

CSCE 612

HDL-Based Systems Design

2003/4/23

Spring 2003 - Lecture 39

Design Project #2 Example – 8251 UART

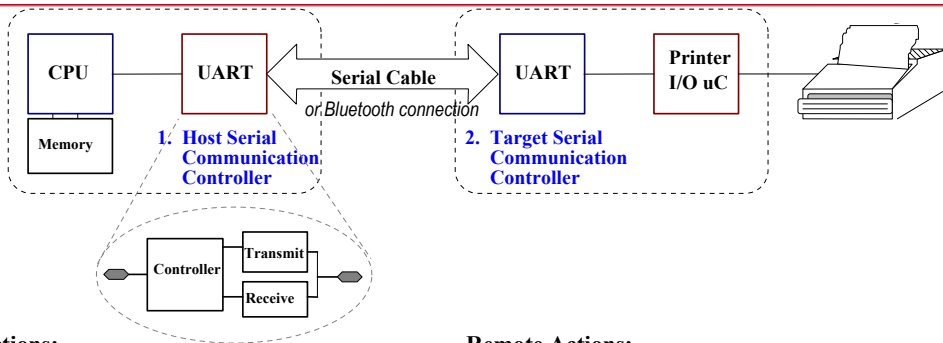
© 2003 Dr. James P. Davis

UART Function Definition-1

- Purpose.
 - ✓ Handle half-duplex serial data traffic between two connection points of a communications channel. Most UARTs are full-duplex, handling both send and received traffic simultaneously. We limit the scope to minimize complexity, to create a design model on which we can layer additional capabilities later.
 - ✓ This gives a very basic model of transmit/receive protocol handshaking, that will be the basis for examining the more complex handshaking associated with the 802.11 protocol.
- Algorithmic approach.
 - ✓ Use a pipelined “delegation” scheme, where the CPU enables the UART controller, and the controller enables either transmitter or receiver.
 - ✓ Data transfer between sender and receiver is done according to a 4-stage handshaking protocol:
 - ✓ RTS: sender endpoint requests to send data
 - ✓ CTS: receiver indicates that sender is “clear” to send the data stream.
 - ✓ The actual data is sent as a bit serial data stream, and the data carries additional parity bits.
 - ✓ ACK: receiver acknowledges that the data was received successfully.
 - ✓ We adopt a sequence of polling loops to synchronize activities of the protocol between sender and receiver end points.
- ASM diagrams/flowcharts.
 - ✓ We create a design with 3 threads, plus a 4th thread that provides an abstraction of the CPU interface (with limited functionality to initiate activity in the UART blocks, to be encapsulated in a VHDL testbench).

See the abstract flowcharts and block diagrams in the following pages.

UART Function Definition-2



Host Actions:

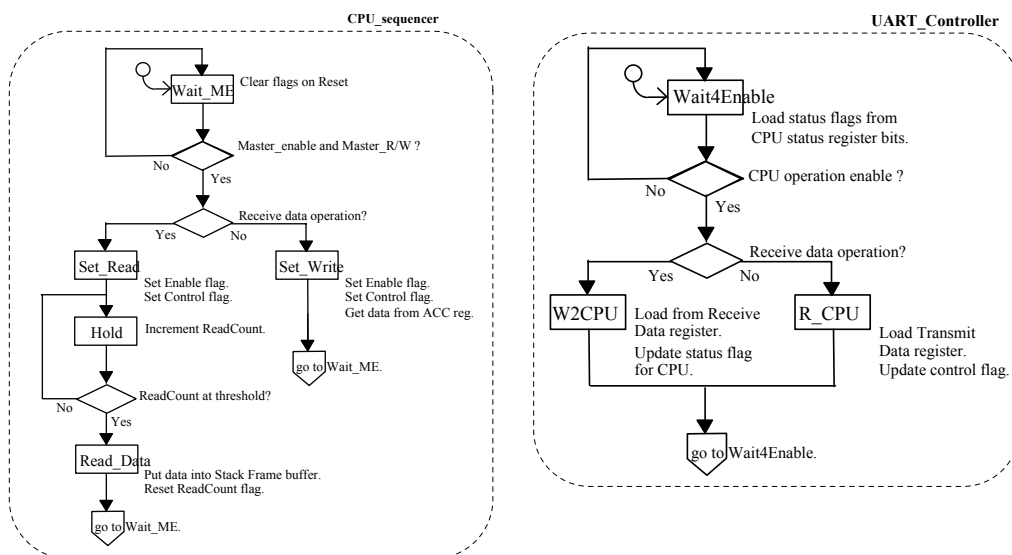
1. Set master enable; set control flags; load 8-bit data word for transmit sequence (LSB is parity bit).
3. Load 'transmit data' register; buffer with start, stop and parity bits; set 'ready to send' flag.
5. Reset counter; shift each of 10-bits onto serial bus; increment transfer counter.
8. Check and clear status flags; setup for next transmit.

Remote Actions:

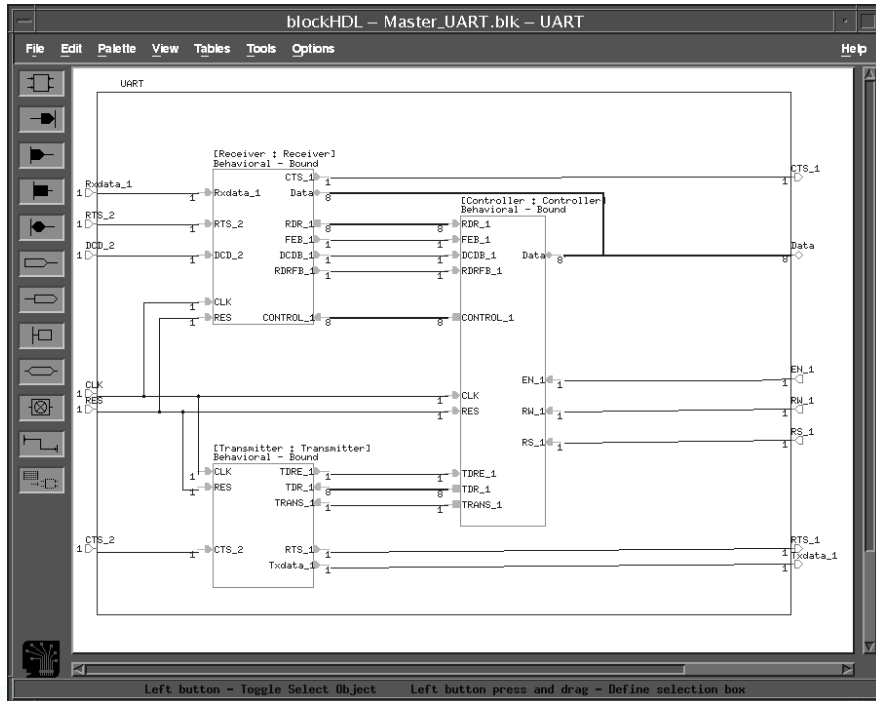
2. Clear flags on Reset; drop 'clear to send' flag; poll for 'ready to send'.
4. Read 'ready to send' and acknowledge with 'clear to send' flag; set up the 'receive data counter'.
6. Shift in each of 10-bits from serial bus into buffer; check parity; strip off start and stop bits.
7. Check and clear status flags; transfer data to buffer.



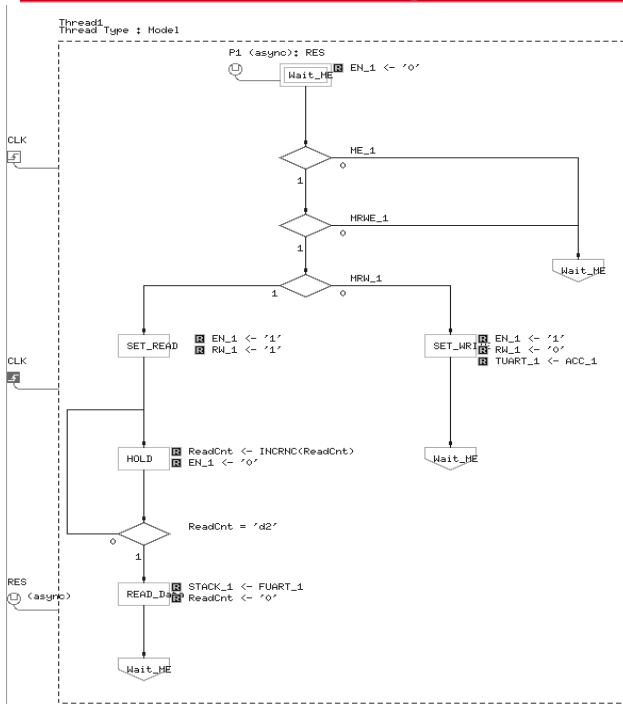
UART Architecture-1



UART Architecture – Blocks and Interfaces



ASM Diagram – CPU Interface



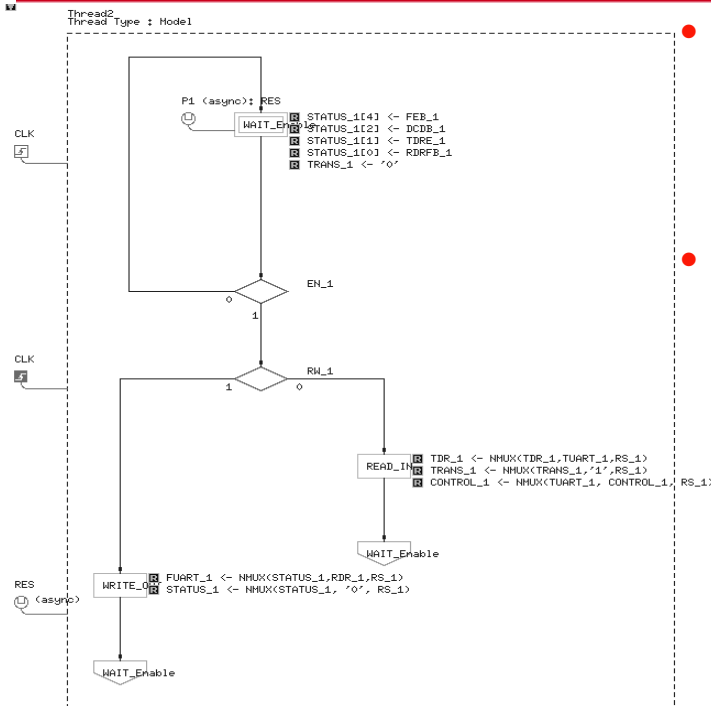
• Purpose.

- ✓ We choose to create an abstract model of the CPU interface. Since our focus is on the UART itself, and not the CPU, we need only model that behavior of the CPU relevant to execution of the UART.

• CPU Interface.

- ✓ The CPU enables the UART controller.
- ✓ EN is enable signal to start the execution of the CPU thread.
- ✓ ME: Master Enable – to kick off the UART controller.
- ✓ MRWE: Master Read/Write enable – to enable the UART controller for the Read/Write functions (there are other UART function modes, but we are not modeling these here).
- ✓ MRW: Master Read/Write select flag - to tell the UART whether the operation will be a Read or Write.

UART Architecture - Controller



• Purpose.

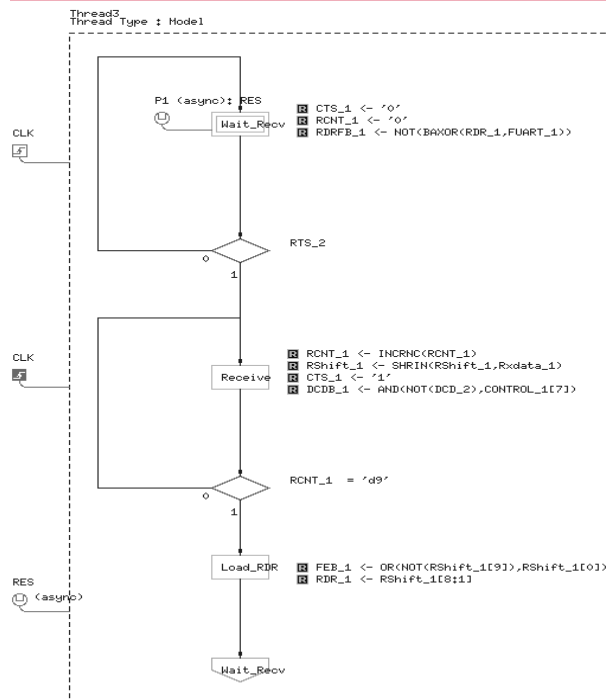
- ✓ We model the basic control functions of the UART first. Since most of its capabilities deal with transmit and receive of data, we model these basic functions.

• UART Controller.

- ✓ The Controller waits to be awakened and selected by the CPU. It loops in the WAIT_En state until the EN_1 signal goes high.
- ✓ The RW_1 signal sets whether the Controller will initiate a read or write operation. This is handled after testing the RW_1 signal, and setting control signals for the Transmit and Receive threads of the UART.



UART Architecture - Receiver



• Purpose.

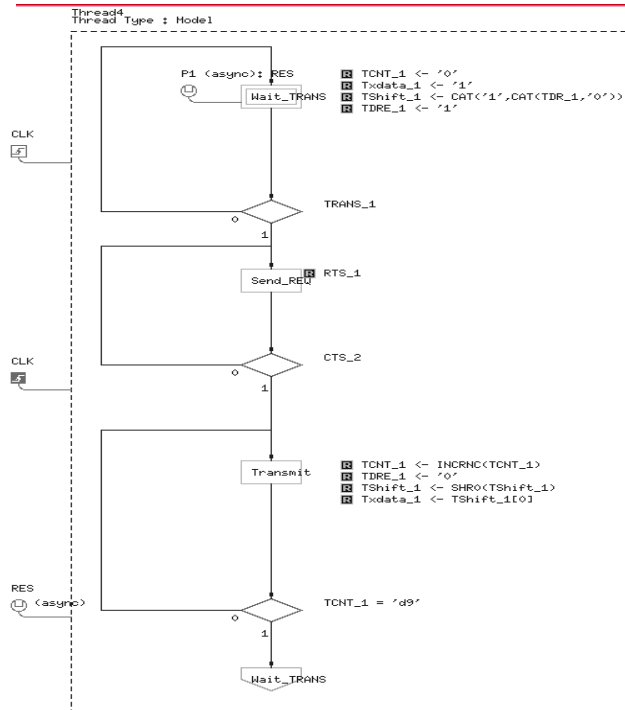
- ✓ We model the basic Receiver functions, most of which deal with waiting for a new data stream and reading this stream, shifting the data, then signaling the Controller.

• UART Receiver.

- ✓ We start up in a poll loop, waiting for the RTS signal, indicating we have a request from the remote host to send data to us.
- ✓ The RTS_2 signal controls whether we initiate preparation to receive data. One thing we do is set the CTS_1 flag, initialize the RCNT counter, and shift in the serial bit.
- ✓ We'll loop on the reading and shifting until the RCNT flag indicates we have a full 10-bit data word (1 start bit, 7 ASCII bits, 1 parity bit, 1 stop bit).
- ✓ We then load the Read Data register RDR, and check the start and parity bits.



UART Architecture - Transmitter



- Purpose.

- ✓ We model the basic Transmitter functions to send data to the remote unit across the serial line.

- UART Transmitter.

- ✓ The Transmitter waits to be signaled by the Controller, by the TRANS_1 signal. A number of signals are sampled in Wait_TRANS state continuously.
- ✓ When enabled, the RTS for request to send signal is set, and we go into a poll loop awaiting the remote side to pull the CTS_2 line, where we will let go of the RTS_1 line.
- ✓ We go into the Transmit poll loop, cycling this state to send out data, bit by bit, incrementing the bit counter on each cycle.
- ✓ We complete when the counter reaches word length.