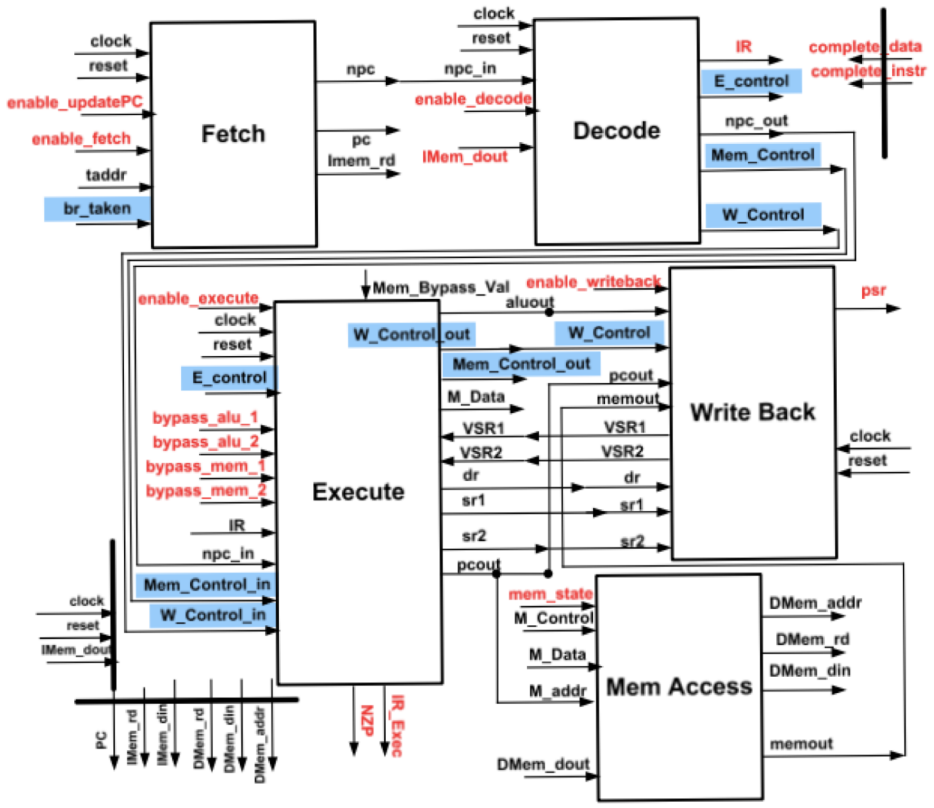
ECE 748

Advanced Verification with UVM

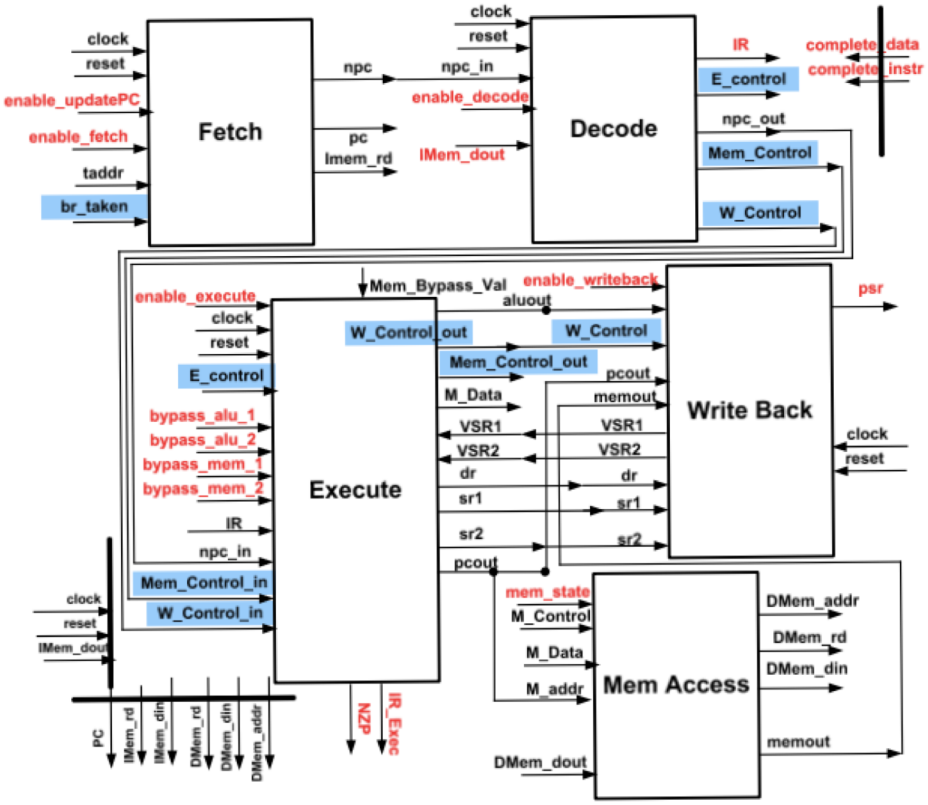
Class Projects – LC3

- Project 1
  - Create UVM interface package for decoder input
- Project 2
  - Create a UVM environment and test bench for the decoder block
- Project 3
  - Create a UVM environment and test bench for LC3 using UVMF

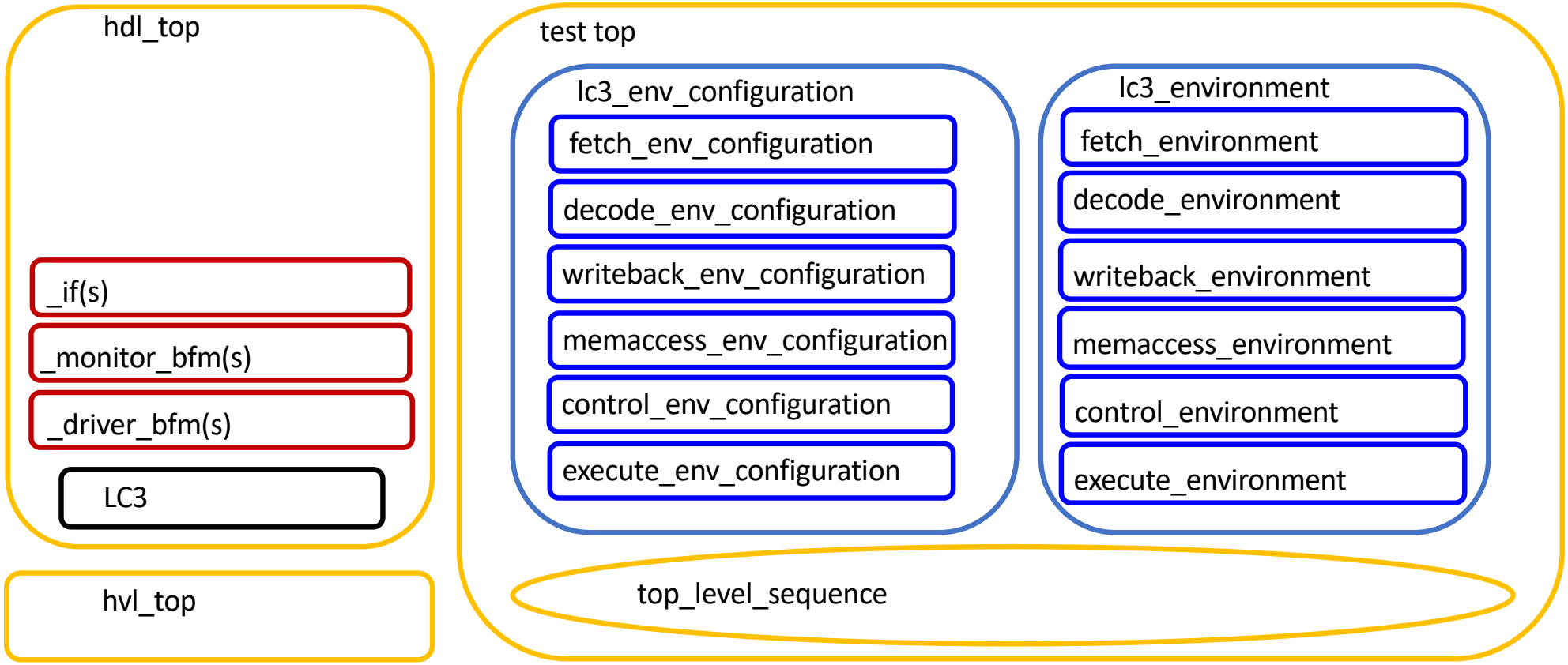


Project 3 – LC3 Environment

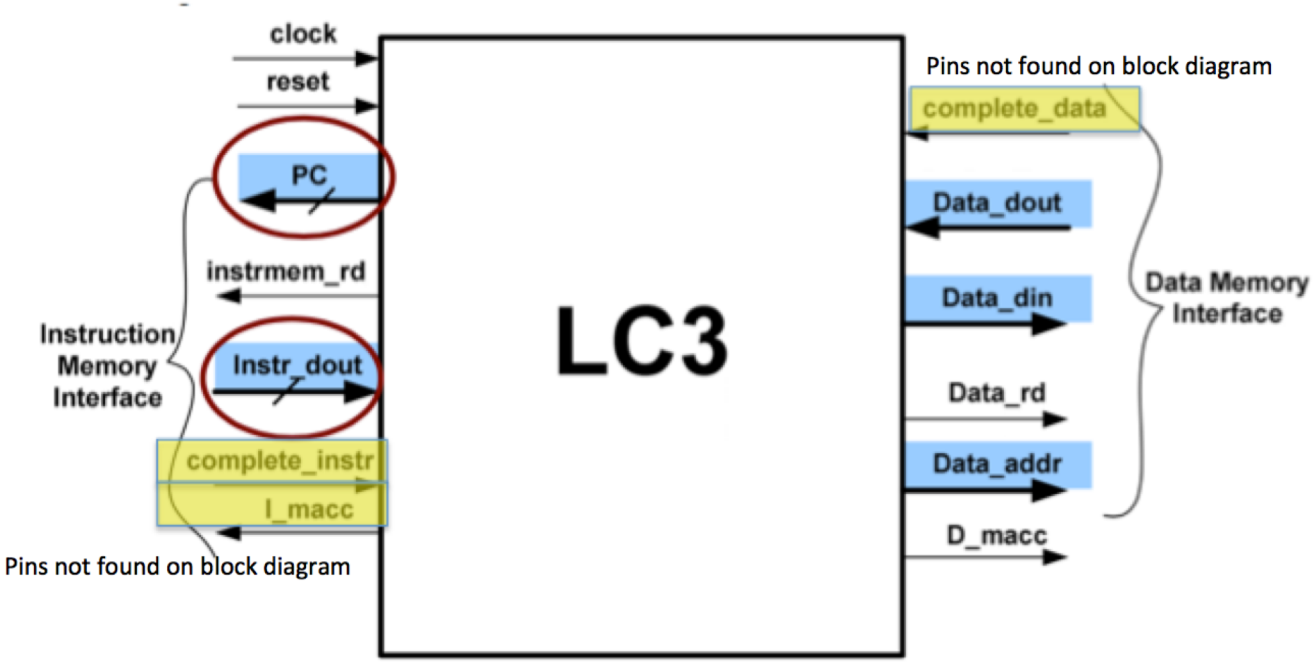
- Project 3
  - Create UVM environment packages and test bench for LC3 using UVMF generators
    - Include sub-environments within LC3 environment for the following modules
      - Fetch
      - Decode
      - Write Back
      - Mem Access
      - Control
      - Execute
    - Run tests in simulation
    - Find bugs in LC3 design



LC3 Test Bench



LC3 Top Level Diagram




```
Instruction Register = IR <= IMem_dout
```

Figure 1: Top Level Block Diagram of LC3

Project 3 – Directory Structure

- Environment package location/name
  - verification_ip/environment_packages/*envPackageName*
  - verification_ip/interface_packages/*interfacepackageName*
- Test package location/name
  - project_benches/lc3/tb/tests/lc3_test_pkg
- Top level sequence package location/name
  - Project_benches/lc3/tb/sequences/lc3_sequence_pkg
- LC3 RTL location
  - Project_benches/lc3/rtl
- Provided files
  - Common files provided on Moodle

Project 3 – Interface Instructions

- Use UVMF interface generator to create interface packages
  - Responder agents:
    - Instruction memory package: imem_pkg
    - Data memory package: dmem_pkg
  - Passive, monitoring only agents:
    - Fetch interface packages: fetch_in_pkg, fetch_out_pkg
    - Decode interface packages: decode_in_pkg, decode_out_pkg
    - Writeback interface packages: writeback_in_pkg, writeback_out_pkg
    - Memaccess interface packages: memaccess_in_pkg, memaccess_out_pkg
    - Control interface packages: control_in_pkg, control_out_pkg
    - Execute interface packages: execute_in_pkg, execute_out_pkg

Project 3 – Environment Instructions

- Use UVMF environment generator to create environment packages
  - Fetch environment: fetch_env_pkg
  - Decode environment: decode_env_pkg
  - Writeback environment: writeback_env_pkg
  - Memaccess environment: memaccess_env_pkg
  - Control environment: control_env_pkg
  - Execute environment: execute_env_pkg
  - LC3 environment: lc3_env_pkg
    - Includes all LC3 stage sub-environments
    - Includes instruction memory agent and data memory agent
- All environments include predictor and scoreboard
- Instruction coverage location options:
  - decode_in agent coverage component
  - Instruction memory agent coverage component

Project 3 – Bench Instructions

- Use UVMF bench generator to create LC3 test bench
  - Instantiates lc3 environment
  - Add LC3 rtl to LC3 bench
  - Use all .compile files as generated except for dut.compile
  - Use uvmf_bcr.py to run simulations
- Add tests to find bugs in LC3
  - Generated top level sequence instantiates and starts responder sequence on instruction and data memory agents
  - Tests further constrain instruction generation
    - Extend instruction memory transaction class to add constraints
    - Use factory type override to use instruction memory transaction class extensions

Project 3 – Bench Instructions

- Run tests in simulation
  - Collect instruction coverage in simulation
    - Create test plan showing instruction coverage groups, coverpoints, etc.
    - Merge test coverage with test plan
    - Generate coverage report

Project 3 – LC3 Requirements

- Point allocation based on table provided in Moodle
- Deposit all source in Moodle on due date
  - File name: group##_p3.zip
  - Be sure to remove compiled libraries from sim directory
- Provide simulation results documentation
  - Transcripts from simulation runs
  - List of test name and seed number combinations used to achieve full coverage in the sim/testlist.yaml file
    - The testlist will be used by TA to reproduce your coverage results
  - UCDB file containing merged coverage results
  - Coverage report, text or pdf, of coverage result