Práctica 4: Comunicación por puerto serie asíncrono

Prof. Alberto A. Del Barrio

1 Objetivos de la práctica

En esta práctica nos centraremos en conocer un dispositivo de comunicaciones serie estándar, la UART. Este dispositivo nos permitirá comunicar la placa del laboratorio con el PC del puesto por medio de un protocolo serie asíncrono. Los principales objetivos de la práctica son:

- Conocer los conceptos básicos de la comunicación serie asíncrona.
- Familiarizarse con la unidad UART del S3C44BOX.
- Conectar dos maletines por medio del puerto serie e intercambiar información entre ellos.

2 Comunicación serie

La comunicación entre dos dispositivos puede ser en serie o en paralelo. La comunicación en paralelo utiliza varios cables en paralelo para transmitir simultáneamente un valor de varios bits. La comunicación digital serie se caracteriza porque la información viaja en un solo cable, llamado lńea de datos.

En la comunicación serie los bits se inyectan en la línea de datos y viajan secuencialmente, uno detrás de otro, del origen al destino. Además de la línea de datos pueden existir otras líneas de control y de alimentación. A su vez, la comunicación serie se divide en dos grandes grupos:

- *Síncrona*. En la comunicación serie síncrona los dos extremos utilizan una señal de reloj común. Esta señal se transmite normalmente del emisor al receptor por un cable adicional.
- *Asíncrona*. En la comunicación serie asíncrona cada sistema utiliza su propia señal de reloj, aunque los dos sistemas deben ponerse de acuerdo en la tasa de transferencia.

En la comunicación serie síncrona los dos sistemas se sincronizan al comienzo de la emisión. La detección de los bits se hace coincidir con los flancos de la señal de reloj. Suele permitir mayores tasas de transferencia aunque tiene el inconveniente de que la señal de reloj debe ser transmitida, limitando normalmente la distancia entre los dos equipos.

En la comunicación serie asíncrona se presenta el problema de la detección de bits. Aunque los dos sistemas se pongan de acuerdo en la frecuencia de transmisión, es prácticamente imposible que las frecuencias de reloj con las que transmiten sean iguales. Para empezar puede tratarse de equipos que funcionen con distintas frecuencias (por ejemplo PC e impresora serie) y sólo puedan generar, a partir de su reloj, una señal aproximada de la velocidad

de transmisión (normalmente utilizando divisores de frecuencia como los utilizados en los temporizadores vistos en las prácticas anteriores). Además, estos relojes pueden variar su frecuencia con el tiempo, por ejemplo por variaciones de temperatura. Pequeñas diferencias en esta frecuencia producen a la larga grandes desfases en la sincronización de los equipos. La única solución es limitar la cantidad de bits que podemos transmitir hasta forzar una nueva sincronización. Por eso muchos protocolos serie asíncronos se limitan al envío de pequeños paquetes. Estos paquetes reciben el nombre de tramas. Al comienzo de cada trama se fuerza la sincronización entre los dos dispositivos. Para enviar varios bytes hay que enviar varias tramas.

La implementación habitual de un sistema de comunicación serie construye una señal de reloj para la transmisión que sea múltiplo de la señal del sistema, por ejemplo dividiendo la frecuencia de la señal del sistema entre 16. Esto permite muestrear la línea de datos para detectar el comienzo de una trama. El desfase desde que se detecta la trama hasta que se pone en marcha el sistema para la recepción será como mucho de uno o dos ciclos del reloj del sistema, lo que supone un desfase pequeño respecto de la frecuencia de transmisión.

3 Unidad UART del S3C44BOX

El sistema S3C44BOX dispone de dos dispositivos para la comunicación serie: un controlador de bus I^2C (síncrono) y una unidad UART (*Universal Asynchronous Receiver and Transmitter*).

La unidad UART proporciona dos puertos serie asíncronos independientes: *UART0* y *UART1*, también llamados canales, que permiten una comunicación serie bidireccional de hasta 115.2 Kbps. La gestión de estos canales puede llevarse a cabo mediante encuesta (*polling*), interrupciones o DMA. En esta práctica utilizaremos el método de encuesta.

Cada canal consta de los siguientes elementos (ver figura 1):

- Un generador de frecuencia (baudios).
- Un módulo transmisor, con un registro de desplazamiento y un registro de buffer.
- Un módulo receptor, con un registro de desplazamiento y un registro de buffer.
- Una unidad de control.

El generador de frecuencia, que marca la duración de un bit (y por tanto la tasa de transferencia) emplea como entrada el reloj principal del sistema (*MCLK*), cuya frecuencia es dividida.

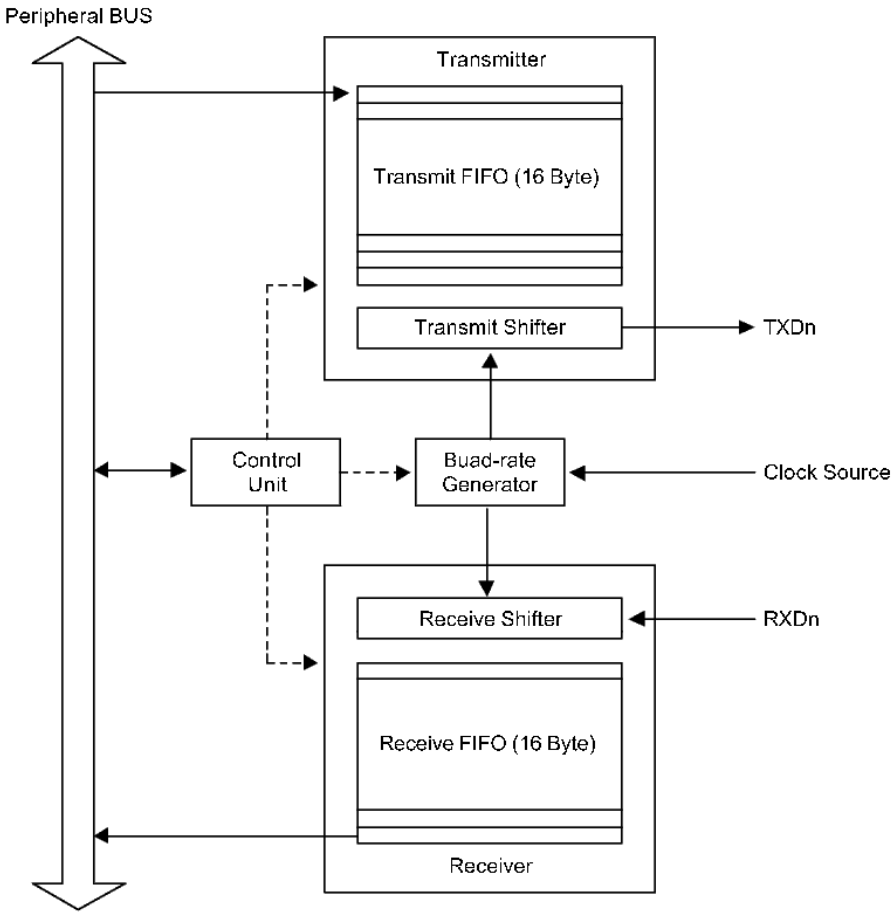

Figura 1: Estructura de un canal de la UART.

La transmisin se realiza a través de un registro de desplazamiento. Cuando el procesador quiere enviar un dato por la línea lo escribe en el buffer de emisión. Cuando la línea está lista el valor del registro se copia en el registro de desplazamiento. Este registro irá desplazando el contenido para que los bits vayan saliendo secuencialmente por la línea de datos, a la velocidad indicada por el generador de frecuencia. La recepción se realiza de forma similar por medio de otro registro de desplazamiento. La velocidad de transmisión/recepción es programable, modificando el divisor del generador de frecuencia.

Además, tanto el transmisor como el receptor incorporan una cola FIFO de 16 bytes. Si están activas, estas colas permiten desacoplar aún más la CPU y la comunicación de los datos por el puerto serie. En lugar de ir escribiendo/leyendo los caracteres uno a uno, la CPU escribe/lee bloques de hasta 16 caracteres. El contenido de la cola FIFO se va enviando por el puerto (o se recibe y se copia en la FIFO) de forma automática. La UART permite activar una interrupción cuando la cola FIFO de recepción sobrepasa un determinado umbral (o la de emisión se vacía por debajo de un umbral), de forma que el procesador tenga cierto margen para poder tratar la situación.

La comunicación a través de la línea serie se realiza carácter a carácter debido a la naturaleza asíncrona de la misma. Los registros de desplazamiento se encargan de ir transmitiendo o recibiendo los caracteres, bit a bit, por la línea correspondiente *TXDn* para transmisión o *RXDn* para recepción (donde n representa el n del puerto).

Finalmente, la UART del S3C44BOX tiene capacidad de transmisión/recepción por infrarrojos siguiendo el estándar *IrDA 1.0*.

A continuación describiremos brevemente el funcionamiento de la UART. La información detallada se encuentra en el Capítulo 10 del Manual de Referencia del S3C44B0X [1].

3.1 Formato de trama

El formato de trama (*frame*) enviado por la UART tiene la siguiente estructura:

- Un primer bit de *start*. Sirve para que el receptor detecte el comienzo de la trama y se sincronice. Tiene el valor contrario al valor de reposo de la línea, en nuestro caso el bit de *start* es cero.
- 5-8 bits de datos.
- 1 bit de paridad (opcional). Se pueden escoger paridad par o impar.
- 1-2 bits de *stop*, que tienen el valor de reposo de la línea.

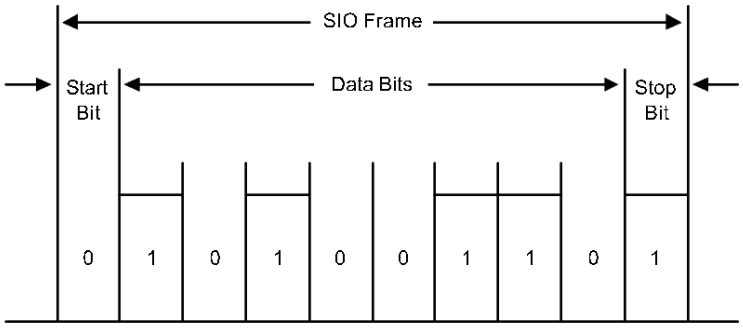

Figura 2: Diagrama temporal de una trama de datos serie (*SIO Frame*) con un tamaño de carácter de 8 bits, 1 bit de stop, sin paridad.

La Figura 2 muestra un ejemplo de trama con el siguiente formato: 1 bit de *start*, carácter de 8 bits, 1 bit de *stop* y sin bit de paridad.
En la UART del S3C44BOX el formato de trama se configura mediante el registro de control de lnea *UCONn*.

El transmisor también puede producir un *frame* de pausa o *break* que consiste en una señal de 0 lógico de un frame de duración. El propósito de este tipo de marco es informar al receptor de una pausa en la comunicación.

Al igual que para la transmisión, el formato de la trama en recepción también es configurable. Como es lógico, para una correcta comunicación es necesario que la configuración en ambos dispositivos sea la misma.

3.2 Errores

El receptor es responsable de detectar varios tipos de error de comunicación:

- Error de superposicin. Indica que un dato ha sido sobrescrito por otro ms reciente antes de ser ledo.
- Error de paridad. Indica que la paridad del dato recibido no concuerda con la esperada.
- Error de trama. Indica que el dato recibido no tiene un bit de *stop* válido.
- Solicitud de *break*. Indica que se ha recibido una trama de pausa (el valor de la lńea se ha mantenido inactivo por un tiempo mayor que el de una trama).
- Error de *timeout* (sólo para gestión por DMA usando colas FIFO). Indica que ha transcurrido un intervalo de tiempo de 3 tramas sin recibir ningún dato, y el nivel de ocupacin no llega al umbral fijado para la transmisión. Esto impide que el sistema se quede esperando por nuevos datos que quizá no vayan a llegar.

3.3 Control de flujo

La UART del S3C44BOX admite control de flujo. En este caso el emisor debe esperar a que el receptor esté preparado para recibir. Esto se consigue añadiendo una línea adicional de control al puerto. Así la señal *nCTS* [1] (*Clear To Send*) del emisor se conecta con la señal *nRTS* (*Request To Send*) del receptor. Cuando éste está listo para recibir activa la señal *nRTS*. El emisor notará que se activa su señal *nCTS* y entonces podrá enviar el dato (ver Figura 3).

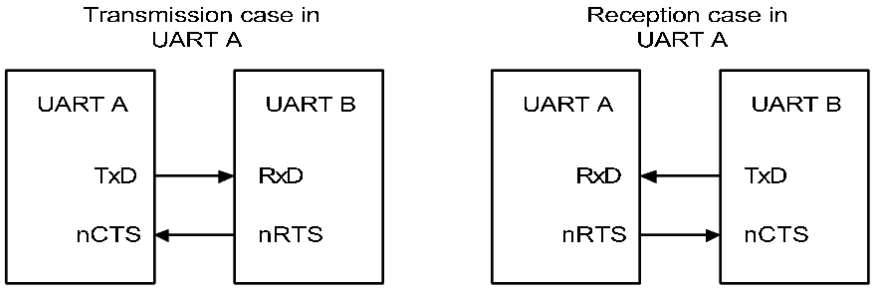

Figura 3: Conexión de dos UARTs empleando líneas de control de flujo.

En la UART del S3C44BOX la gestión de estas líneas puede llevarse a cabo de manera automática por HW (*Auto Flow Control*), o bien manualmente por SW, escribiendo en los registros de control y estado del módem (*UMSTATn* y *UMCONn*). Es preciso resaltar que el control de flujo automático sólo se puede emplear cuando se conectan dos UART entre sí. Para la conexión de un módem, es preciso usar el control SW, y, si se desea emplear un interfaz RS-232, es preciso usar además los puertos de E/S generales (ver Figura 4).

3.4 Velocidad en baudios

La velocidad de trabajo de la UART se establece dividiendo la señal de reloj principal (*MCLK*), 64MHz en nuestro caso, por un valor almacenado en el registro *UBRDIVn*. Dicho valor se obtiene mediante la fórmula:

$$UBRDIVn = (round\_off)(MCLK/(bps \times 16)) - 1$$

El resultado de esta división debe estar comprendido entre 1 y $2^{16} - 1$. Por ejemplo si la velocidad que se desea es de 115200 bps y la frecuencia de *MCLK* fuese de 40 MHz:

$$\begin{aligned} UBRDIVn &= (int)(40000000/(115200 \times 16) + 0.5) - 1 \\ &= (int)(21.7 + 0.5) - 1 \\ &= 22 - 1 = 21 \end{aligned}$$

[1] Es preciso destacar que aquí la n indica que la señal se activa a baja.

3.5 Modo loop-back

La UART del S3C44BOX proporciona un modo de test que podríamos denominar de lazo cerrado o *loop-back*, cuyo propósito es ayudar a aislar errores en la comunicación y comprobar la corrección del SW. En este modo, el dato enviado es inmediatamente recibido por la propia UART.
La activación de este modo de test se realiza mediante un bit del registro de control de la UART (*UCONn*).

3.6 Configuración de la UART

A continuación enunciaremos los principales registros de la UART, describiendo la funcionalidad de cada uno de ellos (para ampliar la descripción consultar el Manual de Referencia del S3C44BOX [1]):

UART Line Control Register (*ULCONn*: 0x01D00000, 0x01D04000). Este registro se utiliza para la configuración de la línea de los puertos serie:

- Activar/desactivar modo IrDA.
- Configurar paridad (none/even/odd).
- Determinar bits de stop (1/2).
- Longitud del carácter (5/6/7/8).

La descripción del registro aparece en la Tabla 1.

UART Control Register (*UCONn*: 0x01D00004, 0x01D04004). Este registro se utiliza para configurar el funcionamiento de la UART:

- Tipo de interrupción de envío/recepción (pulso/flanco).
- Activar interrupción por timeout.
- Activar interrupción por error (*break*, *frame*, *parity*, *overrun*).
- Activar modo *loop-back*.
- Mandar señal de *break*.
- Determinar modo de transmisión para envío/recepción (desactivado, interrupción/encuesta o DMA).

La descripción del registro aparece en la Tabla 2.

Table 1: Registros de la UART *ULCONn* para configuración de la línea

ULCONn	Bit	Description
Reserved	[7]	
Infra-Red Mode	[6]	The Infra-Red mode determines whether or not to use the Infra-Red mode. 0 = Normal mode operation 1 = Infra-Red Tx/Rx mode
Parity Mode	[5:3]	The parity mode specifies how parity generation and checking are to be performed during UART transmit and receive operation. 0xx = No parity 100 = Odd parity 101 = Even parity 110 = Parity forced/checked as 1 111 = Parity forced/checked as 0
Number of stop bit	[2]	The number of stop bits specifies how many stop bits are to be used to signal end-of-frame. 0 = One stop bit per frame 1 = Two stop bit per frame
Word length	[1:0]	The word length indicates the number of data bits to be transmitted or received per frame. 00 = 5-bits 01 = 6-bits 10 = 7-bits 11 = 8-bits

UART FIFO Control Register (*UFCONn*: 0x01D00008, 0x01D04008). Estos registros se utilizan para la configuración de las colas FIFO de la UART:

- Habilitar FIFO.
- Determinar el nivel de ocupación con el que se desencadenan las interrupciones de envío/recepción FIFO.
- Borrar FIFO.

La descripción del registro aparece en la Tabla 3.

UART MODEM Control Register (*UMCONn*: 0x01D0000C, 0x01D0400C). Registro utilizado para configurar el control de flujo de la UART:

- Habilitar control de flujo automático (AFC).
- Activar la señal RTS (Ready To Send) cuando el control de flujo es SW.

La descripción del registro aparece en la Tabla 4.

UART Tx/Rx Status Register (*UTRSTATn*: 0x01D00010, 0x01D04010). Registros de estado de transmisión y recepción, utilizados para:

- Determinar si el shifter de transmisión está vacío.

Table 2: Registros *UCONn* para configuración del funcionamiento de la UART

UCONn	Bit	Description
Tx interrupt type	[9]	Interrupt request type 0 = Pulse (Interrupt is requested the instant Tx buffer becomes empty) 1 = Level (Interrupt is requested while Tx buffer is empty)
Rx interrupt type	[8]	Interrupt request type 0 = Pulse (Interrupt is requested the instant Rx buffer receives the data) 1 = Level (Interrupt is requested while Rx buffer is receiving data)
Rx time out enable	[7]	Enable/Disable Rx time out interrupt when UART FIFO is enabled. The interrupt is a receive interrupt. 0 = Disable 1 = Enable
Rx error status interrupt enable	[6]	This bit enables the UART to generate an interrupt if an exception, such as a break, frame error, parity error, or overrun error occurs during a receive operation. 0 = Do not generate receive error status interrupt 1 = Generate receive error status interrupt
Loop-back Mode	[5]	Setting loop-back bit to 1 causes the UART to enter the loop-back mode. This mode is provided for test purposes only. 0 = Normal operation 1 = Loop-back mode
Send Break Signal	[4]	Setting this bit causes the UART to send a break during 1 frame time. This bit is auto-cleared after sending the break signal. 0 = Normal transmit 1 = Send break signal
Transmit Mode	[3:2]	These two bits determine which function is currently able to write Tx data to the UART transmit holding register. 00 = Disable 01 = Interrupt request or polling mode 10 = BDMA0 request (Only for UART0) 11 = BDMA1 request (Only for UART1)
Receive Mode	[1:0]	These two bits determine which function is currently able to read data from UART receive buffer register. 00 = Disable, 01 = Interrupt request or polling mode 10 = BDMA0 request (Only for UART0) 11 = BDMA1 request (Only for UART1)

Table 3: Registros *UFCONn* para configuración de las colas FIFO de la UART

UFCONn	Bit	Description
Tx FIFO Trigger Level	[7:6]	These two bits determine the trigger level of transmit FIFO. 00 = Empty 01 = 4-byte 10 = 8-byte 11 = 12-byte
Rx FIFO Trigger Level	[5:4]	These two bits determine the trigger level of receive FIFO. 00 = 4-byte 01 = 8-byte 10 = 12-byte 11 = 16-byte
Reserved	[3]	
Tx FIFO Reset	[2]	This bit is auto-cleared after resetting FIFO 0 = Normal 1= Tx FIFO reset
Rx FIFO Reset	[1]	This bit is auto-cleared after resetting FIFO 0 = Normal 1= Rx FIFO reset
FIFO Enable	[0]	0 = FIFO disable 1 = FIFO mode

Table 4: Registros *UFCONn* para configurar el control de flujo de la UART

UMCONn	Bit	Description
Reserved	[7:5]	These bits must be 0's
AFC(Auto Flow Control)	[4]	0 = Disable 1 = Enable
Reserved	[3:1]	These bits must be 0's
Request to Send	[0]	If AFC bit is enabled, this value will be ignored. In this case the S3C44B0X will control nRTS automatically. If AFC bit is disabled, nRTS must be controlled by S/W. 0 = 'H' level(Inactivate nRTS) 1 = 'L' level(Activate nRTS)

Table 5: Registros *UTRSTAT* con información del estado de la transmisión y la recepción

UTRSTATn	Bit	Description
Transmit shifter empty	[2]	This bit is automatically set to 1 when the transmit shift register has no valid data to transmit and the transmit shift register is empty. 0 = Not empty 1 = Transmit holding & shifter register empty
Transmit buffer empty	[1]	This bit is automatically set to 1 when the transmit buffer register does not contain valid data. 0 =The buffer register is not empty 1 = Empty If the UART uses the FIFO, users should check Tx FIFO Count bits and Tx FIFO Full bit in the UFSTAT register instead of this bit.
Receive buffer data ready	[0]	This bit is automatically set to 1 whenever the receive buffer register contains valid data, received over the RXDn port. 0 = Completely empty 1 = The buffer register has a received data If the UART uses the FIFO, users should check Rx FIFO Count bits in the UFSTAT register instead of this bit.

- Determinar si el buffer de transmisión está vacío.
- Determinar si el buffer de recepción tiene datos listos.

La descripción del registro aparece en la Tabla 5.

UART Rx Error Status Register (*UERSTATn*: 0x01D00014, 0x01D04014). Determina el tipo de error que ha desencadenado la interrupción por error en la recepción. Su descripción aparece en la Tabla 6.

UART FIFO Status Registers (*UFSTATn*: 0x01D00018, 0x01D04018). Registros de estado para las colas FIFO de la UART, utilizados para:

- Determinar si el FIFO de envío/recepción está lleno/vacío.
- Determinar el n de elementos presentes en la cola FIFO.

La descripción del registro aparece en la Tabla 7.

Table 6: Registros *UERSTAT* con información del estado de los errores en la recepción.

UERSTATn	Bit	Description
Break Detect	[3]	This bit is automatically set to 1 to indicate that a break signal has been received. 0 = No break receive 1 = Break receive
Frame Error	[2]	This bit is automatically set to 1 whenever a frame error occurs during receive operation. 0 = No frame error during receive 1 = Frame error
Parity Error	[1]	This bit is automatically set to 1 whenever a parity error occurs during receive operation. 0 = No parity error during receive 1 = Parity error
Overrun Error	[0]	This bit is automatically set to 1 whenever an overrun error occurs during receive operation. 0 = No overrun error during receive 1 = Overrun error

Table 7: Registros *UFSTAT* con información del estado de las colas FIFO de la UART.

UFSTATn	Bit	Description
Reserved	[15:10]	
Tx FIFO Full	[9]	This bit is automatically set to 1 whenever transmit FIFO is full during transmit operation 0 = 0-byte ≤ Tx FIFO data ≤ 15-byte 1 = Full
Rx FIFO Full	[8]	This bit is automatically set to 1 whenever receive FIFO is full during receive operation 0 = 0-byte ≤ Rx FIFO data ≤ 15-byte 1 = Full
Tx FIFO Count	[7:4]	Number of data in Tx FIFO
Rx FIFO Count	[3:0]	Number of data in Rx FIFO

Table 8: Registros *UMSTAT* para observar la línea CTS en el control de flujo software.

UMSTATn	Bit	Description
Delta CTS	[4]	This bit indicates that the nCTS input to S3C44B0X has changed state since the last time it was read by CPU. (Refer to Fig. 10-7) 0 = Has not changed 1 = Has changed
Reserved	[3:1]	Reserved
Clear to Send	[0]	0 = CTS signal is not activated(nCTS pin is high) 1 = CTS signal is activated(nCTS pin is low)

Table 9: Direcciones de los registros *URXH*.

Register	Address	R/W	Description	Reset Value
URXH0	0x01D00024(L) 0x01D00027(B)	R (by byte)	UART channel 0 receive buffer register	–
URXH1	0x01D04024(L) 0x01D04027(B)	R (by byte)	UART channel 1 receive buffer register	–

UART Modem Status Register (*UMSTATn*: 0x01D0001C, 0x01D0401C). Gestión de la señal CTS (*Clear To Send*) en el control de flujo SW. Su descripción aparece en la Tabla 8.

UART Transmit Holding Register (*UTXHn*) y UART Receive Holding Register (*URXHn: UTXH0, UTXH1*). *UTXHn* son los registros en los que debemos escribir el dato que se desea transmitir. En *URXHn* podremos leer el dato recibido. Debemos resaltar que cuando se produce un error de superposición (*overrun*) se debe leer *URXHn* o de lo contrario, al recibir el siguiente dato, se volverá a producir un nuevo error. Las direcciones de estos registros están descritas en las Tablas 9 y 10:

UART Baud Rate Division Register (*UBRDIV*: 0x01D00028, 0x01D04028). Permite establecer la frecuencia de comunicación. La fórmula para calcular el ratio y la velocidad en baudios se ha explicado anteriormente en la sección 3.4.

3.7 Conexión de la UART a los puertos serie de la placa

La Figura 4 muestra cómo se establece la conexión entre el S3C44BOX y los conectores DB9 de la placa S3CEV40. Como puede apreciarse en la figura, el conector UART0 dispone

Table 10: Direcciones de los registros *UTXH*.

Register	Address	R/W	Description
UTXH0	0x01D00020(L) 0x01D00023(B)	W (by byte)	UART channel 0 transmit holding register
UTXH1	0x01D04020(L) 0x01D04023(B)	W (by byte)	UART channel 1 transmit holding register

únicamente de dos lneas (*RXD* y *TXD*) y por lo tanto sólo puede emplearse para comunicaciones serie simples (sin control de flujo). El conector UART1 además de las líneas de transmisión y recepción, está conectado a 6 pines de E/S generales lo que permite que, con una gestión adecuada de los mismos, se pueda conectar a un módem RS-232.

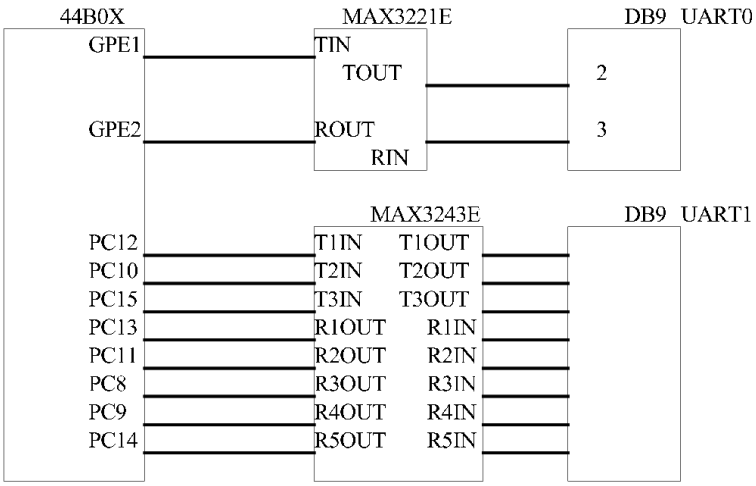

Figura 4: Conexión de la UART a los puertos serie de la placa (DB9).

4 Desarrollo de la práctica

En esta práctica vamos a dividir el trabajo en dos partes. En la primera, completamente guiada, nos familiarizaremos con el uso de UART y la comunicación serie con el PC.

En la segunda, partiendo del código resuelto que tenemos para la primera parte se tendrá que conectar el maletín a otro maletín por medio de la UART1.

4.1 Primera parte

En esta primera parte comunicaremos la placa Embest con el PC del laboratorio y analizaremos un programa sencillo que realiza una comunicación serie usando la UART mediante *polling* (E/S programada con espera de respuesta).

1. Al conectar el maletín por USB, el PC crea un puerto serie virtual denominado *COMX*, siendo *X* un número. Para saber cuál es el valor de *X*, se puede consultar el *Administrador de dispositivos* de Windows.

2. Probamos la conexión serie entre ambos. Para ello:

   - Abrir el programa *Termite*, situado en *${eclipse_home}/../termite30*.

- Pulsar en *Settings* y seleccionar el puerto COM correspondiente.
- Introducir la siguiente configuración: 115200 bps, 8 bits, sin paridad, 1 bit de parada, sin control de flujo (ver Figura 5).

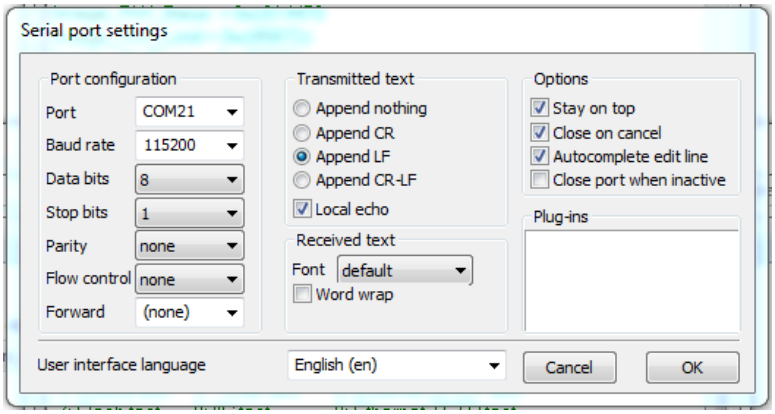

Figura 5: Ventana del programa *Termite*, pestaña *Settings*.

- Encender la placa y comprobar que se muestra el mismo mensaje que en el LCD (ver Figura 6).

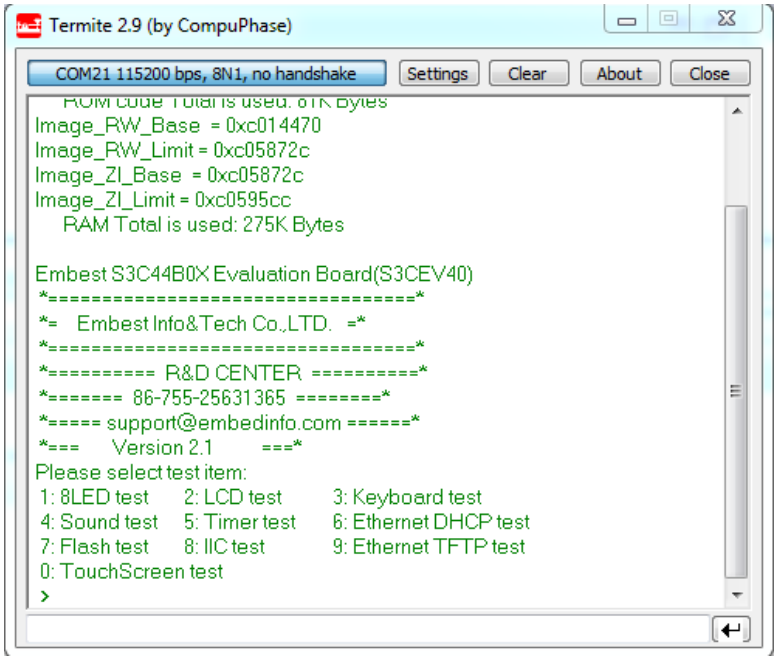

Figura 6: Mensaje por defecto de la placa S3CEV40 enviado por el puerto serie UART0.

3. Descomprimimos el fichero *p4Skeleton.zip* y creamos un proyecto nuevo.
4. Configuramos el proyecto, sin olvidarnos de añadir las librerias *c, gcc* y *nosys*, así

como los path asociados, tal y como se hacía en la práctica 3. El diagrama de flujo del programa dado se muestra en la Figura 7:

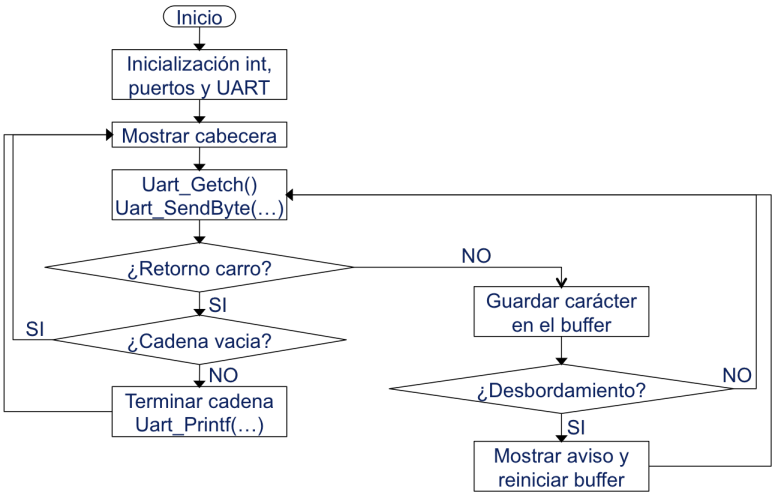

Figura 7: Diagrama de flujo del programa principal.

4.2 Segunda parte

Establecer comunicación entre dos maletines a través de sus respectivos puertos UART1. Dicha comunicación consistirá en enviar desde cada maletín un byte que indentifique el botón pulsado. El montaje resultante dará como resultado que al pulsar en un maletín el botón 1 o 2, cambiará el estado en el otro maletín de los leds 1 ó 2, respectivamente.

4.2.1 Opcional

Incorporar el teclado matricial al programa anterior, de tal forma que la tecla que se pulse en un maletín se muestre en el 8-segmentos del otro maletín.

Bibliografía

[1] S3C44B0X RISC Microprocessor. Accesible en el Campus Virtual.