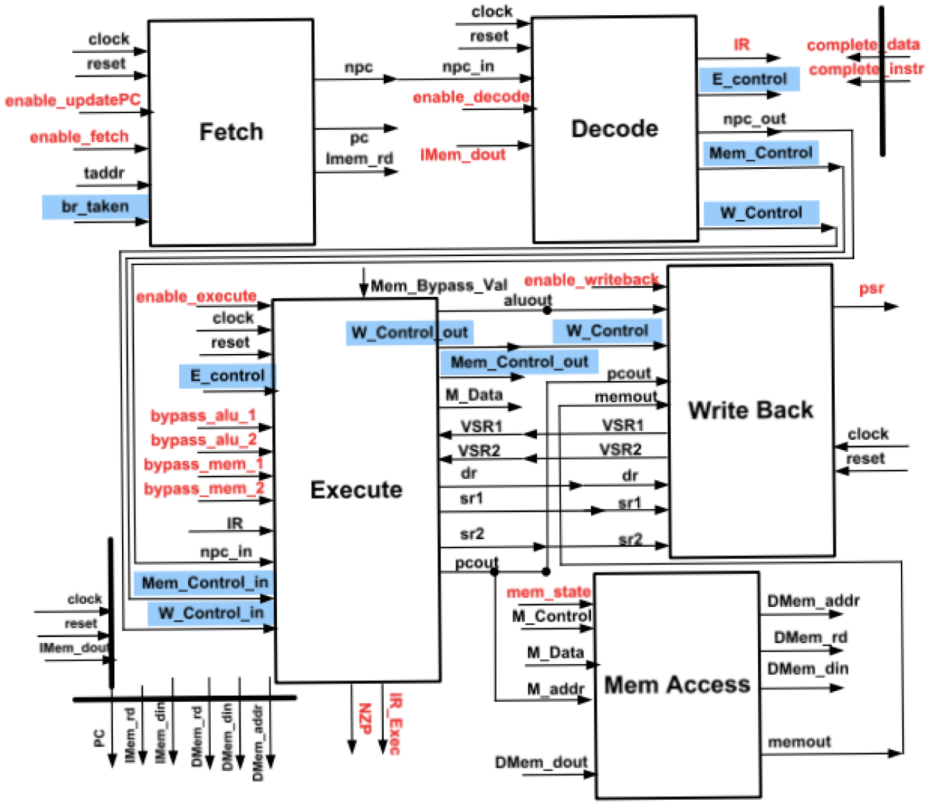
ECE 748

Advanced Verification with UVM

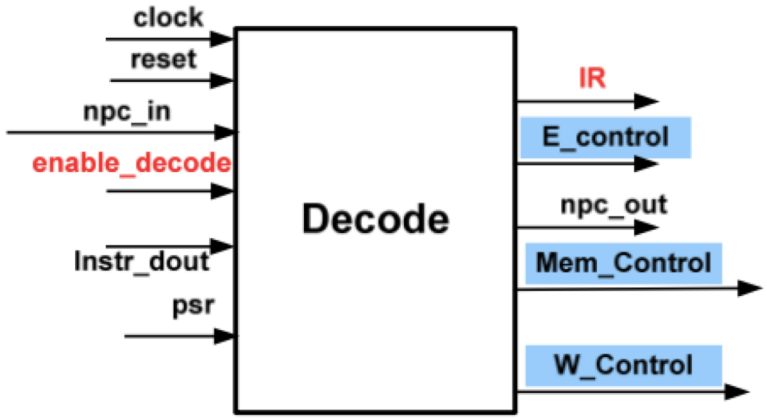
Class Projects – LC3

- Project 1
  - Create UVM interface package for decode input
- Project 2
  - Create a UVM environment and test bench for the decode block
  - Create a UVM interface package for decode output
- Project 3
  - Create a UVM environment and test bench for LC3 that includes the decode environment



Project 1 – Decode_in Interface

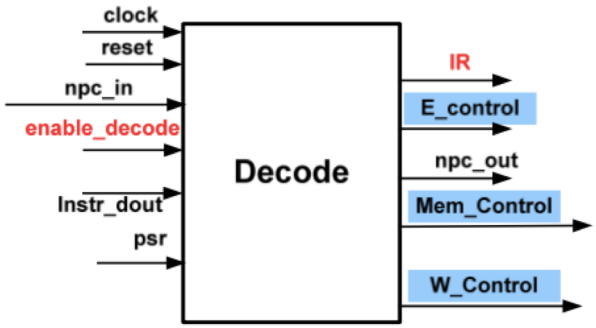
- Project 1
  - Create UVM interface package for decode input interface



Project 1 – Decode_in Interface

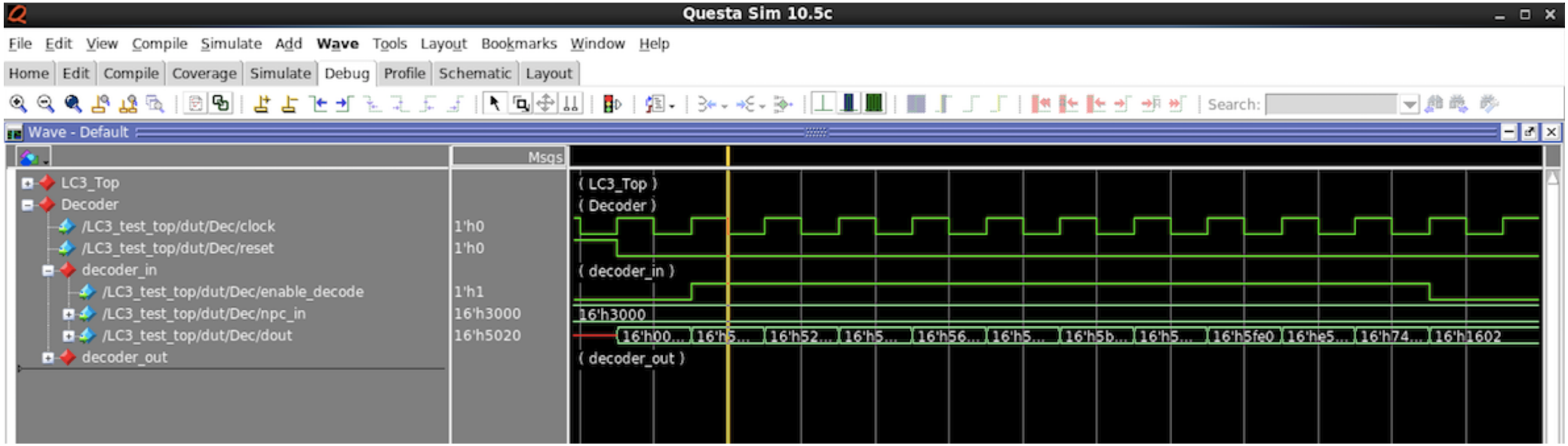
- From the LC3 Specification you will need to identify
  - Signals used
  - Protocol signaling
  - Transaction variables
  - Type-definitions
  - Parameters
  - Functional coverage

```
2  module Decode(/* System */ clock, reset,
3                /* decode_in */ enable_decode, dout, npc_in,
4                /* decode_out */ E_Control, Mem_Control, W_Control,IR, npc_out);
5
6      input            clock, reset, enable_decode;
7      input [15:0]     dout;
8      input [15:0]     npc_in;
9      output [1:0]     W_Control;
10     output           Mem_Control;
11     output [5:0]     E_Control;
12     output [15:0]    IR;
13     output [15:0]    npc_out;
```

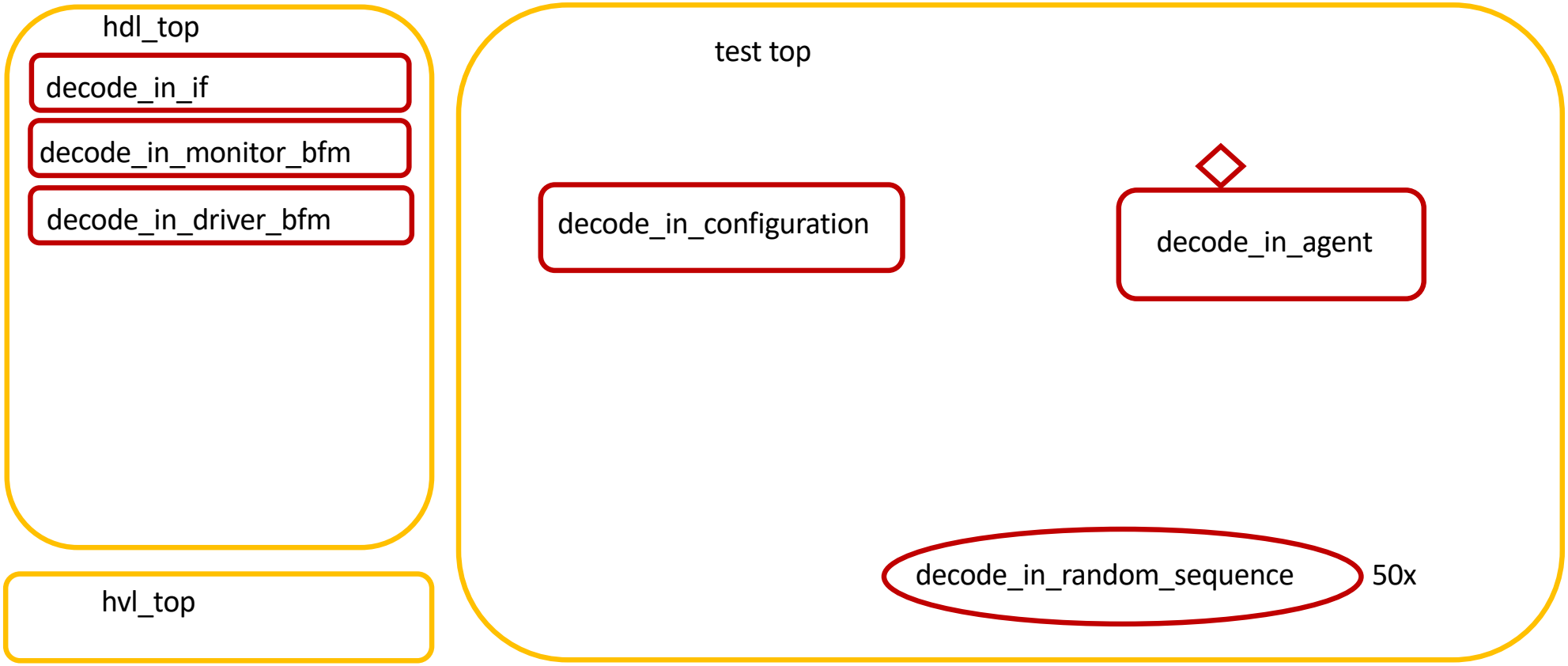


Instruction	15 12	11 9	8 6	5	4 3	2 0
ADD	0 0 0 1	DR	SR1	0	0 0	SR2
	0 0 0 1	DR	SR1	1	imm5	
AND	0 1 0 1	DR	SR1	0	0 0	SR2
	0 1 0 1	DR	SR1	1	imm5	
NOT	1 0 0 1	DR	SR1	1	1 1 1 1 1	1

Project 1 – Decode_in waveforms



Decode Test Bench



Project 1 – Directory Structure

- Interface package location/name
  - verification_ip/interface_packages/decode_in_pkg
- Test package location/name
  - project_benches/decode/tb/tests/decode_test_pkg
- Top level modules location/name
  - project_benches/decode/tb/testbench/hdl_top.sv and hvl_top.sv
- Decode RTL location
  - project_benches/decode/rtl
  - Decode RTL will be provided on Moodle
- Run simulation in
  - project_benches/decode/sim
- Run simulation command: make p1_debug
  - Create makefile using qrun command based on sequence_sequencer_driver example makefile

Project 1 – Decode_in Requirements

- Lecture material and examples to be used as basis for completing decode_in interface package
- Instantiate decode_in agent, agent configuration, and decode random sequence in uvm_test extension named test_top
- Import decode_test_pkg in hvl_top module
- Run random sequence 50 times to generate 50 random instructions

Project 1 – Decode_in Requirements cont'd

- Stimulus flow as shown in lecture material
- Emulatable stimulus and analysis flow as shown in lecture material
- Transaction viewing in the wave window
- Package structure as shown in ECE745 lectures
- Agent and BFM hierarchy as shown in lectures
- Agent configuration containing BFM handles
- Agent configuration retrieves BFM handles using uvm_config_db
- Agent retrieves its configuration using uvm_config_db
- Stimulus flow including sequence item, sequence base and random sequence
- Transaction coverage within agent
- Covergroup for coverage of ISA – Bits 15:12 only to show 50 valid instructions
  - At least one of each valid operation, bits 15:12, must be generated. Invalid operations shall not be generated.
- Convert2string method in transaction and configuration classes

Project 1 – Decode_in Submission

- Deposit all source in Moodle on due date
  - Parent directory name: <unityId>_p1
    - Contains project_benches sub-directory
    - Contains verification_ip sub-directory
  - Be sure to remove compiled libraries from sim directory
  - Compress the project into a single file for submission
    - File name: <unityId>_p1.zip