APPLICATION NOTE

ABSTRACT
This application note shows how a Philips' 16C UART can be used to implement a simple RS-485 transmitter and receiver node.

AN10250

Using a Philips 16C UART to implement a simple RS-485 transmitter and receiver node

Author: Dong Nguyen
2004 Aug 20
Supersedes data of 2003 Sep 24

Philips
Semiconductors

PHILIPS

Using a Philips 16C UART to implement a simple RS-485 transmitter and receiver node

AN10250

This application note shows how a Philips' 16C UART can be used to implement a simple RS-485 transmitter and receiver node. The software that drives the UART provides 9-bit message oriented communication protocol that transfers over a RS-485 link. Characters are transferred the same way as the RS-232 asynchronous protocol accept that the parity bit is used to differentiate a data character from a control character.

The software (RS485.C & RS485.H) can be downloaded from our web site:
http://www.philipslogic.com/support/

1.0 DATA FRAME AND RS-485 PROTOCOL

The data frame is a basic unit which is to be sent each time the sender wants to send data to the receiver. The data frame consists of: the receiver address byte, followed by a sequence of data bytes, and ends with an EOF (end of frame) byte. Each character is sent asynchronously; therefore, each must have one start bit, 8 data bits, a parity bit and a stop bit. The total number of bits to be sent for each character is 11 bits. Address and control characters are sent with the parity bit set to '1', while data characters are sent with the parity bit set to '0'.

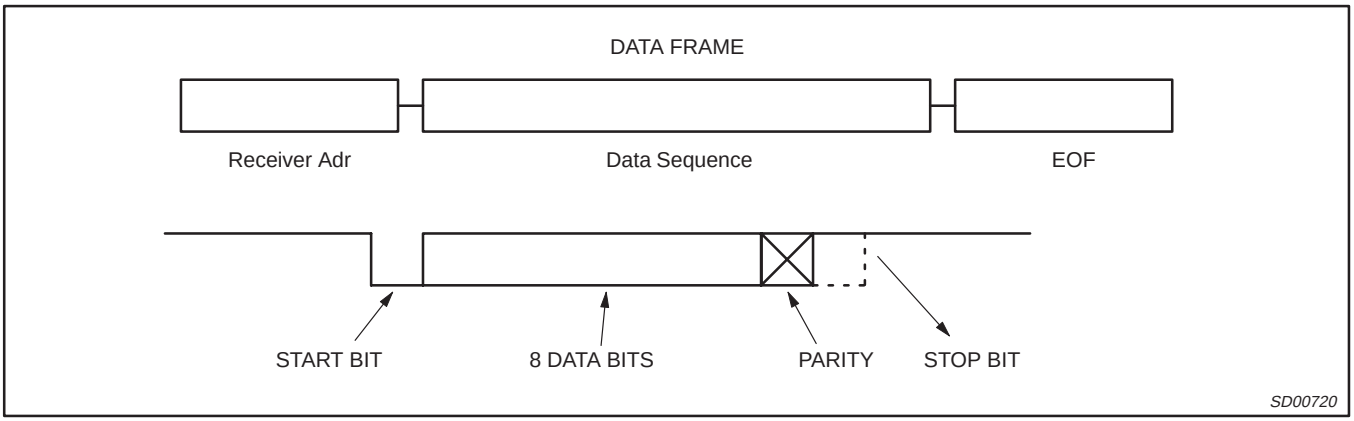

Figure 1.

The data frame begins with the control character (parity bit = 1) set to the address of the receiver. All receivers will wake up to receive this byte. Each receiver then determines if this address matches its own address; the receiver with matching address then configures the UART to receive the rest of the frame. The other receivers will ignore the entire frame until the next "address" byte is sent.

The addressed receiver will receive all characters until the EOF character is received. The receiver then turns around and sends the ACK (acknowledge) character back to the sender.

2.0 SOFTWARE STRUCTURE

2.1 Control characters

Control characters have parity bit set to ‘1’. They are different from data characters which have their parity bit set to ‘0’.

Name	Value	Description
Addresses	0x70 .. 0x7F	Node addresses
EOF_CHAR	0x60	End of frame character
ACK_ACK	0x61	Acknowledge character

2.2 Code structure

The transmit routine is very straightforward; all it has to do is send the address of the receiver followed by the sequence of data. The receiver routine is not that simple—it utilizes the parity detection capability of the UART to try to detect the address character. Since the address character is always sent with the parity bit set to ‘1’, all the software has to do is to set the UART to detect a parity of ‘0’. Once the address character is received, the software checks the LSR register for a parity error. If the parity error is detected, then the address character has been received, otherwise, the received character is not an address character.

The workhorse of the program is the state machine in the interrupt routine, and this state machine is called every time a hardware interrupt is generated. This state machine is driven by a variable called EVENT_FLAG. Depending on the settings of this flag, the following cases are taken care of by the state machine:

- RECEIVE ADDRESS
- RECEIVE DATA
- TRANSMIT FRAME
- SEND ACKNOWLEDGE

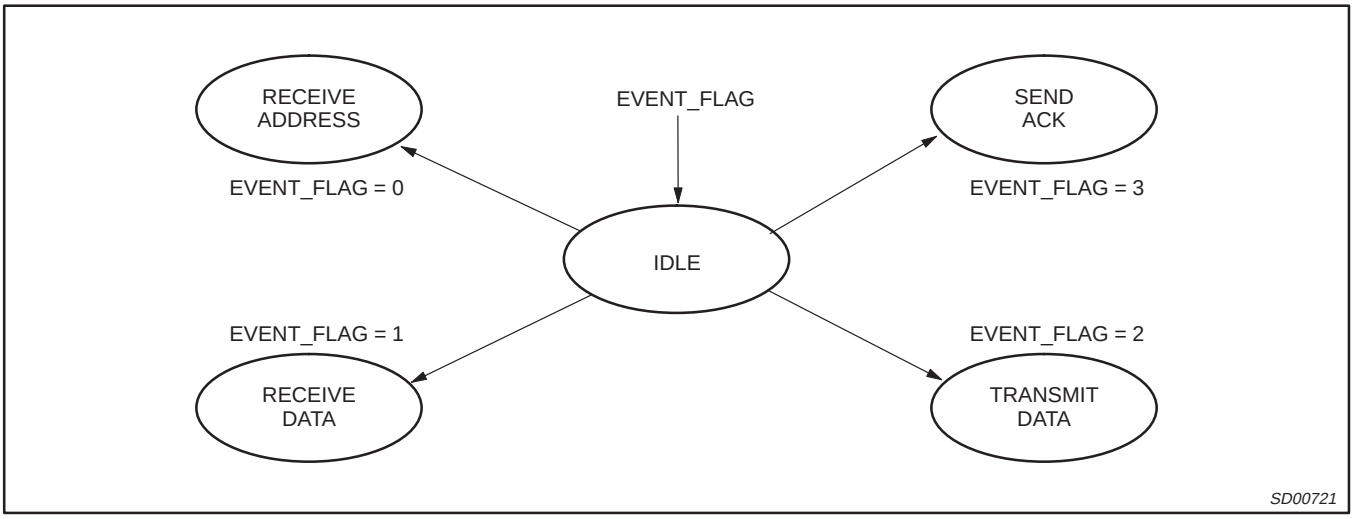

Figure 2.

3.0 SOFTWARE LIMITATIONS

- Other receivers on the link besides the addressed receiver will always see traffic data—whether the data is for them or not.
- There isn't an error checking mechanism sent with the frame, such as CRC.
- The transmitter does not check for the ACK byte sent from the receiver, therefore, it does not have any routine to tack care of the case when the receiver does not receive the entire frame correctly.
- Only a limited number of control bytes are defined: ADDRESS, ACK, EOF.
- The interrupt routine does not handle other interrupt sources such as: line status, modem status.
- The software only works in half-duplex environment.
- The turn-around time for the line transceiver depends on the settings of the FIFO empty bits—LSR bit 6 and bit 5.

4.0 CONCLUSION

Philips' SC16C650/SC16C650B offers other advanced features such as "special character recognition" that can be used to detect the address byte only. If this feature is used by the software, it can prevent other receivers from seeing other traffic data unless a valid address has been received by this receiver.

Certainly users can implement other software features to handle the following cases:

- Check for ACK byte after the frame has been sent.
- Resend the frame if the ACK byte is not received after a given period of time.
- Send a CRC byte with the frame to ensure the frame is received correctly by the receiver.
- Send source address along with the length of the package with the frame.
- Bus free and collision detection.
- Bus request and arbitration.

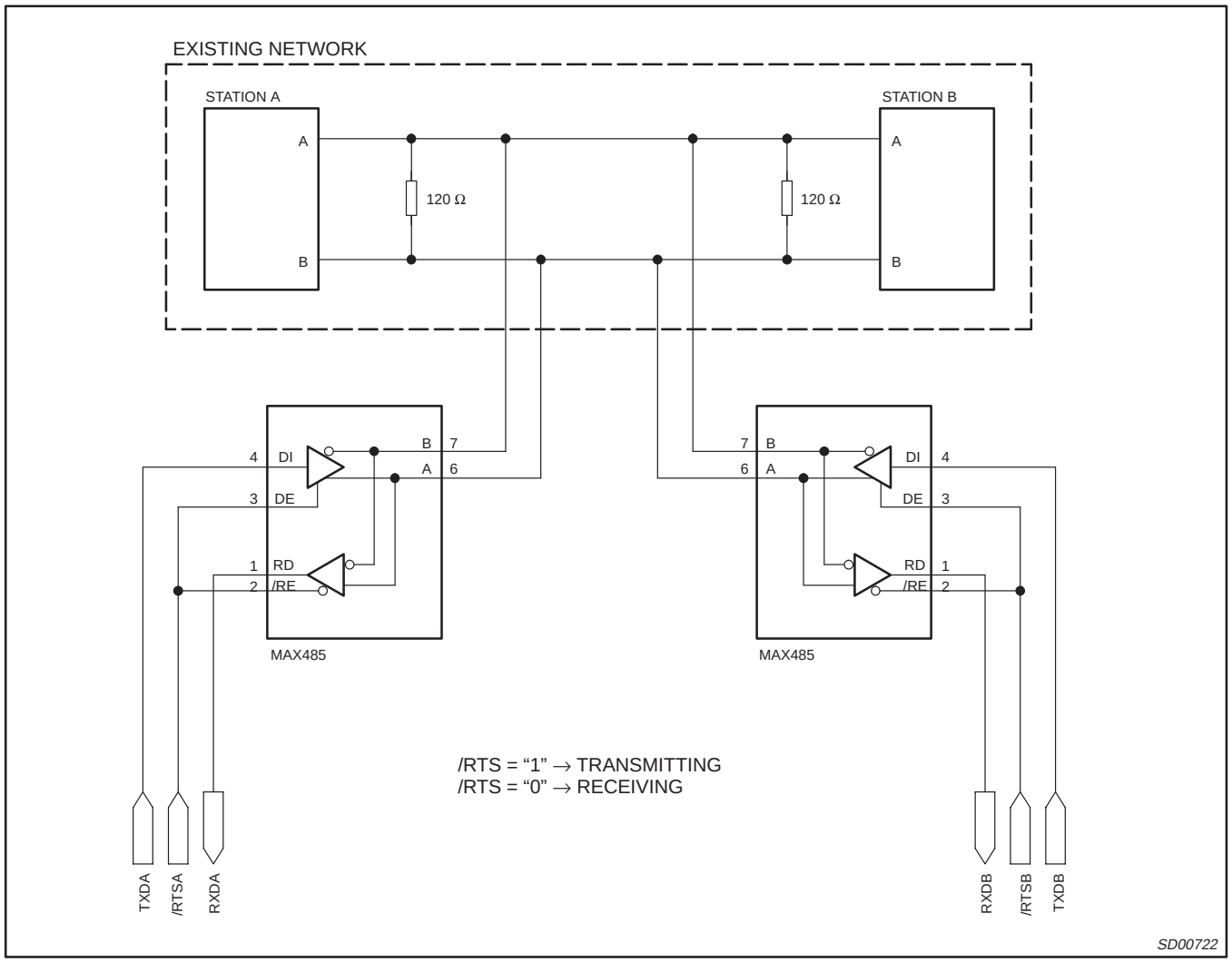
EXISTING NETWORK
STATION A
STATION B
A
B
120 Ω
120 Ω
4 DI
3 DE
1 RD
2 /RE
B 7
A 6
MAX485
7 B
6 A
DI 4
DE 3
RD 1
/RE 2
MAX485
/RTS = “1” → TRANSMITTING
/RTS = “0” → RECEIVING
TXDA
/RTSA
RXDA
RXDB
/RTSB
TXDB
SD00722

Figure 3.

Using a Philips 16C UART to implement a simple RS-485 transmitter and receiver node

AN10250

REVISION HISTORY

Rev	Date	Description
_3	20040820	**Application note (9397 750 13518).** Modifications: • Section 4.0 'Conclusion', first sentence: change from "Philips' SC16C650 offers ... " to "Philips' SC16C650/SC16C650B offers ... "
_2	20030924	**Application note (9397 750 12084).**
_1	20030904	**Application note, initial version (9397 750 11997).**

Contact information

For additional information please visit
http://www.semiconductors.philips.com. Fax: **+31 40 27 24825**

For sales offices addresses send e-mail to:
sales.addresses@www.semiconductors.philips.com.

Date of release: 08-04

Document order number: 9397 750 13518

Let's make things better.

Philips Semiconductors

PHILIPS