MP-2 Write-Up

Blake Vermeer
Kris Hall
Rohit Zambre

October 9, 2014

Date Due: October 10, 2014
Instructors: Joseph Zambreno

1 Objective

2 Platform Overview

In this section we were asked to review the project and provide a short description of the project structure. Here is our description:

The core part of this project is split into three main files: *v5_emac_v1_4_example_design.vhd*, *v5_emac_v1_4_locallink.vhd*, and *v5_emac_v1_4_block.vhd*. The *v5_emac_v1_4_block.vhd* is the lowest level element of the three. This wrapper file is responsible for interfacing with the physical Ethernet hardware on the FPGA. The *v5_emac_v1_4_locallink.vhd* is the next level up file. It is responsible for interfacing with the with the block level wrapper file and providing FIFOs for the input and output of the Ethernet module. The highest level of the design is *v5_emac_v1_4_example_design.vhd*. This module interfaces with the locallink wrapper file and provides a basic loopback interface.

3 Basic Scanning

The state machine diagram for the detection of "CO" is as follows: This diagram is correct so long as the string "CO" is not split up over multiple packets.

The hardware diagram for how the UDP checksum is being set to zero is shown below:

4 String Detection

Adding on to the previous section, the task for this section is to extend the finite state machine that can detect the string "CO" to be able to detect the string "CORN!" along with the creation of two extra finite state machines that detect the strings "ECE", and "GATAGA".

The extended finite state machine for detecting the string "CO", now extended to detect the string "CORN!", has the following state diagram:
A new finite state machine for the detection of the string "ECE" has the following state diagram:
Another new finite state machine for the detection of the string "GATAGA" has the following state diagram:

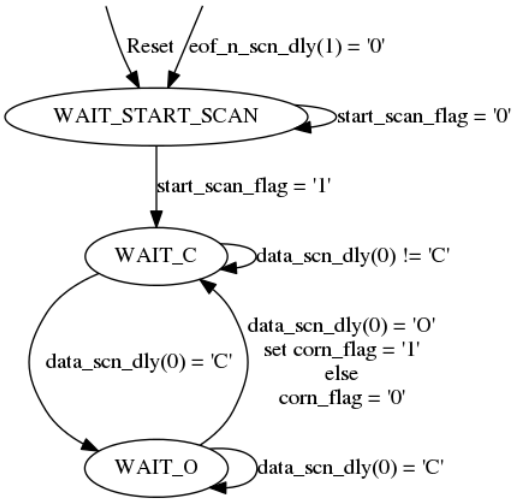

Figure 1: State machine diagram for the detection of "CO"

5 String Counting

In this section, the task given is that of creating counters to count how many times the strings CORN!, ECE, or GATAGA are detected in a given UDP packet. These counters are to be 8-bit counters, and the bits of the separate counters are to be tied to the following LEDs:

corn_counter[0] = LED0
corn_counter[1] = LED1
ece_counter[0] = LED2
ece_counter[1] = LED3
ece_counter[2] = LED4
gataga_counter[0] = LED5
gataga_counter[1] = LED6
gataga_counter[2] = LED7

In simulation, the counters work as expected, as can be seen in this screenshot of Modelsim.

6 Message Return

7 Bonus - Inner-Packet Counting

8 Conclusion

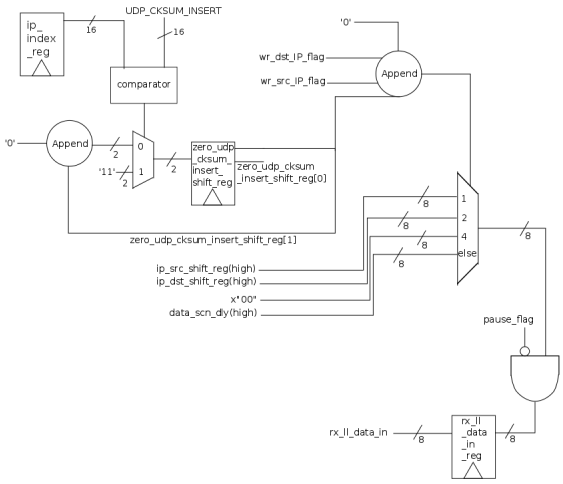

Figure 2: UDP checksum zeroing diagram

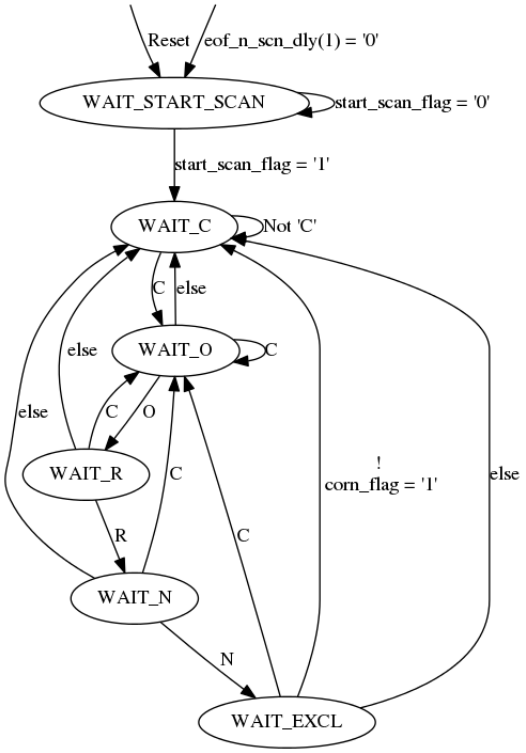

Figure 3: Extended "CO" FSM to detect "CORN!"

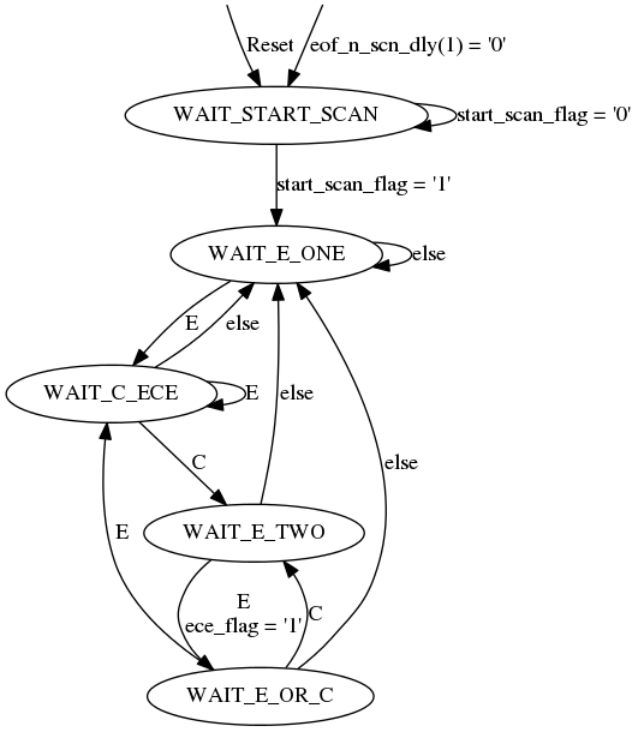

Figure 4: FSM to detect "ECE"

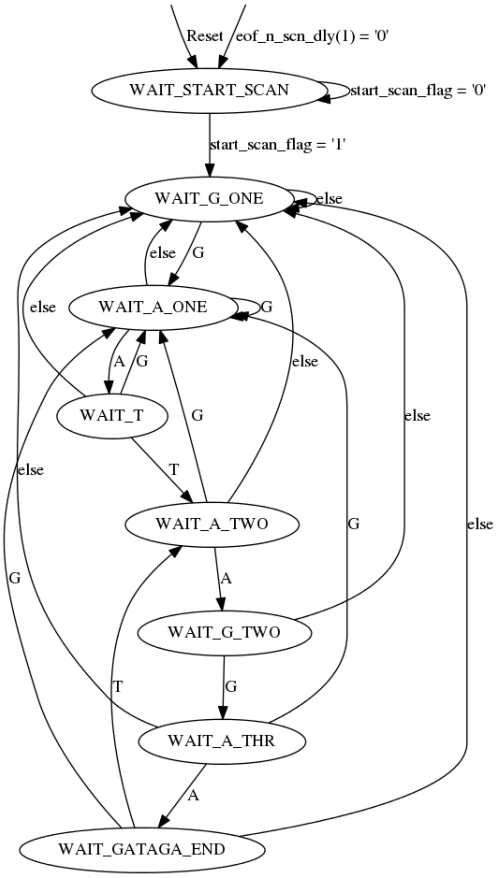

Figure 5: FSM to detect "GATAGA"

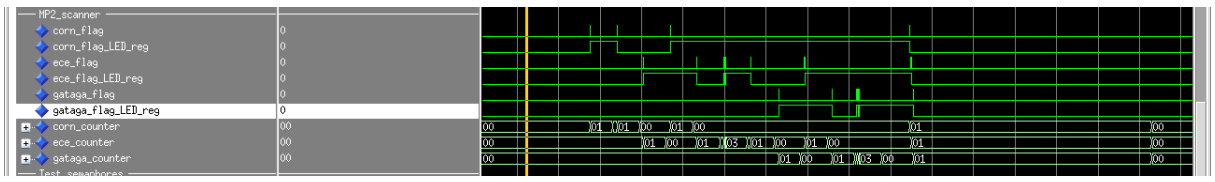

Figure 6: Relevant signals for the simulation of the counters