

iCE65 as 1 to 4 UART Expander

Overview

UART to UART expander provides a cost-effective solution for mobile application processor comprising of single UART to integrate and communicate with multiple UARTs. This design example illustrates the implementation of 1 UART to 4 UART expansion using iCE65 FPGAs.

Description

UART's supports auto-flow control(AFC) signals, viz., nRTS (Request To Send) and nCTS (Clear To Send) signals to connect one UART to another UART. In AFC, nRTS is controlled by condition of the receiver and operation of transmitter is controlled by the nCTS signal. The UART's transmitter transfers the data only when nCTS signal active. Before the UART receives data, nRTS has to be activated.

1 UART to 4 UART expansion module generally used when host processor consists of a single UART, which is to be communicated with multiple UART terminal devices. Data is transmitted when any of the Four UART's (UART1 to UART4) nRTS signal goes low and Host UART(UART0) is ready to communicate serial data to one of the UART(UART 1 to 4). Similarly, Data is received when host UART0's nRTS line goes low. CPU redirects the data to one among UARTs 1 to 4 by sending the appropriate address. Fig 1.1 and Fig 1.2. shows simple UART to UART communicating scheme(Transmission and Reception) for single UARTs.

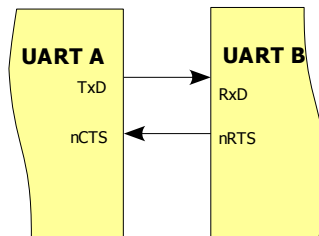


Fig 1.1: UART A as Transmitter

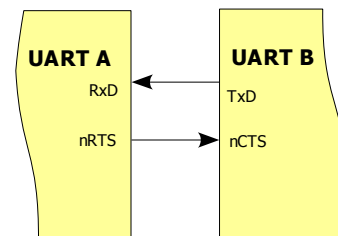


Fig1.2 UART A as Receiver

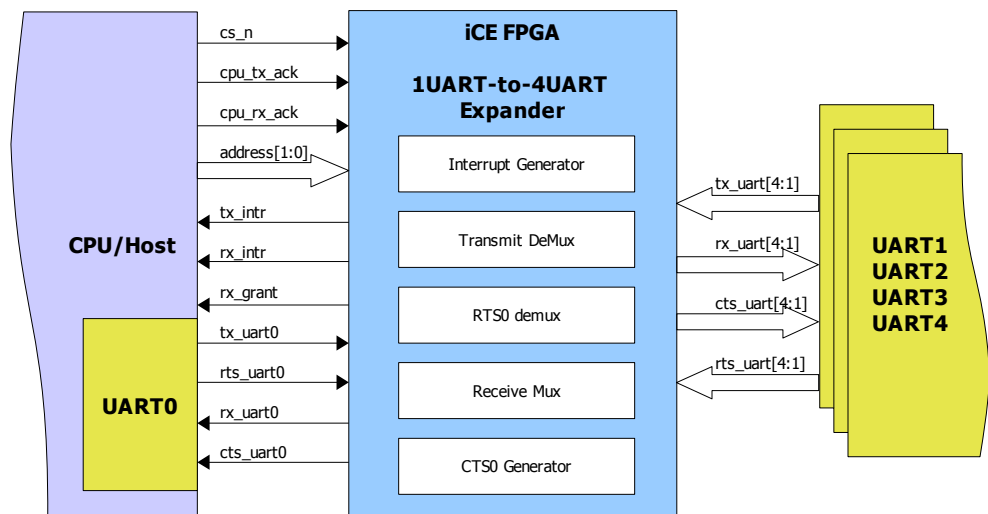


Fig 2: Block diagram of 1 UART to 4 UART Expansion

Implementation

Data Transmission

- UART0 sends a low on rts_uart0 pin when cs_n is low. This in turn generates a low on rx_intr pin.
- CPU processes this interrupt, acknowledges it by sending an active low on cpu_rx_ack pin. Also, places the address of UART to which it wants to send the data.
- Interface logic generates a low on the corresponding cts_uart based on address and connects rx_uart0 to tx_uart of UART corresponding to the address.
- Communication between those UART terminates when CPU brings cs_n high or a CPU sends a different address and falling edge of cpu_rx_ack.

Data Reception

- One or multiple of UART{1,2,3,4} sends rts_uart request to the interface. If any of these UARTs sends a low on rts_uart signals, then interface logic generates an interrupt (active low) on cpu_tx_intr pin.
- CPU processes this interrupt, acknowledges it by sending an active low on cpu_tx_ack pin. Also, places the address of UART from which it would like to receive the data.
- If there is an nRTS low signal from the UART address specified by the CPU (receive address supplied by CPU and rts_uart from UART{1,2,3,4} match), then interface sends a high rx_grant signal to CPU and generates a low on the cts_uart0. Also, connects rx_uart0 to the corresponding tx_uart pin based on the address. .
- Communication between those UART terminates when CPU brings cs_n high or a CPU sends a different address and falling edge of cpu_tx_uart.

Pin	Direction	Description
cs_n	Input	Active low chip select
tx_intr	output	Tx interrupt to CPU
rx_intr	output	Rx interrupt to CPU
cpu_tx_ack	input	Active low Tx Acknowledge from CPU
cpu_rx_ack	input	Active low Rx Acknowledge from CPU
rx_grant	output	Rx grant input to CPU
Address[1:0]	input	Rx/Tx address from CPU
tx_uart0	input	UART0 TX pin
rx_uart0	output	UART0 Rx pin
rts_uart0	input	nRTS pin of UART0
cts_uart0	output	nCTS pin of UART0
tx_uart[4:1]	input	Tx pin of UART 1 to 4
rx_uart[4:1]	output	Rx pin of UART 1 to 4
rts_uart[4:1]	input	nRTS pin of UART 1 to 4
cts_uart[4:1]	output	nCTS pin of UART 1 to 4

Table 2: Pin Description

Device	Logic Cells	IO Cells
iCE65L04-UCB284	18	28

Table 3: Resource Utilization

This design example demonstrates the implementation of a UART to multiple UART using iCE FPGAs. iCE FPGA's very low power capabilities makes iCE FPGAs an obvious choice for implementing a UART to multiple UART for battery operated compact and handheld devices like PDAs, cellular phones etc..

Conclusion

Disclaimer

SiliconBlue is providing this document, design example, or information "as is." SiliconBlue does not ensure that this implementation or information is void of any claims of infringement. As a reader / implementer, you are responsible for obtaining any rights you may require for your implementation. SiliconBlue also does not warranty the fitness of this design example to be readily implemented in any specific context. It is your sole responsibility to validate this design example and its correctness while implementing it for your requirement.