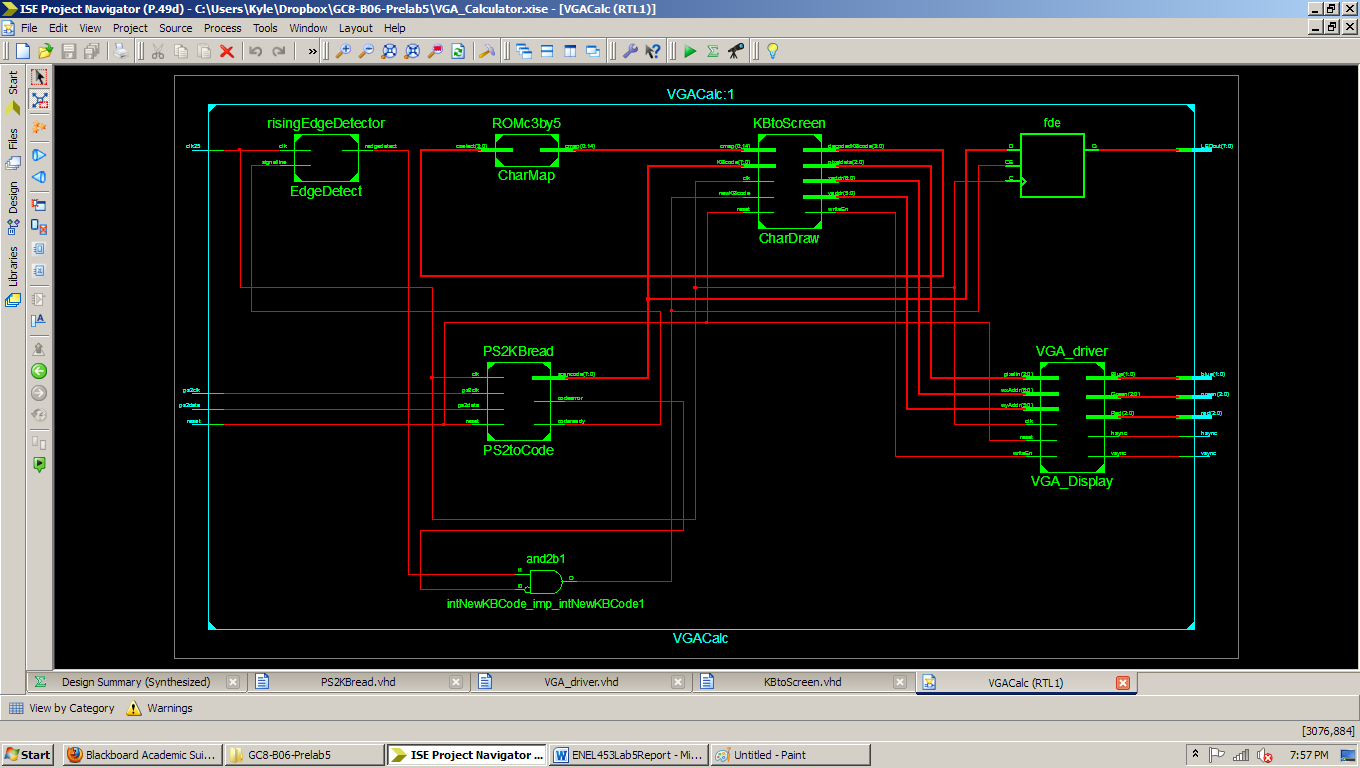
|  |
| --- |
| University of Calgary |
| ENEL 453 |
| Laboratory #5 – Simple VGA Calculator |
|  |
| **Kyle Derby MacInnis**  **Sajna Massey** |
| **4/2/2013** |

We declare that this laboratory report is entirely our own work and includes no material which has been copied from any other source excepting that material which is clearly identified as the work of others .

**Synthesis (w/ KB to LED Output Included):**

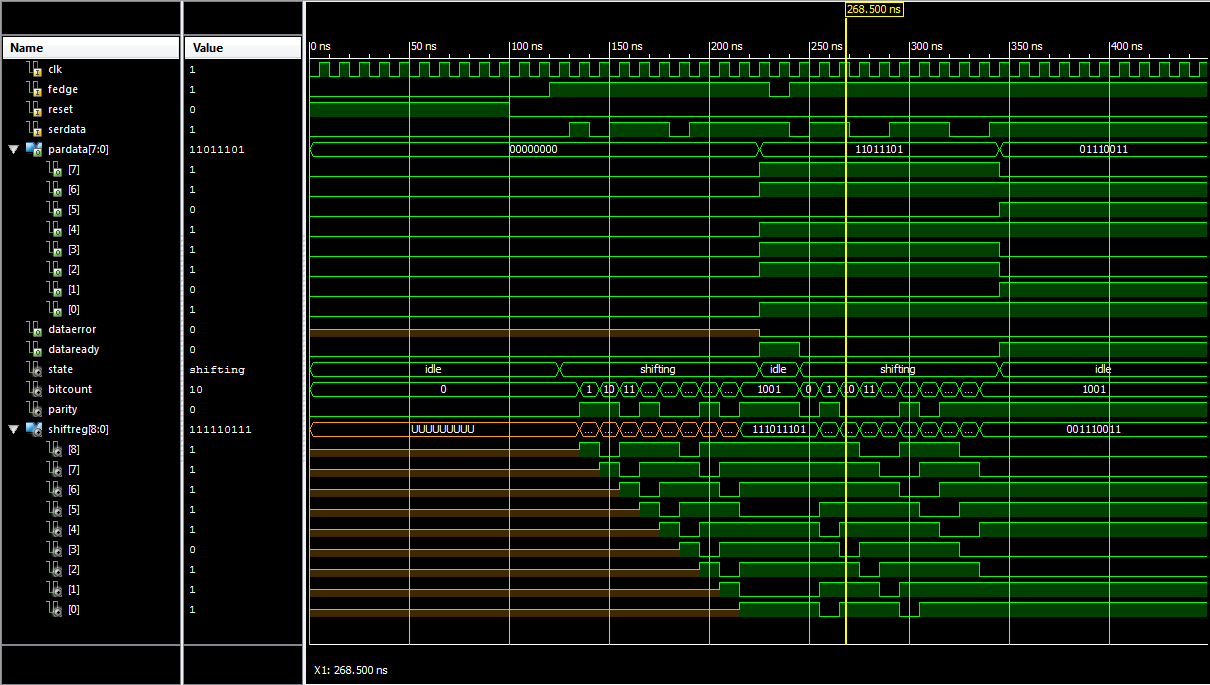
The following is the RTL diagram for the final design. This design includes outputs to the LED’s as being part of the design:



As can be seen, the PS/2 Data and Clock are fed into the reader which then outputs the parallel Scan Code to the Digit Drawer. This then writes to the Frame Buffer in the VGA driver. In addition the Scan Code is also fed directly to LED’s on the board via the use of a DFF controlled by the NewKBCode signal.

**Simulation of Test Bench:**

The following Waveform shows the expected results of the test bench. As can be seen, the serial data is indeed read into the register and then outputted in parallel. This occurs when the state changes from idle to shifting.

****

**Additional Questions:**

1 – The reason for two clock signals is because the PS/2 keyboard has its own internal debouncer inside. This clock signal is then used to ensure synchronization with both the keyboard, and with the VGA output.

2 – The screen remains showing the previous calculations as it had prior to being reset. The reason this happens is due to the DualPortRAM unit within the Frame Buffer. The reset does nothing to the RAM unit and as such, it maintains its memory which is the frame of the screen.

3 – The reason for the bidirectional ability of the keyboard is so that configurations can be sent to the keyboard controller. This would include things like remembering Num-lock, Cap-Lock, etc.

4 – There are three major types of warnings present:

1. – The warning below is referring to using only 4-bits of the 18-bit multipliers. This won’t cause any issues in our case, but it is a sorry use of resources:

*WARNING:Xst:643 - "C:/Users/Kyle/Dropbox/GC8-B06-Prelab5/KBtoScreen.vhd" line 165: The result of a 4x4-bit multiplication is partially used. Only the 4 least significant bits are used. If you are doing this on purpose, you may safely ignore this warning. Otherwise, make sure you are not losing information, leading to unexpected circuit behavior.*

1. – The next three warnings below are referring to unused connections. This again is no problem as we are not using these connections for anything in our design.

*WARNING:Xst:646 - Signal <vaddr<9>> is assigned but never used. This unconnected signal will be trimmed during the optimization process.*

*WARNING:Xst:646 - Signal <vaddr<2:0>> is assigned but never used. This unconnected signal will be trimmed during the optimization process.*

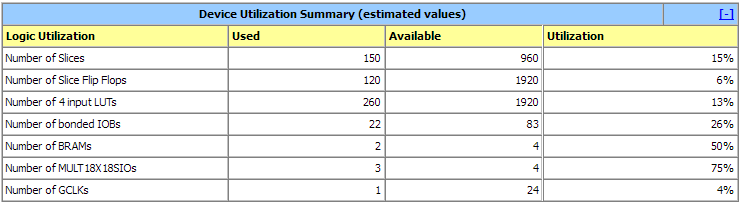
*WARNING:Xst:646 - Signal <haddr<2:0>> is assigned but never used. This unconnected signal will be trimmed during the optimization process.*

1. – The final three warnings below refer to the declaration of apply a constant to a variable during our KBtoScreen method. This is just saying that they will not be using connections as the connection will just be logic 0 anyways.

*WARNING:Xst:1293 - FF/Latch <ypos\_0> has a constant value of 0 in block <KBtoScreen>. This FF/Latch will be trimmed during the optimization process.*

*WARNING:Xst:1896 - Due to other FF/Latch trimming, FF/Latch <xpos\_0> has a constant value of 0 in block <KBtoScreen>. This FF/Latch will be trimmed during the optimization process.*

*WARNING:Xst:1896 - Due to other FF/Latch trimming, FF/Latch <xpos\_1> has a constant value of 0 in block <KBtoScreen>. This FF/Latch will be trimmed during the optimization process.*

5 – The Device Size Utilization for the implementation was:

The reason for the 18x18 bit multipliers is because we are using multiplication in our design. These have the values hard coded into them and all we are doing is reading the last 4 bits which could result in arithmetic errors if the value goes over 4-bits in size (ie. >15).

6 – The ranges are specified so that the circuit logic can be implemented in the FPGA. Since the FPGA uses various logic cells, it must know how many bits are required to perform calculations and as such the range for this amount must be known.

7 – Since this calculator will be used in a known radix, it is safe to assume you can do a simple division-comparator on individual positions which then could be used to output the characters to the screen. Possible states could include a state for outputting all the first number’s digits, and another state for putting the second number’s digits. These would do a check for operator otherwise loop back on themselves. Then the same thing would apply for the answer. It would use the calculated amount and loop through each position using division and comparators.

8 – The division operation could be implemented just as the multiplication was implemented. The only difference would be to do a check for 0 in the denominator, and if so either output an error, null, or zero depending on preference.