: hiding the complexity of the design.

1

The realises the data operations required by the FSM.

2

The issues signals to control the operation of the datapath.

3

A Reduced Instruction Set Computer (RISC) has three instruction types. What are they, and in brief what do they do? Hint: Stump is a RISC computer

4

You use bit to determine whether an instruction is Type 1 or 2.

5

Shifter operations are applied to the register in operations.

6

In Verilog, there are four options for a bit to be. 0, 1, X and Z.
In brief, what does each one mean?

7

What type of circuits do you associate the use of
i) blocking statements, and
ii) non-blocking statements
in Verilog code?

8

The datapath realises the data operations required by the FSM.

2

Abstraction: hiding the complexity of the design.

1

Register to Register: Instructions that perform operations on values inside of registers.
Load/Store: The only operations that will be on memory.
Branch: Operations that change the value of the PC depending on a certain condition.

4

The control issues signals to control the operation of the datapath.

3

Shifter operations are applied to the srcA register in type 1 operations.

6

You use bit 12 to determine whether an instruction is Type 1 or 2.

5

Blocking statements use =, and are associated with combinatorial logic and execute one at a time.
Non-blocking statements use <=, and are associated with sequential systems that use a clock and actions are executed concurrently.

8

1 and 0 are True and False respectively.
A value of X is unknown logic.
A value of Z would be high impedance.

7

Do you have to use the default case in Verilog? When would you use it?

9

*What is the difference between a Verilog **task** and a Verilog **function**?*

10

*Where would you locate a **task** or **function** in your Verilog code?*

11

Why does Stump's R0 exist? Give some examples illustrating its use.

12

What is a load/store architecture?

13

How can pipelining accelerate performance of a processing system?

14

A task can change any number of outputs whereas a function can update only one.

10

Not necessarily. It is useful to cover cases not listed in the case statement, trapping any invalid states you shouldn't be in.

9

Allows you to enable further operations to be implemented that are not part of the ISA.

- *MOV: ADD R3, R2, R0*
- *NOP: ADD R0, R0, R0*
- *CMP: SUBS R0, R3, R4*

12

It needs to be within the module but outside any always/initial blocks.

11

By introducing parallelism at a small cost. It allows more instruction throughput at a given clock rate than would traditionally be possible.

14

Data within registers will only be operated on, with results being written back to registers. LD/ST operations will be included for memory operations.

13