3º INGENIERÍA INDUSTRIAL

AUTÓMATAS Y SISTEMAS DE CONTROL

PRÁCTICA SISTEMAS DE CONTROL

IDENTIFICACIÓN DE UN SERVOMOTOR CON COMPUTADOR

OBJETIVOS

- Introducir al alumno en la técnica de control de procesos utilizando un computador como elemento de control digital.
- Familiarizar al alumno con los métodos de trabajo en control digital (utilización de tarjetas de adquisición de datos y realización del algoritmo de control).
- Identificación de un sistema de primer orden mediante un computador.
- Control de un servomotor mediante los distintos tipos de reguladores discretos estudiados implementados en un computador.

1. INTRODUCCIÓN AL CONTROL CON COMPUTADOR

Actualmente el control digital ofrece unas mayores prestaciones que el control analógico. Las causas de este hecho son las ventajas que éste ofrece. Entre otras podemos citar las siguientes:

- Potencia: el control digital permite un procesamiento más sofisticado que el control analógico. La implementación de funciones no lineales complejas ya no suponen ningún problema pues se reducen a una línea de programa dentro del algoritmo de control.
- Precisión: aunque en el control digital aparecen retardos de cálculo, efectos de cuantización (errores en las conversiones A/D y D/A), su precisión es superior a los sistemas de control analógico
- Versatilidad: el cambio de un esquema de control a otro consiste únicamente en el cambio de una línea de programa.
- Eficacia: un único computador puede implementar múltiples tareas, desde varios lazos de control hasta sistemas monitor, comprobación de errores, presentación de resultados, etc.
- Fiabilidad: toda la teoría desarrollada para la fiabilidad y tolerancia a fallos en sistemas de microprocesadores son también aplicables en el control, haciendo de éste un método fiable y seguro.
- Interfaz con el operador más amigable y potente por medio de periféricos fáciles de utilizar (teclados, ratones...) y de presentaciones gráficas, ventanas, menues, ...

Como gran inconveniente se puede citar el retardo de cálculo inherente al procesamiento de la información (aunque con la potencia de los microprocesadores actuales se considera casi despreciable), los efectos de cuantización y un diseño más laborioso que en el control analógico.

El control digital se basa en la utilización de un elemento de naturaleza discreta para la realización del control. Dicho elemento se introduce en el lazo de control para hacer las veces de regulador. Este hecho supone introducir un elemento de naturaleza discreta en un entorno donde los sistemas a controlar son continuos y por tanto las señales que éstos generan también. Es por tanto necesario el uso de elementos adicionales que acomoden las señales de distinta naturaleza (continuas y discretas) para que estas puedan

ser utilizadas por ambos elementos. Dichos elementos son los muestreadores, que permiten a un elemento discreto manipular señales continuas y los retenedores que realizan la operación inversa, es decir, a un elemento continuo manipular señales de naturaleza discreta.

En principio, el computador, por si solo, es un elemento pasivo en el lazo de control. Para que éste pueda funcionar como regulador necesita un mecanismo que permita por una parte establecer una comunicación con el entorno, es decir, capturar y enviar señales, y por otro adecuar estas señales a la naturaleza del entorno con el que se está trabajando (continuo o discreto). Dicho mecanismo viene implementado por una tarjeta de adquisición de datos. Dicha tarjeta permite la comunicación del computador con el entorno (adquiriendo y enviando señales) a la vez que realiza las conversiones necesarias mediante muestreadores o retenedores. El hecho de adquirir una señal continua por el computador (elemento de naturaleza discreta) se denomina conversión analógico – digital (abreviadamente conversión A/D), mientras que la operación inversa, el envío de una señal discreta a un sistema o medio continuo se denomina conversión digital – analógica (conversión D/A).

Además de la tarjeta de adquisición de datos, el computador necesita un entorno de programación que permita realizar el algoritmo de control. No olvidemos que un regulador discreto se puede representar mediante una ecuación en diferencias de forma que se puede convertir ésta en una línea de programa. Obviamente, dicho entorno de programación debe ofrecer los mecanismos adecuados (librerías) para la correcta comunicación entre el computador y la tarjeta de adquisición de datos.

Una vez expuestos estos términos preliminares, pasemos a describir el sistema experimental que se va a utilizar en la práctica.

2. DESCRIPCIÓN DEL SISTEMA EXPERIMENTAL

Esquemáticamente, el diagrama de bloques general de un sistema de control por computador es el siguiente:

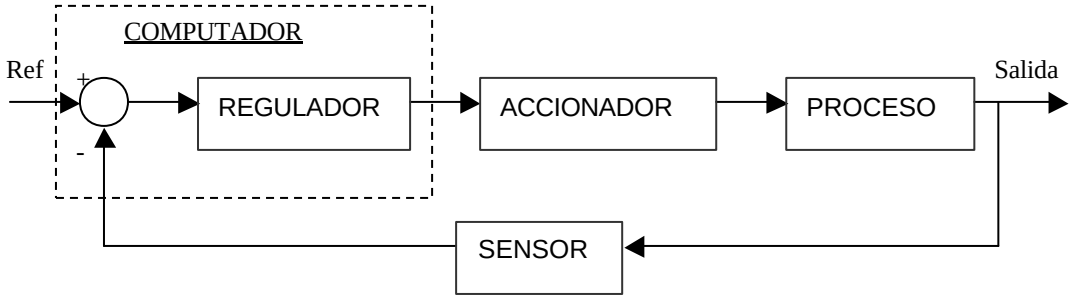

Figura 1. Esquema general de un sistema de control por computador

El funcionamiento del lazo de control anterior es el siguiente: el operador genera una señal de consigna o referencia que será utilizada por el regulador para que la salida del sistema coincida con dicha señal. El regulador a partir de dicha consigna, de la salida del sistema y con el algoritmo de control adecuado, genera una acción de control que pasa al accionador que se encargará de adecuar dicha señal a las características de la entrada del proceso. Un sensor lee la salida del sistema y la traslada al controlador. Nótese que en los pasos anteriores se han obviado las operaciones de conversión A/D y D/A realizador por el sistema de adquisición de datos conectado al computador.

Tomando como referencia el esquema de bloques anterior, identifiquemos los componentes físicos correspondientes que se van a utilizar en esta práctica:

a) Proceso: el proceso será el servomotor de corriente continua Feedback ya utilizado en otras prácticas.

b) Accionador: está incluido en el servomotor y es la electrónica necesaria generar la señal necesaria que actúa sobre el motor. Nótese que este elemento al estar incluido en el propio proceso no nos debe preocupar en principio a la hora del diseño del regulador ya que la identificación del proceso engloba el comportamiento del accionador.
c) Sensor: el sensor en este caso también se encuentra englobado en el mismo proceso. Los sensores de que se dispone son entre otros un encoder que nos proporcionan la posición del eje del servomotor y un tacogenerador que nos proporciona la velocidad angular de giro del eje del motor. Las señales proporcionadas por ambos sensores son valores de tensión, que se pueden medir directamente con el sistema de adquisición de datos. Además, dichos sensores son de ganancia unitaria y no presentan ninguna dinámica asociada por lo que tampoco es necesario contemplarlos en la identificación del proceso.
d) Regulador: el regulador en este caso está formado por el computador (con su sofware asociado) y el sistema de adquisición de datos, que en este casó será una tarjeta de adquisición de datos de la casa NuDAQ, en concreto la ACL-8112PG cuyas prestaciones y características se comentarán más adelante.

Hasta aquí se ha descrito brevemente el sistema con el que se va a trabajar, sin embargo, es necesario realizar algunas matizaciones. Es necesario aclarar que se trata de un equipo de aplicación docente y como tal se han previsto algunos de los efectos no deseables que se presentan en entornos industriales como pueden ser el ruido electromagnético, las condiciones ambientales y climáticas, etc. Es por esto por lo que en aplicaciones industriales se suelen utilizar los denominados PC's industriales que están preparados para soportar estos efectos y por tanto están provisto de una eficaz protección contra ruidos e interferencias (mediante el correspondiente apantallamiento), polvo y otros elementos perjudiciales (mediante complejos sistemas de ventilación y filtrado) y temperaturas excesivamente bajas o elevadas (mediante la calidad de sus componentes). Además, en aplicaciones donde tanto la precisión como las prestaciones exigidas son elevadas se suele trabajar con sistemas distribuidos donde por ejemplo el sistema de adquisición de datos es independiente del computador (mucho más preciso y más caro) y la comunicación entre ambos y/u otros subsistemas se realiza mediante buses industriales especiales.

Veamos ahora las características de la tarjeta de adquisición de datos y del software utilizado para el manejo de la misma y de la realización del algoritmo de control.

3. LA TARJETA DE ADQUISICIÓN DE DATOS ACL-8112PG

Sobre los sistemas de adquisición de datos y también sobre la descripción y manejo de esta tarjeta se podría escribir mucho. En cualquier caso, no es el objetivo de esta práctica el estudio profundo de los sistemas de adquisición de datos ni del funcionamiento y programación de esta tarjeta por lo que nos limitaremos únicamente a una exposición básica de sus prestaciones y modos de trabajo.

La tarjeta de adquisición de datos ACL – 8112PG es una tarjeta de adquisición de datos de altas prestaciones, multifunción y de alta velocidad diseñada para trabajar con computadores IBM PC o compatibles. Dicha tarjeta esta diseñada para combinar todas las funciones de la adquisición de datos, tales como las conversiones A/D, D/A, la entrada – salida de señales digitales (DIO), y cuenta además con temporizadores y contadores. Las especificaciones finales de esta tarjeta la hacen ideal para un amplio abanico de aplicaciones que requieren una alta velocidad de adquisición a una resolución de 12 bits. Como resumen de sus características , la tarjeta ACL-8112PG poses 16 entradas analógicas single – ended que soportan una tasa de transferencia de hasta 100 KHz mediante transferencia por DMA, 2 canales multiplexados de entrada analógica de 12 bits, 16 entradas y salidas digitales y un canal temporizador/contador.

Veamos más detalladamente sus características y especificaciones.

Características de la tarjeta ACL – 8112PG

- AT – Bus (interfaz ISA)
- 16 single – ended canales de entrada

- Ganancia programable de x1, x2, x4, x8 y x16
- Circuito integrado de muestreo y retención
- 2 canales de salida analógicos de 12 bits monolíticos multiplexados
- 16 canales de entrada digital y 16 de salida digital
- 3 contadores independientes de cuenta atrás de 16 bits
- Frecuencia de muestreo programable de hasta 100KHz
- Tres modos de disparo del convertidor A/D: disparo por software, disparo programable mediante temporizador/contador y disparo mediante pulso externo.
- Capacidad para trabajar mediante interrupción: 11 niveles de interrupción (IRQ3 a IRQ15) seleccionables mediante jumper
- Conector de 37 pines tipo D
- Tamaño compacto (half – size PCB)

Especificaciones de la tarjeta ACL – 8112PG

- Entrada analógica (A/D)
  - Convertidor A/D: B.B. ADS774 (aproximaciones sucesivas)
  - Canales de entrada: 16 single – ended
  - Resolución: 12 bits
  - Rango de entrada (controlados por sofware): Bipolar de ±10V, ±5V, ±2.5±, ±0.625V y ±0.3125V (a menor rango mayor resolución)
  - Tiempo aproximado de conversión: 8μs
  - Protección de sobretensión: ±35V DC
  - Precisión: 0.015% de la lectura ±1 bit
  - Impedancia de entrada: 10MΩ
  - Modos de disparo: por software, por temporizador/contador y disparo externo
  - Modos de transferencia: por software, por interrupción y por DMA
  - Velocidad de transferencia de datos (frecuencia de muestreo): 100KHZ (máximo)

- Salida analógica (D/A)
  - Canales de salida: 2 salidas analógicas con doble buffer
  - Resolución: 12 bits
  - Rango de salida: unipolar de 0 a 5V o 0 a 10V mediante referencia interna y unipolar de +10V o –10V con referencia externa
  - Convertidor D/A: B.B. DAC7541 de multiplexación monolítica
  - Tiempo de conversión: 30μs
  - Linealidad: ±1/2 bit LSB
  - Intensidad máxima: ±5mA

- Entradas y salidas digitales (DIO)
  - Canales: 16 de entrada y 16 de salida compatibles TTL
  - Tensión de entrada: mínimo de 0V y máximo de 0.8V para estado 0 y mínimo de +2.0V para estado 1
  - Input load: +0.5V a –0.2mA máximo para estado 0 y +2.7V a +10mA máximo para estado 1
  - Tensión de salida: mínimo de 0V y máximo de 0.4V para estado 0 y mínimo de +2.4V para estado 1

- Contador programable
  - Dispositivo: 8254
  - Temporizador para conversión A/D: temporizador de 32 bits (dos contadores de 16 bits en cascada) con base de tiempos de 2MHz
  - Contador: un contador de 16 bits con base de tiempos de 2MHz
  - Frecuencia de salida del temporizador: de 0.00046 Hz a 0.5 MHz

Estas son a grandes rasgos las principales características de la tarjeta de adquisición de datos ACL 8112PG. Para realizar la conexión con los dispositivos se utiliza una tarjeta adicional conectada mediante un cable y conector de 37 pines a la ACL - 8112PG. Dicha tarjeta adicional consiste en una especie de regleta donde se puede acceder fácilmente cada una de los canales a la vez que aporta la electrónica necesaria para acondicionar las señales. La tarjeta en cuestión es la ACL – 9138 y la disposición de los distintos canales y señales de la ACL - 8112PG se muestran en la siguiente figura:

PIN	DESCRIPCIÓN
1 a 8	Canal 0 a 7 de entrada analógica
9 – 10	Toma de tierra de los canales de entrada analógica
11	Referencia de tensión externa 2
13	Salida de +12 V
14	Toma de tierra de los canales de entrada analógica
15	Toma de tierra de los canales de salida digital
16	Señal de salida del contador 0
17	Señal de disparo externo
18	No conectada
19	+5V
20 a 27	Canal 8 a 15 de entrada analógica
28 – 29	Toma de tierra de los canales de entrada analógica
30	Canal 0 de salida analógica
31	Referencia de tensión externa 1
32	Canal 1 de salida analógica
33	Entrada de reloj para 8254
34	Entrada para 8254
35 – 36	No conectada
37	Entrada de señal de reloj externo
38	Toma de tierra de los canales de salida analógica

Tabla 1. Conexiones de la tarjeta ACL - 9138

En la práctica que nos ocupa, los pines que nos van a interesar son los correspondientes a las entradas analógicas, 1 a 8 (y los correspondientes de tierra, 9 a 10) para leer la señal de salida del sistema, y los de salida analógica 30 y 32 (y su señal de tierra correspondiente, 38) para aplicar la acción de control generada. Más adelante veremos cómo conectar cada una de las señales del sistema a la tarjeta ACL – 9138.

4. SOFTWARE DE COMUNICACIÓN CON LA TARJETA ACL - 8112PG: EXTENDED REAL TIME TOOLBOX PARA MATLAB / SIMULINK

4.1. Descripción del software

La tarjeta de adquisición ACL – 8112PG posee sus propios drivers, librerías en C y DLL's para ser programada desde casi cualquier plataforma. La forma más usual de trabajar con esta tarjeta es realizar un programa en C que realice el algoritmo de control. Dicha solución es la ideal de cara a presentar al usuario final una aplicación completa de adquisición de datos y control pero puede resultar incómoda cuando se está desarrollando el prototipo. En otras palabras, mediante un entorno de programación podemos manipular la tarjeta e implementar el control necesario pero no se nos permite analizar directamente los resultados, tarea esta muy usual cuando se está en la etapa de desarrollo. Sería más conveniente realizar esta etapa de desarrollo donde las pruebas y análisis de resultados son muy habituales en un entorno que nos permitiera tanto visualizar las señales en tiempo real así como el análisis de los datos en tiempo real con el fin de mejorar el control propuesto y/o detectar posibles errores o fallos. Uno de los entornos que ofrece este tipo de prestaciones es el paquete CACSD Matlab / Simulink con la ayuda del Real Time Workshop o Extended Real Time Toolbox. Ambos paquetes realizan la misma tarea,

permitir la comunicación directa entre Matlab / Simulink y la tarjeta de adquisición de datos. En nuestro caso se ha elegido el Extended Real Time Toolbox de la empresa Humusoft. Este paquete permite trabajar con la tarjeta de adquisición de datos directamente desde Matlab e incluso desde esquemas Simulink, ofreciendo así una gran potencia y facilidad en la etapa de desarrollo de un sistema de control.

En este punto se van a presentar las características principales del paquete Extended Real Time Toolbox (a partir de ahora RTT) así como una introducción a su funcionamiento y su método de trabajo. Obviamente, dicho paquete presenta ofrece muchas más posibilidades de las que aquí se presentan, pero al igual que con la tarjeta de adquisición de datos, no es este el lugar para describir ampliamente el funcionamiento de dicho paquete. Para más información se puede consultar el manual en línea del mismo.

Como se ha comentado, el Extended Real Time Toolbox permite la comunicación directa entre el sistema de adquisición de datos y Matlab / Simulink, es decir, permite acceder a la tarjeta desde la línea de comando de Matlab o directamente desde un esquema Simulink. Por su simplicidad, se elegirá la segunda opción de forma que se pueda acceder directamente a la tarjeta desde Simulink. Para ello el RTT dispone de una librería de bloques Simulink que realizan dicha tarea. Dicha librería se denomina Real Time Toolbox Simulink Block Library (a partir de ahora RTLIB) y provee los bloques Simulink necesarios para comunicar el entorno con la tarjeta.

La utilización de la RTLIB en un esquema Simulink pasa por dos fases: primero la inclusión y configuración del driver del hardware de nuestra tarjeta y segundo la correcta utilización de los diferentes bloques de entrada/salida en tiempo real.

El acceso a dicha librería se puede realizar de dos formas: directamente desde Simulink o desde la línea de comandos de Matlab mediante el comando **`rtlib`**. Puesto que el acceso a la RTLIB desde Simulink difiere dependiendo de la versión de Matlab/Simulink, accederemos a dicha librería mediante el comando **`rtlib`**. Al ejecutar dicho comando aparecerá un ventana de Simulink como la que se muestra en la figura 2:

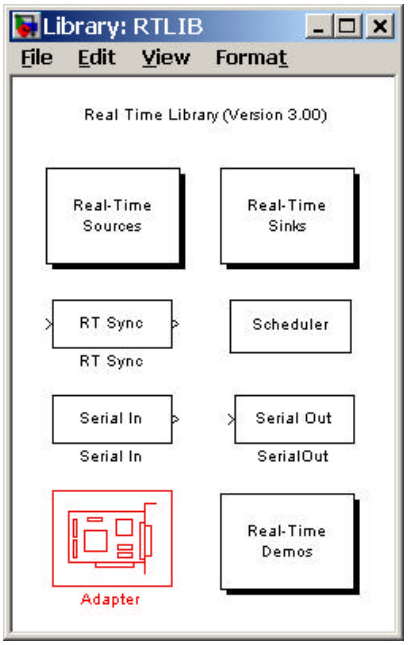

Figura 2. Librería RTLIB

A partir de aquí abriremos un nuevo esquema Simulink y podemos comenzar a trabajar con los bloques de la RTLIB. El primer paso es incluir en nuestro esquema un bloque adaptador. El bloque A*dapter* es un bloque que representa la tarjeta de adquisición de datos y hace las veces de driver o interfaz entre Simulink y la tarjeta. Se utiliza para disponer dentro del esquema Simulink de los canales de la tarjeta de adquisición de datos. Una vez incluido dicho bloque en el esquema Simulink, lo primero que hay que

hacer es configurar el mismo para que utilice el driver de la tarjeta que poseemos. Para ello haremos doble clic con el ratón sobre el mismo y se abrirá una ventana como la que se muestra en la figura 3.

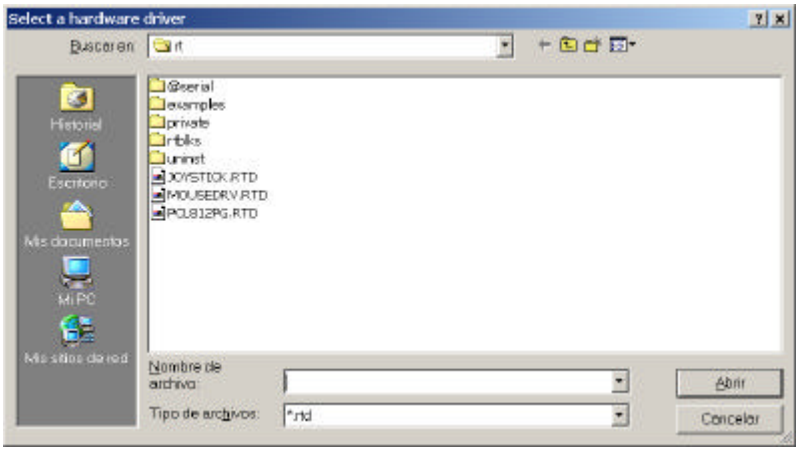

Figura 3. Ventana de selección de driver para el adaptador

En nuestro caso, el driver que debemos cargar es el correspondiente al fichero PCL812PG.RTD, para ello lo seleccionamos y pulsamos *'Aceptar'*. A partir de este momento ya tenemos cargado en nuestro esquema Simulink los drivers de la tarjeta ACL – 8112PG con las opciones por defecto. En el caso de querer cambiar dichas opciones haremos de nuevo doble clic sobre el bloque *Adapter* y se abrirá una ventana de selección de parámetros. En principio y a menos que se indique lo contrario, se trabajará con las opciones del driver por defecto por lo que no será necesario realizar este último paso.

Ahora ya podemos empezar a utilizar los bloques de entrada/salida de la librería. Se dispone de dos tipos de bloques: *sources* y *sinks*. Para acceder a los mismos haremos doble clic sobre los bloques *Real – Time Sources* o *Real – Time Sinks* respectivamente. De esta forma aparecerán las ventanas correspondientes tal y como se muestra en la siguiente figura:

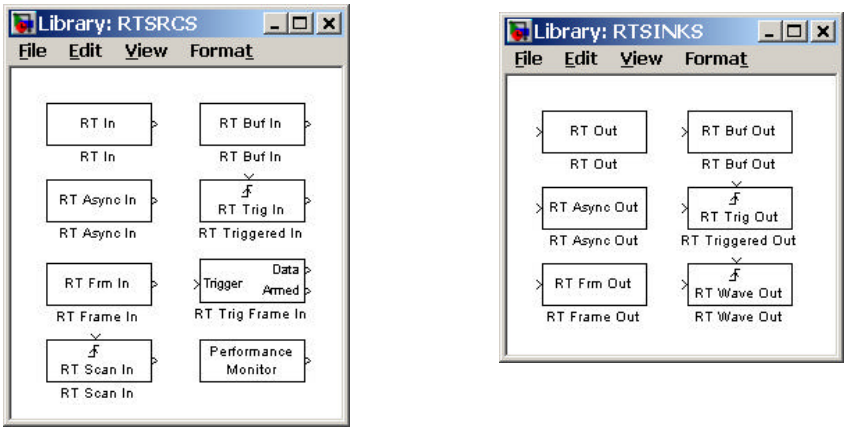

Figura 4. Bloques de Sources y Sinks de RTLIB

Como se puede apreciar aparecen multitud de bloques tanto en una como en otra ventana. En este documento sólo se van a analizar aquellos que se van a utilizar en la práctica. Dichos bloques son *'RT In'* y *'RT Out'* ya que son los mejores para realizar el control en tiempo real.

El bloque *'RT In'* está diseñado para la adquisición de datos en tiempo real, el procesamiento de señales, y las aplicaciones de control donde los datos han de ser procesados tan rápido como sea posible. Para

configurar este bloque haremos doble clic sobre el mismo, apareciendo así una ventana como la que se muestra a continuación:

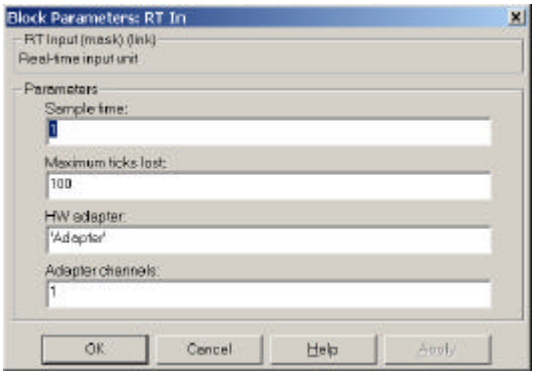

Figura 5. Ventana de configuración del bloque *RT In*

Veamos los parámetros configurables:

- *Sample Time:* especifica el periodo de muestreo al cual se debe realizar la adquisición de los datos. Se permite especificar un vector en el caso de que se quiera acceder a más de un canal con el mismo bloque. Asimismo permite establecer un *offset*.
- *Maximum ticks lost:* especifica el máximo número de 'tics' (muestreos) que se pueden perder antes de mostrar un mensaje de error. Esto es un mecanismo de "seguridad" para asegurar que la adquisición se ha realizado de forma correcta (no ha habido excesivas pérdidas de datos) ya que hay que tener en cuenta que este sistema en tiempo real se ejecuta sobre Matlab y Simulink, y éstos a su vez sobre Windows por lo que cabe la posibilidad de que en un determinado instante el sistema esté tan cargado que no se pueda realizar la adquisición con la frecuencia correcta. Esto, claro está, depende tanto de la potencia del microprocesador como de la cantidad de recursos de la máquina que están ocupados (aplicaciones, ventanas abiertas, etc.).
- *HW adapter:* indica el nombre del bloque *Adapter* incluido en el esquema Simulink. Esto es así pues cabe la posibilidad de tener en un mismo esquema más de un adaptador para gestionar dos sistemas de adquisición distintos (por ejemplo uno de entradas analógicas y otro de salidas analógicas independientes).
- *Adapter channels:* especifica el canal analógico de entrada al que se quiere acceder. En este caso, puesto que son canales de entrada analógica tendremos 8 (ya que la ACL – 811PG tiene 8) numerados del 1 al 8 correspondiendo a los canales de entrada analógica 0 a 7 de la tarjeta de adquisición. Es posible especificar un vector de canales por si se quiere acceder a más de uno de ellos desde el mismo bloque.

Los parámetros a fijar en el bloque *RT Out* son idénticos a los anteriores (y por tanto la ventana de configuración) con la salvedad de que en este caso se trata de un bloque de salida y se emplea para generar las señales necesarias con el mínimo retardo posible. En este caso, el parámetro *Sample Time* especifica el periodo al cual se actualiza el canal de salida correspondiente en la ACL – 8112PG, *Maximum ticks lost*, el número de veces que no se ha podido actualizar dichos canales debido a la carga del sistema, *HW adapter* es idéntico al caso anterior, y *Adapter channels* especifica el canal de salida analógica al que se accede que en este caso son 2, numerados del 1 al 2 correspondiendo a los canales de salida analógica 0 a 1 de la tarjeta ACL – 8112PG.

4.2. Metodología de trabajo con *Extended Real – Time Toolbox.*

Antes de realizar las primeras pruebas de adquisición y generación de señales es importante matizar algunos aspectos.

El rango de tensiones de salida y de entrada soportables por la tarjeta (en nuestro caso 0 a 10V. y –10 a 10V. respectivamente) se transforma en una rango de –1 a 1 en los bloques del RTT. Esto implica realizar

una transformación previa para obtener a la salida la tensión deseada. Así, si necesitamos generar una tensión de 5 voltios en un canal de salida, el valor que debemos indicar al bloque de salida *RT Out* será un valor numérico de 0, de esta forma es necesario realizar antes del bloque de salida la transformación siguiente:

De la misma forma, si leemos desde un bloque de entrada *RT In* un valor de 0, éste corresponderá a una tensión de 5 V. y por tanto es necesario realizar después del bloque de entrada la transformación siguiente:

Dicha transformación se realizará en el bloque Simulink mediante un bloque de función *Fcn* de forma que las funciones que se deben realizar son:

- Antes del bloque de salida: (u(1)/5)-1
- Después del bloque de entrada: (u(1)*10)

Además, por problemas, errores o imperfecciones del driver de la tarjeta ACL – 8112PG, el modo en que el sistema inicializa la tarjeta cuando se inicia una simulación (en este caso la ejecución en tiempo real del esquema Simulink) es poniendo las salidas analógicas de la misma a una tensión de 5 voltios en vez de a los 0 voltios esperados. Por esto, cuando ejecutemos el esquema Simulink (más adelante veremos como hacerlo), es necesario que la señal de referencia empiece a actuar unos dos segundos después de iniciar el programa. Para ello, los bloques que generan esta señal de referencia (por ejemplo un escalón que es lo más habitual) deben comenzar con un valor inicial de 0 y a los dos segundos generar el valor de referencia deseado. Por ejemplo para una referencia en escalón de 1 voltio la configuración de dicho bloque será:

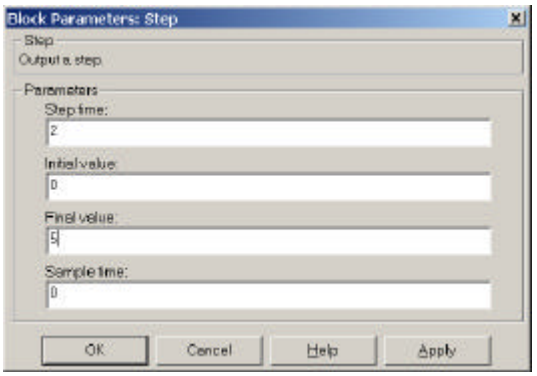

Figura 6. Configuración del escalón de referencia

De esta forma estamos indicando que el valor inicial será 0 y que a los dos segundos el valor inicial será el de la referencia deseada, es decir, 5. Notemos también que el último parámetro *Sample Time* puede permanecer a 0 ya que el periodo de muestreo al que se va a realizar la conversión se especifica en el bloque *RT Out*.

También es característica deseable que todos los bloques que aparecen el esquema Simulink sean de naturaleza discreta (excepto los bloques de generación de señal si fuera necesario) ya que la ejecución es más exacta.

Pasemos ahora a construir un primer esquema para afianzar lo expuesto hasta ahora. El primer ejemplo va a consistir en adquirir datos por el canal 0 de entrada analógica y generar una tensión de 3 voltios por el canal 0 de salida analógica (recordemos que ambos canales se denominan 1 en los bloques de RTT). El periodo de muestreo debe ser de 0.01 seg. tanto para la entrada como para la salida de datos. Además, se desea visualizar la señal adquirida mediante un osciloscopio.

Dicho esquema tendría la forma siguiente:

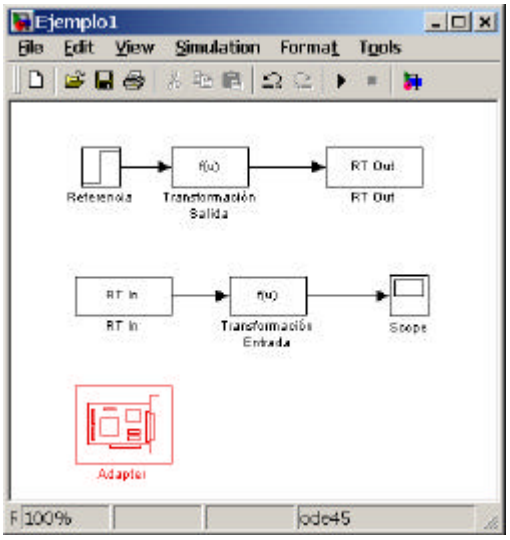

Figura 7. Esquema básico de generación y lectura de señales

En este esquema, la señal de referencia se genera tal y como se ha explicado previamente, los bloques de ganancia de conversión implementan las funciones de transformación ya vistas. Ahora lo único que resta es ejecutar dicha simulación. Los parámetros de simulación idóneos son los siguientes:

- Solver options – Type: Fixed – Step (discrete (no continuous states))
- Fixed step size: 0.01 (periodo de muestreo al que realizamos la adquisición)

Por supuesto, en los bloques *RT Out* y *RT In*, debemos especificar tanto el canal de entrada o salida (en ambos caso el canal 0 que corresponde al *Adapter Channel* 1) y el periodo de muestreo (0.01 seg.).

Si ejecutamos la simulación teniendo el motor conectado a la tarjeta de adquisición de datos, de forma que la señal que se lee es una tensión proporcional a la velocidad del mismo (señal del tacogenerador), lo que se está haciendo es inyectar al motor una entrada de 3 voltios y recogiendo la salida en velocidad. Así, la señal que debe aparecer en el osciloscopio es algo parecido a lo que se muestra en la figura siguiente:

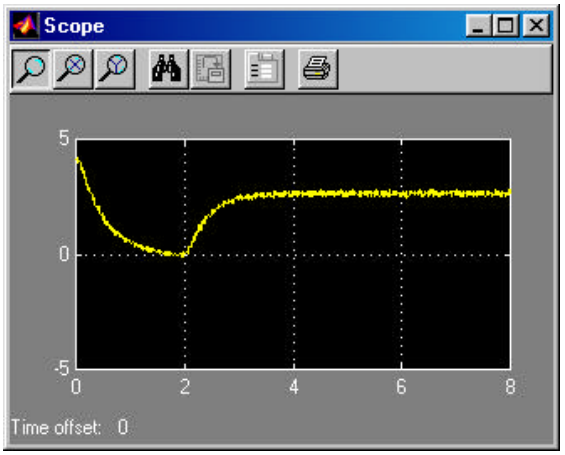

Figura 8. Visualización de la adquisición de datos en un oscilioscopio

Como vemos, debido al un error en los drivers, inicialmente el valor de la salida es la correspondiente a la un escalón de entrada de 5 voltios. Casi inmediatamente dicho escalón pasa a cero voltios de valor inicial

de la referencia y el motivo de esperar dos segundos a generar el escalón de 3 voltios es esperar a que el motor pare. Transcurridos dos segundos aparece el escalón de 3 voltios a la entrada del motor y se genera la salida que se observa en la gráfica anterior. Si se hubiera recogido dicha señal en un bloque *To Workspace* que es como se debe hacer ya que el bloque osciloscopio consume muchos más recursos y hace la ejecución más lenta, podríamos representar la señal mediante el comando *plot*. Además, puesto que sabemos el número de muestras que se recogen en los 2 segundos iniciales puesto que conocemos el periodo de muestreo, (nº muestras = 2 / T), podemos eliminar del vector en el que se almacena la señal los datos correspondientes a los 2 primeros segundos. En este caso, el número de muestras recogidas en los dos primeros segundos es 2/0.01 = 200. Puesto que la ejecución ha durado 8 segundos, se han recogido en total 800 muestras. Para eliminar las 200 primeras muestras de la variable en la que están almacenadas, desde la línea de comandos de Matlab haremos lo siguiente:

```
» variable = variable(201:800)
```

De esta forma ya se pueden analizar y representar los datos como si se hubiera realizado el escalón de referencia en el instante cero de la ejecución. La representación de esta última señal se muestra a continuación:

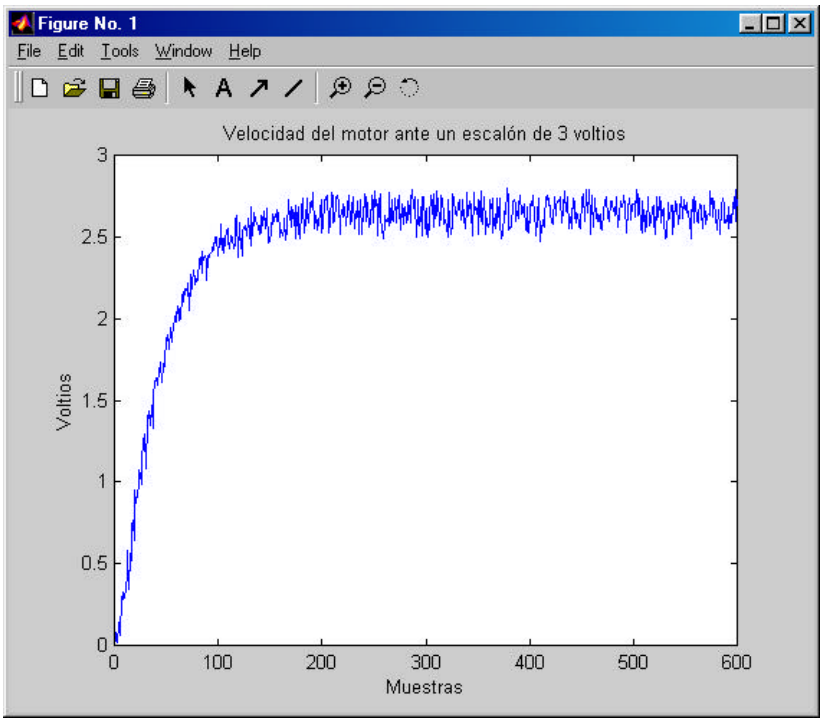

Figura 9. Representación gráfica de la respuesta una vez eliminados los dos segundos iniciales-

Para ver que la señal aplicada al motor son realmente los 3 voltios generados se puede utilizar un voltímetro u osciloscopio.

Otro aspecto a destacar es que la tarjeta de adquisición de datos, una vez ejecutado el programa, mantiene en sus canales de salida el último valor (en el caso anterior 2 voltios) y es recomendable (por no decir necesario) dejar el valor de la salida de estos canales a 0 voltios. Para ello se hará lo siguiente: suponiendo que la ejecución (tiempo de simulación) dura 8.5 segundos, la referencia a generar será la suma de las señales de dos bloques escalón. El primero será idéntico al anterior, y el otro será un escalón con valor inicial 0, y a los 8 segundos generará un valor final opuesto en signo al valor final del primero (en nuestro ejemplo –3 voltios). De esta forma, al ser la referencia la suma de ambos hará que a los ocho segundos la referencia sea 0 y por tanto se permita parar el motor. El esquema Simulink resultante sería:

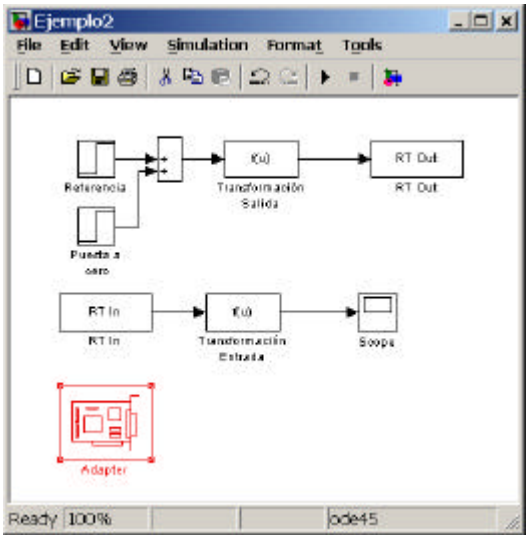

Figura 10. Esquema de generación y lectura de datos con puesta a cero

Al igual que en el caso anterior, puesto que conocemos el número de muestras que se toman, podemos eliminar aquellas que no nos interesan. Notemos que este tipo de aspectos sólo es importante cuando se desea analizar los datos obtenidos ya que para el modo de operación normal, los instantes iniciales y finales apenas importan y más cuando el tiempo de ejecución es muy elevado o infinito.

5. REALIZACIÓN DE LA PRÁCTICA

En este punto se va a describir tanto la conexión de los distintos elementos así como el proceso seguir para la identificación del motor y la realización de un regulador PID discreto para el mismo.

5.1. Conexión de los componentes

El material que se suministra para la realización de la práctica es:

- Computador con la tarjeta ACL – 8112PG y tarjeta de conexiones ACL – 9138
- Servomotor con fuente de alimentación
- Tarjeta de expansión ADAM 3937, dispone de las entradas y salidas de las señales del servomotor
- Cables de conexión
- Destornillador
- Multímetro y osciloscopio en el caso de que sean necesarios

Puesto que tanto la identificación como el control se van a hacer en velocidad, nos limitaremos a describir las conexiones necesarias para ello sin entrar en detalle de la descripción de las diferentes señales que pueden suministrar los distintos elementos utilizados.

Las conexiones que se deben realizar son:

Pin ACL - 9138	**Servomotor (** tarjeta ADAM 3937))
1 (canal 0 de entrada analógica)	Señal TACHO + (pin 35)
10 (GND de entrada analógica)	Señal de tierra del propio servomotor (pin 11)
30 (canal 0 de salida analógica)	Señal PA+ INPUT (pin 33)
GND (GND de salida analógica)	Señal PA- INPUT (pin 14)

Tabla 2. Conexiones entre el servomotor y ACL - 9138 para lectura de velocidad

La señal TACHO + proporciona una tensión entre 0 y 10 voltios proporcional a la velocidad del motor. La masa de dicha señal se tomará del propio servomotor ya que de esta forma se obtiene una señal mucho más "limpia" pues la electrónica de la caja siempre introduce algo de ruido.

La señal PA+ INPUT (pin 33) es la entrada al servomotor y es donde se aplicará por tanto la acción de control. La masa de dicha señal se tomará del terminal PA- INPUT (pin 14).

NOTA: Todas las conexiones se deben realizar con la fuente de alimentación desconectada y se deben repasar las mismas antes de conectar. Además, siempre que no se esté ejecutando un programa sobre el motor, la fuente de alimentación debe permanecer también desconectada.

5.2. El proceso de identificación

El proceso de identificación pretende caracterizar matemáticamente la función de transferencia del proceso mediante la respuesta del sistema, obtenida en la entrada del convertidor A/D, como resultado de aplicar una tensión de excitación o control generada por el computador mediante el convertidor D/A.

Para realizar la identificación de este sistema se debe estudiar su respuesta en bucle abierto frente a una entrada en escalón. Puesto que el rango de tensiones que se puede suministrar va de 0 a 10v, se deben tomar medidas de la respuesta del sistema para una entrada de 5V. utilizando para ello el esquema propuesto en el apartado 2.3.2. Una vez obtenida esta medida, se puede comprobar que se aproxima bastante a un sistema de primer orden cuya función de transferencia se puede expresar como

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

donde k es la ganancia del sistema y se obtiene como el cociente del valor final alcanzado por la señal de salida y la magnitud del escalón aplicado al sistema y la constante de tiempo τ corresponde al tiempo transcurrido desde el inicio del escalón hasta que la salida alcanza en 63%, aproximadamente, del valor final. Es decir, estimando estos parámetros podemos modelar matemáticamente el proceso y por tanto hacerlo susceptible de ser analizado y tratado con las técnicas habituales.

No obstante no es suficiente hallar la función de transferencia evaluando los resultados obtenidos ante un único escalón. Se debe comprobar también la linealidad del sistema, es decir, comprobar si con escalones de entrada de distinta magnitud, se obtiene la misma función de transferencia. Para ello, obtendremos la función de transferencia para escalones de 1, 3, 5, 7 y 9 voltios (por ejemplo). A partir de las respuestas obtenidas (en el caso de que el proceso no sea lineal, y siempre es no lineal en mayor o menor medida) debemos elegir una función de transferencia que sea un compromiso entre todas ellas.

En prácticas anteriores se ha visto como hallar la función de transferencia cuando se conoce la magnitud del escalón de entrada y la secuencia de salida.

TAREA

La tarea a realizar consiste en hallar la función de transferencia del motor sin freno y con el freno en su posición máxima

Para tomar los datos de cada uno de los experimentos (para escalones de distinta magnitud) utilizaremos el esquema siguiente:

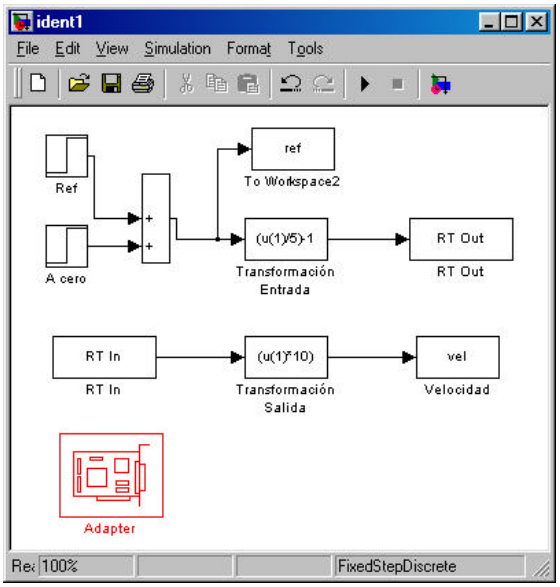

Figura 11. Esquema básico para la toma de datos e identificación del sistema

Como se puede apreciar, el esquema es bastante parecido al estudiado anteriormente con la particularidad de que se ha sustituido el bloque *Scope* por bloques *To Workspace*. Como se ha comentado antes, esto permite por una parte almacenar los datos adquiridos, y por otra mejora el rendimiento de la ejecución pues este tipo de bloques consume menos recursos que el bloque *Scope*. En cuanto al periodo de muestreo, puesto que a priori no conocemos las características del sistema, caben dos opciones: o bien se realiza una prueba donde se pueda medir la respuesta mediante un dispositivo continuo (p.e. osciloscopio) lo cual nos permite hallar los parámetros que van a determinar el periodo de muestreo correcto (constante de tiempos o frecuencia de oscilación), o bien optamos por utilizar un periodo de muestreo que consideremos lo suficientemente pequeño (por ejemplo el máximo que permita nuestro sistema de adquisición de datos). Puesto que el servomotor con el que trabajamos ya ha sido convenientemente identificado y analizado en continuo, podemos decir que un periodo de muestreo de 0.01 seg. es más que suficiente y por tanto este será el periodo de muestreo al que realizaremos la adquisición de los datos y que por consiguiente especificaremos tanto en aquellos bloques que lo requieran como en los parámetros de simulación.

Una vez obtenidas las funciones de transferencia buscadas, debemos calcular las equivalentes discretas. Para ello podemos utilizar el comando de Matlab **`c2dm`** (Continuous To Discrete with Method). Dicho comando tiene la siguiente sintaxis:

```
[numd, dend] = c2dm(num, den, Ts, 'method')
```

de esta forma se convierte la función de transferencia continua *G(s) = num(s)/den(s)* en la función de transferencia discreta *G(z)=numd(z)/dend(z)* a un periodo de muestreo *Ts* utilizando el método de discretización *'method'*, donde *num*, *den*, *numd*, *dend* son los vectores cuyos elementos son los coeficientes en orden decreciente de los polinomios numerador y denominador de las funciones de transferencia continua y discreta respectivamente. En cuanto a los métodos de dicretización, en nuestro caso utilizaremos el que antepone un retenedor de orden cero a la entrada del sistema, *'zoh'*, que es el que nos interesa en este tipo de aplicaciones.

Por ejemplo, si deseamos obtener el equivalente discreto del sistema $G(s) = \frac{9}{s+10}$ a un periodo de muestreo de T = 0.01 seg. escribiremos lo siguiente en la línea de comandos de Matlab:

```
» [numd, dend]=c2dm([9], [1 10], 0.01, 'zoh')

numd =
```

```
         0    0.0856
dend =
    1.0000   -0.9048
```

Con lo que la función de transferencia discreta equivalente es $G(z) = \frac{0.0856}{z - 0.9048}$.

TAREA

Obtener mediante simulación las respuestas de las funciones de transferencia obtenidas experimentalmente tras la identificación y las obtenidas mediante la discretización de éstas. Comparar y comentar los resultados obtenidos.