3/11/2022

Lab #6

UART receiver

Spence Johnston
CST 231 – DIGITAL SYSTEM DESIGN I
Spence Johnston

Table of Contents

Abstract

The object of this lab is to receive input from a computer through UART communication. Once received, the character is displayed on the seven segment displays. It then casts lower case letter to uppercase, and outputs it back to the computer.

Two clock dividers were necessary for this lab. Both branch from the system clock. The transmitter needs to run at 9600 Baud. The receiver needs to run at 16 times this speed. There is a manual switch, named reset. There is a UART input that is connected to the USB adapter. The transmit module outputs to the adapter. The seven segment displays on the development board were used to display the input.

The only issue I had with this lab was with timing. I was having an off by one error with my bounds when dealing with a slower clock speed.

Overall, I found this lab to be very educational. It has shown me how to interact with a computer using an embedded device. Although difficult, I found this lab to be very rewarding.

Introduction

The object of this lab is to receive input from a computer through UART communication. Once received, the character is displayed on the seven segment displays. It then casts lower case letter to uppercase, and outputs it back to the computer.

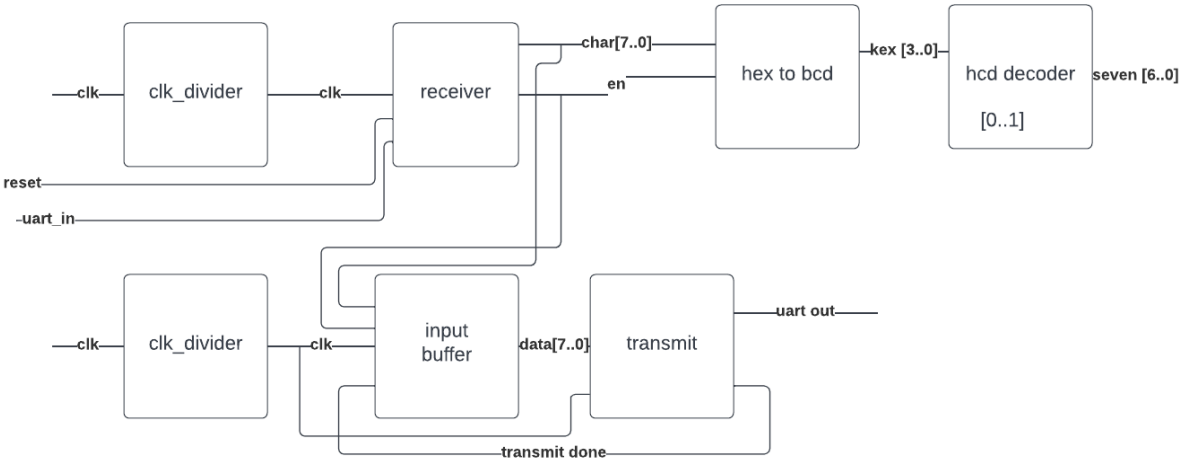

Figure 1: Block diagram

Design

Physical hardware design

The hardware used for this lab was the development board Cyclone V and a USB to UART adapter. No other components were necessary. The board was connected to the adapter via the GPIO pins. One connects to the adapters Rx and Tx cables. A common ground is necessary for reference voltages.

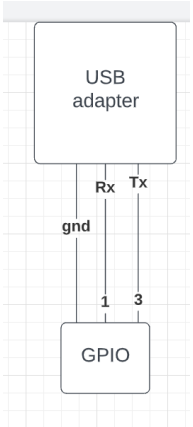

Figure 2: wiring schematic

Synthesized hardware design

Two clock dividers were necessary for this lab. Both branch from the system clock. The transmitter needs to run at 9600 Baud. The receiver needs to run at 16 times this speed. There is a manual switch,

named reset. There is a UART input that is connected to the USB adapter. The transmit module outputs to the adapter. The seven segment displays on the development board were used to display the input.

Design Structure

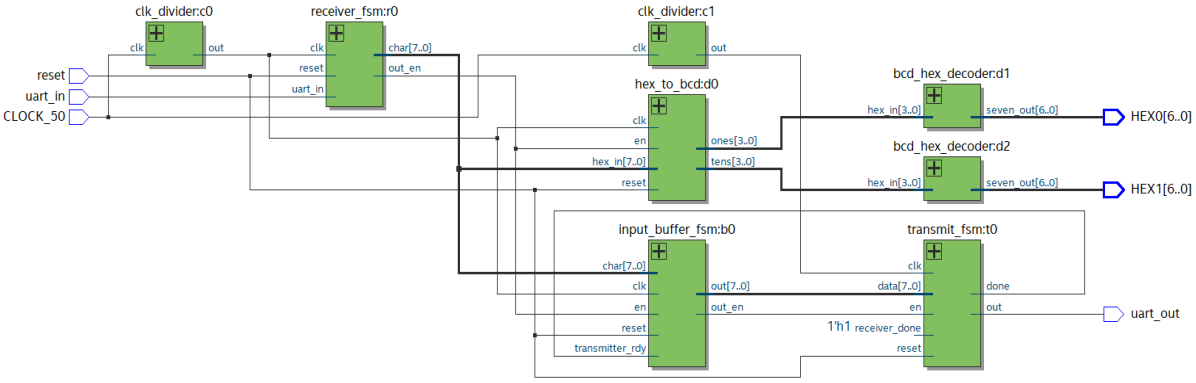

Figure 3: Top level RTL

Modules

Clk_divider

Accepts the system clock. Outputs a slower clock. Two used for this lab. One for the receiver and one for the transmitter. Receiver set to 16 times the Baud rate. See previous labs for further details.

Receiver_fsm

Accepts UART communication. Runs at 16 times the transmitter speed. Polls for data on the eighth pulse. Outputs the character once the data is received. Sends an enable signal. As shown below, the data gets collected then it sends a enable signal via the SEND state.

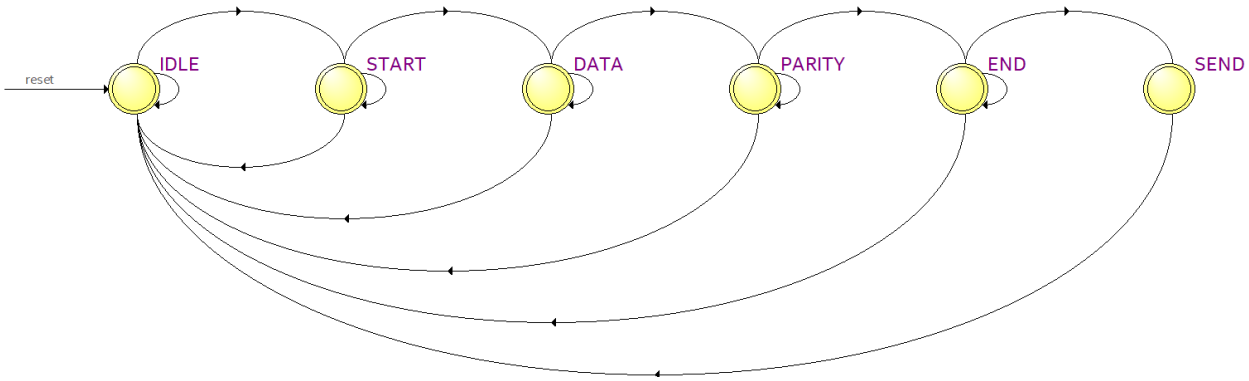

Figure 4: receiver_fsm state diagram

Hex_to_bcd

This module breaks up the value from receiver for seven segment display. Similar to bcd, but for hex. On enable, the state changes to CHANGE. This changes the values on the seven-segment display.

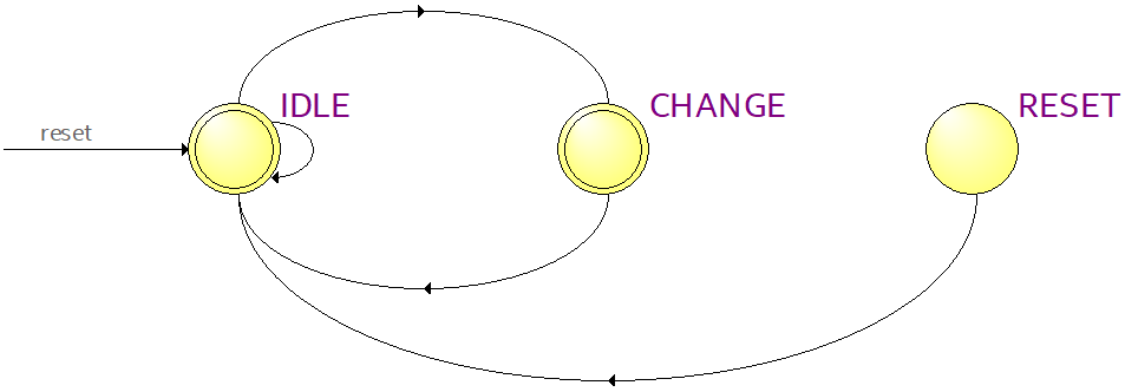

Figure 5:hex_to_bcd state diagram

Bcd_hex_decoder

This decodes a binary coded hex and outputs the appropriate seven segment display output.

Input_buffer_fsm

This module is the buffer between the clock domains. It stores the input and releases it to the transmitter.

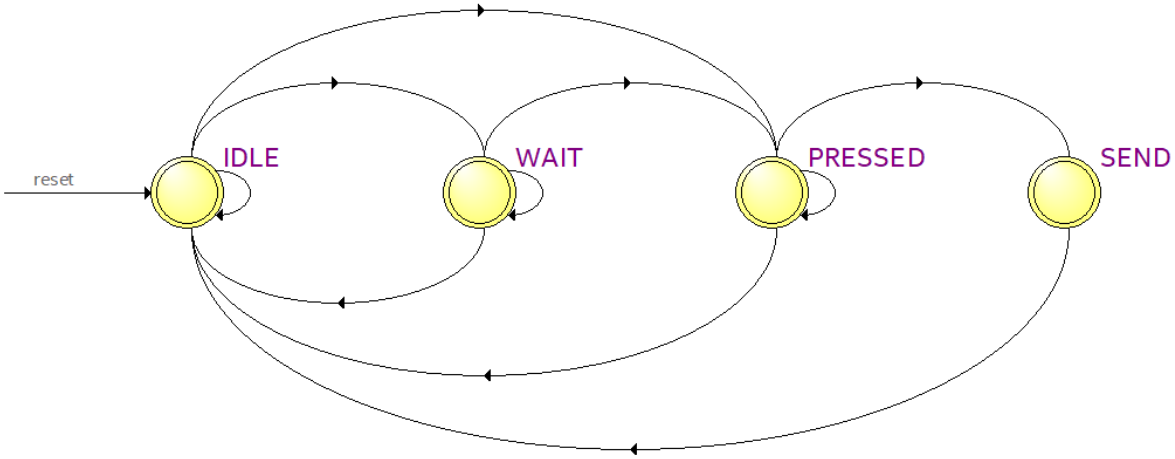

Figure 6: input_buffer_fsm state diagram

Transmit_fsm

This module handles the UART communication. This accepts an enable and a character. It wraps the data in the UART specifications, and outputs it serially.

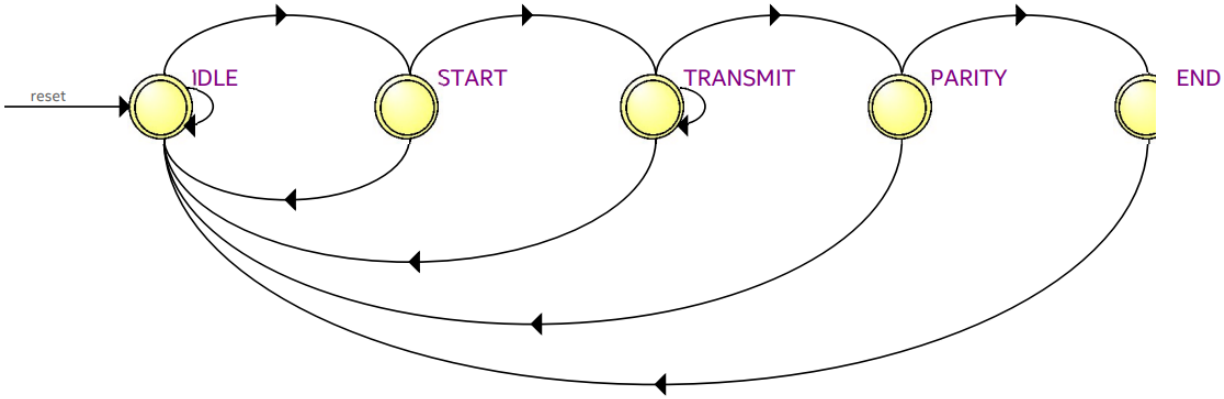

Figure 7: transmit state machine

Simulation and Testing

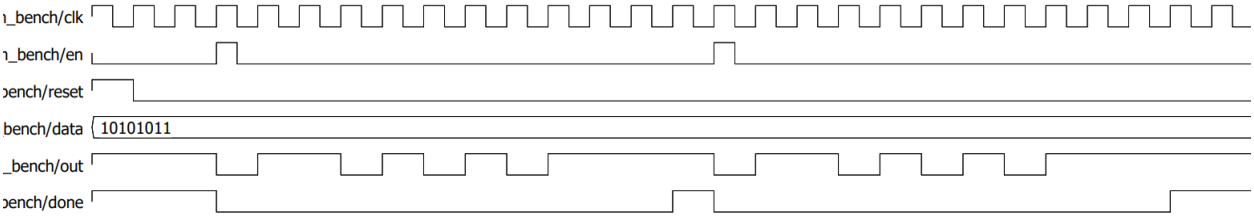

Figure 8: transmit simulation

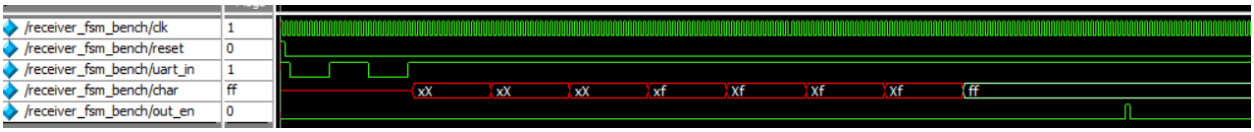

Figure 9: receiver simulation

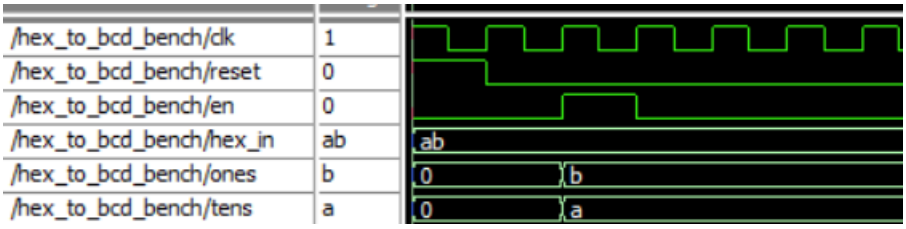

Figure 10: hex_to_bcd simulation

Problems

The only issue I had with this lab was with timing. I was having an off by one error with my bounds when dealing with a slower clock speed.

Results and Conclusion

Overall, I found this lab to be very educational. It has shown me how to interact with a computer using an embedded device. Although difficult, I found this lab to be very rewarding.

Appendix

CLOCK_50	Input	PIN_AF14	3B	B3B_N0	3.3-V LVTTL	16mA ...ault)	
HEX0[6]	Output	PIN_AH28	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX0[5]	Output	PIN_AG28	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX0[4]	Output	PIN_AF28	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX0[3]	Output	PIN_AG27	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX0[2]	Output	PIN_AE28	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX0[1]	Output	PIN_AE27	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX0[0]	Output	PIN_AE26	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX1[6]	Output	PIN_AD27	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX1[5]	Output	PIN_AF30	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX1[4]	Output	PIN_AF29	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX1[3]	Output	PIN_AG30	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX1[2]	Output	PIN_AH30	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX1[1]	Output	PIN_AH29	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
HEX1[0]	Output	PIN_AJ29	5A	B5A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)
reset	Input	PIN_AB12	3A	B3A_N0	3.3-V LVTTL	16mA ...ault)	
uart_in	Input	PIN_Y17	4A	B4A_N0	3.3-V LVTTL	16mA ...ault)	
uart_out	Output	PIN_Y18	4A	B4A_N0	3.3-V LVTTL	16mA ...ault)	1 (default)

Figure 11: pinout

Table of Figures

References

There are no sources in the current document.