COL216 Computer Architecture

Lab Assignments 6 : CPU Design Synthesis and Testing on FPGA Board

This assignment involves synthesizing the ARM design done in the previous assignments and testing it on BASYS3 FPGA board. For this, some features need to be added to make testing on the board easier. These are as follows.

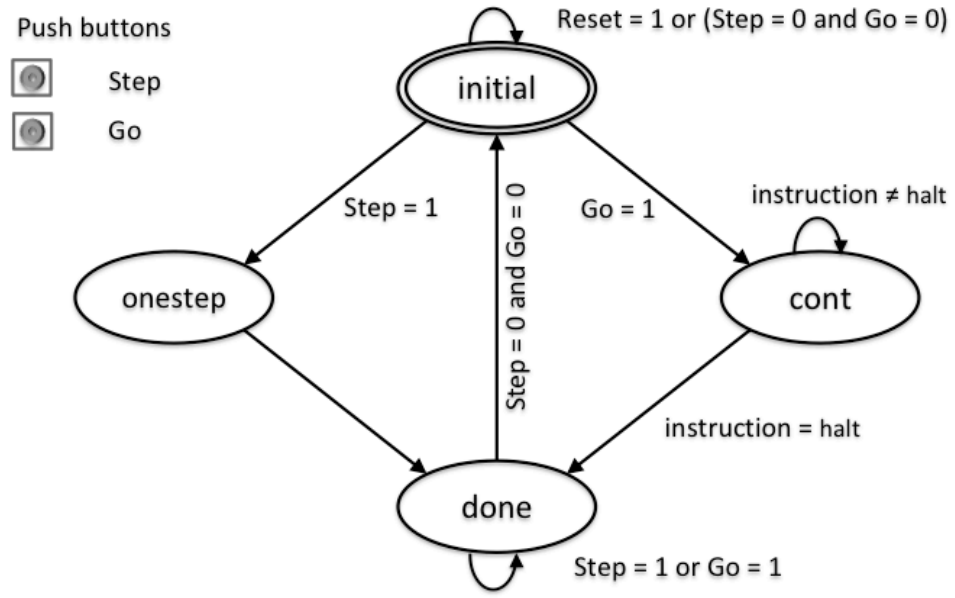
1. Connect signals to be observed to LED display through a display interface.
2. Add a small FSM to facilitate step-by-step operation.
3. Make provision for multiple test programs.

As described in Lab Assignment 5, the display interface is basically a multiplexer, with select input coming from slide switches. The other two features are described here.

FSM for step-by-step operation

The purpose of this FSM is to allow execution of one instruction at a time using a push button on the board so that the signal may be observed on LEDs after each instruction to see its effect. We also need to retain the option of running a program in continuous mode. Once started in continuous running mode, when does the execution stop? To answer this question for the purpose of this assignment, we will treat the instruction "ANDEQ r0, r0, r0", which has a code 0x00000000, as HALT instruction.

The state transition diagram shown below depicts the behavior of this FSM. A push button labeled "Step" is used to execute a single instruction and another push button labeled "Go" is used for continuous execution till a HALT instruction (as defined above) is reached.



After halting, execution can be resumed in single step or continuous mode. Thus, HALT instructions can be inserted in a test program like break points. The CPU design will now have one more clocked process to implement the FSM described here. The other clocked process (the one that updates the processor state) will now need to check whether the state of this FSM is "onestep" or "cont" before updating the processor state.

Provision for multiple test programs

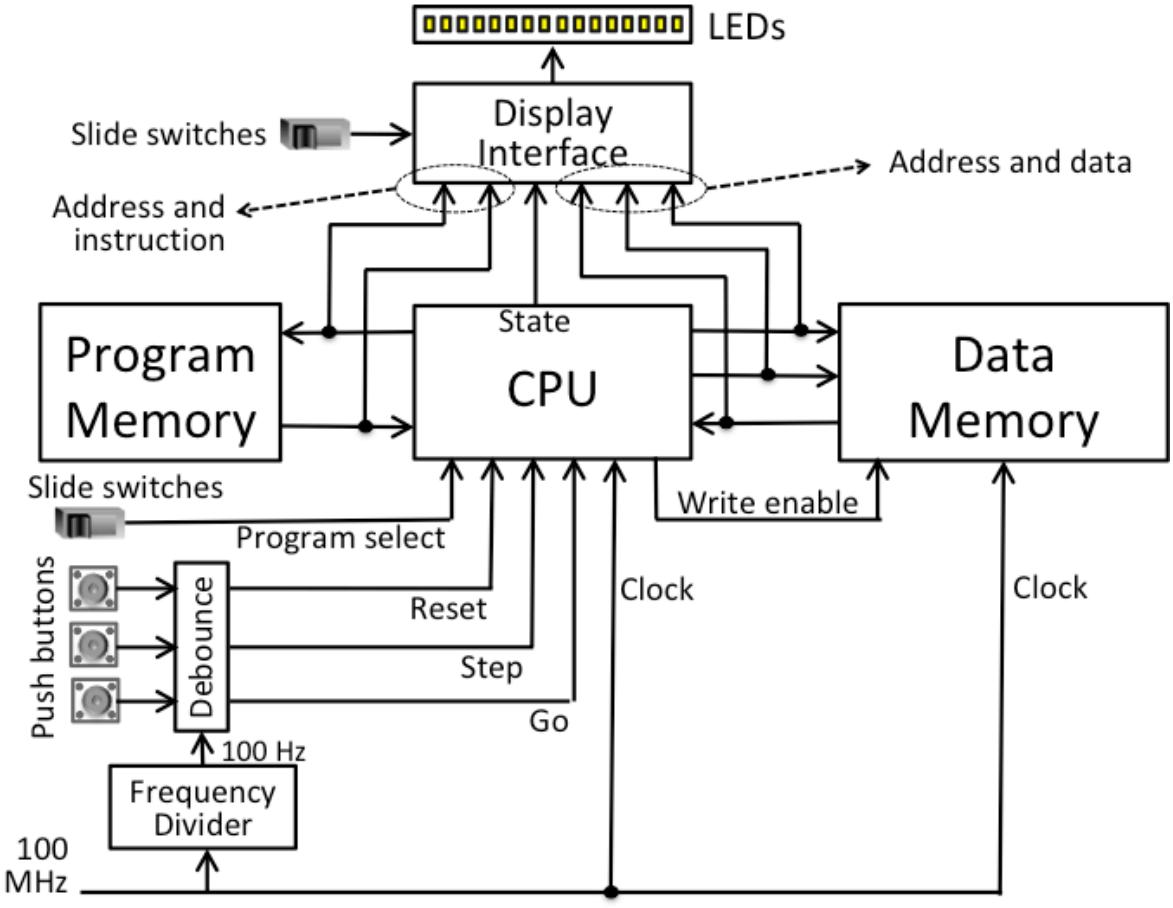
It is a useful feature to have multiple test programs stored in Program memory. With multiple programs already loaded, there is no need to repeated bit stream generation just in order to change the test program. All that is required to implement this is to control initialization of PC at the time of reset. Instead of loading 0x00000000, if we load the starting address of the required program, as specified by slide switches, the execution would begin at that address. To keep it simple, we can start all programs at addresses that are multiple of some power of 2, say 2^7. This would mean that a program memory of size 256 words or 1024 bytes can have 8 test programs (of maximum size 128 bytes or 32 instructions) beginning at addresses 0, 128, 256, 384, 512, 640, 768 and 896. These addresses will be of the following form in binary.

0000 0000 0000 0000 0000 00xx x000 0000

The bits shown as xxx are determined by three slide switches.

Overall System

The figure below shows the overall system with the three features described above. In case the design does not work at 100 MHz, reduce the clock frequency to 50 MHz. Remember to de-bounce all the push button switches.



Looking at synthesis results

Once the design is successfully tested and debugged, open the “Reports” tab. Look at the following reports.

1. “Utilization Report” under “Place Design”

Note down the table given under Slice Logic and include in your submission.

2. “Timing Summary Report” under “Route Design”

Note down Design Timing Summary and include in your submission.

Try to understand what these figures mean.