Lab 2 Report

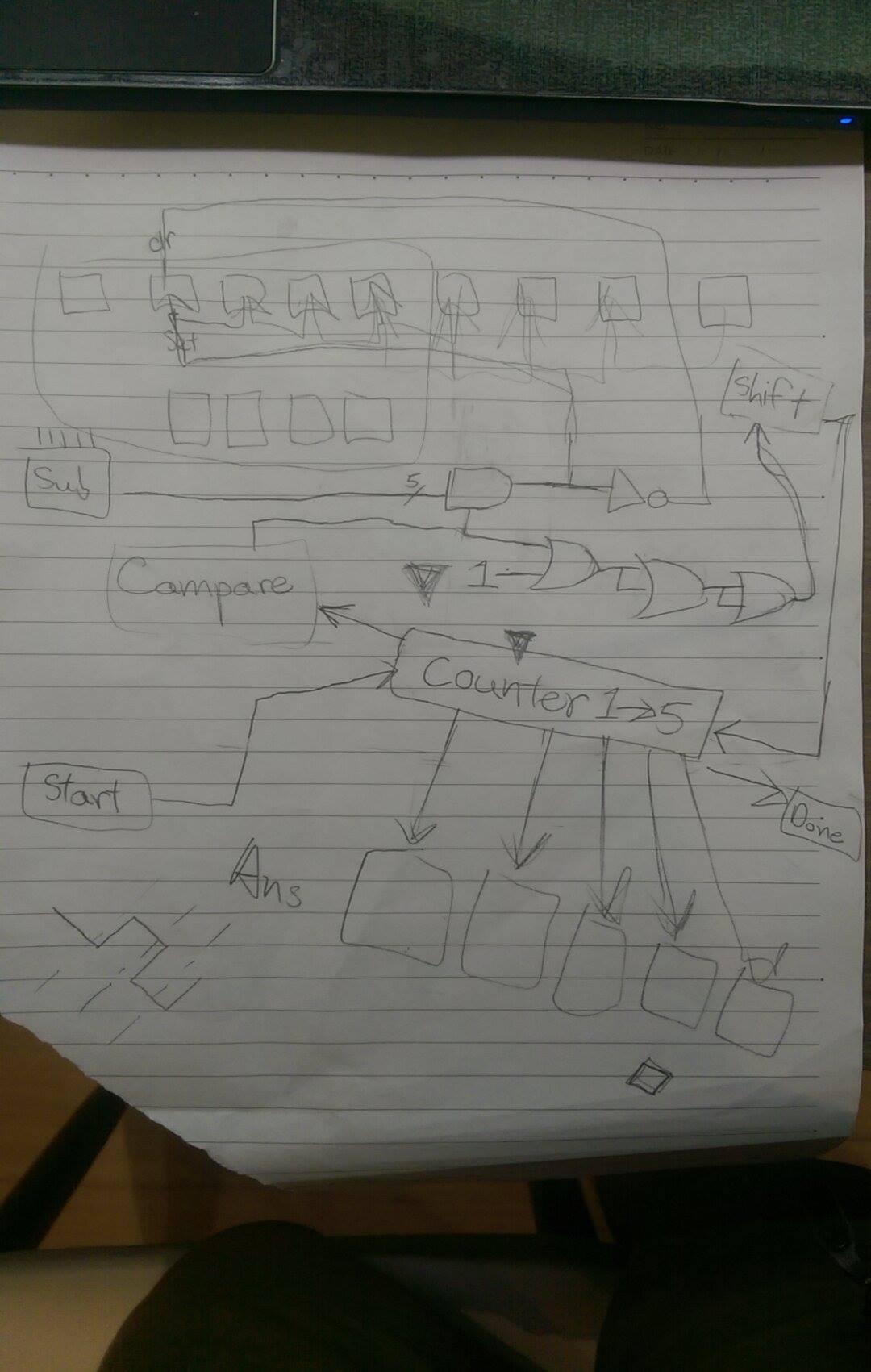
Topic: Divider

Group: 楊孝政 B03901178， 溫明浩 B03901179

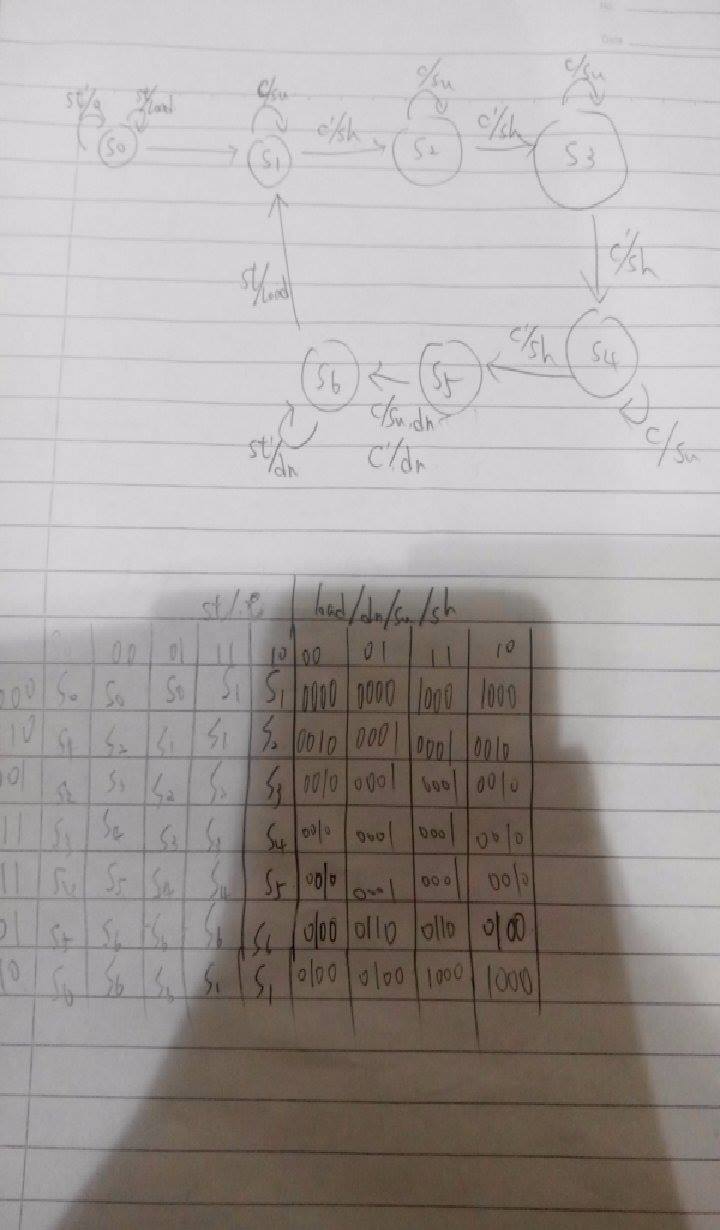
Achievements:

* Pass basic test: X
* Willing to present: X
* Special Design: X

To start of our project, we thought about the parts that we would need in order create a divider. Although I did study the design found in the text book, I decided to create my divider with my own design. The setup itself isn’t really too complicated. The main components of this divider include a comparator, subtractor, counter, shift, and several registers to store the value that we have for our divider. My design was slightly adjusted while we kept working on the actual components. The start input would first trigger the counter, but after that, the counter would be connected to the shift and allow the shift the registers containing the dividend. Before the shift in the dividend, the comparator would compare the first 5 bits of the dividend with the divisor, and if the dividend is greater than or equal to then the subtractor would input the subtracted values into the first 5 bits of the dividend. If the comparator yielded that the dividend was less than the divisor, then the values from the subtractor would not be inputted into the registers. At the same time, the value of the comparator would be inputted into another set of shift registers that would store the quotient. If the dividend was larger than the divisor, then a 1 would be stored into the registers. The shift function would be controlled by the counter, which means that both the shift registers of the dividend and the shift registers of the quotient would shift four times. After the counter counts the four shifts, then it would stops shifting as the quotients are in the registers, and the remainder should be in the first 2 to 5 bits of the dividend. The counter would also output the done output signaling that the divider was ready for the next problem.



For the most parts, the design work quite well, but the counter presented more of a problem than the rest of the components. In order to get a design for the counter, first we drew a state map and state graph. Then we had to draw k-maps for the next states and outputs. After putting together the counter, another difficult part was to put the counter into the divider. There was numerus problems with the counter not working with the register, some of which includes the registers not shifting the correct number of times because of the loading and subtracting values.



In the end, we had to adjust several things compared to the original design. We decided it was better not to use the ClrN and PreN functions of the D-flip flops and instead use just input the value into the registers via shifting. This made the control of when to shift the dividend register and when to shift the quotient register really difficult to maintain. Another small problem was that since we are inputting the value using the D input of the registers, the input depends on the clk, which causes some delay, and this delay sometimes causes false output in the quotient.

So I think this lab really let us understand that when designing the solution to a real life problem, all the things that we learned in class really helps us find a solution. Just like what we learned in class, the problems of delay is a really big problem while drawing state graphs and k-maps really help us be able to create a component(counter) that we needed. In the end, we could not get the divider to work completely and only correctly for test1. We have narrowed down our problem to the clk control for our quotient; we couldn’t successfully get the timing in when to shift our quotient down. We’ve tried several designs for it already but none of the clk controls for the quotient works completely correctly.