NORTH CAROLINA STATE UNIVERSITY
ECE 745 ASIC VERIFICATION

Fall 2018

BUG REPORT
LC3 MICROCONTROLLER VERIFICATION

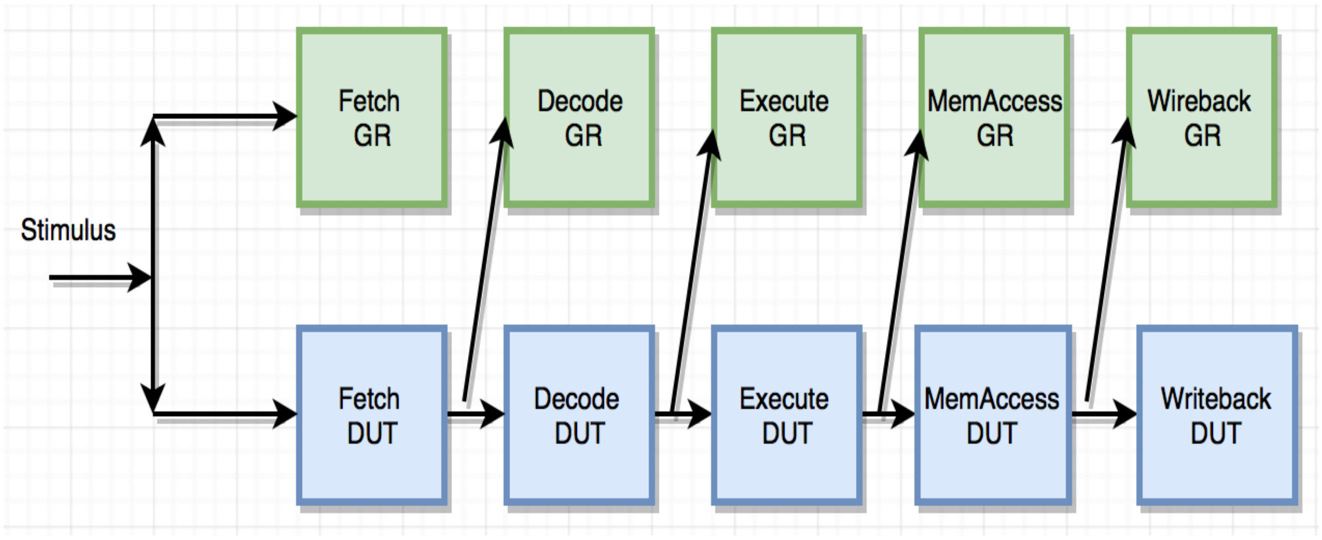
GROUP 9

Ryan Geary
Tushar Trehon
Harish Jamakhandi
Mehrnaz Sadeghian

Brief Introduction:
The purpose of verification is to determine whether a design meets its specifications.
In this project, five unknown bugs for an LC3 Microcontroller were provided and our testbench was able detect them all.

Methodology Adopted:
Within our software, we have created five golden reference models, one for each stage of the DUT in which each stage is modeled as a SystemVerilog class and is driven with the inputs from the DUT. At each stage, outputs of golden reference are checked against the DUT's outputs to find bugs that are contained within the stage. Once blocks are individually evaluated the system can be checked at higher level to observe bugs that occur due to interactions between the blocks.

Description: Golden reference follow the same data/control flow differently than DUT does, in other words, each golden reference is fed by the logically previous DUT's output.
Analysis: Bugs of the first few blocks will not be carried to the successors.



Bugs Identified

1.)

Bug1 file name: data_defs_bug1.vp
Bug #1 is identified in **Controller stage**.

Nature of BUG:

If a ALU instruction is followed by STORE instruction (ST, STR, STI), the bypass ALU_2 is opposite of the expected value. For example, if the correct output (golden reference) for bypass ALU_2 is 1, the actual output is 0.

Report Segment Example of Bug 1 (Transcript Captures):

Scenario 1: NOT instruction followed by STR instruction scenario

```
# 19865Curent PC:0035
# 19865000ps [INS_MEM][OPCODE: BR][NZP: 101][PCOFFSET: 81]
# 19865000--InstructionMemory--
# 19865000ps [CONTROLLER][OPCODE: STR][SR: R1][BASE_R: R7][PCOFFSET: 44]
# 19865000--Instruction_Decode--
# 19865000ps [CONTROLLER][OPCODE: NOT][DR: R4][SR1: R3]
# 19865000--Instruction_Execute--
# 19865000ps [CONTROLLER][OPCODE: IMMEDIATE AND][DR: R3][SR1: R1][Imm5 value: 25]
# 19865001 BR/JMP instr in Inst_dout
# 19865001[TB][CONTROL][ByPass_ALU_2]:[Value from DUT: 1][Value from GR: 0]
```

Scenario 2: AND instruction followed by STI instruction scenario:

```
# 19125Curent PC:397c
# 19125000ps [INS_MEM][OPCODE: AND][DR: R5][SR: R1][SR2: R7]
# 19125000--InstructionMemory--
# 19125000ps [CONTROLLER][OPCODE: STI][SR: R4][PCOFFSET: 181]
# 19125000--Instruction_Decode--
# 19125000ps [CONTROLLER][OPCODE: AND][DR: R4][SR: R5][SR2: R4]
# 19125000--Instruction_Execute--
# 19125000ps [CONTROLLER][OPCODE: NOT][DR: R6][SR1: R6]
# 19125001---InterfaceValues--
# 19125001[TB][CONTROL][ByPass_ALU_2]:[Value from DUT: 0][Value from GR: 1]
```

Scenario 3: AND instruction followed by ST instruction:

```
# 15065Curent PC:ffd2
# 15065000ps [INS_MEM][OPCODE: NOT][DR: R3][SR1: R2]
# 15065000--InstructionMemory--
# 15065000ps [CONTROLLER][OPCODE: ST][SR: R6][PCOFFSET: 56]
# 15065000--Instruction_Decode--
# 15065000ps [CONTROLLER][OPCODE: AND][DR: R0][SR: R4][SR2: R4]
# 15065000--Instruction_Execute--
# 15065000ps [CONTROLLER][OPCODE: ADD][DR: R0][SR: R0][SR2: R3]
# 15065001---InterfaceValues--
# 15065001[TB][CONTROL][ByPass_ALU_2]:[Value from DUT: 1][Value from GR: 0]
```

2.)

Bug2 File: data_defs_bug2.vp
Bug #2 is in the **Writeback stage**.

Nature of the BUG:

If the sr1 value is greater than or equal to 4 (sr1= 3'b100, sr1= 3'b101, sr1= 3'b110, sr1= 3'b111), then VSR1 is taking the value of RF[sr1-4] (RF[DUT]=RF[sr1-4]). For example, in scenario 1 if the correct output (golden reference) for VSR1 is RF [**4**] = 0001, the actual output is RF [**0**] = 0000.

Report Segment Example of Bug 2 (Transcript Captures):

Scenario 1: VSR1 is taking the value of RF [0] instead of RF [4]

```
# 15975001[TB][WB][VSR1]:[Value from
DUT: 0000][Value from GR: 0001]
========================================
# Writeback Golden Reference:
# RF[0]: 0000
# RF[4]: 0001
# VSR1: 0001
# VSR2: ffae
# DUT Writeback Outputs:
====================
# VSR1: 0000
# VSR2: ffae
# sr1: 100
# sr2: 111
```

Scenario 2: VSR1 is taking the value of RF [1] instead of RF [5]

```
# 16005001[TB][WB][VSR1]:[Value from
DUT: 2b06][Value from GR: 2b07]
========================================
# Writeback Golden Reference:
# RF[1]: 2b06
# RF[5]: 2b07
# VSR1: 2b07
# VSR2: 0000
# DUT Writeback Outputs:
====================
# VSR1: 2b06
# VSR2: 0000
# sr1: 101
# sr2: 000
```

Scenario 3: VSR1 is taking the value of RF [2] instead of RF [6]

```
#1595001[TB][WB][VSR1]:[Value from DUT:
0000][Value from GR: 0008]
========================================
# Writeback Golden Reference:
# RF[2]: 0000
# RF[6]: 0008
# VSR1: 0008
# VSR2: 0008
# DUT Writeback Outputs:
===========================

# VSR1: 0000
# VSR2: 0008
# sr1: 110
# sr2: 000
```

Scenario 4: VSR1 is taking the value of RF [3] instead of RF [7]

```
# 16315001[TB][WB][VSR1]:[Value from
DUT: 0001][Value from GR: 0010]
========================================
# Writeback Golden Reference:
# RF[3]: 0001
# RF[7]: 0010
# VSR1: 0010
# VSR2: 0010
# DUT Writeback Outputs:
=======================
# VSR1: 0001
# VSR2: 0012
# sr1: 111
# sr2: 000
```

3.)

Bug3 File: data_defs_bug3.vp
Bug #3 is in the **Memory Access stage**.

Nature of the BUG:

When any path to **mem state 0** is taken through a Load Instruction (LD, LDR, LDI), the DMem_rd is low, while it should have been high. For example, in scenario 1 for going to mem_state 0 throught the LDI instruction, if the correct output (golden reference) for DMem_rd is 1, the actual output for DMem_rd is 0.

Report Segment Example of Bug 3 (Transcript Captures):

Scenario 1: Going to the mem_state 0 through the LDI instruction

```
# 1905000--Instruction_Decode--
# 1905000ps [CONTROLLER][OPCODE: LDI][DR: R7][PCOFFSET: 283]
# 1915000--Instruction_Execute--
# 1915000ps [CONTROLLER][OPCODE: LDI][DR: R7][PCOFFSET: 283]
# 1925000[TB][MA][DMem_rd]:[Value from DUT: 0][Value from GR: 1]
# =======================================================
# Mem_Access Golden Reference:
# DMem_rd: 1
# memout: 08da
#
# DUT Mem Access Outputs:
# mem_state: 0
# M_Control: 1
# DMem_rd: 0
```

Scenario 2: Going to the mem_state 0 through the LDR instruction

```
# 1835000--Instruction_Decode--
# 1835000ps [CONTROLLER][OPCODE: LDR][DR: R2][BASE_R: R5][PCOFFSET: 21]
# 1845000[TB][MA][DMem_rd]:[Value from DUT: 0][Value from GR: 1]
# =====================================================
# Mem_Access Golden Reference:
# DMem_rd: 1
# memout: 30c7
#
# DUT Mem Access Outputs:
# mem_state: 0
# M_Control: 0
# DMem_rd: 0
```

4.)

Bug4 File: data_defs_bug4.vp
Bug #4 is in **Execute stage**.

Nature of the BUG:

For all instructions, apart from ALU instructions when E_Control [3:1] = 3'b011 (BR, LD, LDI, LEA, ST, STI) pcout and aluout are 1 greater than the golden reference. For example, in scenario1 when the instruction in execute stage is BR (E_Control [3:1] = 3'b011) if the correct output (golden reference) for pcout and aluout is 1, the actual output for pcout and aluout are 0.

When E_Control is high pcout value can be determined by:
pcout = npc - 16'b1 + offset of 9
= 0006 - 16'b1 + 00a3
= 00a8
In the scenario shown below we see the pcout_DUT is 00a9 which greater the golder reference pcout by 1.

Report Segment Example of Bug 4:

Scenario 1: BR in execute stage (For BR instruction: E_Control [3:1] = 3'b011)

```
#        7035--Instruction_Execute--
#        7035ps [CONTROLLER][OPCODE: BR][NZP: 101][PCOFFSET: 163]
#        7035000[TB] [EXECUTE] BUG IN EXE STAGE DUT aluout_DUT = 00a9 | aluout = 00a8
#        7035000[TB] [EXECUTE] BUG IN EXECUTE DUT pcout_DUT = 00a9 | pcout = 00a8
#               7035000========EXECUTE ERROR=========
#               7035000==========Inputs==============
#               7035000  npc = 0006
#               7035000==========DUT Outputs===========
#               7035000  aluout = 00a9
#               7035000  pcout = 00a9
#               7035000===========GR OUTPUT===========
#               7035000  aluout = 00a8
#               7035000  pcout = 00a8
```

Scenario 2: ST in execute stage (For ST instruction: E_Control [3:1] = 3'b011)

```
#        5775--Instruction_Execute--
#        5775ps [CONTROLLER][OPCODE: ST][SR: R6][PCOFFSET: 428]
#        5775000[TB] [EXECUTE] BUG IN EXE STAGE DUT aluout_DUT = ffc9 | aluout = ffc8
#        5775000[TB] [EXECUTE] BUG IN EXECUTE DUT pcout_DUT = ffc9 | pcout = ffc8
#               5775000========EXECUTE ERROR=========
#               5775000==========Inputs==============
#               5775000  npc = 001e
#               5775000==========DUT Outputs===========
#               5775000  aluout = ffc9
#               5775000  pcout = ffc9
#               5775000===========GR OUTPUT===========
#               5775000  aluout = ffc8
#               5775000  pcout = ffc8
```

Scenario 3: LEA in execute stage (For LEA instruction: E_Control [3:1] = 3'b011)

```
#         5865--Instruction_Execute--
#         5865ps [CONTROLLER][OPCODE: LEA][DR: R6][PCOFFSET: 351]
#         5865000[TB] [EXECUTE] BUG IN EXE STAGE DUT aluout_DUT = ff63 | aluout = ff62
#         5865000[TB] [EXECUTE] BUG IN EXECUTE DUT pcout_DUT = ff63 | pcout = ff62
#                5865000========EXECUTE ERROR=========
#                5865000==========Inputs==============
#                5865000  npc = 0005
#                5865000==========DUT Outputs===========
#                5865000  aluout = ff63
#                5865000  pcout = ff63
#                5865000===========GR OUTPUT===========
#                5865000  aluout = ff62
#                5865000  pcout = ff62
```

5.)

Bug5 file: data_defs_bug5.vp
Bug#5 is in the **Decode stage**.

Nature of the BUG:

The bug in the decode stage propagates through to the execution and the controller stage as well. IR and npc_out are changing asynchronously while it's expecting to change synchronously. For example, in scenario 1 the instruction that's going to decode stage is BR, so we expected that npc_out and IR are changing on the next clock cycle but it's changing in the same clock cycle.

Report Segment Example of Bug 5:

```
# 335--Instruction_Decode--
# 335ps [CONTROLLER][OPCODE: BR][NZP: 100][PCOFFSET: 404]
# 335--Instruction_Execute--
# 335ps [CONTROLLER][OPCODE: STI][SR: R4][PCOFFSET: 380]
# 335000[TB][DECODE][NPC_OUT]:[Value from DUT: 2ff1][Value from GR: 2ff0]
# 335000[TB][DECODE][IR]:[Value from DUT: 0000100110010100][Value from GR:
1011100101111100]
# ========================================================
# Decode Golden Reference:
# IR: 1011100101111100
# NPC_OUT: 2ff0
# ========================================================
# DUT Decode Outputs:
# npc_in: 2ff1
# enable_decode: 0
# Instr_dout: 0000100110010100
# IR: 0000100110010100
# 36845--Instruction_Decode--
# 36845ps [CONTROLLER][OPCODE: BR][NZP: 001][PCOFFSET: 197]
# 36845--Instruction_Execute--
# 36845ps [CONTROLLER][OPCODE: IMMEDIATE ADD][DR: R0][SR1: R6][Imm5 value: 16]
# 36845000[TB][DECODE][NPC_OUT]:[Value from DUT: fd34][Value from GR: fd33]
# 36845000[TB][DECODE][IR]:[Value from DUT: 0000001011000101][Value from GR:
0001000110110000]
# ========================================================
# Decode Golden Reference:
# IR: 0001000110110000
# E_Control: 00
# NPC_OUT: fd33
# MEM_Control: 0
# W_Control: 0
# ========================================================
# DUT Decode Outputs:
# npc_in: fd34
# enable_decode: 1
# Instr_dout: 0000001011000101
# IR: 0000001011000101
```