FAQ

1.	Can the memory.v file be edited to make the datapath width 32 bit?
	No, the memory file should remain the same. Design the processor such that it is compatible to this datapath width. For eg., storing 32 bit registers in the memory might take 2 clock cycles.
2.	Using testbench.v program in Synopsys Design Compiler gives an error. Why?
	The file testbench.v and memory.v are meant for simulations only. There are many unsynthesizable codes in them which could only be simulated. So use these files only in Modelsim.
3.	What are the instructions b.n and ble
	B.n : Unconditional branch to the memory address specified. : The qualifier . n means narrow and specifies that the assembler must select a 16-bit encoding for the instruction. Ble : Branch if less than or equal. Signed integer comparison
4.	Should the processor have modules like Hazard Detection Unit, Forwarding Unit etc.?
	This is the discretion of the designer. Keep in mind that you are designing a custom-general purpose processor with an architecture capable of executing the given test programs correctly. You can optimize the performance of the processor with respect to maximum operating frequency, area, power etc.
5.	After creating a library in Modelsim and creating a new verilog file, the file doesn't get added automatically to the specified library.
	The modules specified in the verilog file will get added to the specified library only after the successful compilation of the file.
6.	Compilation of memory and testbench file gives errors in modelsim. Why?
	The only error that could arise from these files is due to wrong selection of the type of file. Both these file should be specified as system-verilog not verilog. You can check the properties of these files in modelsim and if the type is mentioned as verilog, change it to system-verilog.
7.	Is stack already a part of memory?
	The memory file which is provided does not have a dedicated section allocated for stack. You can either assign a part of this memory for stack or have a separate stack memory block in your design.
8.	How is the address calculated for the instruction b.n?
	From the opcode the immediate field is left shifted to one bit and added to current instruction's PC + 4. Refer Section A8.6.16 in page no 356 of the Arm manual
9.	The address to fetch data in LDR instruction is not clear.
	The target address is calculated as follows: current instruction's PC + 4 + (imm<<2). Note: If the bit 1 of PC of current instruction is 1 (i.e, PC[1] = 0), then it is forced to zero and then added with 4 and (imm<<2)to find the target address. Refer LDR (literal) and LDR (immediate, Thumb) in pages 434 and 430 for more details