

# Manual de Usuario Librería SITAU SAFT



**DASEL, SL**  
**Avda. de Madrid, 84**  
**Arganda del Rey**  
**28500 MADRID**  
**[info@daselsistemas.com](mailto:info@daselsistemas.com)**  
**[www.daselsistemas.com](http://www.daselsistemas.com)**

**Edición:** 1.2  
**Revisión:** 1  
**Fecha:** 04/08/2009

## ÍNDICE

<b>1</b>	<b>ACTUALIZACIONES .....</b>	<b>3</b>
<b>2</b>	<b>FUNCIONES .....</b>	<b>4</b>
2.1	PROGRAMACIÓN DE PARÁMETROS DE ADQUISICIÓN.....	4
	<i>INT ST_SetGain (FLOAT data) .....</i>	4
	<i>INT ST_SetReceive (INT data) .....</i>	4
	<i>INT ST_SetSignalTest (INT data).....</i>	4
	<i>INT ST_SetSamplingFreq (USHORT data).....</i>	5
	<i>INT ST_SetAcqConfig (FLOAT *tadq, INT ltadq) .....</i>	5
	<i>INT ST_SetFocalLaw (FLOAT *dle, FLOAT *dlr, INT long).....</i>	6
	<i>INT ST_SetIniElement (USHORT pos) .....</i>	6
	<i>INT ST_SetChannelTXEnable (USHORT *htx, INT lhtx, INT enable) .....</i>	6
	<i>INT ST_SetTXCode (USHORT *cpul, INT lcpul) .....</i>	7
	<i>INT ST_SetPulser (INT nPulsos, INT nBitSecuencia, FLOAT anchoPulso).....</i>	7
	<i>INT ST_EnableCat (USHORT data) .....</i>	7
	<i>INT ST_SetCat(FLOAT *Gan, FLOAT *Time, USHORT lvectorGan).....</i>	8
	<i>INT ST_SaveConfig (CHAR *file).....</i>	8
	<i>LONG ST_LoadConfig (CHAR *file, CHAR *date).....</i>	8
2.2	FUNCIONES DE ADQUISICIÓN DE DATOS. ....	9
	<i>INT ST_GetNumData (USHORT *nmu, INT lnmu, ULONG *numdata).....</i>	9
	<i>INT ST_RDTemp (BYTE *tmax) .....</i>	9
	<i>INT ST_GetVersion (CHAR *ver).....</i>	9
	<i>INT ST_RunAcq (SHORT *data, INT ldata) .....</i>	10
2.3	FUNCIONES DE ACCESO A LA LIBRERÍA. ....	10
	<i>INT ST_OpenSys (CHAR * path_bin, USHORT nun_channel, USHORT nun_channel_mux).....</i>	10
	<i>INT ST_CloseSys (void) .....</i>	10
<b>3</b>	<b>CONFIGURACIÓN INICIAL POR DEFECTO .....</b>	<b>10</b>
<b>4</b>	<b>TIPOS DE DATOS .....</b>	<b>11</b>
<b>5</b>	<b>CÓDIGOS DE ERROR.....</b>	<b>11</b>
<b>6</b>	<b>ESQUEMA DE PROGRAMACIÓN.....</b>	<b>13</b>
<b>7</b>	<b>EXPLICACIÓN DETALLADA DE LA RECUPERACIÓN DE UNA ADQUISICIÓN .....</b>	<b>13</b>

# 1 Actualizaciones

Versión	Fecha	Actualización
1.2	29/01/09	- Introducción de las funciones ST_GetHILO, ST_GetCATEnable y modificación de ST_SetGain.
1.1.1	24/09/08	- Cambios en las funciones de guardar y salvar configuración del sistema: ST_SaveConfig, ST_LoadConfig
1.1.0	11/07/08	- Funciones de control de las temperaturas. - Función para el control de versiones.
1.0.0	11/07/08	- Generación de archivo LOG con los errores generados. - Generación de un archivo con los detalles de la configuración del sistema. - Nuevos códigos de errores. - Load y Save Config para guardar y cargar la configuración del equipo. - Implementación de la curva de Ganancia. - Definición de los parámetros por defecto.
0.0.0	09/07/08	- Versión inicial del documento.

## 2 Funciones

### 2.1 Programación de parámetros de adquisición.

Los parámetros configurados con este tipo de funciones se cargan directamente al sistema sin necesidad de ejecutar ninguna función posterior.

<b>INT ST_SetGain (FLOAT data)</b>	
Configura el valor de la ganancia.	
<b>Parámetros</b>	
data (dB)	(FLOAT) Ganancia. [0 dB .. 60 dB]
<b>Devuelve (INT)</b>	
(Ver códigos de error)	
<b>Dependencias</b>	
- Variaciones en la Ganancia pueden provocar cambios en la curva CAT (hilo).	

<b>INT ST_SetReceive (INT data)</b>	
Configura el modo de recepción en transmisión o en pulso-eco.	
<b>Parámetros</b>	
Data	(USHORT) 0 -> Modo pulso-eco. 1 -> Modo transmisión.
<b>Devuelve (INT)</b>	
(Ver códigos de error)	

<b>INT ST_SetSignalTest (INT data)</b>	
La señal recibida por el sistema será la de test (un diente de sierra)	
<b>Parámetros</b>	
Data	(USHORT) 0 -> Señal de test deshabilitada. 1 -> Señal de test habilitada.
<b>Devuelve (INT)</b>	
(Ver códigos de error)	

### INT ST\_SetSamplingFreq (USHORT data)

Configura la frecuencia de muestreo.

#### Parámetros

data	(USHORT)
	<p>Factor de Diezmado (D), valor comprendido entre 1 y 16.</p> <p style="text-align: center;">Frecuencia de Muestreo = 40 MHz / D</p> <p>data = 1 --&gt; fMuestreo = 40 MHz  data = 2 --&gt; fMuestreo = 20 MHz  data = 3 --&gt; fMuestreo = 13,3 MHz  data = 4 --&gt; fMuestreo = 10 MHz  data = 5 --&gt; fMuestreo = 8 MHz  data = 6 --&gt; fMuestreo = 6,6 MHz  data = 7 --&gt; fMuestreo = 5,7 MHz  data = 8 --&gt; fMuestreo = 5 MHz  data = 9 --&gt; fMuestreo = 4,4 MHz  data = 10 --&gt; fMuestreo = 4 MHz  data = 11 --&gt; fMuestreo = 3,6 MHz  data = 12 --&gt; fMuestreo = 3,3 MHz  data = 13 --&gt; fMuestreo = 3 MHz  data = 14 --&gt; fMuestreo = 2,8 MHz  data = 15 --&gt; fMuestreo = 2,6 MHz  data = 16 --&gt; fMuestreo = 2,5 MHz</p>

#### Devuelve (INT)

(Ver códigos de error)

#### Dependencias

- Modifica los ticks de reloj del vector dlr (retardos en emisión), de esta forma, se mantienen constantes los tiempos introducidos por el usuario.
- Modifica el número de muestras adquiridas por cada canal, de esta forma, se mantiene constante el tiempo de adquisición.

### INT ST\_SetAcqConfig (FLOAT \*tadq, INT ltadq)

Configura el tiempo de adquisición para cada canal del sistema expresado en  $\mu$ s. Colocando un 0 en el canal elegido, se deshabilita la recepción por el mismo.

#### Parámetros

*tadq ( $\mu$ s)	(*FLOAT)
	<p>Vector de los tiempos de adquisición de cada canal activo. Se introduce un tiempo de adquisición por canal. Ordenado del canal 0 al canal más alto.</p>
ltadq	(INT)
	<p>Longitud del puntero (*data). Debe corresponderse con el número de canales activos del sistema.</p>

#### Devuelve (INT)

(Ver códigos de error)

#### Dependencias

- Modifica el número de muestras que adquirirá el sistema por cada canal.

### ***INT ST\_SetFocalLaw (FLOAT \*dle, FLOAT \*dlr, INT long)***

Programa en el sistema los valores de los vectores dle y dlr (retardos de emisión y recepción) expresados en  $\mu$ s

#### **Parámetros**

*dle (en $\mu$ s)	(*FLOAT) Vector de retardos de emisión expresados en $\mu$ s.
*dlr (en $\mu$ s)	(*FLOAT) Vector de retardos de recepción expresados en $\mu$ s.
long	(INT) Longitud de los vectores, ambos deben tener la misma longitud y tiene que ser igual a la del número de canales activos del sistema.

#### **Devuelve (INT)**

(Ver códigos de error)

### ***INT ST\_SetIniElement (USHORT pos)***

Coloca los multiplexores en la posición correspondiente al primer elemento de la apertura. Las posiciones van desde la número 1 hasta el número de elementos multiplexados del sistema menos el número de elementos activos.

#### **Parámetros**

pos	(USHORT) Elemento inicial de la apertura
-----	---

#### **Devuelve (INT)**

(Ver códigos de error)

### ***INT ST\_SetChannelTXEnable (USHORT \*htx, INT lhtx, INT enable)***

Programa en el sistema los canales habilitados en emisión.

#### **Parámetros**

*data	(USHORT) Puntero al vector de 0s y 1s con el que se indica que canales estarán en la emisión. Un 1 significa activo en emisión, un 0 indica que el canal correspondiente no se activa para la emisión.
ldata	(INT) Longitud del vector (*data), debe coincidir con el número de canales activos del sistema.
enable	(INT) Control global de emisión. Si vale '0' se deshabilita la emisión por todos los canales, si su valor es '1', se habilitan los canales activos según los valores del vector (*data).

#### **Devuelve (INT)**

(Ver códigos de error)

### **INT ST\_SetTXCode (USHORT \*cpul, INT lcpul)**

Programa en el sistema los códigos de excitación del pulser para cada canal (CPUL).

#### **Parámetros**

	(*USHORT)
*cpul	Puntero al vector códigos de excitación del pulser para cada canal (CPUL). Cada elemento del vector representa la secuencia de excitación del pulser para el canal correspondiente expresada en 16 bits. Un 0 se corresponde con 0V y un 1 con HV. El tiempo de duración de cada bit, la cantidad de bits utilizados (como máximo 16) y el número de repeticiones de la secuencia, se configuran con la función ST_SetPulser().
lcpul	(INT) Longitud del vector cpul, debe coincidir con el número de canales activos del sistema.

#### **Devuelve (INT)**

(Ver códigos de error)

#### **Dependencias**

- El código que finalmente emita cada canal también depende del parámetro "nBitSecuencia" de la función "ST\_SetPulser"

### **INT ST\_SetPulser (INT nPulsos, INT nBitSecuencia, FLOAT anchoPulso)**

Configura los parámetros que definen el código de emisión (CPUL):

- El número de veces que se repite la secuencia de excitación del pulser,
- El número de bits del código de emisión,
- Y la duración de cada bit del código de emisión.

Estos tres parámetros son comunes a todos los canales.

#### **Parámetros**

nPulsos	(INT) Indica el número de veces que se repite la secuencia de excitación del pulser, definida para cada canal en el vector de códigos de emisión (CPUL).
nBitSecuencia	(INT) Indica el número de bits utilizado del código de emisión.
anchoPulso (µs)	(FLOAT) Tiempo de duración de cada bit del código de emisión.

#### **Devuelve (INT)**

(Ver códigos de error)

### **INT ST\_EnableCat (USHORT data)**

Función para la habilitación de la curva CAT

#### **Parámetros**

data	(*USHORT) 0 -> Curva CAT deshabilitada. 1 -> Curva CAT habilitada.
------	--

#### **Devuelve (INT)**

(Ver códigos de error)

### **INT ST\_SetCat(FLOAT \*Gan, FLOAT \*Time, USHORT lvectorGan)**

Carga la curva CAT de Ganancia en la memoria.

El valor para la curva CAT en t=0 es el primer valor de ganancia facilitado por el usuario, independientemente del tiempo especificado para el mismo.

El funcionamiento de la curva CAT es necesario habilitarlo mediante la función: *ST\_EnableCat()*

#### **Parámetros**

*Gan	(*FLOAT)
	Vector que contiene los valores en dB de la ganancia que el usuario desea en los instantes indicados en el vector Time. Los valores de Gan han de ir en el rango de 12 a 60 dB o bien de 0 a 48 dB.
*Time	(*FLOAT)
	Vector de tiempos que indica en qué instantes se desean los valores de ganancia del vector Gan. Todos los elementos han de ser positivos e ir en orden creciente.
lvectorGa	(USHORT)
	Longitud de los vectores Gan y Time. Debe ser menor de 2047.

#### **Devuelve (INT)**

(Ver códigos de error)

#### **Dependencias**

- Debe habilitarse el funcionamiento mediante *ST\_EnableCat()*.

### **INT ST\_SaveConfig (CHAR \*file)**

Guarda la configuración del sistema en un momento dado. Tras recuperar esa información el sistema debe quedar en el estado anterior a la ejecución de la función *ST\_SaveConfig*.

#### **Parámetros**

*file	(*CHAR)
	Cadena de caracteres con el nombre del fichero en el que se desea guardar la configuración actual del equipo.

#### **Devuelve (INT)**

(Ver códigos de error)

### **LONG ST\_LoadConfig (CHAR \*file, CHAR \*date)**

Guarda la configuración del sistema en un momento dado. Tras recuperar esa información el sistema debe quedar en el estado anterior a la ejecución de la función *ST\_SaveConfig*.

#### **Parámetros**

*file	(*CHAR)
	Cadena de caracteres con el nombre del fichero en el que se desea guardar la configuración actual del equipo.
*date	(*CHAR)
	Puntero a una cadena de caracteres en la que se devuelve la fecha en la que se guardó la configuración que ahora se carga en el equipo. La longitud mínima reservada debe ser 15.

#### **Devuelve (LONG)**

Devuelve la posición del último valor leído del archivo de configuración.

En caso de devolver un número negativo, se trata de un código de error.

#### **Dependencias**

- El fichero de lectura ha de ser salvado previamente mediante la función *ST\_SaveConfig*

	<b>Manual de Usuario Librería SITAU SAFT</b>	Versión: 1.2 Revisión: 1 Fecha: 04/08/2009
---	--	--

## 2.2 Funciones de adquisición de datos.

<b><i>INT ST_GetNumData (USHORT *nmu, INT Inmu, ULONG *numdata)</i></b>	
Esta función permite recuperar la siguiente información: - Cantidad de muestras adquiridas por cada uno de los canales del sistema (nmu). - El número de datos total de la adquisición (numdata).	
<b>Parámetros</b>	
*nmu	(*USHORT) Puntero al vector en el que se devuelve al usuario el número de muestras adquiridas por cada canal.
Inmu	(INT) Longitud del puntero nmu. Debe corresponderse con el número de canales activos.
*numdata	(ULONG) Puntero al parámetro en el que se devuelve el número total de datos de la adquisición, se corresponde con la suma del número de muestras adquiridas por todos los canales. Este valor es el tamaño mínimo de memoria necesario para obtener los datos de la adquisición con la función <b>ST_RunAcq()</b> .
<b>Devuelve (INT)</b>	
(Ver códigos de error)	

<b><i>INT ST_RDTemp (BYTE *tmax)</i></b>	
Esta función permite recuperar la siguiente información: - Temperatura máxima del sistema (tmax).	
<b>Parámetros</b>	
*tmax	(*BYTE) Puntero a un BYTE en el que se devuelve la temperatura máxima del sistema en °C.
<b>Devuelve (INT)</b>	
(Ver códigos de error)	

<b><i>INT ST_GetVersion (CHAR * ver)</i></b>	
Esta función permite recuperar la siguiente información: - Versión actual de la librería (ver)	
<b>Parámetros</b>	
*ver	(*CHAR) Puntero a una cadena de caracteres. Su longitud mínima debe ser 15.
<b>Devuelve (INT)</b>	
(Ver códigos de error)	

**INT ST\_RunAcq (SHORT \*data, INT ldata)**

Inicia el ciclo de adquisición.  
Es una función bloqueante, no finaliza hasta que la adquisición se ha completado.

**Parámetros**

*data	(*SHORT) Puntero a la zona de memoria donde se devolverán los datos de la adquisición.
ldata	(INT) Tamaño de la zona de memoria reservada (*data) para la recepción de los datos. El tamaño de memoria necesaria para devolver los datos de la adquisición se puede obtener con la función ST_GetNumData().

**Devuelve (INT)**

(Ver códigos de error)

### 2.3 Funciones de acceso a la librería.

**INT ST\_OpenSys (CHAR \* path\_bin, USHORT nun\_channel, USHORT nun\_channel\_mux)**

Abre la comunicación con el sistema, carga el firmware y configura el sistema con los valores por defecto.

**Parámetros**

*path_bin	(*CHAR) Cadena con la dirección en la que se encuentran los ficheros *.bit y *.bix necesarios para el sistema.
nun_channel	(USHORT) Nº canales de la apertura del sistema. Ha de ser múltiplo de 32
nun_channel_mux	(USHORT) Nº canales del sistema multiplexado. Si no es multiplexado el valor será 0.

**Devuelve (INT)**

(Ver códigos de error)

**Dependencias**

- Depende del valor de la ganancia.

**INT ST\_CloseSys (void)**

Cierra la comunicación entre el PC y el sistema

**Devuelve (INT)**

(Ver códigos de error)

## 3 Configuración inicial por defecto

Después de abrir la conexión con el SITAU y cargar el firmware (**ST\_OpenSys()**), se carga una configuración por defecto que tiene los siguientes valores:

<b>Adquisición</b>	
Frecuencia de muestreo	40 MHz
Tiempo de adquisición	2,5 µs
Código de emisión	0000000000000010
Número de bits activos del código de emisión	2
Ancho de pulso	100 ns.

Número de pulsos	1
Retardos en emisión	0
Retardos en recepción	0
Canales habilitados	Todos
Ganancia	0 dB
Elemento Inicial	1
Curva CAT	Deshabilitada

## 4 Tipos de datos

Tipo	Tamaño (Bytes)	Rango
UCHAR	1	0 ... 255
CHAR	1	-128 ... 127
SHORT	2	-32,768 ... 32,767
UINT	4	0 ... 4,294,967,295
INT	4	-2,147,483,648 ... 2,147,483,647
ULONG	4	0 ... 4,294,967,295
LONG	4	-2,147,483,648 ... 2,147,483,647
FLOAT	4	$\pm 1.18 \cdot 10^{-38} \dots \pm 3.40 \cdot 10^{38}$
DOUBLE	8	$\pm 2.23 \times 10^{-308} \dots \pm 1.79 \times 10^{308}$
LONG DOUBLE	10	$\pm 3.37 \times 10^{-4932} \dots \pm 1.18 \times 10^{4932}$

## 5 Códigos de error

Código	Definición
0	OK, resultado de la operación correcto, no se ha producido ningún error.
-1	Handle no nulo en OpenUsb.
-2	Nombre de fichero inválido en LoadFirmware_FX2.
-3	No se puede abrir el fichero del firmware en LoadFirmware_FX2.
-4	No se pudo cerrar el fichero en LoadFirmware_FX2.
-5	Nombre de fichero inválido en LoadFirmware_UCI.
-6	No se puede abrir el fichero del .bit de la UCI en LoadFirmware_UCI.
-7	No se pudo cerrar el fichero en LoadFirmware_UCI.
-8	Argumentos no válidos en inportBloque.
-9	Falla la cuenta de words_restantes en inportBloque.
-10	Argumentos no válidos en outportBloque.
-11	Argumentos no válidos en Ready.
-12	Argumentos no válidos en is_cypress_on.
-13	No se pudo reservar memoria para bufferdev en is_cypress_on.
-14	No se pudo abrir el fichero de log.
-15	Error en la llamada a CreateFile.
-16	Error en la llamada a CloseHandle.
-17	Error en la llamada a DeviceIoControl.

-101	No se pudo reservar memoria en Escribir.
-102	Argumentos no válidos en Escribir.
-103	Argumentos no válidos en Leer.
-104	Argumentos no válidos en LoadFirmware.
-105	No se pudo abrir el fichero .bit en LoadFirmware.
-106	No se pudo cerrar el fichero .bit en LoadFirmware.
-201	Módulo incorrecto en Test_UCI.
-202	Argumentos no válidos en InsertarPrograma.
-203	El programa de la UCI es demasiado largo (InsertarEnPrograma).
-204	Argumentos no válidos en InicioSubrutina.
-205	No se puede iniciar otra subrutina, se ha alcanzado ya el máximo (InicioSubrutina).
-206	Argumentos no válidos en FijarInicioPrograma.
-207	Argumentos no válidos en CrearEscrituraMultipleRegistro.
-208	Argumentos no válidos en PRG_jump_set.
-209	Argumentos no válidos en CMD_sub_set.
-210	Argumentos no válidos en PRG_jump_cond_set.
-211	Argumentos no válidos en PRG_jump_c_get.
-301	Error de archivo, acceso al archivo incorrecto
-302	Error de rango, la variable pasada como parámetro está fuera de rango
-303	Error de número de módulo, el módulo al que desea acceder está fuera de rango
-304	Error de número de canal, el canal al que se desea acceder está fuera de rango
-401	Error de archivo, acceso al archivo incorrecto
-402	Error de rango, la variable pasada como parámetro está fuera de rango
-403	Error de número de base, la base a la que se desea acceder está fuera de rango
-404	Error en la reserva de memoria
-405	Error Se ha intentado acceder a una posición no válida del vector
-501	Una base no se ha configurado correctamente
-502	Un módulo no se ha configurado correctamente
-503	Se ha llamado a ST_ConfigSys sin una llamada previa a ST_OpenSys
-504	Se ha llamado a alguna función sin una llamada previa a ST_ConfigSys
-505	Rango de datos no valido
-506	Todo el buffer leído está compuesto por 0xCACA, implica que aún no se han adquirido datos válidos se ha leído demasiado pronto
-507	Se ha salido de un bucle por timeout
-508	Error en la reserva de memoria
-509	Error trama incorrecta en el buffer del canal, la trama leída del buffer no tiene la palabra de control
-510	Error trama incorrecta en el buffer del canal, la trama leída del buffer del canal indica que contiene un numero de datos superior al tamaño de la trama
-511	Error desbordamiento de buffer, durante la adquisición, el numero de datos leídos es superior al tamaño del buffer temporal de adquisición
-512	Error de archivo, acceso al archivo incorrecto
-513	Se va a intentar dividir por cero. Se aborta la operación
-514	Programa de la UCI demasiado largo
-515	Fallo en el test de memoria de la UCI
-516	Intento de lectura de Buffer vacío
-517	Error Se ha llenado el buffer de adquisición
-518	Error Se ha intentado acceder a una posición no válida del vector

	<b>Manual de Usuario Librería SITAU SAFT</b>	Versión: 1.2 Revisión: 1 Fecha: 04/08/2009
---	--	--

## 6 Esquema de programación

Los pasos para programar utilizando la librería SITAU SAFT (stlibs.dll) son los siguientes:

- 1 Abrimos la conexión con el equipo y la carga del firmware:

```
ST_OpenSys ("./bin/", canales_activos, canales_mux)
```

- 2 Se programan los parámetros de configuración necesarios para la adquisición con las funciones del tipo:

```
ST_SetXXXXX ()
```

- 3 Una vez programado el sistema según nuestras necesidades, leemos el número de datos que tiene cada adquisición, con el fin de reservar la memoria necesaria para obtener los datos de la adquisición (*imagen*)

```
ST_GetNumData (&NDatosCanal, NCanalesActivos, &NDatosTotal)
```

- 4 Comienza el bucle de adquisición:

- 4.1 Disparo (en el caso de que la fuente de disparo sea software) y la transferencia de todos los datos que se encuentran en la memoria interna del Ultrascopio al buffer de la memoria RAM del PC:

```
ST_RunAcq(imagen, NDatosTotal)
```

- 4.2 Volvemos a ejecutar el inicio del bucle de adquisición 4.1

## 7 Explicación detallada de la recuperación de una adquisición

Una vez configurado el sistema como se desee, dos son las funciones necesarias para lanzar una ejecución, recuperar los datos y poder interpretarlos:

*ST\_GetNumData* y *ST\_RunAcq*

Mediante la función *ST\_GetNumData(&NDatosCanal, NCanalesActivos, &NDatosTotal)* se recupera el número de datos a adquirir por cada canal (*NDatosCanal*) y el número de datos total de la adquisición (*NDatosTotal*).

Utilizando el valor *NDatosTotal* se reserva en memoria un puntero para los datos de la adquisición. Ese puntero junto con el tamaño reservado son los parámetros de entrada a la función *ST\_RunAcq(imagen, NDatosTotal)*. Con dicha función, se lanza la ejecución y se recuperan los datos en *imagen*.

La cantidad de muestras que se van a recibir por canal se obtiene en el parámetro *NDatosCanal* de la función *GetNumData*. Su formato es el siguiente: un vector de longitud el N° de Canales Activos del Sistema y sus elementos son de tipo *unsigned int*

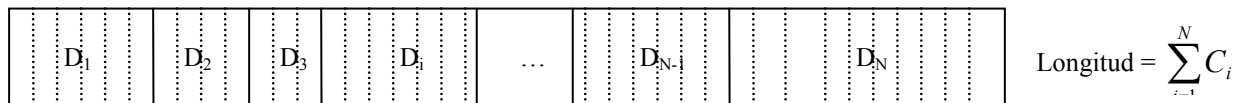


Vector con el número de muestras por canal

$$C_i \in [0, 4096]$$

Empleando la función *ST\_RunAcq*, el usuario recibe en el argumento *imagen* los datos de la última adquisición del equipo. El formato de esos datos es el que se describe a continuación:

En *imagen* se recibe un vector donde cada uno de sus elementos es un *short int*. En el vector *imagen* se obtiene la muestra uno del canal uno, la muestra dos del canal uno, hasta la última muestra del canal uno; seguidamente viene la muestra uno del canal dos, y así sucesivamente hasta llegar a la última muestra del último canal.



Vector de datos obtenidos del sistema

$D_i$  representa todos los datos del canal  $i$ , mientras que  $C_i$  representa el número de muestras adquiridas en el canal  $i$ . Por lo tanto, la longitud del vector de datos

obtenidos del sistema será:  $\sum_{i=1}^N C_i$  donde  $N$  es el número de canales activos del sistema.

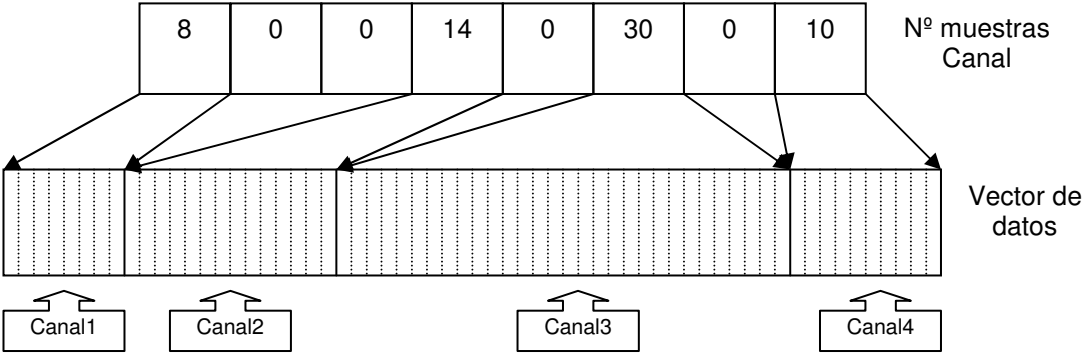
Si lo que se desea es representar las  $N$  a-scans contenidas en el vector *imagen*, el código de MATLAB a utilizar es el siguiente:

```

ndat=1;
for i=1:N
    if C(i)>0
        plot(D(ndat:ndat+C(i)-1));
        hold on
        ndat = ndat + C(i);
    end
end
end

```

Lo que el código hace es establecer la siguiente relación entre los vectores *imagen* (D) y *NDatosCanal* (C), tal y como se muestra en la siguiente figura:



Obteniendo, finalmente un conjunto de N vectores  $D_i$  de longitud  $C_i$ . Puesto que  $C_i$  será 0 para todos aquellos canales en los que se haya programado un tiempo de adquisición de 0, el orden del conjunto de vectores  $D_i$  será, finalmente de  $N - \sum_{C_i=0} i$