Gestión del RELOJ en

En los artículos publicados hasta ahora hemos examinado en detalle un gran número de características del microcontrolador STLITE09, sin omitir absolutamente ninguna información, incluso la que aparentemente es más irrelevante. Siguiendo nuestra línea afrontamos en este artículo la estructura y gestión del reloj, elemento necesario para el correcto funcionamiento de cualquier programa.

Todas las operaciones desarrolladas por un microprocesador se realizan mediante impulsos eléctricos emitidos a intervalos regulares sincronizados por **un reloj** que marca el tiempo de trabajo a todo el sistema, de forma similar a como lo hace un **metrónomo** al que tienen que atender todos los músicos de una orquesta para mantener el sincronismo de los instrumentos individuales.

El objetivo de este artículo es hacer comprender la **gestión** del **reloj** (**clock**) de los microprocesadores **ST7 LITE 09**.

La generación del **reloj final**, el que interesa a la **CPU**, se realiza mediante varias **etapas intermedias** administradas por **registros específicos** y por algunas selecciones realizadas a través de los **registros Option Byte**.

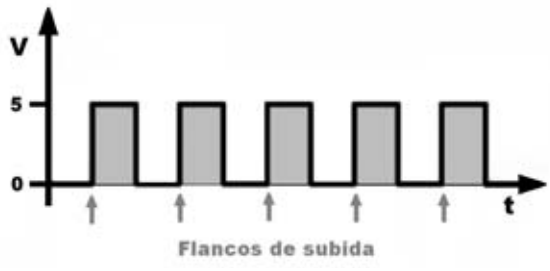


Fig.2 Para generar la señal de reloj de la CPU son necesarias varias etapas administradas por la configuración de los registros OPTION BYTE, el registro RCCR y el registro MCCSR.

NOTA: Es muy conveniente leer previamente el artículo dedicado a los **registros Option Byte** (**Revista Nº236**).

Para empezar, la **frecuencia de reloj** puede ser determinada por una **etapa osciladora RC interna** o a través de un **oscilador externo**.

Posteriormente puede ser "**multiplicada**" **x4** o **x8** a través de la **etapa interna PLL** (multiplicador de frecuencia). Por último puede ser "**dividida**" por **32** a través del registro **MCCSR** (**M**ain **C**ontrol/**S**tatus **R**egister).

Esta es, a grandes rasgos, la **estructura del reloj** que hemos sintetizado en la Fig.2. Para distinguir los diferentes pasos hemos utilizado algunos acrónimos que exponemos a continuación:

Frecuencia generada utilizando el **oscilador RC interno**.

Frecuencia generada por un **reloj externo** y después de la etapa **divisora por 2**.

Frecuencia después del "filtro" impuesto por los valores de los **registros Option Byte**, que también comporta la eventual intervención de la **etapa PLL**.

Frecuencia final del reloj de la **CPU**. Representa el **reloj** que determina el **ciclo maquina**, es decir la frecuencia que impone el ritmo de tiempo a los pasos individuales de cada instrucción (ver Fig.1).

Como ya hemos expuesto, el valor final de la frecuencia de **reloj** está influenciada por algunos valores de los **registros Option Byte** durante programación y por la parametrización de **dos registros** internos de **control** durante

los micros ST7 LITE 09

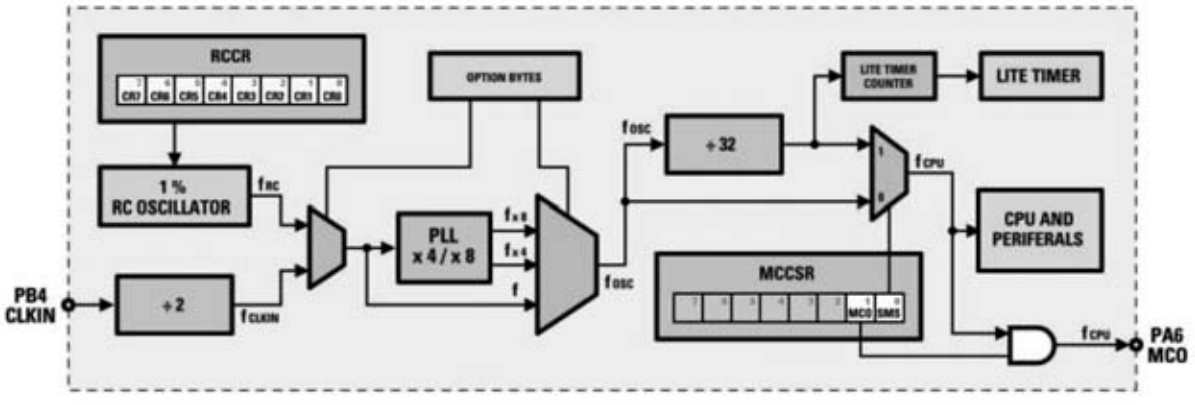


Fig.1 Son los flancos de subida de la onda cuadrada los que marcan con precisión la activación de los pasos elementales necesarios para la ejecución de cada instrucción (ciclos máquina).

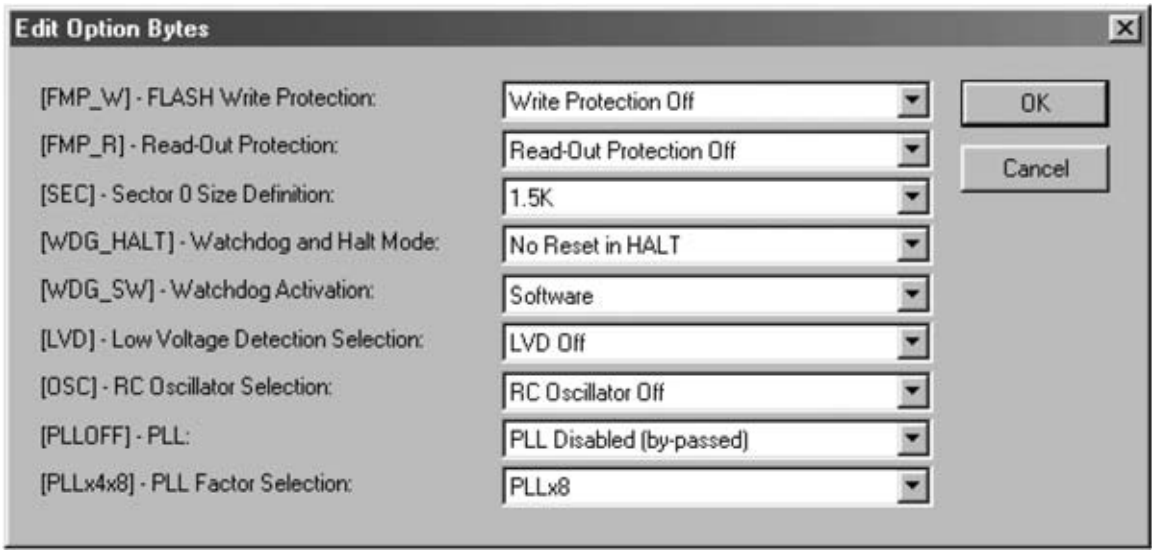


Fig.3 Las opciones de OPTION BYTES que intervienen en la gestión del reloj son tres: Selección del oscilador (interno o externo), utilización del PLL y selección del factor de multiplicación (x4 o x8). Su configuración debe realizarse a través del programa inDart.

la ejecución del programa. Estos dos registros son el **MCCSR** (**M**ain **C**ontrol/**S**tatus **R**egister) y el **RCCR** (**RC C**ontrol **R**egister).

Un tercer registro señala si ha sido activado o no el **PLL**, el registro **SICSR** (**S**ystem **I**ntegrity **C**ontrol **S**tatus **R**egister). Este registro será analizado cuando nos ocupemos de la gestión de la integridad del sistema. Por el momento, teniendo la Fig.2 a la vista, vamos a comenzar analizando **una a una** las **etapas** relacionadas con la generación del **reloj** de la **CPU**.

ETAPA OSCILADORA RC INTERNA

Esta etapa está compuesta por un oscilador **RC parametrizable** y por el registro de sistema **RCCR** (**RC C**ontrol **R**egister), cuyo formato es el siguiente:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0

Se trata de un registro de **8 bits** cuyo valor en la fase de **reset** es **FFh**.

Para activar el oscilador interno es necesario seleccionar el valor **RC Oscillator On** en los registros **Option Byte**.

Esta etapa es **parametrizable** gracias al registro interno **RCCR** que permite generar un rango de frecuencias **frc** entre **720 KHz** (con **RCCR = FFh**) y **1.700 KHz** (con **RCCR = 00h**), siempre y cuando el micro esté alimentado a **5 voltios**.

En la tabla siguiente se reproducen las **frecuencias máximas** y **mínimas** que se pueden conseguir en función de la **tensión de alimentación**, seleccionando los valores indicados en el registro **RCCR**.

Tensión Vcc	RCCR	frc
2,4~3,3 V	FFh	0,31 MHz
	00h	0,63 MHz
3,4~5,5 V	FFh	0,71 MHz
	00h	1,70 MHz

Se puede observar claramente que cuánto más **alto** es el valor contenido en **RCCR** más **bajo** es el valor de la **frecuencia generada** (**frc**), y viceversa. Hay que tener en cuenta que estos valores son los **valores nominales**. En realidad pueden variar ligeramente ya que esta etapa tiene una **tolerancia** del **1%**, es decir podría haber **variaciones** en la frecuencia incluso con un mismo valor en el registro **RCCR**. Esto podría representar un problema si **ST** no lo hubiera solucionado introduciendo dentro de cada microcontrolador **valores de calibración** en direcciones de memoria muy concretas, dos correspondientes a **EEPROM** y dos a **Flash ROM**.

En efecto, estas posiciones de memoria contienen los **valores** de **calibración RC**, válidos tanto si los micros están alimentados a **5 voltios** como si están alimentados a **3 voltios**.

La tabla siguiente contiene las direcciones de memoria y las frecuencias que se pueden obtener.

Tensión	EEPROM	FLASH ROM	frc
2,4~3,3 V	1001h	FFDFh	700 MHz
3,4~5,5 V	1000h	FFDEh	1 MHz

Se trata de **valores específicos** y calibrados para las características y la tolerancia de **cada microcontrolador individual**. Estos valores se obtienen en las direcciones indicadas y se han de cargar en el registro **RCCR**, permitiendo conseguir, como iremos viendo, una **frc** de **1 MHz** a **5,5 voltios** o de **700 KHz** a **3,3 voltios**.

De esta forma se puede obtener una frecuencia inicial muy válida. No obstante es necesario puntualizar dos aspectos.

- **PRIMERO**. Los valores de calibración presentes en las dos áreas de memoria se **pierden**, poniéndose a 0, cuando se reutiliza un micro que ha sido **programado** anteriormente y **protegido contra lectura** (**Read-Out Protection On**), por lo que se ha de tener mucho cuidado con la parametrización de los registros **Option Byte** (si solo se programa el micro, **sin proteger**, **no** se borran los valores de calibración).

Existe la posibilidad de borrar estos valores en fase de programación, pero esto presupone un acto consciente e intencionado. En versiones de **inDart** y **DataBlaze diferentes** a las que nosotros distribuimos es posible, a través de la manipulación de ciertas opciones, borrar estas posiciones de memoria.

- **SEGUNDO**. El valor de calibración utilizado **por defecto** es el correspondiente a la frecuencia de **1 MHz**, utilizándose siempre que no se logre una frecuencia exacta.

Recordamos que los valores indicados en este epígrafe se refieren a la **frecuencia** en la salida de la **etapa RC** (**frc**), **no** a la frecuencia final de **CPU** (**fcpu**).

RELOJ EXTERNO

El microcontrolador **ST7LITE09** cuenta con una **entrada** para la conexión de un **reloj externo** (**CLKIN**) en correspondencia con la patilla **8** del micro. Cuando **no** se utiliza un **reloj externo** este terminal se utiliza como un **puerto** de **entrada/salida** (ver Fig.4).

Seleccionando **RC Oscillator Off** en **Option Bytes** (ver Fig.3) el puerto **PB4** del terminal **8** se convierte automáticamente en la entrada **CLKIN** conectada directamente a un **divisor interno por 2**. Por tanto, aplicando a este terminal un oscilador externo se obtiene una frecuencia **fclkin** con el valor de la frecuencia presente en el terminal **8** (**CLKIN**) **dividido por 2**.

La tabla siguiente contiene los valores de frecuencia **máximos** aplicables a **CLKIN**, divididos en función de la tensión de alimentación.

Tensión	CLKIN (PB4)	fclkin
2,4~3,3 V	8 MHz	4 MHz
3,4~5,5 V	16 MHz	8 MHz

ETAPA PLL (Multiplicador de Frecuencia)

Esta etapa se activa a través del bit **PLL Enabled** de **Option Bytes** y seleccionando el factor de multiplicación (**PLLx4** o **PLLx8**).

Activando el **PLL** la etapa es capaz de multiplicar **x4** o **x8** la frecuencia generada tanto por el oscilador **RC interno** (**frc**) como por el **reloj**

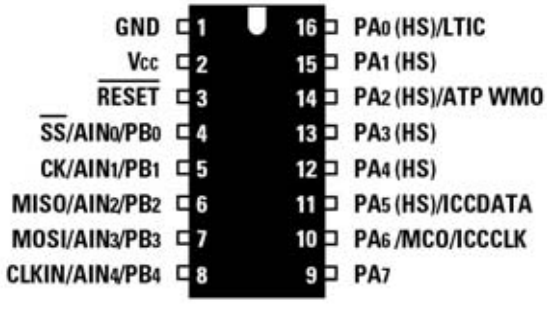


Fig.4 Conexiones del micro ST7LITE09. El oscilador externo se conecta al terminal 8 (CLKIN, AIN4, PB4). Configurando adecuadamente el registro MCCSR en el terminal 10 (PA6, MCO, ICCCLK) se obtiene la frecuencia de trabajo de la CPU.

externo (**fclkin**). De esta forma la frecuencia resultante, que hemos denominado **fosc**, es:

fosc = frc x PLL (reloj interno)
fosc = fclkin x PLL (reloj externo)

Si en **Option Bytes** seleccionamos **PLL Disabled** la etapa PLL se **desactiva**, por lo que el factor de multiplicación elegido es **irrelevante**. En este caso:

fosc = frc (reloj interno)
fosc = fclkin (reloj externo)

Con lo aquí expuesto se puede construir una tabla, que exponemos a continuación, con las frecuencias **fosc mínimas** y **máximas** utilizando el **oscilador RC interno** y la **etapa PLL**

Tensión	RCCR	frc	PLL	fosc
2,4~3,3 V	FFh	0,31 MHz	off	0,31 MHz
	00h	0,63 MHz	off	0,63 MHz
	FFh	0,31 MHz	x4	1,20 MHz
	00h	0,63 MHz	x4	2,50 MHz
	FFh	0,31 MHz	x8	2,40 MHz
	00h	0,63 MHz	x8	5,00 MHz
3,4~5,5 V	FFh	0,71 MHz	off	0,71 MHz
	00h	1,70 MHz	off	1,70 MHz
	FFh	0,71 MHz	x4	3,20 MHz
	00h	1,70 MHz	x4	6,88 MHz
	FFh	0,71 MHz	x8	5,71 MHz
	00h	1,70 MHz	x8	13,7 MHz

En la tabla siguiente se muestran las **frecuencias máximas** aplicables sobre **CLKIN** (**PB4**) utilizando un **oscilador externo** y la **etapa PLL**.

Tensión	CLKIN (PB4)	fclkin	PLL	fosc
2,4~3,3 V	máx. 8 MHz	4 MHz	off	4 MHz
	máx. 4 MHz	2 MHz	x4	8 MHz
	máx. 2 MHz	1 MHz	x8	8 MHz
3,4~5,5 V	máx.16 MHz	8 MHz	off	8 MHz
	máx. 8 MHz	4 MHz	x4	16 MHz
	máx. 4 MHz	2 MHz	x8	16 MHz

Main Control/Status Register (MCCSR)

Para generar la **frecuencia** de la **CPU** (**fcpu**) se utiliza el **registro MCCSR**, cuyo formato es el siguiente:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
						MCO	SMS

Se trata de un registro de **8 bits** que en la fase de **reset** toma el valor **00h**.

Los **bits** del **7 al 2** están **reservados**, siempre valen **0**, mientras que los **bits 1** y **0**, denominados respectivamente **MCO** (**M**ain **C**lock **O**ut Enable) y **SMS** (**S**low **M**ode **S**elect), sí son utilizables. A continuación detallamos la operatividad de estos dos bits.

Bit 1 (MCO, Main Clock Out Enable)

En este bit está permitida tanto la lectura como la escritura. Cuando tiene el **valor 1** habilita el **terminal 10** como salida de **Reloj** con la **frecuencia** de reloj de la **CPU** (**fcpu**). Cuando está a **0** el **terminal 10** se comporta como un terminal normal de **entrada/salida** (**PA6**).

Hay que prestar un poco de atención. Como ya hemos explicado en artículos anteriores el terminal **PA6** del **puerto A** es un terminal **multifunción** (**PA6**, **MCO**, **ICCCLK**). En efecto hay que seleccionar si se quiere utilizar este terminal como **puerto**, como **señal MCO** o como **señal ICCCLK** (**I**n **C**ircuit **C**omunication **CL**oc**K**) reservada para la programación y para la depuración **In Circuit**. Esto significa que quien lo desee utilizar en un programa como **señal MCO** no podrá realizar **depuración** (**Debug**) **In**

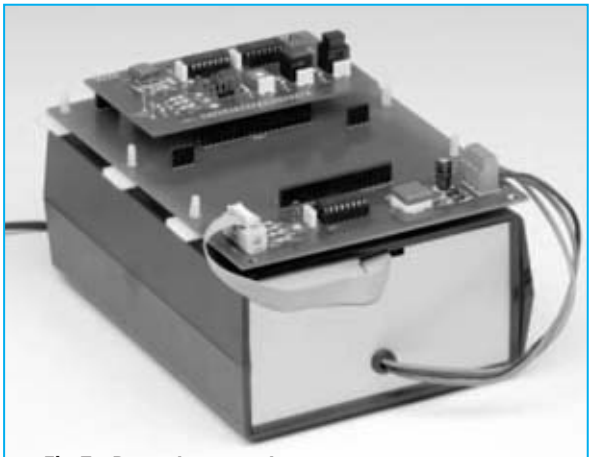


Fig.5 Para las pruebas que proponemos son necesarios el Programador LX.1546, el Bus LX.1547 y la Tarjeta experimental LX.1548. Los esquemas de estos proyectos se encuentran en las revistas Nº227 y Nº228.

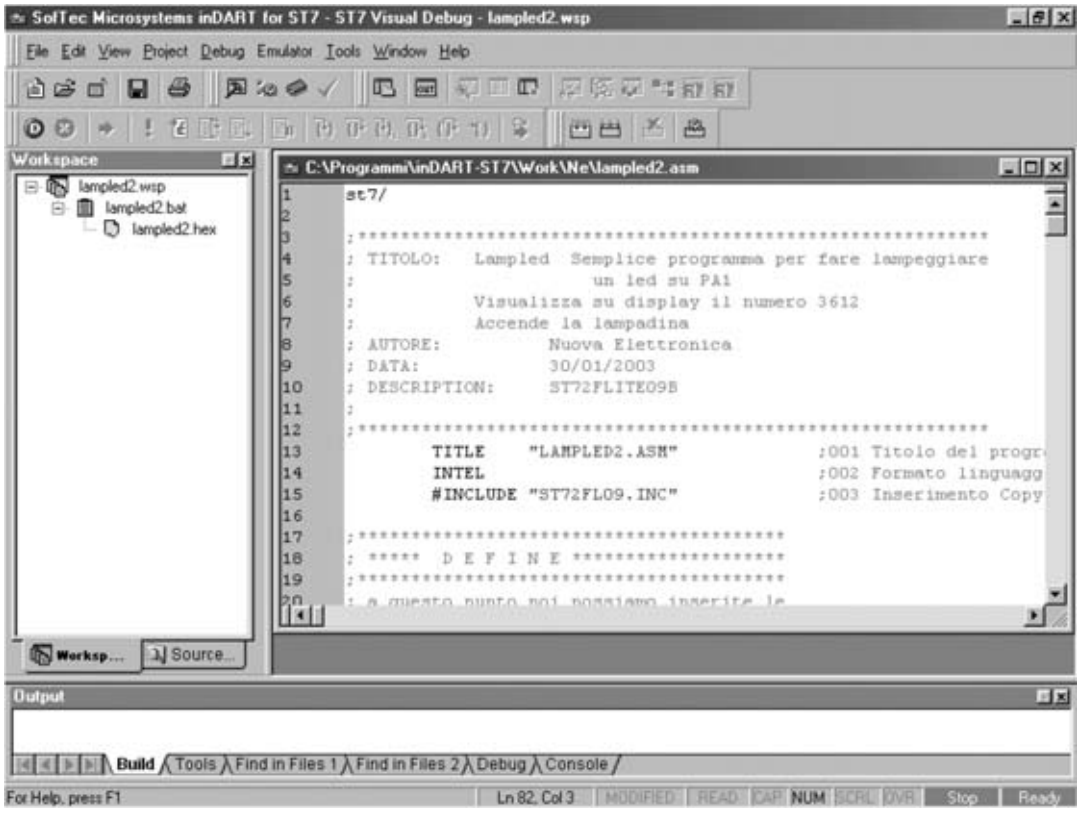


Fig.6 Una vez lanzado el programa inDart y abierto el proyecto LAMPLED2.WSP en la pantalla aparecerá una ventana similar a la aquí mostrada. Para abrir la ventana que visualiza el contenido de la memoria del micro (ver Fig.8) hay que cargar previamente el programa en formato ejecutable haciendo click en el icono START DEBUGGING (ver Fig.7).

Circuit con el programa **inDart**. En este caso sí se podrá cargar con **DataBlaze** y, una vez lanzada la ejecución, efectuar las pruebas de funcionamiento que se consideren pertinentes.

Bit 0 (SMS, Slow Mode Select)

En este bit también está permitida tanto la lectura como la escritura. Cuando tiene el **valor 1** la frecuencia **fosc** se **divide por 32**. Por tanto:

fcpu = fosc: 32

En caso contrario, es decir cuando está a **0**:

fcpu = fosc

Para concluir hay que tener presente que esta es la **frecuencia** con que trabajarán la **CPU** y todos los periféricos (**ARTIMER**, **SPI**, **ADC**, etc.) con la **excepción** del **LITE TIMER**, cuyo contador siempre se incrementa con la señal **fosc:32** y no con la señal **fcpu**, aunque el bit SMS esté a **1** (**fcpu = fosc: 32**).

De la TEORÍA a la PRÁCTICA

Hasta ahora hemos hablado de la **estructura** del **reloj** en términos **teóricos**. Para adquirir un completo domino sobre su **gestión** también es necesario trabajar de forma **práctica**.

Para cumplir este objetivo vamos a utilizar el proyecto **lampled2.wsp**, ya utilizado en artículos anteriores con fines didácticos. También se puede utilizar el proyecto original **lampled.wsp**, teniendo cuenta que en los ejemplos realizaremos modificaciones del **código fuente** y de la configuración de los **registros Option Byte**.

El programa original utiliza un **oscilador externo**. Detallaremos las modificaciones a

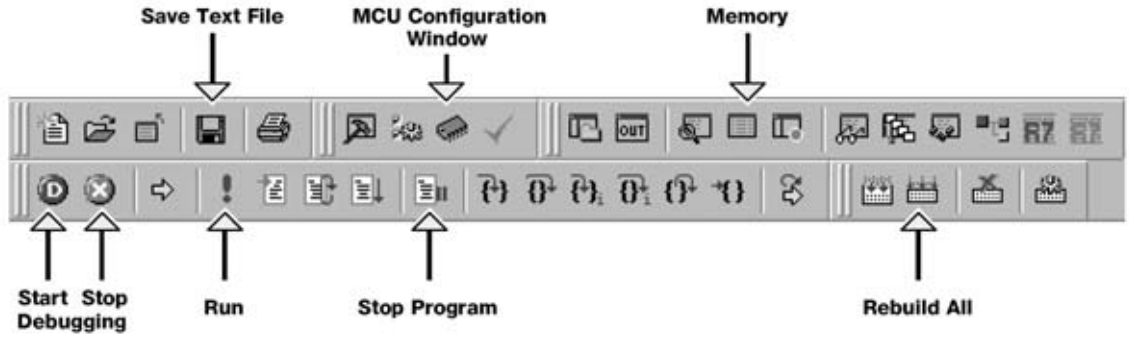


Fig.7 Para facilitar la búsqueda de las instrucciones utilizadas hemos agrupado en una sola imagen todos los iconos indicando sus funciones. Si habéis personalizado las barras de herramientas estos iconos podrían estar colocados de forma diferente, aunque sus funciones son las mismas. Obviamente cuando está activo el icono START DEBUGGING no está activo el icono STOP DEBUGGING y viceversa. Lo mismo sucede con los iconos RUN y STOP PROGRAM.

realizar para utilizar la **frecuencia RC interna** y el **PLLx4** para conseguir una **fcpu** de **4 MHz**. Para realizar nuestras pruebas se precisa el **Programador LX.1546**, el **Bus LX.1547** y la **Tarjeta experimental LX.1548** (ver Fig.5), además, obviamente, del programa **inDart**.

NOTA: El ensamblaje y conexión de las tarjetas, así como la instalación y configuración del programa están expuestos en las **revistas Nº227** y **Nº228**.

Después de tener listas las tarjetas y conectadas al puerto paralelo del ordenador, hay que ejecutar el programa **inDart** y abrir el proyecto **lampled2.wsp**. La situación será similar a la mostrada en la Fig.6.

En primer lugar hay que activar el **depurador** (**debug**) haciendo click en el icono correspondiente (ver Fig.7) de modo que se cargue el programa en formato ejecutable en el micro. **No** hay que lanzar la **ejecución** del **programa**, ya que antes de proceder con el ejemplo queremos mostrar la forma de controlar si están presentes en vuestros micros **ST7LITE09** los valores de calibración para el **oscilador RC interno**.

Los VALORES de CALIBRACIÓN en MEMORIA

Una vez cargado el programa en formato ejecutable en el micro se puede acceder a los valores presentes en su memoria haciendo click en el icono **Memory** (ver Fig.7). La ventana que se abre, reproducida en la Fig.8, muestra el **contenido** de la **memoria** del microcontrolador en **formato hexadecimal** y en **código ASCII**. Esta ventana es muy útil cuando efectuéis **depuraciones** de vuestros programas, aunque por ahora nos limitaremos a utilizarla para controlar los **valores de calibración**.

En la columna izquierda se muestran las **direcciones** de memoria mientras que en la derecha están presentes los **valores contenidos** en cada dirección. Cada línea contiene **16 valores** correspondientes a **16 bytes**, por tanto entre una dirección y la siguiente en la columna izquierda hay un intervalo de 16 bytes.

Nuestro objetivo es averiguar el valor presente a la dirección **1000h**, que, como hemos expuesto, es la dirección de **EEPROM** que contiene el valor de calibración para conseguir una **frc** de **1 MHz** cuando el micro está alimentado, como es nuestro caso, a **5 voltios**. Utilizando la **barra** de **desplazamiento vertical** presente en la parte derecha se puede avanzar hasta encontrar la dirección correspondiente a **1000h**, si bien hay otro método más directo.

Después de hacer click en la parte situada a la derecha del **0x** de la dirección actual situada debajo del encabezado **Memory** (ver Fig.9), hay que escribir **1000** y pulsar la tecla **ENTER**. De esta forma el cursor se desplaza directamente a la dirección de memoria **1000h**.

El número hexadecimal **a5** es el valor de calibración que permite conseguir una **frc** de **1 MHz** para el micro utilizado en nuestras pruebas (ver Fig.10). Ya que, como hemos

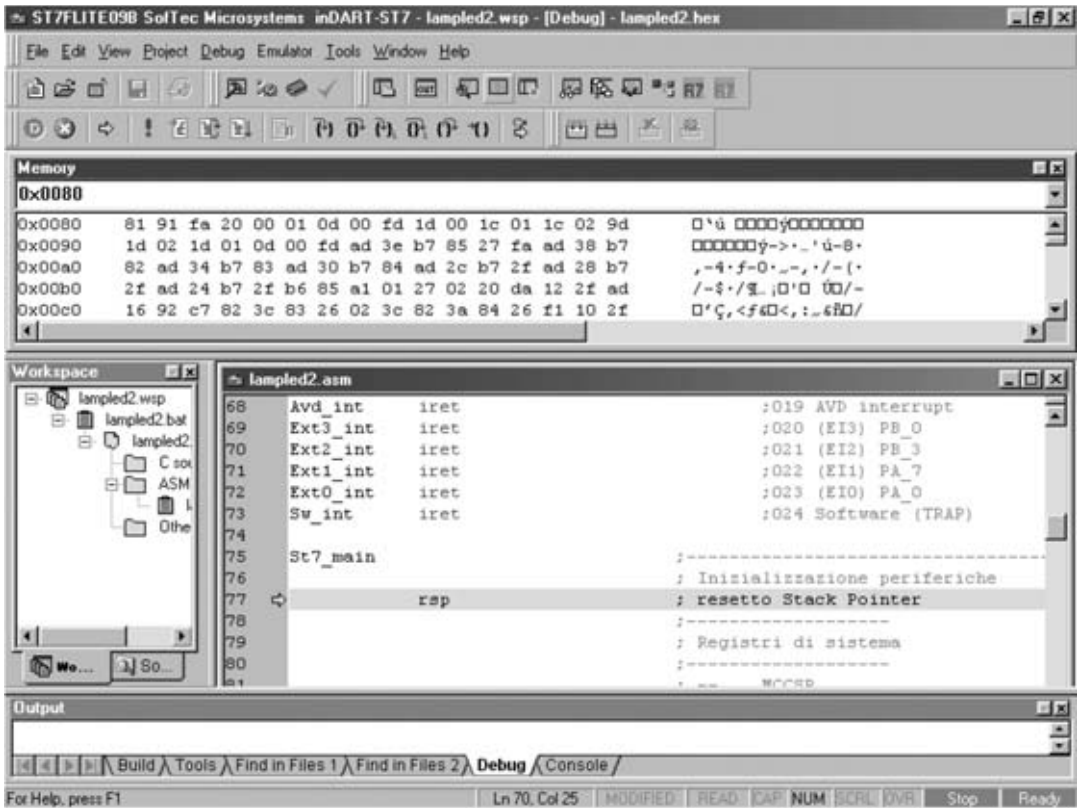


Fig.8 La ventana MEMORY muestra los valores contenidos en la memoria del micro. A izquierda se encuentran las direcciones de memoria y a la derecha su contenido, en formato hexadecimal y en código ASCII.

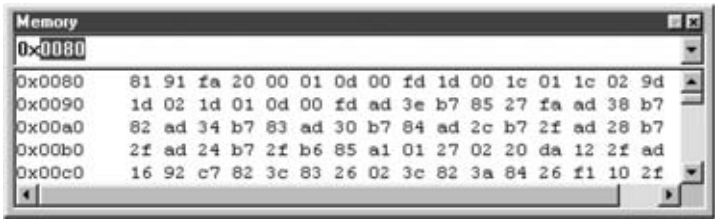
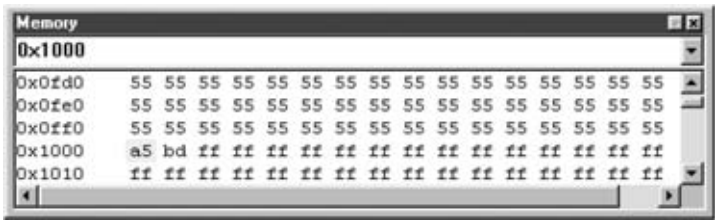


Fig.9 Para localizar la dirección de memoria 1000h hay que seleccionar las últimas cuatro cifras de la línea superior y escribir 1000.

Fig.10 Presionando la tecla ENTER el cursor se desplazará sobre el contenido de la dirección 1000h, que en nuestro caso es A5.



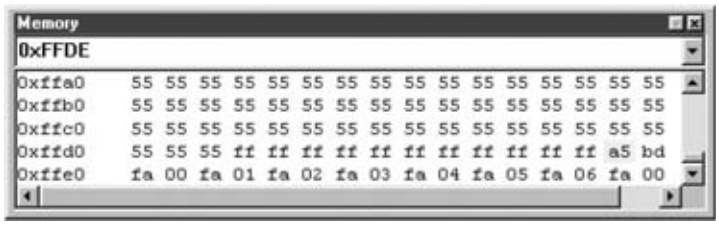


Fig.11 Escribiendo FFDE y pulsando ENTER se puede comprobar que el contenido de esta dirección de memoria también es A5.

explicado, cada **microprocesador** tiene su **propio valor** de **calibración**, es posible que encontréis en vuestro micro un valor diferente

El valor situado al lado (**bd**), es decir el contenido de la dirección de memoria **1001h**, es el que habríamos tenido que utilizar en el caso de alimentar el micro entre **2,4** y **3,3 voltios**. Ahora hay que verificar que también estén presentes estos valores en las direcciones de memoria **FFDE** y **FFDF** (Flash Memory). Para acceder rápidamente hay que escribir **FFDE** con el mismo procedimiento anteriormente utilizado para el **1000**, y pulsar **ENTER**. Como se puede ver en la Fig.11 también los valores contenidos en estas direcciones de memoria son **a5** y **bd**.

NOTA IMPORTANTE: en la **versión 1.11** del programa **inDart** los valores contenidos en las direcciones de Flash ROM **FFDEh** y **FFDFh** no se visualizan correctamente. Esto no ocurre en la **versión 1.18**, que se puede descargar de forma gratuita de la **Web oficial** de **Softec** o en **www.nuevaelectronica.com**.

Siempre es aconsejable tener la **última versión disponible** ya que siempre hay mejoras sobre las versiones anteriores.

Si vais a utilizar el **oscilador RC interno** es muy aconsejable **apuntar** los **valores de calibración** de cada micro, de esta forma aunque se borren al protegerlos contra lectura se podrán **volver a restablecer** los valores ya que se tienen apuntados. No hay que preocuparse si el **contenido** de estas direcciones de memoria se ha **borrado** y **perdido** ya que, como mostraremos en la última parte del artículo, se pueden **recalcular** los valores de calibración con un sistema empírico bastante simple.

CARGAR el valor de calibración en el REGISTRO RCCR

Para continuar con el ejemplo hay que cerrar la ventana **Memory** y posicionar el cursor a la altura de la línea **78**, alineándolo bajo la letra **r** de la instrucción **rsp** de la línea anterior. Aquí hay que escribir:

ld a,1000h (o bien **ld a,FFDEh**)

En la línea siguiente, es decir en la **79**:

ld RCCR,a

Con estas instrucciones el **valor de calibración** se carga en primer lugar en el registro **acumulador A** y luego se lleva al **registro RCCR** para conseguir una **frc** de **1 MHz**.

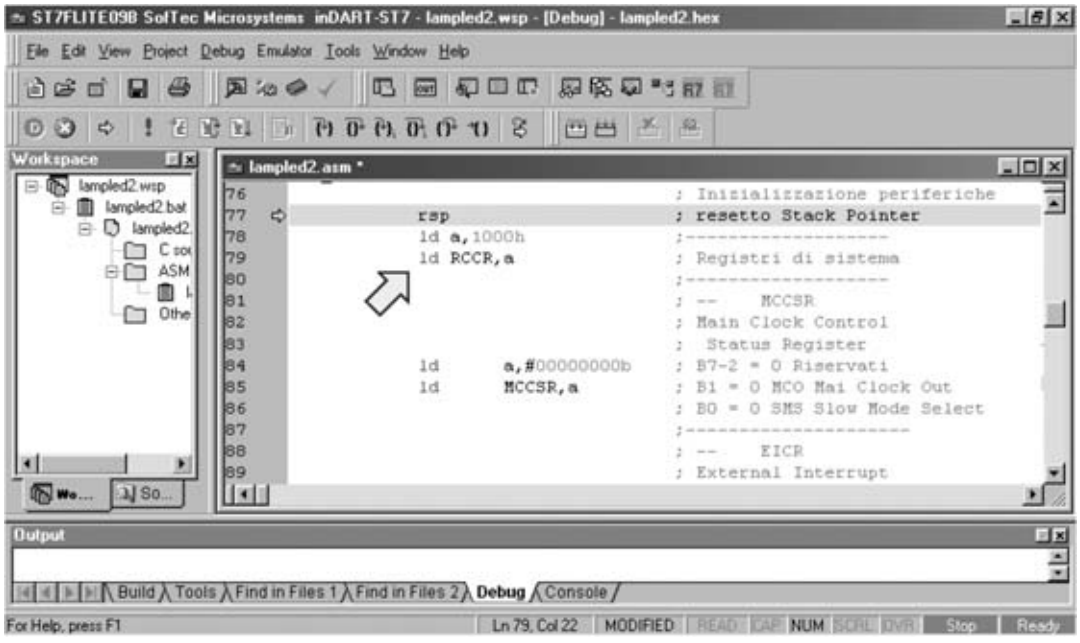


Fig.12 Insertado en las líneas de programa 78-79 las dos instrucciones indicadas en esta figura se carga en el registro RCCR, que administra el oscilador RC interno del micro ST7LITE09, el valor de calibración utilizado para conseguir una frecuencia frc de 1 MHz con el micro alimentado a una tensión de 5 voltios.

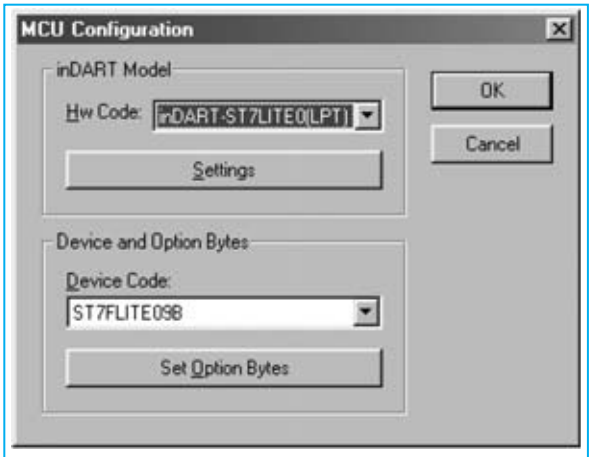


Fig.13 Haciendo click en el icono MCU CONFIGURATION WINDOW (ver Fig.7) se accede a esta ventana. Para configurar los registros OPTION BYTE hay que hacer click en el botón SET OPTION BYTES.

La fuente desde donde obtener el valor, bien de la memoria **EEPROM** o bien de **Flash ROM**, depende del tipo y de las características del proyecto administrado. La experiencia determinará qué elección tomar. Una vez realizadas las modificaciones hay que **detener** el **depurador**, **salvar** el **archivo** y **recompilar** el programa. Para realizar estas operaciones, hay que hacer click, sucesivamente, en los iconos **Stop Debugging**, **Save Text File** y **Rebuild All** (ver Fig.7).

Si se han escrito correctamente las dos instrucciones **no** se producirán **errores** de compilación, en caso contrario hay que verificar lo que se ha escrito. El próximo paso consiste en configurar los **registros Option Byte** para activar el **oscilador RC interno** y el **PLLx4**.

CONFIGURAR OPTION BYTES

Haciendo click en el icono **MCU Configuration Window** (ver Fig.7) se abre la ventana mostrada en la Fig.13. Aquí hay que hacer click en **Set Option Bytes** (los valores predeterminados en **Option Bytes** se muestran en la Fig.14).

Para conseguir nuestro objetivo, es decir una **frecuencia** de trabajo de la CPU de **4 MHz** utilizando el **oscilador RC interno**, hay que modificar las últimas tres líneas ajustando las opciones de configuración tal y como las hemos reproducido en la Fig.15.

De este modo el micro obtendrá la frecuencia de **reloj** de su oscilador interno, que hemos

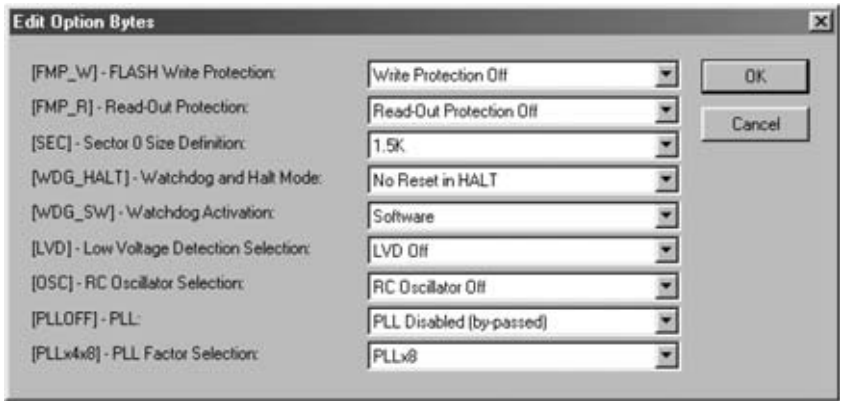


Fig.14 En esta imagen se reproduce la configuración estándar de los registros OPTION BYTE.

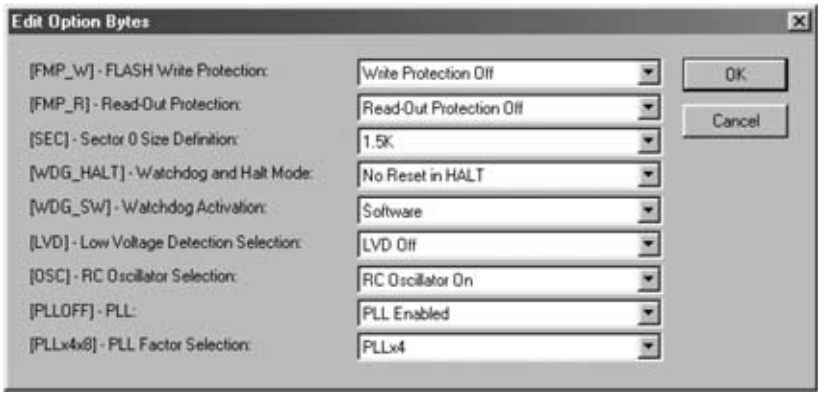


Fig.15 Para conseguir una fcpu de 4 MHz hay que modificar las tres últimas líneas como se indica en esta figura.

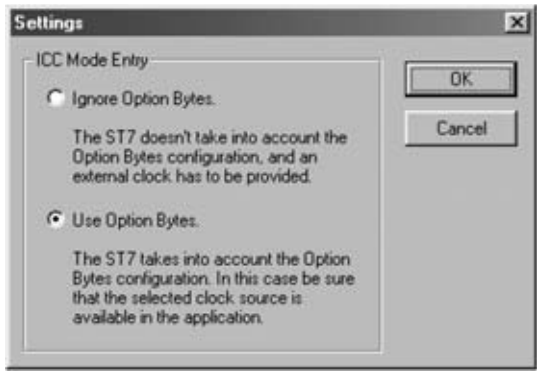


Fig.16 Para cargar los datos en el micro tomando la configuración de OPTION BYTES ajustada en la Fig.15 hay que hacer click en USE OPTION BYTES.

programado a **1 MHz** en el registro **RCCR**, y la multiplicará **x 4**, generando así una frecuencia **fosc** de 4 **MHz**. Confirmar los cambios haciendo click en **OK**. Ya que, como se puede observar en las instrucciones de las líneas **84-85** mostradas en la Fig.12, el **bit 0** del **registro MCCSR** (es decir el bit **SMS**) es puesto a **0** y la **frecuencia fosc no se divide por 32**, la **frecuencia** de trabajo de la **CPU** es de **4 MHz**, como nos proponíamos.

Antes de continuar es necesaria una aclaración. Si no hubiéramos cargado ningún valor en el registro **RCCR**, eligiendo **RC Oscillator On** en **Option Bytes**, el micro habría considerado como frecuencia **frc** la tomada en fase de **reset**, es decir **FFh**, que, para un micro alimentado a **5 voltios**, es de **0,7 MHz**. Considerando el **PLLx4** la frecuencia **fosc** sería de **2,8 MHz**.

Por último es necesario informar al micro de que la carga de los datos en el inicio de la depuración tiene que realizarse teniendo en cuenta el **modo de reloj** recién **programado** y no utilizar una configuración anterior o la predeterminada.

Hay que hacer click en el botón **Settings** de la Fig.13 y, en la ventana que aparece, seleccionar **Use Option Bytes**. Para confirmar estas acciones hay que hacer click en **OK**, solo en este momento podéis estar seguros que la **CPU** trabajará con una **frecuencia** de **4 MHz**. Para **salir** de la ventana mostrada en la Fig.13 hay que hacer click en **OK**.

EJECUCIÓN del PROGRAMA

Para ejecutar el programa **lampled2.asm** hay que comenzar haciendo click en el icono **Start Debugging** y, a continuación, hacer click en el icono **Run** (ver Fig.7). El diodo LED **DL1** de la tarjeta **LX.1548** empezará a **parpadear**.

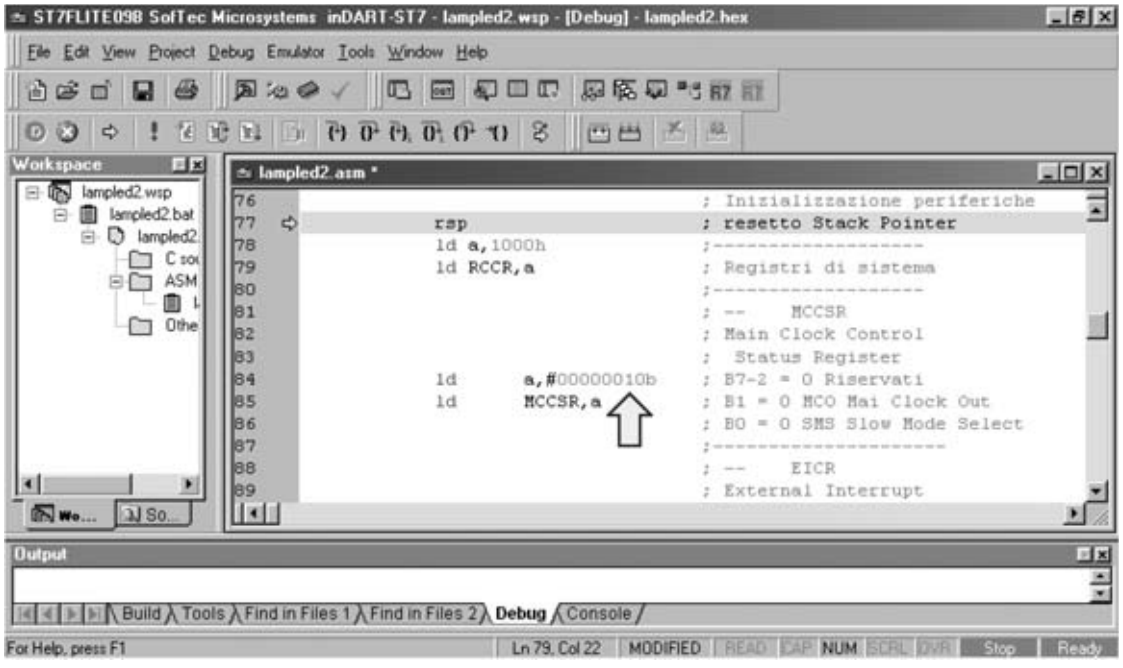


Fig.17 Con la modificación indicada en esta imagen se obtiene la frecuencia de CPU en el terminal 10 (PA6/MCO/ICCCLK).

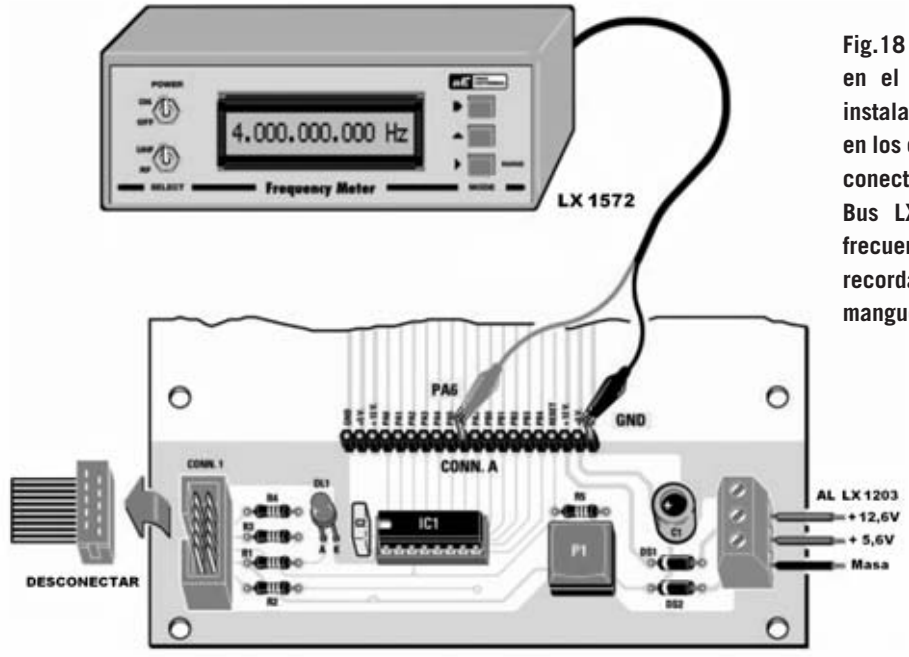


Fig.18 Para leer la frecuencia en el terminal PA6 hay que instalar dos terminales tipo pin en los contactos PA6 y GND del conector CONN.A de la tarjeta Bus LX.1547 y conectar un frecuencímetro. Hay que recordar desconectar la manguera de la tarjeta Bus.

Evidentemente **no parpadea** a **4 MHz** (4 millones de veces por segundo) ya que no se vería. Hemos introducido en el programa una **rutina de retardo** para controlar el encendido y el apagado. De esta forma es difícil determinar si efectivamente la frecuencia **fcpu** es **4 MHz** ya que, aun conociendo la duración de los retardos y de la gestión, se obtendría un resultado **muy aproximado**.

En realidad la solución para conocer la **frecuencia exacta** de la **CPU** es sencilla. La proporciona el propio microcontrolador.

FRECUENCIA de CPU en el terminal PA6

Anteriormente hemos señalado la posibilidad de generar en el terminal **10** del micro (**PA6/MCO/ICCLK**) la **frecuencia** de trabajo de la **CPU**. Para conseguirlo solo hay que poner a **1** el **bit 1** (**MCO**) del registro **MCCSR** (ves Fig.17).

Una vez lanzada la ejecución del programa se puede leer la frecuencia en el terminal **PA6** del micro con un **frecuencímetro**, como por ejemplo nuestro **Frecuencímetro digital LX.1572** (**revista Nº219**), conectándolo a la tarjeta **Bus LX.1547**, tal y como se muestra en la Fig.18. Naturalmente hay que tomar ciertas precauciones básicas.
En primer lugar hay que detener ejecución del programa haciendo click en el icono **Stop Program** y, a continuación, en el icono **Stop Debugging** (ver Fig.7).

Ya hemos expuesto como las líneas de programa **84-85**, como indicamos a continuación, contienen las instrucciones para cargar los valores en el registro **MCCSR** (ver Fig.12):

ld a,#00000000b
ld MCCSR, a

Para tener en la salida del terminal **10** del micro (**PA6**) la **frecuencia** de trabajo de la **CPU** hay que poner a **1** el **bit 1** (**MCO**) del registro **MCCSR** cambiando la primera línea por la siguiente instrucción:

ld a,#00000010b

Ahora hay que salvar el programa, haciendo click en el icono **Save Text File**, y recompilarlo, haciendo click en el icono **Rebuild All** (ver Fig.7). Después de controlar que no hay errores de compilación hay que cargar de nuevo el **depurador** (**Debug**) y lanzar la **ejecución**, haciendo click en el icono **Run**, desconectando enseguida la **manguera** que conecta el **Bus LX.1547** y el **Programador LX.1546** (esto evitará posibles conflictos entre el terminal de **ICCCLK** y la **depuración In-Circuit** del **Programador LX.1546**).

Fig.19 Durante la realización de las pruebas con el frecuencímetro pueden aparecer errores de comunicación entre el Bus y el Programador.

Fig.20 Cuando no se establece una correcta comunicación entre las tarjetas también puede aparecer este mensaje de error.

Ahora el programa trabaja de forma autónoma. Para leer la frecuencia de la **CPU** en el terminal **10** del micro (**PA6/MCO/ICCLK**) hay que insertar dos terminales tipo pin en los terminales **PA6** y **GND** del conector **CONN.A** del Bus a los que se han de conectar las puntas de prueba del frecuencímetro (ver Fig.18). La frecuencia medida puede oscilar un poco, pero siempre estará en la **tolerancia** del **1%**.

NOTA: Hemos determinado que si esta prueba se realiza con un micro programado de forma definitiva con el programa **DataBlaze** el margen de **tolerancia** de la frecuencia es **menor**. Si estáis en disposición de realizar esta prueba lo podréis comprobar vosotros mismos.

Comprobado que la frecuencia es efectivamente de **4 MHz** ya se puede volver a conectar la manguera del Bus y **detener** la **ejecución** del programa. Llegado este punto podéis efectuar otras pruebas, por ejemplo, poniendo a **1** el **bit 0** del registro **MCCSR** para verificar que la **fcpu** queda **dividida por 32**. Hay que recordar que cada vez que se realice una modificación en el programa hay que **salvar el archivo**, **recompilar** el programa y volver a **lanzar** el **depurador**, acordándose de **desconectar** la **manguera** de conexión del **Bus LX.1547**.

Cuando hayáis concluido vuestras pruebas hay que volver a **poner a 0** los bits del registro **MCCSR** y **recompilar** el programa. Si durante las **pruebas** y las **conexiones/desconexiones** de la manguera del **Bus LX.1547** aparece algún **mensaje de error** (ver Figs.19-20), no intentáis cerrar el programa **inDart** ya que se podría quedar bloqueado temporalmente vuestro ordenador. En este caso hay que verificar que la manguera esté correctamente conectada y volver a cargar el depurador.

CALCULAR los valores de CALIBRACIÓN

La posibilidad de obtener en el terminal **MCO** con bastante precisión la **frecuencia** de la **CPU** posibilita definir un sistema empírico para **recalcular** los **valores de calibración** del micro en el caso de que hubieran sido borrados y no se hubieran anotado.

Se trata de reemplazar a la instrucción:

ld a,1000h

que carga en el registro **acumulador A** el valor contenido en la dirección **1000h** (antes de llevarlo al registro **RCCR**) por una instrucción que carga en el registro **A** un valor **numérico inmediato**:

ld a,#nnn

donde **nnn** es un **número decimal** incluido entre **0** y **255**.

Nuestro consejo es comenzar con **128** y leer la frecuencia en salida. Una vez leída hay que corregir el valor, probando con otros valores, hasta que se lea la **frecuencia correcta**.

Para facilitar el cálculo es más conveniente **desactivar** la **etapa PLLx4** y probar directamente con la frecuencia de **1 MHz**. Con este apunte concluimos este artículo dedicado a la **estructura** y **gestión del reloj** de los micros **ST7LITE09**.