Transtech

SIO232 and SIO422 Driver Software
Version 1.1

1 Introduction

This version of the SIO driver software is written in Occam under the TDS but can be ported to the toolset with little effort. The SIO driver is a stand alone program that is loaded onto the SIO TRAM and handles a simple character and string based communication protocol from a Transputer link. Any other Occam or C program can communicate with the driver if this protocol is observed.

2 Communication Protocols

There are two basic communication protocols to the SIO driver. These are:

tag; char	*BYTE; BYTE*
tag; word.count::string	*BYTE; INT16::[]BYTE*

These protocols allow character, string, and command information to be sent to the SIO driver process running on the SIO TRAM. Data is always returned from the driver as single character information. Return data will always be of the form:

tag; char	*BYTE; BYTE*

The demonstration program called TEST_EXE.OCC has examples of how to receive character data and how to locally buffer data until a specified termination character is received.

3 Driver Commands

A command protocol is used to enable or disable hardware handshaking, select one of several preset baud rates, and also to set specially programmed baud rates. Not all of the UARTs possible baud rates are implemented using the preset values, but the capability of programming any rates is provided by directly setting the UARTs *auxiliary control register* (ACR), *mode register 0* (MR0A), and *clock select registers* (CSR).

Note: Care must be taken when programming baud rates using either the preset value or specially programmed values because PORT A and PORT B are not fully independent.

3.1 Preset Command Protocol

There are two tags which are used to signal the driver that the following character is a command. One tag is used to set the characteristics of port A and another is used for port B. Each tag causes an operation on one of the two different ports but the affects are the same otherwise.

Command Protocol:

out ! sio.cmd.A; char or
out ! sio.cmd.B; char

sio.cmd.A causes the following command character to change the characteristics of port A.

sio.cmd.B causes the following command character to change the characteristics of port B.

The character (char) that follow the command tag must be one of the following command instruction bytes:

sio.enable.hs ***Hardware Handshake Command***
sio.disable.hs

sio.baud.1200 ***Normal Baud rates***
sio.baud.2400
sio.baud.4800
sio.baud.9600
sio.baud.19.2K

sio.baud.28.8K ***Extended Baud Rates***
sio.baud.57.6K
sio.baud.115.2K

sio.timer.mode ***Counter/Timer Baud Rate***

Therefore, a typical command would look as follows:

out ! sio.cmd.A; sio.baud.1200

3.2 Preset Command Description

sio.enable.hs enables hardware RTS/CTS handshake checking on the specified communication port.

sio.disable.hs disables hardware RTS/CTS handshake checking on the specified port.

Reference TABLE 3. on page 15 of the Signetics UART manual for further explanation of baud rate interdependencies.

Baud rates are programmed from within two groups of preset values called *normal baud rates* and *extended baud rates*. This version of the software permanently sets bit[7] := 1 of the ACR which selects set 2 of the baud rate generator from the both normal and extended rates.

Preset defaults: (reference Table 3, p15)

ACR[7] := 1
MR0[0] := 0 or 1

The difference between the two groups of baud rates is the value of MR0A[0] which is either set to 0 or 1. There is no corresponding mode bit for MR0B[] so when MR0A is changed the effects are seen for both PORT A and B.

This means that when a baud rate for one port is selected from one group the baud rate for the other port must be from the same group.

Normal Baud Group

sio.baud.1200 Sets the specified port baud rate at 1200.
sio.baud.2400 Sets the specified communication port baud rate at 2400
sio.baud.4800 Sets the specified port baud rate at 4800
sio.baud.9600 Sets the specified prot baud rate at 9600
sio.baud.19.2K Sets the specified port baud rate at 19.2 Kbaud

Extended Baud Group

sio.baud.28.8K Sets the specified port baud rate at 28.8 Kbaud
sio.baud.57.6K Sets the specified port baud rate at 57.6 Kbaud
sio.baud.115.2K Sets the specified port baud rate at 115.2 Kbaud

The current version of the driver has been tested to handle communication on both ports at 115.2 Kbaud per channel for long strings with no character overrun or error. This test was performed

without using hardware handshake testing. If handshake testing is used than the UART device automatically holds off character transfer until the receiving device is ready for another character. This insures that data is not lost or corrupted.

3.3 Specially Programmed Baud Rates

Specially programmed baud rates can be used by setting a particular port to use the on-chip timer as the baud rate generator. There is only one timer so values loaded into the counter timer registers (CTLR and CTUR) apply to any channel set to use the timer as the baud generator.

Special Program Mode TAGS

sio.set.ACR; BYTE	Sets Auxiliary Control Register (affects PORTS A and B)
sio.set.MR0A; BYTE	Sets Mode Register 0 for PORT A
sio.set.CSRA; BYTE	Sets Clock Select Register for PORT A
sio.set.CSRB; BYTE	Sets Clock Select Register for PORT B
sio.set.CTR; BYTE; BYTE	Sets Lower and Upper Counter Timer Register

To set a PORT to use the timer as the baud rate generator first the CSR of the port must be set to use the timer as follows:

```
out ! sio.set.CSRA; (BYTE #DD)      or
out ! sio.set.CSRA; sio.timer.mode
```

Then the proper value must be loaded into the Counter Timer Register (CTR) as follows:

$$\text{Timer Value} == \frac{3.6864 \times 10^6}{32 * \text{baud rate}}$$

The 16-bit timer value must be loaded as two BYTE values. Regardless of the value, both BYTES must be loaded. The sequence is low_byte; high_byte as follows:

```
sio.set.CTR; (low_byte); (high_byte)
```

For example to set PORT A to use the timer at 9600 baud the following would be used:

```
out ! sio.set.CSRA; set.timer.mode
out ! sio.set.CTR; (BYTE 12); (BYTE 0)
```

Note: The maximum baud rate attainable using the timer is 57.6 Kbaud. The minimum value that can be loaded into the CTR register is 2.

3.4 Special Program Modes

The driver provides access to the *Auxiliary Control Register* and *Mode Register.A.0* so that all possible baud rates can be programmed. The user should consult TABLE 3. on page 15 for specific programming examples and port interdependencies

4 Driver Communication

Data is sent to the driver over a Transputer link as byte or character data. This data is then sent to the specified UART channel for RS-232 or RS-422 transfer. This is done as follows:

4.1 Character Data

To send character data from a Transputer node to the SIO TRAM for transfer out of UART channel A the following occam code is executed:

```
out ! sio.char.A; char          or

SEQ i = 0 FOR (SIZE string)
  out ! sio.char.A; string[i]
```

To send data to UART port B the following code would be used:

```
out ! sio.char.B; char

SEQ i = 0 FOR (SIZE string)
  out ! sio.char.A; string[i]
```

4.2 String Data

String data be sent over a channel in the following way

```
out ! sio.string.A; (INT16 length)::string                    or

out ! sio.string.B; (INT16 length)::string
```

The string length indicator must be a 16-bit integer value specifying the number of bytes to in the string to follow.

4.3 Receiving Data

Data is always sent from the SIO driver to the remote process as tagged bytes in the following way:

```
in ? tag; char
```

The tag will have only two values:

```
sio.char.A  or  sio.char.B
```

The remote user process must check the tag and route the following character accordingly.

APPENDIX 1 Programming Examples

Both ports set from same group

```
out ! sio.cmd.A; sio.baud.1200
out ! sio.cmd.B; sio.baud.19.2K

            or

out ! sio.cmd.A; sio.baud.57.6K
out ! sio.cmd.B; sio.baud.115.2K
```

Enable Hardware Handshaking

```
out ! sio.cmd.A; sio.enable.hs
out ! sio.cmd.B; sio.enable.hs
```

Baud Rates from Different Groups

```
out ! sio.cmd.A; sio.115.2K            -- PORT A set to 115.2 KBaud

                                        -- PORT B set to 1200 baud
out ! sio.set.CSRB; (BYTE #DD)         -- set to use counter/timer as BRG
out ! sio.set.CTR; (BYTE 96); (BYTE 0) -- set CTR for 1200 baud
```

APPENDIX 2 Programming TAGS

```
--
-- SIO Driver Protocol TAGS
--
VAL BYTE sio.cmd.A          IS 1 (BYTE) :
VAL BYTE sio.cmd.B          IS 2 (BYTE) :
VAL BYTE sio.char.A         IS 3 (BYTE) :
VAL BYTE sio.char.B         IS 4 (BYTE) :
VAL BYTE sio.string.A       IS 5 (BYTE) :
VAL BYTE sio.string.B       IS 6 (BYTE) :
VAL BYTE sio.error          IS 11 (BYTE) :
VAL BYTE sio.set.ACR        IS 12 (BYTE) :
VAL BYTE sio.set.MR0A       IS 13 (BYTE) :
VAL BYTE sio.set.CSRA       IS 14 (BYTE) :
VAL BYTE sio.set.CSRB       IS 15 (BYTE) :
VAL BYTE sio.set.CTR        IS 16 (BYTE) :

VAL BYTE sio.enable.hs      IS 31 (BYTE) :  -- enable RTS/CTS handshake
VAL BYTE sio.disable.hs     IS 32 (BYTE) :  -- disable RTS/CTS handshake

VAL BYTE sio.baud.1200      IS 40 (BYTE) :
VAL BYTE sio.baud.2400      IS 41 (BYTE) :
VAL BYTE sio.baud.4800      IS 42 (BYTE) :
VAL BYTE sio.baud.9600      IS 43 (BYTE) :
VAL BYTE sio.baud.19.2K     IS 44 (BYTE) :
VAL BYTE sio.baud.28.8K     IS 45 (BYTE) :
VAL BYTE sio.baud.57.6K     IS 46 (BYTE) :
VAL BYTE sio.baud.115.2K    IS 47 (BYTE) :
VAL BYTE sio.timer.mode     IS (BYTE #DD) :
```