Serial Cryptography

Serial communication with a twist

List of Authors: Robert Brosig, Trevor van Loosbroek

Electrical and Computer Engineering Department

School of Engineering and Computer Science

Oakland University, Rochester, MI

e-mails: rmbrosig2@oakland.edu, tvanloosbroek@oakland.edu

Abstract— The purpose of this project is to receive data from a computer onto an FPGA, mix that incoming data using a caesar cipher. The caesar shift value and direction are determined by the user of the board. The data is then transmitted to the computer from the FPGA. Originally the incoming data was going to be over RS232 but that was replaced with USB converted to TTL in order to preserve simplicity.

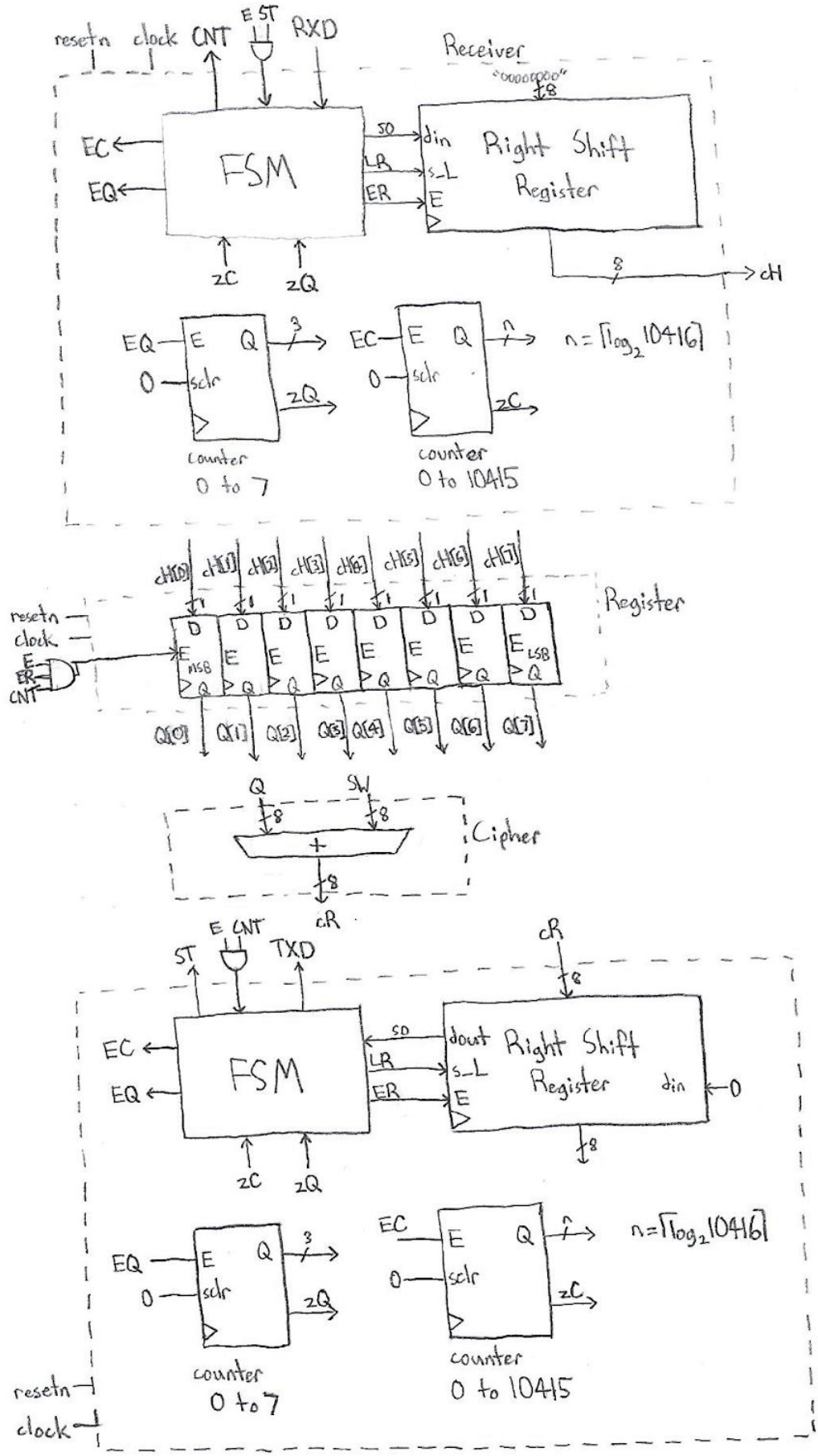
I. Introduction

This report will cover the design, the testing and the results of the project.

The real world application for this project extends far beyond the project itself. Communication between two devices has an unlimited list of applications, and a cryptographic device is simply one example. The in class discussion on transferring data over TXD and RXD proved to be useful in developing the project [1]. The group learnt how to communicate over COM Ports and find out which COM port is active for the FPGA. The receiving circuit is most similar to a state machine for a keypad, or a specific sequence of inputs. The start and stop bit are always given, and so the state machine may progress to the receiving state and return to the initial state once the start and stop bits are received, respectfully. Once finished our project could be used by secret agents trying to draft caesar ciphered messages.

II. Methodology

The project is a compilation of material from Dr. Llamocca [1]. Utilizing his code for a uart transmitter, we extrapolated the design for a uart receiver. This code, as stated in the previous section, is similar to the code for awaiting a sequence of characters. That state machine was therefore used as reference. From there we may use a simple array of D flip-flops to act as a register and hold the data until another character sends, and use an 8-bit adder with the switches of the board in order to create a simplistic cipher to input into the transmitter. This input replaces the switches of the original design we tested. The block diagram sketch of the entire circuit is on the following page.


resetn
clock
CNT
E ST
RXD
Receiver
"00000000"
8
FSM
EC
EQ
SD
LR
ER
din
s_L
E
Right Shift Register
zC
zQ
8
dH
EQ
E
Q
3
0
sclr
zQ
counter
0 to 7
EC
E
Q
n
0
sclr
zC
counter
0 to 10415
n = ⌈log2 10416⌉
dH[0]
dH[1]
dH[2]
dH[3]
dH[4]
dH[5]
dH[6]
dH[7]
Register
resetn
clock
E
ER
CNT
D
E
Q
MSB
LSB
Q[0]
Q[1]
Q[2]
Q[3]
Q[4]
Q[5]
Q[6]
Q[7]
Q
SW
8
+
Cipher
dR
E CNT
ST
TXD
dR
8
FSM
EC
EQ
SD
LR
ER
dout
s_L
E
Right Shift Register
din
0
zC
zQ
EQ
E
Q
3
0
sclr
zQ
counter
0 to 7
EC
E
Q
n
0
sclr
zC
counter
0 to 10415
n = ⌈log2 10416⌉
resetn
clock

III. Experimental Setup

This experiment is to be conducted in Windows 10, although it may work for previous versions as well. The program PuTTY, found here- https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html -is also a necessity for experimentation.

First, the FPGA is plugged into the computer turned on, and programmed. Next, we open the device manager in order to find the digital serial communication port the program is connected to. This may be found under the "Ports (COM & LPT)" tab. PuTTY is then opened- We set the type of communication to serial, and the COM port to the previously discovered port. The serial communication settings in PuTTY are then changed to communicate at 9600 baud, which should be the default of the program. After enabling the two rightmost switches on the FPGA, which are the circuit enable and the register enable, we may open the connection with PuTTY and type to communicate. The expected results are ASCii characters corresponding to the typed characters, modified by flipped switches on the board, received a very short delay after they are typed on the keyboard of the computer. We may also disable the register enable in order to lock the value last received, which it will then use to respond to any character sent until it is enabled again.

IV. Results

The project may be used to relay communication to and from a computer, while modifying the data sent through a simple caesar cipher, or by preventing the register from changing on the next transmission received. The test bench sends the characters sent to it in a bitwise fashion, back in the same fashion. Using the cipher to add one, adds one to the binary value before sending it to the output of the circuit. These results were expected based on the design of the circuit, and we are pleased that it has accomplished the intended design.

Conclusions

The main takeaways of the project is that knowledge of the intricacies of communication are impunitive in developing a program based on communication. There are few challenging concepts but many small bits of required knowledge before beginning. While the current design is not perfect, we believe it to be workable, and future designs could be implemented rather simply by keeping the receiver, transmitter, and register portions of the circuit, and adding other connections in-between. Examples of such possible improvements could be developing a more complex cipher, or adding other ways of modifying the data packet, such as division or multiplication.

References

[1] D. Llamocca, ECE 2700, "Digital Logic Design"