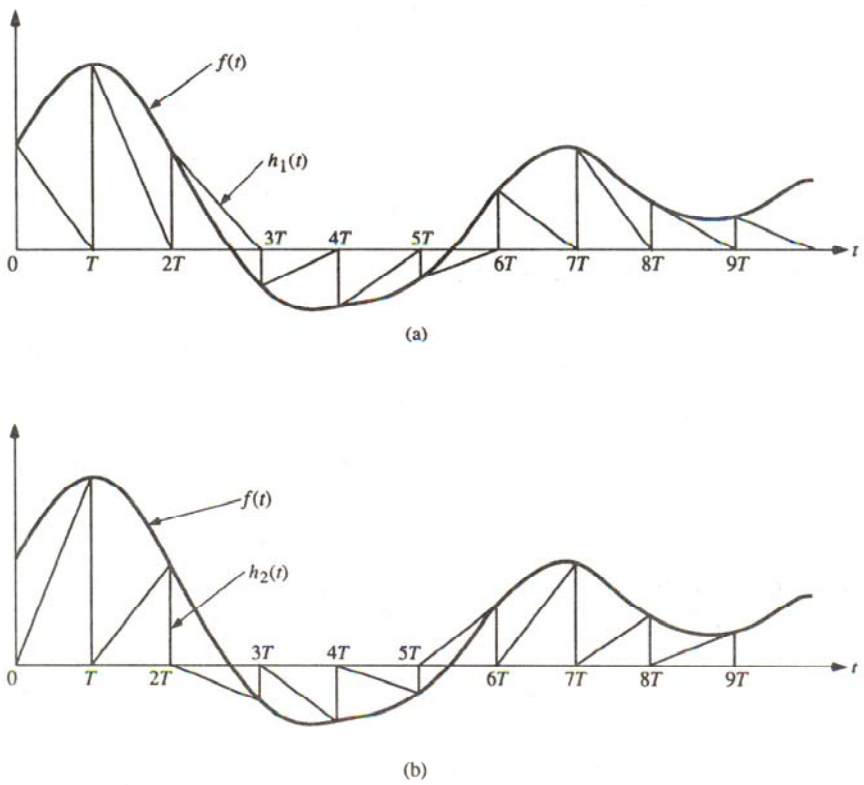


Figura 1.7. Interpretación de la salida del retenedor poligonal como la suma de dos retenedores rectangulares $h_1(t)$ y $h_2(t)$

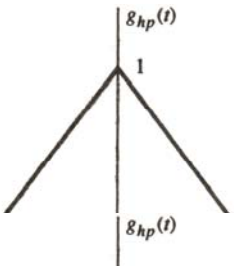


Figura 1.8. Respuesta al impulso del retenedor poligonal

La respuesta al impulso de la figura 1.8 puede escribirse como:

$$g_{hp}(t) = -\frac{(t+T)}{T}u_{-1}(t+T) - \frac{2t}{T}u_{-1}(t) + \frac{(t-T)}{T}u_{-1}(t+T) \qquad (1.15)$$

La función de transferencia del retenedor poligonal se obtiene al calcular la transformada de Laplace de $g_{hp}(t)$:

$$G_{hp}(s) = \frac{e^{Ts} + e^{-Ts} - 2}{Ts^2} \tag{1.16}$$

1.6.2. Integración poligonal (trapezoidal), transformación bilineal

Se obtiene un esquema de integración más exacto con el concepto de retenedor poligonal. Como se ve en la figura 1.9(a), el área bajo la curva $r(\tau)$ puede aproximarse sumando las áreas de varios polígonos de base T. Es evidente que la aproximación es mejor a medida que disminuye el periodo de muestreo T. Esta aproximación se conoce como *integración poligonal* o *integración trapezoidal*. Según se señala en la figura 1.9(b), este tipo de aproximación equivale a insertar un dispositivo de muestreo y retención poligonal antes de cada integrador.

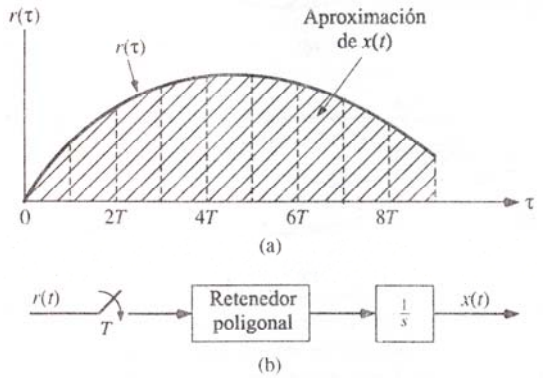


Figura 1.9. Aproximación por integración poligonal o trapezoidal

Puesto que la función de transferencia de un retenedor poligonal es la ecuación (1.16), la función de transferencia del integrador poligonal puede escribirse como:

$$\frac{X[z]}{R[z]} = \frac{z + z^{-1} - 2}{T} \Im\left(\frac{1}{s^3}\right) \tag{1.17}$$

Desarrollando la expresión (1.17) se tiene:

$$\frac{X[z]}{R[z]} = \frac{z + z^{-1} - 2}{T}\left(\frac{T^2}{\left(1-\frac{1}{z}\right)^3 z^2} + \frac{T^2}{2\left(1-\frac{1}{z}\right)^2 z} \right) = \frac{T}{2}\frac{z+1}{z-1} \tag{1.18}$$

"Es interesante hacer notar que la aproximación de la integración dada por la integración poligonal es idéntica a la *transformación r* (1.19), esto es, r corresponde a s. En consecuencia, el integrador poligonal también se denomina *aproximación por transformación bilineal"* **[Kuo 2000]**.

$$z = \frac{r+1}{r-1} \tag{1.19}$$

En la práctica, es preferible la aproximación dada por la transformación bilineal para la simulación digital de sistemas, debido a que mapea el eje $j\omega$ del plano s sobre el círculo unitario del plano z. Esto significa que, dada cualquier función de transferencia racional $G(s)$, los polos y ceros de los semiplanos izquierdo y derecho del plano s se mapean sobre los correspondientes polos y ceros dentro y fuera del plano z, respectivamente, para $H[z]$, cuando se aplica la transformación bilineal:

$$s = \frac{2}{T}\frac{z-1}{z+1} \tag{1.20}$$

Ninguno de los demás esquemas de integración numérica tiene esta propiedad.

A la ecuación (1.20) comúnmente se le denomina mapeo de Tustin.

1.6.3. Funciones de transferencia discretas, asociadas a funciones de transferencia continuas, de sistemas lineales e invariantes de primer y hasta cuarto orden

A continuación se explicará la forma de obtener una función de transferencia discreta $H[z]$ a partir de una función de transferencia continua $G(s)$.

Los pasos para obtener una función de transferencia discreta a partir de una función de transferencia continua son:

1. Obtener el modelo matemático del sistema dinámico lineal e invariante en el tiempo, en forma de ecuación diferencial.
2. Obtener la función de transferencia continua mediante la transformada de Laplace del modelo matemático obtenido.
3. Sustituir la ecuación (1.20) en la función de transferencia continua (mapeo de Tustin) para obtener la función de transferencia discreta.
4. Mediante manipulación algebraica, agrupar en términos el numerador y el denominador de la función de transferencia discreta en z^n, z^{n-1}, z^{n-2}, …, z^0.

Para satisfacer los objetivos de este trabajo, se mostrarán las ecuaciones necesarias, correspondientes a los sistemas lineales e invariantes de primer y hasta cuarto orden.

1.6.3.1. Sistema dinámico lineal e invariante de primer orden

El modelo matemático del sistema es:

$$dy(t) + c\left(\frac{d}{dt}y(t)\right) = bx(t) + a\left(\frac{d}{dt}x(t)\right) \tag{1.21}$$

La función de transferencia continua es:

$$G(s) = \frac{as + b}{cs + d} \tag{1.22}$$

La función de transferencia discreta es:

$$H[z] = \frac{b_0 z + b_1}{a_0 z + a_1} \tag{1.23}$$

Donde:

$$a_0 = 1 \tag{1.24}$$

$$a_1 = \frac{dT - 2c}{dT + 2c} \tag{1.25}$$

$$b_0 = \frac{bT + 2a}{dT + 2c} \tag{1.26}$$

$$b_1 = \frac{bT - 2a}{dT + 2c} \tag{1.27}$$

1.6.3.2. Sistema dinámico lineal e invariante de segundo orden

El modelo matemático del sistema es:

$$fy(t) + e\left(\frac{d}{dt}y(t)\right) + d\left(\frac{d^2}{dt^2}y(t)\right) = cx(t) + b\left(\frac{d}{dt}x(t)\right) + a\left(\frac{d^2}{dt^2}x(t)\right) \tag{1.28}$$

La función de transferencia continua es:

$$G(s) = \frac{as^2 + bs + c}{ds^2 + es + f} \tag{1.29}$$

La función de transferencia discreta es:

$$H[z] = \frac{b_0 z^2 + b_1 z + b_2}{a_0 z^2 + a_1 z + a_2} \tag{1.30}$$

Donde:

$$a_0 = 1 \tag{1.31}$$

$$a_1 = \frac{2\left(fT^2 - 4d\right)}{fT^2 + 2eT + 4d} \tag{1.32}$$

$$a_2 = \frac{fT^2 - 2eT + 4d}{fT^2 + 2eT + 4d} \tag{1.33}$$

$$b_0 = \frac{cT^2 + 2bT + 4a}{fT^2 + 2eT + 4d} \tag{1.34}$$

$$b_1 = \frac{2\left(cT^2 - 4a\right)}{fT^2 + 2eT + 4d} \tag{1.35}$$

$$b_2 = \frac{cT^2 - 2bT + 4a}{fT^2 + 2eT + 4d} \tag{1.36}$$

1.6.3.3. Sistema dinámico lineal e invariante de tercer orden

El modelo matemático del sistema es:

$$\begin{aligned} &hy(t)+g\left(\frac{d}{dt}y(t)\right)+f\left(\frac{d^2}{dt^2}y(t)\right)+e\left(\frac{d^3}{dt^3}y(t)\right)= \\ &dx(t)+c\left(\frac{d}{dt}x(t)\right)+b\left(\frac{d^2}{dt^2}x(t)\right)+a\left(\frac{d^3}{dt^3}x(t)\right) \end{aligned} \tag{1.37}$$

La función de transferencia continua es:

$$G(s)=\frac{as^3+bs^2+cs+d}{es^3+fs^2+gs+h} \tag{1.38}$$

La función de transferencia discreta es:

$$H[z]=\frac{b_0z^3+b_1z^2+b_2z+b_3}{a_0z^3+a_1z^2+a_2z+a_3} \tag{1.39}$$

Donde:

$$a_0=1 \tag{1.40}$$

$$a_1=\frac{3hT^3+2gT^2-4fT-24e}{hT^3+2gT^2+4fT+8e} \tag{1.41}$$

$$a_2=\frac{3hT^3-2gT^2-4fT+24e}{hT^3+2gT^2+4fT+8e} \tag{1.42}$$

$$a_3=\frac{hT^3-2gT^2+4fT-8e}{hT^3+2gT^2+4fT+8e} \tag{1.43}$$

$$b_0 = \frac{dT^3 + 2cT^2 + 4bT + 8a}{hT^3 + 2gT^2 + 4fT + 8e} \tag{1.44}$$

$$b_1 = \frac{3dT^3 + 2cT^2 - 4bT - 24a}{hT^3 + 2gT^2 + 4fT + 8e} \tag{1.45}$$

$$b_2 = \frac{3dT^3 - 2cT^2 - 4bT + 24a}{hT^3 + 2gT^2 + 4fT + 8e} \tag{1.46}$$

$$b_3 = \frac{dT^3 - 2cT^2 + 4bT - 8a}{hT^3 + 2gT^2 + 4fT + 8e} \tag{1.47}$$

1.6.3.4. Sistema dinámico lineal e invariante de cuarto orden

El modelo matemático del sistema es:

$$\begin{aligned} &my(t) + k\left(\frac{d}{dt}y(t)\right) + h\left(\frac{d^2}{dt^2}y(t)\right) + g\left(\frac{d^3}{dt^3}y(t)\right) + f\left(\frac{d^4}{dt^4}y(t)\right) = \\ &ex(t) + d\left(\frac{d}{dt}x(t)\right) + c\left(\frac{d^2}{dt^2}x(t)\right) + b\left(\frac{d^3}{dt^3}x(t)\right) + a\left(\frac{d^4}{dt^4}x(t)\right) \end{aligned} \tag{1.48}$$

La función de transferencia continua es:

$$G(s) = \frac{as^4 + bs^3 + cs^2 + ds + e}{fs^4 + gs^3 + hs^2 + ks + m} \tag{1.49}$$

La función de transferencia discreta es:

$$H[z] = \frac{b_0 z^4 + b_1 z^3 + b_2 z^2 + b_3 z + b_4}{a_0 z^4 + a_1 z^3 + a_2 z^2 + a_3 z + a_4} \tag{1.50}$$

Donde:

$$a_0 = 1 \tag{1.51}$$

$$a_1 = \frac{4\left(mT^4 + kT^3 - 4gT - 16f\right)}{mT^4 + 2kT^3 + 4hT^2 + 8gT + 16f} \tag{1.52}$$

$$a_2 = \frac{2\left(3mT^4 - 4hT^2 + 48f\right)}{mT^4 + 2kT^3 + 4hT^2 + 8gT + 16f} \tag{1.53}$$

$$a_3 = \frac{4\left(mT^4 - kT^3 + 4gT - 16f\right)}{mT^4 + 2kT^3 + 4hT^2 + 8gT + 16f} \tag{1.54}$$

$$a_4 = \frac{mT^4 - 2kT^3 + 4hT^2 - 8gT + 16f}{mT^4 + 2kT^3 + 4hT^2 + 8gT + 16f} \tag{1.55}$$

$$b_0 = \frac{eT^4 + 2dT^3 + 4cT^2 + 8bT + 16a}{mT^4 + 2kT^3 + 4hT^2 + 8gT + 16f} \tag{1.56}$$

$$b_1 = \frac{4\left(eT^4 + dT^3 - 4bT - 16a\right)}{mT^4 + 2kT^3 + 4hT^2 + 8gT + 16f} \tag{1.57}$$

$$b_2 = \frac{2\left(-3eT^4 + 4cT^2 - 48a\right)}{mT^4 + 2kT^3 + 4hT^2 + 8gT + 16f} \tag{1.58}$$

$$b_3 = \frac{4\left(eT^4 - dT^3 + 4bT - 16a\right)}{mT^4 + 2kT^3 + 4hT^2 + 8gT + 16f} \tag{1.59}$$

$$b_4 = \frac{eT^4 - 2dT^3 + 4cT^2 - 8bT + 16a}{mT^4 + 2kT^3 + 4hT^2 + 8gT + 16f} \tag{1.60}$$

En este capítulo se revisaron los conceptos básicos necesarios que permitirán al lector recordar, seguir y comprender con claridad, lo expuesto por los capítulos siguientes.

CAPÍTULO 2

ESTADO DEL ARTE EN LA SIMULACIÓN DE SISTEMAS LINEALES E INVARIANTES

En este capítulo encontrará:

2. ESTADO DEL ARTE EN LA SIMULACIÓN DE SISTEMAS LINEALES E INVARIANTES

La idea de esta sección, es presentar brevemente, las soluciones actuales para realizar simulaciones de sistemas lineales e invariantes con la finalidad de mostrar los distintos enfoques existentes, sus ventajas y desventajas.

2.1. LA COMPUTADORA ANALÓGICA

Como se comentó en el capítulo 1, es posible construir muchos tipos de modelos básicos en base a analogías. Además de modelos analógicos, hay muchos dispositivos físicos cuyo comportamiento es análogo a una operación matemática tal como la adición o la integración. Usando esos dispositivos, es posible formar computadoras analógicas para simular los sistemas. Las computadoras emplean una cantidad de dispositivos analógicos, y cada uno se utiliza para representar una operación matemática en una variable específica. La técnica de aplicar las computadoras, consiste en interconectar los dispositivos en alguna manera especificada por un modelo matemático del sistema. Aunque las computadoras analógicas se construyen de dispositivos físicos, es más exacto considerar que resuelven problemas con modelos matemáticos, que referirse a ellas como modelos físicos.

“La computadora analógica actúa basándose en la propiedad de que el funcionamiento de ciertos circuitos electrónicos, reproduce de manera natural operaciones matemáticas de varios tipos” **[Percuoco 1986]**.

La forma de computadora analógica que se utiliza más extensamente es la computadora analógica electrónica, que se basa en el uso de amplificadores de alta ganancia de corriente directa. Los voltajes en la computadora se hacen iguales a las variables matemáticas y los amplificadores de corriente directa pueden sumar

e integrar los voltajes. Con circuitos apropiados, se puede hacer que un amplificador sume varios voltajes de entrada, cada uno de los cuales, representa la suma de las variables de entrada. Se pueden utilizar distintos factores de entrada en las escalas para representar coeficientes de las ecuaciones del modelo. A esos amplificadores se les conoce como *sumadores.* Otro arreglo del circuito, produce un *integrador* para el que la salida, es la integral con respecto al tiempo de un solo voltaje de entrada o la suma de varios voltajes de entrada. Todos pueden ser positivos o negativos, para corresponder al signo de la variable representada. Para satisfacer las ecuaciones del modelo, a veces, es necesario utilizar un *inversor de signo,* que es un amplificador que se diseña para hacer que la salida cambie de signo en la entrada.

Las computadoras analógicas electrónicas, tienen exactitud limitada por varias razones. Primero, es difícil llevar la exactitud de medir un voltaje más allá de determinado punto. En segundo lugar, se hace una cantidad de suposiciones al deducir las relaciones para los amplificadores de corriente directa, ninguna de las cuales es estrictamente verdadera; por tanto, los amplificadores no resuelven el modelo matemático con total exactitud. Una suposición con especial dificultad, es que debe de haber cero salida para cero entrada. El que los amplificadores de corriente directa, tengan un rango dinámico limitado de salida, presenta otro tipo de dificultad, por lo que deben de introducirse factores de escala para mantenerse dentro de los rangos. Como consecuencia de ello, es difícil mantener una exactitud mejor que $0.1(\%)$ en una computadora analógica electrónica. Otras formas de computadoras analógicas tienen problemas semejantes y sus exactitudes no son notablemente mejores.

2.2. LA COMPUTADORA DIGITAL

“El funcionamiento de la computadora digital se basa en su capacidad de efectuar a altísima velocidad las cuatro operaciones aritméticas: suma, resta, multiplicación y división” **[Percuoco 1986]**.

Una computadora digital no está sujeta al mismo tipo de inexactitudes que la computadora analógica. Se puede programar virtualmente cualquier grado de exactitud, y usando la representación de números en punto flotante, se puede tolerar un rango sumamente amplio de variaciones. La integración de variables no es una capacidad natural de una computadora digital, contraria a la analógica, por lo que la función debe de realizarse mediante aproximaciones numéricas a través algún algoritmo. Sin embargo, se han desarrollado métodos que pueden mantener un grado muy elevado de exactitud.

La computadora digital tiene la ventaja adicional de que se puede utilizar fácilmente para muchos problemas distintos. Por lo general es necesario dedicar una computadora analógica, a una aplicación a la vez.

“Es necesario tener en mente que los sistemas digitales tienen limitaciones físicas debido a la naturaleza discreta inherente de los componentes del sistema, los cuales, no se encuentran en los sistemas analógicos; por ejemplo, el periodo de muestreo de un sistema digital de simulación está gobernado por la frecuencia del reloj y por la rapidez con que se ejecutan las instrucciones y operaciones aritméticas en el procesador digital. En el caso de un Procesador Digital de Señales (DSP, por sus siglas en inglés), la velocidad de ejecución puede ser en extremo alta; no obstante, en algunos microprocesadores, en especial los de tiempo compartido (como los de las PC’s), la velocidad de ejecución puede ser relativamente baja. En consecuencia, la frecuencia de muestreo tiene limites inherentes dictados por el hardware utilizado” **[Kuo 2000]**.

Otra limitación de las computadoras digitales, es la longitud de palabra finita de los procesadores digitales. Esto significa que el procesador digital no puede representar con precisión todos los números.

2.3. DIFERENCIAS ENTRE LA COMPUTADORA ANALÓGICA Y LA COMPUTADORA DIGITAL

"En la computadora analógica, el tiempo de la solución calculada de una ecuación coincide con el tiempo efectivo ("real" en la terminología corriente) de desarrollo del proceso que es descrito por esa ecuación. Esto es debido a que las operaciones son ejecutadas simultáneamente (en paralelo)" **[Percuoco 1986]**.

"El funcionamiento de la computadora digital es diferente. Los cálculos son efectuados numéricamente de manera secuencial y por esto, el tiempo de computación no puede coincidir con el tiempo del proceso a simular" **[Percuoco 1986]**.

Otra diferencia, la podemos encontrar en las entradas y salidas de la computadora analógica, las cuales deben estar constituidas por señales continuas, mientras que en la computadora digital están formadas de números o datos suministrados a intervalos discretos de tiempo. En el caso que el problema contenga señales continuas, como ocurre con frecuencia, éstas son puestas en forma numérica directamente en el programa escrito por el usuario o bien, mediante rutinas internas de la computadora.

La computadora analógica funciona muy bien para desarrollar operaciones del cálculo diferencial como son: integración, solución de ecuaciones diferenciales, desarrollo en serie de funciones, etcétera. Pero no resulta eficiente para cumplir cálculos de tipo numérico; raíces de ecuaciones algebraicas, solución de sistemas de ecuaciones simultáneas, búsqueda de la solución por aproximación numérica, etcétera. En cambio, la computadora digital es muy rápida y exacta en efectuar tales cálculos.

Otra diferencia importante entre ambos tipos de computadora, se refiere a la capacidad de implementar decisiones en el programa de computación. La computadora analógica puede realizar solamente pocas "decisiones", mientras que la capacidad de "decidir" entre varias alternativas que pueden presentarse en el desarrollo de un problema es

realmente grande en la computadora digital, debido a que dispone de numerosos circuitos lógicos.

2.4. COMPUTADORAS HÍBRIDAS

"Para evitar las desventajas de las computadoras analógicas se han descrito muchos sistemas de programación, conocidos como *simuladores digitales a analógicos* (computadora híbrida) que permiten programar un problema de sistema continuo a una computadora digital en esencialmente la misma manera que lo resuelve una computadora analógica" **[Gordon 1982]**.

La computadora híbrida es un sistema constituido de una computadora analógica y una computadora digital conectados a través de una interfase que permite el intercambio de información entre las dos computadoras y el desarrollo de su trabajo conjunto.

La computadora híbrida encuentra su mejor empleo en los problemas en los cuales se utilizan las características de trabajo más eficientes de cada computadora, como la capacidad de la computadora analógica para efectuar rápidamente operaciones de cálculo diferencial y la versatilidad, velocidad y exactitud de la computadora digital en el cálculo numérico.

En la computadora híbrida, "los programas mantienen las mismas técnicas generales que se desarrollaron para resolver problemas con las computadoras analógicas, pero al hacerlo, superan las desventajas de éstas" **[Gordon 1982]**.

Otras aplicaciones que hacen muy apropiado el uso de la computadora híbrida, están constituidas de las simulaciones en tiempo real, como se requiere en los sistemas hombre – máquina. Este es el caso por ejemplo, de un simulador de vuelo.

La aparición de la computadora híbrida de ninguna manera ha reemplazado a las computadoras analógicas, pues la exactitud de éstas, es suficiente para muchos

problemas. Además, con frecuencia proporcionan una manera más económica de resolver problemas, especialmente en el caso de problemas a gran escala. Resuelven las ecuaciones en una forma verdaderamente simultánea, en tanto que la computadora digital debe de resolverlas en forma secuencial, lo que a menudo da a aquellas, una considerable ventaja de velocidad y costo.

Existe otra clase de computadora, la *computadora híbrida paralela*, la cual es sustancialmente diferente de la computadora híbrida. Está basada en la integración de una computadora analógica y un sistema lógico, de manera que no hay una computadora digital que participe en los cálculos. El sistema lógico coordina y controla las unidades de computación analógica.

La computadora híbrida paralela tiene menor capacidad de cómputo que la computadora híbrida. Muchas veces los programas son ejecutados de manera automática sin intervención del usuario después de que ha efectuado el arreglo inicial de los circuitos.

2.5. LENGUAJES DE PROGRAMACIÓN PARA SIMULACIÓN DE SISTEMAS LINEALES E INVARIANTES

El que la simulación comprenda cómputos numéricos conduce en forma natural a utilizar computadoras digitales, al grado que se han diseñado muchos lenguajes de programación para realizar simulaciones. Por lo general, esos lenguajes dan al usuario, un conjunto de conceptos de modelado que se utilizan para describir el sistema, y un sistema de programación que convierte la descripción, a programa de computadora que ejecute la simulación. En consecuencia, se libera al usuario de mucho esfuerzo de programación detallada, especialmente en el caso de programas de simulación discreta, en que puede complicarse demasiado la tarea de administrar, los pasos lógicos que participan en los eventos que se ejecutan. Los sistemas se clasifican como continuos o discretos, y por lo general los lenguajes de programación se diseñan para sistemas continuos o discretos.

Ejemplos de tres lenguajes de simulación de sistemas continuos son: 1130/CSMP, 360/CSMP y DYNAMO. Para lenguajes de simulación de sistemas discretos tenemos: GPSS y SIMSCRIPT.

Existen también paquetes de software comercial que pueden clasificarse como *paquetes de cómputo* o *estuches de herramientas matemáticas*. Tales paquetes, diseñados para resolver una amplia variedad de problemas, incluyen el Derive, IMSL, Macsyma, Maple, Mathcad, Mathematica, MATLAB y TK Solver, entre muchos otros. Estos paquetes pueden subdividirse en *paquetes numéricos* y *paquetes simbólicos*. En general, los primeros evalúan numéricamente las expresiones dadas y proporcionan respuestas numéricas. En contraste, los procesadores simbólicos pueden proporcionar soluciones en forma cerrada. IMSL, Mathcad, MATLAB y TK Solver son paquetes predominantemente numéricos, en tanto que Derive, Maxima, Maple y Mathematica son, sobre todo, paquetes simbólicos. Sin embargo, la tendencia que siguen los paquetes numéricos es incluir ciertas capacidades simbólicas, y los paquetes simbólicos, en general, la de tener cierta capacidad de evaluación numérica de expresiones.

Dos paquetes frecuentemente utilizados para el análisis y simulación de sistemas de control son SIMNON y CC. El primero se utiliza para la simulación de sistemas lineales y no lineales. El segundo es ocupado para la simulación de sistemas lineales.

En el presente capítulo, se mostraron los distintos enfoques e ideas mayores existentes, para llevar acabo la simulación de sistemas lineales e invariantes, con el objetivo de que el lector conozca las formas en que se realiza, a la fecha, la simulación de los sistemas lineales, además de que identifique las ventajas y desventajas de cada uno de éstos métodos.

CAPÍTULO 3

EL PROBLEMA DEL SIMULADOR DIGITAL DE SISTEMAS LINEALES E INVARIANTES (SDSLI)

En este capítulo encontrará:

3. EL PROBLEMA DEL SIMULADOR DIGITAL DE SISTEMAS LINEALES E INVARIANTES (SDSLI)

El presente capítulo pretende mostrar una declaración concisa de la cuestión que la tesis aborda. Pretende revisar los problemas que no han sido resueltos en el estado del arte de la simulación de sistemas lineales e invariantes y además, desarrolla las razones por las cuáles es valioso resolver los problemas antes mencionadas en el capítulo 2.

3.1. EL SDSLI DE PRIMER Y HASTA CUARTO ORDEN

Como se ha visto en el capítulo 1, la importancia de la simulación en las ramas de la ingeniería y otras disciplinas, es enorme.

En particular, en teoría de señales y sistemas, sistemas dinámicos, circuitos eléctricos y electrónicos y control analógico y digital, un concepto fundamental, es el modelado y la simulación de sistemas dinámicos lineales e invariantes en el tiempo. En el proceso enseñanza - aprendizaje de estas asignaturas en la Facultad de Ingeniería de la UNAM, se requiere contar con bloques funcionales que simulen el comportamiento de un sistema dinámico real, con la finalidad de que puedan ser utilizados en prácticas estudiantiles, que ayuden a la mejor asimilación de conceptos matemáticos alrededor de la dinámica y control de los sistemas mencionados con anterioridad.

El presente trabajo muestra los resultados obtenidos en la implementación de un Simulador Digital de Sistemas Lineales e Invariantes (SDSLI) como un bloque funcional, capaz de simular los sistemas antes mencionados, que abarcan a los sistemas de primer y hasta cuarto orden.

3.2. CUESTIONES NO RESUELTAS EN EL ESTADO DEL ARTE DE LA SIMULACIÓN DE SISTEMAS DINÁMICOS LINEALES E INVARIANTES

El capítulo 2 nos ha permitido revisar las tecnologías actuales que realizan la simulación de sistemas dinámicos lineales e invariantes. Nos ha dado un panorama de los distintos enfoques existentes, así como sus ventajas y desventajas.

La Facultad de Ingeniería de la UNAM cuenta actualmente con costosos y complicados simuladores de tipo analógico (equipo analógico), que han permitido a varias generaciones de estudiantes, corroborar (con ciertas restricciones y complicaciones) la teoría vista en las asignaturas relacionadas con el tema. Además, la facultad también cuenta con muchos de los lenguajes de programación mencionados en el capítulo anterior.

Desafortunadamente, los equipos analógicos con los que cuenta la facultad, son instrumentos costosos y delicados (como todas las computadoras analógicas), cuyo manejo requiere de mucho cuidado, pues es fácil que un alambrado o disposición incorrecta del circuito los dañe, o bien, no se obtengan los resultados deseados. Su uso requiere cierto nivel de preparación, tanto por parte del profesor encargado del laboratorio como de los alumnos. El hecho de tener que realizar actividades extraordinarias a la simulación, hace que en gran medida, se pierda el enfoque de las prácticas, es decir, la comprobación de la teoría.

Por tanto, se requiere de un simulador que sea sencillo de utilizar y amigable. Su uso no debe interponerse con el objetivo principal de comprobación de la teoría vista en clase, o bien, en la experimentación e investigación de sistemas lineales. Además, el simulador debe permitir realizar simulaciones de un grupo grande de sistemas lineales, para poder desarrollar distintas prácticas o investigaciones sin tener que dedicar un simulador a una práctica, aplicación o investigación en particular, y tener que adquirir otro simulador distinto para realizar una práctica diferente, ahorrando tiempo y dinero.

El uso de los equipos de electrónica analógica con los que cuentan los laboratorios de la facultad, requieren de la estricta vigilancia por parte del profesor de laboratorio durante su colocación, alambrado y funcionamiento dentro de los laboratorios. El profesor de laboratorio, debe verificar que los alumnos tengan un correcto alambrado (en el cual es frecuente encontrar errores), para evitar daños en los equipos o accidentes. La actividad de revisión puede tornarse tardada y difícil, pues es necesario verificar que todas las brigadas tengan sus equipos correctamente alambrados y conectados, antes de iniciar la operación de simulación. Nuevamente, encontramos actividades que se interponen con el objetivo de verificación de la teoría vista en clase.

Es necesario el desarrollo de un simulador que requiera de alambrado mínimo, robusto y protegido, para evitar daños al equipo y así aprovechar el tiempo en realizar más corridas de simulación, permitiendo a los alumnos enfocarse en el objetivo de comprobación de la teoría vista en clase. Este tiempo puede ser aprovechado por el profesor del laboratorio, para resolver dudas e incluso, proponer diversos experimentos adicionales, haciendo más productivo el tiempo de laboratorio.

También es importante considerar un simulador cuyo mantenimiento sea mínimo y fácil de realizar.

MATLAB y SIMULINK son tal vez, las herramientas de simulación más poderosas con las que cuenta la facultad para apoyar a los alumnos en las prácticas de simulación de sistemas.

Por una parte, MATLAB requiere que el usuario (en este caso el alumno o el investigador) conozca la forma de trabajar con el software, sus comandos y su sintaxis, además de algunas horas de entrenamiento para poder desarrollar programas que desempeñen las simulaciones deseadas, y permitan realizar los experimentos que lleven a comprobar o refutar la teoría. Además, MATLAB requiere de equipo de cómputo relativamente moderno y con suficiente capacidad, para poder ejecutar sus últimas versiones.

Como ya se explicó en la sección 1.3.3 del capítulo 1, la producción de un modelo de un sistema, la construcción del programa de computadora para la simulación del modelo y la validación del modelo, son tareas que están íntimamente ligadas y muchas veces se efectúan en paralelo, pues hay ocasiones en que una tarea no se puede realizar sin la otra y viceversa. El alumno o el investigador que desee realizar la simulación de sus modelos en algún lenguaje de programación, deberá tener una buena idea de la programación y evitar cometer errores, para así poder avanzar y verificar sus modelos con éxito. Muchas veces, el hecho de tener errores en el programa que realiza la simulación, retrasa y hace perder el objetivo de comprobación de la teoría, enfocando la atención del alumno o del investigador en un problema de programación fuera de los objetivos principales.

Por otra parte, SIMULINK posee módulos que permiten al estudiante o al investigador realizar simulaciones mediante la unión de bloques funcionales, a los cuales se les puede asignar una función de transferencia y conectar instrumentos virtuales para observar en la pantalla de la computadora el comportamiento del sistema, evitando un poco, la tarea de programación y los errores.

Los lenguajes de programación presentan actualmente una enorme desventaja, pues son ejecutados en computadoras digitales que poseen sistemas operativos multitarea (como Windows), que atienden varios procesos a la vez y hacen lenta la simulación y muy difícil de obtener simulaciones en tiempo “real”, debido a que es complicado obtener tiempos de muestreo demasiado pequeños. Además, si se desea obtener una señal eléctrica de la respuesta del sistema simulado, o bien, excitar el sistema con una señal eléctrica externa a la computadora digital, se requieren dispositivos que se deben conectar al bus de la computadora digital, los cuales en muchas ocasiones, son costosos, delicados, difíciles de conseguir y requieren del equipo de cómputo más moderno.

Por lo general, se considera que los módulos de los programas de software de cómputo disponibles en el mercado están verificados, y se utilizan de *buena fe*, creyendo que de

hecho lo están. La programación con tales paquetes simplemente supone reunir en cierta secuencia lógica una colección de los módulos relevantes disponibles en el paquete. La facilidad con la que esto puede hacerse, varía de acuerdo con el paquete de software, así como el problema a simular.

Es prioritario el desarrollo de un simulador que evite al alumno o al investigador, la tediosa tarea de programar, pues finalmente, lo que se desea es simular y no programar, además de dotar al simulador de herramientas que realicen la captura y almacenamiento de datos, que ayuden en el análisis posterior de los mismos. También es de gran importancia, permitir que de una simulación, se pueda obtener una señal eléctrica de salida y que a su vez, se pueda excitar al sistema con una señal eléctrica de entrada, teniendo la posibilidad de ser observado esto, en instrumentos como el osciloscopio. Así entonces, no sólo es posible obtener señales almacenadas en la computadora, sino también de manera física. Es importante desarrollar un simulador sencillo y que no requiera equipo de cómputo demasiado moderno, pues no debe consumir demasiados recursos.

La idea de poder tener señales eléctricas de entrada y de salida en el simulador, puede permitir llevarlo a realizar tareas de control, con sólo configurar el controlador que se desee simular.

El problema del tiempo de muestreo que se observa en las computadoras digitales, como las computadoras de escritorio, es posible resolverlo si se dedica una pequeña computadora digital, a la tarea exclusiva de la simulación. La computadora digital se puede implementar con un microprocesador, un microcontrolador o bien un DSP.

No pretendemos desplazar a los equipos analógicos ni al software que posee la Facultad de Ingeniería de la UNAM, sino complementar las herramientas de simulación, con un simulador digital que resuelva los problemas aquí expuestos, teniendo en cuenta las limitaciones propias de una computadora digital. Pensamos que es de importancia el

desarrollar un simulador de tipo digital para complementar los laboratorios de esta institución.

3.3. VALOR DE LA SOLUCIÓN DE LOS PROBLEMAS EXPUESTOS

Consideramos de gran valor el desarrollo de un simulador digital de sistemas lineales e invariantes, pues creemos que nuestra solución, permite desarrollar simulaciones de sistemas lineales de manera muy sencilla y directa, ahorrando tareas de alambrado de circuitos, evitando el desarrollo de programas de computadora y permitiendo almacenar datos de una simulación para su análisis posterior. El estudiante o el investigador lo único que van a observar, es un bloque funcional que lleva a cabo la simulación sin tener que preocuparse de la manera en que ésta se lleva a cabo.

Al tener una computadora digital totalmente dedicada a una simulación, es posible obtener tiempos de muestreo relativamente pequeños, que incluso, satisfagan estándares industriales, objetivo que con las computadoras digitales de escritorio, aún con el hardware apropiado, es difícil de lograr, debido a que el sistema operativo que se ejecuta en éstas, atiende varios procesos a la vez y no se dedica solamente a la simulación.

Otra característica de importancia, es el hecho de poder introducir excitaciones en forma de señales eléctricas al sistema que se simula, y poder obtener como salida, una señal eléctrica que representa a la respuesta del sistema. Esto último, permite al alumno y al investigador, poder tener señales físicas de su sistema simulado y así, poder realizar conexiones con otros dispositivos electrónicos o eléctricos (con las interfases apropiadas), y desarrollar una gran variedad de experimentos y aplicaciones como por ejemplo, controladores de tipo digital.

Con este simulador, pretendemos aportar a los laboratorios de la Facultad de Ingeniería de la UNAM, una herramienta de simulación digital (tomando en cuenta las limitaciones inherentes de las computadoras digitales), que complemente a los simuladores

analógicos que posee esta institución, de tal manera, que los alumnos e investigadores realicen las simulaciones que requieran, con la herramienta de simulación más adecuada.

3.4. METODOLOGÍA

El análisis anterior, arroja los problemas no resueltos en la cuestión de la simulación de sistemas lineales e invariantes, además de los requerimientos que el SDSLI debe satisfacer, para formar parte de las herramientas de simulación de la Facultad de Ingeniería.

Debido a que se desarrolló un dispositivo electrónico con su respectivo software, fue necesario proceder sistemáticamente en el análisis y diseño de sistemas de información, para poder desarrollar el SDSLI. El marco de referencia para el enfoque sistemático, es proporcionado por lo que es llamado el *ciclo de vida del desarrollo de sistemas* (SDLC por sus siglas en inglés). El SDLC es un enfoque por fases del análisis y diseño, que sostiene que los sistemas son desarrollados de mejor manera mediante el uso de un ciclo específico de actividades del analista y el usuario. Este puede ser dividido en siete fases secuenciales, aunque en realidad las fases están interrelacionadas y frecuentemente se llevan a cabo simultáneamente. Las siete fases son: identificación de problemas, oportunidades y objetivos, determinación de los requerimientos de información, análisis de las necesidades del sistema, diseño del sistema recomendado, desarrollo y documentación, prueba y mantenimiento del sistema e implementación del mismo.

Figura 3.1. Las siete fases del ciclo de vida del desarrollo de sistemas

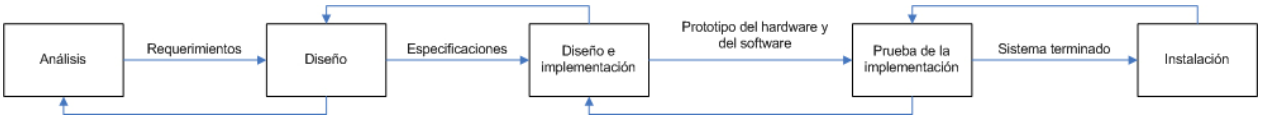


Figura 3.2. Ciclo de vida tradicional del desarrollo de sistemas

Paralelamente a la metodología de análisis y diseño estructurado anteriormente mencionada, se utilizó la metodología de *elaboración de prototipos*, utilizada como respuesta al tiempo de desarrollo largo, asociado con el enfoque del ciclo de vida del desarrollo de sistemas y a la incertidumbre que frecuentemente rodea los requerimientos de los usuarios.

Los enfoques utilizados para la elaboración de prototipos fueron:

1. **Prototipo parchado.** Se construyeron sistemas que trabajan, pero que estaban parchados. En ingeniería a este enfoque se le conoce como breadboarding (creación de un modelo operable y parchado de circuitos electrónicos).

2. **Prototipo primero de una serie.** Creación de un primer modelo a escala completa del sistema, llamado a veces piloto. Este tipo de prototipo es útil cuando se tienen planeadas muchas instalaciones del mismo sistema de información. El modelo funcional a escala completa permite la interacción realista

con el nuevo sistema y minimiza el costo de superar cualquier problema que presente.

3. **Prototipo de características seleccionadas.** Se construyeron modelos operacionales de software que incluían algunas, pero no todas, las características esenciales que tiene el modelo final. El software fue construido por módulos, de tal manera que si las características del módulo recibían una evaluación satisfactoria, éstas podían incorporarse al sistema final, mucho más grande sin tener que hacer un trabajo inmenso en interfaces.

El proyecto se desarrolló siguiendo las siguientes etapas:

Etapa 1

En esta etapa se implementaron y calibraron los circuitos electrónicos de entrada y salida empleando para ello tarjetas para alambrado experimental de circuitos electrónicos, tarjetas Project Board.

Para la computadora monotablilla requerida, se empleó una tarjeta elaborada previamente por el M. I. Antonio Salvá Calleja, la tarjeta FACIL_11B.

Etapa 2

En esta etapa se desarrollaron versiones piloto del software que debe ejecutar la computadora monotablilla implicada.

Para cargar los programas de prueba en la computadora monotablilla, se empleó software desarrollado previamente por el M. I. Antonio Salvá Calleja, el software PUMMA_11, y un compilador cruzado de lenguaje C comercial (ICC11), que genera código ejecutable en el procesador de la computadora monotablilla.

Etapa 3

Se desarrolló la interfaz de usuario para una PC que ejecuta Windows y que permite controlar con facilidad el hardware del SDSLI. La interfaz se desarrollo en el lenguaje Visual Basic.

En esta etapa se probaron y se hicieron los ajustes necesarios del software y hardware utilizando la metodología de prototipos.

Etapa 4

Se diseñó el circuito impreso del hardware de entrada y de salida elaborado en la etapa 1. Se diseñó la fuente de alimentación y su circuito impreso correspondiente. El diseño de los circuitos impresos, se realizó en TANGO.

Se contrató un proveedor externo para la manufactura de los circuitos impresos aquí mencionados.

Etapa 5

Se diseñó y construyó el gabinete que contiene el prototipo desarrollado del SDSLI.

Etapa 6

Se elaboraron los manuales técnicos y del usuario del sistema.

En el presente capítulo, se mostró una declaración concisa de la cuestión que la tesis aborda, también, se justificó por referencia directa al capítulo 2, en cuanto a las cuestiones que no han sido resueltas en el tema de la simulación de sistemas lineales, sobre todo, en la Facultad de Ingeniería de la UNAM. Finalmente, se entrega una breve

discusión de por qué es valioso resolver la cuestión planteada en este capítulo, así como la metodología que se siguió para resolver el problema del SDSLI.

CAPÍTULO 4

IMPLEMENTACIÓN DEL SIMULADOR DIGITAL DE SISTEMAS LINEALES E INVARIANTES CON INTERFAZ DE CONFIGURACIÓN EN AMBIENTE WINDOWS

En este capítulo encontrará:

4. IMPLEMENTACIÓN DEL SIMULADOR DIGITAL DE SISTEMAS LINEALES E INVARIANTES CON INTERFAZ DE CONFIGURACIÓN EN AMBIENTE WINDOWS

En este apartado se describe la solución propuesta al problema del Simulador Digital de Sistemas Lineales e Invariantes (SDSLI).

A continuación, se reportan los resultados obtenidos en la implementación de un simulador digital de sistemas dinámicos lineales e invariantes.

4.1. INTRODUCCIÓN

La implementación de un simulador digital requirió del empleo de una computadora digital, la cual realiza en tiempo real, la función de transferencia $H[z]$ asociada a la discretización de la función de transferencia $G(s)$, correspondiente al sistema dinámico analógico que se pretende simular.

La figura 4.1 muestra un esquema global del simulador.

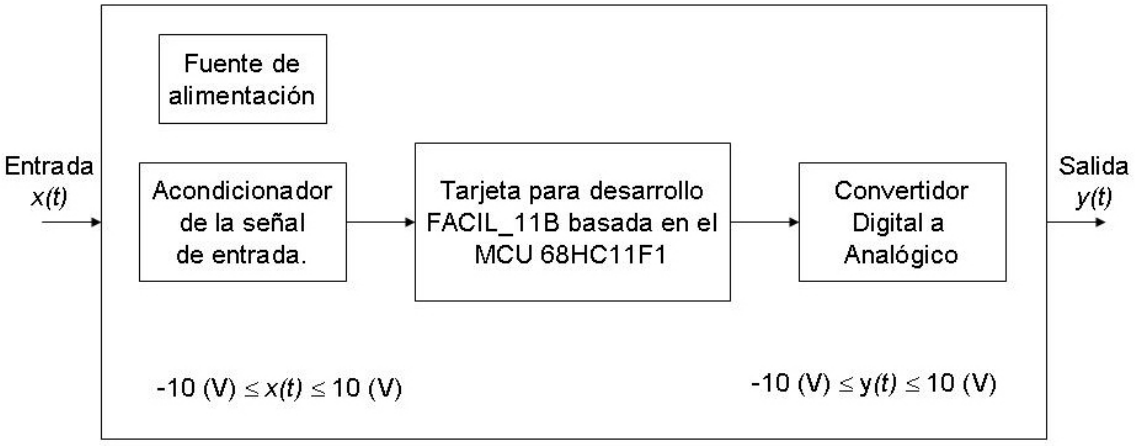


Figura 4.1. Representación en forma de bloques del SDSLI

El simulador digital está implementado a partir de un microcontrolador 68HC11F1 operando en modo expandido.

Para la etapa de conversión Analógica a Digital, se empleó el convertidor Analógico a Digital propio del microcontrolador. Para la fase de conversión Digital a Analógica, fue necesario construir hardware adicional a la tarjeta de desarrollo FACIL11B. Tanto para el convertidor Analógico a Digital como para el convertidor Digital a Analógico, se utilizaron 8 bits de resolución. Los rangos analógicos de entrada y salida están comprendidos entre ± 10 (V).

En la figura 4.2, se puede apreciar mediante un diagrama de bloques simple el proceso que lleva acabo el simulador digital

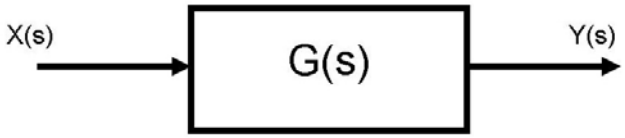


Figura 4.2. Diagrama de bloques de un sistema lineal

Mediante el software adecuado, con el SDSLI, es posible simular funciones de transferencia de la siguiente forma:

$$G(s) = \frac{e^{-T_r s}\left(b_n s^n + b_{n-1}s^{n-1} + \ldots + b_1 s + b_0\right)}{s^n + a_{n-1}s^{n-1} + \ldots + a_1 s + a_0} \tag{4.1}$$

Donde:

$$0 \le n \le 4 \tag{4.2}$$

Conforme aumenta el orden del sistema a simular, los requerimientos de memoria y el tiempo de ejecución, aumentan.

A la fecha de elaboración de esta tesis, se probó el hardware y software que validan la simulación de funciones de transferencia que pueden presentar una de las siguientes cuatro formas:

Forma para la función de transferencia continua de primer orden:

$$G(s) = \frac{e^{-T_r s}(as + b)}{cs + d} \tag{4.3}$$

Forma para la función de transferencia continua de segundo orden:

$$G(s) = \frac{e^{-T_r s}(as^2 + bs + c)}{ds^2 + es + f} \tag{4.4}$$

Forma para la función de transferencia continua de tercer orden:

$$G(s) = \frac{e^{-T_r s}(as^3 + bs^2 + cs + d)}{es^3 + fs^2 + gs + h} \tag{4.5}$$

Forma para la función de transferencia continua de cuarto orden:

$$G(s) = \frac{e^{-T_r s}(as^4 + bs^3 + cs^2 + ds + e)}{fs^4 + gs^3 + hs^2 + ks + m} \tag{4.6}$$

Para todos los casos aquí expuestos y para fines de este trabajo, se tiene que el tiempo de retardo T_r, deberá ser un múltiplo entero del periodo de muestreo T, esto es:

$$T_r = kT \tag{4.7}$$

Donde k es un entero cuyo rango de valores dependerá del software y los recursos del hardware empleado para llevar a cabo una simulación. Para efectos del simulador aquí expuesto, k es un valor de tipo entero que estará comprendido entre 0 y 54.

Es momento apropiado de mencionar que el software de la interfaz del SDSLI, tiene la posibilidad de recomendar al usuario un tiempo de muestreo cuyo valor se calcula mediante el siguiente criterio empírico.

Para el caso de un sistema de primer orden, el cual sólo presenta un polo real, el periodo de muestreo asignado, es la décima parte de la constante de tiempo asociada al sistema.

Para el caso de un sistema de segundo orden, cuando éste presenta polos reales, el periodo de muestreo asignado, corresponde a la décima parte de la constante de tiempo más pequeña. Si el sistema tiene polos complejos, entonces el periodo de muestreo se calcula tomando la décima parte del periodo de oscilación amortiguada de la respuesta al escalón unitario del sistema a simular.

En los casos de un sistema de tercer y cuarto orden cuando presenten polos reales, el periodo de muestreo calculado, será la décima parte de la constante de tiempo más pequeña. Si todos los polos de este tipo de sistemas son complejos, entonces el periodo de muestreo corresponde a la décima parte del periodo de oscilación amortiguada de la respuesta al escalón unitario del sistema a simular. Finalmente, si existen combinaciones de polos reales y polos complejos, entonces el periodo de muestreo es calculado en función del polo dominante del sistema. Si el polo dominante del sistema es un polo real, entonces el tiempo de muestreo calculado es la décima parte de la constante de tiempo calculada con este polo. Por otro lado, si se trata de un polo dominante de tipo complejo, entonces el periodo de muestreo se obtiene tomando

la décima parte de la oscilación amortiguada de la respuesta a escalón unitario del sistema a simular calculada con este polo.

El cálculo de la función de transferencia discreta asociada con la simulación, requirió del mapeo de Tustin, explicado brevemente en la ecuación (1.20) del capítulo 1.

El mapeo de Tustin de la función de transferencia de un sistema de primer orden en el dominio de s, de la forma expresada por la ecuación (4.3), da como resultado a la siguiente función de transferencia de un sistema de primer orden en el dominio de z:

$$H[z] = \frac{b_0 z + b_1}{z^k \left(a_0 z + a_1\right)} \tag{4.8}$$

Donde los valores de a_0, a_1, b_0 y b_1 es posible calcularlos correspondientemente, con las ecuaciones (1.24), (1.25), (1.26) y (1.27), expuestas en el capítulo 1.

Cuando el sistema que se desea simular tiene una función de transferencia continua de la forma dada por la ecuación (4.4), la correspondiente función de transferencia discreta es la siguiente.

$$H[z] = \frac{b_0 z^2 + b_1 z + b_2}{z^k \left(a_0 z^2 + a_1 z + a_2\right)} \tag{4.9}$$

Donde los valores de a_0, a_1, a_2, b_0, b_1 y b_2 es posible obtenerlos correspondientemente, con las ecuaciones (1.31), (1.32), (1.33), (1.34), (1.35) y (1.36), presentadas en el capítulo 1.

Si el sistema que se desea simular tiene una función de transferencia continua de la forma dada por la ecuación (4.5), la función de transferencia discreta asociada, es la que a continuación se muestra:

$$H[z] = \frac{b_0 z^3 + b_1 z^2 + b_2 z + b_3}{z^k \left(a_0 z^3 + a_1 z^2 + a_2 z + a_3 \right)} \tag{4.10}$$

Donde los valores de a_0, a_1, a_2, a_3, b_0, b_1, b_2 y b_3 es posible calcularlos correspondientemente, con las ecuaciones (1.40), (1.41), (1.42), (1.43), (1.44), (1.45), (1.46) y (1.47), descritas previamente en el capítulo 1.

Finalmente, para el sistema a simular que tenga una función de transferencia continua de la forma dada por la ecuación (4.6), la función de transferencia discreta relacionada es la siguiente:

$$H[z] = \frac{b_0 z^4 + b_1 z^3 + b_2 z^2 + b_3 z + b_4}{z^k \left(a_0 z^4 + a_1 z^3 + a_2 z^2 + a_3 z + a_4 \right)} \tag{4.11}$$

Donde los correspondientes valores de a_0, a_1, a_2, a_3, a_4, b_0, b_1, b_2, b_3 y b_4 se calculan con las ecuaciones (1.51), (1.52), (1.53), (1.54), (1.55), (1.56), (1.57), (1.58), (1.59) y (1.60), detalladas con anterioridad en el capítulo 1.

A continuación, se presenta la tabla 4.1 para diversos valores de k, de los tiempos de ejecución experimental del procesamiento aritmético requerido para un sistema de primer orden, utilizando el hardware y el software desarrollado:

Tabla 4.1

k	Tiempo de procesamiento (ms)
0	4.5
5	4.7
10	4.9
15	5.1
20	5.3
25	5.5
30	5.7
35	5.9
40	6.1
45	6.3
50	6.5
54	6.6

Tabla 4.1. Tiempos de ejecución para diversos valores de k, correspondientes a un sistema de primer orden

De la misma manera, la tabla 4.2 muestra los tiempos para un sistema de segundo orden:

Tabal 4.2

k	Tiempo de procesamiento (ms)
0	5.5
5	5.7
10	5.9
15	6.1
20	6.3
25	6.5
30	6.7
35	6.9
40	7.1
45	7.3
50	7.5
54	7.6

Tabla 4.2. Tiempos de ejecución para diversos valores de k, correspondientes a un sistema de segundo orden

La tabla 4.3 a su vez, muestra los tiempos de ejecución experimental para un sistema de tercer orden.

Tabla 4.3

k	Tiempo de procesamiento (ms)
0	6.5
5	6.7
10	6.9
15	7.1
20	7.3
25	7.5
30	7.7
35	7.9
40	8.1
45	8.3
50	8.5
54	8.6

Tabla 4.3. Tiempos de ejecución para diversos valores de k, correspondientes a un sistema de tercer orden

Finalmente, la tabla 4.4 aporta los tiempos de ejecución experimentales de un sistema de cuarto orden:

Tabla 4.4

k	Tiempo de procesamiento (ms)
0	7.5
5	7.7
10	7.9
15	8.1
20	8.3
25	8.5
30	8.7
35	8.9
40	9.1
45	9.3
50	9.5
54	9.6

Tabla 4.4. Tiempos de ejecución para diversos valores de k, correspondientes a un sistema de cuarto orden

El SDSLI está pensado para simular sistemas cuyas constantes de tiempo se encuentren comprendidas entre $250(ms)$ y hasta varios minutos. En la práctica, para los sistemas con dinámicas descritas por las ecuaciones (4.3), (4.4), (4.5) y (4.6), los

tiempos de retardo se ubican entre 10 y 20 por ciento de la constante de tiempo más grande. Por lo tanto, es posible apreciar que la velocidad del microprocesador elegido para la implementación del SDSLI es adecuada, esto de acuerdo con los tiempos mostrados en las tablas 4.1, 4.2, 4.3 y 4.4.

4.2. DISEÑO DEL HARDWARE DEL SDSLI

Los bloques de hardware utilizados en la realización del simulador, son los siguientes:

1) Tarjeta FACIL_11B para desarrollo con el microcontrolador 68HC11F1. Esta tarjeta, fue creada en la Facultad de Ingeniería de la UNAM por el M.I. Antonio Salvá Calleja.
2) Tarjeta SDSLI_B integrada por los siguientes circuitos:
    a. Circuito de adecuación de la señal de entrada al simulador.
    b. Circuito convertidor Digital a Analógico, mediante el cual se implementó el bloque funcional que genera la señal de salida del simulador digital.
    c. Fuente de alimentación, por medio de la cual, se energizan todos los circuitos que componen el simulador.

4.2.1. Tarjeta FACIL_11B

Las características principales de la tarjeta FACIL_11B, arquitectura en la cual el simulador esta basado, son:

1. Está basada en el microcontrolador 68HC11F1 y puede operar en cualquiera de los cuatro modos asociados con el 68HC11 en general.
2. Contiene un firmware interlocutor que permite manejarla, vía enlace serie, desde una PC mediante la ejecución del software clásico para este fin como PCBUG11, o bien, del manejador visual PUMMA_11, que corre bajo WINDOWS y contempla los

comandos de manejo más usuales, además de algunos otros adicionales asociados con características propias de la tarjeta FACIL_11B.

3. Es compatible con otras herramientas de software asociadas con el microcontrolador 68HC11, lo cual permite la ejecución en la tarjeta, de los programas originalmente escritos en lenguaje C o ensamblador, lográndose esto, mediante la carga y ejecución del archivo objeto S19 que haya sido generado por el software de ensamble o compilación respectivo.
4. Tiene la capacidad para configurar diversos mapas de memoria al operar en modo expandido; por ejemplo: 8 (Kb) de RAM y 8 (Kb) de EPROM, o bien, 32 (Kb) de RAM y 32 (Kb) de EPROM.
5. Puede configurarse para ser alimentada eléctricamente, con una fuente de laboratorio de cinco volts, o bien, con un eliminador de baterías.
6. Contiene postes para conexión de unidades desplegadoras alfanuméricas comunes en la industria, así como también, postes para conexión de teclados matriciales de 4 x 4.
7. Posee dos puertos de entrada y dos puertos de salida visibles al operar en modo expandido, además de líneas de paginación de puerto adicionales, que permiten al usuario experimentar con la conexión de dispositivos externos tales como: puertos serie o paralelos adicionales, chips de reloj, o hardware específico diseñado para una determinada aplicación.
8. Contiene un circuito para respaldo de memoria RAM externa al microcontrolador (esto requiere una batería de 3 a 4.5 (V) conectada al conector 10).
9. Además, es posible utilizar el programador integrado de memorias EPROM manejado por opciones especiales del software PUMMA_11. Las memorias que pueden ser programadas son: 27C64, 27C128, 27C256 y 27C512. Esta característica, permite efectuar el desarrollo de una aplicación desde la prueba y depuración del software asociado con la misma, hasta la programación final de la memoria EPROM requerida para almacenar el código correspondiente para la ejecución autónoma, todo en un solo equipo.

A continuación, se muestra la tarjeta FACIL_11B:

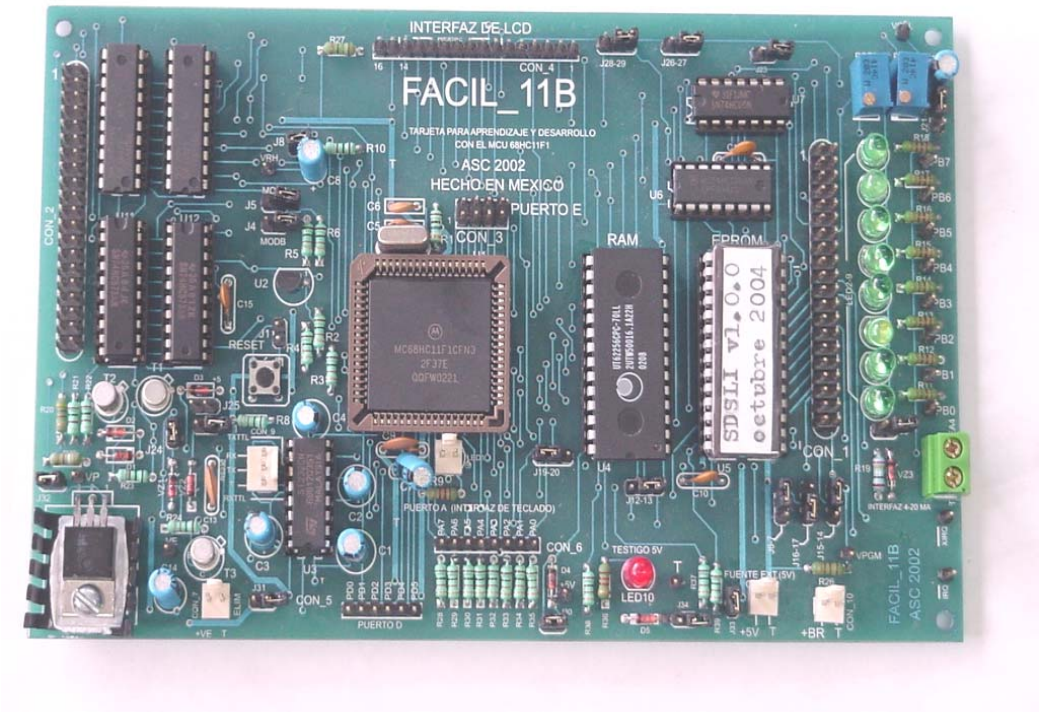


Figura 4.3. Tarjeta FACIL_11B

4.2.2. Tarjeta SDSLI_B

Como se mencionó con anterioridad, está formada por tres circuitos:

1. Circuito acondicionador de la señal de entrada.
2. Circuito del convertidor Digital a Analógico de 8 bits.
3. Circuito de la fuente de alimentación.

En la siguiente figura, se puede observar el circuito impreso de la tarjeta SDSLI_B.

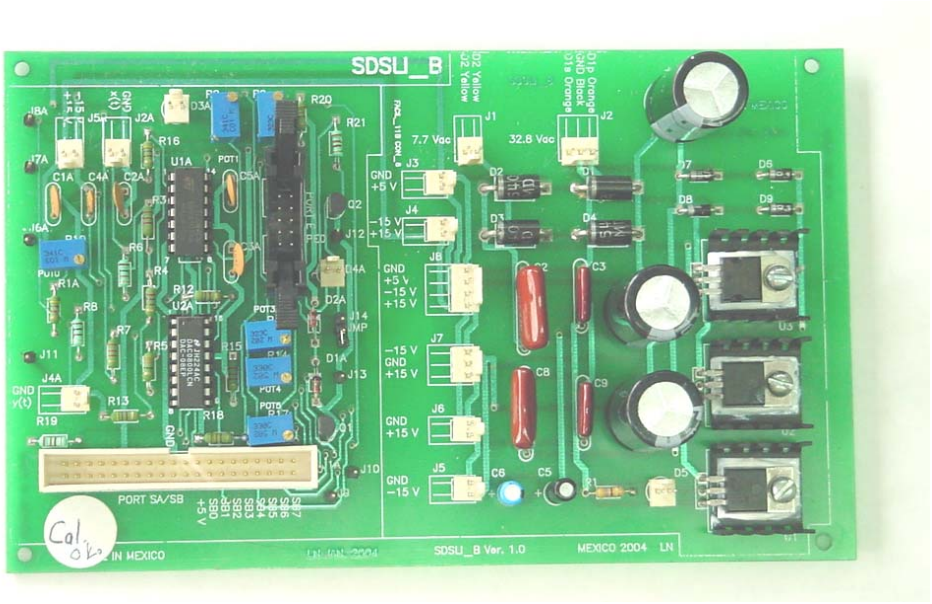


Figura 4.4. Tarjeta SDSLI_B

4.2.2.1. Circuito de adecuación de la señal de entrada

Debido a que el convertidor Analógico a Digital, interno del microcontrolador 68HC11, integrado en la tarjeta FACIL_11B, requiere señales cuyo voltaje estén en el intervalo de 0 a 5 (V), se necesita acondicionar la señal de entrada $x(t)$, ubicada en un rango de -10 a 10 (V) para que se ajuste a los valores de voltaje solicitados por el convertidor Analógico a Digital interno del microcontrolador. Esto es precisamente de lo que se encarga el circuito de acondicionamiento de la señal de entrada al SDSLI.

El circuito empleado para realizar el acondicionador de la señal de entrada se muestra en la figura 4.5, en ella, se aprecian dos resistencias variables (potenciómetros), R2 y R9, las cuales se ajustan de tal modo que la característica de entrada / salida de este bloque, sea la mostrada en la figura 4.6

En la figura 4.5, también es posible visualizar una pequeña tabla que muestra los valores críticos de calibración de la señal de entrada y de la señal de salida de este circuito.

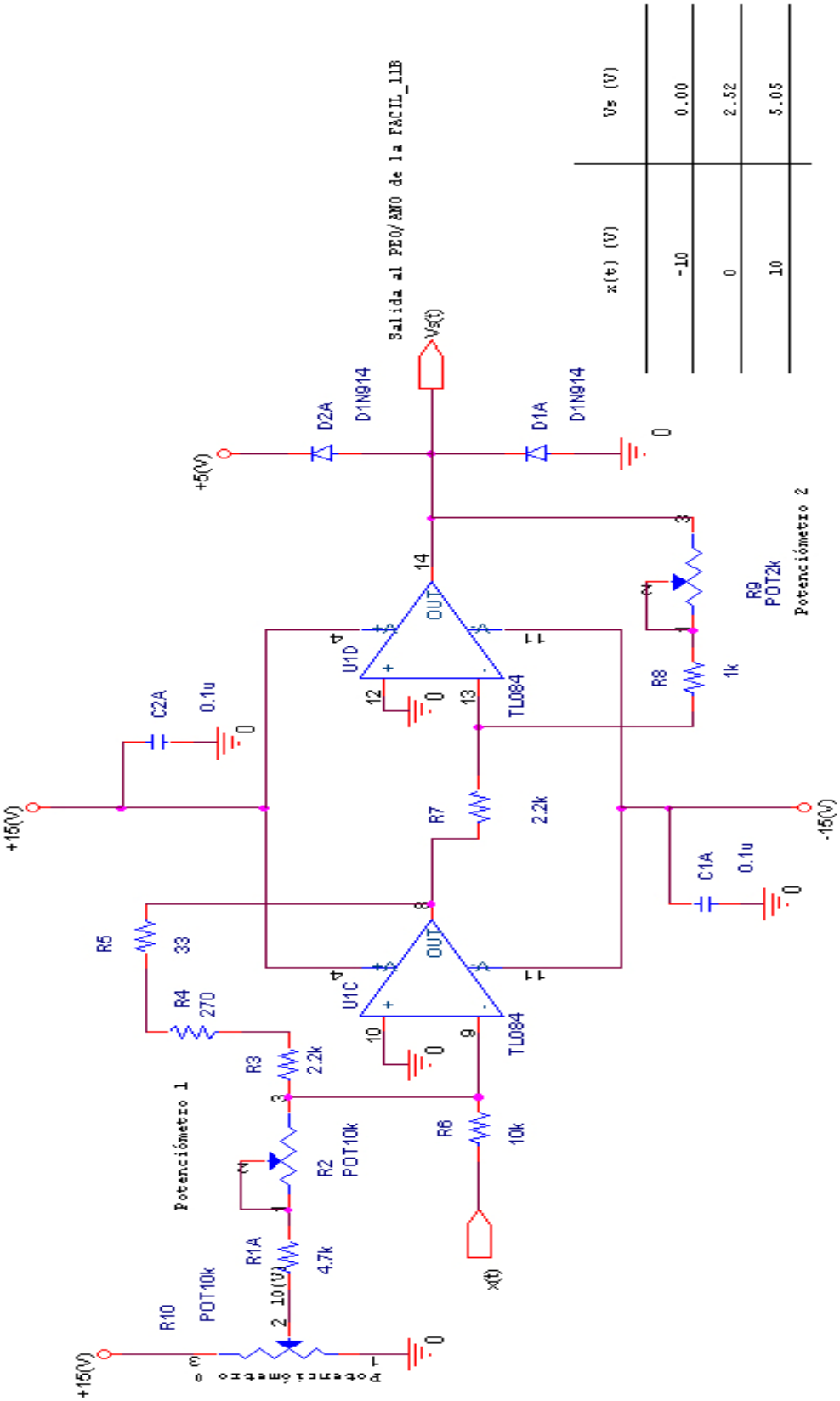


Figura 4.5. Circuito acondicionador de la señal de entrada al SDSLI

En el conector J2A, de la tarjeta SDSLI_B, se introduce la señal de entrada $x(t)$. El valor de voltaje de esta señal, es sumado a un voltaje constante de 10 (V) y multiplicado por una ganancia de 0.25, de tal forma, que es posible obtener en el conector J3A ($Vs(t)$) de la tarjeta SDSLI_B, la señal apropiada para el convertidor Analógico a Digital interno del microcontrolador, cuya entrada corresponde al canal AN_0 del microcontrolador, que físicamente se ubica en el pin 8 del conector CON_3 de la tarjeta FACIL_11B.

La figura 4.6, muestra la curva característica de operación del circuito de acondicionamiento de la señal de entrada $x(t)$ al SDSLI. El eje de las abscisas, representa los valores posibles de voltaje de entrada (Ve) de la señal de entrada $x(t)$, y el eje de las ordenadas, muestra los valores correspondientes al voltaje de salida (Vsalida o $Vs(t)$) del circuito de acondicionamiento de la señal de entrada.

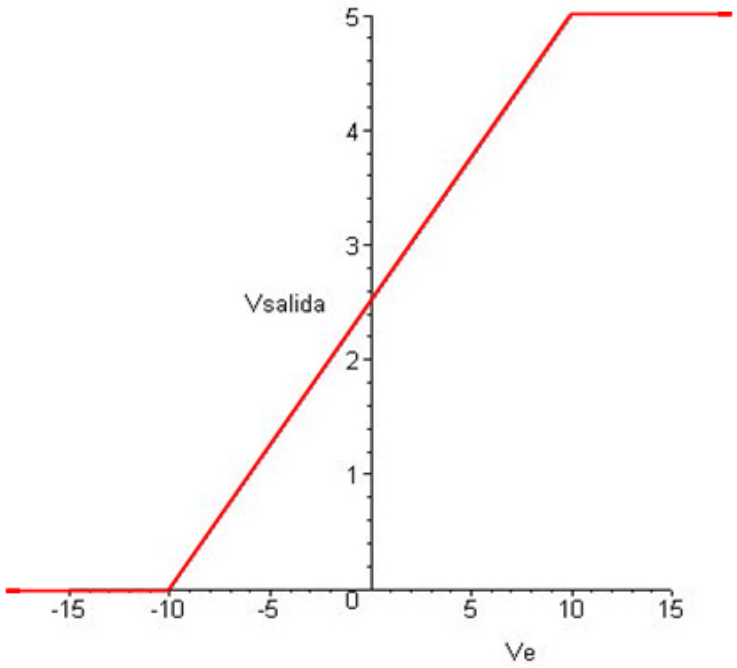


Figura 4.6. Curva característica de operación del circuito de acondicionamiento de la señal de entrada al SDSLI

La ecuación matemática que representa a la curva característica es:

$$Vsalida = \frac{Ve}{4} + 2.5 \quad (\mathrm{V}) \tag{4.12}$$

Donde:

$$-10(V) \leq Ve \leq +10(V) \tag{4.13}$$

4.2.2.2. Circuito convertidor Digital a Analógico

Una vez que el microcontrolador ha obtenido, mediante la ecuación en diferencias, el valor de $y[n]$ en un formato de 8 bits, este resultado es enviado al puerto de salida SB (CON_2) de 8 bits, propio de la tarjeta FACIL_11B (dirección 1980h), el cual a su vez, es enviado por cable al conector J1A, de la tarjeta SDSLI_B. El convertidor Digital a Analógico, toma el valor de 8 bits y lo transforma a un valor de voltaje de salida $y(t)$ (señal de salida del SDSLI), dentro de un intervalo de -10 a 10 (V).

Para mayores detalles acerca de la estructura de puertos de la tarjeta FACIL_11B, puede consultarse **[A. Salvá 2002]**.

El corazón del circuito de conversión Digital a Analógica, es el circuito integrado DAC0800 de National.

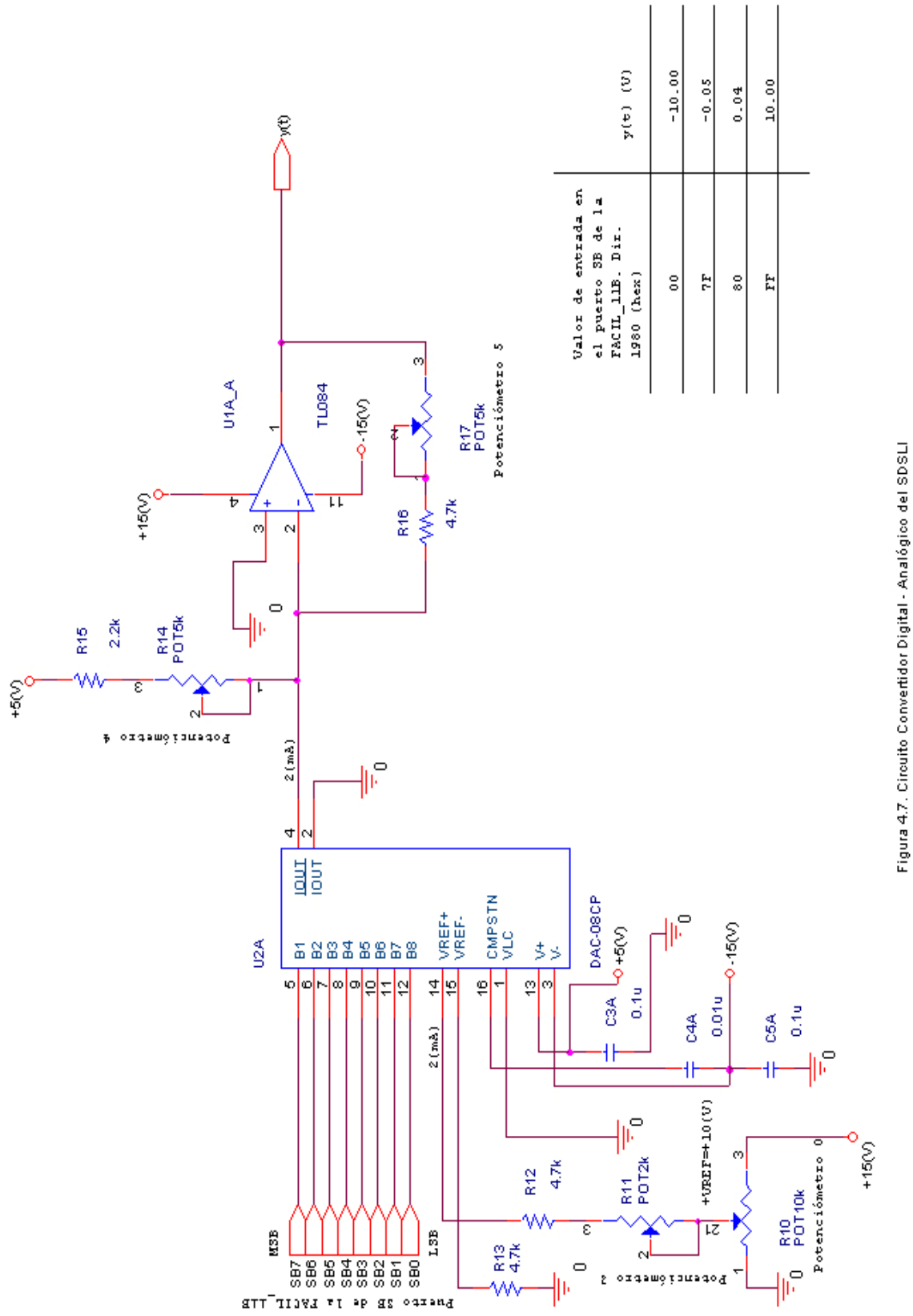


Figura 4.7. Circuito Convertidor Digital - Analógico del SDSLI

La figura 4.7, presenta el circuito de conversión digital a analógico del SDSLI, junto con una tabla que indica los valores de calibración de la señal de entrada y de la señal de salida de este circuito.

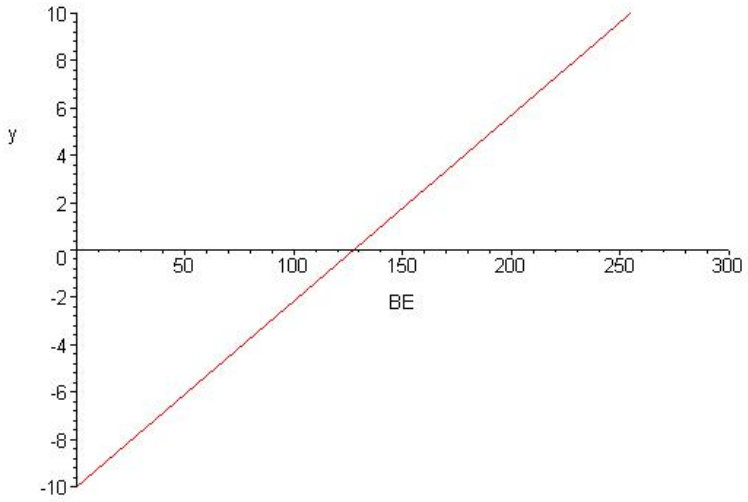


Figura 4.8. Curva característica de operación del circuito de conversión Digital a Analógica

En la figura 4.8, es posible observar la curva característica de operación del circuito de conversión Digital a Analógica de la señal de salida $y(t)$ del SDSLI. El eje de las abscisas representa los valores del byte de entrada (BE) al convertidor Digital a Analógico (en su respectivo valor decimal), y el eje de las ordenadas, señala los valores correspondientes al voltaje de salida (y) de este circuito de conversión.

La ecuación matemática que representa a la curva característica de operación del circuito de conversión es:

$$y = \frac{4BE}{51} - 10 \quad (\mathrm{V}) \tag{4.14}$$

Donde:

$$0d \leq BE \leq 255d \tag{4.15}$$

4.2.2.3. Fuente de alimentación

El circuito de la fuente de alimentación es el encargado de convertir la corriente alterna entregada por el transformador del SDSLI, en corriente directa para alimentar el resto de los circuitos que componen al simulador.

La figura 4.9, corresponde al diagrama esquemático de la fuente de alimentación del SDSLI.

En J1 de la tarjeta SDSLI_B, se reciben 7.7 (Vac) del transformador, los cuales, son convertidos en 5 (Vdc) @ 1 (A) para alimentar a la tarjeta del simulador SDSLI_B y a la tarjeta FACIL_11B.

En el conector J2 de la tarjeta SDSLI_B, se reciben 32.8 (Vac) directamente del transformador, que son convertidos en +15 (Vdc) y -15(Vdc) @ 0.5 (A), utilizados para alimentar el circuito de acondicionamiento de la señal de entrada y el circuito de conversión Digital a Analógico de la tarjeta SDSLI_B.

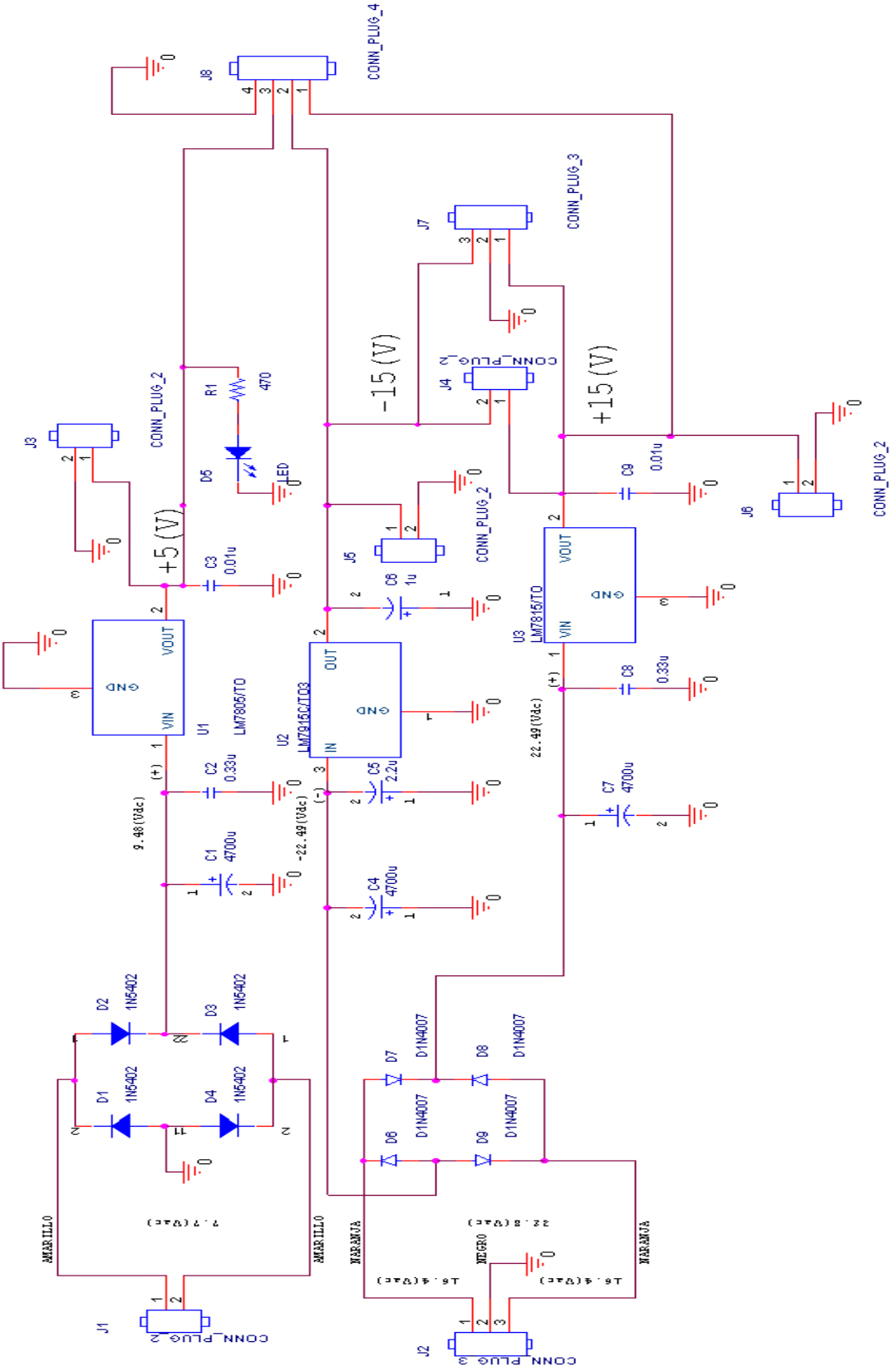


Figura 4.9. Fuente de alimentación del SDSLI

4.3. DISEÑO DEL SOFTWARE DEL SDSLI

Para la realización de la simulación, se efectúa en tiempo real, el cálculo de la salida presente en términos del valor de la señal de entrada presente, así como salidas y entradas anteriores. Todo el cálculo se realiza de acuerdo a la ecuación en diferencias, asociada con la función de transferencia $H[z]$, que corresponde a la discretización del sistema analógico a simular.

El software del simulador se divide en tres partes:

1. Firmware de simulación del SDSLI (Back - End).
2. Interfaz gráfica del usuario (GUI, por sus siglas en inglés). Este módulo de software es el Front – End del sistema.
3. Manual del usuario y manual técnico.

El firmware de simulación del SDSLI está integrado por rutinas básicas escritas en lenguaje C y en ensamblador, que permiten realizar las tareas necesarias para llevar a cabo una simulación. Estas tareas son:

1. Interrupción para realizar el muestreo de la señal de entrada.
2. Lectura del convertidor Analógico a Digital.
3. Adecuación aritmética del valor de entrada.
4. Cálculo de la salida presente mediante la ecuación en diferencias.
5. Generación del byte a colocar en el puerto ligado al hardware de conversión Digital a Analógica.
6. Actualización de variables de entrada y salida presentes y anteriores.

La interfaz gráfica del usuario, es un módulo de software pensado para la plataforma Windows, desarrollado en Visual Basic y que provee al usuario de un conjunto de herramientas amigables mediante las cuales, el usuario podrá desarrollar fácilmente

simulaciones de funciones de transferencia en el dominio de la variable s y en el dominio de la variable z. Además, provee herramientas como un osciloscopio virtual que permite visualizar y medir las señales de entrada y salida durante una simulación, así como asistentes para la programación autónoma del simulador y para realizar la calibración del mismo.

El manual del usuario y el manual técnico no se detallan en este apartado, pero es posible consultarlos en la sección de anexos de este trabajo.

4.3.1. Firmware de simulación del SDSLI

El firmware es el programa que realiza la simulación, consta globalmente de los siguientes dos componentes:

1. **Función *main()***, la cual inicialmente, reserva memoria para desarrollar la simulación. Posteriormente, inicializa los vectores de interrupción del canal OC2 del temporizador y del puerto serial, lee los valores enviados al hardware del simulador, mediante la interfaz de Windows del SDSLI, los cuales son: el tiempo de muestreo, tiempo de retardo, orden del sistema y los coeficientes de la función de transferencia $H[z]$ a simular. Estos valores se pueden leer de la memoria RAM o de la memoria EEPROM dependiendo del modo de operación en que se encuentre el hardware del simulador. Además de esto, calcula el valor inicial de la variable *escaladorTiempo*, inicializa variables auxiliares, inicializa los puertos SA y SB de la FACIL_11B, inicializa el convertidor Analógico a Digital y prepara el temporizador para comenzar la simulación.

2. **Rutina de servicio de interrupción del canal OC2 del temporizador.** Esta rutina realiza las tareas principales para desarrollar una simulación y se encarga de invocar a la función *diffEquation()* en intervalos iguales al periodo de muestreo.

Dentro del código escrito en lenguaje C para realizar una simulación, la rutina más importante y que permite la determinación de la salida presente en función del valor de la señal de entrada presente, así como de los valores de las salidas y entradas anteriores, es la que se denominó *diffEquation()*. Esta función se invoca a intervalos iguales al periodo de muestreo asignado. Para lograr invocar esta rutina periódicamente, se decidió utilizar la interrupción disparada por el temporizador del 68HC11F1 utilizando el canal OC2, configurando el temporizador, para generar una interrupción cada $10(ms)$, debido a esto, el periodo de muestreo, debe ser un múltiplo de $10(ms)$.

La rutina *servInt_OC2()*, es la rutina de servicio asociada a la interrupción generada por el temporizador en el canal OC2. En esta rutina, se actualiza un contador llamado *escaladorTiempo*, el cual al llegar a cero, primeramente invoca al convertidor Analógico a Digital para tomar una muestra del valor presente de la señal de entrada al hardware del simulador, a continuación, lee el resultado de la conversión para luego, ejecutar la rutina de adecuación del valor de entrada nombrada *xn()*, posteriormente, llama a la función *diffEquation()*, para calcular el valor presente de la señal de salida mediante la ecuación en diferencias y luego, llama a la subrutina *BDAC()* para generar el byte de salida que luego se envía por el puerto SB (dirección 1980h) de la tarjeta FACIL_11B y además, también se envía por el puerto serial para comunicar el resultado a la interfaz para Windows del SDSLI y finalmente, resetea el valor del contador *escaladorTiempo* a su valor inicial.

En las figuras 4.10, 4.11 y 4.12 se detalla el diagrama de flujo de nivel cero, correspondiente al firmware desarrollado para realizar simulaciones de sistemas de primer y hasta cuarto orden con el SDSLI.

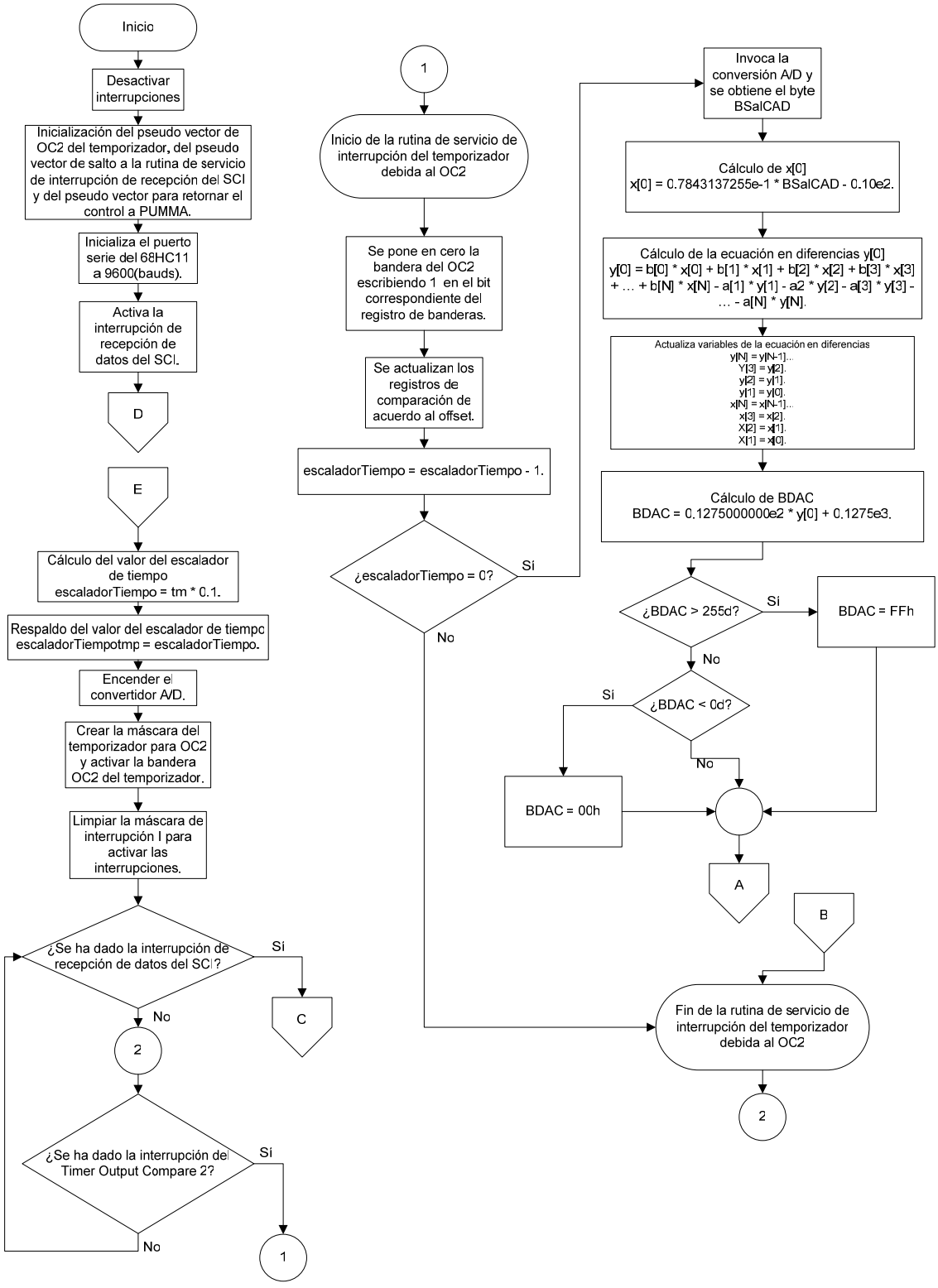


Figura 4.10. Diagrama de flujo del código que ejecuta el microcontrolador 68HC11F1, cuando el SDSLI realiza una simulación (parte uno de tres)

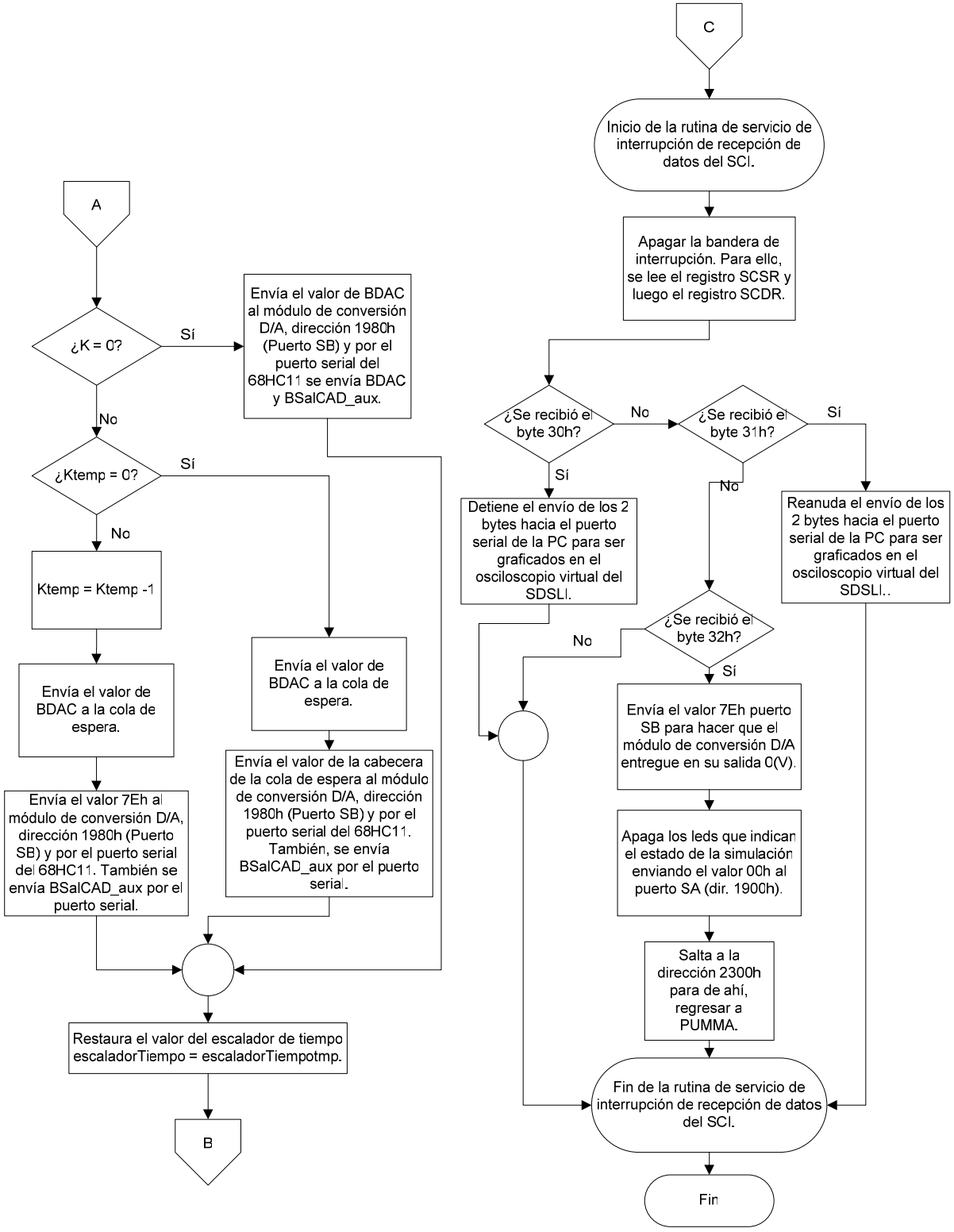


Figura 4.11. Diagrama de flujo del código que ejecuta el microcontrolador 68HC11F1, cuando el SDSLI realiza una simulación (parte dos de tres)

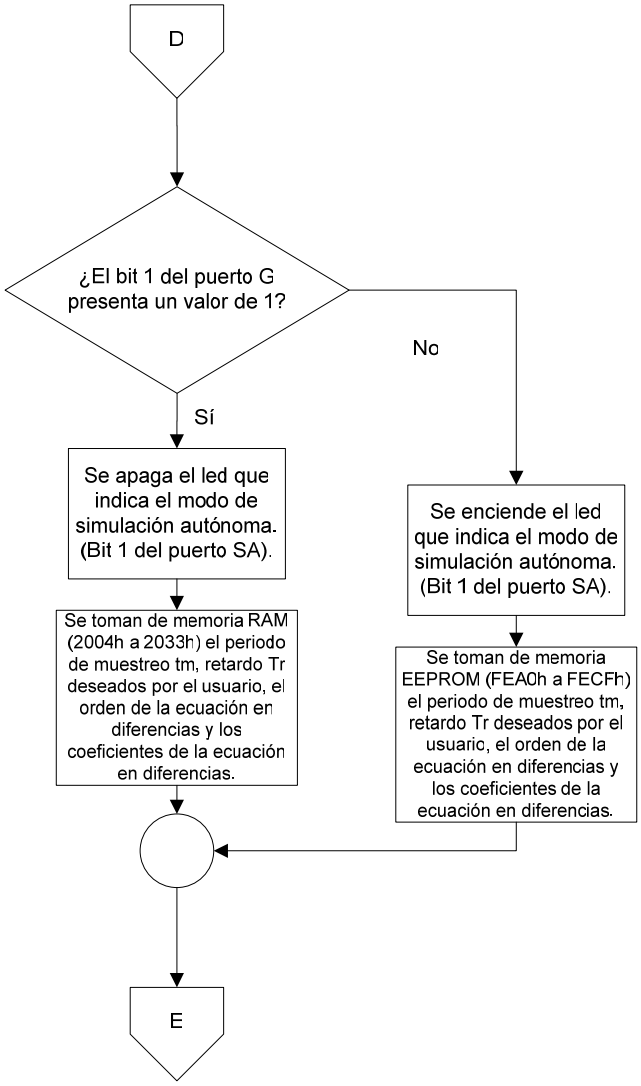


Figura 4.12. Diagrama de flujo del código que ejecuta el microcontrolador 68HC11F1, cuando el SDSLI realiza una simulación (parte tres de tres)

4.3.2. Interfaz gráfica del usuario

Como ya se mencionó con anterioridad, la interfaz gráfica del usuario, es un módulo de software, cuyo objetivo es permitir al usuario manejar de una manera amigable el SDSLI para desarrollar simulaciones rápida y fácilmente desde el sistema operativo Windows mediante una PC.

La interfaz esta programada en su mayor parte en Visual Basic y contiene algunos módulos empaquetados que se encuentran escritos en lenguaje ensamblador para el 68HC11F1, que realizan tareas especiales para la comunicación y control del hardware del SDSLI.

El hardware del SDSLI debe estar conectado a la PC tal y como lo muestra el esquema de la figura 4.13.

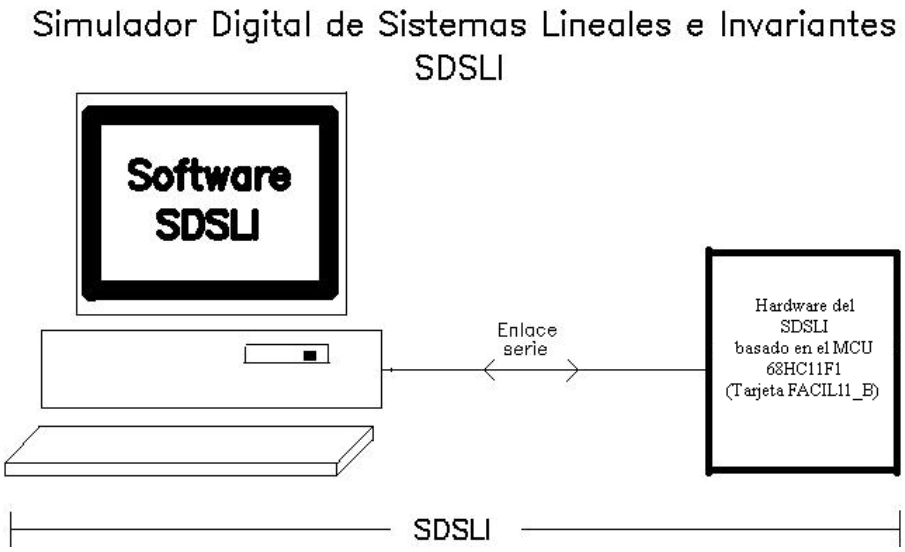


Figura 4.13. Conexión del hardware del SDSLI a la PC

El módulo fundamental de software para poder establecer una comunicación entre la PC y el hardware del SDSLI, es el denominado software PUMMA, escrito por el M. I. Antonio Salvá Calleja.

El núcleo central de software alrededor del cual se construyó este desarrollo, es un tramo de código que aquí se denominará como NÚCLEO BÁSICO DE COMUNICACIONES DE PUMMA_11 (NBCP11), el cual se ejecuta en el MCU de la arquitectura destino (FACIL_11) y se localiza en la página cero de la memoria del 68HC11 en el intervalo de direcciones 0000h a 008Fh; al iniciar la ejecución de la interfaz de usuario en la computadora anfitriona. El NBCP11 es cargado en la

arquitectura destino para su ejecución inmediata, empleando para esto el programa *bootloader*; que es parte del firmware residente en ROM (*bootrom)* presente en el mapa de memoria del HC11 cuando éste opera en modo bootstrap

"La función del NBCP11 fundamentalmente consiste en recibir vía serie de la PC, un programa que podrá estar localizado a partir de cualquier dirección fuera del intervalo mencionado anteriormente, debiendo existir obviamente memoria RAM en cantidad suficiente a partir de la dirección inicial especificada, una vez que ha terminado el envío, el programa recibido es ejecutado de inmediato" **[A. Salvá 2000]**.

En la figura 4.14, se muestra el diagrama de flujo del NBCP11.

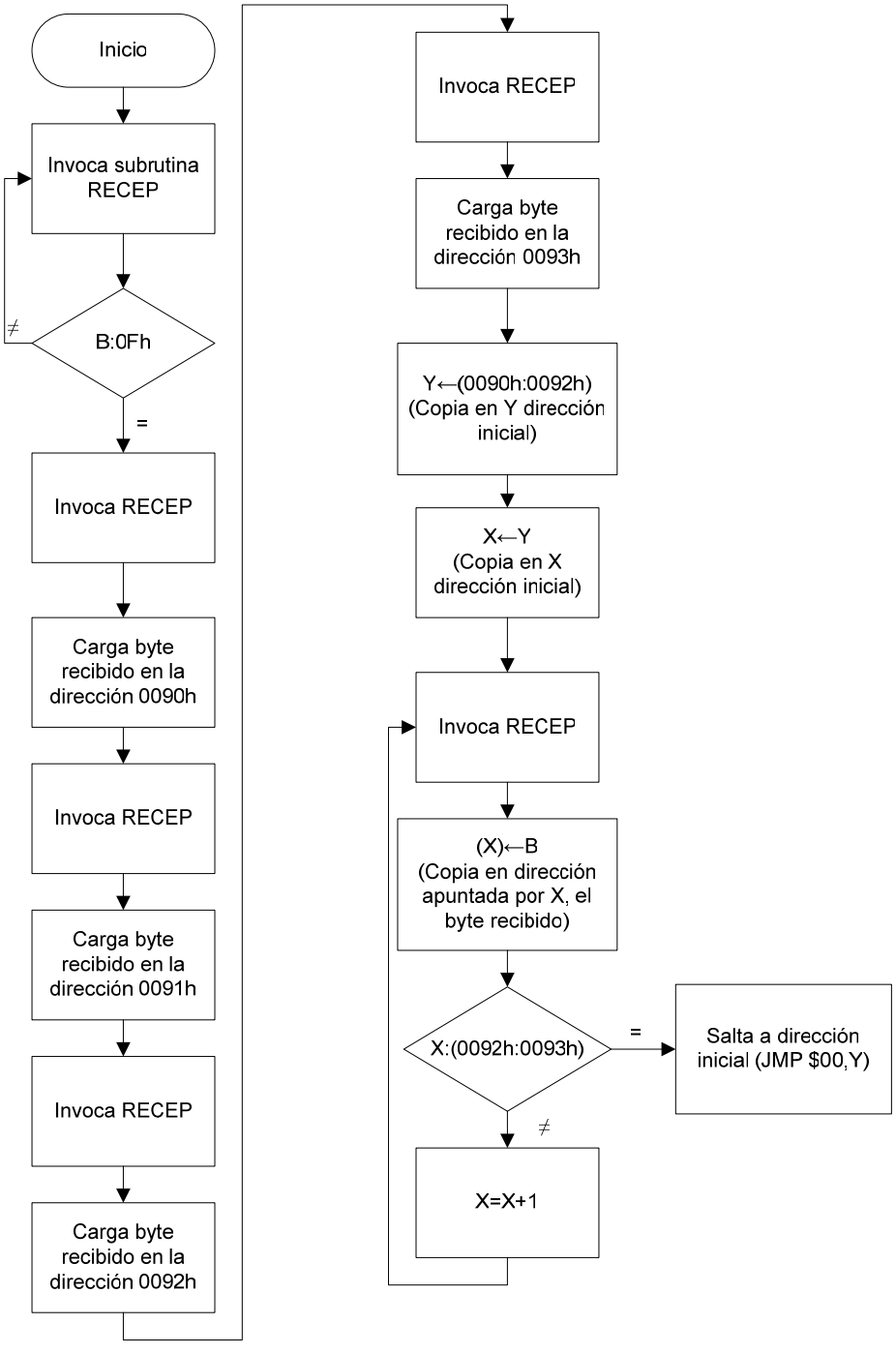


Figura 4.14. Diagrama de flujo del NBCP11

"Para una mejor comprensión del diagrama de flujo del NBCP11, es conveniente señalar que la subrutina RECEP que es invocada en diversos bloques del mismo, es simplemente una rutina de recepción serial, que opera por poleo y cuando es invocada, no se sale de la misma sino hasta que se haya recibido un byte, retornándose en el registro B del CPU, el dato recibido" **[A. Salvá 2000]**.

"Además de la rutina de recepción aquí mencionada, el NBCP11 cuenta con una subrutina de transmisión serial, que opera también por poleo, debiendo precargarse en el registro B el dato a transmitir; la dirección de colocación de la subrutina de transmisión es la 0050h y la que corresponde a la subrutina de recepción es la 0040h; la subrutina de transmisión es empleada cuando algún comando de la interfaz de usuario requiere información que está en la arquitectura destino como podría ser, por ejemplo, el contenido de una localidad de memoria en la misma" **[A. Salvá 2000]**.

Para bajar un programa para su ejecución en el simulador, se requiriere que el NBCP11 se esté ejecutando en éste, lo cual sucede como parte de la secuencia de inicialización del software de la interfaz para Windows del SDSLI; bajo esta circunstancia y de acuerdo al diagrama de flujo de la figura 4.14, la secuencia de pasos que sigue la interfaz del SDSLI, al igual que PUMMA_11, para hacer que un programa se ejecute en la memoria RAM de la arquitectura destino, es la siguiente:

a) Se envía por el puerto serie a la arquitectura destino un byte cuyo valor es 0Fh.
b) Se envía por el puerto serie a la arquitectura destino el byte alto de la dirección inicial de carga del programa.
c) Se envía por el puerto serie a la arquitectura destino el byte bajo de la dirección inicial de carga del programa.
d) Se envía por el puerto serie a la arquitectura destino el byte alto de la dirección final de carga del programa.
e) Se envía por el puerto serie a la arquitectura destino el byte bajo de la dirección final de carga del programa.

f) Se envían secuencialmente a la arquitectura destino, los bytes que representan el programa en código de máquina.

"A la secuencia anterior la denominaremos aquí como protocolo PUMMA. En la subrutina de recepción (RECEP), existe código que hace parpadear, con una velocidad apreciable a simple vista, un LED conectado al bit cinco del puerto D (en el caso del hardware del SDSLI es el led nombrado como LISTO), testificándose con este hecho, la disponibilidad de la arquitectura destino para recibir un programa desde la computadora anfitriona, empleando para ello, el protocolo descrito en el párrafo anterior. El parpadeo aquí mencionado indicaría al usuario que la arquitectura destino está en posibilidad de recibir comandos o programas desde la computadora anfitriona" **[A. Salvá 2000]**.

Una vez que la interfaz del SDSLI para Windows, ha iniciado la arquitectura destino, el hardware del SDSLI ya está listo para recibir los datos del usuario para comenzar a realizar una simulación.

La ventana principal del software del SDSLI, permite acceder a todas las opciones y herramientas que se requieren para desarrollar una simulación con el hardware del SDSLI. Su objetivo principal, es permitir que se controle el hardware del SDSLI, de una manera simple y directa mediante una PC.

La ventana principal, le posibilita el desarrollo de simulaciones con funciones de transferencia de tipo $G(s)$.

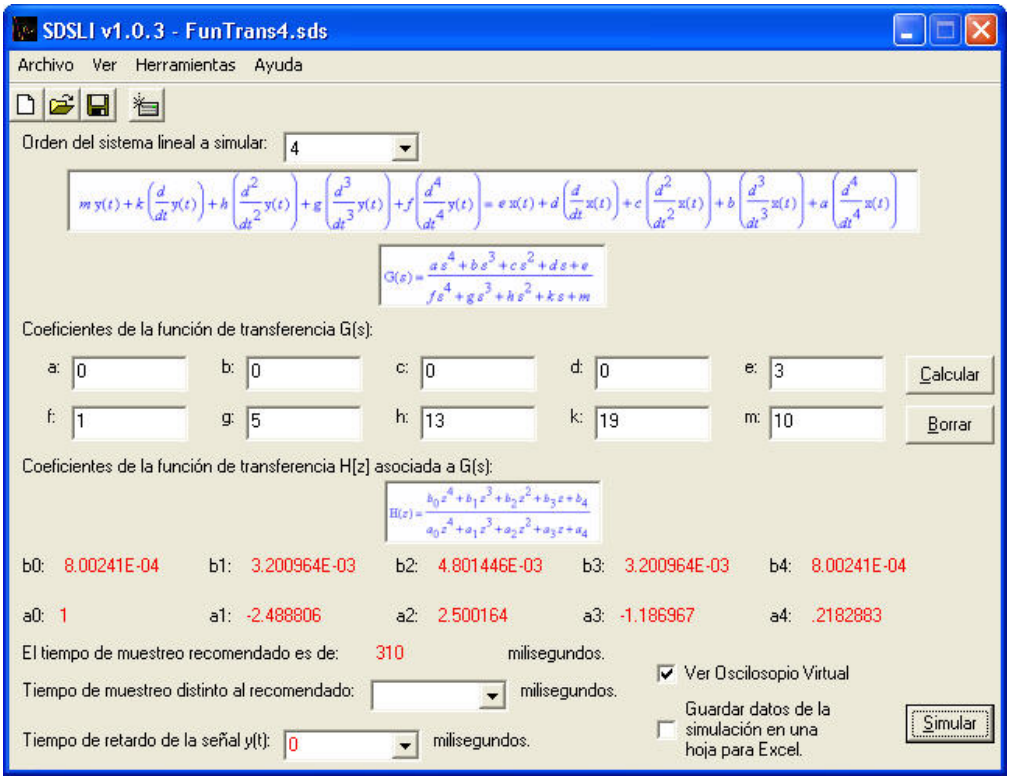


Figura 4.15. Ventana principal de la interfaz del SDSLI

Además de la ventana principal, el software cuenta con la ventana *Simulación de función de transferencia* $H[z]$ que, al igual que la ventana principal, permite al usuario acceder a algunas de las opciones y herramientas que se requieren para desarrollar una simulación con el hardware del SDSLI.

La diferencia más importante entre la ventana principal y esta ventana es que, posibilita al usuario desarrollar simulaciones directamente con funciones de transferencia de tipo $H[z]$ y no con funciones de transferencia de tipo $G(s)$ como sucede con la ventana principal.

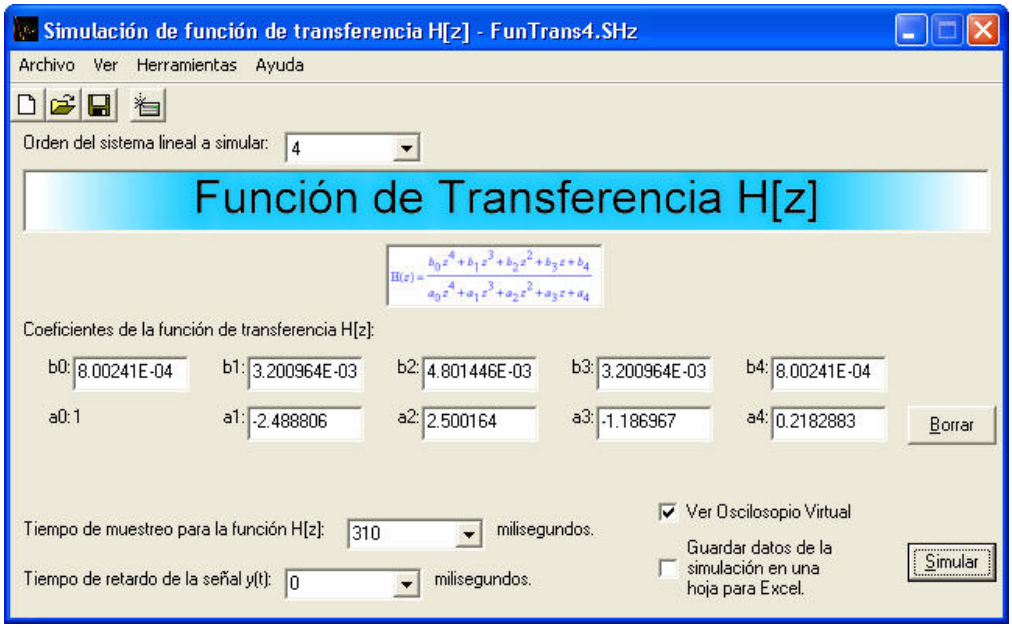


Figura 4.16. Ventana Simulación de función de transferencia H[z] de la interfaz del SDSLI

Para realizar una simulación tanto en la ventana principal como en la ventana *Simulación de función de transferencia* $H[z]$, se siguen estos pasos:

1. **Elegir el orden del sistema lineal**. El orden del sistema puede ser 0, 1, 2, 3 y 4 para el caso de la ventana principal y 1, 2, 3 y 4 para el caso de la ventana *Simulación de función de transferencia* $H[z]$. Si desea realizar simulaciones con sistemas lineales de orden cero, el software sólo requiere que se introduzca un valor de constante de proporcionalidad Kp. Para los sistemas lineales de orden uno, dos, tres y cuatro, el sistema requerirá que introduzca los coeficientes respectivos de la función de transferencia $G(s)$.

2. **Introducir los coeficientes de la función de transferencia que desea simular.** Para el caso de la ventana principal, deberá introducir los coeficientes de la función de transferencia $G(s)$ y para el caso de la ventana *Simulación de función de transferencia* $H[z]$ deberá introducir los coeficientes de la función de transferencia $H[z]$.

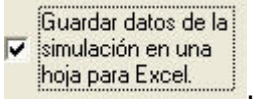
3. **Hacer click en el botón "Calcular".** Para calcular los coeficientes de la función de transferencia $H[z]$ así como un tiempo de muestreo recomendado, es necesario hacer click en el botón **"Calcular"**, en caso de que esté en la ventana principal. En caso de encontrarse en la ventana *Simulación de función de transferencia* $H[z]$*,* no es necesario este paso. Si se hace click en **"Borrar"**, se borrarán los coeficientes de la función de transferencia que se ven en la ventana.

4. **Elegir el valor del tiempo de muestreo.** En el caso de la ventana principal, si se desea, es posible que se elija un tiempo de muestreo distinto al recomendado. En el caso de la ventana *Simulación de función de transferencia* $H[z]$ es obligatorio que indique el tiempo de muestreo correspondiente a los coeficientes de la función de transferencia $H[z]$ que se introdujeron, pues en esta ventana no existe un tiempo de muestreo recomendado pero sí por omisión.

5. **Seleccionar el tiempo de retardo de la señal y(t).** Tanto en la ventana principal, como en la ventana *Simulación de función de transferencia* $H[z]$, si lo desea, puede elegir un valor de tiempo de retardo de la señal de salida $y(t)$.

6. **Activar la casilla de verificación para visualizar el Osciloscopio Virtual.** Si se desea que al momento de iniciar la simulación, aparezca el Osciloscopio Virtual, deberá marcar la casilla de verificación 

.

7. **Activar la casilla para guardar los datos de una simulación en una hoja para Excel.** Si se requiere que el software almacene los datos de la simulación en una hoja para Excel, entonces marque la casilla de verificación 

.

8. **Almacenar en disco la simulación.** Antes de iniciar la simulación, se recomienda que se almacene en disco los datos de su simulación. Para esto, es

posible hacer click en el botón  de la barra de herramientas de la ventana o bien acceder a las opciones **"Guardar"** o **"Guardar como…"** del menú **"Archivo"**.

9. **Para simular el sistema, hacer click en el botón "Simular".** Finalmente, si desea ejecutar la simulación y observar las señales resultantes de la simulación, se debe presionar el botón **"Simular"** de la ventana.

En el momento que se hace click en el botón **"Simular"**, el software descarga en el hardware del SDSLI, mediante el protocolo PUMMA, un programa que contiene el tiempo de muestreo, tiempo de retardo, orden del sistema y los coeficientes de la función de transferencia $H[z]$ a simular. Estos datos son ubicados de la localidad 2004h a la localidad 2032h, bajo la norma 754 de la IEEE. Posteriormente, se descarga otro programa que ordena la ejecución del firmware del SDSLI mediante un salto a la localidad 8000h y de esta manera, la simulación inicia.

Es posible que durante la simulación, se haya elegido observar las señales de entrada y salida del simulador, por lo cual en la pantalla de la PC se desplegará la siguiente ventana:

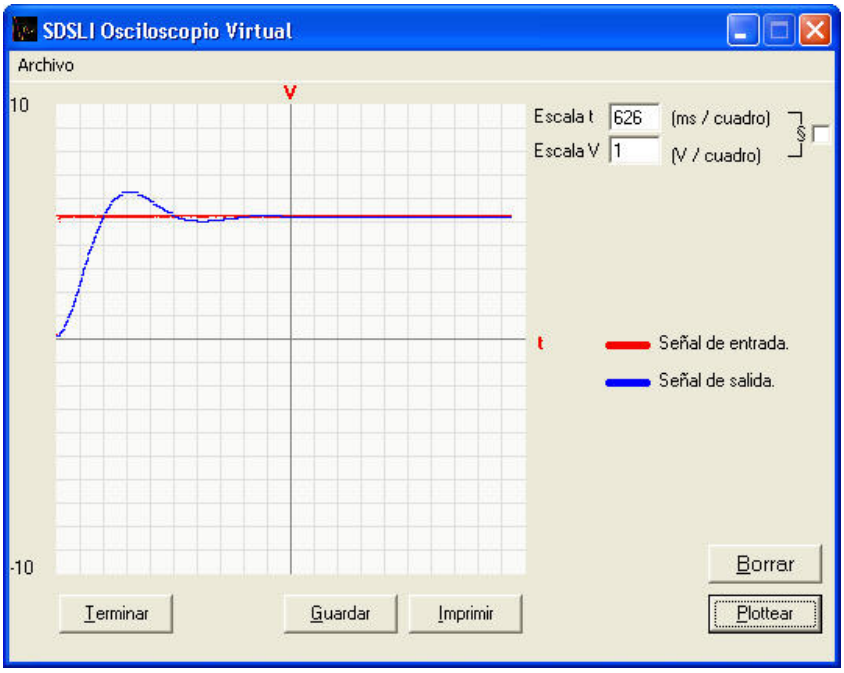


Figura 4.17. El Osciloscopio Virtual de la interfaz del SDSLI

El Osciloscopio Virtual, es la herramienta le permite observar gráficamente las señales de entrada y salida al hardware del SDSLI, mientras se encuentra una simulación activa.

Si se desea mayor información respecto a las distintas opciones y herramientas que ofrece la interfaz del SDSLI para Windows, consulte el Manual del Usuario del SDSLI en la sección de anexos de este trabajo.

4.4. PRUEBAS REALIZADAS AL SDSLI

La calidad de un sistema de información, depende de su diseño, desarrollo, prueba e implantación. Un aspecto de la calidad del sistema es su confiabilidad. Un sistema es confiable si, al usarse de manera razonable, no produce fallas peligrosas o costosas. Esta definición distingue entre los errores del software y el hardware, en lo que el sistema no produce los resultados esperados, y las fallas que se presentan. Aunque es muy difícil desarrollar hardware y software que se pueda demostrar que esté libre de

errores, se debe hacer lo posible por prevenir su aparición, usando métodos y técnicas que incluyan la detección de errores, su corrección y tolerancia.

La prueba, es el proceso de hacer funcionar un sistema con la intención de hallar errores, es decir, hacer que el sistema falle. Así, una prueba exitosa, es la que encuentra un error.

Las pruebas se realizaron a todo lo largo del desarrollo del sistema y cumplieron con el propósito de identificar aquellos problemas desconocidos, más no demostrar la perfección del sistema.

Las pruebas aplicadas al SDSLI fueron exitosas, es decir, encontraron errores y fueron las siguientes:

- **Prueba de la caja blanca.** Se realizaron pruebas basadas en el conocimiento sobre la lógica y estructura interna del sistema.

- **Prueba de código.** Se examinó la lógica del software. Se examinó cada ruta de cada uno de los módulos del software.

- **Prueba de señales.** Se revisó la lógica del hardware. Se evaluó cada ruta de cada módulo electrónico que conforma la tarjeta SDSLI_B.

- **Pruebas parciales.** Se probaron todos los módulos que conforman el software. Las pruebas se llevaron de forma ascendente, comenzando con los módulos más pequeños y de nivel inferior y continuando de uno en uno. Se probaron los módulos de forma individual y después conjuntamente.

- **Pruebas de sistema.** Se probó la integración de cada módulo en el sistema (tanto hardware como software). Se buscaron discrepancias entre el sistema y su

objetivo original, especificaciones y documentación del sistema. La preocupación principal fue la compatibilidad de los módulos individuales.

- **Pruebas de especificación.** Se examinaron las especificaciones que señalan lo que el SDSLI debe hacer y cómo lo debe llevar a cabo, bajo diferentes condiciones. Se desarrollaron casos de prueba para cada condición o combinación de condiciones y se enviaron al sistema para su procesamiento. Se probó al sistema como una caja negra, pues se supuso que si el sistema cumple las especificaciones, no fallará.

- **Pruebas alfa.** Se ejecutó el firmware del SDSLI en un ambiente simulado y además se utilizaron emuladores de Terminal para realizar pruebas con el SDSLI. Se preparó una versión alfa del sistema, para lo cual se simuló el ambiente de trabajo en el que se supone que operaría el sistema, y se solicitó la cooperación de un futuro usuario del sistema para operar el sistema con datos ficticios.

- **Pruebas beta.** Se utilizó el SDSLI en un ambiente no simulado, es decir real, y se realizaron varias instalaciones del sistema en diversas versiones de Windows para encontrar errores. Se usó el sistema en las actividades cotidianas, se procesaron transacciones en directo y se produjeron salidas normales del sistema.

- **Prueba de carga máxima o volumen.** Se determinó si el sistema manejará el volumen de actividades que ocurran cuando esté en su punto más alto de demanda de procesamiento.

- **Prueba de almacenamiento.** Se determinó la capacidad del sistema para almacenar datos de transacciones en disco o en archivos.

- **Prueba de tiempo de ejecución.** Se determinó el tiempo máquina que el sistema necesita para procesar los datos de una transacción.

- **Pruebas de recuperación.** Se observó la capacidad del usuario para recuperar los datos o reestablecer el sistema después de una falla.

- **Prueba de procedimientos.** Se revisó la claridad de la documentación en los aspectos de operación y uso del sistema, haciendo que los usuarios lleven a cabo exactamente lo que el manual pide.

- **Prueba de factores humanos.** Se determinó cómo utilizarán los usuarios el sistema al procesar los datos o preparar informes. Aún falta realizar pruebas con los alumnos de la facultad.

- **Prueba de aceptación del usuario.** Se realizaron pruebas con usuarios finales, para asegurar que el sistema satisfaga las necesidades requeridas.

- **Pruebas estáticas.** Se revisaron y verificaron los documentos generados en las distintas fases de la vida del proyecto.

- **Pruebas funcionales.** Se validaron los requerimientos de la Facultad de Ingeniería (lo que se supone que el sistema debe hacer).

- **Pruebas estructurales.** Se validó la arquitectura del sistema, confirmando que todos sus componentes funcionen de manera armónica y hasta la fecha, se está validando que la tecnología esté siendo utilizada apropiadamente.

- **Prueba de seguridad.** Se ha verificado que los mecanismos de protección incorporados en el sistema lo protegen de la penetración impropia.

- **Prueba de regresión.** Se han detectado y corregido diversas fallas que se han introducido al realizar modificaciones y ajustes al sistema o a alguno de sus componentes. También se ha revisado que estas modificaciones no tengan un impacto negativo sobre el sistema y que siga cumpliendo con los requerimientos planteados.

4.5. MANTENIMIENTO DEL SDSLI

Una parte fundamental para el buen funcionamiento y posible crecimiento del SDSLI, se basa en el mantenimiento que se brinde a éste, es de hecho, la última fase dentro de la construcción del sistema, ya que de este paso dependen los cambios necesarios al sistema para que éste siga siendo funcional y operativo.

Los tipos de mantenimiento que hasta la fecha se han realizado al SDSLI son:

- **Perfectivo.** Se han realizado cambios a nivel programación de software. Conforme se ha instalado y utilizado el SDSLI, los usuarios han realizado observaciones y recomendaciones relativas a problemas, errores y nuevas posibilidades acerca de modificaciones funcionales ya existentes en el sistema.

- **Preventivo.** Se han evitado algunos errores y problemas en el sistema. Este mantenimiento se ha realizado durante cambios y actualizaciones en el software con el fin de mejorar los procesos de éste. Además, se ha dado mantenimiento a la información que el sistema maneja con la finalidad de que los resultados devueltos por éste, sean correctos.

- **Adaptativo aumentativo.** Se tienen planeados y observados algunos cambios que incluyen nuevas funciones que no han sido contempladas al inicio del desarrollo del sistema y que han surgido como necesidades del usuario. Hasta la

fecha, sólo se han realizado algunas adaptaciones aumentativas en el software, pero el hardware también las contempla y no se han realizado por el momento.

- **Adaptativo tecnológico.** Se han realizado algunos cambios y optimizaciones en el software del sistema, pero se tienen planeados hacer más cambios. Respecto al hardware del SDSLI, se tienen planeados mejoras en cuanto a la velocidad de procesamiento del sistema. Se han realizado algunos cambios sobre el hardware del sistema, para garantizar que se encuentren refacciones para éste, por lo menos, durante los próximos 10 años.

- **Correctivo.** Se han diagnosticado y corregido varios errores en el sistema. Este mantenimiento se ha aplicado para corregir varios errores no descubiertos y que han sido arrojados por la etapa de pruebas antes de poner en uso el sistema.

Las versiones del software que se tienen a la fecha son:

1. *Interfaz para Windows del SDSLI:* versión 1.0.4
2. *Firmware del SDSLI:* versión 1.0.0
3. *Tarjeta SDSLI:* versión B
4. *Tarjeta FACIL_11:* versión B

Este capítulo, tuvo como objetivo mostrar al lector, la propuesta que hacemos, con la implementación del SDSLI, para dar solución a las cuestiones aún no resueltas en el tema de la simulación de sistemas lineales e invariantes.

CAPÍTULO 5

EJEMPLO PRÁCTICO: MODELADO Y SIMULACIÓN DE UN SISTEMA DE SUSPENSIÓN DE UN AUTOMÓVIL, UTILIZANDO LA FUNCIÓN DE TRANSFERENCIA

En este capítulo encontrará:

5. EJEMPLO PRÁCTICO: MODELADO Y SIMULACIÓN DE UN SISTEMA DE SUSPENSIÓN DE UN AUTOMÓVIL, UTILIZANDO LA FUNCIÓN DE TRANSFERENCIA

El objetivo de este capítulo, es ilustrar el uso de SDSLI con una aplicación práctica.

Este ejemplo se enfoca en el modelado y simulación del movimiento vertical de suspensión de un automóvil. Tiene en cuenta la inercia del transporte y la inercia del conjunto suspensión-ruedas, así como muelles y amortiguadores. Se posiciona un actuador entre la suspensión y el cuerpo del automóvil. Este sistema de cuarto orden es particularmente difícil de simular y controlar debido a la existencia de dos ceros cercanos al eje imaginario, lo cual requiere una compensación cuidadosa.

5.1. PLANTEAMIENTO FÍSICO

Modelar un sistema de suspensión automático para un automóvil, se convierte en un interesante problema. Para el diseño del sistema de suspensión, se observa sólo un cuarto del modelo total del automóvil (una de las cuatro ruedas), con la finalidad de simplificar el problema a un sistema unidimensional de resorte y amortiguador. A continuación se muestra un diagrama de este sistema:

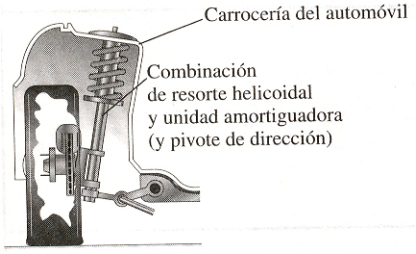


Figura 5.1. Amortiguador del automóvil

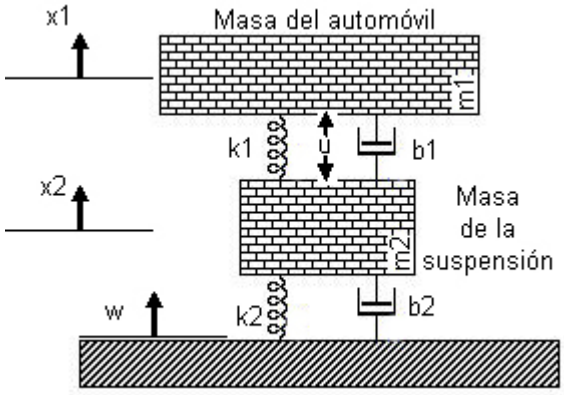


Figura 5.2. Diagrama del sistema de suspensión de una de las cuatro ruedas del automóvil

Se tienen los siguientes datos:

Masa del automóvil: $m_1 = 364.85(kg)$

Masa de la suspensión: $m_2 = 29.19(kg)$

Constante del resorte del sistema de suspensión: $k_1 = 14593.90\left(\frac{N}{m}\right)$

Constante del resorte de la rueda: $k_2 = 65672.56\left(\frac{N}{m}\right)$

Constante de amortiguamiento del sistema de suspensión: $b_1 = 291.88\left(\frac{Ns}{m}\right)$

Constante de amortiguamiento de la rueda: $b_2 = 12525.74\left(\frac{Ns}{m}\right)$

Fuerza de control: $u =$ Fuerza proporcionada por el controlador que se podría diseñar.

5.2. REQUERIMIENTOS DEL DISEÑO

Un buen sistema de suspensión automotriz, debe sostenerse satisfactoriamente en el pavimento, además de proveer comodidad cuando el automóvil pasa por topes y baches en la carretera. Cuando el automóvil experimenta algún tipo de perturbación en carretera (por ejemplo baches, fracturas y superficies disparejas), no debe presentar

oscilaciones considerables, y éstas deben disiparse rápidamente. En tanto que la distancia $x_1 - w$ es muy difícil de medir, y la deformación de la llanta $x_2 - w$ es despreciable, utilizaremos la distancia $x_1 - x_2$ en lugar de la distancia $x_1 - w$ como la salida en nuestro problema. Es importante recordar que esta medición es sólo una estimación de lo que sucede en la realidad.

La perturbación en carretera w, en este problema, será simulada por una entrada escalón. Esta entrada escalón, representará al automóvil saliendo de un bache.

5.3. ECUACIONES DE MOVIMIENTO

De la figura 5.2 y aplicando la ley de Newton, es posible obtener las ecuaciones dinámicas de la manera siguiente.

Para el análisis de la masa m_1:

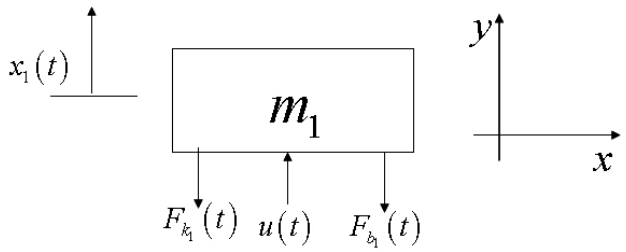


Figura 5.3. Diagrama de cuerpo libre correspondiente a la masa m_1

Del diagrama de cuerpo libre de la figura 5.3, se obtiene la siguiente ecuación:

$$F_1(t) = -F_{k_1}(t) - F_{b_1}(t) + u(t) \tag{5.1}$$

Donde:

$$F_1(t) = m_1 x''_{m_2}(t)$$
$$F_{k_1}(t) = k_1\left(x_1(t) - x_2(t)\right)$$
$$F_{b_1}(t) = b_1\left(x'_1(t) - x'_2(t)\right)$$

De tal manera que:

$$m_1 x''_1(t) = -k_1\left[x_1(t) - x_2(t)\right] - b_1\left[x'_1(t) - x'_2(t)\right] + u(t) \tag{5.2}$$

Para el análisis de la masa m_2:

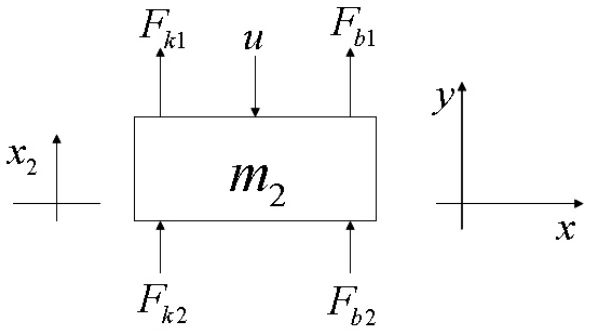


Figura 5.4. Diagrama de cuerpo libre correspondiente a la masa m_2

Ahora, por otro lado, del diagrama de cuerpo libre de la figura 5.4, se obtiene la siguiente ecuación:

$$F_2(t) = F_{k_1}(t) + F_{b_1}(t) + F_{k_2}(t) + F_{b_2}(t) - u(t) \tag{5.3}$$

Donde:

$$F_2(t) = m_2 x''_{m_2}(t)$$
$$F_{k_1}(t) = k_1\left(x_1(t) - x_2(t)\right)$$
$$F_{b_1}(t) = b_1\left(x_1'(t) - x_2'(t)\right)$$
$$F_{k_2}(t) = k_2\left(w(t) - x_2(t)\right)$$
$$F_{b_2}(t) = b_2\left(w_1'(t) - x_2'(t)\right)$$

De tal manera que:

$$m_2 x_2''(t) = k_1\left[x_1(t) - x_2(t)\right] + b_1\left[x_1'(t) - x_2'(t)\right] + k_2\left[w(t) - x_2(t)\right] + b_2\left[w'(t) - x_2'(t)\right] - u(t) \quad (5.4)$$

5.4. ECUACIÓN DE LA FUNCIÓN DE TRANSFERENCIA

Suponiendo que todas las condiciones iniciales son cero, estas ecuaciones representan el momento en el que la rueda del automóvil pasa por un bache. Las ecuaciones dinámicas descritas arriba, pueden ser expresadas en su forma de función de transferencia mediante la aplicación de la Transformada de Laplace a cada una de ellas. Las funciones de transferencia $G_1(s)$ y $G_2(s)$ con salida $x_1(t) - x_2(t)$, y dos entradas $u(t)$ y $w(t)$ se obtienen de la siguiente forma:

Aplicando la Transformada de Laplace a la ecuación (5.2) se tiene:

$$m_1 s^2 X_1(s) = -k_1\left[X_1(s) - X_2(s)\right] - b_1\left[sX_1(s) - sX_2(s)\right] + U(s)$$

Despejando la entrada $U(s)$:

$$m_1 s^2 X_1(s) + k_1\left[X_1(s) - X_2(s)\right] + b_1\left[sX_1(s) - sX_2(s)\right] = U(s)$$

Desarrollando la ecuación anterior:

$$m_1 s^2 X_1(s) + k_1 X_1(s) - k_1 X_2(s) + b_1 s X_1(s) - b_1 s X_2(s) = U(s)$$

Factorizando $X_1(s)$ y $X_2(s)$ de los términos de la ecuación anterior:

$$\left(m_1 s^2 + b_1 s + k_1\right) X_1(s) - \left(b_1 s + k_1\right) X_2(s) = U(s) \tag{5.5}$$

Aplicando la Transformada de Laplace a la ecuación (5.4) se tiene:

$$m_2 s^2 X_2(s) = k_1\left[X_1(s) - X_2(s)\right] + b_1\left[sX_1(s) - sX_2(s)\right] + k_2\left[W(s) - X_2(s)\right]$$
$$+ b_2\left[sW(s) - sX_2(s)\right] - U(s)$$

Despejando la entrada $U(s)$:

$$U(s) = -m_2 s^2 X_2(s) + k_1\left[X_1(s) - X_2(s)\right] + b_1\left[sX_1(s) - sX_2(s)\right]$$
$$+ k_2\left[W(s) - X_2(s)\right] + b_2\left[sW(s) - sX_2(s)\right]$$

Expandiendo la ecuación anterior:

$$U(s) = -m_2 s^2 X_2(s) + k_1 X_1(s) - k_1 X_2(s) + b_1 s X_1(s) - b_1 s X_2(s)$$
$$+ k_2 W(s) - k_2 X_2(s) + b_2 s W(s) - b_2 s X_2(s)$$

Factorizando $X_1(s)$, $X_2(s)$ y $W(s)$ de los términos de la ecuación anterior se tiene:

$$U(s) = \left(b_1 s + k_1\right) X_1(s) - \left(m_2 s^2 + k_1 + b_1 s + k_2 + b_2 s\right) X_2(s) + \left(k_2 + b_2 s\right) W(s)$$

Despejando los términos $U(s)$ y $W(s)$ se tiene:

$$-(b_1 s + k_1) X_1(s) + \left[m_2 s^2 + (b_1 + b_2) s + k_1 + k_2\right] X_2(s) = (b_2 s + k_2) W(s) - U(s) \qquad (5.6)$$

Organizando las ecuaciones (5.5) y (5.6) en forma de matrices:

$$\begin{bmatrix} m_1 s^2 + b_1 s + k_1 & -(b_1 s + k_1) \\ -(b_1 s + k_1) & m_2 s^2 + (b_1 + b_2) s + k_1 + k_2 \end{bmatrix} \begin{bmatrix} X_1(s) \\ X_2(s) \end{bmatrix} = \begin{bmatrix} U(s) \\ (b_2 s + k_2) W(s) - U(s) \end{bmatrix} \qquad (5.7)$$

Sea la matriz A:

$$A = \begin{bmatrix} m_1 s^2 + b_1 s + k_1 & -(b_1 s + k_1) \\ -(b_1 s + k_1) & m_2 s^2 + (b_1 + b_2) s + k_1 + k_2 \end{bmatrix} \qquad (5.8)$$

Y Δ el determinante de la matriz A:

$$\Delta = \begin{vmatrix} m_1 s^2 + b_1 s + k_1 & -(b_1 s + k_1) \\ -(b_1 s + k_1) & m_2 s^2 + (b_1 + b_2) s + k_1 + k_2 \end{vmatrix} \qquad (5.9)$$

De la ecuación (5.9), se tiene que el determinante Δ se calcula de la siguiente forma:

$$\Delta = (m_1 s^2 + b_1 s + k_1)\left[m_2 s^2 + (b_1 + b_2) s + k_1 + k_2\right] - (b_1 s + k_1)(b_1 s + k_1) \qquad (5.10)$$

Calculando la matriz inversa de A y premultiplicando la ecuación matricial (5.7) por esta última, se tiene:

$$\begin{bmatrix} X_1(s) \\ X_2(s) \end{bmatrix} = \frac{1}{\Delta} \begin{bmatrix} m_2 s^2 + (b_1 + b_2) s + k_1 + k_2 & b_1 s + k_1 \\ b_1 s + k_1 & m_1 s^2 + b_1 s + k_1 \end{bmatrix} \begin{bmatrix} U(s) \\ (b_2 s + k_2) W(s) - U(s) \end{bmatrix} \qquad (5.11)$$

Desarrollando la ecuación matricial (5.11) se tiene:

$$\begin{bmatrix} X_1(s) \\ X_2(s) \end{bmatrix} = \frac{1}{\Delta}\begin{bmatrix} m_2 s^2 + b_2 s + k_2 & b_1 b_2 s^2 + (b_1 k_2 + b_2 k_1) s + k_1 k_2 \\ -m_1 s^2 & m_1 b_2 s^3 + (m_1 k_2 + b_1 b_2) s^2 + (b_1 k_2 + b_2 k_1) s + k_1 k_2 \end{bmatrix} \begin{bmatrix} U(s) \\ W(s) \end{bmatrix} \tag{5.12}$$

De la ecuación matricial (5.12), es claro que:

$$X_1(s) = \frac{(m_2 s^2 + b_2 s + k_2) U(s) + [b_1 b_2 s^2 + (b_1 k_2 + b_2 k_1) s + k_1 k_2] W(s)}{\Delta} \tag{5.13}$$

$$X_2(s) = \frac{(-m_1 s^2) U(s) + [m_1 b_2 s^3 + (m_1 k_2 + b_1 b_2) s^2 + (b_1 k_2 + b_2 k_1) s + k_1 k_2] W(s)}{\Delta} \tag{5.14}$$

Si solo se desea considerar la entrada $U(s)$, debe hacerse $W(s) = 0$ y se puede obtener la función de transferencia $G_1(s)$ de la manera siguiente:

$$G_1(s) = \frac{X_1(s) - X_2(s)}{U(s)} = \frac{(m_1 + m_2) s^2 + b_2 s + k_2}{\Delta} \tag{5.15}$$

Por otro lado, si se desea considerar solamente la entrada $W(s)$, se hace $U(s) = 0$ y a continuación se calcula la función de transferencia $G_2(s)$:

$$G_2(s) = \frac{X_1(s) - X_2(s)}{W(s)} = \frac{-m_1 b_2 s^3 - m_1 k_2 s^2}{\Delta} \tag{5.16}$$

Donde, tanto para la función de transferencia (5.15), como para la función de transferencia (5.16), el valor de Δ es:

$$\Delta = m_1 m_2 s^4 + \left[m_1 \left(b_1 + b_2 \right) + m_2 b_1 \right] s^3 + \left[m_1 \left(k_1 + k_2 \right) + m_2 k_1 + b_1 b_2 \right] s^2 + \left(b_1 k_2 + b_2 k_1 \right) s + k_1 k_2 \tag{5.17}$$

Con la finalidad de poder simular cómodamente este sistema en el SDSLI, se realizará un escalamiento en tiempo y en magnitud de las funciones de transferencia $G_1(s)$ y $G_2(s)$.

Se sabe que el escalamiento en tiempo de una ecuación diferencial, se rige por la ecuación:

$$\frac{d^n}{dt^n} x(t) = h^n \frac{d^n}{dT_{maq}{}^n} x\left(T_{maq}\right) \tag{5.18}$$

Donde:

$t =$ Tiempo real.

$T_{maq} =$ Tiempo de máquina.

$h =$ Factor de escala en tiempo.

Por otra parte, la Transformada de Laplace de la relación

$$\frac{d^n}{dt^n} x(t) \rightarrow s^n X(s) \tag{5.19}$$

Transforma una ecuación diferencial del dominio del tiempo t a la forma operacional donde es expresada en función de la variable compleja s.

Lo anterior indica que el procedimiento para escalar en tiempo las funciones de transferencia se hace la sustitución de la variable:

$$s \to h\rho \tag{5.20}$$

en cada término de la función de transferencia.

Si:

$h > 1$ El tiempo de computación resultará mayor que el tiempo real del proceso.
$h < 1$ Hace que el tiempo de computación sea menor que el tiempo real del proceso.

Aplicando la sustitución de variable (5.20) en las funciones de transferencia (5.15) y (5.16) se tiene:

Para $G_1(s)$:

$$G_1(h\rho) = \frac{(m_1+m_2)h^2\rho^2 + b_2 h\rho + k_2}{m_1 m_2 h^4\rho^4 + (m_1 b_1 + b_1 m_2 + m_1 b_2)h^3\rho^3 + (m_1 k_2 + k_1 m_2 + b_1 b_2 + m_1 k_1)h^2\rho^2 + (k_1 b_2 + b_1 k_2)h\rho + k_1 k_2}$$

Al ser la variable ρ una variable muda, es válido hacer $\rho = s$:

$$G_1(hs) = \frac{(m_1+m_2)h^2 s^2 + b_2 hs + k_2}{m_1 m_2 h^4 s^4 + (m_1 b_1 + b_1 m_2 + m_1 b_2)h^3 s^3 + (m_1 k_2 + k_1 m_2 + b_1 b_2 + m_1 k_1)h^2 s^2 + (k_1 b_2 + b_1 k_2)hs + k_1 k_2} \tag{5.21}$$

Para $G_2(s)$:

$$G_2(h\rho) = \frac{-m_1 h^2\rho^2 (b_2 h\rho + k_2)}{m_1 m_2 h^4\rho^4 + (m_1 b_1 + b_1 m_2 + m_1 b_2)h^3\rho^3 + (m_1 k_2 + k_1 m_2 + b_1 b_2 + m_1 k_1)h^2\rho^2 + (k_1 b_2 + b_1 k_2)h\rho + k_1 k_2}$$

Regresando a la variable s, se hace $\rho = s$:

$$G_2(hs) = \frac{-m_1 b_2 h^3 s^3 - m_1 k_2 h^2 s^2}{m_1 m_2 h^4 s^4 + (m_1 b_1 + b_1 m_2 + m_1 b_2) h^3 s^3 + (m_1 k_2 + k_1 m_2 + b_1 b_2 + m_1 k_1) h^2 s^2 + (k_1 b_2 + b_1 k_2) hs + k_1 k_2} \quad (5.22)$$

Para realizar el escalamiento en magnitud de las funciones de transferencia (5.21) y (5.22), se indican con X_{Max} y Y_{Max} los valores máximos estimados de las variables de entrada y salida, respectivamente, de la función de transferencia. Entonces, se tienen los siguientes factores de escalamiento de magnitud, F_{EE} en la entrada y F_{ES} en la salida de una función de transferencia $G(s)$.

$$F_{EE} = \frac{1}{X_{Max}}$$

$$F_{ES} = \frac{1}{Y_{Max}}$$

El escalamiento en magnitud se obtiene multiplicando la entrada $X(s)$ por F_{EE} y la salida $Y(s)$ por F_{ES}. Por tanto, la función de transferencia escalada en magnitud tiene la expresión:

$$G_{EscMag}(s) = \frac{F_{ES} Y(s)}{F_{EE} X(s)} = F_{Mag} G(s)$$

Con F_{Mag} que representa el factor de escalamiento en magnitud. Obsérvese que:

$$F_{Mag} = \frac{F_{ES}}{F_{EE}} = \frac{X_{Max}}{Y_{Max}} \quad (5.23)$$

Aplicando el escalamiento de magnitud a las funciones de transferencia (5.21) y (5.22) se tiene:

$$G_1\left(h, X_{Max}, Y_{Max}, s\right) = \frac{\frac{X_{Max}}{Y_{Max}}\left[\left(m_1 + m_2\right)h^2 s^2 + b_2 hs + k_2\right]}{m_1 m_2 h^4 s^4 + \left(m_1 b_1 + b_1 m_2 + m_1 b_2\right)h^3 s^3 + \left(m_1 k_2 + k_1 m_2 + b_1 b_2 + m_1 k_1\right)h^2 s^2 + \left(k_1 b_2 + b_1 k_2\right)hs + k_1 k_2} \quad (5.24)$$

$$G_2\left(h, X_{Max}, Y_{Max}, s\right) = \frac{\frac{X_{Max}}{Y_{Max}}\left(-m_1 b_2 h^3 s^3 - m_1 k_2 h^2 s^2\right)}{m_1 m_2 h^4 s^4 + \left(m_1 b_1 + b_1 m_2 + m_1 b_2\right)h^3 s^3 + \left(m_1 k_2 + k_1 m_2 + b_1 b_2 + m_1 k_1\right)h^2 s^2 + \left(k_1 b_2 + b_1 k_2\right)hs + k_1 k_2} \quad (5.25)$$

Las ecuaciones (5.24) y (5.25) son las funciones de transferencia $G_1(s)$ y $G_2(s)$ escaladas en tiempo y magnitud, donde h es el factor de escalamiento de tiempo, X_{Max} y Y_{Max}, son los valores máximos estimados de las variables de entrada y salida, respectivamente.

Es importante recordar que si se realiza un escalamiento en el tiempo de una función de transferencia, al simular ésta en el SDSLI, también se debe escalar el tiempo de muestreo con el mismo factor de escala, para evitar que la simulación se submuestree.

$$Tm_{Esc} = hTm \quad (5.26)$$

Donde:

$Tm_{Esc} =$ Tiempo de muestreo escalado.

$Tm =$ Tiempo de muestreo.

5.5. SIMULACIÓN DE LA RESPUESTA EN LAZO ABIERTO DE LAS FUNCIONES DE TRANSFERENCIA

Es posible utilizar el SDSLI, para observar el comportamiento en lazo abierto de $G_1(s)$ y $G_2(s)$, escaladas en tiempo y magnitud.

Realizando la sustitución y simplificación numérica en la función de transferencia (5.24) con un escalamiento $1{:}1$ en el tiempo y de $66666.67{:}1$ en la magnitud la función de transferencia $G_1(s)$ escalada queda de la siguiente manera:

$$G_1(s) = \frac{26269025.287s^2 + 835049227.1133s + 4378170881.1667}{10649.1001s^4 + 4684995.5447s^3 + 33366977.6207s^2 + 201967790.7888s + 958419013.2333} \quad (5.27)$$

Introduciendo la función de transferencia (5.27) en el SDSLI, se observa lo siguiente:

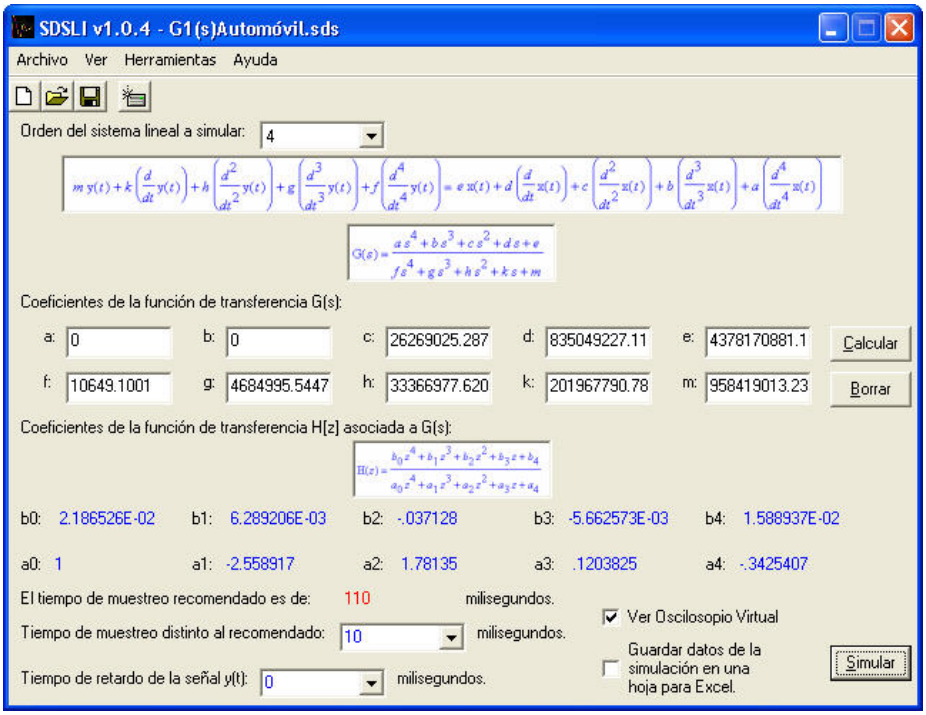


Figura 5.5. Función de transferencia $G_1(s)$ en la interfaz para Windows del SDSLI

No es necesario escalar el tiempo de muestreo, pues el tiempo de $10(ms)$ distinto al recomendado por el SDSLI, es el tiempo de muestreo adecuado para realizar la simulación con claridad.

Al ejecutar la simulación se obtiene la siguiente gráfica:

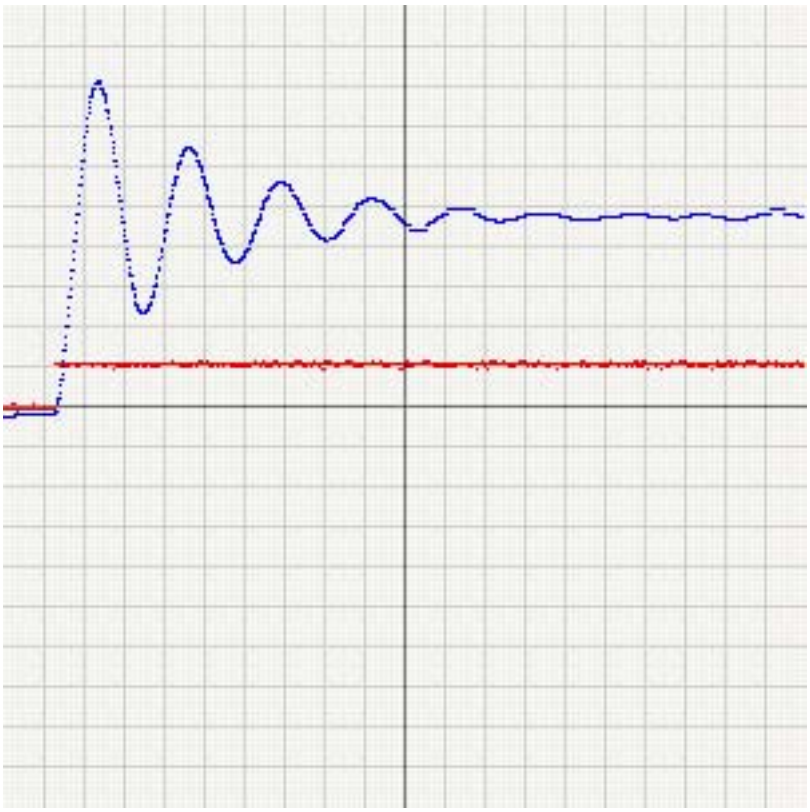


Figura 5.6. Simulación de la función de transferencia $G_1(s)$ en el SDSLI

La gráfica de la figura 5.6 tiene los siguientes parámetros:

Unidades:

Abscisas: $[tiempo]_u = (ms)$.

Ordenadas: $[voltaje]_u = (V)$.

Escala:

$Tiempo = 450\left(\frac{ms}{cuadro}\right)$.

$Voltaje = 1\left(\frac{V}{cuadro}\right)$.

Leyenda:

Color rojo: Señal de entrada $x(t)$.

Color azul: Señal de salida $y(t)$.

Para la señal de entrada $x(t) = u(t) = u_{-1}(t)$, la cual corresponde a un escalón unitario de $1(V)$ que simula a un escalón unitario de fuerza del actuador, se tiene una respuesta subamortiguada, como es posible apreciar en la figura 5.6.

La duración de la respuesta es de aproximadamente $9(s)$ de tiempo real.

La magnitud de la respuesta se amortigua al pasar el tiempo, oscilando entre los $0(V)$ de magnitud máquina (que equivalen a los $0(m)$ de magnitud real) y los $8(V)$ de magnitud máquina (equivalentes aproximadamente a $1.2\times10^{-4}(m)$ de magnitud real), estabilizándose la respuesta en $4.7(V)$ de magnitud máquina cuyo valor equivale a los $7.05\times10^{-5}(m)$ de magnitud real.

Las personas que viajen en el automóvil, sentirían pequeñas oscilaciones. Pero el automóvil tomaría un inaceptable y largo tiempo, para regresar a su estado estable. Es decir, el tiempo de asentamiento es muy grande. La solución a este problema, es añadir al sistema, un controlador realimentado.

Realizando la sustitución y simplificación numérica en la función de transferencia (5.25), con un escalamiento $1{:}1$ en el tiempo y de $1{:}1$ en la magnitud, la función de transferencia $G_2(s)$ escalada, queda de la siguiente manera:

$$G_2(s) = \frac{-4569985.2631s^3 - 23960475.3308s^2}{10649.1001s^4 + 4684995.5447s^3 + 33366977.6207s^2 + 201967790.7888s + 958419013.2333} \quad (5.28)$$

Tecleando la función de transferencia (5.28) en el SDSLI, se aprecia lo siguiente:

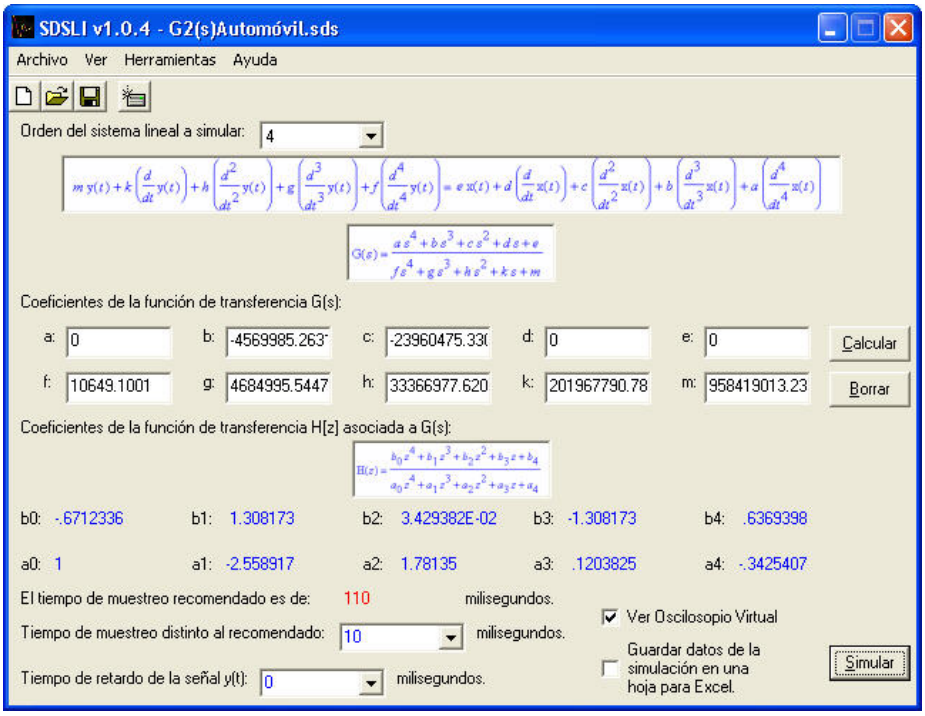


Figura 5.7. Función de transferencia $G_2(s)$ en la interfaz para Windows del SDSLI

También para este caso, no es necesario escalar el tiempo de muestreo, pues el tiempo de $10(ms)$ distinto al recomendado por el SDSLI, es el tiempo de muestreo adecuado para realizar la simulación.

Al ejecutar la simulación se obtiene la siguiente gráfica:

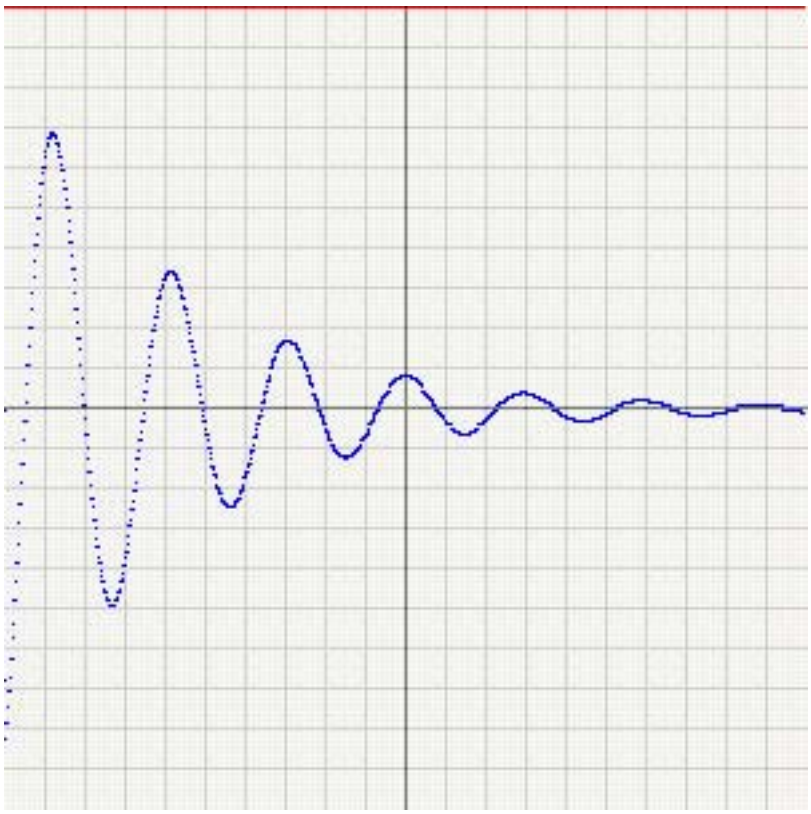


Figura 5.8. Simulación de la función de transferencia $G_2(s)$ en el SDSLI

La gráfica de la figura 5.8 tiene los siguientes parámetros:

Unidades:

Abscisas: $[tiempo]_u = (ms)$.

Ordenadas: $[voltaje]_u = (V)$.

Escala:

$$Tiempo = 350\left(\frac{ms}{cuadro}\right).$$

$$Voltaje = 1\left(\frac{V}{cuadro}\right).$$

Leyenda:

Color rojo: Señal de entrada $x(t)$.

Color azul: Señal de salida $y(t)$.

Para simular un bache de $10(cm)$ de alto, se utilizó la señal de entrada $x(t)=w(t)=10u_{-1}(t)$ la cual, corresponde a un escalón de $10(V)$ de magnitud. Nuevamente, se tiene una respuesta subamortiguada, como se puede observar en la figura 5.8.

La duración aproximada de la respuesta, es de $7(s)$ de tiempo real.

La magnitud de la respuesta se amortigua al transcurrir el tiempo, oscilando entre los $-10(V)$ de magnitud máquina (que equivalen a los $-0.1(m)$ de magnitud real) y los $7(V)$ de magnitud máquina (equivalentes a $0.07(m)$ de magnitud real), estabilizándose alrededor de los $0(V)$ de magnitud máquina cuyo valor se equipara a los $0(m)$ de magnitud real.

Para observar esto con mayor detalle, se observa la gráfica modificando en valor de $\left(\frac{ms}{cuadro}\right)$ del osciloscopio virtual del SDSLI. Teniendo como resultado la gráfica siguiente:

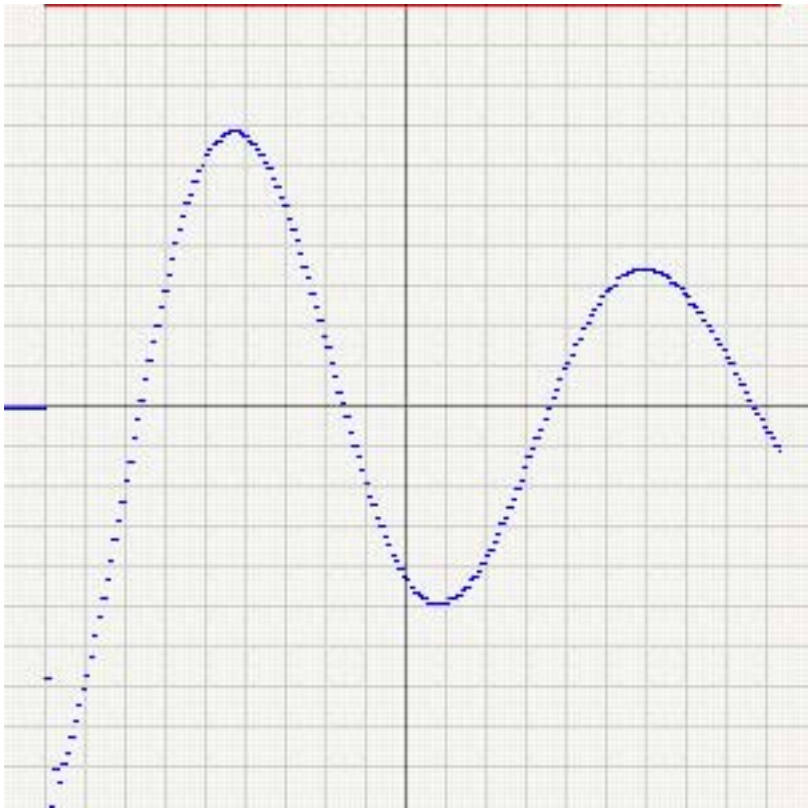


Figura 5.9. Acercamiento en el tiempo, de la simulación de la función de transferencia $G_2(s)$ en el SDSLI

La gráfica de la figura 5.9 tiene los siguientes parámetros:

Unidades:

Abscisas: $[tiempo]_u = (ms)$.

Ordenadas: $[voltaje]_u = (V)$.

Escala:

$$Tiempo = 100\left(\frac{ms}{cuadro}\right).$$

$$Voltaje = 1\left(\frac{V}{cuadro}\right).$$

Leyenda:

Color rojo: Señal de entrada $x(t)$.

Color azul: Señal de salida $y(t)$.

De las gráficas 5.8 y 5.9, para una perturbación que simula un bache de $0.1(m)$ de altura, se puede concluir que, cuando el automóvil pasa un bache de $10(cm)$ de alto en la carretera, el cuerpo del automóvil oscilaría por un tiempo inaceptable de casi $7(s)$, con una amplitud mayor al impacto inicial (aproximadamente $17(cm)$). Los pasajeros no se sentirán cómodos con tales oscilaciones. El gran sobretiro (del propio impacto) y el tardado tiempo de asentamiento, causarían además, daños en el sistema de suspensión. Nuevamente, la solución a este problema es agregar un controlador realimentado al sistema, con la finalidad de mejorar el desempeño de éste.

Debido a que no se dispone físicamente del sistema real, para poder comparar los resultados obtenidos de las simulaciones con el SDSLI, y aunque se tuviera el sistema físicamente; la obtención de estas respuestas es difícil, pues no sería fácil obtener la respuesta gráfica a una entrada escalón en el sistema real. Se ha decidido utilizar MATLAB para validar los resultados obtenidos.

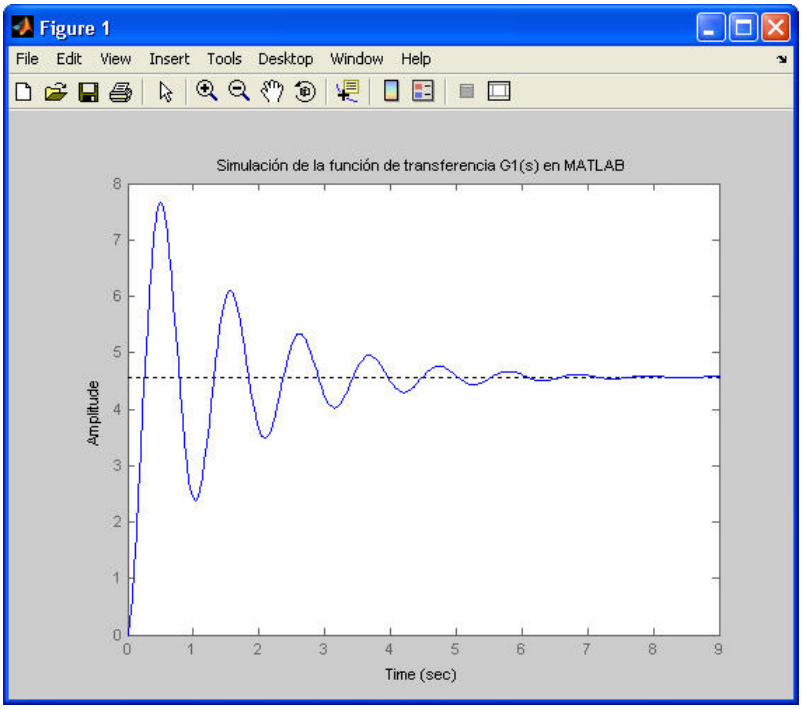


Figura 5.10. Simulación de la función de transferencia $G_1(s)$ en MATLAB

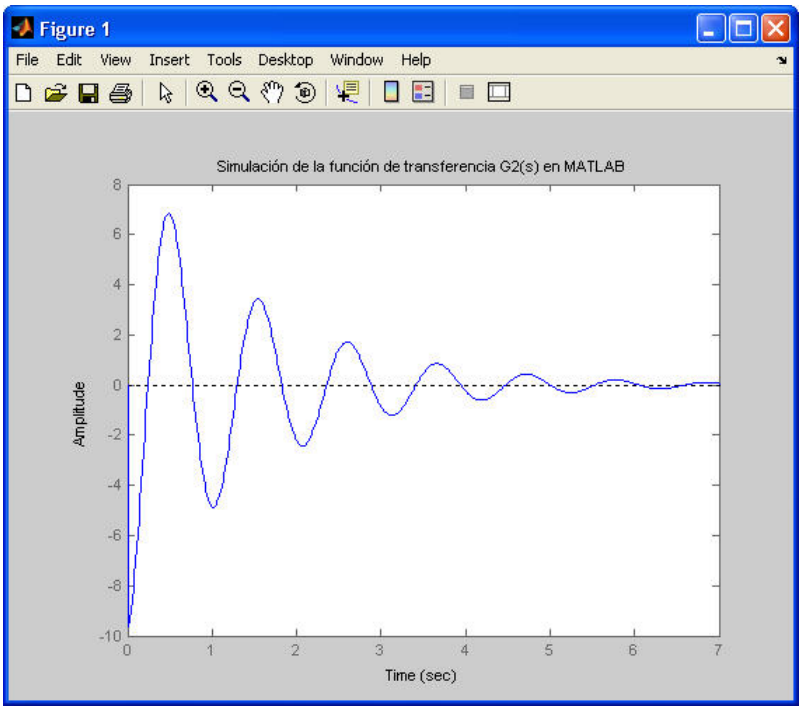


Figura 5.11. Simulación de la función de transferencia $G_2(s)$ en MATLAB

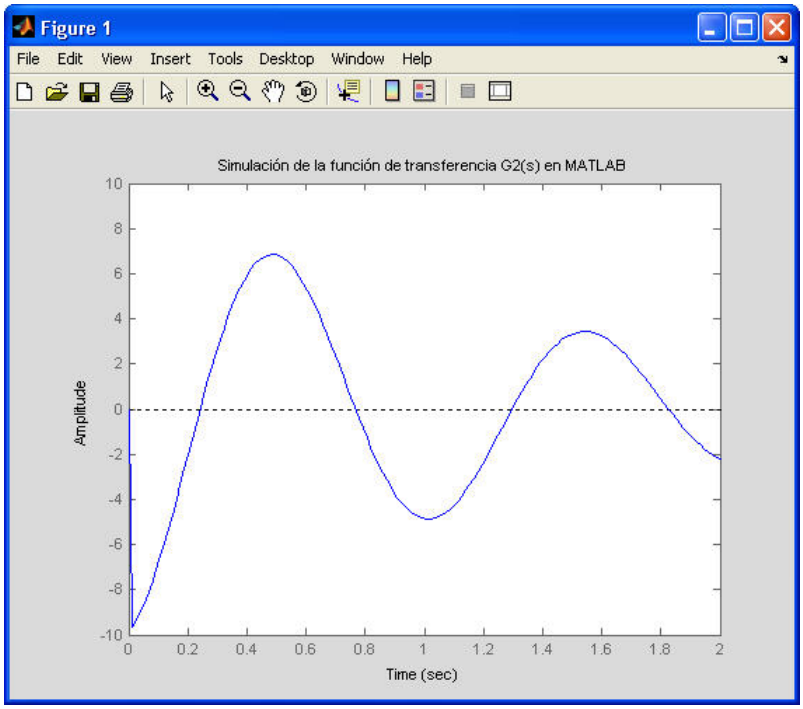


Figura 5.12. Acercamiento de la simulación de la función de transferencia $G_2(s)$ en MATLAB

Las respuestas a la entrada escalón que se obtuvieron con el SDSLI, se compararon con las producidas por MATLAB, con funciones de transferencia iguales a las funciones de transferencia (5.27) y (5.28). Dichas comparaciones resultaron satisfactorias, mostrando de esta manera, la viabilidad del simulador para reproducir el comportamiento de sistemas dinámicos.

Este ejemplo fue tomado del sitio de la Universidad de Michigan, Carnegie Mellon, Control Tutorials for Matlab. La página donde se ubica este ejemplo, se encuentra en la referencia **[UMich 1997]**.

En tanto a los datos numéricos obtenidos para realizar la simulación, éstos fueron tomados de la referencia **[Kuo 1965]**.

6

CONCLUSIONES

En esta sección encontrará:

6. CONCLUSIONES

6.1. CONCLUSIONES GENERALES

- El problema propuesto en el capítulo 3, ha sido resuelto satisfactoriamente, como se muestra en la sección 4. Se logró diseñar e implementar hardware y software para crear un Simulador Digital de Sistemas Lineales e Invariantes (SDSLI) de primer y hasta cuarto orden.

- El objetivo principal, fue la obtención de un bloque funcional configurable, que permita desarrollar simulaciones de Sistemas Lineales e Invariantes en el Tiempo (SLIT) de primer y hasta cuarto orden, cuya respuesta en el tiempo sea relativamente lenta y sus constantes de tiempo más pequeñas, no sean menores de $0.25(s)$.

- Fue posible utilizar la arquitectura FACIL_11B, junto con una tarjeta adicional llamada SDSLI_B, que implementa un Convertidor Digital a Analógico, un acondicionador de señales de entrada y una fuente de poder. Todo esto, para construir el hardware del simulador.

- Los dispositivos electrónicos diseñados e implementados para la creación del SDSLI son:

1. Un módulo convertidor digital a analógico de ocho bits, utilizando el chip DAC0800 de National Instruments.
2. Una etapa de acondicionamiento de señales de entrada, para acondicionar las señales de entrada al convertidor analógico a digital del microprocesador 68HC11F1.
3. Una fuente de poder y un transformador, que sean capaces de utilizar la corriente alterna suministrada por un toma corriente mexicano, para alimentar al

hardware del simulador.

- El software diseñado e implementado, para el control y manejo del hardware del SDSLI es el siguiente:

1. Un programa en lenguaje C, que evalúa periódicamente la ecuación en diferencias correspondiente al SLIT que se desee simular. Este programa tiene rutinas básicas de software para la recepción y el envío de datos a la PC vía puerto serial. El software se ubica en forma de firmware, en una memoria EPROM colocada en la tarjeta FACIL_11B.
2. Una interfaz gráfica que permite al usuario del simulador, controlar y desarrollar con facilidad la simulación de los sistemas lineales invariantes en el tiempo. Este software escrito en gran parte en Visual Basic, contiene módulos para el cálculo de una función de transferencia discreta, a partir de una función de transferencia continua (válidos sólo para sistemas de primer y hasta cuarto orden), módulos para la comunicación (vía puerto serie) e inicialización del hardware del SDSLI, además de módulos para la carga y ejecución de programas en la tarjeta FACIL_11B.
3. Pequeños módulos de software escritos en lenguaje ensamblador para el microcontrolador 68HC11F1, que permiten implementar órdenes de la interfaz en Windows para el SDSLI y utilerías de calibración y configuración del hardware, para su operación en modo autónomo.

- Además del hardware electrónico y el software del SDSLI, se diseño e implementó un gabinete que contiene las tarjetas SDSLI_B y FACIL_11B, switches, bornes y leds. Este gabinete, no estándar, tiene el objetivo de proteger la electrónica del hardware del SDSLI y ayudar a su transporte y almacenamiento.

- Fue posible actualmente, diseñar e implementar hardware y software que trabajan en conjunto, para poder desarrollar una simulación de sistemas lineales e invariantes en el tiempo, de primer y hasta cuarto orden.

- El SDSLI es un prototipo de auxilio didáctico, que se podrá utilizar en laboratorios de la Facultad de Ingeniería. Aunque sólo se pueden simular sistemas con constantes de tiempo grandes, como es el caso de muchos sistemas en la industria, permite que sus usuarios puedan verificar y comprobar en forma práctica, los conceptos que se estudian en diversos cursos relacionados con teoría de control y disciplinas afines.

- El SDSLI desarrollado, opera en dos modalidades, la modalidad cliente (esclavo) y la modalidad autónoma. La primera modalidad, permite al usuario, mediante una PC con Windows, realizar una simulación y poder observar en el Osciloscopio Virtual de la interfaz del SDSLI, las señales de entrada y salida de la simulación. La segunda modalidad de operación, permite al usuario dejar configurado el hardware del SDSLI de tal manera que, cuando éste sea encendido, una simulación comience a ejecutarse sin requerir de órdenes extras por parte del usuario. Esta segunda modalidad es útil cuando se desea simular plantas o controladores, que se quedarán fijos y se pueden conectar a otros dispositivos como por ejemplo, otro SDSLI, alguna planta industrial o con fines educativos o bien, algún controlador. Sin embargo, para visualizar la forma de la respuesta en esta modalidad, es necesario contar con un osciloscopio o un graficador.

- Además de lo antes mencionado, el SDSLI se ha diseñado con la finalidad de utilizarlo en un esquema de control de lazo cerrado, empleando al mismo, para la realización de la planta a controlar, esto en conjunción con otro prototipo: CONTROLADOR DIGITAL PARA AUXILIO DIDÁCTICO BASADO EN EL MICROCONTROLADOR 68HC11, ya desarrollado en el Departamento de

Ingeniería de Control de la Facultad de Ingeniería **[V. Sánchez 2000]**, que permita implementar y verificar la validez de la teoría de control.

- A la fecha de realización de este trabajo, se han fabricado en serie, seis prototipos del hardware del SDSLI y se cuenta con una versión instalable en Windows, de la interfaz del SDSLI, cuya liberación corresponde a la versión 1.0.4.

6.2. RESUMEN DE CONTRIBUCIONES

- Se ha desarrollado un simulador basado en la arquitectura FACIL_11B, que es amigable y sencillo de utilizar. Su uso no se interpone con el objetivo principal de comprobación de la teoría respectiva, vista en clase, o bien, en la experimentación e investigación de sistemas lineales.

- El simulador desarrollado, permite realizar simulaciones de un grupo grande de sistemas lineales, con la finalidad de poder desarrollar distintas prácticas o investigaciones, sin tener que dedicar un simulador a una práctica, aplicación o investigación en particular, y sin verse obligados a adquirir otro simulador distinto, para realizar una práctica diferente, ahorrando tiempo y dinero a la Facultad de Ingeniería de la UNAM.

- El simulador requiere de alambrado mínimo, es robusto y está protegido para evitar daños al equipo y así, aprovechar el tiempo en realizar más corridas de simulación, permitiendo a los alumnos enfocarse en el objetivo de comprobación de la teoría vista en clase.

- El dispositivo desarrollado requiere mantenimiento mínimo y fácil de realizar.

- El simulador evita al alumno o al investigador, la tediosa tarea de programar para poder realizar una simulación, además de dotar al usuario, de herramientas que realicen la captura y almacenamiento de datos, que ayuden en el análisis posterior de los mismos.

- El simulador permite que de una simulación se pueda obtener, con facilidad, una señal eléctrica de salida y que a su vez, se pueda excitar al sistema con una señal eléctrica de entrada, teniendo la posibilidad de ser observado y medido esto, en instrumentos como el osciloscopio. De esta manera, no sólo es posible obtener señales almacenadas en la computadora, sino también poder experimentarlas de manera física.

- El problema del tiempo de muestreo que se observa en las computadoras digitales, como las computadoras de escritorio, fue posible resolverlo, dedicando una pequeña computadora digital a la tarea exclusiva de la simulación. La computadora digital se implementó con un microcontrolador 68HC11F1, montado en la arquitectura FACIL_11B.

- No se pretende desplazar a los equipos analógicos ni al software que posee la Facultad de Ingeniería de la UNAM, para desarrollar simulaciones de sistemas lineales sino, complementar las herramientas de simulación con un simulador digital, teniendo en cuenta las limitaciones intrínsecas de una computadora digital.

- Se ha proporcionado una metodología básica, herramientas, código binario de software y dibujos para elaboración de circuitos impresos, con la finalidad de la futura elaboración de más simuladores para uso interno de la facultad.

6.3. FUTURA INVESTIGACIÓN

- Con el hardware empleado y el software adecuado, se puede lograr la simulación de bloques funcionales con características entrada - salida no lineales.

- El firmware desarrollado para el hardware del SDSLI, se encuentra escrito en lenguaje C y cumple con los estándares mínimos de portabilidad, permitiendo que con pequeñas modificaciones, este software pueda ser compilado para arquitecturas 68HC12, 68HC05 68HC908 y se pueda implementar un SDSLI basado en estas arquitecturas.

- La factibilidad del uso de una arquitectura distinta a la FACIL_11B es posible. Si se desea implementar simuladores digitales con mayor potencia, se deben utilizar microcontroladores de otras familias, microprocesadores más rápidos o bien, lo ideal, sería el uso de una arquitectura basada en algún DSP. Esto, con la finalidad de simular sistemas de mayor orden y con requerimientos de procesador y memoria mayores, como aquellos que se requieren, para la simulación e implementación de filtros digitales.

- Si se decide cambiar a otra arquitectura distinta a la FACIL_11B, es necesario verificar que dicha arquitectura tenga la disponibilidad de al menos, dos puertos de ocho bits, circuitería de conversión Analógica a Digital y por lo menos, 512 bytes de memoria RAM y 32 (Kb) de memoria EPROM o EEPROM. El software desarrollado en lenguaje C, puede servir como una base robusta para implementar el software para esta nueva arquitectura. Respecto a la interfaz en Windows, se requeriría hacer modificaciones mínimas, si el código binario que actualmente se descarga a la arquitectura FACIL_11B, mediante el protocolo PUMMA, es compilado por un módulo de software (que se requeriría diseñar e implementar) y que haga las veces de interfaz de software, entre la interfaz

actual para Windows del SDSLI y la nueva arquitectura destino. Este módulo, tendría que compilar el código binario actual, en un código binario apropiado al formato que se requiera en la nueva arquitectura y su talker respectivo, para poder ejecutar y realizar la simulación.

- El firmware del SDSLI y el software de interfaz del SDSLI, como todo software con propósitos serios, se encuentran en depuración y mejoras constantes. Se requiere mejorar el Osciloscopio Virtual para que se logren gráficas de mayor calidad y precisión. Mejores gráficas, permitirán al usuario realizar con mayor facilidad y por lo tanto claridad, sus análisis.

- Sería importante la implementación de módulos de software dentro de la interfaz para Windows, que permitan al usuario escalar tanto en tiempo como en magnitud los sistemas a simular, ahorrando al usuario cálculos extras para llevar acabo una simulación. También sería conveniente la adición de módulos dentro de la interfaz de Windows, que permitan simular bloques funcionales con características entrada - salida no lineales.

- Otro avance importante que podría contemplar la interfaz para Windows del simulador, sería la adición de un módulo para graficar y analizar polos y ceros del sistema, además de un módulo para hacer análisis y graficación en el dominio de la frecuencia, del sistema a simular y así, graficar la respuesta en frecuencia de éste.

- Respecto al hardware del SDSLI, sería conveniente añadirle un módulo que permita la adición de señales en un rango de $-10(V)$ a $10(V)$, para introducir el resultado de esta suma a la entrada del simulador y así, implementar simulaciones de sistemas en lazo cerrado con otros SDSLI u otro hardware que opere en los intervalos de voltaje del SDSLI. Esto con el objetivo de evitar que el

usuario tenga que construir hardware adicional para sumar señales, y facilitar la implementación de simulaciones de sistemas en lazo cerrado.

- Con la finalidad de tener señales de prueba de entrada al SDSLI, sería interesante adicionar al hardware, un pequeño módulo generador de señales con la finalidad de tener fácilmente, las señales de prueba más comunes como son: el escalón unitario, la señal rampa, etcétera. Otra característica importante, una vez que se tenga el generador de señales, podría ser la adición de algún mecanismo, como por ejemplo un switch, que permita sincronizar la señal de entrada al SDSLI, para poder tener con mayor precisión y facilidad, la respuesta de un sistema simulado en $t = 0$.

- Una buena idea, es la implementación de diversas interfaces de adecuación y potencia, para poder implementar controladores de hardware externo, como motores, bombas u otro tipo de plantas que requieran para su operación, un consumo grande de corriente eléctrica.

- Sería importante, la elaboración de prácticas de laboratorio en las cuales se simulen sistemas lineales mediante el SDSLI y además, se implementen controladores digitales, con la finalidad de que el alumno tenga la oportunidad de experimentar físicamente con las señales producidas por el SDSLI y así, tener una mejor y más clara comprensión de la teoría de control vista en clase.

7

ANEXOS

En esta sección encontrará:

- **MANUAL TÉCNICO DEL SDSLI**
- **MANUAL DEL USUARIO DEL SDSLI**

MANUAL TÉCNICO DEL SDSLI

En esta sección encontrará:

1. TARJETAS Y TRANSFORMADOR DEL SDSLI

1.1. TARJETA FACIL_11B

Las características principales de la tarjeta FACIL_11B, arquitectura en la cual el simulador esta basado, son:

1. Está basada en el microcontrolador 68HC11F1 y puede operar en cualquiera de los cuatro modos asociados con el 68HC11 en general.
2. Contiene un firmware interlocutor que permite manejarla vía enlace serie desde una PC mediante la ejecución del software clásico para este fin como PCBUG11, o bien, del manejador visual PUMMA_11 que corre bajo WINDOWS y contempla los comandos de manejo más usuales, además de algunos otros adicionales asociados con características propias de la tarjeta FACIL_11B.
3. Es compatible con otras herramientas de software asociadas con el microcontrolador 68HC11, lo cual permite la ejecución en la tarjeta, de los programas originalmente escritos en lenguaje C o ensamblador, lográndose esto, mediante la carga y ejecución del archivo objeto S19 que haya sido generado por el software de ensamble o compilación respectivo.
4. Tiene la capacidad para configurar diversos mapas de memoria al operar en modo expandido; por ejemplo: 8 (Kb) de RAM y 8 (Kb) de EPROM, o bien, 32 (Kb) de RAM y 32 (Kb) de EPROM.
5. Puede configurarse para ser energizada con una fuente de laboratorio de cinco volts, o bien con un eliminador de baterías.
6. Contiene postes para conexión de unidades desplegadoras alfanuméricas comunes en la industria, así como también, postes para conexión de teclados de 4 x 4.
7. Posee dos puertos de entrada y dos puertos de salida visibles al operar en modo expandido, además de líneas de paginación de puerto adicionales, que permiten al usuario experimentar con la conexión de dispositivos externos tales como: puertos

serie o paralelos adicionales, chips de reloj, o hardware específico diseñado para una determinada aplicación.

8. Contiene un circuito para respaldo de RAM externa al MCU (esto requiere una batería de 3(V) a 4.5(V) conectada al conector 10).
9. Además, es posible utilizar el programador integrado de memorias EPROM manejado por opciones especiales del software PUMMA_11. Las memorias que pueden ser programadas son: 27C64, 27C128, 27C256 y 27C512. Esta característica, permite efectuar el desarrollo de una aplicación desde la prueba y depuración del software asociado con la misma, hasta la programación final de la memoria EPROM requerida para almacenar el código correspondiente para la ejecución autónoma, todo en un solo equipo.

A continuación, se muestra la tarjeta FACIL_11B.

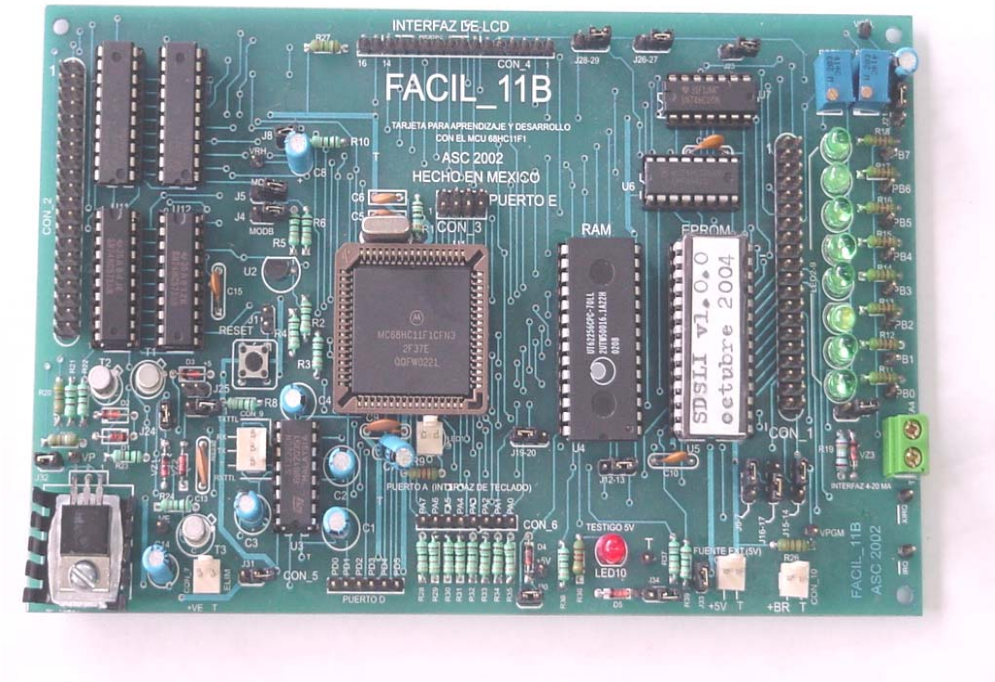


Figura 1. Tarjeta FACIL_11B

1.2. TARJETA SDSLI_B

Esta tarjeta consta principalmente de tres circuitos:

1. Circuito de la fuente de poder.
2. Circuito acondicionador de la señal de entrada.
3. Circuito del convertidor digital a analógico de 8 bits.

En la siguiente figura, se puede observar el circuito impreso de la tarjeta SDSLI_B.

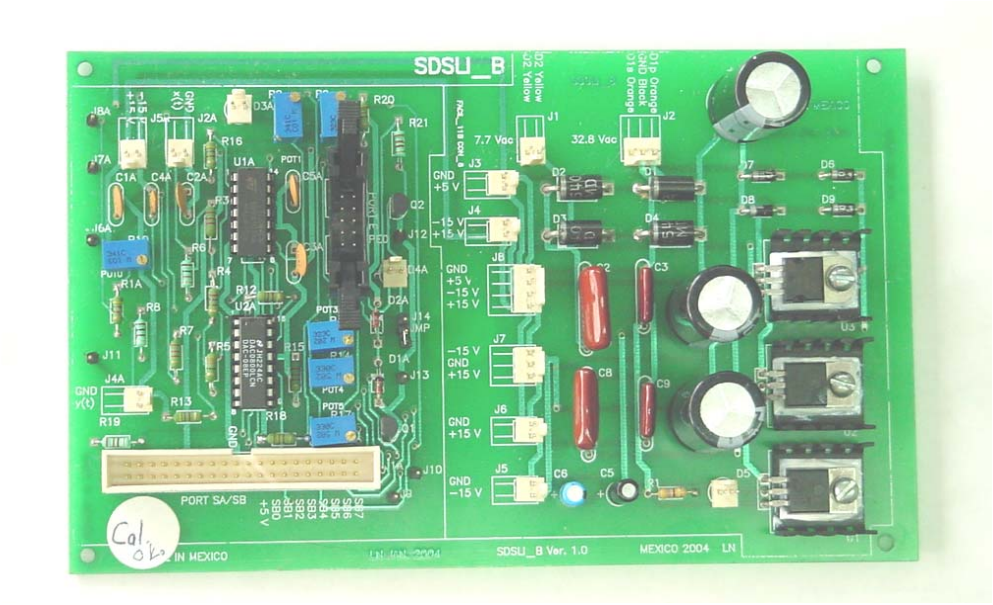


Figura 2. Tarjeta SDSLI_B

1.3. TRANSFORMADOR DEL SDSLI

Para poder explicar el funcionamiento de los circuitos que componen esta tarjeta, se detalla brevemente el funcionamiento del transformador del SDSLI.

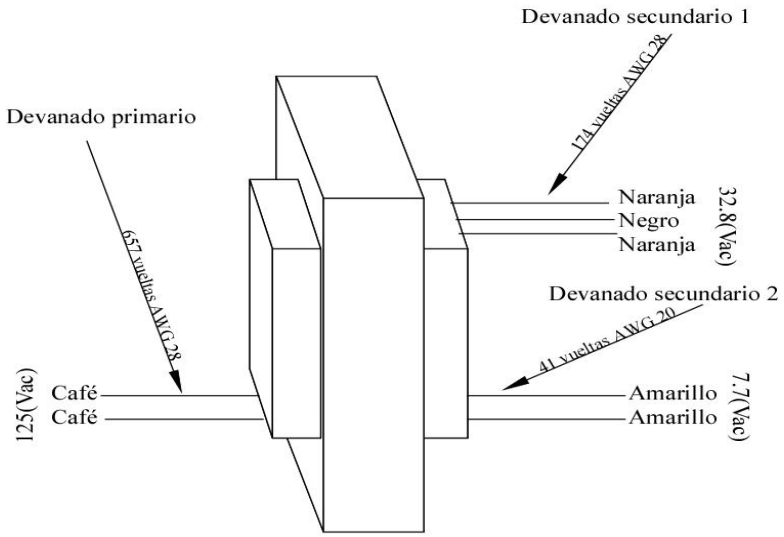


Figura 3. Transformador del SDSLI

La figura mostrada en la parte superior, figura 3, representa al transformador del SDSLI, que se encarga de obtener del toma corriente, $125(Vac)$ a $127(Vac)$ en su devanado primario y de entregar por un lado, $32.8(Vac)$ @ $0.5(A)$ y por el otro, $7.7(Vac)$ @ $1(A)$, en sus devanados secundarios.

2. FUNCIONAMIENTO DE LOS CIRCUITOS QUE COMPONEN LA TARJETA SDSLI_B

A continuación, se explican de manera concisa, los circuitos que forman parte de la tarjeta SDSLI_B.

2.1. FUENTE DE PODER

El circuito de la fuente de poder es el encargado de convertir la corriente alterna entregada por el transformador, en corriente directa para alimentar el resto de los circuitos que componen el simulador.

La figura 4, corresponde al diagrama esquemático de la fuente de poder del SDSLI.

En J1 se reciben $7.7(Vac)$ los cuales, son transformados a $5(Vdc)$ @ $1(A)$ para alimentar a la tarjeta del simulador SDSLI_B y a la tarjeta FACIL_11B.

En J2 se reciben $32.8(Vac)$, que son convertidos en $+15(Vdc)$ y $-15(Vdc)$ @ $0.5(A)$, utilizados para alimentar el circuito acondicionador de la señal de entrada y el circuito convertidor digital a analógico de la tarjeta SDSLI_B.

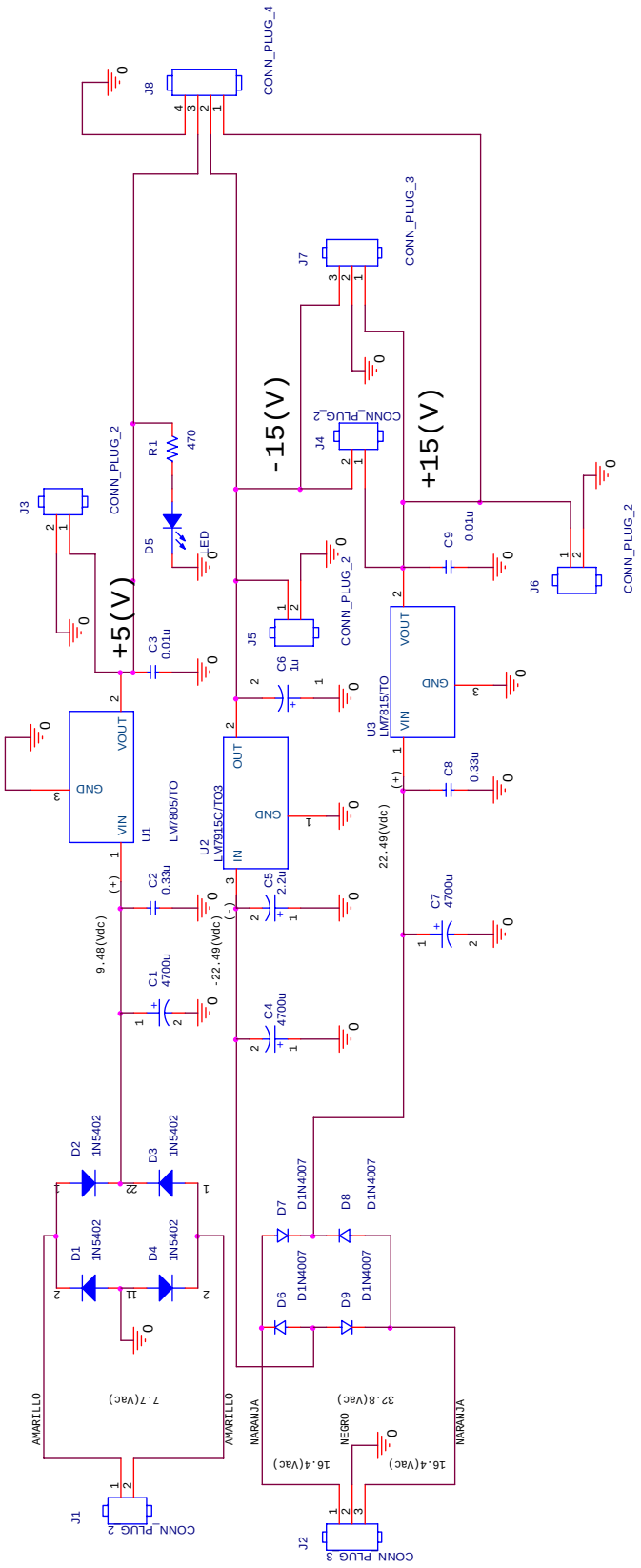


Figura 4. Fuente de poder del SDSLI

2.2. CIRCUITO ACONDICIONADOR DE LA SEÑAL DE ENTRADA Y CIRCUITO CONVERTIDOR DIGITAL A ANALÓGICO DE 8 BITS

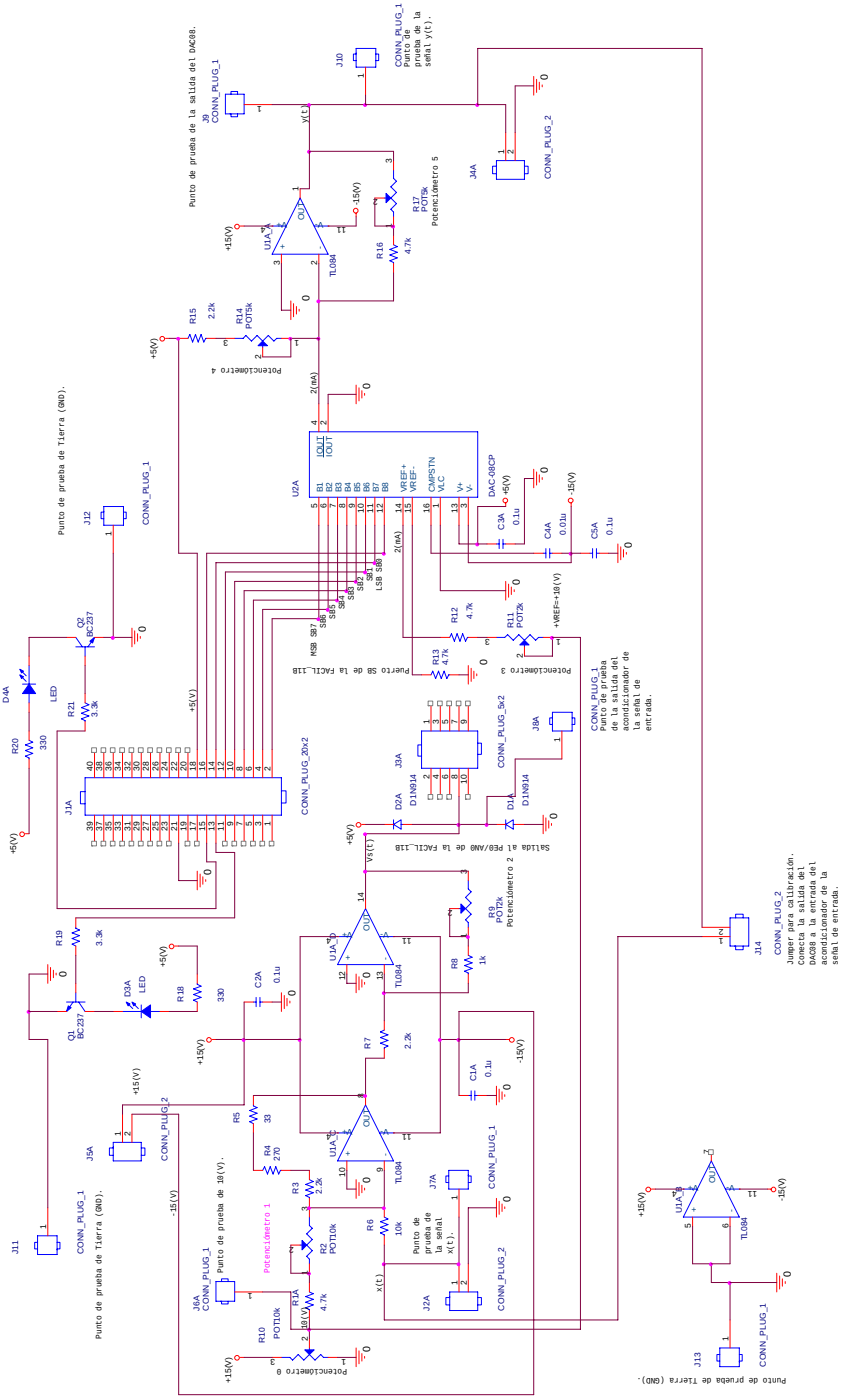


Figura 5. Circuito acondicionador de la señal de entrada al simulador y circuito convertidor digital - analógico del simulador

El diagrama esquemático correspondiente a la figura 5, muestra tanto el circuito acondicionador de la señal de entrada al simulador, como el circuito de conversión digital - analógica de 8 bits que entrega la señal de salida del SDSLI.

Para explicar de manera más simple el funcionamiento de ambos circuitos, se presentan a continuación, los diagramas esquemáticos por separado.

2.2.1. Circuito acondicionador de la señal de entrada al SDSLI

Debido a que el Convertidor Analógico - Digital (CAD) interno del microcontrolador 68HC11, integrado en la tarjeta FACIL_11B, requiere señales cuyo voltaje estén en el intervalo de $0(V)$ a $5(V)$, se necesita acondicionar la señal de entrada $x(t)$, ubicada en un rango de $-10(V)$ a $10(V)$ para que se ajuste a los valores de voltaje solicitados por el CAD interno del microcontrolador. Esto es precisamente de lo que se encarga el circuito de acondicionamiento de la señal de entrada del SDSL.

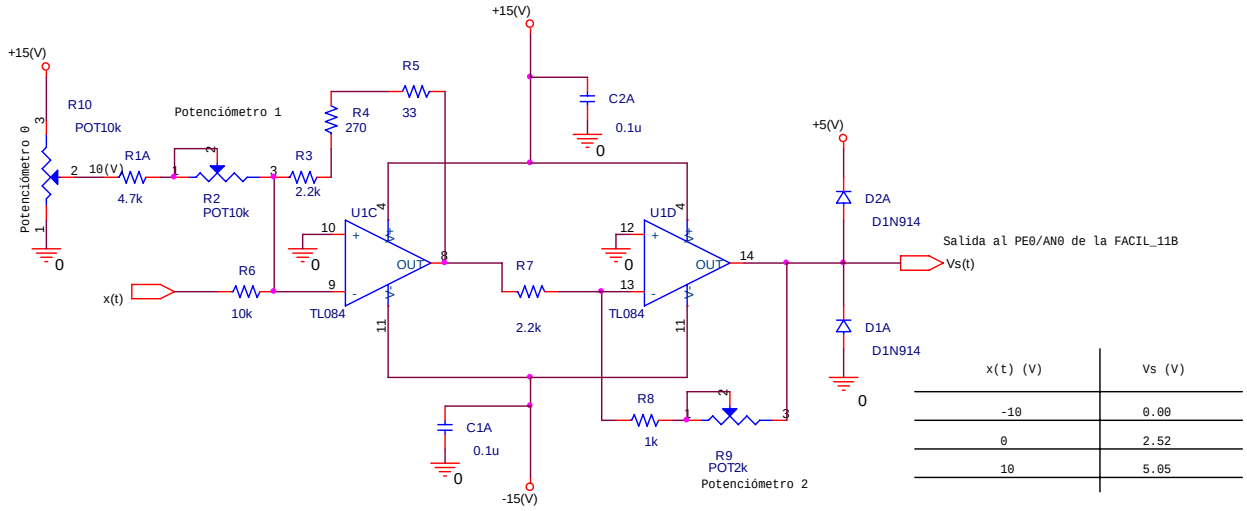


Figura 6. Circuito acondicionador de la señal de entrada al SDSLI

En el conector J2A, de la tarjeta SDSLI_B (ver figura 5), se introduce la señal de entrada $x(t)$. El valor de voltaje de esta señal, es sumado a un voltaje constante de

$10(V)$ y multiplicado por una ganancia de 0.25, de tal forma, que es posible obtener en el conector J3A (ver figura 5), la señal apropiada para el CAD interno del microcontrolador, cuya entrada se encuentra en el conector CON_3 de la tarjeta FACIL_11B.

En la figura 6, es posible visualizar por separado, el circuito de acondicionamiento de la señal de entrada al SDSLI, así como una pequeña tabla que muestra los valores críticos de calibración de la señal de entrada y de la señal de salida de este circuito.

La figura 7, muestra la curva característica de operación del circuito de acondicionamiento de la señal de entrada $x(t)$ al SDSLI. El eje de las abscisas, representa los valores posibles de voltaje de entrada (Ve) de la señal de entrada $x(t)$, y el eje de las ordenadas, muestra los valores correspondientes al voltaje de salida $(Vsalida)$ del circuito de acondicionamiento de la señal de entrada.

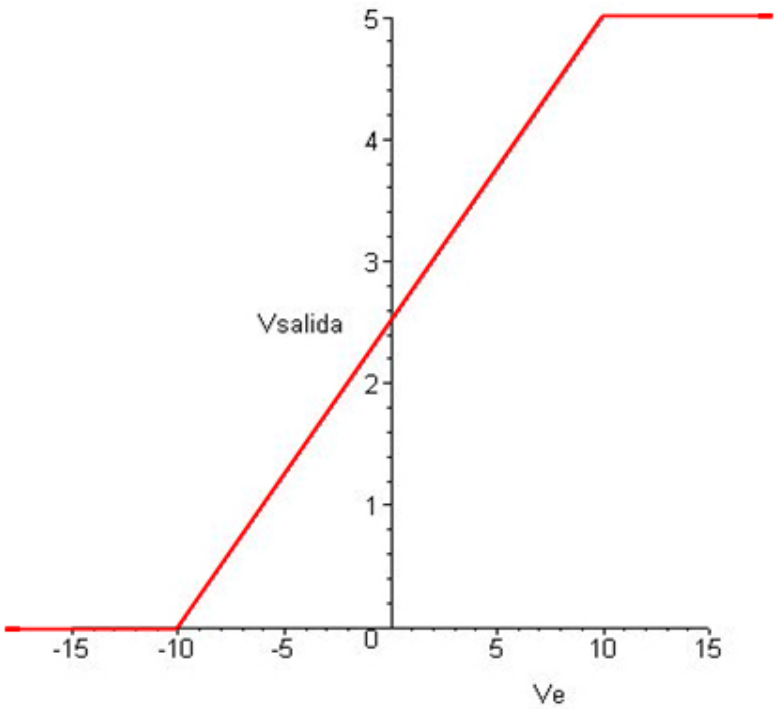


Figura 7. Curva característica de operación del circuito de acondicionamiento de la señal de entrada al SDSLI

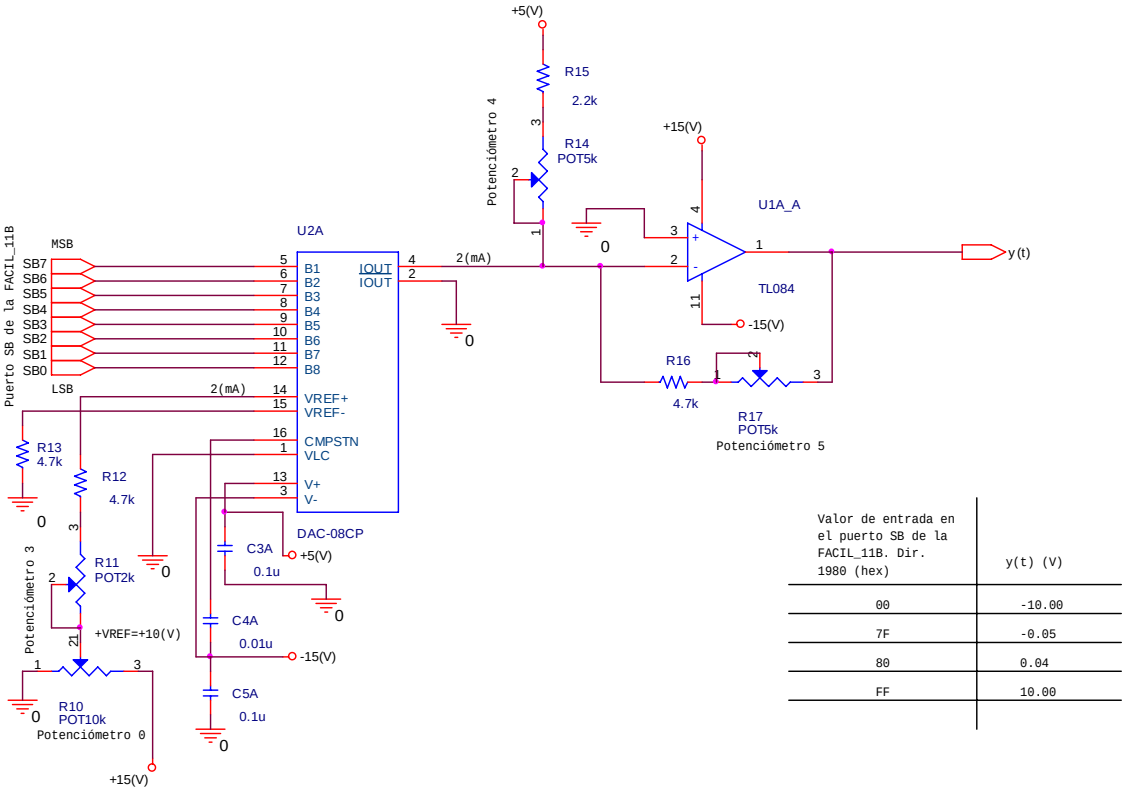
La ecuación matemática que representa a la curva característica de operación del circuito acondicionador de la señal de entrada al SDSLI es:

$$Vsalida = \frac{Ve}{4} + 2.5 \ \ (\mathrm{V})$$

Ecuación 1

2.2.2. Circuito Convertidor Digital - Analógico

Una vez que el microcontrolador ha obtenido, mediante la ecuación en diferencias, el valor de y[n] en un formato de 8 bits, este resultado es enviado al puerto de salida SB (CON_2) de 8 bits, propio de la tarjeta FACIL_11B (dirección 1980h), el cual a su vez, es enviado por cable al conector J1A (ver figura 5), de la tarjeta SDSLI_B. El Convertidor Digital - Analógico (CDA), toma el valor de 8 bits y lo transforma a un valor de voltaje de salida $y(t)$ (señal de salida del SDSLI), dentro del intervalo de $-10(V)$ a $10(V)$.



Valor de entrada en el puerto SB de la FACIL_11B. Dir. 1980 (hex)	y(t) (V)
00	-10.00
7F	-0.05
80	0.04
FF	10.00

Figura 8. Circuito Convertidor Digital - Analógico del SDSLI

La figura 8, presenta sólo el circuito de conversión digital a analógico del SDSLI, junto con una tabla que indica los valores de calibración de la señal de entrada y de la señal de salida de este circuito.

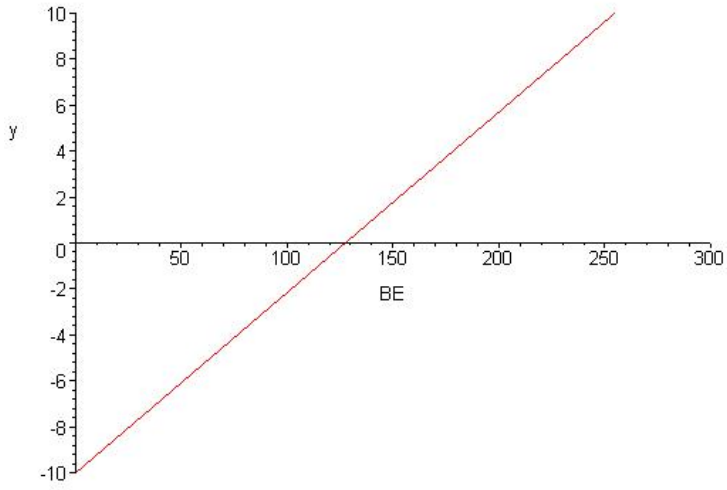


Figura 9. Curva característica de operación del circuito CDA del SDSLI

En la figura 9, es posible observar la curva característica de operación del circuito de CDA de la señal de salida $y(t)$ del SDSLI. El eje de las abscisas representa los valores del byte de entrada (BE) al CDA (en su respectivo valor decimal), y el eje de las ordenadas señala los valores correspondientes al voltaje de salida (y) del circuito CDA.

La ecuación matemática que representa a la curva característica de operación del circuito CDA del SDSLI es:

$$y = \frac{4BE}{51} - 10 \ \ (\mathrm{V})$$

Ecuación 2

3. DESCRIPCIÓN DE LOS POSTES Y LOS CONECTORES DE LA TARJETA SDSLI_B

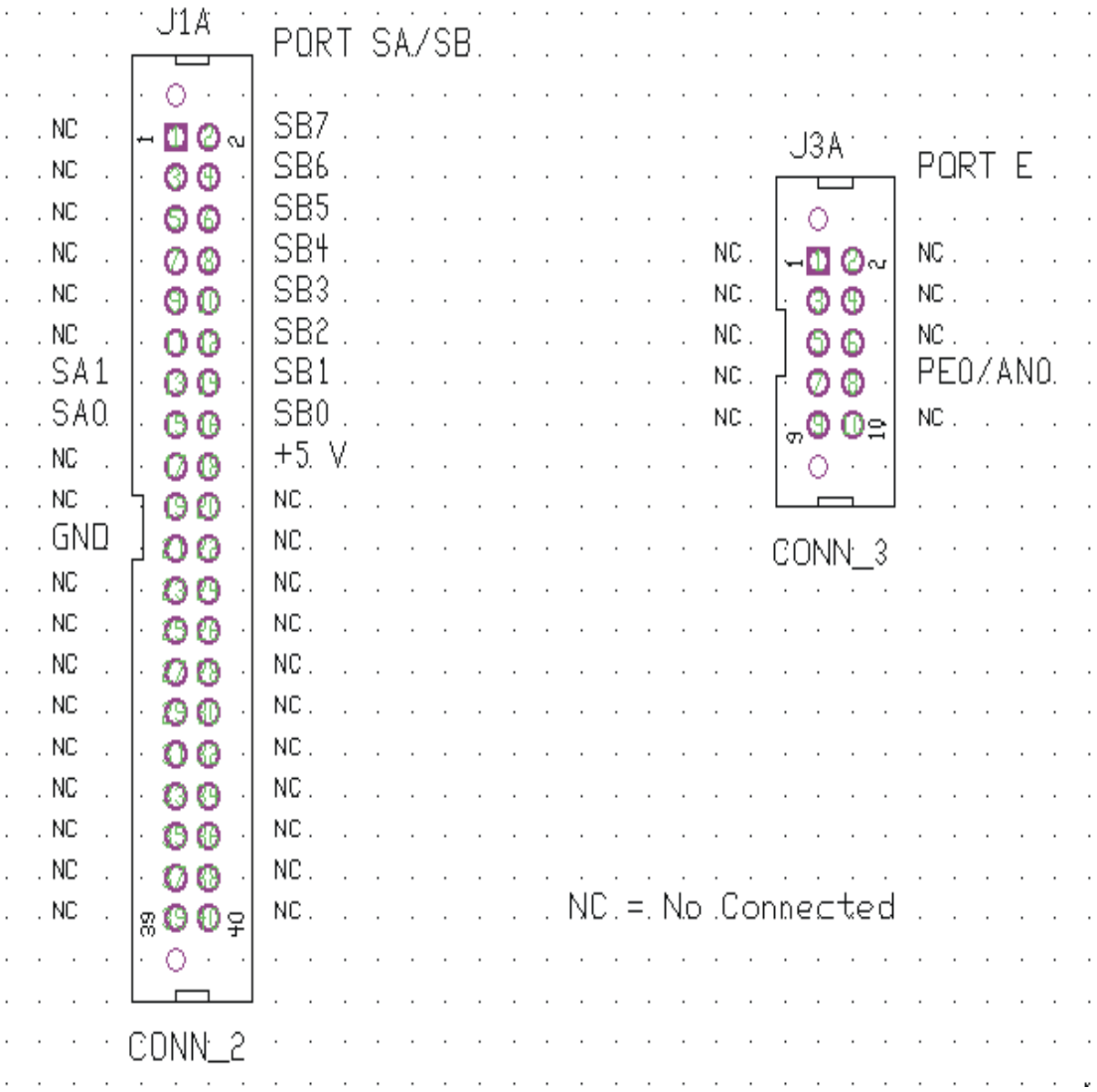


Figura 10. Conectores J1A y J3A de la tarjeta SDSLI_B

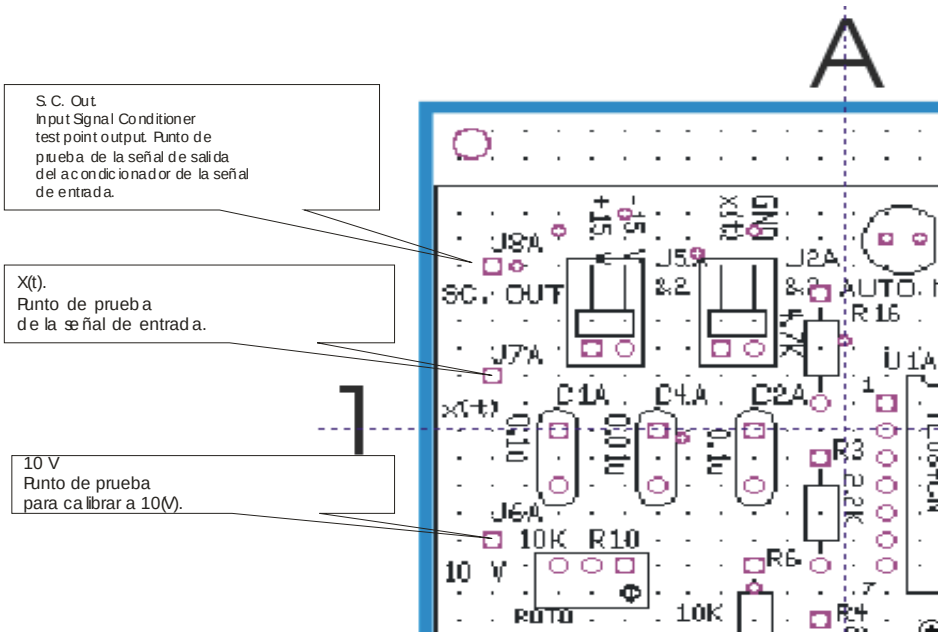


Figura 11. Puntos de prueba J6A, J7A y J8A de la tarjeta SDSLI_B

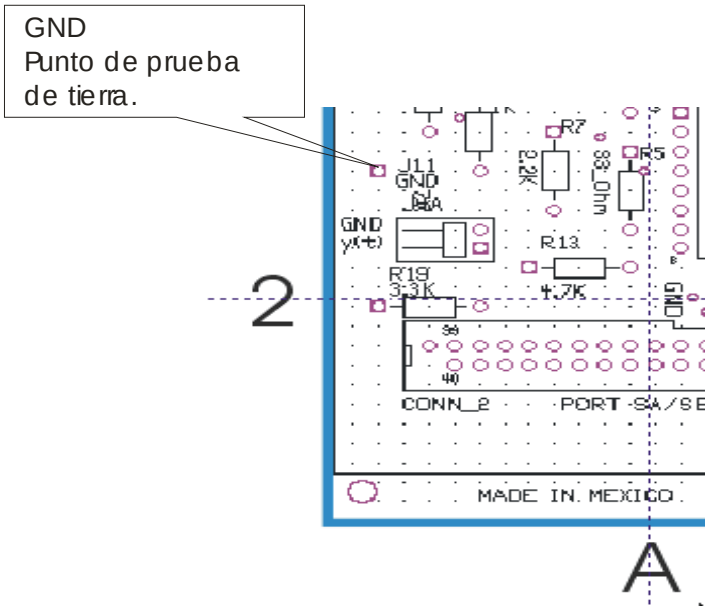


Figura 12. Punto de prueba J11 de la tarjeta SDSLI_B

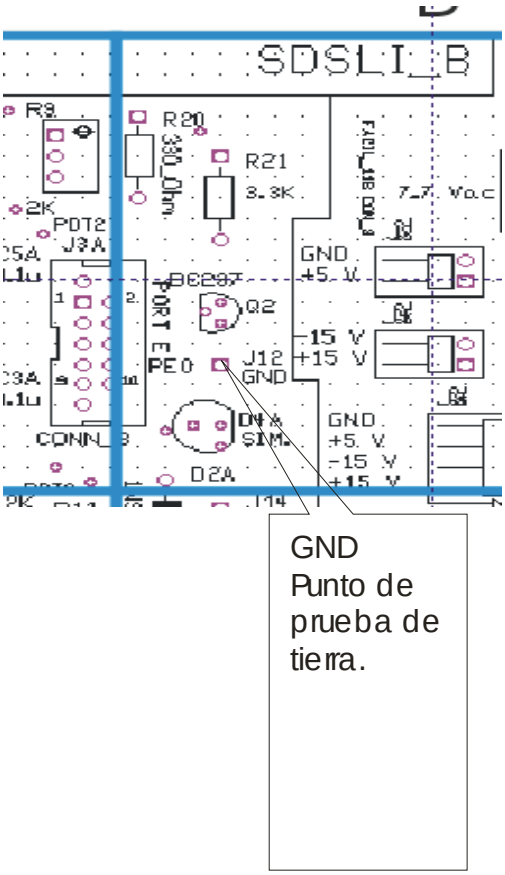


Figura 13. Punto de prueba J12 de la tarjeta SDSLI_B

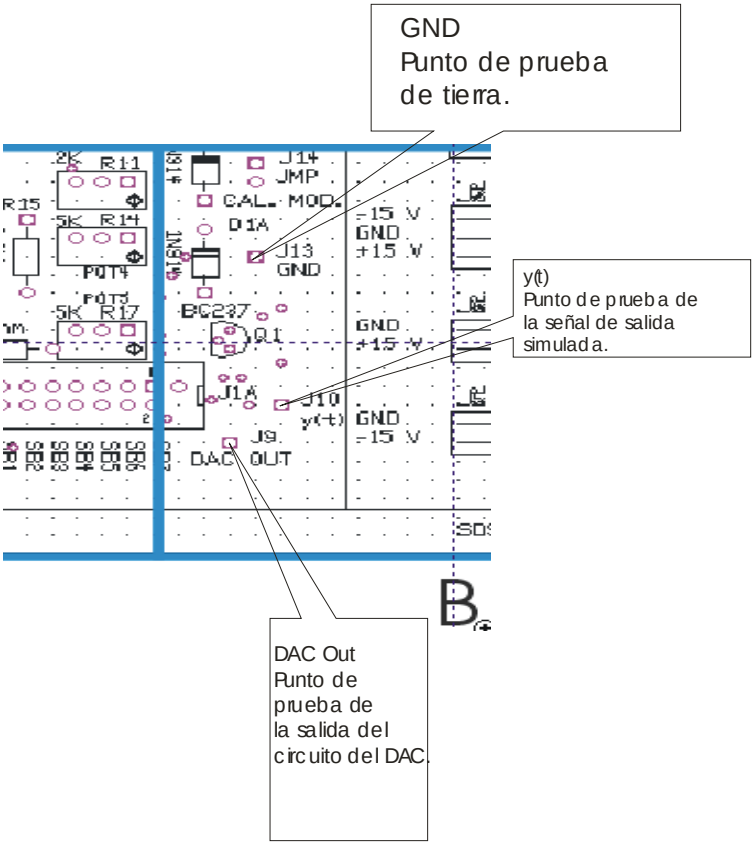


Figura 14. Puntos de prueba J9, J10, J13 de la tarjeta SDSLI_B

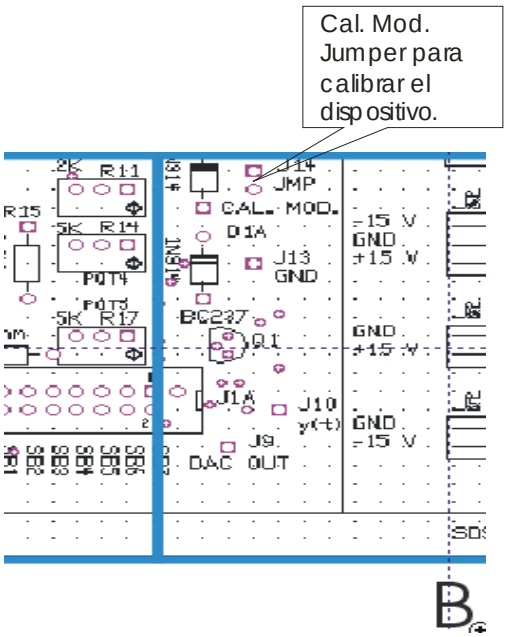


Figura 15. Puente (jumper) J14 de la tarjeta SDSLI_B, para poner el dispositivo en modalidad de calibración

4. CABLES DE COMUNICACIÓN ENTRE EL HARDWARE DEL SDSLI Y LA PC

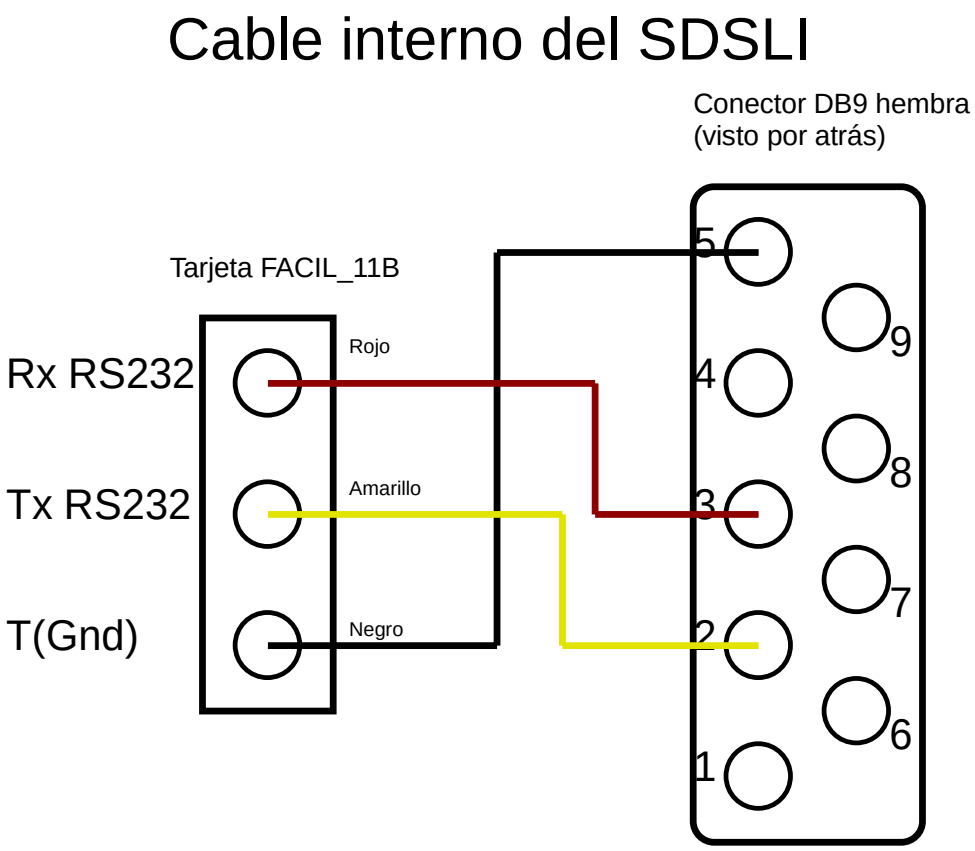


Figura 16. Cable de interno de comunicación del hardware del SDSLI

Cable de comunicación entre el SDSLI y la PC

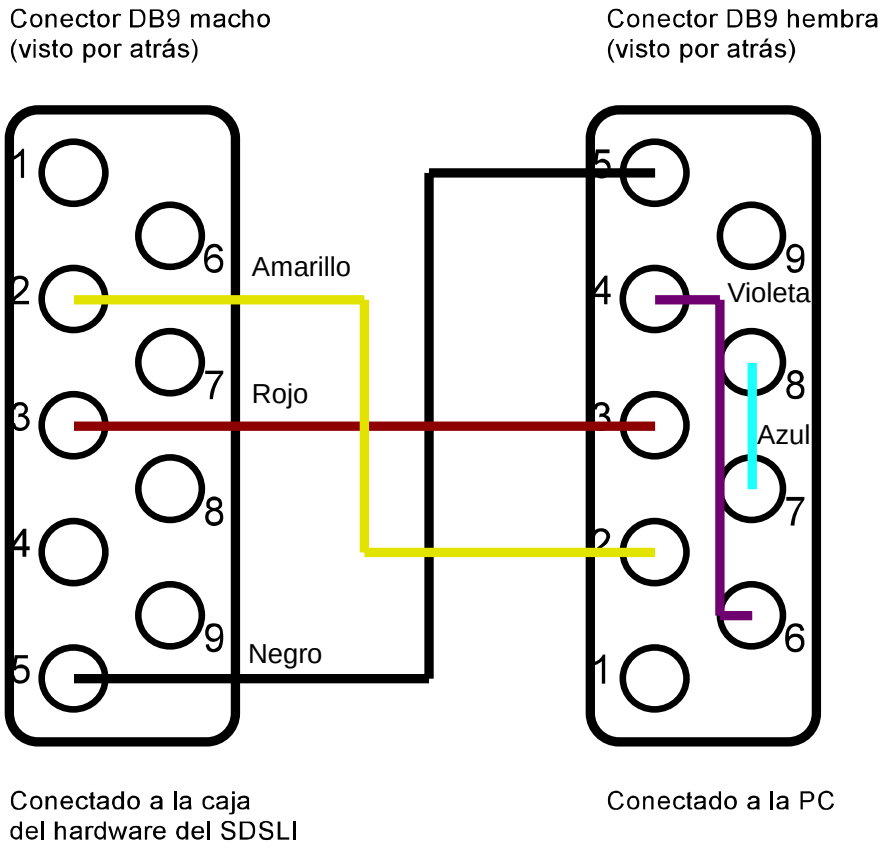


Figura 17. Cable de comunicación entre el SDSLI y la PC

5. MAPAS DE LOCALIZACIÓN DE LOS COMPONENTES ELECTRÓNICOS DE LA TARJETA SDSLI_B

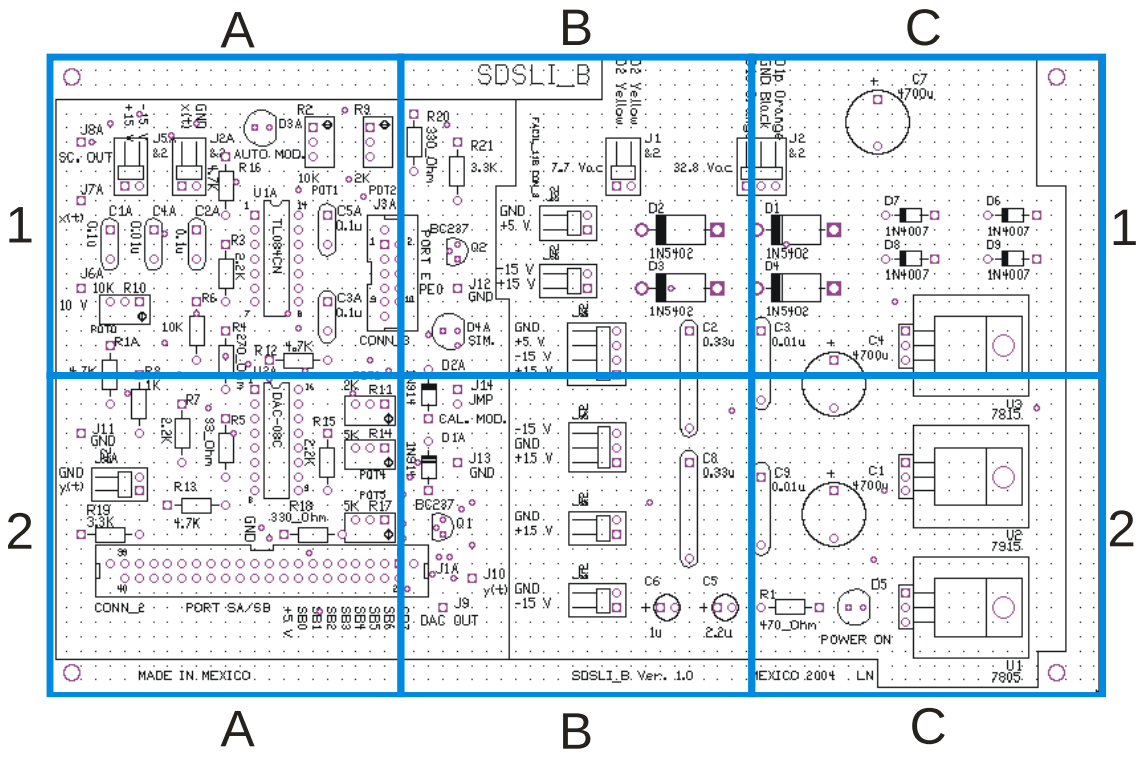


Figura 18. Mapa global de localización de los componentes electrónicos de la tarjeta SDSLI_B

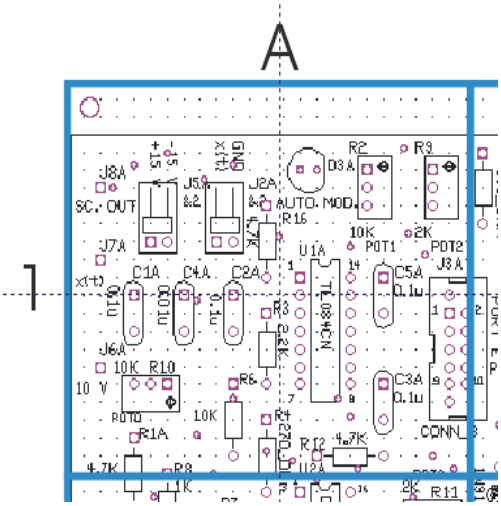


Figura 19. Submapa 1 de localización de los componentes electrónicos de la tarjeta SDSLI_B

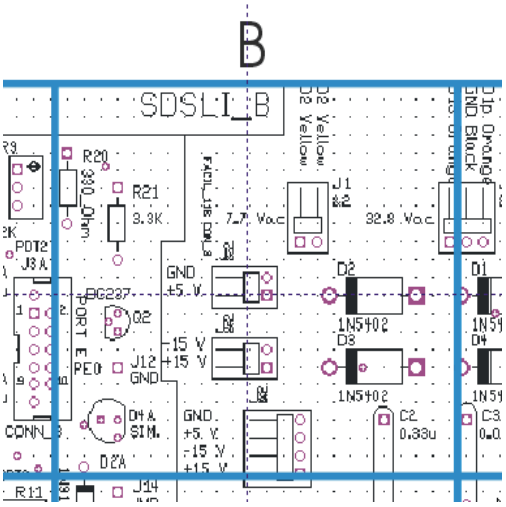


Figura 20. Submapa 2 de localización de los componentes electrónicos de la tarjeta SDSLI_B

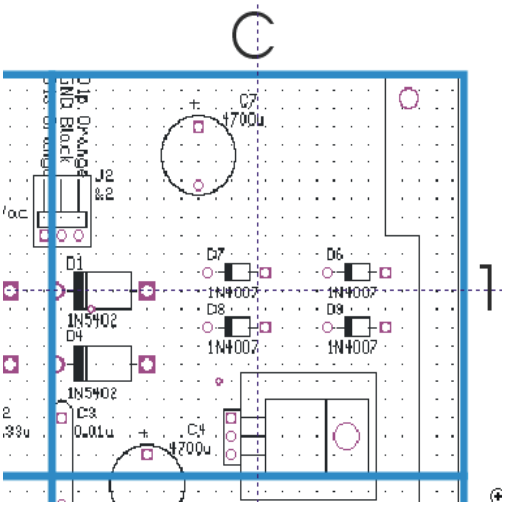


Figura 21. Submapa 3 de localización de los componentes electrónicos de la tarjeta SDSLI_B

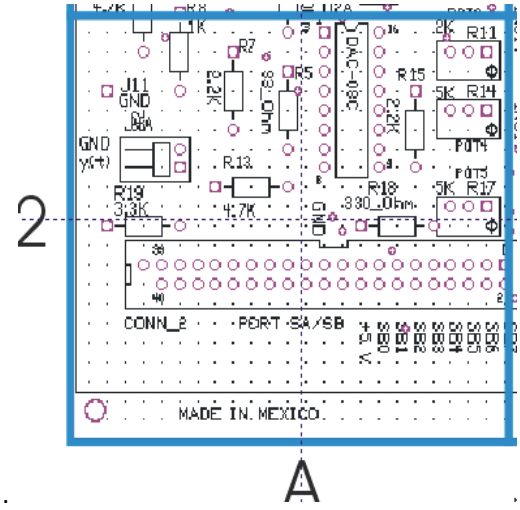


Figura 22. Submapa 4 de localización de los componentes electrónicos de la tarjeta SDSLI_B

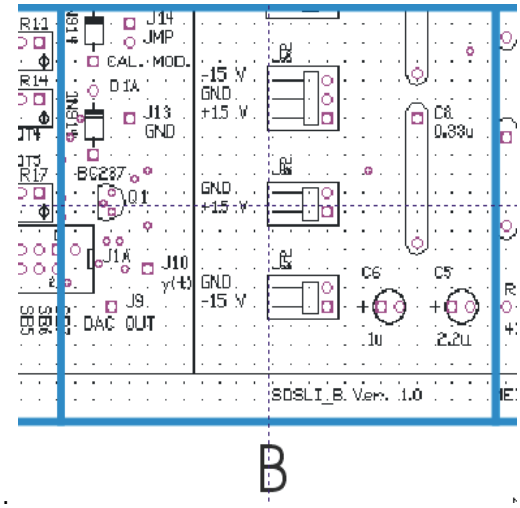


Figura 23. Submapa 5 de localización de los componentes electrónicos de la tarjeta SDSLI_B

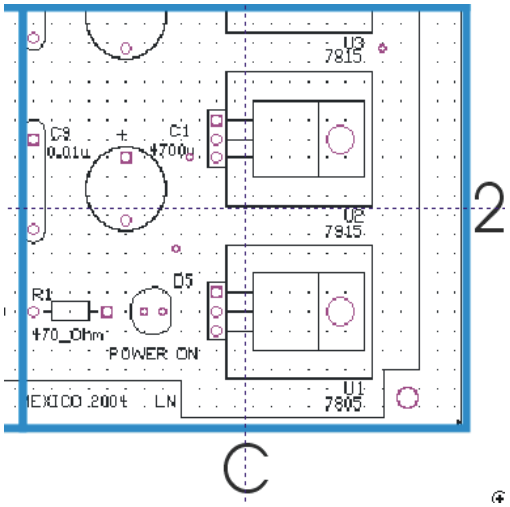


Figura 24. Submapa 6 de localización de los componentes electrónicos de la tarjeta SDSLI_B

Número de C.I. (I.C. Number)	Referencia de diseño (Design reference)	Localización (Map location)	Submapa de localización (Submap)
TL084CN	U1A	A1	1
DAC-08CP o bien DAC-0800LCN	U2A	A2	4

Tabla 1. Localización de los circuitos integrados en la tarjeta SDSLI_B

Capacitor (Capacitor)	Valor (Value)	Localización (Map location)	Submapa de localización (Submap)
C1	4700(µF) @ 25(V)	C2	6
C4	4700(µF) @ 25(V)	C1	3
C7	4700(µF) @ 25(V)	C1	3
C1A	0.1(µF) [100(nF)]	A1	1
C2A	0.1(µF) [100(nF)]	A1	1
C3A	0.1(µF) [100(nF)]	A1	1
C5A	0.1(µF) [100(nF)]	A1	1
C2	0.33(µF)	B1	2
C8	0.33(µF)	B2	5
C3	0.01(µF)	C1	3
C4A	0.01(µF) [10(nF)]	A1	1
C9	0.01(µF)	C2	6
C5	2.2(µF) @ 63(V)	B2	5
C6	1(µF) @ 50(V)	B2	5

Tabla 2. Localización de los capacitores en la tarjeta SDSLI_B

Semiconductor (Semiconductor)	Referencia de diseño (Design reference)	Valor (Value)	Localización (Map location)	Submapa de localización (Submap)
Diodo 1N914 (Diode)	D1A	1N914	B2	5
Diodo 1N914 (Diode)	D2A	1N914	B2	5
Diodo 1N5402 (Diode)	D1	1N5402	C1	3
Diodo 1N5402 (Diode)	D2	1N5402	B1	2
Diodo 1N5402 (Diode)	D3	1N5402	B1	2
Diodo 1N5402 (Diode)	D4	1N5402	C1	3
LED (testigo Auto Mod., verde) [green]	D3A	LED 10(mA)	A1	1
LED (testigo Sim., azul) [blue]	D4A	LED 10(mA)	B1	2
LED (testigo Power ON, rojo) [red]	D5	LED 10(mA)	C2	6
Diodo 1N4007 (Diode)	D6	1N4007	C1	3
Diodo 1N4007 (Diode)	D7	1N4007	C1	3
Diodo 1N4007 (Diode)	D8	1N4007	C1	3
Diodo 1N4007 (Diode)	D9	1N4007	C1	3
Transistor BC237 o transistor BC547 (Transistor)	Q1	BC547	B2	5
Transistor BC237 o transistor BC547 (Transistor)	Q2	BC547	B1	2
Transistor regulador LM7805/TO (Voltaje regulator)	U1	LM7805	C2	6
Transistor regulador LM7915C/TO3 (Voltage regulator)	U2	LM7915	C2	6
Transistor regulador LM7815/TO (Voltage regulator)	U3	LM7815	C1	3

Tabla 3. Localización de los semiconductores discretos en la tarjeta SDSLI_B

Conector (Connector)	Referencia de diseño (Design reference)	Valor (Value)	Localización (Map location)	Submapa de localización (Submap)
(1x2) Molex o equivalente (CONN. PLUG 1x2)	J1	Molex 1x2	B1	2
(1x2) Molex o equivalente (CONN. PLUG 1x2)	J2A	Molex 1x2	A1	1
(1x2) Molex o equivalente (CONN. PLUG 1x2)	J3	Molex 1x2	B1	2
(1x2) Molex o equivalente (CONN. PLUG 1x2)	J4A	Molex 1x2	A2	4
(1x2) Molex o equivalente (CONN. PLUG 1x2)	J4	Molex 1x2	B1	2
(1x2) Molex o equivalente (CONN. PLUG 1x2)	J5A	Molex 1x2	A1	1
(1x2) Molex o equivalente (CONN. PLUG 1x2)	J5	Molex 1x2	B2	5
(1x2) Molex o equivalente (CONN. PLUG 1x2)	J6	Molex 1x2	B2	5
(1x2) Postes sencillos (Test point)	J14	Jumper 1x2	B2	5
(2x20) macho x o equivalente (CONN. PLUG 2x20)	J1A	Box header. Conector macho 2x20	A2	4
(1x3) Molex o equivalente (CONN. PLUG 1x3)	J7	Molex 1x3	B2	5
(1x3) Molex o equivalente (CONN. PLUG 1x3)	J2	Molex 1x3	C1	3
(2x5) macho o equivalente (CONN. PLUG 2x5)	J3A	Box header. Conector macho 2x5	A1	1
(1x1) Poste sencillo (Test point)	J6A	Poste para punto de prueba	A1	1
(1x1) Poste sencillo (Test point)	J7A	Poste para punto de prueba	A1	1
(1x1) Poste sencillo (Test point)	J8A	Poste para punto de prueba	A1	1
(1x1) Poste sencillo (Test point)	J9	Poste para punto de prueba	B2	5
(1x1) Poste sencillo (Test point)	J10	Poste para punto de prueba	B2	5
(1x1) Poste sencillo (Test point)	J11	Poste para punto de prueba	A2	4
(1x1) Poste sencillo (Test point)	J12	Poste para punto de prueba	B1	2
(1x1) Poste sencillo (Test point)	J13	Poste para punto de prueba	B2	5
(1x4) Molex o equivalente (CONN. PLUG 1x4)	J8	Molex 1x4	B1	2

Tabla 4. Localización de los conectores en la tarjeta SDSLI_B

Resistencia (Resistor)	Valor (Value)	Localización (Map location)	Submapa de localización (Submap)
R1	470(Ω) @ 1/4(W) Tol. 5(%)	C2	6
R1A	4.7(KΩ) @ 1/4(W) Tol. 5(%)	A1	1
R12	4.7(KΩ) @ 1/4(W) Tol. 5(%)	A1	1
R13	4.7(KΩ) @ 1/4(W) Tol. 5(%)	A2	4
R16	4.7(KΩ) @ 1/4(W) Tol. 5(%)	A1	1
R10	TRIMPOT Vertical de 10(KΩ)	A1	1
R2	TRIMPOT Vertical de 10(KΩ)	A1	1
R3	2.2(KΩ) @ 1/4(W) Tol. 5(%)	A1	1
R7	2.2(KΩ) @ 1/4(W) Tol. 5(%)	A2	4
R15	2.2(KΩ) @ 1/4(W) Tol. 5(%)	A2	4
R4	270(Ω) @ 1/4(W) Tol. 5(%)	A1	1
R5	33(Ω) @ 1/4(W) Tol. 5(%)	A2	4
R6	10(KΩ) @ 1/4(W) Tol. 5(%)	A1	1
R8	1(KΩ) @ 1/4(W) Tol. 5(%)	A2	4
R9	TRIMPOT Vertical de 2(KΩ)	A1	1
R11	TRIMPOT Vertical de 2(KΩ)	A2	4
R14	TRIMPOT Vertical de 5(KΩ)	A2	4
R17	TRIMPOT Vertical de 5(KΩ)	A2	4
R18	330(Ω) @ 1/4(W) Tol. 5(%)	A2	4
R20	330(Ω) @ 1/4(W) Tol. 5(%)	B1	2
R19	3.3(KΩ) @ 1/4(W) Tol. 5(%)	A2	4
R21	3.3(KΩ) @ 1/4(W) Tol. 5(%)	B1	2

Tabla 5. Localización de las resistencias en la tarjeta SDSLI_B

Tarjeta SDSLI_B

Cantidad (Quantity)	Especificaciones del componente (Part specifications)	Referencia de diseño (Design reference)	Precio Unitario (Pesos mexicanos, oct. 2004)	Precio unitario en dólares	Subtotal en dólares	Proveedor
3	Capacitores electrolíticos 4700(µF) @ 25(V).	C1, C4, C7	10.435	0.91695958	2.750878737	Steren
4	Capacitores de cerámica de disco 0.1(µF)[100(nF)].	C1A, C2A, C3A,C5A	1.739	0.15281195	0.6112478	Steren
2	Capacitores de poliester metalizado 0.33(µF).	C2, C8	4	0.35149385	0.7029877	Steren
2	Capacitores de poliester metalizado 0.01(µF) [10(nF)].	C3, C9	4	0.35149385	0.7029877	Steren
1	Capacitor de cerámica de disco 0.01(µF) [10(nF)].	C4A	0.87	0.07644991	0.07644991	Steren
1	Capacitor electrolítico 2.2(µF) @ 63(V).	C5	1.739	0.15281195	0.15281195	Steren
1	Capacitor electrolítico 1(µF) @ 50(V).	C6	1.739	0.15281195	0.15281195	Steren
2	Diodos 1N914 (o bien 1N4148).	D1A, D2A	0.87	0.07644991	0.15289982	Steren
4	Diodos 1N5402 (Diodos rectificadores 3(A) 200 (V)).	D1, D2, D3, D4	1.739	0.15281195	0.6112478	Steren
1	Diodo LED 5(mm.) económico verde claro.	D3A	1.739	0.15281195	0.15281195	Steren
1	Diodo LED 5(mm.) económico azul claro.	D4A	1.739	0.15281195	0.15281195	Steren
1	Diodo LED 5(mm.) económico rojo claro.	D5	1.739	0.15281195	0.15281195	Steren
4	Porta Led.		3	0.26362039	1.05448155	Ryzenman's
4	Diodos 1N4007 (Diodos rectificadores 1(A) 1000 (V)).	D6, D7, D8, D9	0.87	0.07644991	0.30579965	Steren
2	Transistores BC547-B	Q1, Q2	1.74	0.15289982	0.30579965	Steren
8	Conectores Molex (montaje vertical, recto) de dos pines (dos vías) (postes, caja y pines) Machos y hembras correspondientes de paso de 100.	J1, J2A, J3, J4A, J4, J5A, J5, J6	1.15	0.10105448	0.80843585	Ele y Ele
2	Tiras de postes sencillos de paso de 100.	J14, J6A, J7A, J8A, J9, J10, J11, J12, J13	3.48	0.30579965	0.6115993	Steren
1	Jumper (paso de 100 milésimas de pulgada).	J14	0.87	0.07644991	0.07644991	AG Electrónica
2	Conectores Molex (montaje vertical, recto) de tres pines (3 vías) (postes, caja y pines). Machos y hembras correspondientes de paso de 100.	J2, J7	1.85	0.16256591	0.32513181	Ele y Ele y MAK Electronics.
1	Conector Molex (montaje vertical, recto) de cuatro pines (4 vías) (postes, caja y pines). Machos y hembras correspondientes de paso de 100.	J8	2.58	0.22671353	0.22671353	Ele y Ele
26	Pines para conector molex paso de 100.		0.5	0.04393673	1.14235501	Ele y Ele
5	Tramos de cable forrado calibre 22 de 3(m) de longitud de color naranja, amarillo, negro, rojo y café.		3	0.26362039	3.9543058	Ele y Ele
1	Conector standard PCB BOX HEADER polarizado de 20x2 macho o en su defecto una tira de postes dobles (40 pines). (Header recto caja de 40 PDS).	J1A	10	0.87873462	0.87873462	MAK Electronics.
2	Conectores quickies (para cable plano) de 20x2 hembra. (FID de 40 pines cable plano)	J1A	3.64	0.3198594	0.6397188	MAK Electronics.
1	Tira de cable plano de 40 hilos, 30(cm.) de longitud.	J1A	17.39	1.52811951	1.52811951	Steren
1	Conector standard PCB BOX HEADER polarizado de 5x2 macho o en su defecto una tira de postes dobles (10 pines) (Header vertical H1BS-10).	J3A	4.43	0.38927944	0.38927944	DECSA
2	Conectores quickies (para cable plano) de 5x2 hembra. (FID de 10 pines cable plano)	J3A	2	0.17574692	0.35149385	MAK Electronics.
1	Tira de cable plano de 10 hilos, 30(cm.) de longitud.	J3A	4.783	0.42029877	0.42029877	AG Electrónica
5	4.7(KΩ) @ 1/4(W) Tol. 5(%)	R1A, R12, R13, R16	0.435	0.03822496	0.19112478	Steren
5	470(Ω) @ 1/4(W) Tol. 5(%)	R1	0.435	0.03822496	0.19112478	Steren
2	TRIMPOT Vertical de 10(KΩ)	R10, R2	24.348	2.13954306	4.27908612	Steren
5	2.2(KΩ) @ 1/4(W) Tol. 5(%)	R3, R7, R15	0.435	0.03822496	0.19112478	Steren
5	270(Ω) @ 1/4(W) Tol. 5(%)	R4	0.435	0.03822496	0.19112478	Steren
5	33(Ω) @ 1/4(W) Tol. 5(%)	R5	0.435	0.03822496	0.19112478	Steren
5	10(KΩ) @ 1/4(W) Tol. 5(%)	R6	0.435	0.03822496	0.19112478	Steren
5	1(KΩ) @ 1/4(W) Tol. 5(%)	R8	0.435	0.03822496	0.19112478	Steren
2	TRIMPOT Vertical de 2(KΩ)	R9, R11	16	1.4059754	2.81195079	Ele y Ele
2	TRIMPOT Vertical de 5(KΩ)	R14, R17	24.348	2.13954306	4.27908612	Steren
5	330(Ω) @ 1/4(W) Tol. 5(%)	R18, R20	0.435	0.03822496	0.19112478	Steren
5	3.3(KΩ) @ 1/4(W) Tol. 5(%)	R19, R21	0.435	0.03822496	0.19112478	Steren
1	Circuito integrado, amplificador operacional TL084CN de STMicroelectronics.	U1A	7.826	0.68769772	0.68769772	Steren
1	Base de 14 pines.	U1A	1	0.08787346	0.08787346	Ele y Ele
1	Convertidor digital/analógico 8 bits DAC-08CP de Motorola o bien DAC-0800LCN de National Semiconductors.	U2A	30	2.63620387	2.63620387	AEESA
1	Base de 16 pines.	U2A	1	0.08787346	0.08787346	Ele y Ele
1	Regulador de voltaje de 5(V) LM7805/TO (Circuito Integrado lineal MC7805CT)	U1	4.348	0.38207381	0.38207381	Steren
1	Regulador de voltaje de -15(V) LM7915C/TO3 (Circuito Integrado lineal MC7915CT)	U2	5.217	0.45843585	0.45843585	Steren
1	Regulador de voltaje de 15(V) LM7815/TO (Circuito Integrado lineal MC7815CT)	U3	4.348	0.38207381	0.38207381	Steren
3	Disipadores para los reguladores: 7805, 7915 y 7815 (Disipador Elect. Peine de 10 A modelo TO-218 o bien de preferencia el TO-220 o TO-200).	U1, U2, U3	6.087	0.53488576	1.60465729	Steren
3	Juegos de tuerca y tornillo para asegurar los disipadores a los reguladores de voltaje. Tornillo cabeza de gota 1/8" de diámetro x 3/8" de longitud. Tuerca hexagonal Sakamura 1/8" de diámetro.	U1, U2, U3	0.1	0.00878735	0.02636204	Servicoapa
3	Juegos de micas con rondana de plástico NEGRAS Y PEQUEÑAS para los reguladores: 7805, 7915 y 7815. (empaque TO-220)	U1, U2, U3	3.479	0.30571178	0.91713533	Steren
1	Grasa de silicón para unir los reguladores y los disipadores.		26.96	2.36906854	2.36906854	Steren
1	Rollo de 200(Grs.) de soldadura.		51.3	4.50790861	4.50790861	Steren
1	Circuito impreso SDSLI_B.		250	21.9683656	21.9683656	M. I. Antonio Salvá Calleja
1	Tarjeta FACIL_11B armada y configurada con firmware del SDSLI.		750	65.9050967	65.9050967	M. I. Antonio Salvá Calleja
1	AU-105 Switch push button redondo. Botón normalmente abierto para el RESET de la FACIL_11B.		9.57	0.84094903	0.84094903	Steren
1	B2600 Switch Palanca 2 polos, 1 tiro, 2 posiciones. Switch para el encendido de la fuente.		9.57	0.84094903	0.84094903	Steren
2	S-119 Switch palanca mini, 2 polos, 2 tiros, 2 posiciones.		6.09	0.53514938	1.07029877	Steren
1	GMA-5 Fusible tipo europeo 5(A).		3.478	0.3056239	0.3056239	Steren
1	AMPF-A Porta fusible de cartucho		4.348	0.38207381	0.38207381	Steren
1	Transformador con clavija. El transformador debe ser de 125(Vac) a 32.8(Vac) [cables naranja] con tap central [cable negro] y de 125(Vac) a 7.7(Vac) [cables amarillos].		0	0	0	
1	Cable para conexión eléctrica tipo computadora. Interlock para computadora.		19.13	1.68101933	1.68101933	Steren
1	Clavija macho para conectar cable para conexión eléctrica tipo computadora. Toma corriente macho con oreja.		13	1.14235501	1.14235501	Auroelectroni'k
2	Conector DB9 hembra.		4.348	0.38207381	0.76414763	Steren
1	Conector DB9 macho.		3.478	0.3056239	0.3056239	Steren
2	Cubierta para conector tipo DB9. Conchas para DB9.		3.48	0.30579965	0.6115993	Steren
1	2 metros M-10x22 Cable multimalla maylar.		50.434	4.43181019	4.43181019	Steren
4	Conector PLUG Banana o Philips negro.		3.48	0.30579965	1.22319859	Steren
4	Conector PLUG Banana o Philips rojo.		3.48	0.30579965	1.22319859	Steren
4	Borne para banana negro 10(A).		1.8	0.15817223	0.63268893	Steren
4	Borne para banana rojo 10(A).		1.8	0.15817223	0.63268893	Steren
1	Caja para montar tarjeta FACIL_11B y tarjeta SDSLI_B junto con el transformador.		800	70.2987698	70.2987698	VESAM
				Subtotal 1	220.889543	

Para el armado del transformador

Cantidad (Quantity)	Especificaciones del componente (Part specifications)	Referencia de diseño (Design reference)	Precio Unitario (Pesos mexicanos, dic. 2003)	Precio unitario en dólares	Subtotal en dólares	Proveedor
1	Núcleos L-112		26.955	2.36862917	2.36862917	Bolívar Ochenta Electrónica
	Laminación "E" de 1+1/8 de pulgada.					
	Laminación "I" correspondiente a la laminación "E".					
1	Carrete de 1+1/8 de pulgada x 1+1/8 de pulgada. (Carrete 112)		3.044	0.26748682	0.26748682	Bolívar Ochenta Electrónica
4	Láminas "L" para montaje horizontal. (Escuadras 112) .		0.60875	0.05349297	0.21397188	Bolívar Ochenta Electrónica
1	Pliego de papel pescado (no muy delgado). 25 (cm.) de papel pescado.		4.78	0.42003515	0.42003515	Bolívar Ochenta Electrónica
1	Pliego de papel pescado grueso (para la última capa del transformador).		4.78	0.42003515	0.42003515	Bolívar Ochenta Electrónica
1	(1/4(K)) Alambre AWG 28. (250 (grs.) alambre magneto de 28)		22.76	2	2	Bolívar Ochenta Electrónica
1	(1/4(K)) Alambre AWG 20. (250 (grs.) alambre magneto de 20)		19.17	1.68453427	1.68453427	Bolívar Ochenta Electrónica
4	Tornillos con tuerca para fijar las láminas "L" al transformador. Tornillo cabeza de gota 3/16" de diámetro x 1 1/2" de longitud. Tuerca hexagonal Sakamura 3/16" de diámetro.		0.22	0.01933216	0.07732865	Servicoapa
				Subtotal 2	5.08339192	

Total en pesos	2571.572

Precio del dólar (oct. 2004)
11.38

Proveedor	Dirección
Steren	Villa Coapa Electrónica, S. A de C. V. Canal de Miramontes 2681. Col. Prados de Coyoacán. C.P. 04810 México D. F.
AG Electrónica	República del Salvador No. 20-F. Colonia Centro. México D. F. C. P. 06000. TEL. y Fax. 5130-7210.
Ele y Ele	Electricidad y Electrónica. Aldaco No. 6 Loc. 7 Col. Centro. C.P. 06000. México D. F. TEL. 5709-5240.
AEESA	AEE S. A. de C. V. Se trata del local que vende componentes electrónicos en
MAK Electronics	MAK Electronics, S. A de C. V. República del Salvador 26-A C.P. 06000 México D. F.
DECSA	Dispositivos Electrónicos y de Control S. A. de C. V. República del Salvador No. 20-D 3er. Piso C. P. 06000 Col. Centro. México D. F.
Bolívar Ochenta Electrónica	Bolívar Ochenta Electrónica. Bolívar 80-B Col. Centro C. P. 06080. Delegación Cuauhtémoc,. México D. F. TEL.:5709-7672
Ryzenman's	Pasaje Salvador Aldaco No. 6 local 17. Colonia Centro. México D. F., Delegación Cuauhtémoc. TEL. y Fax. 5709-2162
Auroelectroni'k	República del Salvador No. 24 local 14 y 16. Colonia Centro. México D. F. C. P. 06000. TEL. y Fax. 5512-5231.
M. I. Antonio Salvá Calleja	División de Ingeniería Eléctrica. Facultad de Ingeniería, campus C. U. UNAM. México D. F. C. P. 04510. Delegación Coyoacán. TEL. 5622-3108. Email: salva@dctrl.fi-b.unam.mx.
VESAM	D. I. Asaf Venegas Sam. TEL. (0155)5601-5211 y (01777)3177-672.
Servicoapa	Servicoapa S. A de C. V. Cafetales No. 230. Col. Rinconada Coapa. Delegación Tlalpan. C. P. 14330 México D. F. Tels. 5673-7775 y 5603-4526.

Tabla 6. Lista de compras para la tarjeta SDSLI_B

6. EL TRANSFORMADOR DEL SDSLI

6.1. MEDIDAS

A. Tamaño del carrete: $1+\frac{1}{8}(pu\lg ada) \times 1+\frac{1}{8}(pu\lg ada)$.

B. Laminación “E” e “I” de $1+\frac{1}{8}(pu\lg ada)$.

6.2. CONSTRUCCIÓN DEL TRANSFORMADOR

1. Devanado primario para $127(Vac)$, formado por 657 vueltas con alambre AWG 28.
2. Dos capas de papel pescado.
3. Devanado secundario con tap central para $\pm 16.4(Vac)$ o en otras palabras, devanado secundario para $32.8(Vac)$ con tap central.
    a. Devanar el alambre de modo bifiliar respetando los puntos de polaridad para obtener el tap central. Dar 87 vueltas de manera bifiliar con alambre AWG 28.
4. Dos capas de papel pescado.
5. Devanado secundario de $7.7(Vac)$, constituido por 41 vueltas con alambre AWG 20.
6. Dos capas de papel pescado.
7. Rematar cables y terminales.
    a. Para las terminales de $127(Vac)$ utilizar cable calibre 20 color café.
    b. Para las terminales de $7.7(Vac)$ utilizar cable calibre 20 color amarillo.
    c. Para las terminales de $32.8(Vac)$, utilizar cable calibre 20 color naranja.
    d. Para el tap central (GND) utilizar cable calibre 20 color negro.
8. Dos o cuatro vueltas de papel pescado para cerrar el transformador.

9. Aplicar barniz transparente para transformador, utilizando la técnica de inmersión durante 1 hora y dejar secar 24 horas.

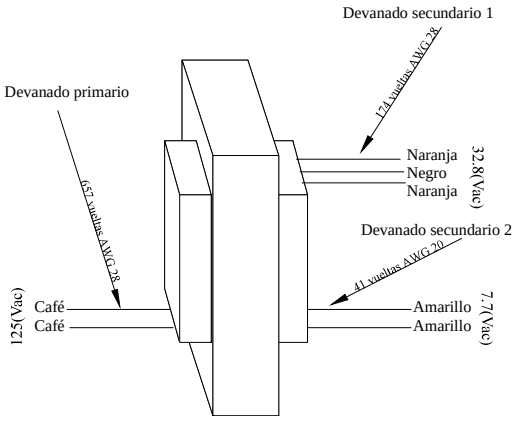


Figura 25. Posición de las terminales del transformador del SDSLI

10. Añadir cables calibre 20 para las terminales, de los colores y las posiciones especificadas en el diagrama.
11. Colocar conector molex hembra paso 100 de dos entradas en el extremo de los cables color amarillo.
12. Colocar conector molex hembra paso 100 de tres entradas en el extremo de los cables naranja y negro, haciendo que el cable negro ocupe la posición central.

7. ESPECIFICACIONES Y PLANOS DE LA CAJA METÁLICA

Medidas:

- Alto: $13.5(cm)$.
- Ancho: $23(cm)$.
- Profundo: $30(cm)$.

Materiales:

1. Lámina negra calibre 20.
2. Acrílico de $3(mm)$ de espesor.

Acabados:

- Poliuretano alifático: Acabado catalizador de poliuretano, para aplicación directa a metal (DMT), formulado con pigmentos inhibidores de corrosión libres de plomo y otros metales pesados.

Modo de aplicación de acabados:

- Se utilizó pistola de compresión y un horno.

Procesos:

- Cortes: Se utilizó una cortadora de lámina con sistema de cizalla.
- Dobleces: Se realizaron con una dobladora de cortina.

- Soldadura: Se utilizó una punteadora de metal.
- Perforaciones: Se hicieron con taladros y brocas de tamaño estándar. En algunos casos, se utilizó taladro fijo.

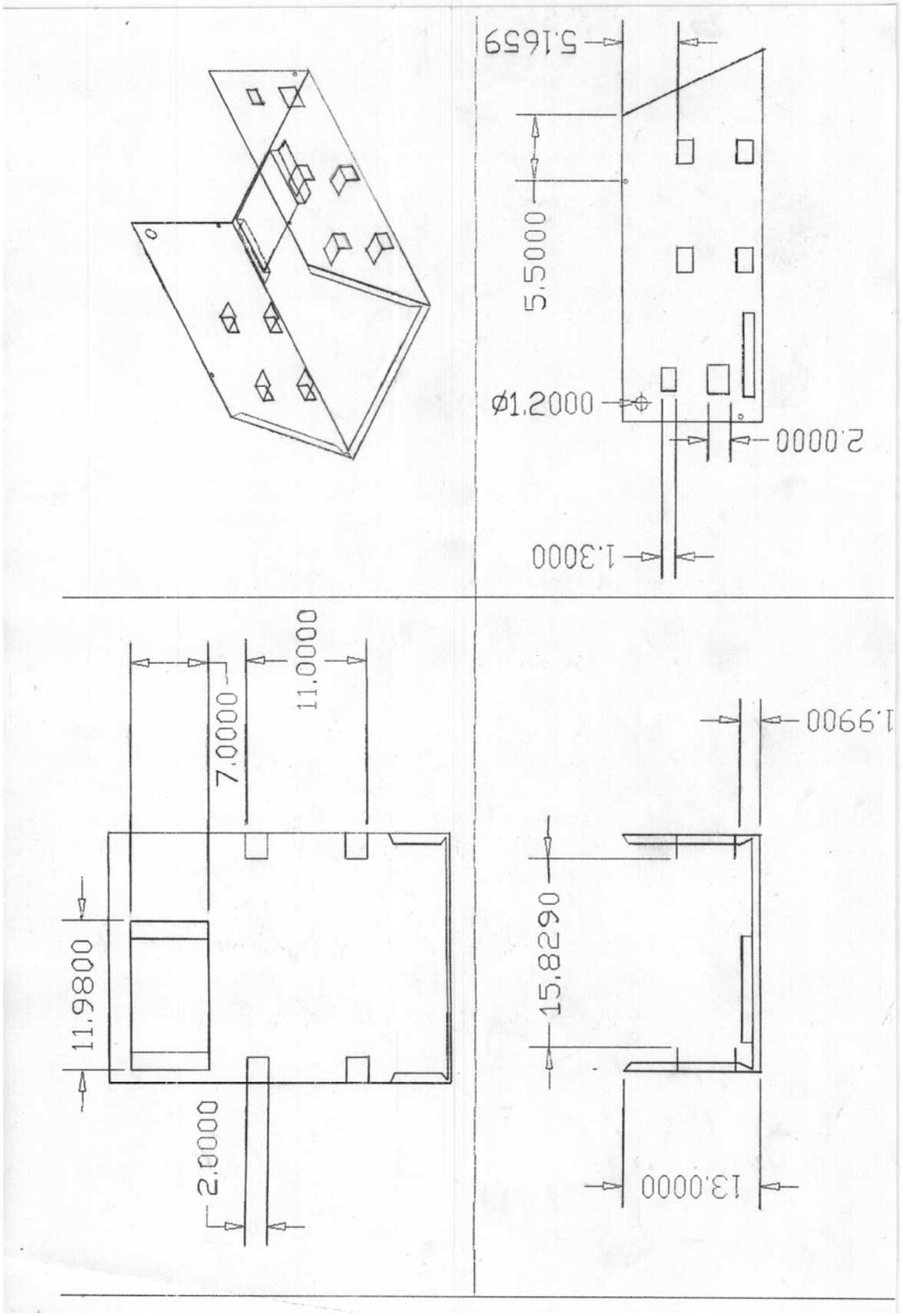


Figura 26. Plano 1 de la caja metálica del SDSLI. Cotas en (cm.)

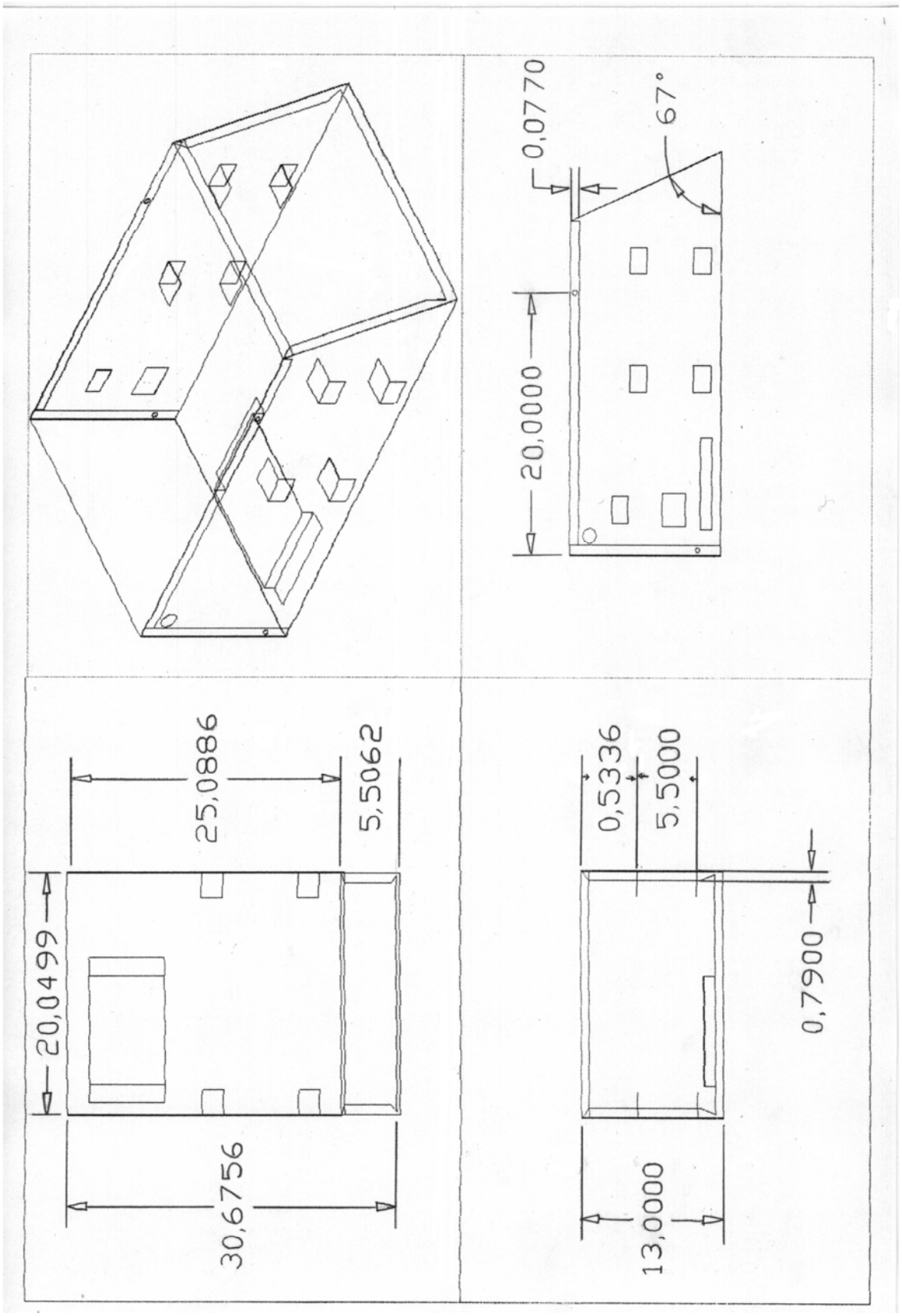


Figura 27. Plano 2 de la caja metálica del SDSLI. Cotas en (cm.) y ángulos en grados sexagesimales

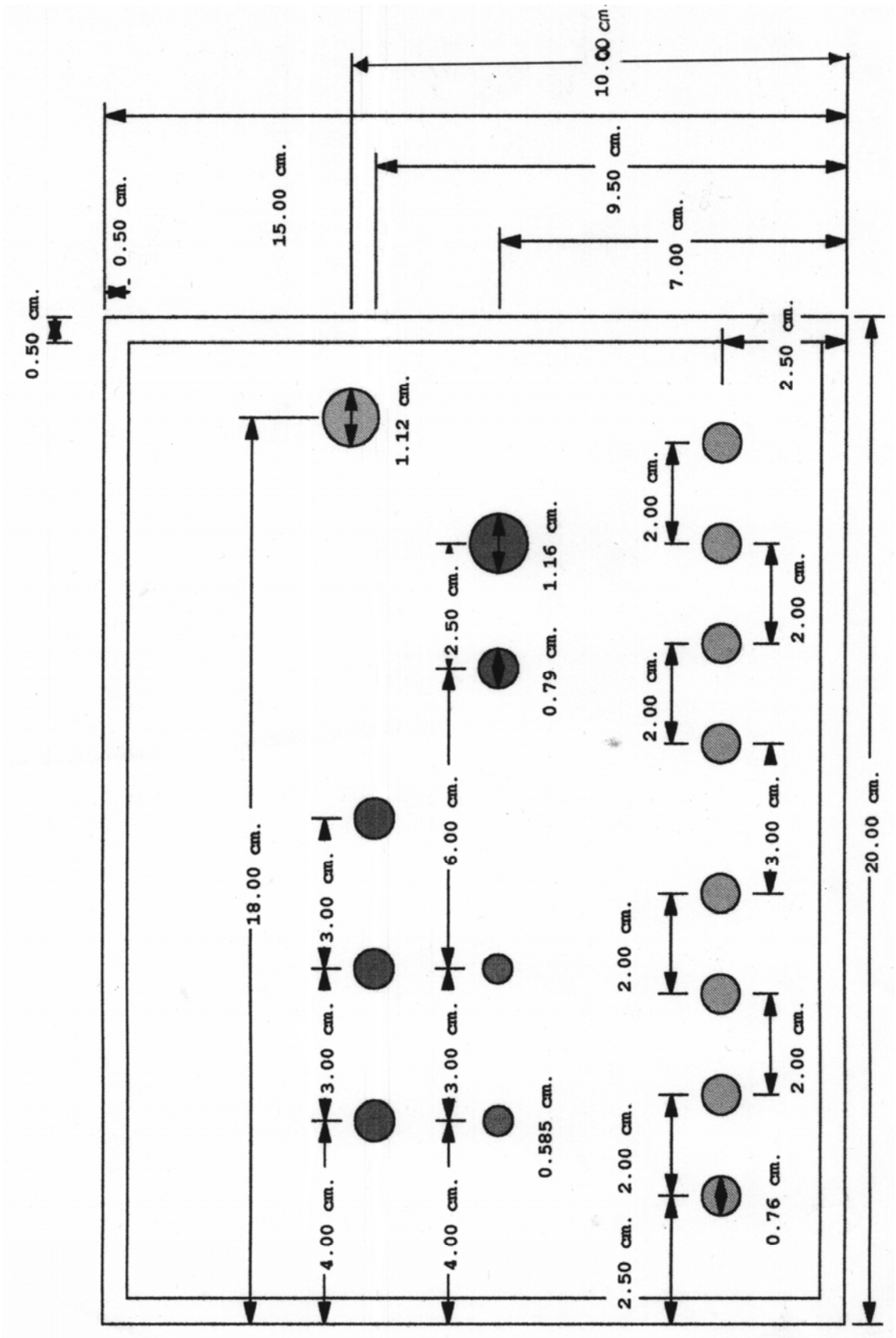


Figura 28. Plano de la carátula de la caja metálica del SDSLI. Cotas en (cm.)

8. CALIBRACIÓN DEL SDSLI

Para garantizar que el funcionamiento del simulador sea óptimo y los valores tanto de las señales de entrada como de las señales de salida sean correctos, se requiere calibrar la circuitería que hace funcionar al simulador.

A continuación, se presentan los pasos a seguir para calibrar de modo correcto el hardware del simulador. Para facilitar esta tarea, el software cuenta con la opción **"Calibrar SDSLI"** en el menú **"Herramientas"** de la pantalla principal.

1. Conectar el hardware del simulador a la PC a través del cable de comunicación serial y además, conectarlo a la corriente eléctrica con su cable correspondiente.
2. Encender el simulador.
3. Iniciar el software del simulador.
4. Una vez que el software esté listo para recibir ordenes del usuario y el led **"LISTO"** de la carátula del simulador se encuentre parpadeando, seleccione el menú **"Herramientas"** y elija la opción **"Calibrar SDSLI"**. Con esta acción, iniciará el asistente de calibración del hardware del SDSLI.
5. Leer con detenimiento cada una de las pantallas que el asistente le mostrará. Siga exactamente los pasos que le indica. Cabe mencionar, que la calibración del simulador es un procedimiento delicado que de no ser realizado cuidadosamente, puede dañar la circuitería de forma permanente.

No obstante que los pasos son indicados con precisión por el asistente de calibración, se presentan en seguida, con la finalidad de que el usuario tenga conocimiento de ellos, sin tener que iniciar el asistente de calibración del software.

Antes de iniciar la calibración, deberá tener a mano las siguientes herramientas:

1. Desarmador hexagonal de ¼ de pulgada.
2. Desarmador plano de relojero.

3. Multímetro.
4. Osciloscopio.
5. Un par de cables caimán – caimán.

Nota: Todos los ajustes y puntos de prueba se encuentran en la tarjeta SDSLI_B. Por ningún motivo, deberá realizar ajustes en la tarjeta FACIL_11B.

Paso 1

Apague el hardware del SDSLI moviendo el switch **"ENCENDIDO"** a la posición **"OFF"** y desconecte el cable de la toma de corriente eléctrica. También, desconecte el cable de comunicación con la PC.

Asegúrese de desconectar todos los cables en las terminales de **"SEÑAL DE ENTRADA"** y **"SEÑAL DE SALIDA"** del hardware del SDSLI.

Paso 2

Para tener acceso a la circuitería del hardware del SDSLI, quite los tornillos de la caja y destápela.

Paso 3

Mueva el switch **"MODO DE OPERACIÓN"** a la posición **"CLIENTE"**.

Paso 4

Mueva el switch a la posición **"SIMULACIÓN"**.

Paso 5

Verifique que no haya dejado ningún tornillo ni herramientas sueltas dentro de la caja destapada del hardware del SDSLI. Posteriormente, vuelva a conectar cuidadosamente el cable a la toma de corriente eléctrica y además, vuelva a conectar el cable de comunicación con la PC.

Paso 6

Encienda el hardware del SDSLI, moviendo el switch **“ENCENDIDO”** a la posición **“ON”**.

Paso 7

Debe observar que el led **“LISTO”** de la carátula, esté parpadeando. En caso de que el led **“LISTO”** no esté encendido y tampoco parpadee, revise que el hardware esté encendido, que el cable de comunicación esté debidamente conectado tanto al SDSLI, como a la PC y finalmente, asegúrese de que el cable toma corriente esté apropiadamente conectado y presione **“RESET”**.

Paso 8

Coloque el multímetro en la modalidad de medición de voltajes.

Coloque la punta común (Tierra) de este instrumento, en el pin J11 (GND). A continuación, coloque la punta de medición de voltaje, en el pin J6A $\left[10(V)\right]$.

Deberá obtener una lectura de $10(V)$. En caso de que no sea así, ajuste el voltaje para que sea igual a $10(V)$, mediante POT0 (R10) utilizando el desarmador de relojero.

Paso 9

Coloque la Tierra del osciloscopio en el pin J13 (GND) y la punta de medición en el pin J9 (DAC OUT).

Encienda el osciloscopio y ajústelo a una medida de $5\left(\frac{V}{cm}\right)$ y un barrido de aproximadamente $1\left(\frac{ms}{cm}\right)$. No olvide ajustar la referencia de la señal de Tierra del osciloscopio, verificar que esté calibrado y hacer las lecturas en DC.

Deberá ver en su osciloscopio una señal rampa que va de $-10(V)$ a $10(V)$ con una pendiente aproximada de $16.6\left(\frac{V}{ms}\right)$. En caso de no ser así, utilice el desarmador de relojero para ajustar los siguientes potenciómetros, hasta obtener la señal rampa indicada con anterioridad:

- POT3 (R11).- Sirve para indicar el voltaje de referencia del Convertidor Digital - Analógico. Da pendiente a la señal rampa.
- POT4 (R14).- Sirve para aumentar o disminuir la corriente de salida proporcionada por el Convertidor Digital - Analógico. Permite centrar la señal rampa.
- POT5 (R17).- Amplifica la señal de salida del Convertidor Digital - Analógico. Sirve para dar amplitud a la señal rampa.

Presione **"ACEPTAR"** hasta que haya obtenido la señal rampa indicada.

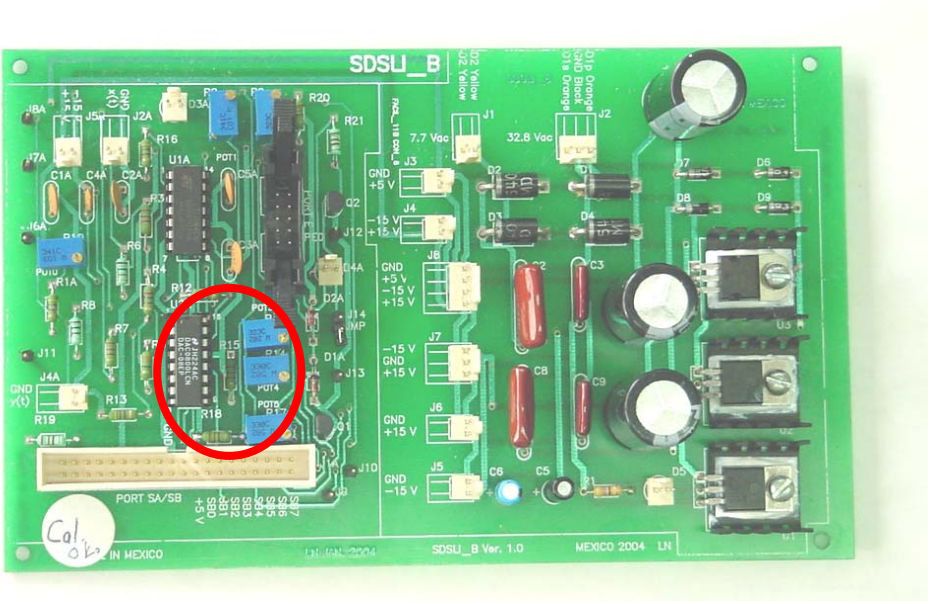


Figura 29. POT3, POT4 y POT5

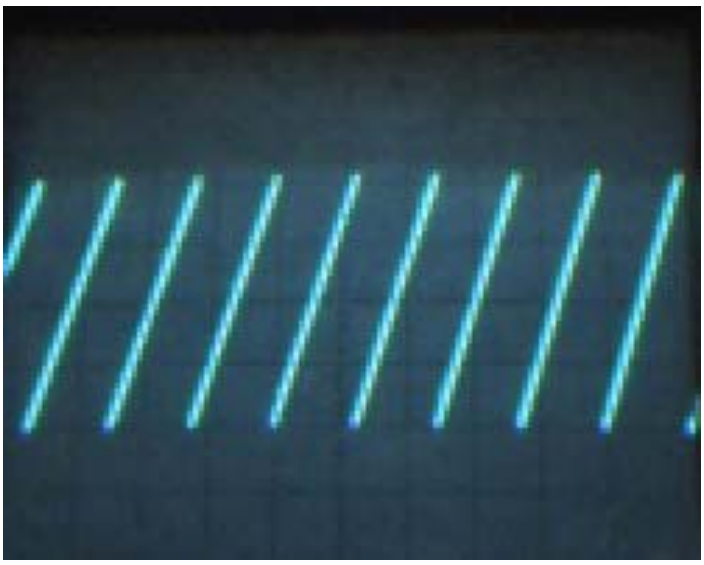


Figura 30. Señal rampa que se debe observar en el paso 9.

Escala de voltaje: $5\left(\frac{V}{cm}\right)$

Escala de tiempo de barrido: $1\left(\frac{ms}{cm}\right)$

Paso 10

El siguiente paso, es realizar un ajuste fino con el objeto de asegurar la calidad de la señal de salida.

Coloque la punta de medición de voltaje del multímetro, en el pin J10 $\left[y(t)\right]$, dejando la punta común (Tierra) en el pin J11 (GND).

Presione el botón **"RESET"** del hardware del SDSLI y posteriormente, el botón **"ACEPTAR"** de la ventana que aparece.

Paso 11

Debe observar que el led **"LISTO"** de la carátula, esté parpadeando.

Paso 12

Debe leer en su multímetro un valor próximo a los $0(V)$. De preferencia, este valor debe ser igual a $0(V)$. En caso de no ser así, utilice el desarmador de relojero para ajustar los siguientes potenciómetros, hasta obtener el valor de voltaje indicado con anterioridad:

- POT3 (R11).- Sirve para indicar el voltaje de referencia del Convertidor Digital - Analógico.
- POT4 (R14).- Sirve para aumentar o disminuir la corriente de salida proporcionada por el Convertidor Digital - Analógico.
- POT5 (R17).- Amplifica la señal de salida del Convertidor Digital - Analógico.

Consejo: Comience por ajustar el potenciómetro POT4 (R14)

Presione **"ACEPTAR"** hasta que haya obtenido el valor de voltaje indicado.

Paso 13

Ahora, debe leer en su multímetro un valor próximo a los $-10(V)$. De preferencia, este valor debe ser igual a $-10(V)$. En caso de no ser así, utilice el desarmador de relojero para ajustar los siguientes potenciómetros, hasta obtener el valor de voltaje indicado con anterioridad:

- POT3 (R11).- Sirve para indicar el voltaje de referencia del Convertidor Digital - Analógico.
- POT4 (R14).- Sirve para aumentar o disminuir la corriente de salida proporcionada por el Convertidor Digital - Analógico.
- POT5 (R17).- Amplifica la señal de salida del Convertidor Digital - Analógico.

Consejo: Comience por ajustar el potenciómetro POT4 (R14) y POT5 (R17)

Presione **"ACEPTAR"** hasta que haya obtenido el valor de voltaje indicado.

Paso 14

Finalmente, debe leer en su multímetro un valor próximo a los $10(V)$. De preferencia, este valor debe ser igual a $10(V)$. En caso de no ser así, utilice el desarmador de relojero para ajustar los siguientes potenciómetros, hasta obtener el valor de voltaje indicado con anterioridad:

- POT3 (R11).- Sirve para indicar el voltaje de referencia del Convertidor Digital - Analógico.
- POT4 (R14).- Sirve para aumentar o disminuir la corriente de salida proporcionada por el Convertidor Digital - Analógico.
- POT5 (R17).- Amplifica la señal de salida del Convertidor Digital - Analógico.

Consejo: Comience por ajustar el potenciómetro POT4 (R14) y POT5 (R17)

Presione **"ACEPTAR"** hasta que haya obtenido el valor de voltaje indicado.

Paso 15

En caso de que desee verificar que los voltajes de $-10(V)$, $0(V)$ y $10(V)$ estén en sus valores correctos, vuelva a generar y a medir estos voltajes.

Paso 16

Coloque el puente (jumper) J14 (CAL. MOD.) para poner la tarjeta SDSLI_B, en modo de calibración.

Presione **"ACEPTAR"** una vez que haya colocado el puente.

Paso 17

Coloque la Tierra del osciloscopio en el pin J13 (GND) y la punta de medición, en el pin J8A (SC OUT).

Encienda el osciloscopio y ajústelo a una medida de $5\left(\frac{V}{cm}\right)$ y un barrido de aproximadamente $1\left(\frac{ms}{cm}\right)$. No olvide ajustar la referencia de la señal de Tierra del osciloscopio, verificar que esté calibrado y hacer las lecturas en DC.

Deberá ver en su osciloscopio una señal rampa que va de $0(V)$ a $5(V)$ con una pendiente aproximada de $4.16\left(\frac{V}{ms}\right)$. En caso de no ser así, utilice el desarmador de relojero para ajustar los siguientes potenciómetros, hasta obtener la señal rampa indicada con anterioridad:

- POT1 (R2).- Sirve para indicar el voltaje de referencia del circuito acondicionador de la señal de entrada. Permite centrar la señal rampa, desplazándola verticalmente.
- POT2 (R9).- Sirve para amplificar o disminuir la magnitud de la señal de salida proporcionada por el circuito acondicionador de la señal de entrada. Da amplitud a la señal rampa.

Presione **“ACEPTAR”** hasta que haya obtenido la señal rampa indicada.

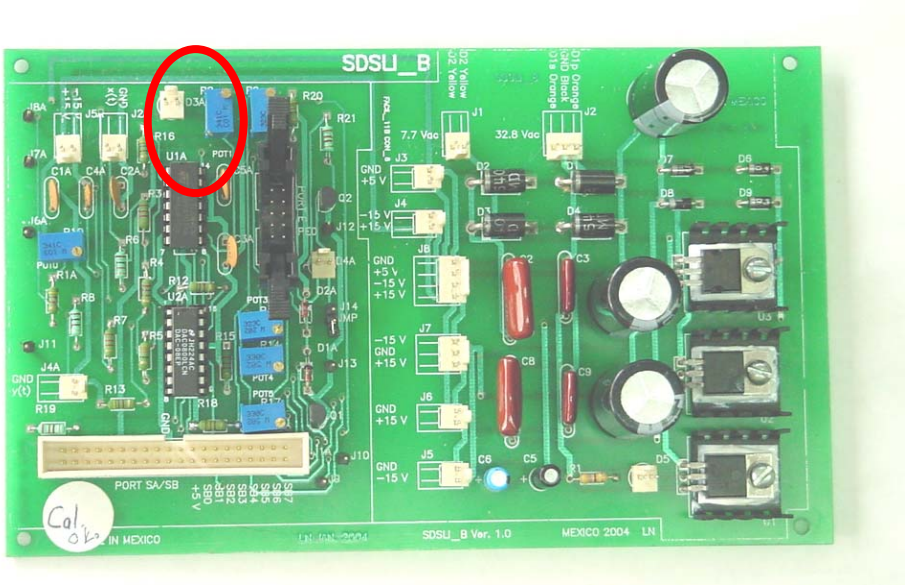


Figura 31. POT1 y POT2

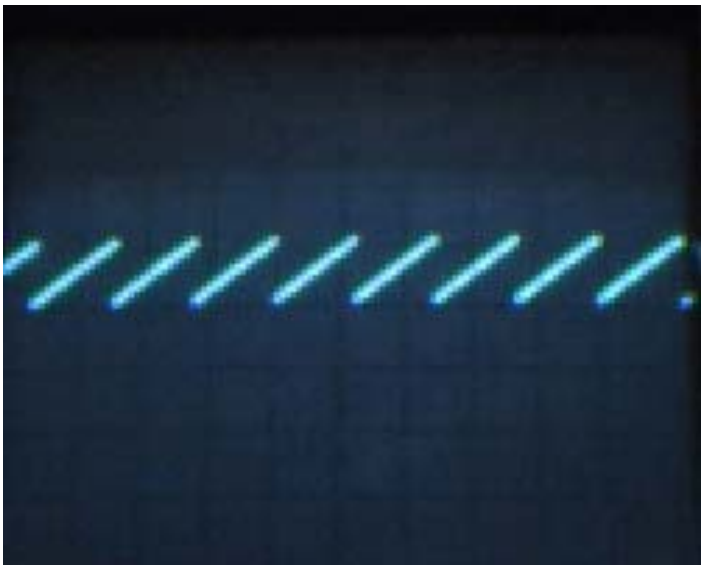


Figura 32. Señal rampa que se debe observar en el paso 17.

Escala de voltaje: $5\left(\frac{V}{cm}\right)$

Escala de tiempo de barrido: $1\left(\frac{ms}{cm}\right)$

Paso 18

Debe realizar un ajuste fino para asegurar la calidad de la señal de entrada.

Quite la punta de medición del osciloscopio del pin J8A (SC OUT) y coloque en este pin, la punta de medición de voltaje del multímetro, dejando la punta común (Tierra), en el pin J11 (GND).

Presione el botón **"RESET"** del hardware del SDSLI y posteriormente, el botón **"ACEPTAR"** de la ventana que aparece.

Paso 19

Debe observar que el led **"LISTO"** de la carátula, esté parpadeando.

Paso 20

Debe leer en su multímetro un valor próximo a los $2.5(V)$. De preferencia, este valor debe ser igual a $2.5(V)$. En caso de no ser así, utilice el desarmador de relojero para ajustar los siguientes potenciómetros, hasta obtener el valor de voltaje indicado con anterioridad:

- POT1 (R2).- Sirve para indicar el voltaje de referencia del circuito acondicionador de la señal de entrada.
- POT2 (R9).- Sirve para amplificar o disminuir la magnitud de la señal de salida proporcionada por el circuito acondicionador de la señal de entrada.

Consejo: Comience por ajustar el potenciómetro POT1 (R2).

Presione **"ACEPTAR"** hasta que haya obtenido el valor de voltaje indicado.

Paso 21

A continuación, debe leer en su multímetro un valor próximo a los $0(V)$. De preferencia, este valor debe ser igual a $0(V)$. En caso de no ser así, utilice el desarmador de relojero para ajustar los siguientes potenciómetros, hasta obtener el valor de voltaje indicado con anterioridad:

- POT1 (R2).- Sirve para indicar el voltaje de referencia del circuito acondicionador de la señal de entrada.
- POT2 (R9).- Sirve para amplificar o disminuir la magnitud de la señal de salida proporcionada por el circuito acondicionador de la señal de entrada.

Consejo: Comience por ajustar el potenciómetro POT2 (R9).

Presione **"ACEPTAR"** hasta que haya obtenido el valor de voltaje indicado.

Paso 22

Por último, debe leer en su multímetro un valor próximo a los $5(V)$. De preferencia, este valor debe ser igual a $5(V)$. En caso de no ser así, utilice el desarmador de relojero para ajustar los siguientes potenciómetros, hasta obtener el valor de voltaje indicado con anterioridad:

- POT1 (R2).- Sirve para indicar el voltaje de referencia del circuito acondicionador de la señal de entrada.
- POT2 (R9).- Sirve para amplificar o disminuir la magnitud de la señal de salida proporcionada por el circuito acondicionador de la señal de entrada.

Consejo: Comience por ajustar el potenciómetro POT2 (R9).

Presione **"ACEPTAR"** hasta que haya obtenido el valor de voltaje indicado.

Paso 23

En caso de que desee verificar que los voltajes de $0(V)$, $2.5(V)$ y $5(V)$ estén en sus valores correctos, vuelva a generar y a medir los voltajes antes mencionados.

Paso 24

Quite el puente (jumper) J14 (CAL. MOD.) para poner la tarjeta SDSLI_B, en modo de simulación.

Presione **“ACEPTAR”** una vez que haya quitado el puente.

Paso 25

Apague el hardware del SDSLI, moviendo el switch **“ENCENDIDO”** a la posición **“OFF”** y desconecte el cable de la toma de corriente eléctrica. También, desconecte el cable de comunicación con la PC.

Asegúrese de desconectar todos los cables en las terminales de **“SEÑAL DE ENTRADA”** y **“SEÑAL DE SALIDA”** del hardware del SDSLI.

Desconecte las puntas del multímetro y del osciloscopio recién utilizadas, para realizar la calibración.

Paso 26

Verifique que no haya dejado ningún tornillo ni herramientas sueltas dentro de la caja destapada del hardware del SDSLI, y con la finalidad de proteger la circuitería del hardware, ponga la tapa de la caja y atorníllela con sus respectivos tornillos.

Paso 27

Vuelva a conectar, cuidadosamente, el cable a la toma de corriente eléctrica y además, reconecte el cable de comunicación con la PC.

Paso 28

Encienda el hardware del SDSLI, moviendo el switch **“ENCENDIDO”** a la posición **“ON”**.

Paso 29

Debe observar que el led **“LISTO”** de la carátula, esté parpadeando.

Por último, el software informará que la calibración ha finalizado exitosamente.

9. INFORMACIÓN TÉCNICA

Peso	$2.5(kg)$
Tamaño	$13.5\times23\times30(cm)$ $(\text{altura}\times\text{anchura}\times\text{profundidad})$
Voltaje de operación	$127(Vac)$
Corriente consumida	$230(mA_{ac})$
Potencia consumida	$29.21(Wac)$
Temperatura de funcionamiento	$25(^{\circ}c)$

10. CUIDADO Y MANTENIMIENTO DEL SDSLI

El SDSLI es un producto de diseño y acabado superior, por lo tanto deberá ser tratado con cuidado. Las siguientes sugerencias le ayudarán a cumplir con cualquier obligación de garantía y disfrutar de este producto por muchos años.

- Mantenga el SDSLI seco. Las precipitaciones, la humedad y los líquidos contienen minerales que corroen los circuitos electrónicos. Si su dispositivo llega a mojarse, déjelo secar completamente antes de utilizarlo nuevamente.
- No use el SDSLI ni lo almacene en lugares polvorientos o sucios. Ello podría dañar sus partes movibles y componentes electrónicos.
- No almacene el SDSLI en lugares calurosos. Las temperaturas altas pueden acortar la vida de los dispositivos electrónicos, o torcer o derretir ciertos plásticos.
- No almacene el SDSLI en lugares fríos. Cuando el SDSLI alcanza su temperatura normal se puede producir humedad internamente, lo cual podría dañar las tarjetas de circuitos electrónicos.
- No trate de abrir el SDSLI a no ser para un procedimiento tratado en esta guía.
- No deje caer el SDSLI, no lo sacuda, ni lo golpee. Los manejos bruscos pueden dañar las tarjetas interiores de circuitos y mecanismos delicados.
- No use productos químicos abrasivos, solventes de limpieza, ni detergentes para limpiarlo.
- No pinte el SDSLI. La pintura puede bloquear las partes movibles e impedir un funcionamiento apropiado.

Todas estas sugerencias sirven para su SDSLI o cualquier accesorio.

MANUAL DEL USUARIO DEL SDSLI

En esta sección encontrará:

1. INTRODUCCIÓN

¡Felicidades! Gracias por elegir el Simulador Digital de Sistemas Lineales e Invariantes (SDSLI). Este simulador le permite, junto con el software propietario, desarrollar de una manera muy sencilla, simulaciones de sistemas lineales e invariantes en el tiempo de primer y hasta cuarto orden, con la finalidad de probar modelos matemáticos de sistemas lineales, sin tener que implementar el sistema físicamente.

1.1. ¿POR QUÉ NECESITO UN SIMULADOR COMO SDSLI?

Un simulador en términos actuales, es comprendido como un programa de computadora para representar y estudiar sistemas y procesos. Con equipos de laboratorio como osciloscopios, generadores de señales y multímetros conectados a una computadora a través de interfases, se puede, ya sea simular o bien, controlar el o los comportamientos de un proceso, sea lineal o no.

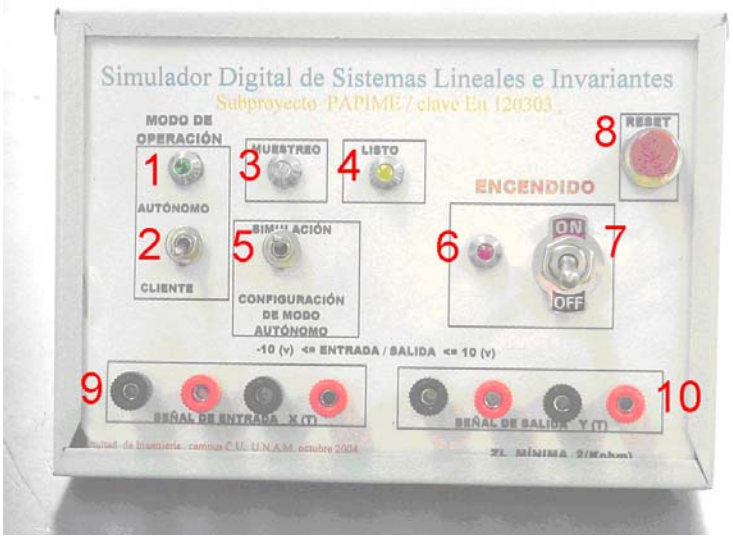
SDSLI, nos ofrece hoy en día, una gran gama de posibilidades, ya que es factible por medio de éste, representar un grupo grande de procesos, realizar operaciones matemáticas complejas y de una manera rápida, obtener gráficamente, las características de comportamiento del proceso. Esto representa una ventaja para el análisis de sistemas; ya que este tipo de simulador tiene la ventaja de que en unos instantes, permite realizar cambios graduales o sustanciales en nuestro proceso y rápidamente podemos ver cuales serán las respuestas de los sistemas simulados, ahorrando tiempo y proporcionando información sobre cuáles podrían ser los errores que puede presentar el proceso lo cual, permite a su vez, diseñar un sistema lineal más robusto.

1.2. ¿CÓMO ES QUE SDSLI LLEVA A CABO UNA SIMULACIÓN?

SDSLI es un conjunto de hardware y software que actúan simultáneamente para realizar una simulación. Usted conecta a su PC (vía puerto serial) el hardware del SDSLI, ejecuta la aplicación de software que corre bajo el ambiente Windows e introduce los datos del sistema lineal que desea simular. Una vez capturados los datos en la aplicación de Windows, inicia la simulación. El software comunica los datos a la computadora monotablilla del hardware del SDSLI. Ésta última, realiza los cálculos necesarios y envía los resultados de regreso a la PC vía puerto serial y además, convierte los datos en formato digital a un formato analógico, de tal manera que usted los pueda medir en forma de señal eléctrica, conectando un osciloscopio o un multímetro a los bornes indicados como **"SEÑAL DE SALIDA Y(T)"** de la carátula del hardware del simulador.

Este breve instructivo, le guiará rápidamente, para conectar el hardware del SDSLI a su PC, conocer las funciones básicas del software e iniciar una simulación utilizando para ello, la configuración más sencilla.

1.3. LOS CONTROLES DEL HARDWARE DEL SDSLI



Los controles del hardware del SDSLI

1. **Led "MODO DE OPERACIÓN".** Se enciende cuando el hardware del SDSLI se encuentra operando el modo autónomo. De otro modo, se mantiene apagado indicando que el hardware opera en modo cliente.
2. **Switch "MODO DE OPERACIÓN".** Se utiliza para hacer que el hardware del SDSLI opere en modo "AUTÓNOMO" (sin necesidad de la PC) o bien en modo "CLIENTE" (espera a recibir órdenes de una PC).
3. **Led "MUESTREO".** Se enciende cuando el hardware del SDSLI se encuentra ejecutando una simulación. El parpadeo de este led, es un testigo visual de la velocidad del periodo de muestreo de la señal de entrada $x(t)$.
4. **Led "LISTO".** Se enciende cuando el hardware del SDSLI se encuentra listo para recibir órdenes de una PC. Si el led "LISTO" se encuentra parpadeando, significa que está esperando órdenes del usuario mediante una PC. Si se queda TOTALMENTE ENCENDIDO, significa que el hardware del SDSLI está ejecutando una simulación y no recibirá ordenes del usuario hasta finalizar la simulación en curso. Si el led "LISTO" NO ESTÁ ENCENDIDO, no podrá realizar ninguna clase de simulación.
5. **Switch "SIMULACIÓN / CONFIGURACIÓN DE MODO AUTÓNOMO".** Este switch permite poner al hardware del SDSLI en la modalidad de simulación de un sistema lineal o bien en la modalidad para configuración del hardware para que ejecute posteriormente, una simulación en modo autónomo.
6. **Led "ENCENDIDO".** Este led permite visualizar si el hardware del SDSLI está encendido.
7. **Switch "ENCENDIDO".** Switch que permite prender o apagar el hardware del SDSLI.
8. **Botón "RESET".** Permite reinicializar el hardware del SDSLI.
9. **Bornes "SEÑAL DE ENTRADA X(T)".** Estos bornes permiten al usuario conectar señales eléctricas que corresponden a las excitaciones del sistema que se simule. En estos bornes, es posible conectar otro simulador SDSLI, un generador de señales, una planta de control, etcétera. Es importante cuidar que las señales que se introduzcan en estos bornes sean señales que deben encontrarse en el intervalo de $-10(V)$ a $10(V)$. Se sugiere no introducir señales

eléctricas cuyo voltaje exceda estos límites, pues se pueden provocar daños irreversibles a los circuitos del hardware del simulador. Los bornes **NEGROS** corresponden a tierra o negativo y los bornes **ROJOS** indican vivo o positivo.

10. **Bornes "SEÑAL DE SALIDA Y(T)".** Estos bornes permiten al usuario obtener señales eléctricas que corresponden a la respuesta del sistema que se simule. En estos bornes es posible conectar otro simulador SDSLI o un osciloscopio. Es importante cuidar que NO SE INTRODUZCAN SEÑALES en estos bornes. pues es posible provocar daños irreversibles a los circuitos del hardware del simulador. Cabe señalar que el hardware del simulador, ES UN SISTEMA QUE ENTREGA SEÑAL, NO POTENCIA. Por ningún motivo deberá conectar carga a los bornes de salida $y(t)$ cuya impedancia sea menor a $2(K\Omega)$ porque la circuitería del simulador corre el riesgo de quemarse. Si desea conectar sistemas a los bornes **"SEÑAL DE SALIDA Y(T)"** cuya impedancia sea menor a la indicada con anterioridad o bien, sistemas que consuman demasiada corriente eléctrica, como por ejemplo un motor, deberá adaptar un circuito acondicionador y una fuente de poder para poder realizar esta acción. Los bornes **NEGROS** corresponden a tierra o negativo y los bornes **ROJOS** indican vivo o positivo.

2. INSTALACIÓN Y CONFIGURACIÓN DEL HARDWARE DEL SDSLI

2.1. REQUERIMIENTOS DE SU SISTEMA PARA EJECUTAR SDSLI

Estos son los requerimientos mínimos que SDSLI necesita para su instalación y ejecución:

- Microprocesador Pentium®.
- Microsoft® Windows® 98 Second Edition, Windows NT Workstation 4.0 con Service Pack 6, Windows Me, Windows 2000 Professional con Service Pack 2, Windows XP Professional o Home Edition.
- 64 (MB) de RAM [recomendados 128 (MB)]
- 22 (MB) de espacio disponible en su unidad de disco duro.
- Unidad de CD-ROM para instalar el software desde CD.
- Internet Explorer 5.01 o más actual.
- Adobe Acrobat Reader® 5 o superior.
- Al menos un puerto serial disponible con conexión DB9 macho.
- Hardware del SDSLI con todos sus cables correspondientes.

2.2. INSTALANDO EL SDSLI

1. Coloque el hardware del SDSLI en una superficie horizontal y plana. De preferencia, el hardware del SDSLI debe estar cerca de su PC para que el cable de comunicaciones se pueda conectar con facilidad.

El hardware del SDSLI posee, en su parte lateral 2 conectores:

1) El conector superior, permite conectar el cable de comunicación serial del hardware del SDSLI, a la PC.

2) El conector inferior, permite conectar el hardware del SDSLI a la toma de corriente eléctrica.

2. Con su PC apagada, conecte el hardware del SDSLI a su PC mediante el cable de comunicación serial. Su PC debe de disponer forzosamente, de por lo menos, un puerto serial libre para que pueda comunicarse con el hardware del SDSLI. En caso de que no posea puertos seriales libres y que tenga la posibilidad de utilizar puertos USB, deberá adquirir una interfaz que emule un puerto serial de baja velocidad.

3. Conecte el hardware del SDSLI a la corriente eléctrica mediante el cable correspondiente.

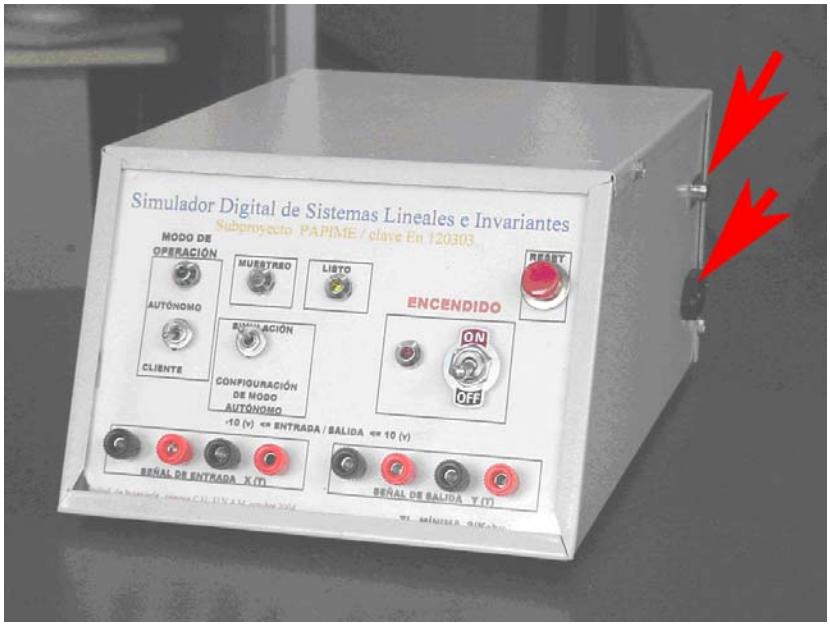


Figura 1. Vista del hardware del SDSLI donde se aprecian el conector de comunicaciones seriales (parte superior) y el conector de toma de corriente eléctrica (parte inferior)

El hardware del SDSLI debe estar conectado a su PC tal y como lo muestra el esquema de la figura 2.

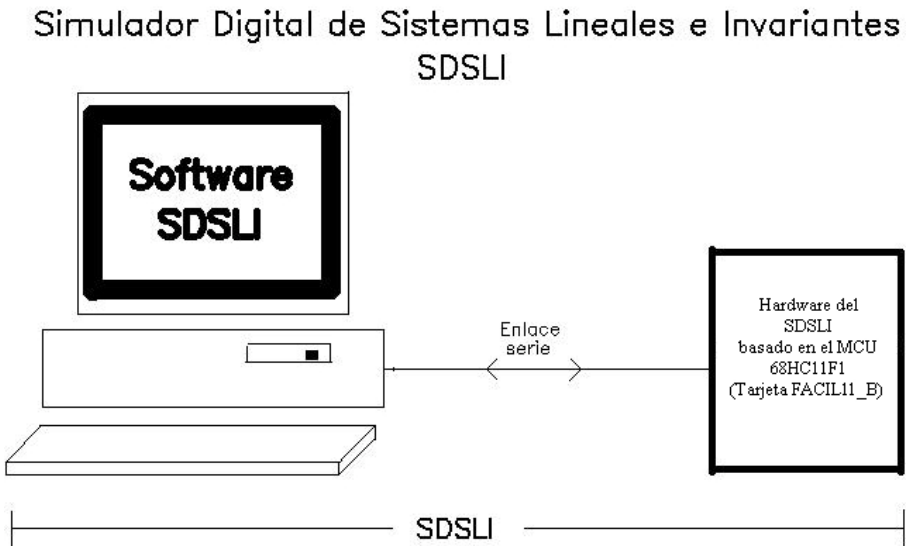


Figura 2. Conexión del hardware del SDSLI a la PC

4. Mueva el switch **"MODO DE OPERACIÓN"** a la posición **"CLIENTE"**. Tal y como lo muestra la figura 3.

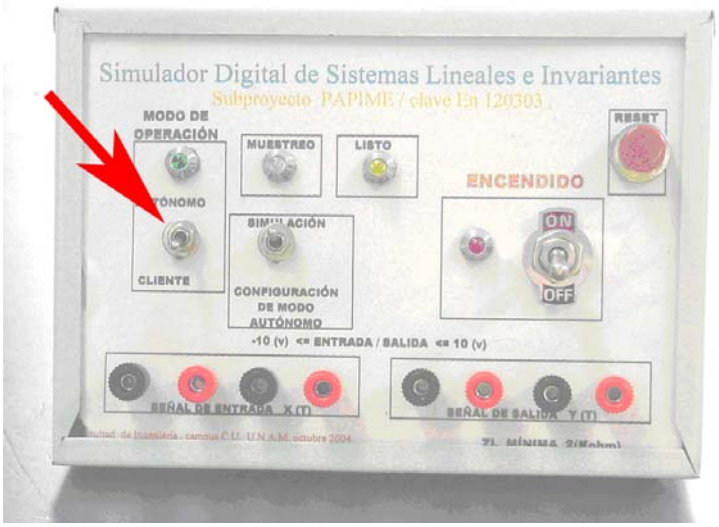


Figura 3. Switch "MODO DE OPERACIÓN" en la posición "CLIENTE"

5. Mueva el switch a la posición **"SIMULACIÓN"** tal y como lo muestra la figura 4.

Figura 4. Switch en la posición "SIMULACIÓN"

6. Encienda su PC y espere a que esta inicialice totalmente.

7. Inserte el CD correspondiente del software para Windows del SDSLI. El CD auto arrancará y verá en la pantalla de su PC la ventana señalada en la figura 5. En caso de que el CD no auto arranque, vaya a **Mi PC** y ejecute el archivo **AUTORUN.EXE** contenido en el directorio raíz del CD.

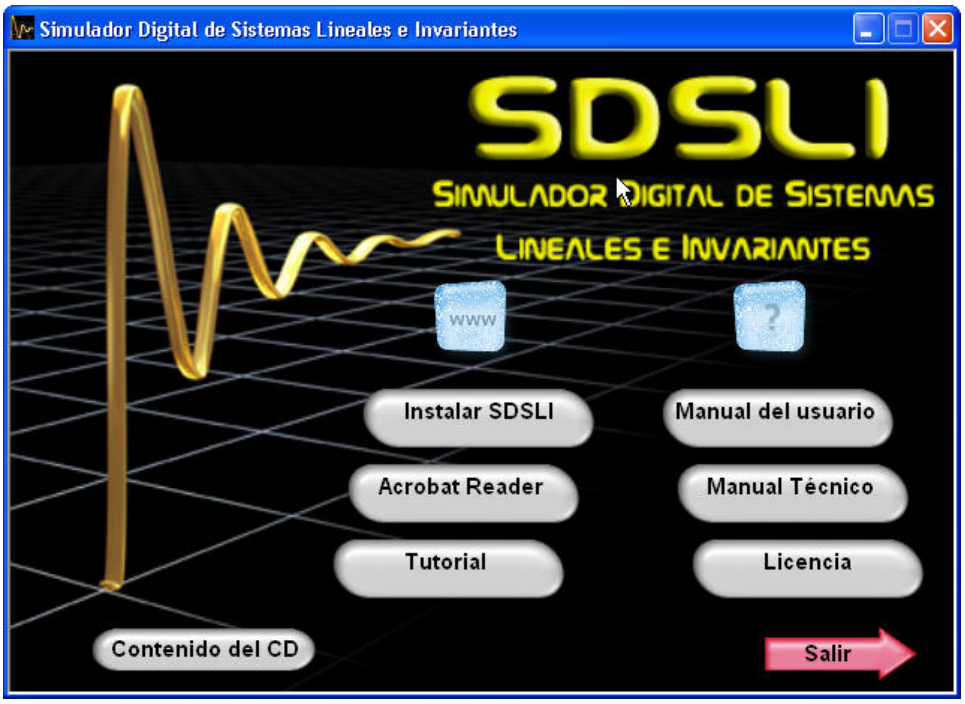


Figura 5. Ventana del CD de auto arranque del software del SDSLI

8. Del menú de auto arranque del CD, seleccione la opción **"Instalar SDSLI"**. Verá una ventana como la mostrada en la figura 6. Siga las instrucciones para instalar el software del SDSLI en su PC.

Figura 6. Ventana de instalación del software del SDSLI

Una vez que la instalación del software del SDSLI ha finalizado, su PC estará lista para ejecutar el software y realizar simulaciones.

9. Encienda el hardware del SDSLI moviendo el switch **"ENCENDIDO"** a la posición **"ON"**. Vea la figura 7.

Figura 7. Switch "ENCENDIDO"

10. Vaya a **Inicio** → **Programas** → **SDSLI** y seleccione la opción **SDSLI 1.0** o bien, en el escritorio de Windows, podrá encontrar el icono 

, haga doble click sobre éste, para iniciar la aplicación.

11. El software del SDSLI iniciará, verá la ventana de presentación del software.

Figura 8. Ventana de presentación del software del SDSLI

12. El programa le indicará el puerto y la velocidad de conexión con la cual se comunicará con el hardware del SDSLI. Vea la figura 9.

Si el hardware del SDSLI está conectado al puerto que le indica esta pantalla, entonces haga click en botón **"Sí"** de otra forma haga click en el botón **"No"** y aparecerá la ventana mostrada en la figura 10; elija el puerto de comunicaciones al cual conectó el hardware del SDSLI, haga click en "Aplicar" y luego haga click en "Aceptar".

Figura 9. Ventana que indica la velocidad y el puerto de comunicaciones mediante el cual se comunica la PC con el hardware del SDSLI

Figura 10. Ventana que permite cambiar el puerto de comunicaciones de la PC

13. Si el puerto de comunicaciones de la PC es el correcto, unos instantes después, verá en su pantalla la ventana principal del software del SDSLI (ver figura 11) y verá que el led **"LISTO"** del hardware del SDSLI está parpadeando; esta listo para simular. En caso de que el puerto de comunicaciones sea incorrecto o bien el hardware del SDSLI esté apagado en el momento en que inicializa el software del SDSLI, verá en su pantalla diversos mensajes de error, apague el hardware del SDSLI y regrese al paso 9.

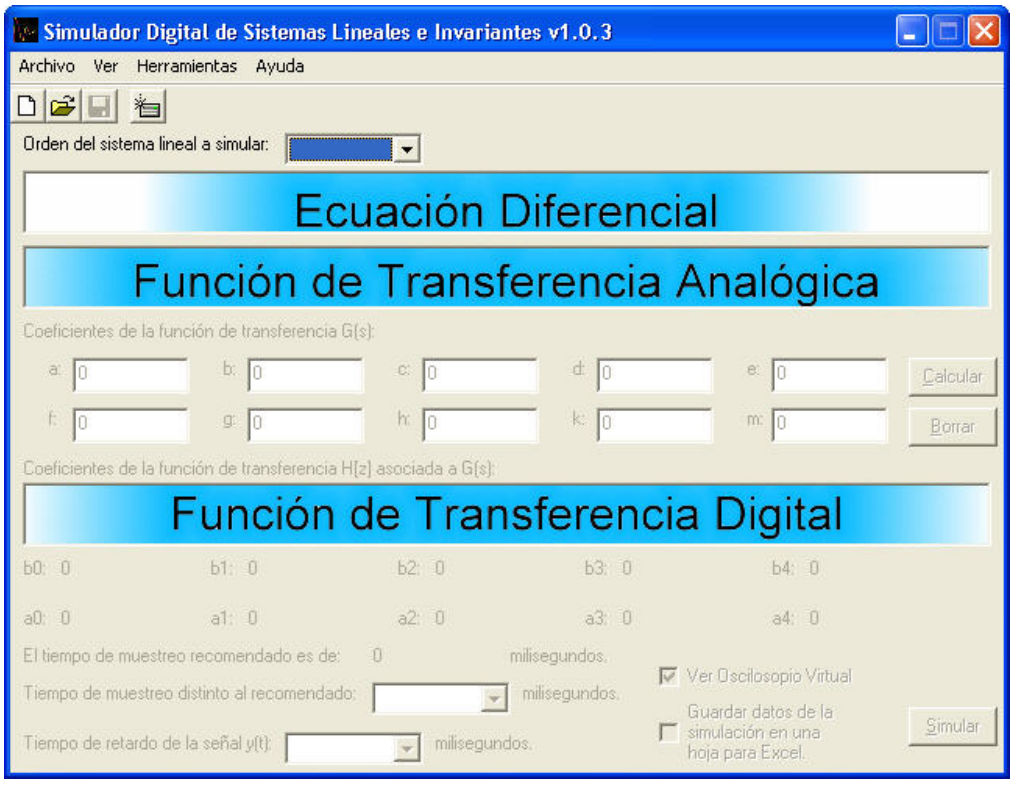


Figura 11. Ventana principal del software del SDSLI

14. Si ve en la pantalla de su PC la ventana principal del software del SDSLI y además el led **"LISTO"** del hardware del SDSLI está parpadeando, ¡Felicidades, ha usted conseguido instalar exitosamente el hardware y el software del SDSLI! Esta listo para comenzar a realizar simulaciones. En caso de que no vea la ventana principal del software en la pantalla de su PC y tampoco el led **"LISTO"** del hardware del SDSLI esté parpadeando, repita los pasos de **"2. Instalación y configuración del hardware del SDSLI"** cuidadosamente.

3. EL SOFTWARE DEL SDSLI

3.1. LA VENTANA PRINCIPAL

La ventana principal del software del SDSLI, le permite acceder a todas las opciones y herramientas que se requieren para desarrollar una simulación con el hardware del SDSLI. Su objetivo principal, es permitir que controle el hardware del SDSLI de una manera simple y directa mediante una PC.

La ventana principal, le posibilita el desarrollo de simulaciones con funciones de transferencia de tipo analógico $G(s)$.

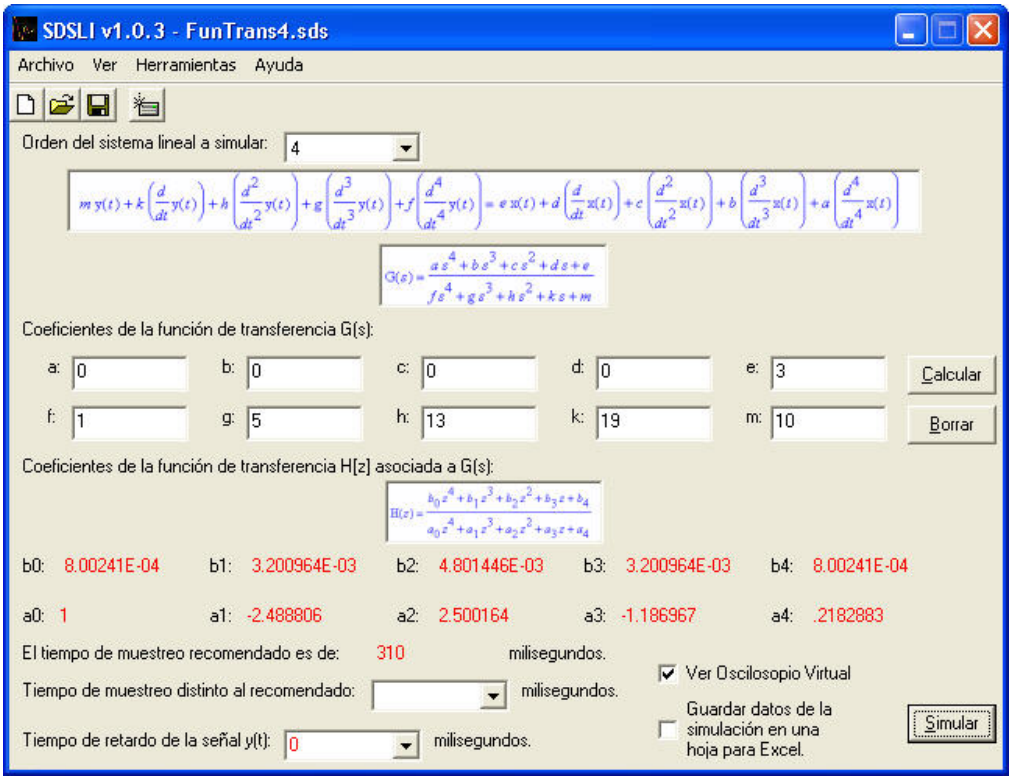


Figura 12. Ventana principal del software SDSLI

En la figura 12 es posible apreciar la ventana principal del software del SDSLI a través de la cual se ordena al hardware del SDSLI, realizar simulaciones con datos que se introducen en esta ventana.

3.1.1. Las barras de la ventana principal

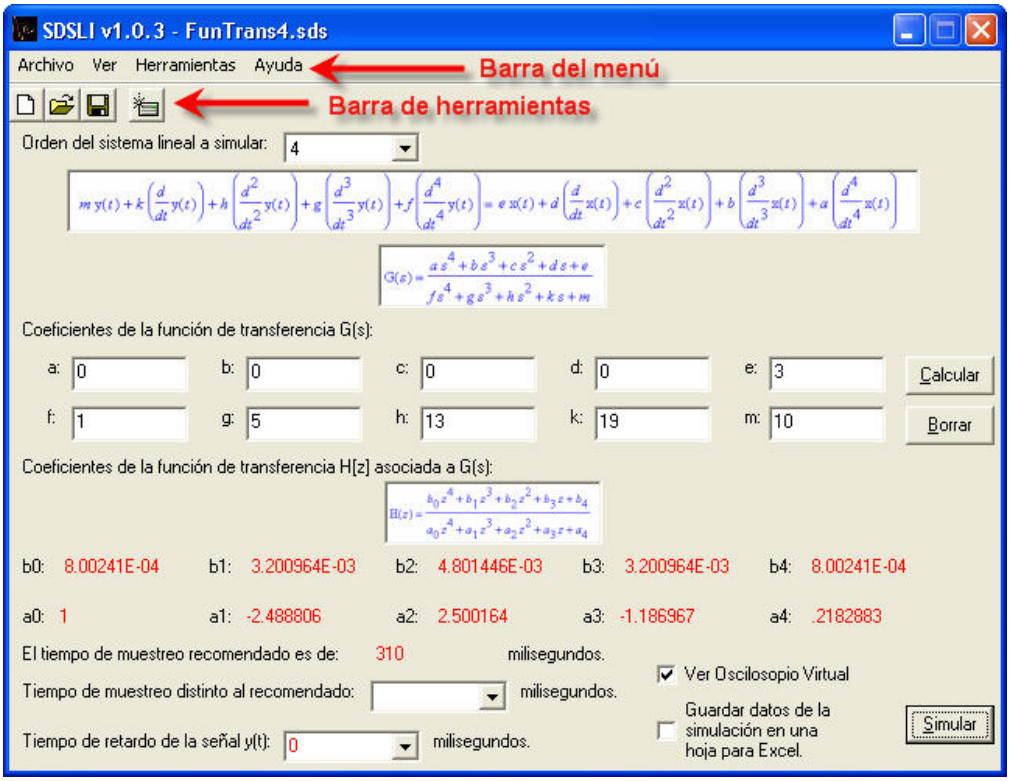


Figura 13. La barra del menú y la barra de herramientas

La barra del menú, contiene algunas opciones que le permiten guardar y abrir datos de una simulación, cambiar el puerto serial de comunicaciones, simular una función de transferencia $H[z]$, configurar el hardware del SDSLI para que funcione en modo autónomo, entre otras.

La barra de herramientas, le provee de un acceso abreviado a las funciones más utilizadas del software como son: *abrir un nuevo archivo*, *abrir un archivo*

previamente almacenado en disco, *guardar un archivo en disco* y *reinicializar el hardware del SDSLI*.

3.1.2. El menú Archivo

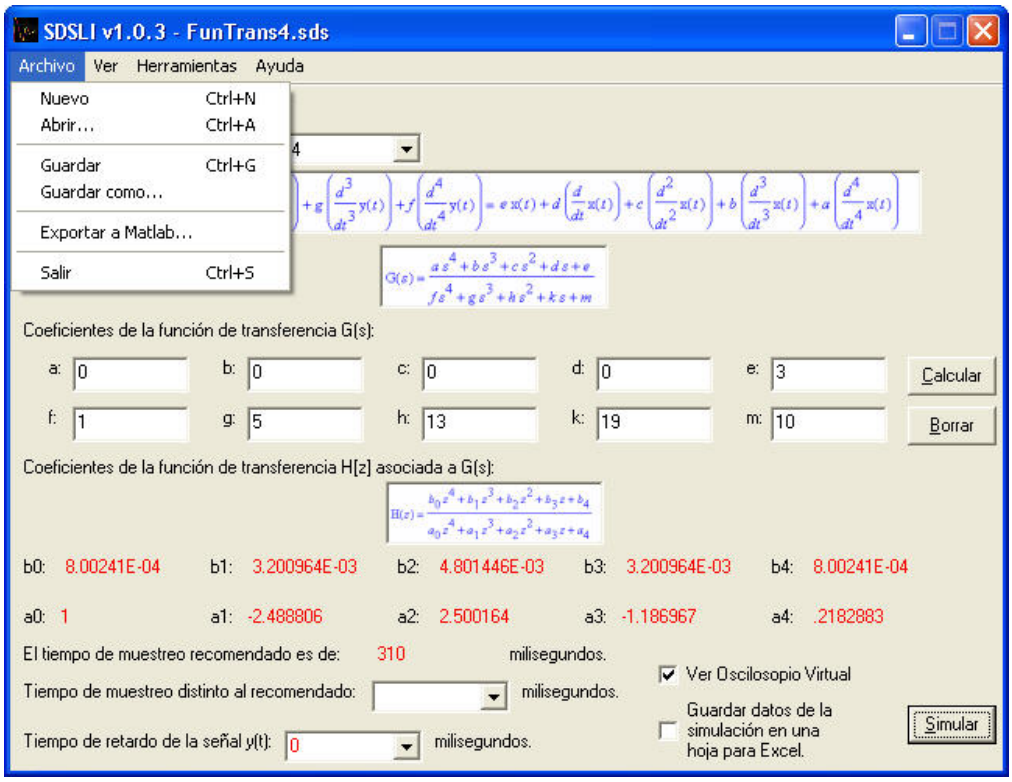


Figura 14. El menú Archivo de la ventana principal

3.1.2.1. Las opciones del menú Archivo

1. **Nuevo:** Permite crear una nueva simulación de una función de transferencia $G(s)$.
2. **Abrir…:** Permite abrir un archivo ***.SDS*** previamente almacenado en disco.
3. **Guardar como…:** Permite almacenar en disco, por primera vez, los datos de una simulación que se muestran en la ventana. Permite establecer el nombre del archivo, en el que se almacenarán los datos de la simulación.

4. **Guardar:** Permite almacenar los datos de una simulación en un archivo previamente almacenado en disco con el comando **Guardar como…**.
5. **Exportar a Matlab…:** Crea automáticamente un programa ***.M*** con los coeficientes presentes en la ventana, de la función de transferencia $G(s)$. El programa creado automáticamente por el software del SDSLI, es posible ejecutarlo en Matlab.
6. **Salir:** Permite terminar la aplicación.

3.1.3. El menú Ver

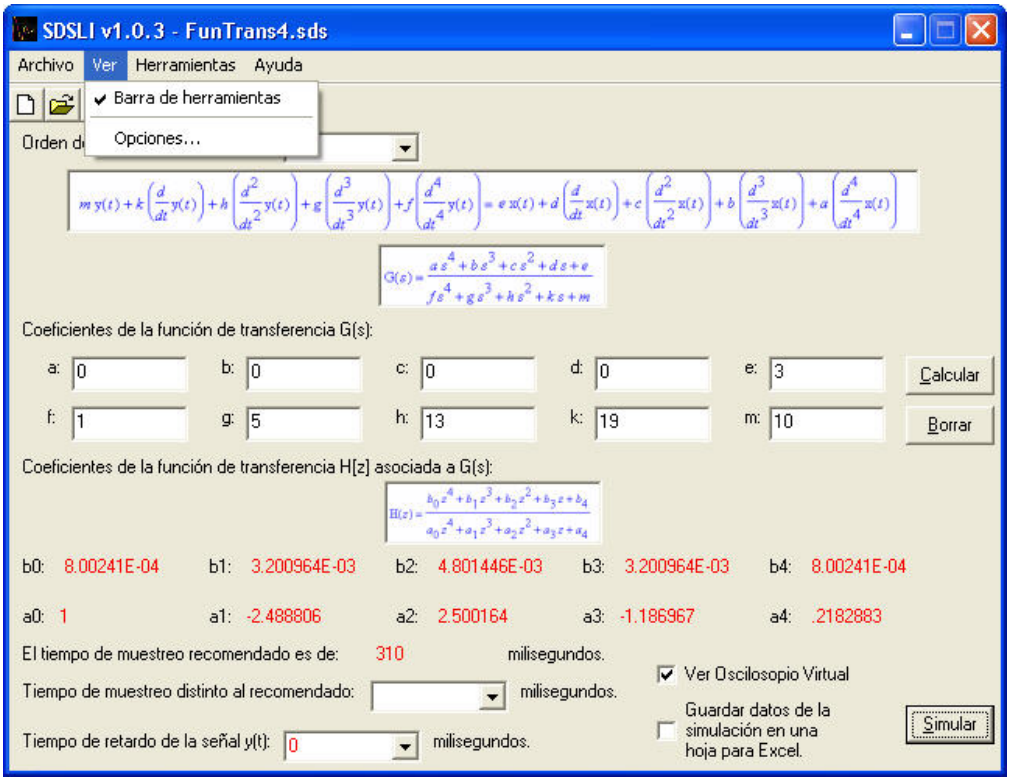


Figura 15. El menú Ver de la ventana principal

3.1.3.1. Las opciones del menú Ver

1. **Barra de herramientas:** Permite hacer aparecer o desaparecer la barra de herramientas de la ventana principal.

2. **Opciones…:** Permite cambiar las opciones y distintas configuraciones del software SDSLI.

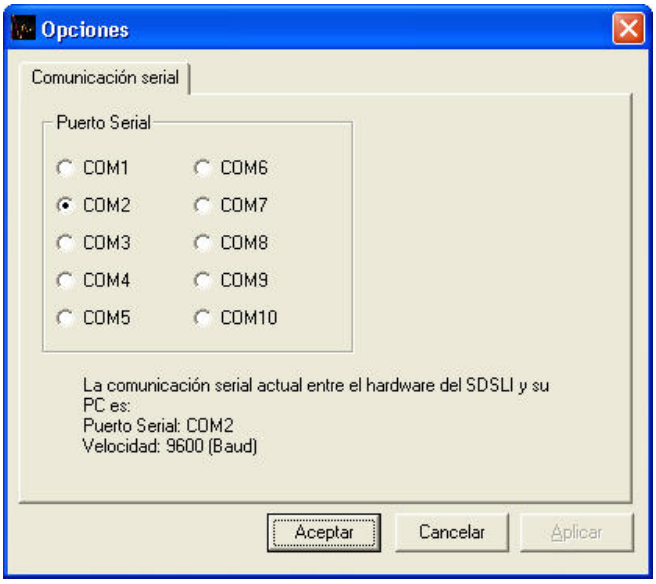


Figura 16. La ventana de opciones

En la ventana Opciones, es posible cambiar el puerto serial de comunicaciones a través del cual, la PC se comunica con el hardware del SDSLI. El botón **"Aplicar"** almacena para sesiones posteriores, los datos del puerto serial de comunicaciones, por otra parte, el botón **"Aceptar"** almacena los cambios sólo para la sesión actual y el botón **"Cancelar"** ignora los cambios hechos, dejando la configuración previa a la apertura de esta ventana.

3.1.4. El menú Herramientas

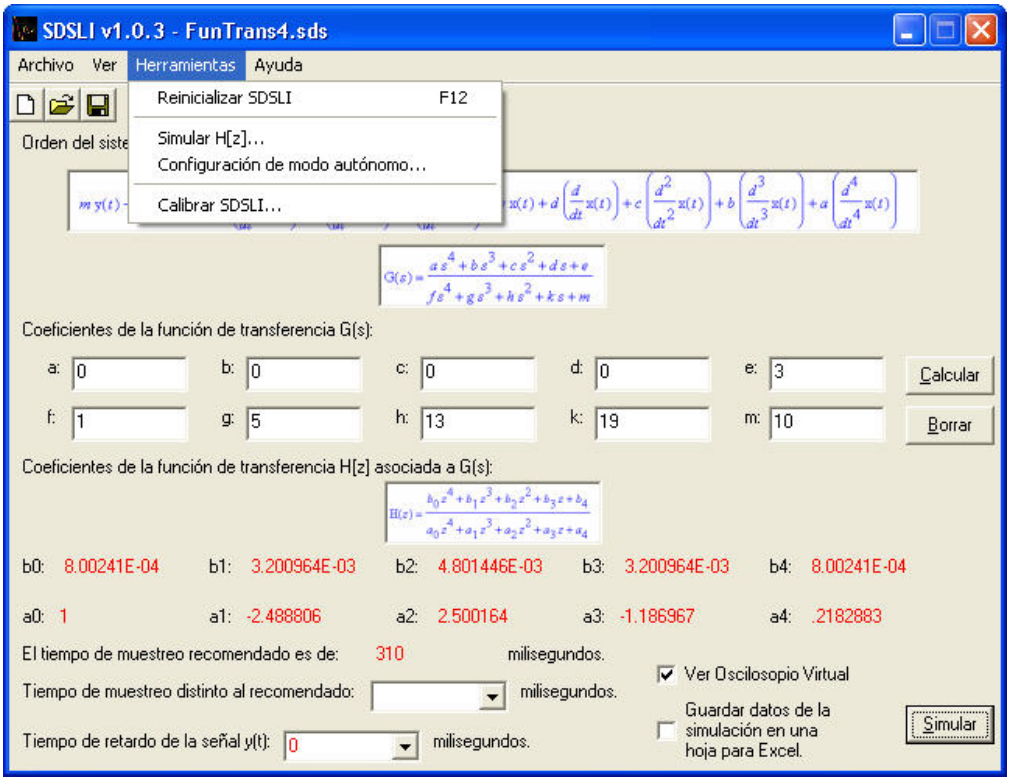


Figura 17. El menú Herramientas de la ventana principal

3.1.4.1. Las opciones del menú Herramientas

1. **Reinicializar SDSLI:** Esta opción al igual que el botón **"Reinicializar SDSLI"** de la barra de herramientas, permiten realistar el hardware del SDSLI, haciendo que el led **"LISTO"** del hardware, comience a parpadear y preparando al hardware, para recibir órdenes. Esta acción es muy utilizada cuando se apaga el hardware del SDSLI o bien cuando se presiona el botón **"RESET"** del hardware sin haber salido del software.

Es muy importante resaltar que si el led **"LISTO"** no se encuentra parpadeando, el hardware del SDSLI no podrá recibir órdenes del usuario a través de la PC y tampoco podrá ejecutar una simulación.

Cuando el led **"LISTO"** está encendido, pero no está parpadeando, significa que el hardware del SDSLI se encuentra ejecutando una simulación. Cuando el led está parpadeando, significa que el hardware del SDSLI esta listo y esperando a recibir órdenes. Cuando este led está apagado, significa que el hardware no está listo. Para este último caso, si desea realizar una simulación, deberá presionar el botón **"RESET"** del hardware del SDSLI y seleccionar la opción **"Reinicializar SDSLI"** del menú **"Herramientas"** o bien seleccionar el botón **"Reinicializar SDSLI"** de la barra de herramientas.

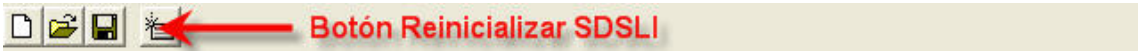


Figura 18. El botón Reinicializar SDSLI de la barra de herramientas

2. **Simular H[z]…:** Le permite realizar una simulación introduciendo los coeficientes de una función de transferencia $H[z]$. Al usar este comando, aparecerá una ventana muy parecida a la ventana principal. En la sección 3.2 de este capítulo se explica la ventana **"Simulación de función de transferencia H[z]"**.
3. **Configuración de modo autónomo…:** Inicia el asistente para configurar la operación del hardware del SDSLI en modo autónomo. El modo autónomo, es una modalidad en la que el SDSLI puede trabajar sin recibir órdenes del usuario mediante una PC. Consiste en dejar programado el hardware del SDSLI, para que cuando éste sea encendido o reseteado, automáticamente comience a ejecutar una simulación sin tener que recibir órdenes del usuario a través de una PC.

Figura 19. Ventana inicial del asistente de configuración del modo autónomo

4. **Calibrar SDSLI…:** Ejecuta el asistente de calibración del SDSLI. Para más información, revisar el Manual Técnico en la sección "**8. Calibración del SDSLI"**.

Figura 20. Ventana inicial del asistente de calibración del hardware del SDSLI

3.1.5. El menú Ayuda

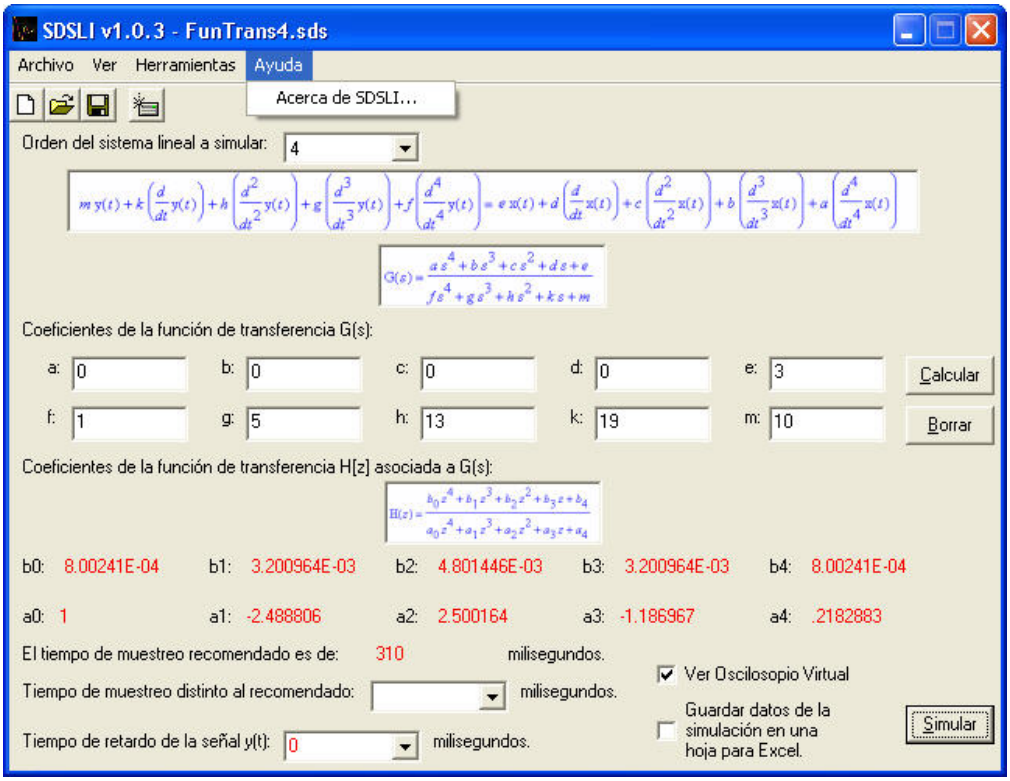


Figura 21. El menú Ayuda de la ventana principal

3.1.5.1. Las opciones del menú Ayuda

1. **Acerca de SDSLI…:** Abre una ventana con información relativa a la versión del software del SDSLI y los nombres de los autores que participaron en el desarrollo del SDSLI.

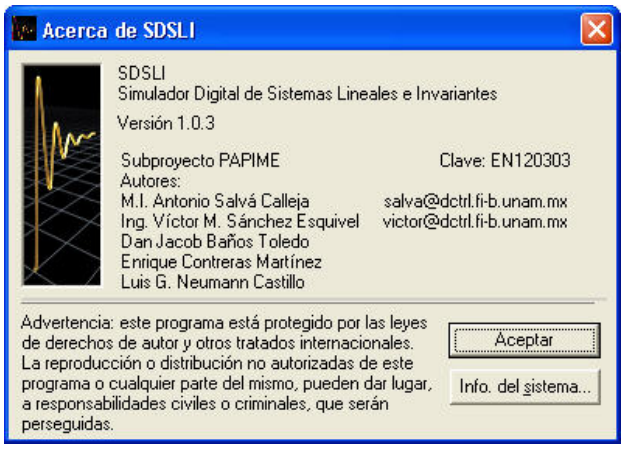


Figura 22. La ventana Acerca de SDSLI

3.1.6. Orden del sistema lineal a simular

En esta sección de la ventana principal, tiene la posibilidad de indicar el orden del sistema lineal e invariante que desea simular. Las posibilidades de orden que puede seleccionar van desde el orden cero hasta el cuarto orden.

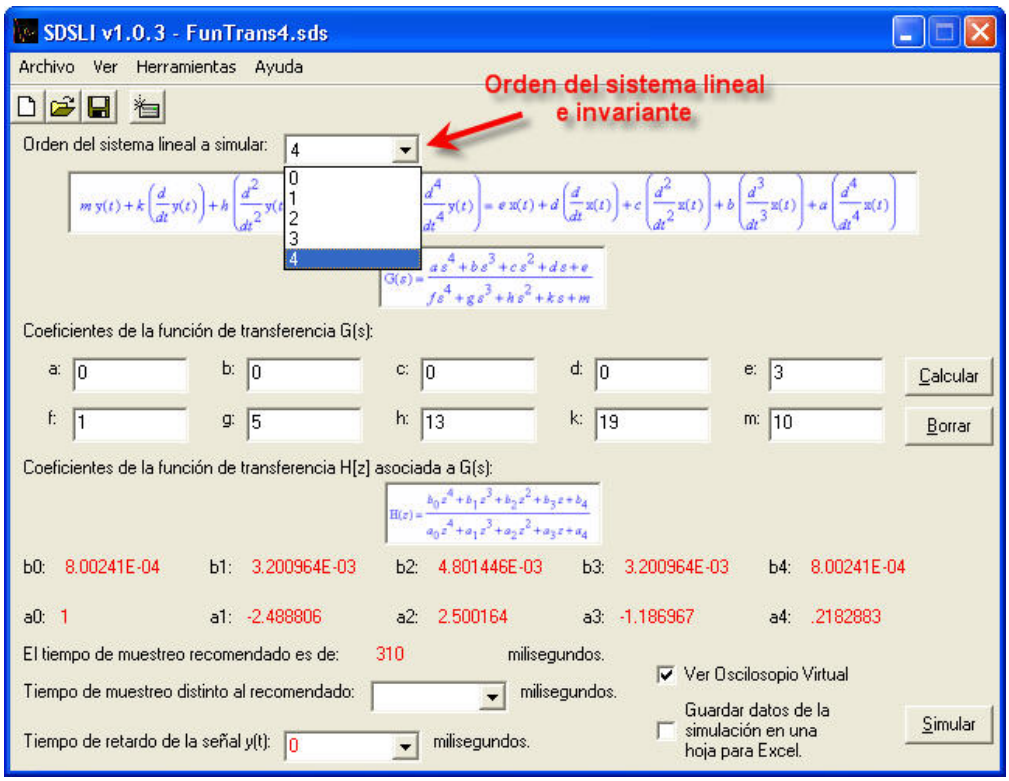


Figura 23. Orden del sistema lineal en la ventana principal

3.1.7. Ecuación diferencial y función de transferencia analógica

En esta parte de la ventana principal, se le informa sobre la forma genérica de la ecuación diferencial lineal que representa al sistema lineal e invariante que se desea simular. También, se despliega la forma genérica de la función de transferencia $G(s)$ a simular. Es importante observar estas formas genéricas, pues proporcionan la información sobre la posición de los coeficientes que deberá introducir como valores para realizar una simulación.

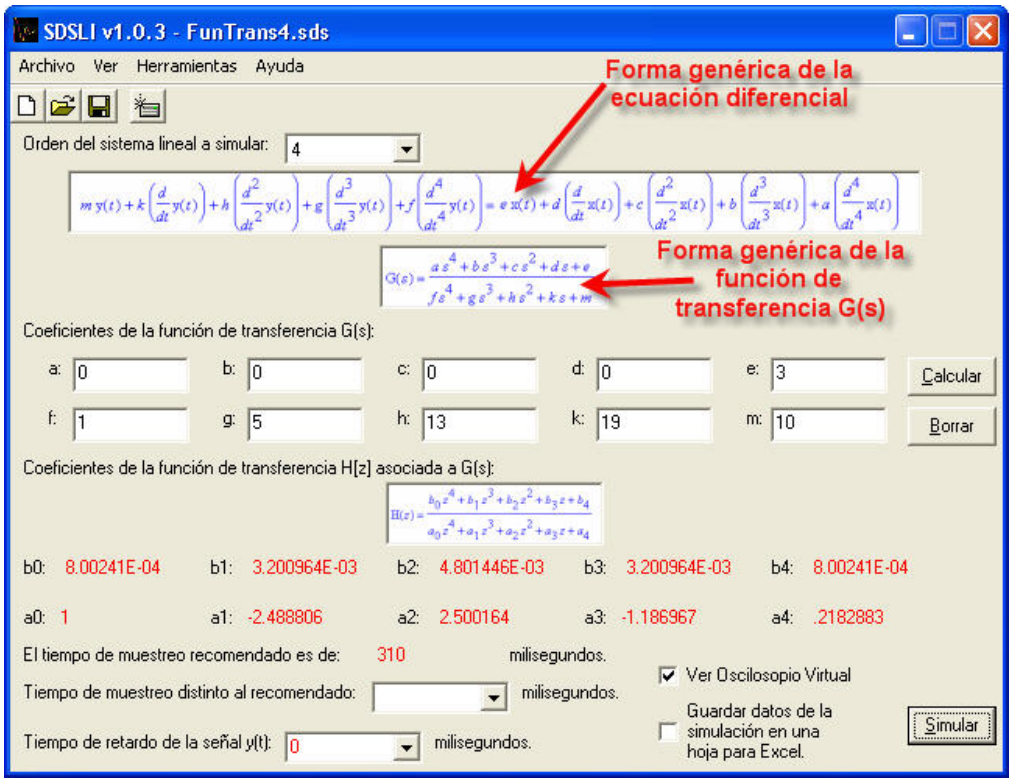


Figura 24. Ecuación diferencial y función de transferencia G(s) en la ventana principal

3.1.8. Los coeficientes de la función de transferencia G(s)

La ventana principal del software del SDSLI, cuenta con esta sección para permitirle introducir los valores de los coeficientes que corresponden a la función de transferencia $G(s)$ que desea simular. Nótese que los nombres de los coeficientes coinciden con los nombres mostrados en la ecuación diferencial lineal y en la función de transferencia.

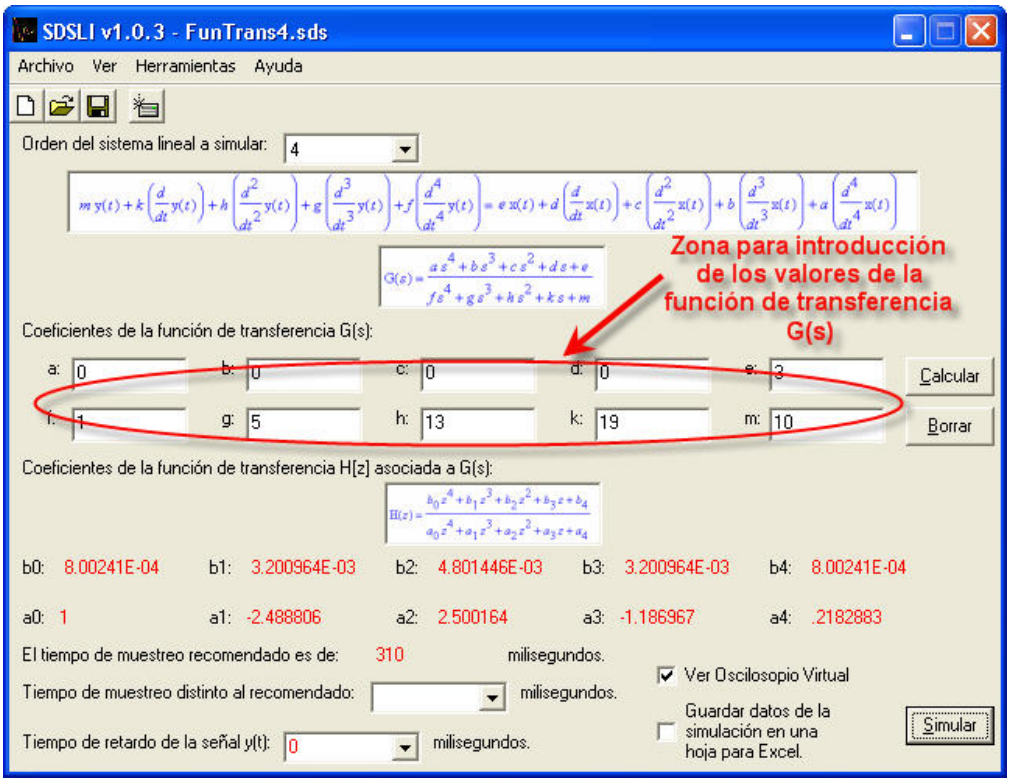


Figura 25. Zona de la ventana principal, para la introducción de los coeficientes del sistema lineal e invariante a simular

3.1.9. El botón calcular y el botón borrar

Debido a que el simulador es de tipo digital, no es posible simular funciones de transferencia $G(s)$ de forma directa; es necesario convertir los valores de los coeficientes de ésta función, a valores equivalentes para obtener una función de transferencia digital $H[z]$ que sí sea posible simular de forma inmediata mediante el simulador. Para este efecto, la ventana principal cuenta con el botón **"Calcular"**, el cual le permite, una vez que ha introducido los valores de la función de transferencia $G(s)$ que desea simular, calcular de forma automática, los valores equivalentes y correspondientes de la función de transferencia $H[z]$ además, de calcular un tiempo de muestreo recomendado para realizar la simulación de la función de transferencia deseada.

Por otro lado, el botón **"Borrar"**, le permite eliminar TODOS los coeficientes introducidos, estableciendo una configuración por defecto para éstos.

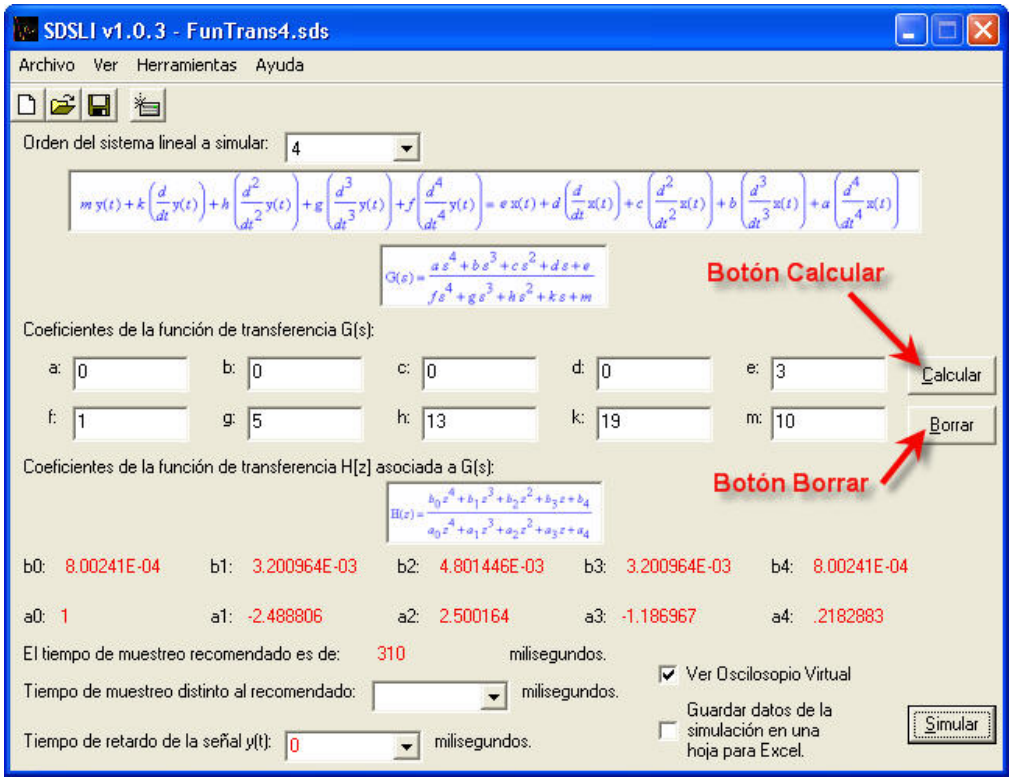


Figura 26. El botón Calcular y el botón Borrar de la ventana principal

3.1.10. La función de transferencia H[z]

En esta parte de la ventana principal, se muestra la forma genérica de la función de transferencia $H[z]$ que se va a simular y que es equivalente a la función de transferencia $G(s)$ que desea simular. Nuevamente, esta función muestra la posición de los coeficientes dentro la función de transferencia $H[z]$.

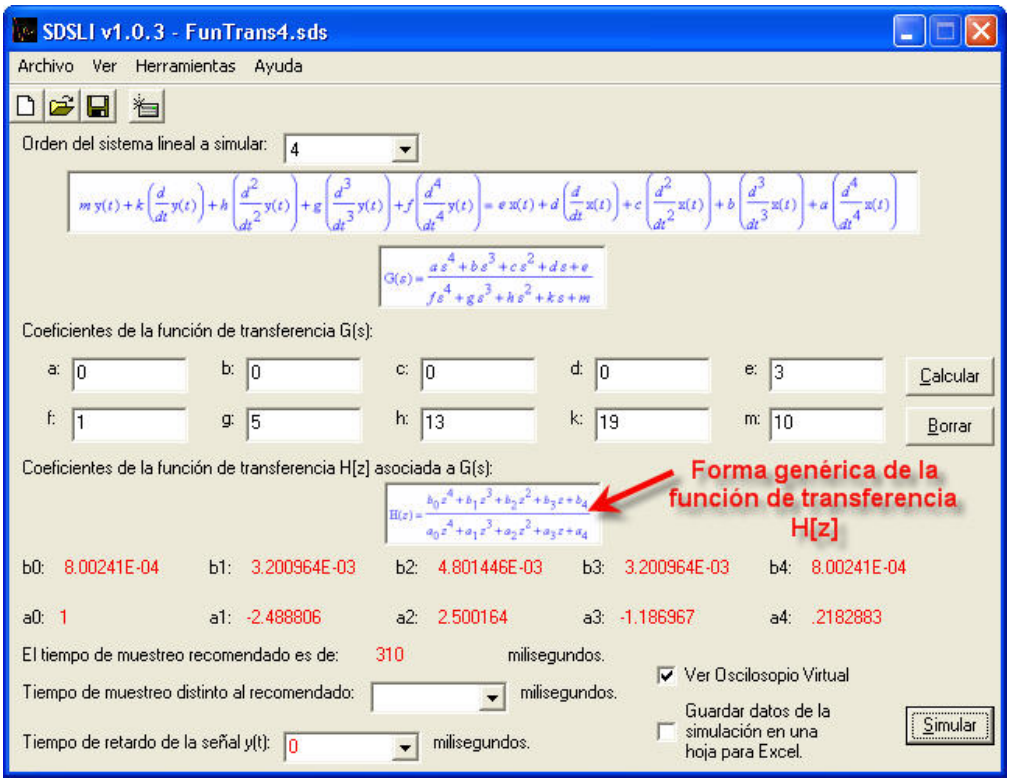


Figura 27. Función de transferencia H[z] en la ventana principal

3.1.11. Valores de los coeficientes de la función de transferencia H[z] y el tiempo de muestreo recomendado

La ventana principal posee esta sección, para informarle sobre los valores calculados correspondientes a la función de transferencia $H[z]$ que equivale a la función de transferencia $G(s)$ que desea simular.

Como se comentó con anterioridad, ya que el simulador es de tipo digital, además del cálculo de la función de transferencia $H[z]$, se requiere indicar el valor en milisegundos, del periodo de muestreo de la señal de entrada $x(t)$. Cuando presiona el botón **"Calcular"** de la ventana principal, se calcula un tiempo de muestreo recomendado, basado en un criterio empírico, que utiliza los valores de los polos de la función de transferencia $G(s)$.

Una vez obtenido éste valor, es posible calcular los coeficientes de la función de transferencia $H[z]$.

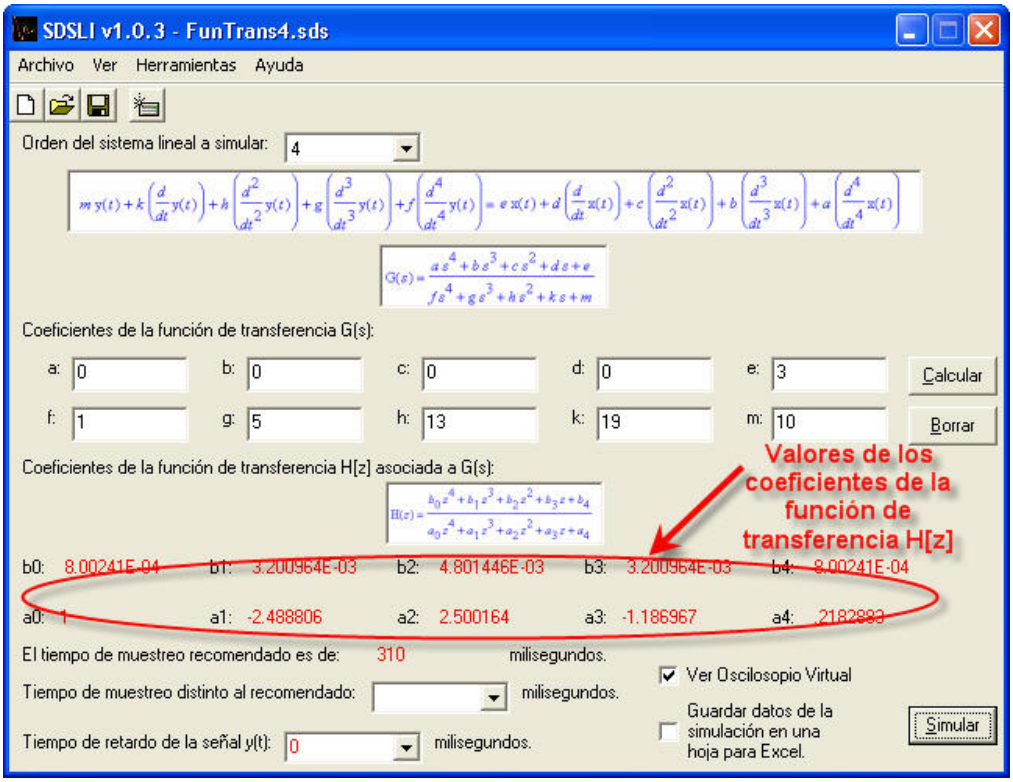


Figura 28. Valores de los coeficientes de la función de transferencia H[z]

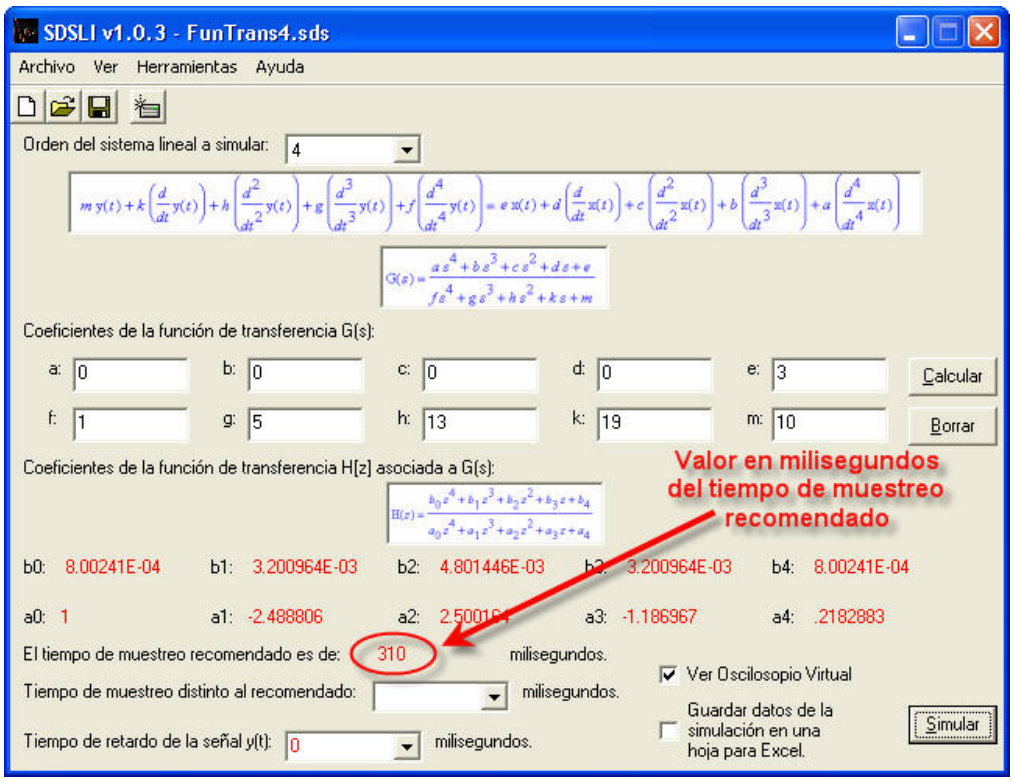


Figura 29. Valor en milisegundos del periodo de muestreo recomendado

3.1.12. El tiempo de muestreo distinto al recomendado

El tiempo de muestreo distinto al recomendado, le permite establecer un tiempo de muestreo de la señal de entrada $x(t)$, distinto al que el software calcula como tiempo de muestreo recomendado. Los valores posibles de tiempo de muestreo distinto al recomendado, son múltiplos de $10(ms)$ y es posible establecer tiempos de muestreo de hasta $60000(ms)$ ó $1(\min)$ como máximo.

Cabe resaltar que, cuando los coeficientes de la función de transferencia $H[z]$ se han calculado con el tiempo de muestreo recomendado éstos se encontrarán en color **ROJO** al igual que el tiempo de muestreo recomendado. Cuando se selecciona un tiempo de muestreo distinto al recomendado, los coeficientes de la función de transferencia $H[z]$ se recalcularán automáticamente y se indicarán en color **AZUL**, al igual que el tiempo de muestreo distinto al recomendado. Esta indicación con colores,

se realiza para que pueda saber fácilmente con qué tiempo de muestreo se han calculado los valores de la función de transferencia $H[z]$ presente en la ventana.

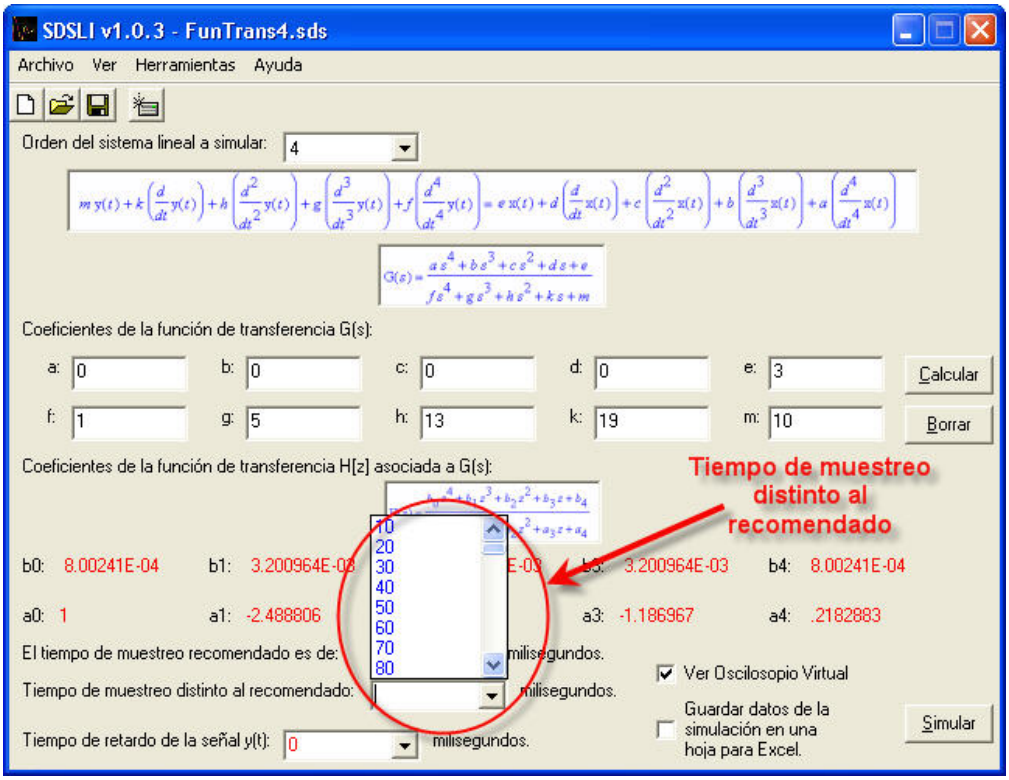


Figura 30. Valores en milisegundos de los tiempos de muestreo distintos al recomendado

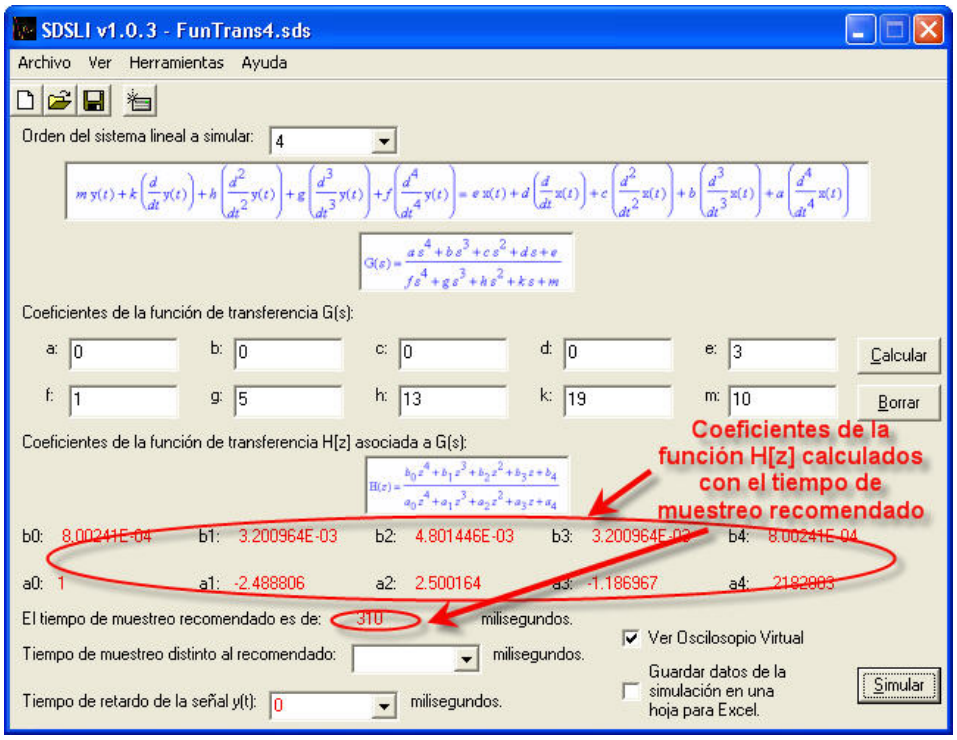


Figura 31. Función de transferencia H[z] (en color rojo) calculada con el tiempo de muestreo recomendado (también en color rojo)

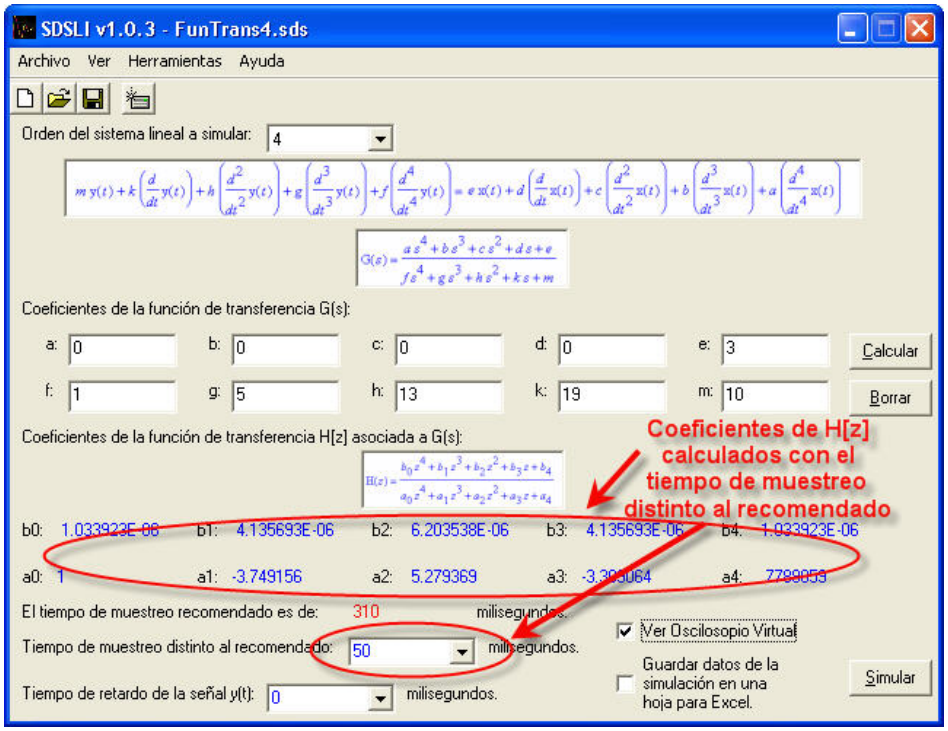


Figura 32. Función de transferencia H[z] (en color azul) calculada con un tiempo de muestreo distinto al recomendado (también en color azul)

3.1.13. El tiempo de retardo de la señal y(t)

El tiempo de retardo de la señal $y(t)$ es un tiempo en milisegundos el cual le permite definir el tiempo de retraso que presentará la señal de salida $y(t)$ respecto a la señal de entrada $x(t)$.

Los posibles valores de tiempo de retardo de la señal $y(t)$, van desde $0(ms)$ y hasta, 54 veces el tiempo de muestreo con el que se encuentre calculada la función de transferencia $H[z]$.

Al igual que en el tiempo de muestreo distinto al recomendado, cuando los coeficientes de la función de transferencia $H[z]$ se han calculado con el tiempo de muestreo recomendado éstos se encontrarán en color **ROJO** del mismo modo que el tiempo de retardo de la señal $y(t)$. Cuando se selecciona un tiempo de muestreo distinto al recomendado, los coeficientes de la función de transferencia $H[z]$ se recalcularán automáticamente y se indicarán en color **AZUL**, al igual que el tiempo de retardo de la señal $y(t)$. Esta indicación con colores, se realiza para que pueda saber fácilmente, con que tiempo de muestreo se han calculado los valores de la función de transferencia $H[z]$ y los posibles valores del tiempo de retardo de la señal $y(t)$ presentes en la ventana.

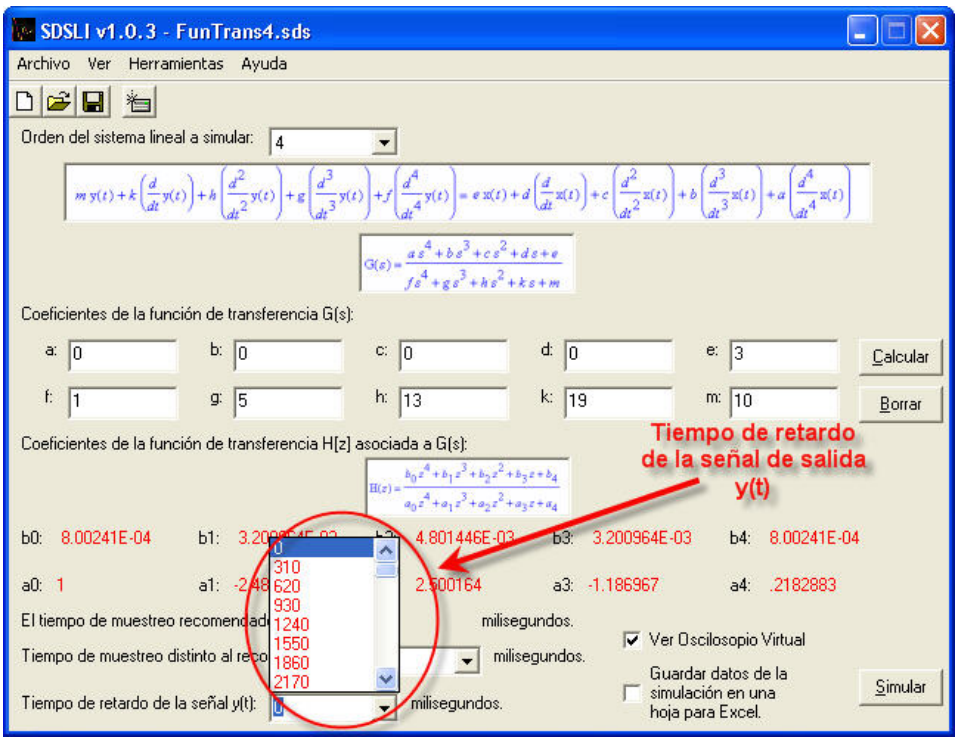


Figura 33. Valores en milisegundos de los tiempos de retardo de la señal $y(t)$

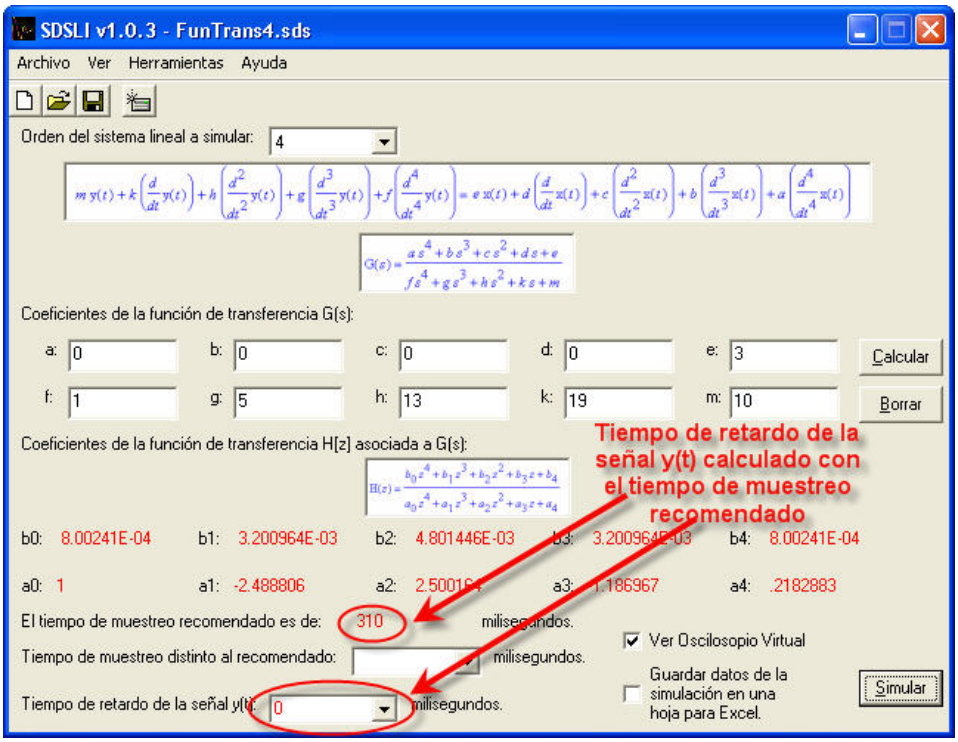


Figura 34. Valor del tiempo de retardo de la señal $y(t)$ (en color rojo) calculado con el tiempo de muestreo recomendado (también en color rojo)

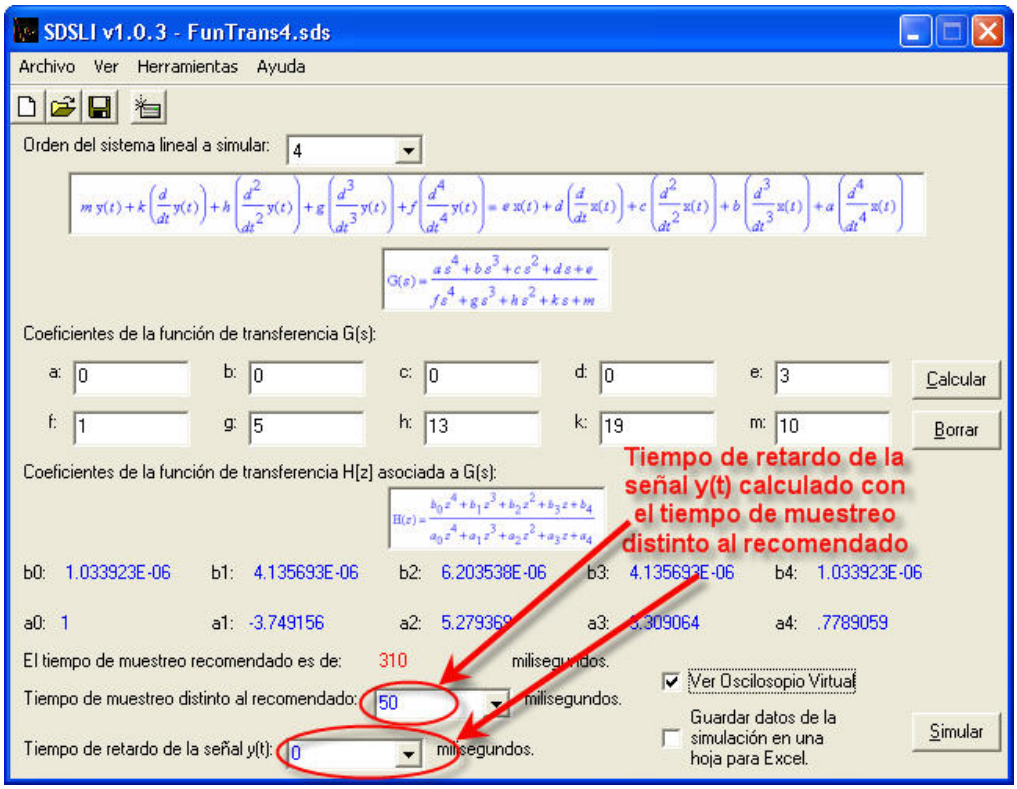


Figura 35. Valor del tiempo de retardo de la señal $y(t)$ (en color azul) calculado con el tiempo de muestreo recomendado (también en color azul)

3.1.14. Visualización del Osciloscopio virtual

Esta opción le permite ver en la pantalla de su PC, una ventana que hemos denominado **"Osciloscopio Virtual"** y que le proporciona información gráfica sobre la forma de las señales de entrada y salida del SDSLI. Cuando la simulación se inicia, esta pantalla aparece para que pueda apreciar las señales.

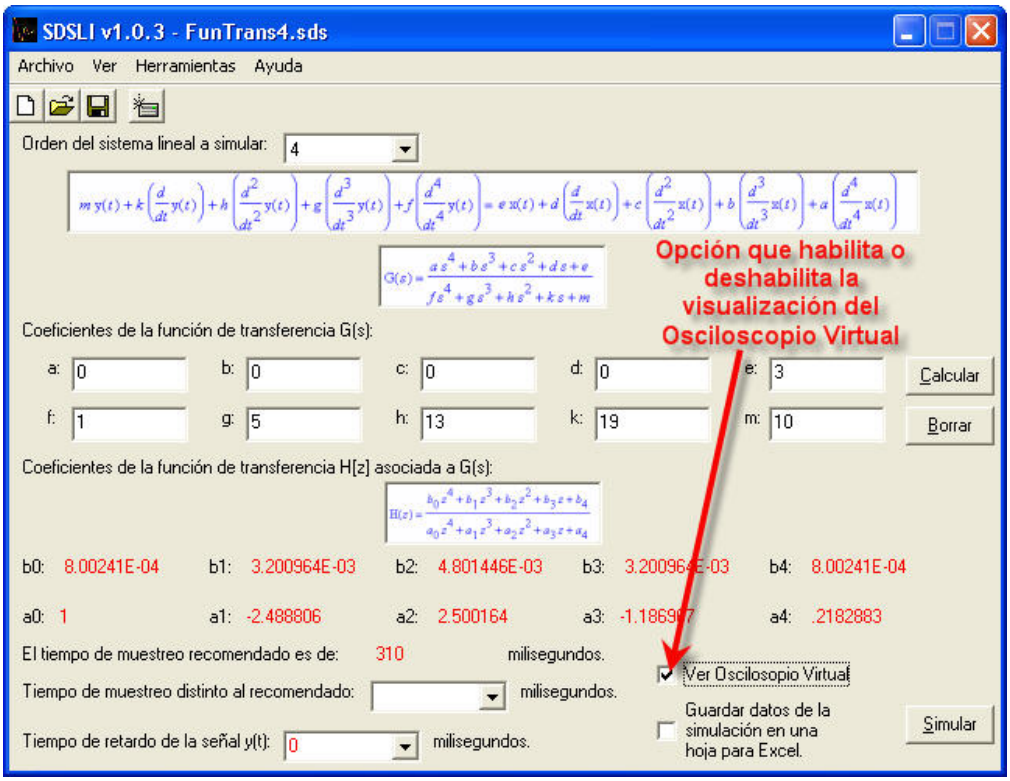


Figura 36. Casilla que permite activar o desactivar el Osciloscopio Virtual al iniciar una simulación

Si la casilla de verificación está marcada 

, significa que esta opción está activa y el osciloscopio virtual aparecerá en la pantalla de su PC al iniciar la simulación. De lo contrario, si la casilla de verificación está en blanco 

, al iniciar la simulación, NO verá el osciloscopio virtual y tendrá que recurrir a conectar un osciloscopio en los bornes de señal de entrada y señal de salida del hardware del SDSLI, para poder observar las señales de entrada y salida del simulador.

3.1.15. Almacenar datos de una simulación en una hoja para Excel

Algunas veces tendrá la necesidad de almacenar los datos numéricos de los valores de las señales de entrada y salida para poder realizar con ellos, análisis más detallados. Para este fin, se cuenta con la opción **"Guardar datos de la simulación en una hoja**

para Excel" y la cual, se encarga de almacenar los datos de la simulación que se está llevando acabo, en un archivo con extensión ***.CSV*** que es compatible con la hoja de cálculo EXCEL de Microsoft.

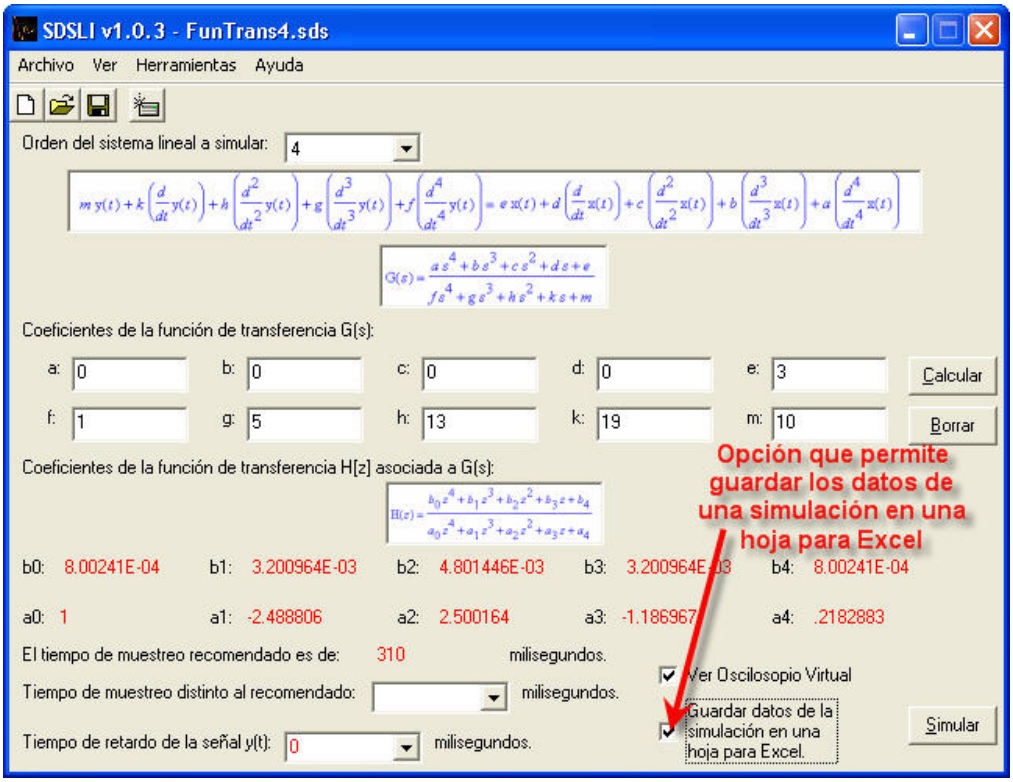
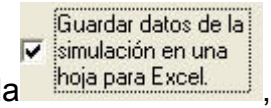
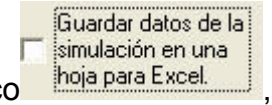


Figura 37. Casilla que permite activar o desactivar la opción que brinda la posibilidad de guardar los datos de una simulación en una hoja para Excel.

Si la casilla de verificación está marcada 

, significa que esta opción está activa y al iniciar la simulación se le pedirá que indique la ruta y el nombre de la hoja en la que desea almacenar los datos de la simulación. De lo contrario, si la casilla de verificación está en blanco 

, al iniciar la simulación, NO almacenará los datos de la simulación en una hoja para Excel.

Es importante mencionar que esta opción sólo está disponible si la casilla que permite visualizar el osciloscopio virtual está activa 

. Esto se debe a que, la opción que permite guardar los datos de una simulación en una hoja de Excel, GENERA ARCHIVOS DE TAMAÑO CONSIDERABLE MUY RÁPIDAMENTE y dependiendo del tiempo de muestreo indicado. Usted, al poder visualizar las señales de entrada y salida en el Osciloscopio Virtual, puede controlar el inicio y final de la simulación y con esto, determinar dónde inicia el almacenamiento de datos en la hoja de Excel y el momento en que esto termina, al finalizar o pausar la simulación en el osciloscopio virtual.

Se recomienda que utilice con cuidado esta opción para que no genere archivos demasiado grandes y difíciles de manejar.

3.1.16. El botón Simular

Este botón permite iniciar una simulación, una vez que los coeficientes de la función de transferencia $H[z]$ han sido calculados.

Si el usuario ha desactivado la casilla de visualización del Osciloscopio Virtual, este botón también permite detener una simulación que se encuentre ejecutando en el hardware del SDSLI. Es decir, el botón **"Simular"** puede iniciar o detener una simulación.

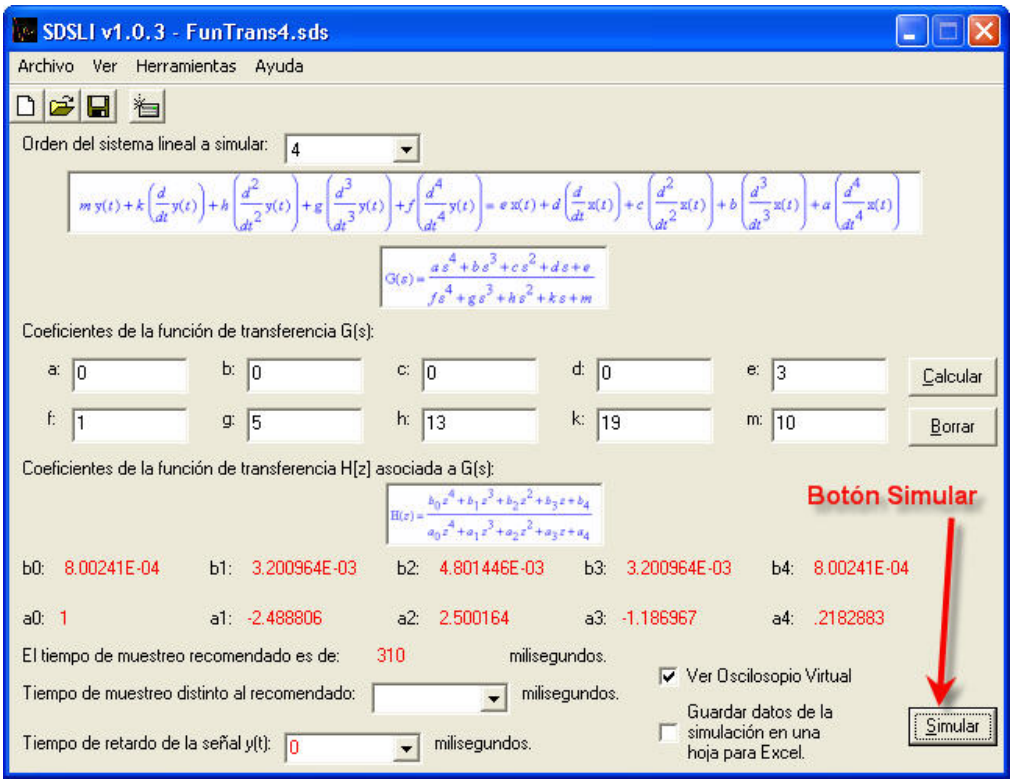


Figura 38. El botón Simular

3.2. LA VENTANA SIMULACIÓN DE FUNCIÓN DE TRANSFERENCIA H[Z]

La ventana Simulación de función de transferencia $H[z]$ del software del SDSLI, al igual que la ventana principal, permite al usuario acceder a algunas de las opciones y herramientas que se requieren para desarrollar una simulación con el hardware del SDSLI. Esta ventana es muy parecida a la ventana principal, así que, si ya conoce la ventana principal, le será muy fácil reconocer y utilizar las opciones y herramientas que ofrece esta ventana.

La diferencia más importante entre la ventana principal y esta ventana es que, posibilita al usuario desarrollar simulaciones directamente con funciones de transferencia de tipo digital $H[z]$ y no con funciones de transferencia de tipo analógico $G(s)$.

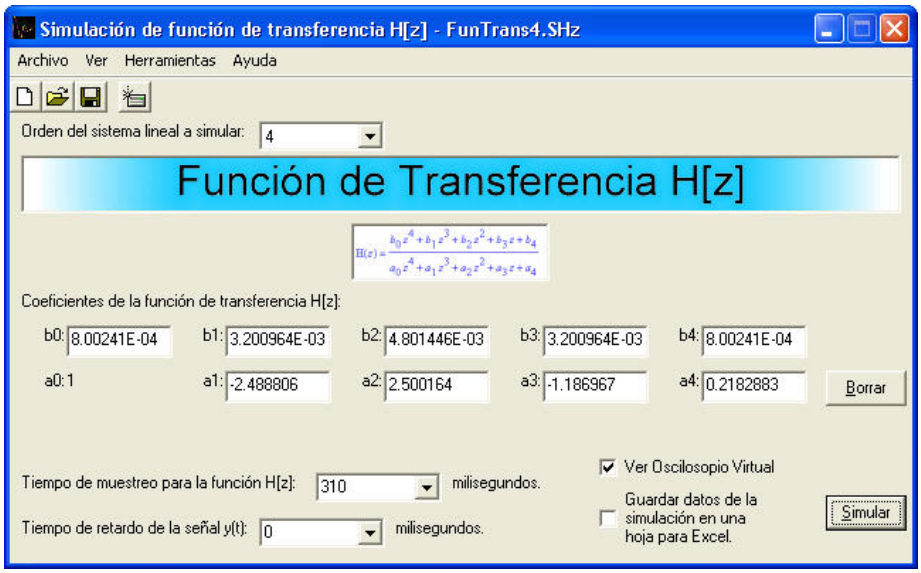


Figura 39. Ventana Simulación de función de transferencia H[z] del software SDSLI

En la figura 39 se muestra la ventana Simulación de función de transferencia $H[z]$ del software del SDSLI a través de la cual se ordena al hardware del SDSLI realizar simulaciones directamente, con funciones de transferencia $H[z]$ que se introducen en esta ventana.

3.2.1. Las barras de la ventana principal

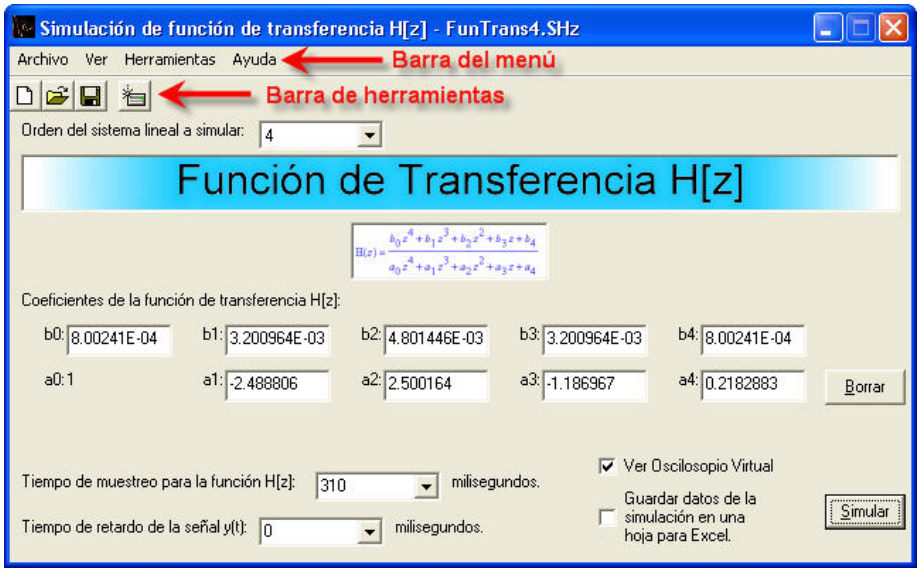


Figura 40. La barra del menú y la barra de herramientas

La barra del menú, contiene algunas opciones que le permiten guardar y abrir datos de una simulación y configurar el hardware del SDSLI para que funcione en modo autónomo, entre otras.

La barra de herramientas, le provee de un acceso rápido a las funciones más utilizadas del software como son: *abrir un nuevo archivo*, *abrir un archivo previamente almacenado en disco*, *guardar un archivo en disco* y *reinicializar el hardware del SDSLI*.

3.2.2. El menú Archivo

Figura 41. El menú Archivo de la ventana Simulación de función de transferencia H[z]

3.2.2.1. Las opciones del menú Archivo

1. **Nuevo:** Permite crear una nueva simulación de una función de transferencia $H[z]$.
2. **Abrir…:** Permite abrir un archivo ***.SHz*** previamente almacenado en disco.

3. **Guardar como…:** Permite almacenar en disco, por primera vez, los datos de una simulación que se muestran en la ventana. Permite establecer el nombre del archivo, en el que se almacenarán los datos de la simulación.
4. **Guardar:** Permite almacenar los datos de una simulación en un archivo previamente almacenado en disco con el comando **Guardar como…**.
5. **Regresar:** Permite regresar a la ventana principal de la aplicación.

3.2.3. El menú Ver

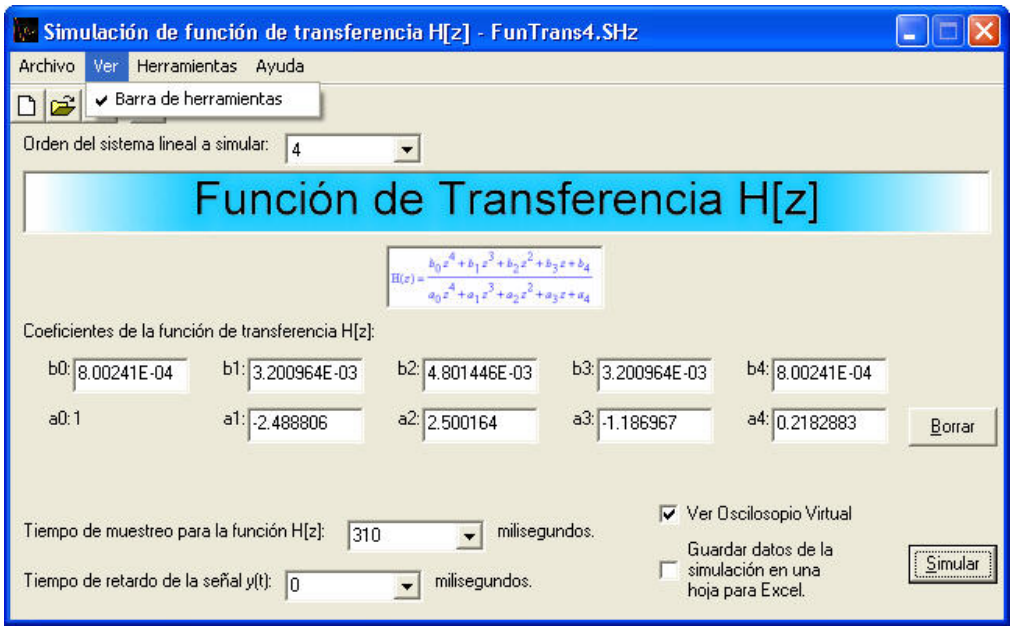


Figura 42. El menú Ver de la ventana Simulación de función de transferencia H[z]

3.2.3.1. Las opciones del menú Ver

1. **Barra de herramientas:** Permite hacer aparecer o desaparecer la barra de herramientas de la ventana.

3.2.4. El menú Herramientas

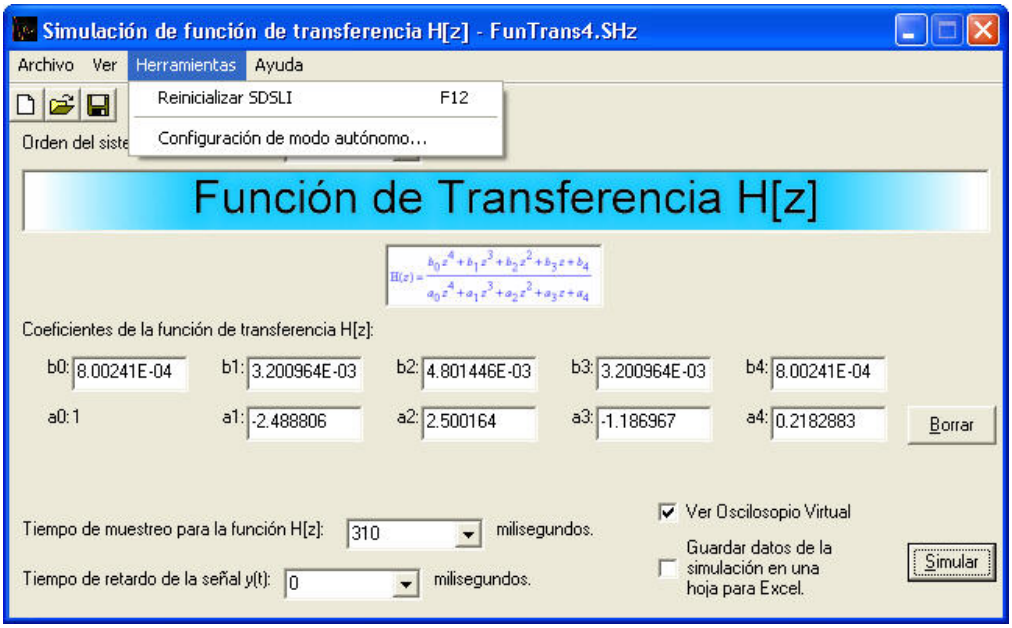


Figura 43. El menú Herramientas de la ventana Simulación de función de transferencia H[z]

3.2.4.1. Las opciones del menú Herramientas

1. **Reinicializar SDSLI:** Al igual que en la ventana principal, esta opción del mismo modo que el botón Reinicializar SDSLI de la barra de herramientas, permite realistar el hardware del SDSLI, haciendo que el led **"LISTO"** del hardware, comience a parpadear y preparando al hardware para recibir órdenes.
2. **Configuración de modo autónomo…:** Inicia el asistente para configurar la operación del hardware del SDSLI en modo autónomo.

3.2.5. El menú Ayuda

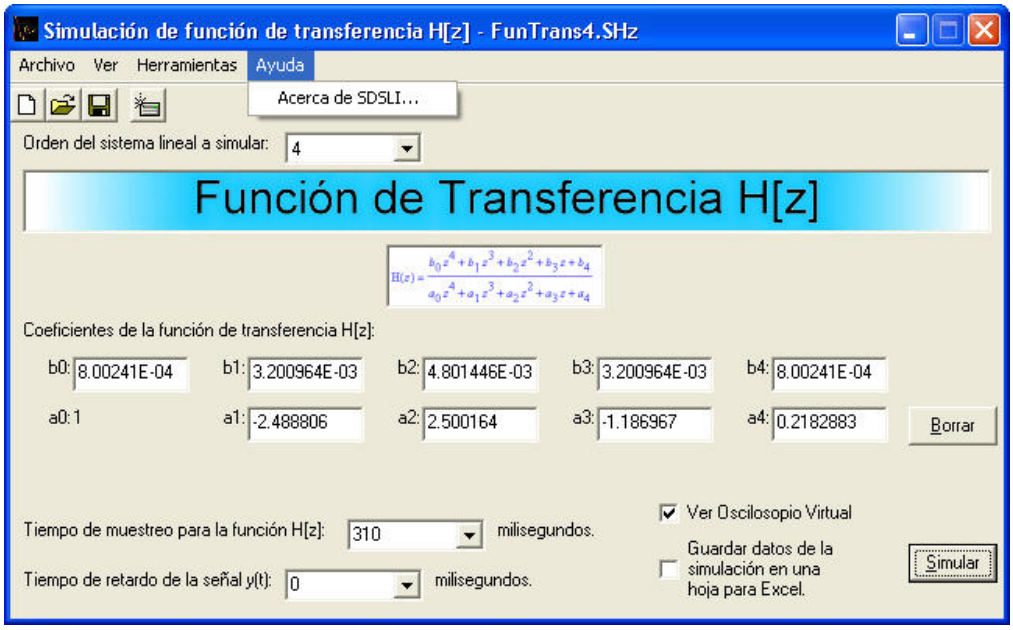


Figura 44. El menú Ayuda de la ventana Simulación de función de transferencia H[z]

3.2.5.1. Las opciones del menú Ayuda

1. **Acerca de SDSLI...:** Abre una ventana con información relativa a la versión del software del SDSLI y los nombres de los autores que participaron en el desarrollo del SDSLI.

3.2.6. Orden del sistema lineal a simular

En esta parte de la ventana, tiene la posibilidad de indicar el orden del sistema lineal e invariante que desea simular. Las posibilidades de orden que usted puede seleccionar van desde el primer y hasta el cuarto orden.

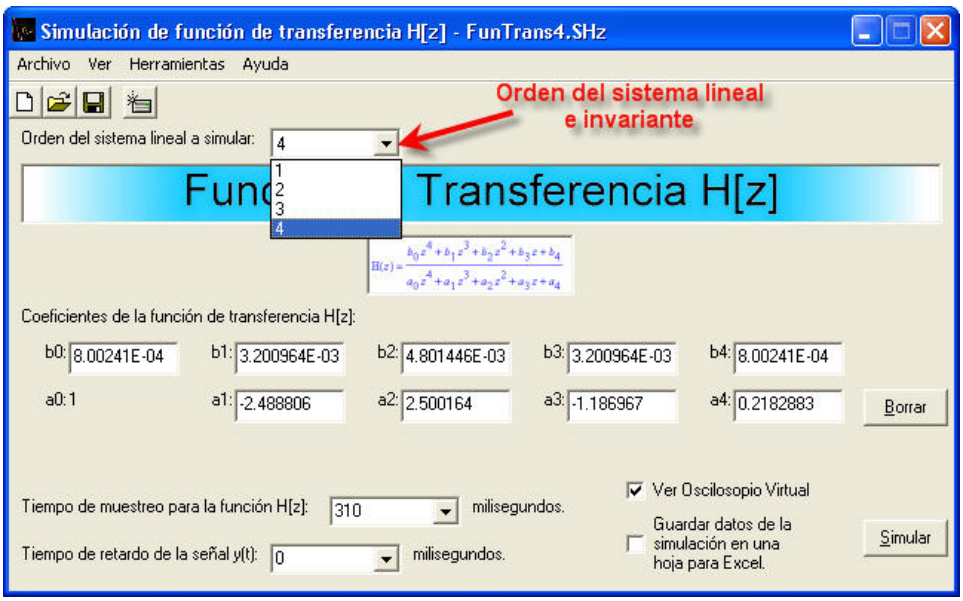


Figura 45. Orden del sistema lineal en la ventana Simulación de función de transferencia H[z]

3.2.7. La función de transferencia H[z]

En esta parte de la ventana, se muestra la forma genérica de la función de transferencia $H[z]$ que se va a simular. Nuevamente, esta función muestra la posición de los coeficientes dentro de la función de transferencia $H[z]$.

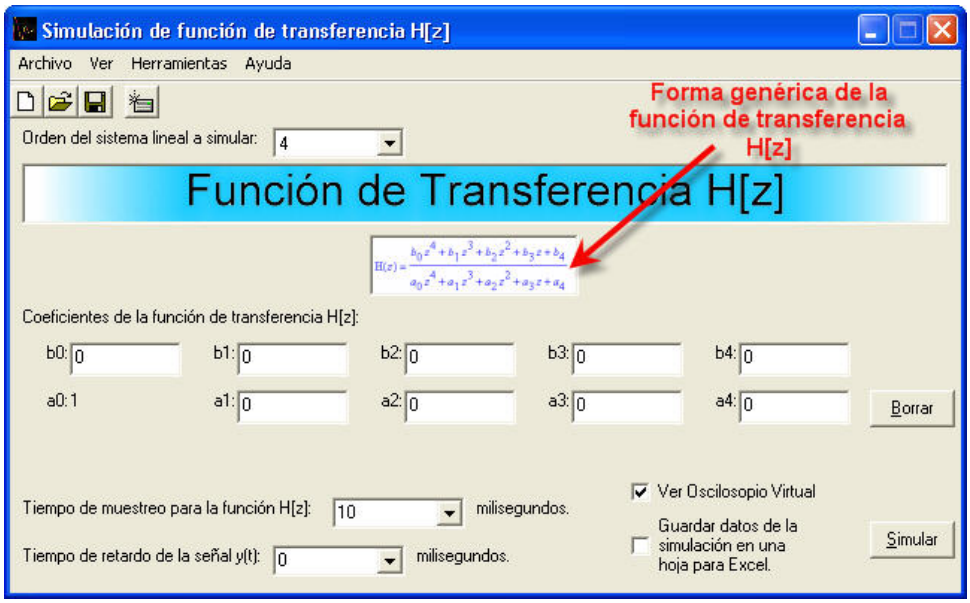


Figura 46. Función de transferencia H[z] en la ventana Simulación de función de transferencia H[z]

3.2.8. Los coeficientes de la función de transferencia H[z]

Esta ventana del software del SDSLI, cuenta con esta sección para permitirle introducir los valores de los coeficientes que corresponden a la función de transferencia $H[z]$ que desea simular. Nótese que los nombres de los coeficientes coinciden con los nombres mostrados en la función de transferencia.

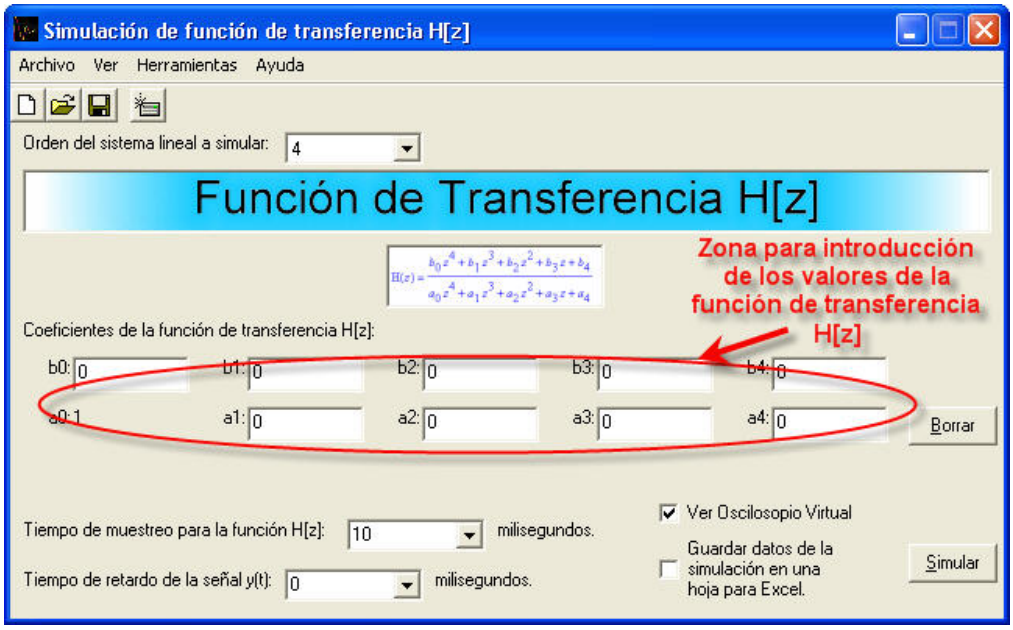


Figura 47. Zona de la ventana Simulación de función de transferencia H[z], para la introducción de los coeficientes del sistema lineal e invariante a simular

3.2.9. El botón borrar

El botón **"Borrar"**, le permite eliminar TODOS los coeficientes introducidos, estableciendo una configuración por defecto para éstos.

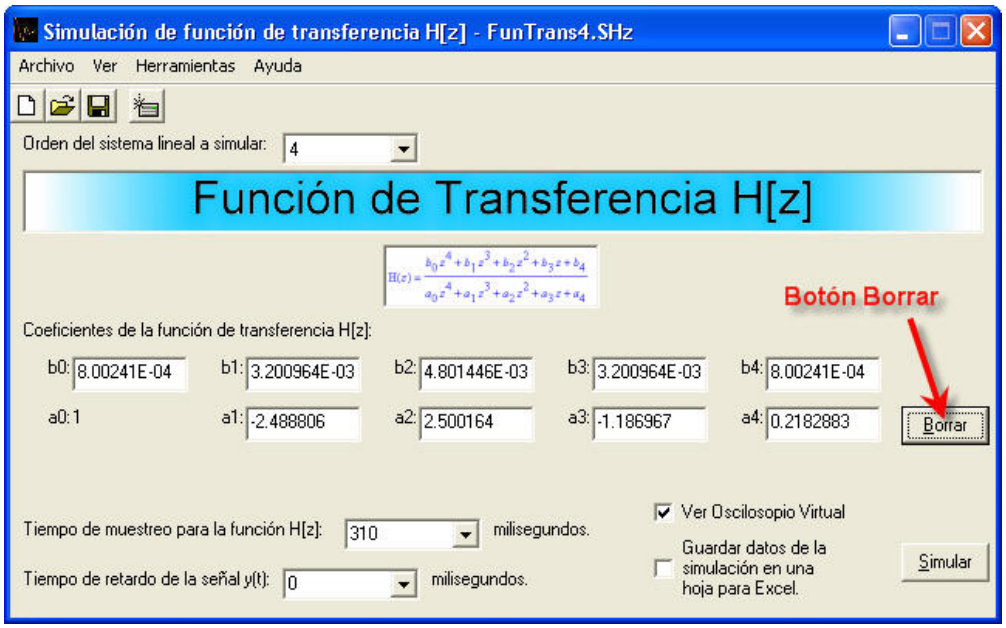


Figura 48. El botón Borrar de la ventana Simulación de función de transferencia H[z]

3.2.10. El tiempo de muestreo para la función H[z]

El tiempo de muestreo para la función $H[z]$, le permite establecer un tiempo de muestreo para la señal de entrada $x(t)$. Los valores posibles de tiempo de muestreo para la función $H[z]$, son múltiplos de $10(ms)$ y es posible establecer tiempos de muestreo de hasta $60000(ms)$ ó $1(\min)$ como máximo. Recuerde que el tiempo de muestreo que usted indique en esta casilla, debe coincidir con el tiempo de muestreo con el cual fueron calculados los coeficientes de la función de transferencia $H[z]$ que va a simular.

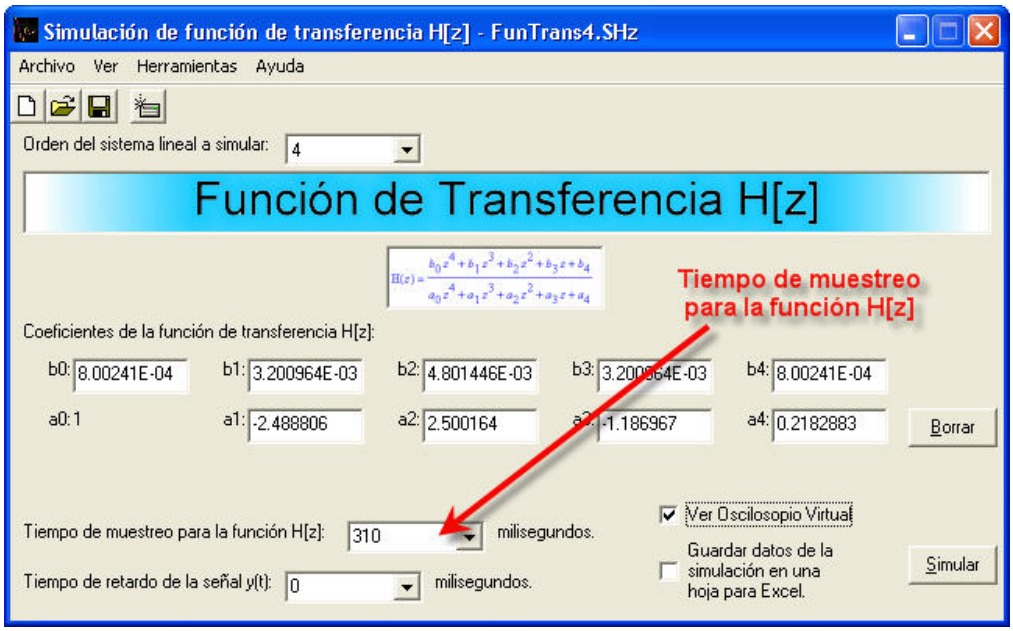


Figura 49. Valor en milisegundos del tiempo de muestreo para la función de transferencia H[z]

3.2.11. El tiempo de retardo de la señal y(t)

El tiempo de retardo de la señal $y(t)$ es un tiempo en milisegundos el cual le permite definir el tiempo de retraso que presentará la señal de salida $y(t)$ respecto a la señal de entrada $x(t)$.

Los posibles valores de tiempo de retardo de la señal $y(t)$, van desde $0(ms)$ y hasta, 54 veces el tiempo de muestreo que se indique para la función de transferencia $H[z]$.

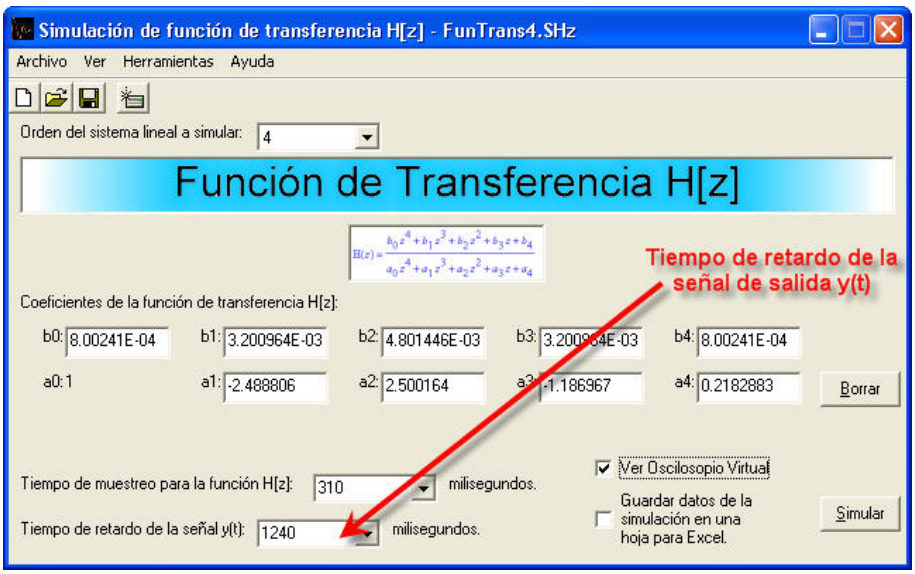


Figura 50. Valor en milisegundos del tiempo de retardo de la señal $y(t)$

3.2.12. Visualización del Osciloscopio virtual

Esta opción, al igual que en la ventana principal, le permite ver en la pantalla de su PC el **"Osciloscopio Virtual"** que le proporciona información gráfica sobre la forma de las señales de entrada y salida del SDSLI. Cuando la simulación se inicia, esta pantalla aparece para que pueda apreciar las señales.

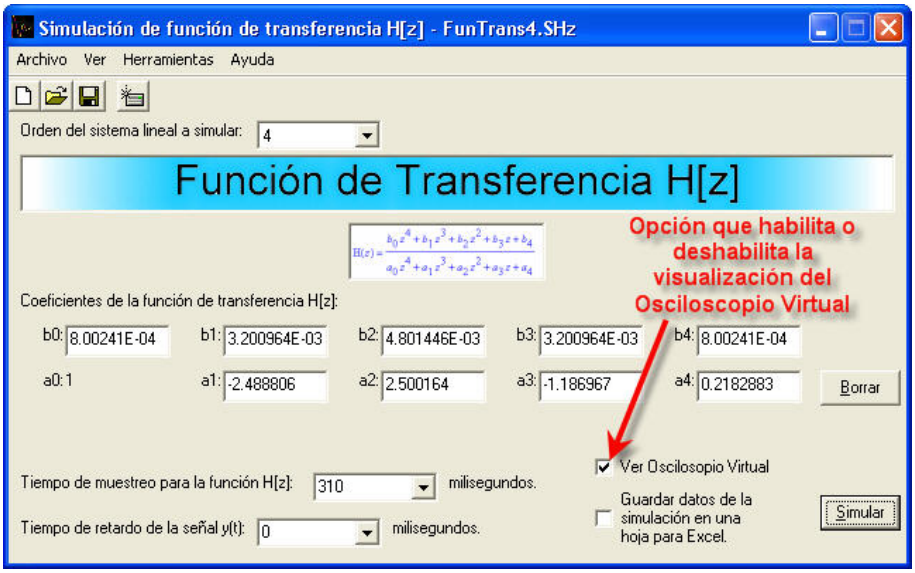


Figura 51. Casilla que permite activar o desactivar el Osciloscopio Virtual al iniciar una simulación

Si la casilla de verificación está marcada 

, significa que esta opción está activa y el Osciloscopio Virtual aparecerá en la pantalla de su PC al iniciar la simulación. De lo contrario, si la casilla de verificación está en blanco 

, al iniciar la simulación, NO verá el Osciloscopio Virtual.

3.2.13. Almacenar datos de una simulación en una hoja para Excel

De la misma manera que sucede en la ventana principal, algunas veces tendrá la necesidad de almacenar los datos numéricos de los valores de las señales de entrada y salida para poder realizar con ellos, análisis más detallados. Para este fin, se cuenta con la opción **"Guardar datos de la simulación en una hoja para Excel"** la cual, almacena los datos de la simulación que se está llevando acabo, en un archivo con extensión ***.CSV*** que es compatible con la hoja de cálculo EXCEL de Microsoft.

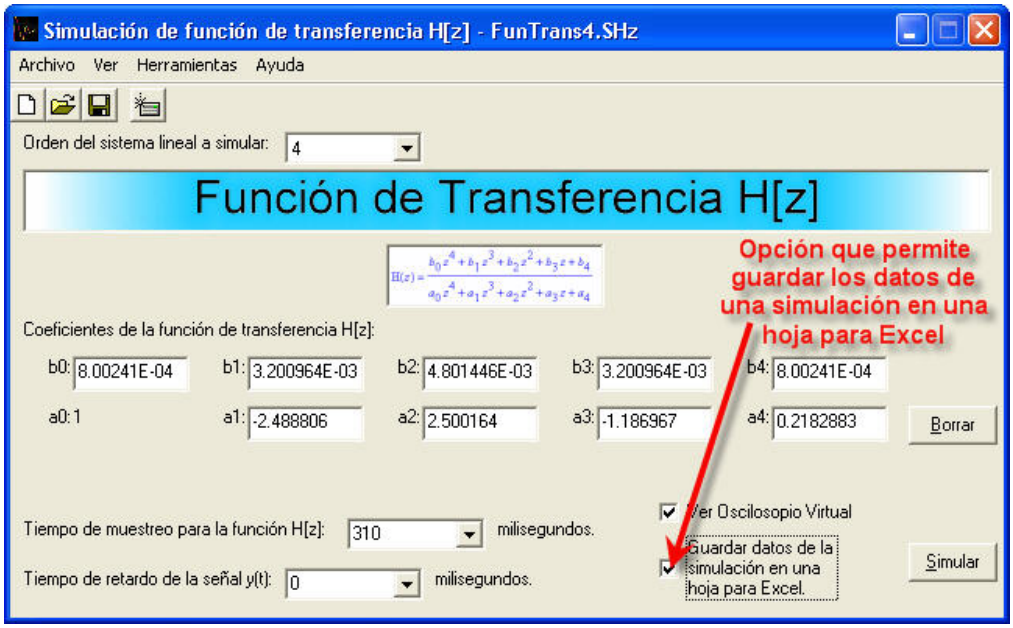
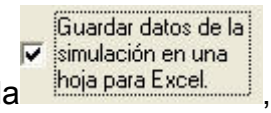
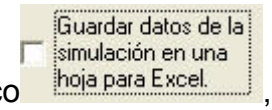


Figura 52. Casilla que permite activar o desactivar la opción que brinda la posibilidad de guardar los datos de una simulación en una hoja para Excel.

Si la casilla de verificación está marcada 

, significa que esta opción está activa y al iniciar la simulación, se le pedirá que indique la ruta y el nombre de la hoja en la que desea almacenar los datos de la simulación. De lo contrario, si la casilla de verificación está en blanco 

, al iniciar la simulación, NO almacenará los datos de la simulación en una hoja para Excel.

Al igual que en la ventana principal, esta opción sólo está disponible si la casilla que permite visualizar el Osciloscopio Virtual, está activa 

.

Nuevamente, se recomienda que utilice con cuidado esta opción para que no genere archivos demasiado grandes y difíciles de manejar.

3.2.14. El botón Simular

Este botón permite iniciar una simulación, una vez que los coeficientes de la función de transferencia $H[z]$ han sido introducidos en la ventana.

Si el usuario ha desactivado la casilla de visualización del Osciloscopio Virtual, este botón permite también detener una simulación que se encuentre ejecutando en el hardware del SDSLI.

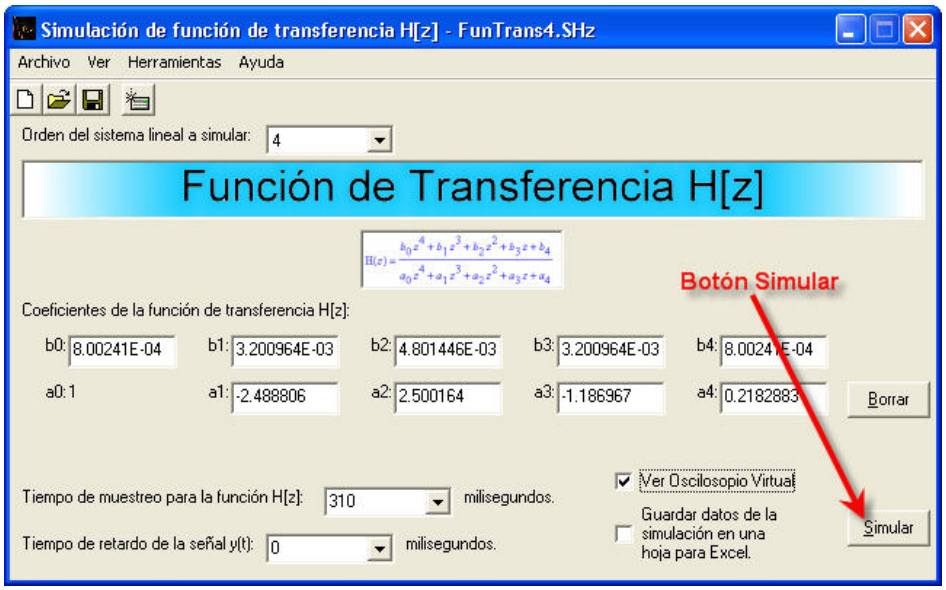


Figura 53. El botón Simular

3.3. EL OSCILOSCOPIO VIRTUAL

El Osciloscopio Virtual es la herramienta que le permite observar gráficamente las señales de entrada y salida al hardware del SDSLI mientras se encuentra una simulación activa.

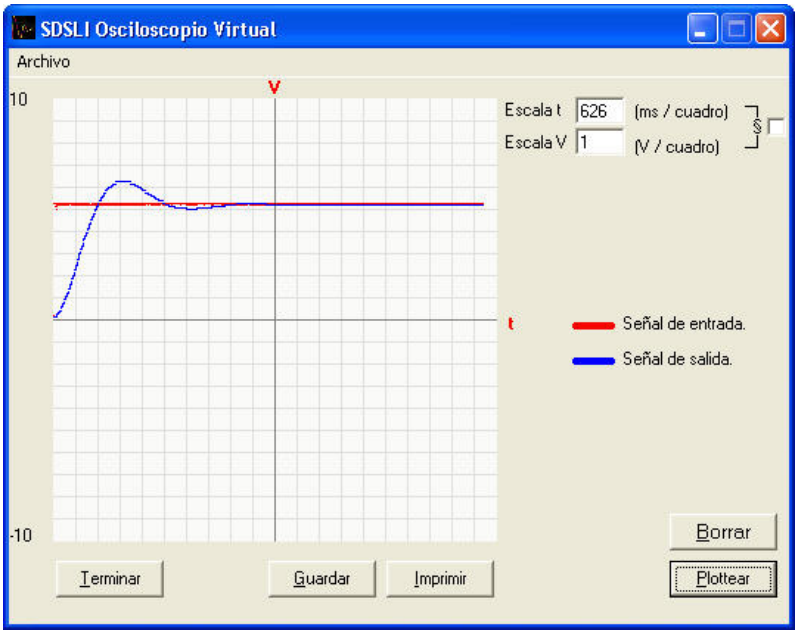


Figura 54. El Osciloscopio Virtual

3.3.1. La barra del Osciloscopio Virtual

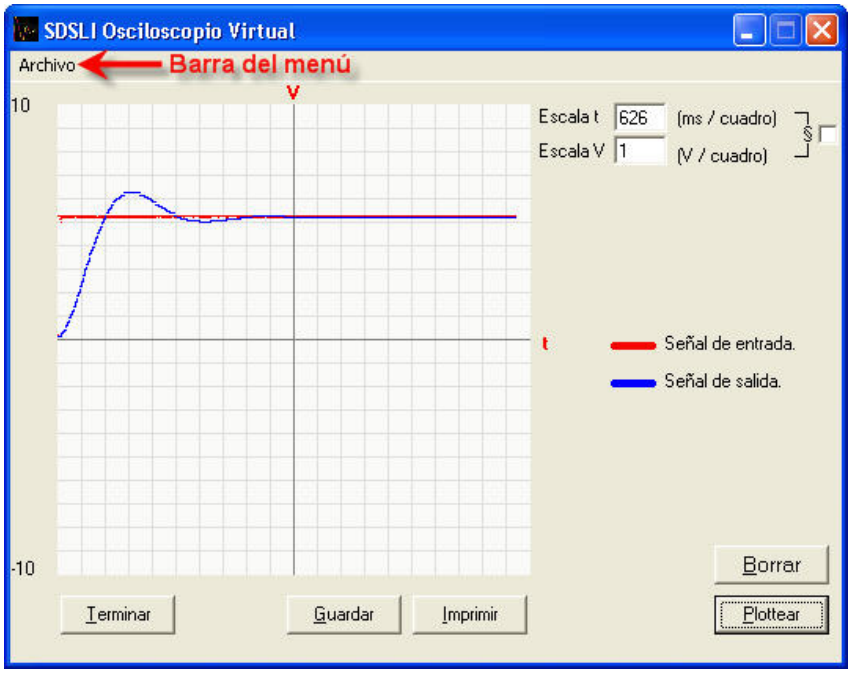


Figura 55. La barra del menú

La barra del menú, contiene algunas opciones que le permiten guardar una gráfica correspondiente a una simulación, configurar la impresora, imprimir una gráfica, entre otras. A continuación se detallan sus opciones.

3.3.2. El menú Archivo

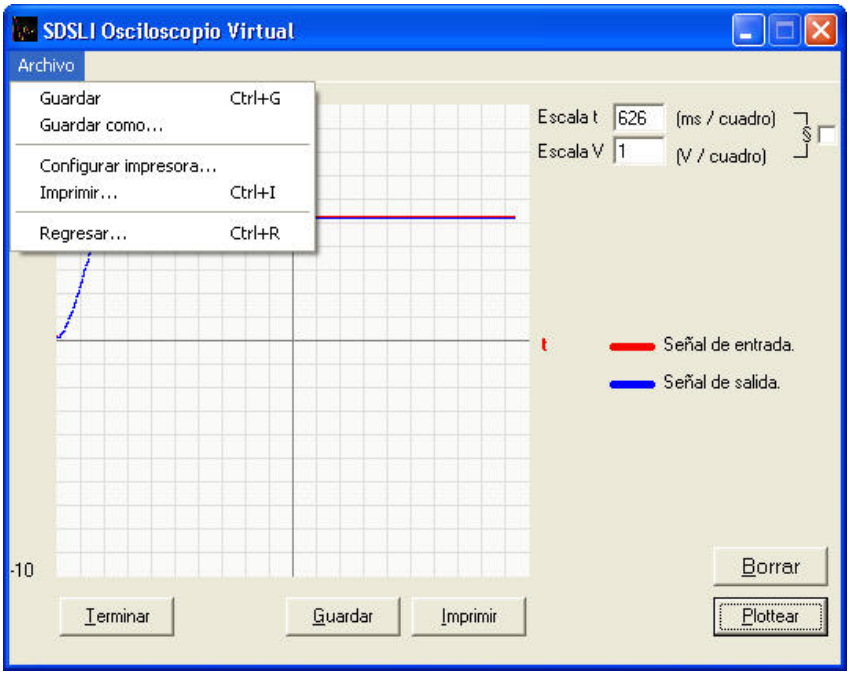


Figura 56. El menú Archivo del Osciloscopio Virtual

3.3.2.1. Las opciones del menú Archivo

1. **Guardar como…:** Permite almacenar en disco, por primera vez, las gráficas correspondientes a una simulación que se muestran en la ventana. Permite establecer el nombre del archivo, en el que se almacenarán los datos de la simulación. Se generan 2 archivos, un archivo ***.BMP*** que contiene la imagen de la gráfica y un archivo ***.TXT*** que contiene datos importantes sobre la gráfica almacenada en el archivo ***.BMP***.
2. **Guardar:** Permite almacenar una gráfica correspondiente a una simulación en un archivo previamente almacenado en disco con el comando **Guardar como…**.
3. **Configurar impresora…:** Permite establecer las configuraciones necesarias de la impresora.

4. **Imprimir…:** Permite imprimir la gráfica presente en la ventana del Osciloscopio Virtual correspondiente a una simulación, en la impresora activa o predeterminada.
5. **Regresar…:** Permite retornar a la ventana principal.

3.3.3. La ventana de despliegue de las gráficas

En la ventana de despliegue de las gráficas, se presentan las señales de entrada y salida casi al mismo tiempo en que éstas se presentan en el hardware del simulador.

En esta ventana, podrá medir, al igual que en un osciloscopio, los valores de voltaje de las señales y los tiempos en los que éstas se presentan para de este modo, tener conocimiento del funcionamiento del sistema lineal que se está simulando.

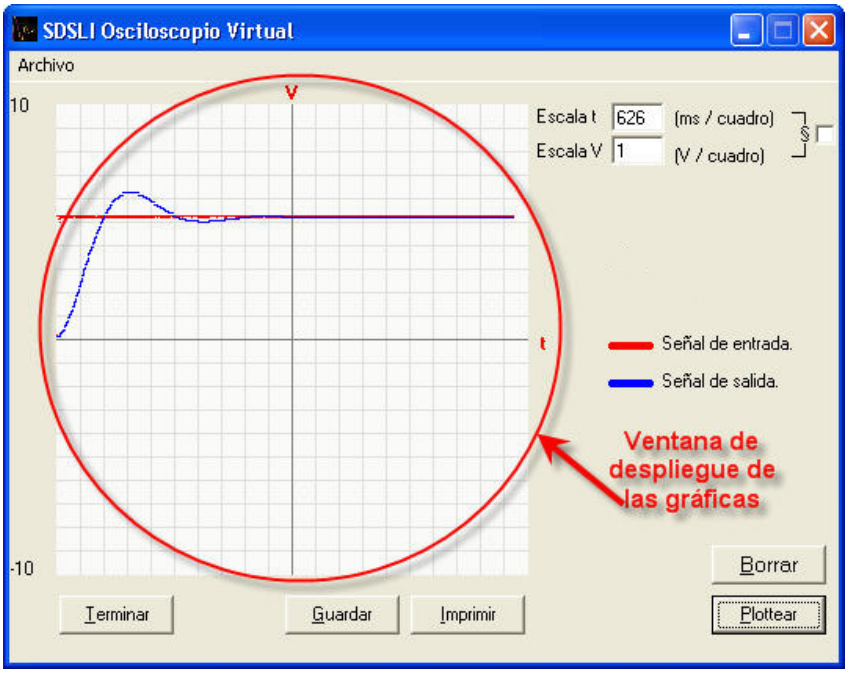


Figura 57. La ventana de despliegue de las señales de entrada y salida del SDSLI

El eje de las abscisas representa al tiempo en milisegundos y el eje de las ordenadas, representa al valor del voltaje en volts de la magnitud de las señales.

La señal en color **ROJO** representa a la señal de entrada $x(t)$ que se introduce al SDSLI y la señal en color **AZUL**, representa a la respuesta de salida del sistema lineal, es decir, la señal de salida $y(t)$.

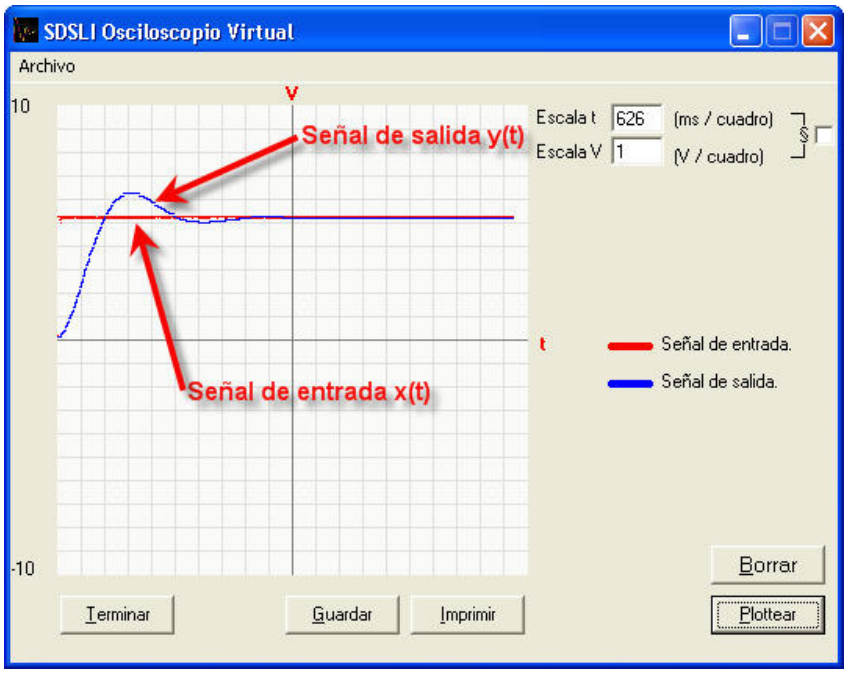


Figura 58. Las señales de entrada y salida del SDSLI

3.3.4. Los valores de ajuste de escala de tiempo y voltaje

De la misma manera que sucede en un osciloscopio, el Osciloscopio Virtual tiene capacidad de permitir al usuario ajustar la escala de tiempo y voltaje para poder apreciar en distintos tamaños la señal que se presenta en la ventana de visualización.
El valor que corresponde a **"Escala t"** es el valor de escala de tiempo que representa el número de milisegundos por cuadro en la ventana de visualización de las gráficas.

El valor **"Escala V"** es el valor de escala de voltaje que representa el número de volts por cuadro en la ventana de visualización de las gráficas.

Es importante hacerle saber que, cuando cambia algún valor de escala, éste no se mostrará de inmediato, sino se mostrará hasta el siguiente barrido de la pantalla de visualización, en donde podrá ver las señales en la escala que haya indicado con anterioridad.

Si usted lo desea, puede hacer que tanto el valor de escala de voltaje como el valor de escala de tiempo sean iguales, con solo marcar la casilla de verificación que se encuentra delante de ambas escalas. Al marcar esta casilla, el valor de escala de voltaje tomará el valor presente de la escala de tiempo y viceversa.

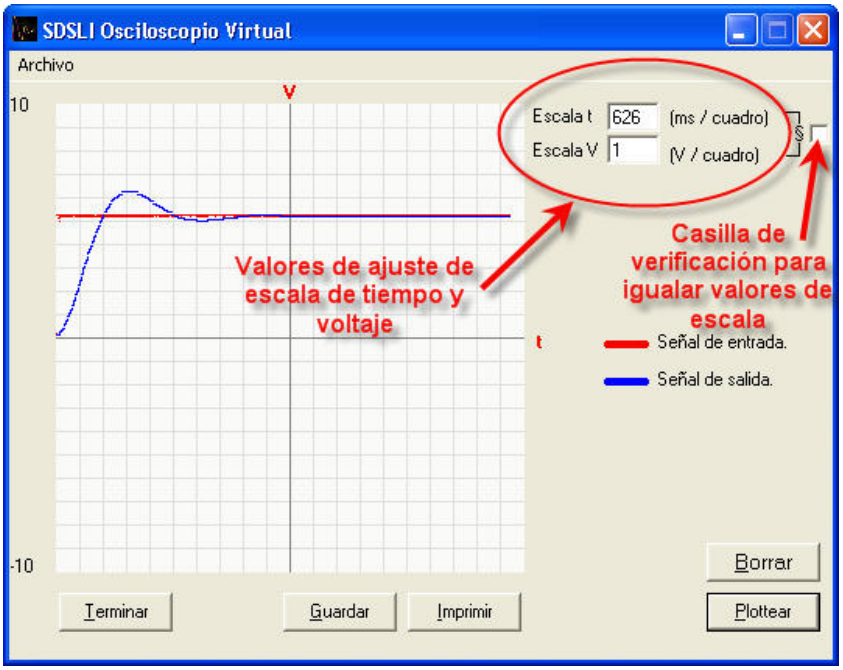


Figura 59. Los valores de ajuste de escala de tiempo y voltaje y la casilla de verificación para igualar estos valores

3.3.5. Valor de tiempo y voltaje en un punto determinado de una curva

Otra de las posibilidades que ofrece el Osciloscopio Virtual, es la de poder conocer las coordenadas aproximados de un punto de una curva en una gráfica presente en la ventana. Para esto, basta posicionar la flecha del ratón en el punto que se desea

conocer sus coordenadas, y verá del lado derecho de la ventana del Osciloscopio Virtual, los valores del tiempo en milisegundos y del voltaje en volts de ese punto en particular.

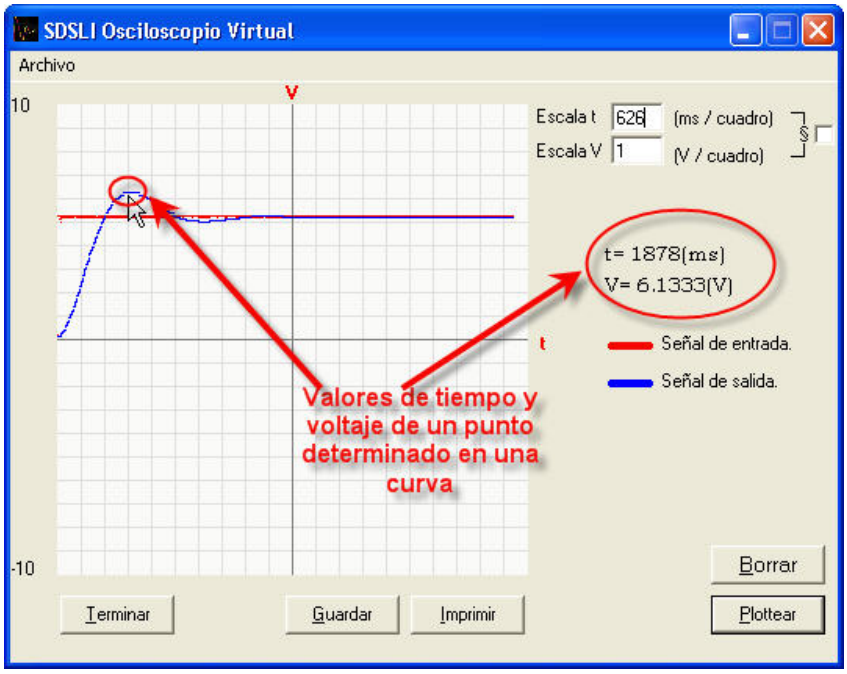


Figura 60. Los valores de tiempo y voltaje de un punto particular de una curva

3.3.6. El botón Pausar y Plottear

Uno de los botones más usados en esta ventana es precisamente este botón, el cual tiene una doble función: Pausar o detener momentáneamente la graficación de las señales sin detener la simulación y la otra función es la de reanudar la graficación de las señales. Inicialmente el botón dice **"Pausar"**, pero cuando se presiona, SE DETIENE LA GRAFICACIÓN DE LA SIMULACIÓN, PERO NO LA SIMULACIÓN EN EL HARDWARE DEL SDSLI y la etiqueta de este botón cambia a **"Plottear"**. Si se presiona el botón cuando se encuentra etiquetado de la última manera, se reanudará la graficación de las señales y el botón cambiará su etiqueta nuevamente a **"Pausar"**.

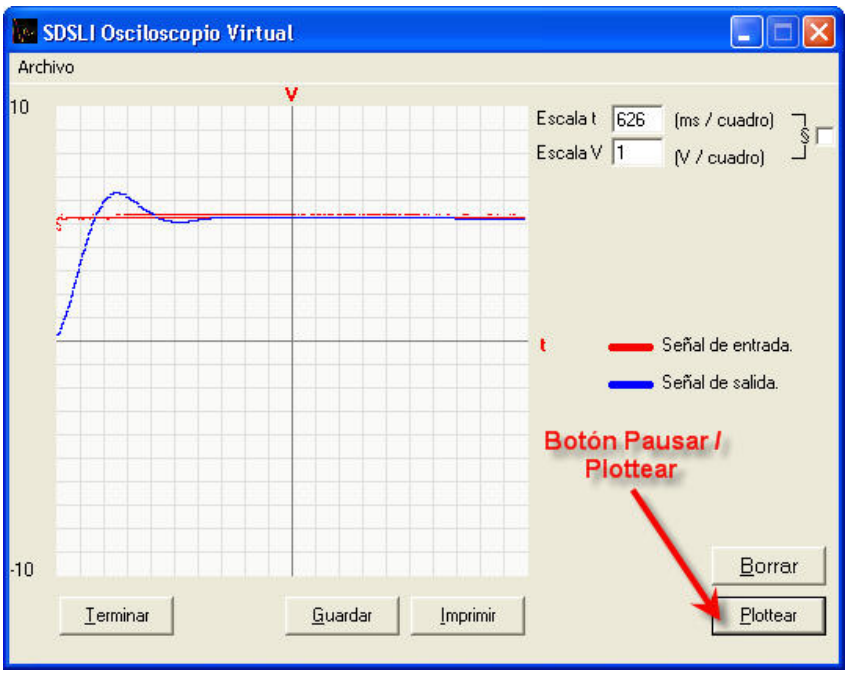


Figura 61. El botón Pausar / Plottear del Osciloscopio Virtual

3.3.7. El botón Borrar

Este botón permite borrar el contenido de la ventana de visualización de gráficas, reiniciando el barrido de izquierda a derecha.

El botón **"Borrar"** sólo se activa cuando el botón **"Plottear"** está activo.

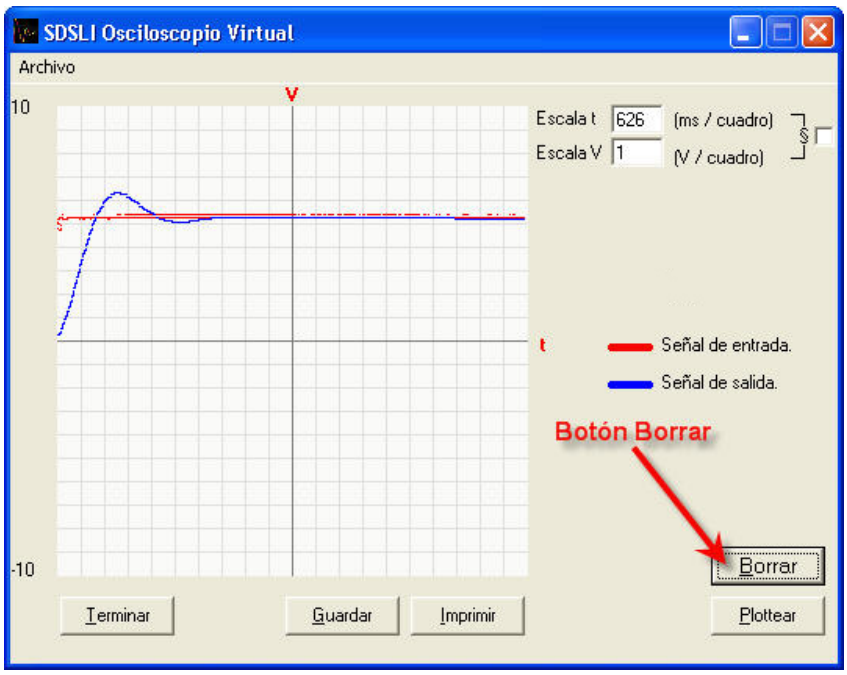


Figura 62. El botón Borrar del Osciloscopio Virtual

3.3.8. El botón Guardar

Al igual que la opción **"Guardar"** del menú **"Archivo"** de esta ventana, el botón **"Guardar"** permite almacenar en disco una imagen de las gráficas presentes en pantalla, en un archivo cuya extensión sea ***.BMP*** y generar la información correspondiente a estas gráficas en un archivo ***.TXT***.

Al igual que el botón **"Borrar"**, este botón también se activa si el botón **"Plottear"** se encuentra activo.

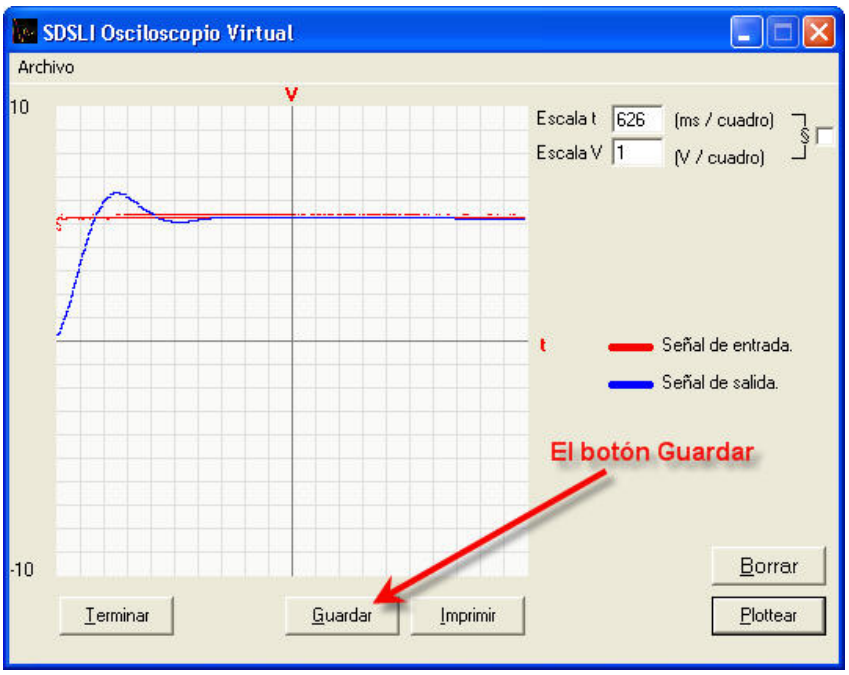


Figura 63. El botón Guardar del Osciloscopio Virtual

3.3.9. El botón Imprimir

El botón que permite imprimir rápidamente una gráfica presente en la ventana es el botón **"Imprimir"**, el cual permite un acceso rápido a la opción **"Imprimir…"** del menú **"Archivo"**.

De la misma manera en que se activan los dos botones anteriormente mencionados, el botón **"Imprimir"** se encuentra activo solamente, si el botón **"Plottear"** también lo está.

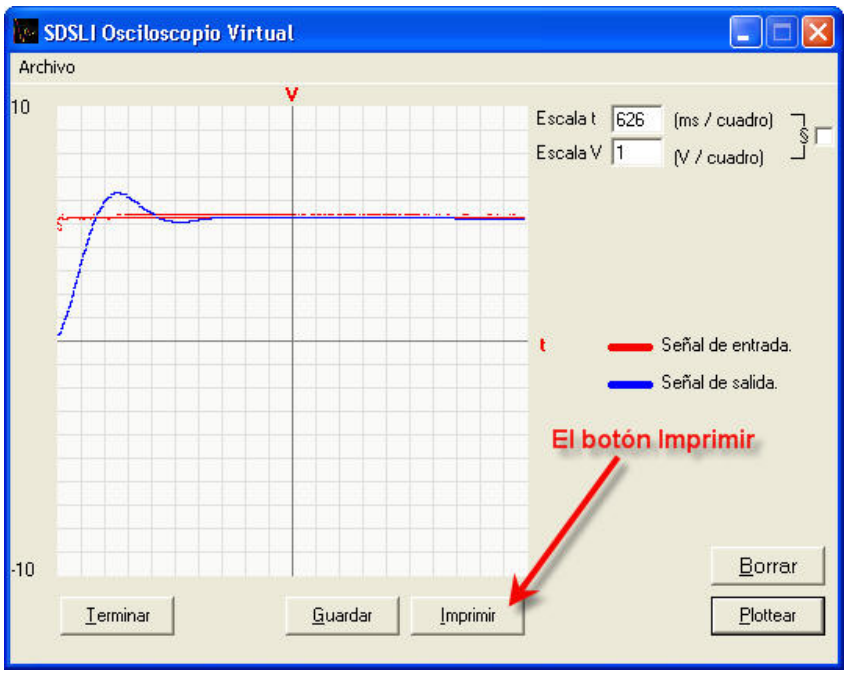


Figura 64. El botón Imprimir del Osciloscopio Virtual

3.3.10. El botón Terminar

Para poder finalizar una simulación y regresar a la ventana principal o a la ventana de Simulación de función de transferencia $H[z]$, existen tres posibilidades, hacer click en el botón **"Terminar"**, acceder a la opción **"Regresar…"** del menú **"Archivo"** o bien utilizar el botón que cierra esta ventana 

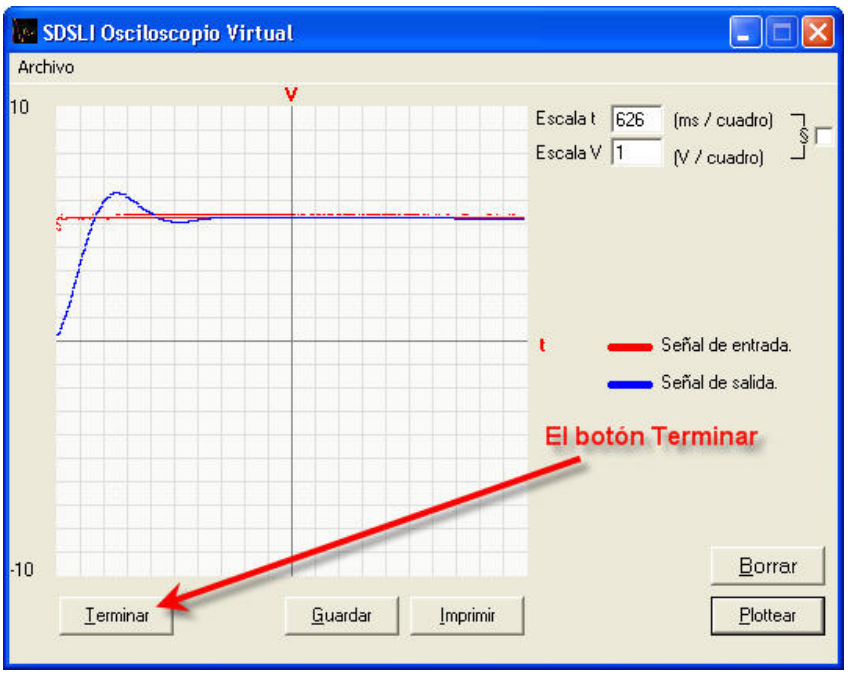


Figura 65. El botón Terminar del Osciloscopio Virtual

4. ¿CÓMO REALIZAR UNA SIMULACIÓN CON EL SDSLI?

Como ya se mencionó con anterioridad, debido a que el simulador es digital, éste trabaja con datos de la función de transferencia $H[z]$ y no con los de la función de transferencia $G(s)$ directamente. En la ventana principal, se le requiere al usuario que introduzca los valores de los coeficientes de la función de transferencia $G(s)$ para posteriormente, poder calcular los valores de los coeficientes de la función de transferencia $H[z]$ que serán utilizados para realizar la simulación en el hardware del SDSLI.

Por otro lado, en la ventana *Simulación de función de transferencia* $H[z]$, los coeficientes de la función de transferencia $H[z]$ son introducidos en esta ventana y son con los que el simulador trabajará directamente, sin tener que realizar cálculos adicionales como en el caso de la ventana principal.

Las instrucciones que a continuación se presentan, sirven tanto para el caso de la ventana principal como para el caso de *Simulación de función de transferencia* $H[z]$.

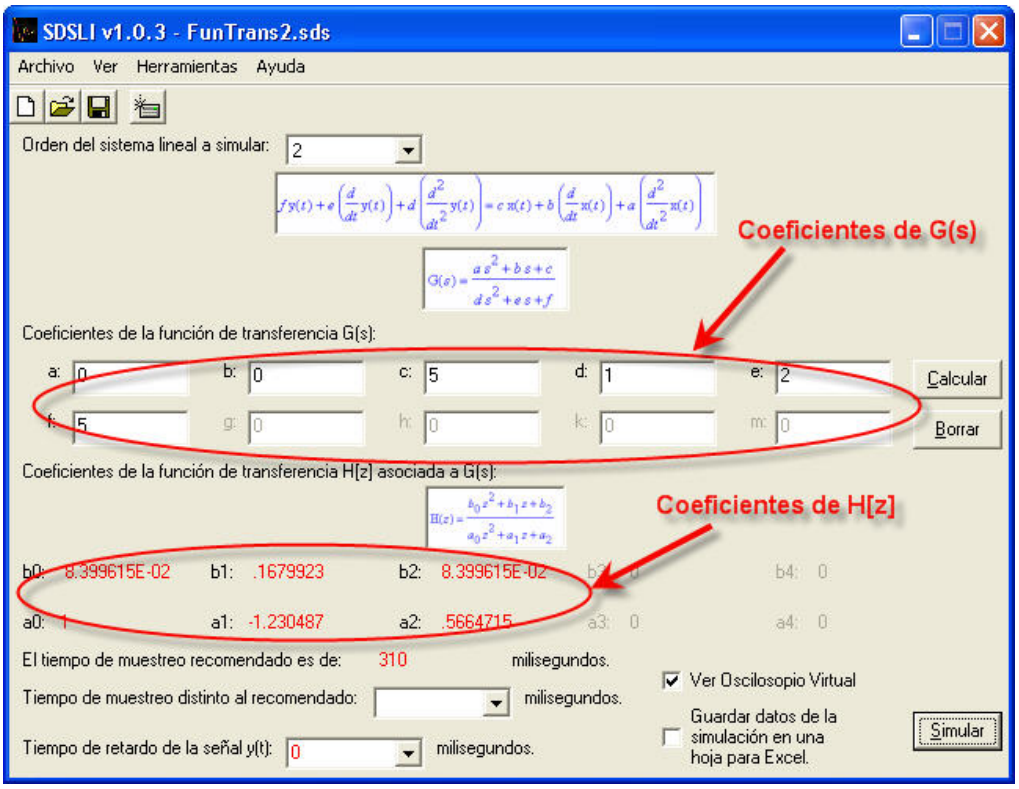


Figura 66. Coeficientes de la función de transferencia G(s) y coeficientes de la función de transferencia H[z] en la ventana principal

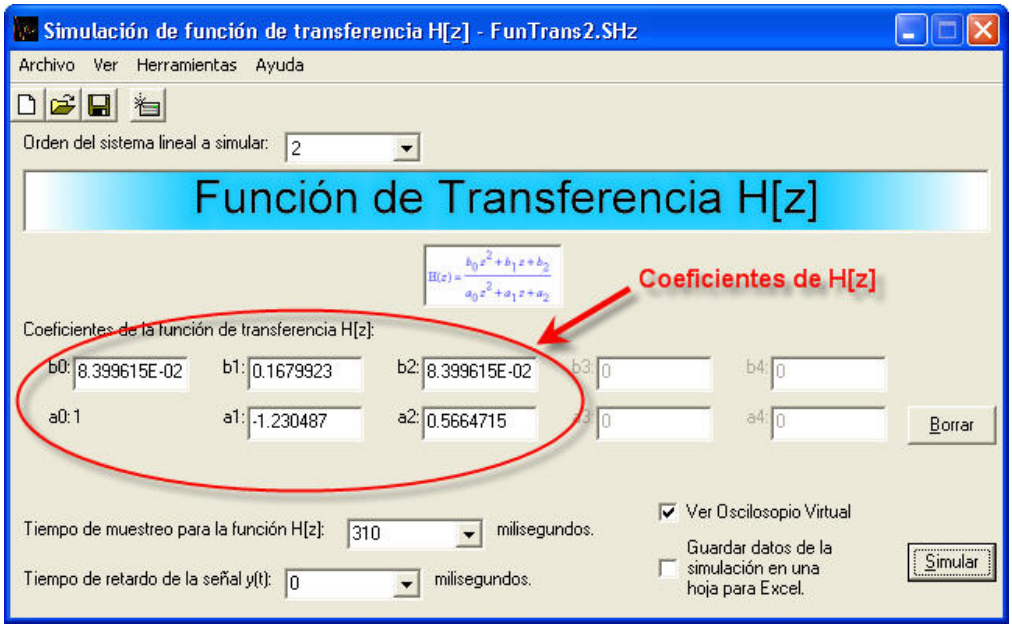


Figura 67. Coeficientes de la función de transferencia H[z] en la ventana Simulación de función de transferencia H[z]

Para realizar una simulación tanto en la ventana principal como en la ventana *Simulación de función de transferencia* $H[z]$, siga los siguientes pasos:

1. **Elija el orden del sistema lineal**. El orden del sistema puede ser 0, 1, 2, 3 y 4 para el caso de la ventana principal y 1, 2, 3 y 4 para el caso de la ventana *Simulación de función de transferencia* $H[z]$. Si desea realizar simulaciones con sistemas lineales de orden cero, el software sólo requiere que se introduzca un valor de constante de proporcionalidad Kp. Para los sistemas lineales de orden uno, dos, tres y cuatro, el sistema requerirá que introduzca los coeficientes respectivos de la función de transferencia $G(s)$.

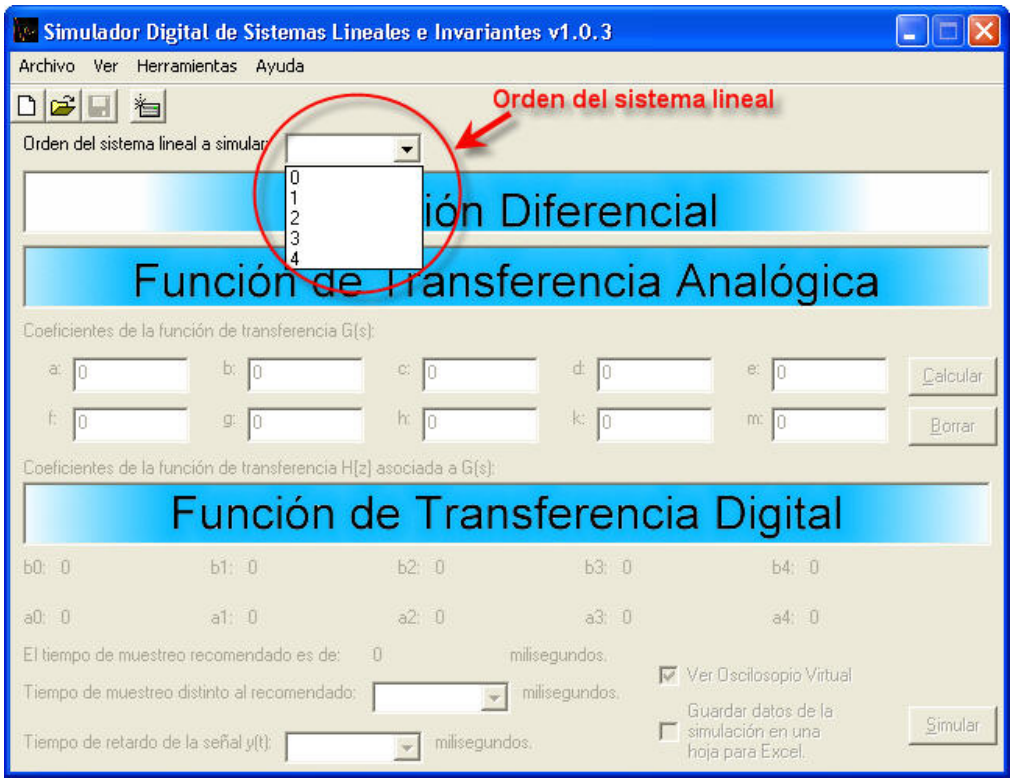


Figura 68. Elección del orden del sistema lineal a simular

2. **Introduzca los coeficientes de la función de transferencia que desea simular.** Para el caso de la ventana principal, deberá introducir los coeficientes de la función de transferencia $G(s)$ y para el caso de la ventana *Simulación de*

función de transferencia $H[z]$ deberá introducir los coeficientes de la función de transferencia $H[z]$.

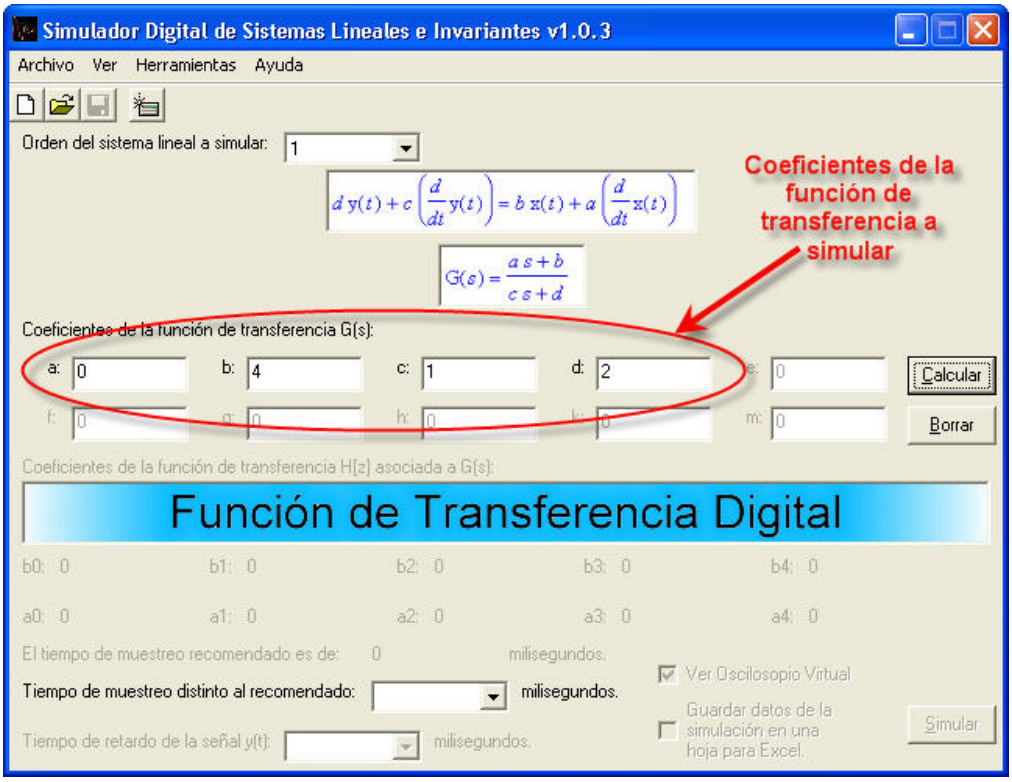


Figura 69. Introducción de los coeficientes de la función de transferencia a simular

3. **Haga click en el botón "Calcular".** Para calcular los coeficientes de la función de transferencia $H[z]$ así como un tiempo de muestreo recomendado, es necesario hacer click en el botón **"Calcular"**, en caso de que esté en la ventana principal. En caso de que se encuentre en la ventana *Simulación de función de transferencia* $H[z]$, no es necesario este paso. Si hace click en **"Borrar"**, borrará los coeficientes de la función de transferencia que se ven en la ventana.

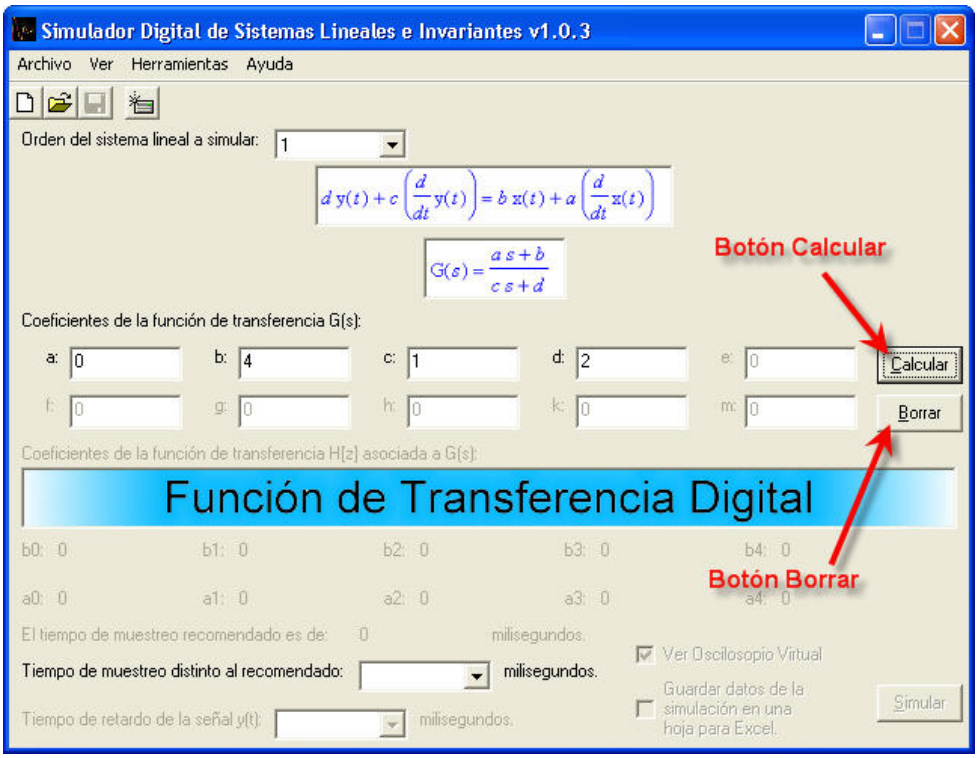


Figura 70. Botón Calcular y botón Borrar

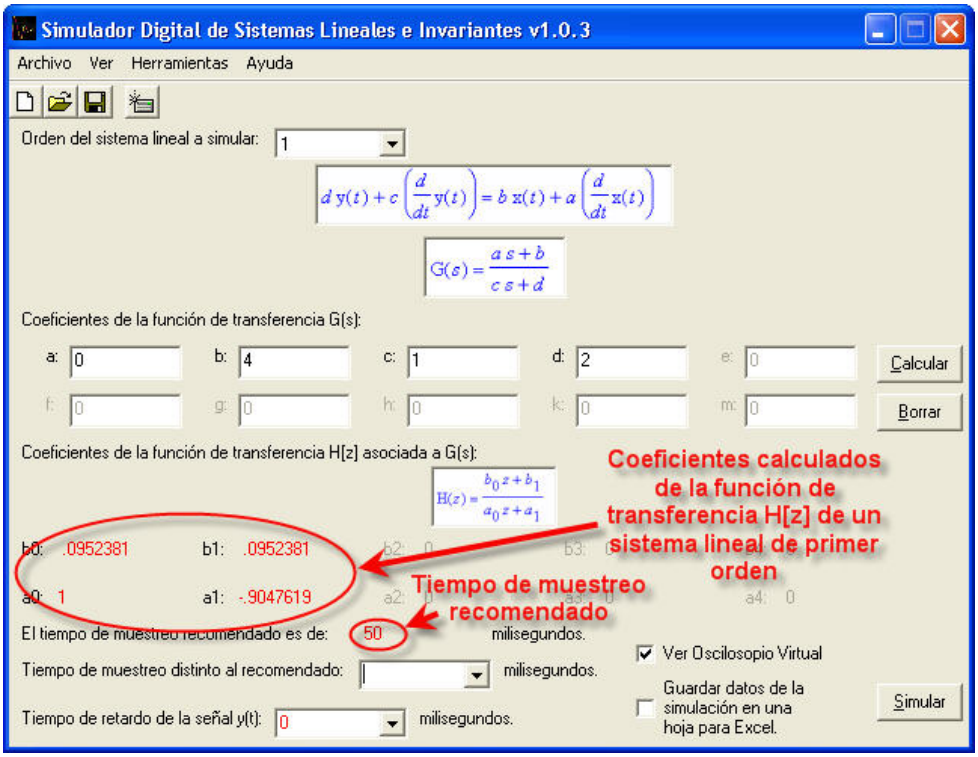


Figura 71. Coeficientes calculados de la función de transferencia H[z] y el tiempo de muestreo recomendado

4. **Elija el valor del tiempo de muestreo.** En el caso de la ventana principal, si lo desea, es posible que elija un tiempo de muestreo distinto al recomendado. En el caso de la ventana *Simulación de función de transferencia* $H[z]$ es obligatorio que indique el tiempo de muestreo correspondiente a los coeficientes de la función de transferencia $H[z]$ que introdujo, pues en esta ventana no existe un tiempo de muestreo recomendado pero sí por omisión.

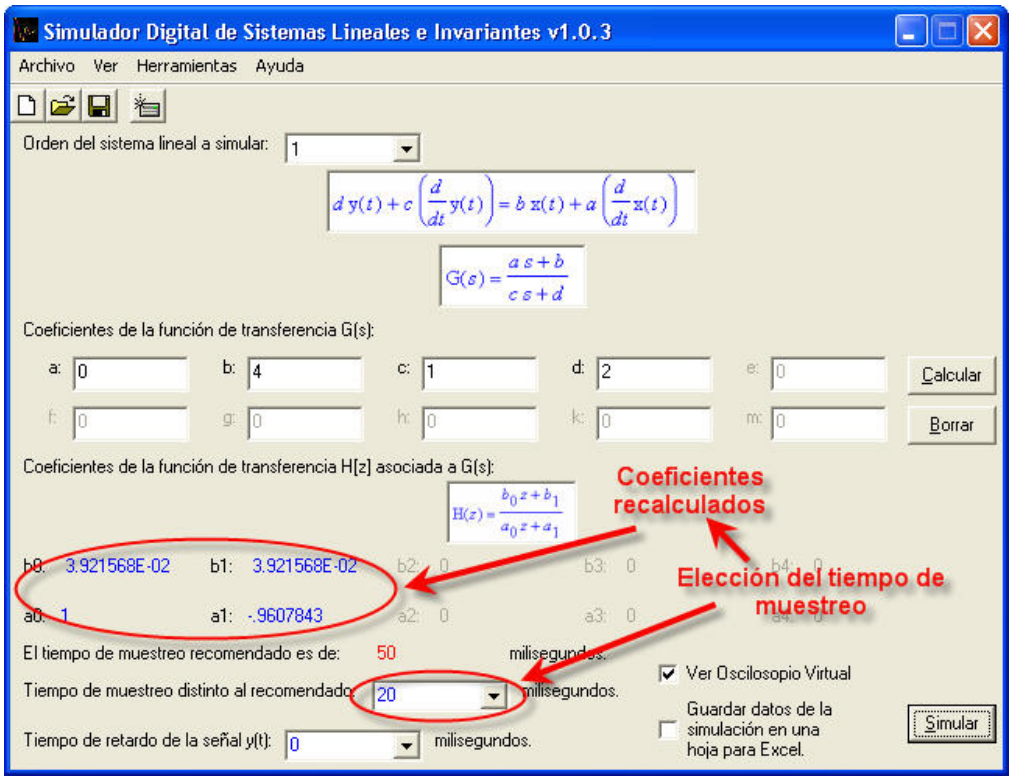


Figura 72. Coeficientes recalculados de la función de transferencia H[z] y el tiempo de muestreo distinto al recomendado elegido por el usuario. Caso de la ventana principal

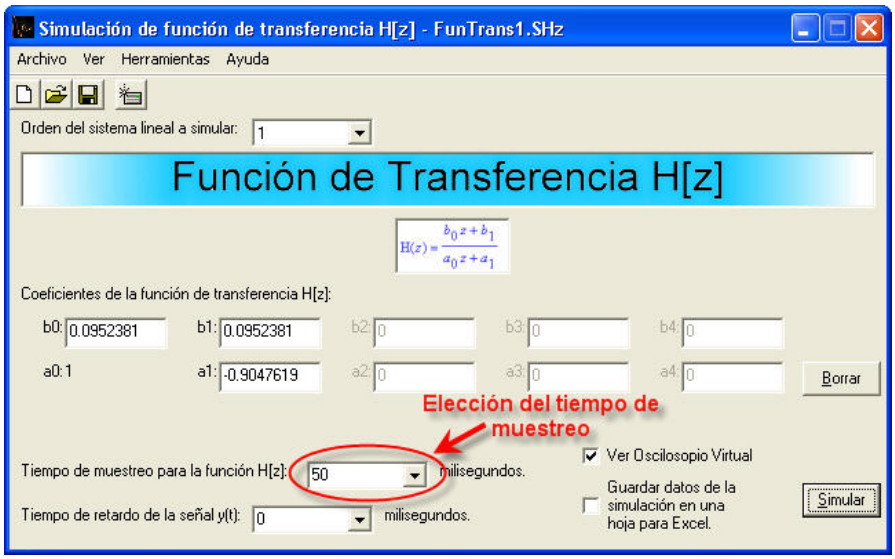


Figura 73. El tiempo de muestreo seleccionado por el usuario. Caso de la ventana Simulación de función de transferencia H[z]

5. **Seleccione el tiempo de retardo de la señal y(t).** Tanto en la ventana principal, como en la ventana *Simulación de función de transferencia* $H[z]$, si lo desea, puede elegir un valor de tiempo de retardo de la señal de salida $y(t)$.

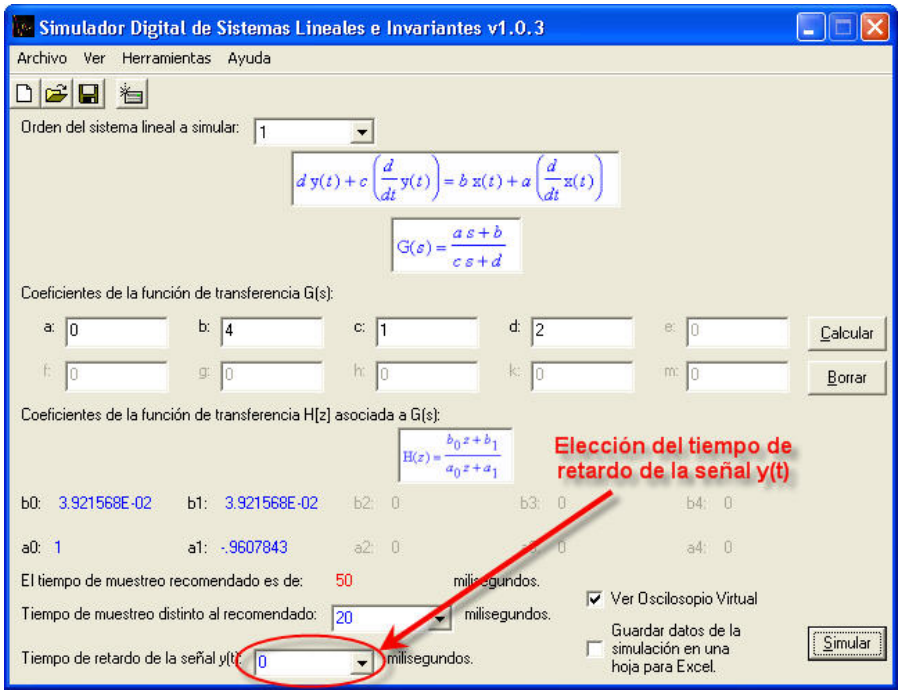
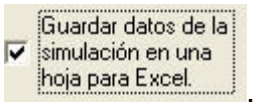


Figura 74. El tiempo de retardo de la señal de salida $y(t)$ seleccionado por el usuario

6. **Active la casilla de verificación para visualizar el Osciloscopio Virtual.** Si desea que al momento de iniciar la simulación, aparezca el Osciloscopio Virtual, marque la casilla de verificación 

.

7. **Active la casilla para guardar los datos de una simulación en una hoja para Excel.** Si es de su elección o requerimientos que el software almacene los datos de la simulación en una hoja para Excel, entonces marque la casilla de verificación 

.

8. **Almacene en disco su simulación.** Antes de iniciar la simulación, se recomienda que almacene en disco los datos de su simulación. Para esto, puede hacer click en el botón  de la barra de herramientas de la ventana o bien acceder a las opciones **"Guardar"** o **"Guardar como…"** del menú **"Archivo"**.

9. **Para simular el sistema, haga click en el botón "Simular".** Finalmente, si desea ejecutar la simulación y observar las señales resultantes de la simulación, presione el botón **"Simular"** de la ventana.

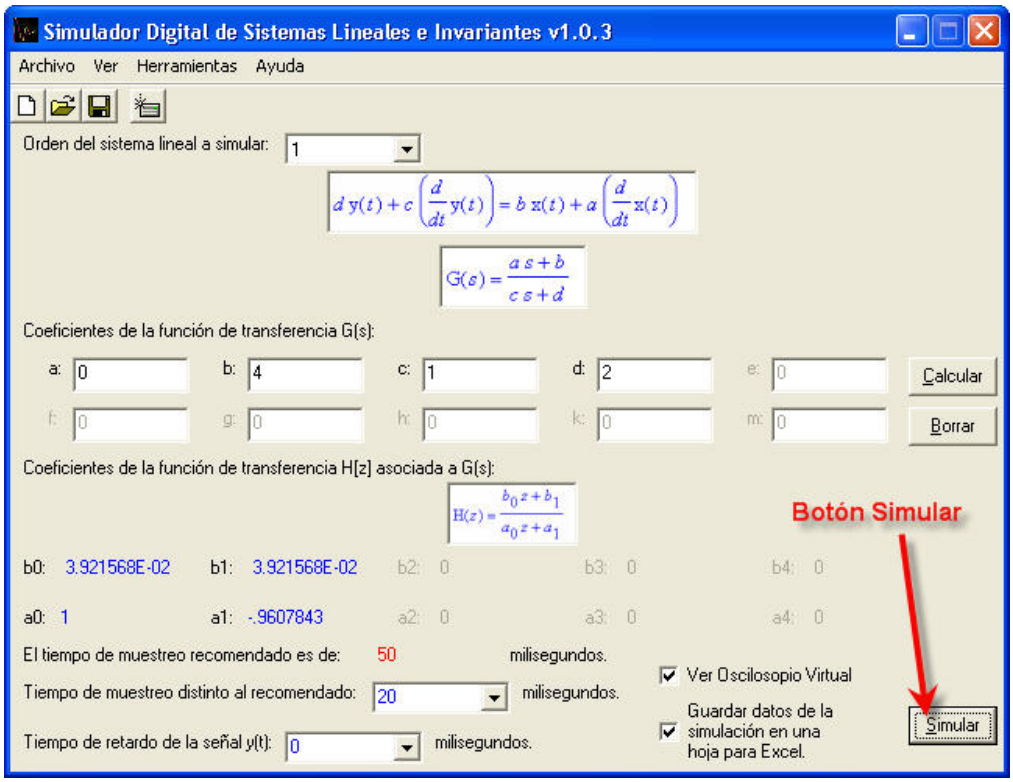


Figura 75.- El botón Simular

En el ejemplo, se ha elegido simular un sistema lineal en invariante en el tiempo de primer orden, con una ganancia 2, un tiempo de muestreo de $20(ms)$ y sin retardo de la señal de salida $y(t)$ respecto de la señal de entrada $x(t)$. Adicionalmente se ha elegido ver el Osciloscopio Virtual durante la simulación y almacenar los datos en una hoja para Excel.

Para efectos ilustrativos, se ha introducido como señal de entrada $x(t)$ una señal escalón de magnitud de $5(V)$.

Se deberá ver en la pantalla de la PC la ventana del Osciloscopio Virtual desplegando las señales siguientes:

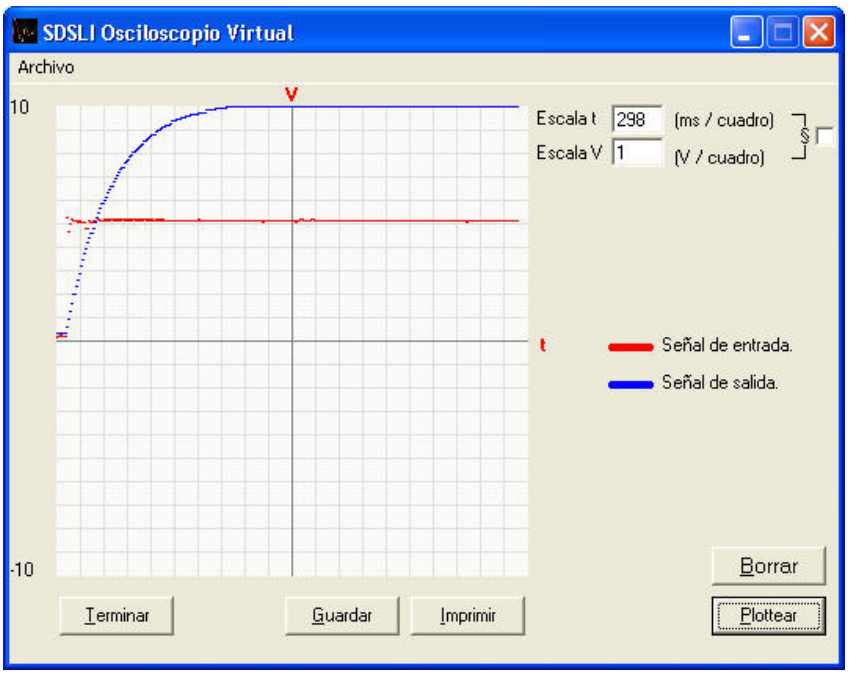


Figura 76.- El resultado de la simulación del ejemplo

¡Listo! Usted ha realizado una simulación con SDSLI.

5. CONFIGURACIÓN DE UN SISTEMA LINEAL PARA SU SIMULACIÓN EN MODO AUTÓNOMO

El modo **AUTÓNOMO** adicionalmente al modo **CLIENTE**, es otra modalidad en la que el hardware del SDSLI puede operar.

Cuando el simulador se encuentra operando en modo **CLIENTE**, permite al usuario darle órdenes al hardware del SDSLI mediante una PC. Una vez que la PC y el hardware del SDSLI se han enlazado apropiadamente, es posible utilizar el software del SDSLI para realizar simulaciones.

Puede existir la necesidad de que el hardware al encenderlo o resetearlo, comience a desarrollar una simulación sin tener que ser instruido por el usuario mediante una PC. Un ejemplo común sucede, cuando se están probando plantas de control o controladores digitales. Entonces, el usuario ya no desea tener que disponer de una PC ni del software del SDSLI para que el simulador comience a desarrollar una simulación. La modalidad de operación del hardware del SDSLI en modo autónomo satisface esta necesidad.

Esta modalidad consiste en dejar “programado” el hardware del SDSLI para que cuando éste se encienda o sea reseteado, comience automática e inmediatamente a desarrollar una simulación recibiendo la señal de entrada $x(t)$ y entregando la señal de salida $y(t)$ sin requerir intervención extra por parte del usuario.

Es muy sencillo configurar un sistema lineal en el hardware del simulador.

1. Una vez que ha probado y simulado el sistema lineal (PRESENTE EN LA VENTANA) que desea configurar para su funcionamiento en modo autónomo, ya sea en la ventana principal en forma de función de transferencia $G(s)$ o bien en la ventana *Simulación de función de transferencia* $H[z]$ en forma de función de

transferencia $H[z]$, acceda al menú **"Herramientas"** y elija la opción **"Configuración de modo autónomo…"**.

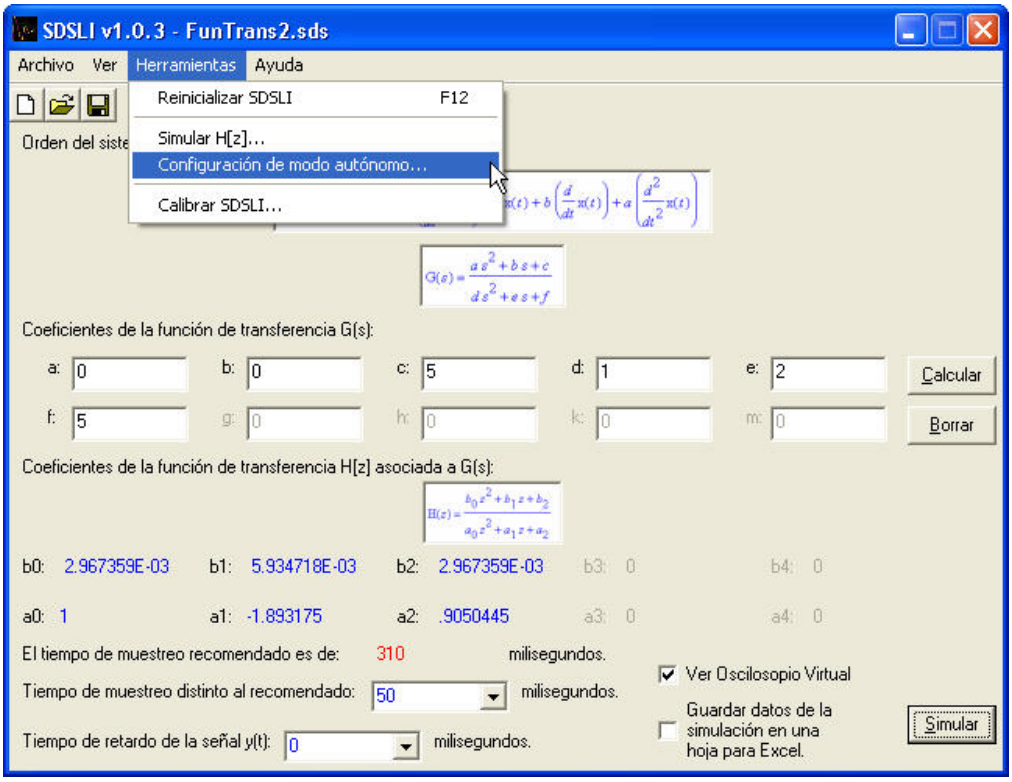


Figura 77. Opción de configuración de modo autónomo del menú "Herramientas"

2. Iniciará el asistente de configuración de modo autónomo. Siga las instrucciones detallada y cuidadosamente.

Figura 78.-Ventana inicial del asistente de configuración de modo autónomo

3. Mueva el switch a la posición **"CONFIGURACIÓN DE MODO AUTÓNOMO"**.

Figura 79.-Switch que se debe mover a la posición "CONFIGURACIÓN DE MODO AUTÓNOMO"

4. Presione el botón **"RESET"** del hardware del SDSLI.

Figura 80.-Botón "RESET"

5. Haga click en el botón **"Aceptar"** de la ventana que aparece.

Figura 81.-Botón "Aceptar" de la ventana del paso 1 del asistente de configuración de modo autónomo

6. Verá una barra de progreso. En este momento se está configurando la simulación en el hardware del SDSLI para que opere en modo autónomo.

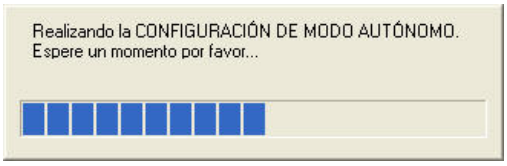


Figura 82.-Barra que indica el progreso de la configuración de modo autónomo

7. Mueva el switch a la posición **"SIMULACIÓN"**

Figura 83.-Switch que se debe mover a la posición "SIMULACIÓN"

8. Presione el botón **"RESET"** del hardware del SDSLI.

Figura 84.-Botón "RESET"

9. Haga click en el botón **"Aceptar"** de la ventana que aparece.

Figura 85.-Botón "Aceptar" de la ventana del paso 2 del asistente de configuración de modo autónomo

¡Listo! Usted ha dejado configurado el sistema lineal presente en la ventana, para que la próxima vez que mueva el switch **"MODO DE OPERACIÓN"** a la posición **"AUTÓNOMO"** y encienda o resetee el hardware del SDSLI, la simulación comience a ejecutarse automáticamente.

Si desea probar el funcionamiento de la simulación en modo autónomo, siga los pasos que se describen a continuación:

1. Apague el hardware del SDSLI.

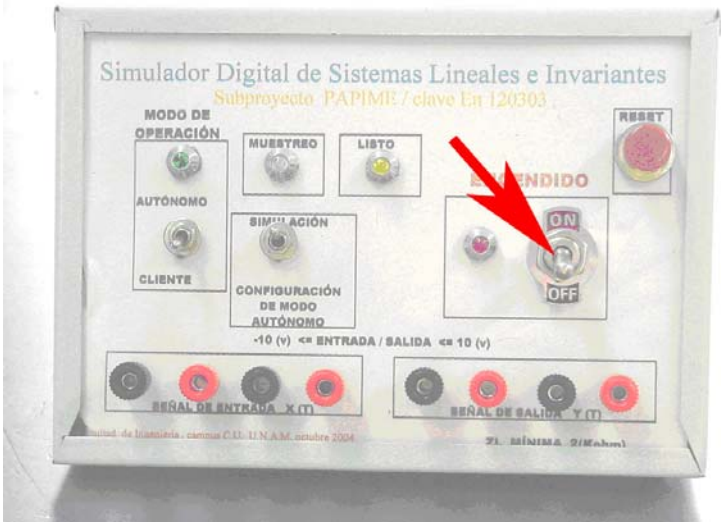


Figura 86.-Switch "ENCENDIDO" que se debe mover a la posición "OFF"

2. Mueva el switch **"MODO DE OPERACIÓN"** a la posición **"AUTÓNOMO"**.

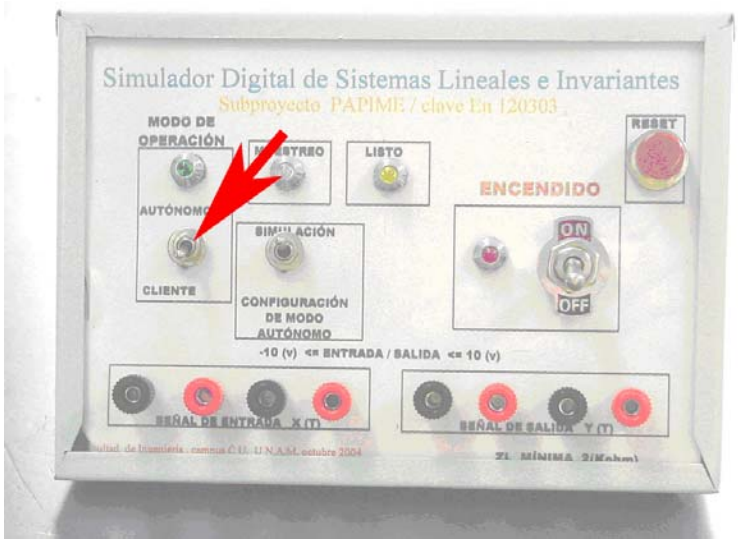


Figura 87.-Switch "MODO DE OPERACIÓN" que se debe mover a la posición "AUTÓNOMO"

3. Mueva el switch a la posición **"SIMULACIÓN"**.

Figura 88.-Switch que se debe mover a la posición "SIMULACIÓN"

4. .Vuelva a encender el hardware del SDSLI y presione el botón **"RESET"**.

Figura 89.- Switch "ENCENDIDO" que se debe mover a la posición "ON" y presionar "RESET"

5. Verá que el led **"MUESTREO"** comienza a parpadear indicando que se encuentra muestreando la señal de entrada $x(t)$ y entregando como resultado la señal de salida $y(t)$.

Figura 90.- Led "MUESTREO" del SDSLI

6. Introduzca una señal $x(t)$ dentro de los rangos adecuados y lea, con un osciloscopio, la señal de salida $y(t)$ que el simulador entrega como resultado de la señal de entrada.

Figura 91.- Bornes para la señal de entrada $x(t)$ y bornes para la señal de salida $y(t)$

6. CALIBRACIÓN DEL HARDWARE DEL SDSLI

El procedimiento de calibración se detalla en el capítulo 8 del Manual Técnico del SDSLI.

Sin profundizar, el software del SDSLI posee un asistente de calibración del hardware del SDSLI que ayuda y guía al usuario en la calibración de éste. Para iniciar el asistente, basta acceder desde la ventana principal al menú **"Herramientas"** y elegir la opción **"Calibrar SDSLI…"**. De inmediato se iniciará el asistente de calibración del SDSLI.

Se recomienda que la calibración del hardware del SDSLI la realice personal calificado y con conocimientos de electrónica para evitar que la circuitería del hardware o el propio usuario, sufran algún daño.

REFERENCIAS BIBLIOGRÁFICAS

[Salvá 2002]	Salvá, Antonio; Sánchez, Víctor. **"Simulador Digital de Sistemas Dinámicos Lineales e Invariables para auxilio didáctico".** Versión disponible en archivo PDF. México 2002.
[A. Salvá 2002]	Salvá, Antonio. **"Mapas de memoria y configuración de la tarjeta FACIL_11B".** Versión disponible en archivo PDF. México 2002.
[A. Salvá 2000]	Salvá, Antonio. Memoria de ELECTRO 2000, **"PUMMA_11, software en ambiente Windows, para desarrollo con el microcontrolador 68HC11".** Versión disponible en archivo PDF. México Chihuahua, Chihuahua 2000.
[V. Sánchez 2000]	Salvá, Antonio; Sánchez, Víctor M. Memoria SOMI, **"Controlador digital para auxilio didáctico basado en el microcontrolador 68HC11".** Versión disponible en archivo PDF. México Guadalajara, Jalisco 2000.
[Gordon 1982]	Gordon, Geoffrey. **"Simulación de Sistemas".** Editorial DIANA. México D. F. 1982. Páginas: 15 a 31, 33 a 46, 50 a 51.
[Percuoco 1986]	Percuoco, Alfonso. **"Sistemas Analógicos e Híbridos".** Editorial LIMUSA. México D. F. 1986. Vol. 1. Páginas: 19 a 22, 123 a 127.
[Neff 1984]	Neff, Herbert P. **CONTINUOUS AND DISCRETE LINEAR SYSTEMS "The Z transform".** Editorial Harper & Row Publishers. U.S. A. New York 1984.
[Ogata 1980]	Ogata, Katsuhiko. **"Ingeniería de Control Moderna".** Prentice-Hall Hispanoamericana, S. A. México D. F. 1980.

[Ogata 1998]	Ogata, Katsuhiko. **"Ingeniería de Control Moderna".** Tercera Edición. Prentice-Hall. México D. F. 1998. Páginas: 3, 57 a 61.
[Mata 2002]	Mata H., Gloria; Sánchez, Víctor M.; Gómez, Juan M. "Análisis de sistemas y señales con cómputo avanzado". Primera Edición. UNAM Facultad de Ingeniería. México D. F. 2002.
[Eronini 1990]	Eronini Umez-Eronini. **"Dinámica de Sistemas y Control".** Thomson Learning. México D. F. 1990.Páginas: 318 a 321.
[Kuo 2000]	Kuo, Benjamín C. **"Sistemas de Control Digital".** CECSA. México D. F.2000. Páginas: 64 a 66, 286 a 288, 418 a 419, 428 a 429, 688 a 689.
[Oppenheim 1998]	Oppenheim, Alan; Willsky, Alan; Nawab, Hamid. **"Señales y Sistemas".** Prentice-Hall. México D. F. 1998. Páginas: 1 a 5, 46 a 48.
[UMich 1997]	http://www.engin.umich.edu/group/ctm/examples/susp/susp.html
[Kuo 1965]	Kuo, Shan S. **"Numerical Methods and Computers".** University of New Hampshire, Addison Wesley. U.S.A. New York 1965. Second Edition. Página: 149.