VERIFICATION PLAN PROPOSAL
ECE 510
FINAL PROJECT

Rohit Kulkarni Anuja Vaidya

Contents

Introduction: 2

Design components: 3

IFD Unit 3

Description: 3

Execution Unit 4

Description: 4

Memory 5

Description: 5

Clock Gen 5

Description: 5

Unit Level Testbenches 6

Unit Level Testbench for IFD unit 6

Stimulus: 7

Checks to perform: 7

Coverage: 7

Unit Level Testbench for EXEC unit 8

Stimulus: 8

Checks to perform: 9

Coverage: 10

Chip level testbench 11

Stimulus: 11

Checks to perform: 12

Coverage: 12

Figure 1: FSM decode unit 3

Figure 2:Fsm: Execution unit 4

Figure 3: Unit Level test Bench: IFD unit 6

Figure 4: UNIT level test bench: Execution unit 8

Figure 5: FULL chip test bench 11

Introduction:

Four principal modules in design:

- Clkgen_driver
- Instr_decode
- Instr_exec
- Memory_pdp

Goal: Build a verification environment around the design

Verification Strategy:

- Unit Level Testbenches for
  - IFD Unit
  - EXEC Unit
- Full chip validation for entire design

Design components

IFD Unit

Description:

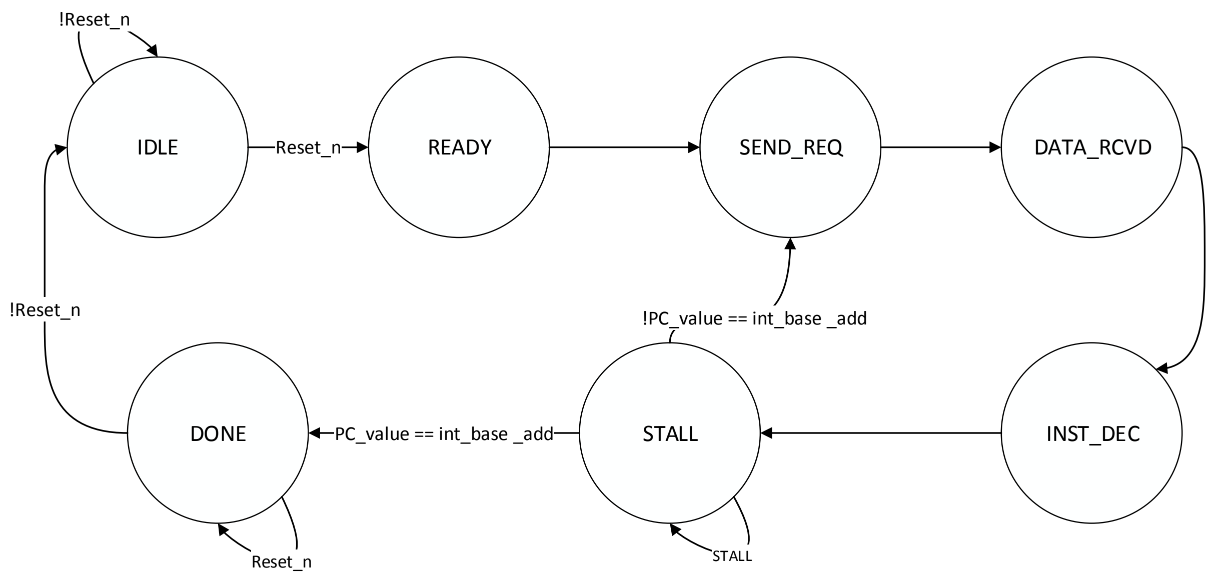


FIGURE 1: FSM DECODE UNIT

- Fetches instruction
- Decodes fetched instruction
- Supports decoding of
  - 6 memory reference instructions
    - Opcode 0 to 5
  - 22 microinstructions
    - Opcode 7 : Group 1 and Group 2

Execution Unit

Description:

- Executes instruction decoded by IFD unit.
- Accesses memory if instruction among opcodes 0 to 5.
- Reads operand from memory
- Writes result into memory
- Stalls IFD unit until current instruction completes execution

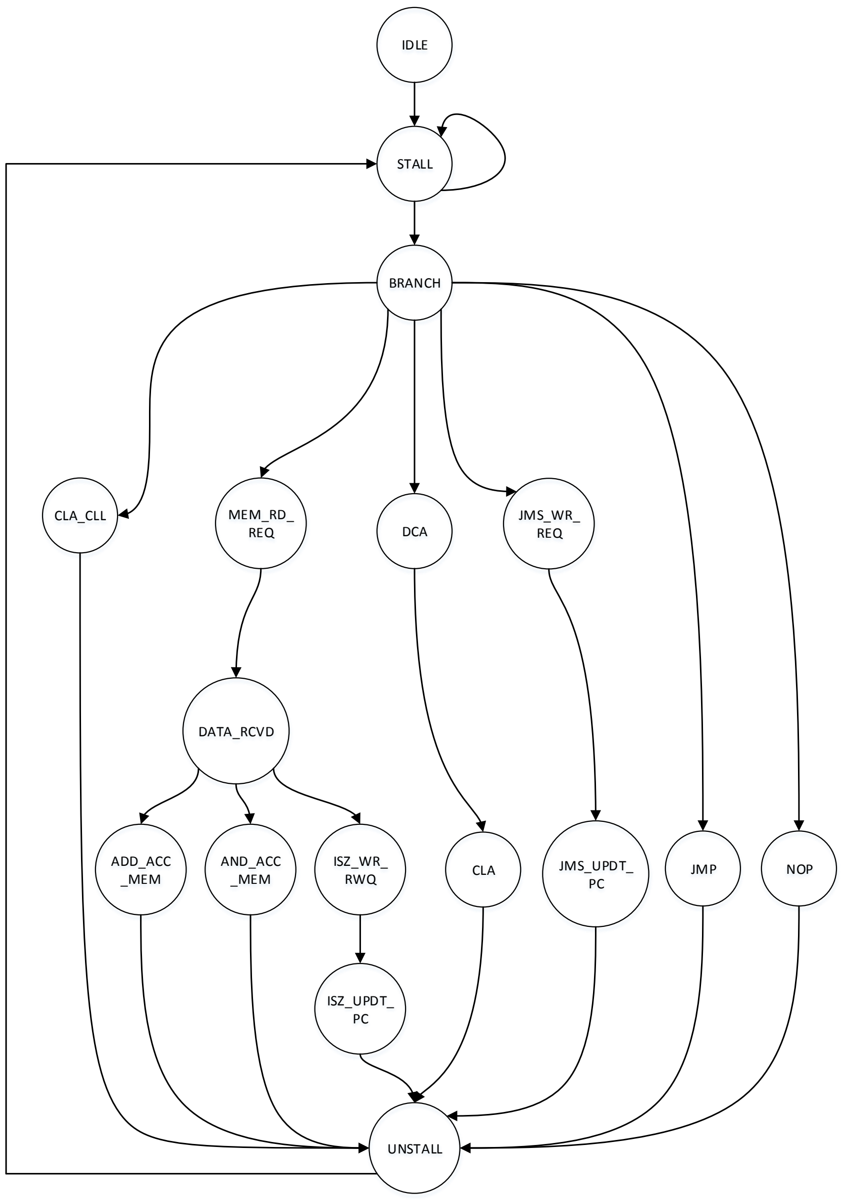
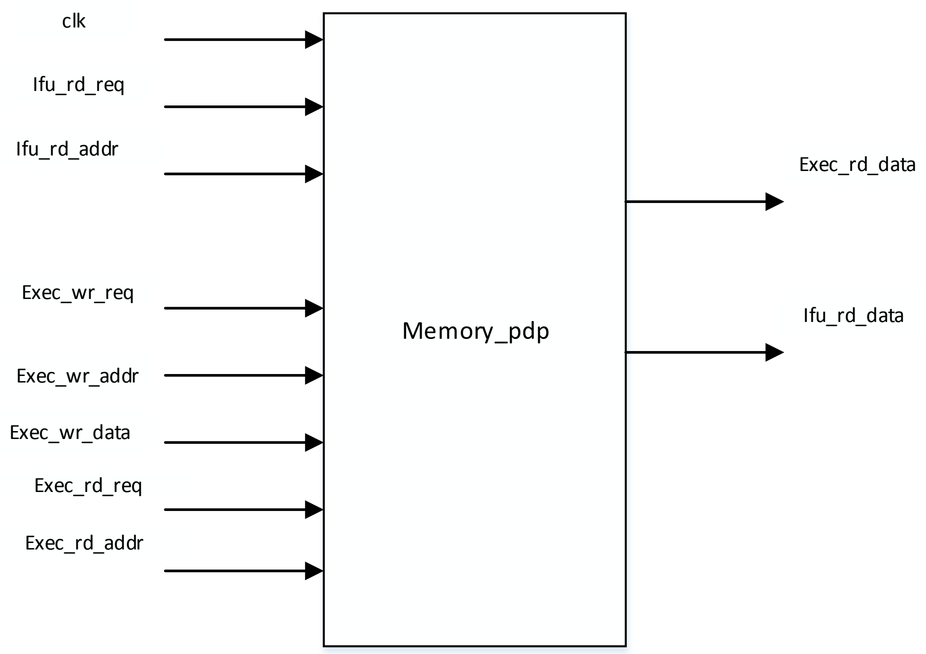


FIGURE 2:FSM: EXECUTION UNIT

Memory

Description:

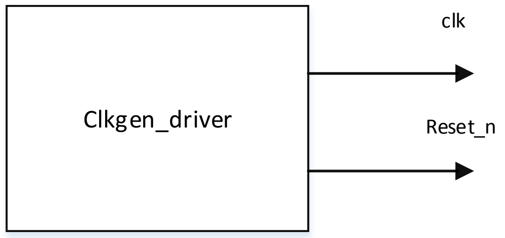
- The memory unit (4K words of 12bit each) is pre-loaded with data derived from a compiled PDP8 assembly language test. IFD unit fetches instruction from memory. EXEC unit fetches operand from memory and writes back the result.



Clock Gen

Description:

- This module provides clock and reset to all the modules in this design.



Unit Level Testbenches

Unit Level Testbench for IFD unit

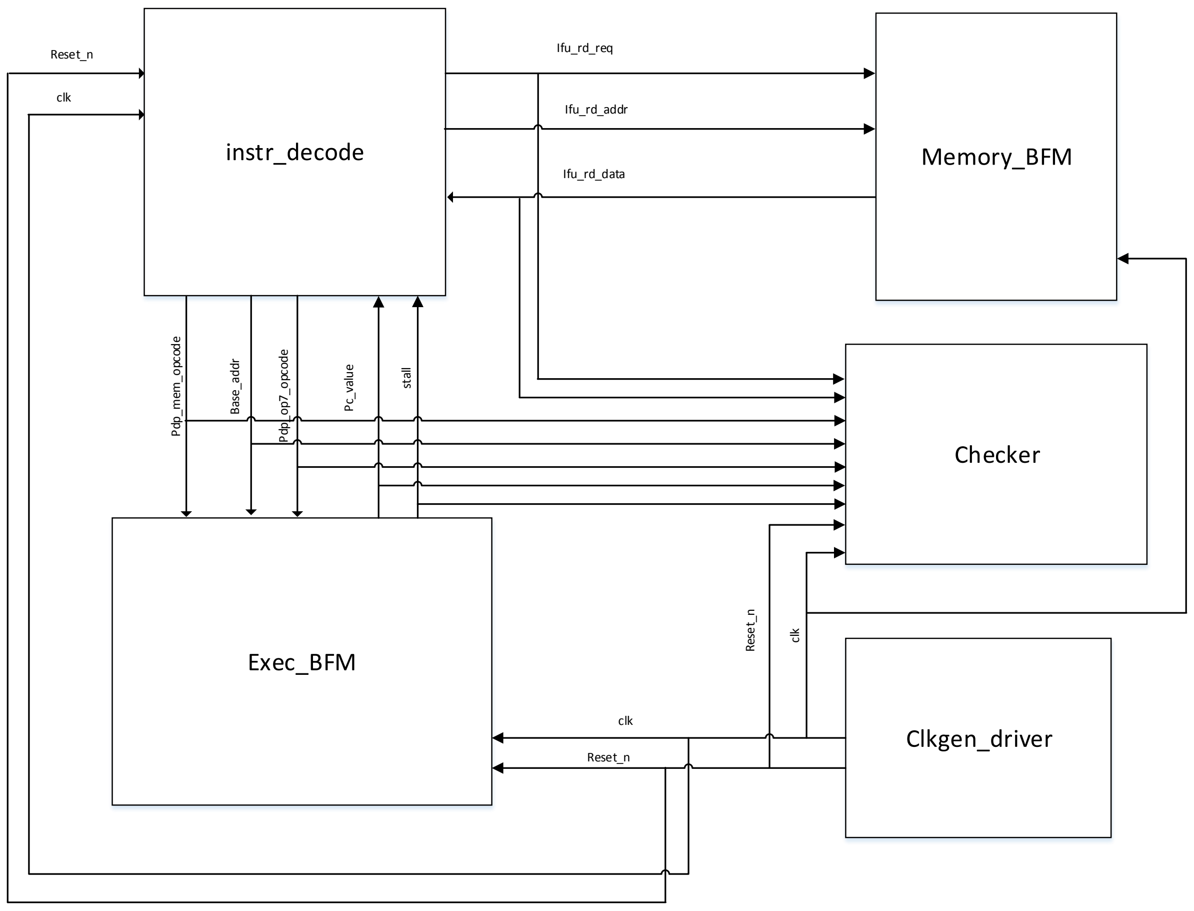


FIGURE 3: UNIT LEVEL TEST BENCH: IFD UNIT

Stimulus:

- Stimulus:
  - IFU_RD_DATA from MEMORY_BFM
  - STALL from EXEC_BFM
  - PC_VALUE from EXEC_BFM
  - RESET_N from CLK_GEN

Checks to perform:

- Checks to perform:
  - Check if opcode sent to EXEC is correctly decoded
  - Check if address sent to EXEC is correctly decoded
  - When STALL is asserted, IFD should not fetch the next instruction and no new requests sent to EXEC unit
  - If reset_n asserted at any point, then all outputs are cleared within 1 clock cycle
  - On asserting reset, next instruction should be fetched from o200
  - If current state = STALL and PC = 0200, then next state = DONE
  - No instructions are fetched after going into DONE state
  - Check timing requirements such that the FSM stays in the particular state only for 1 cycle.
  - Check whether illegal opcodes are decoded as NOPs
  - If the STALL = 0 and base address is not reached then the FSM should jump into send_req state or if STALL = 1 and base address has been reached then the FSM should jump to DONE state.
  - On reset internal registers should be reset and the FSM should be in the IDLE state.

Coverage:

- State coverage, Arc coverage, Path Coverage

Unit Level Testbench for EXEC unit

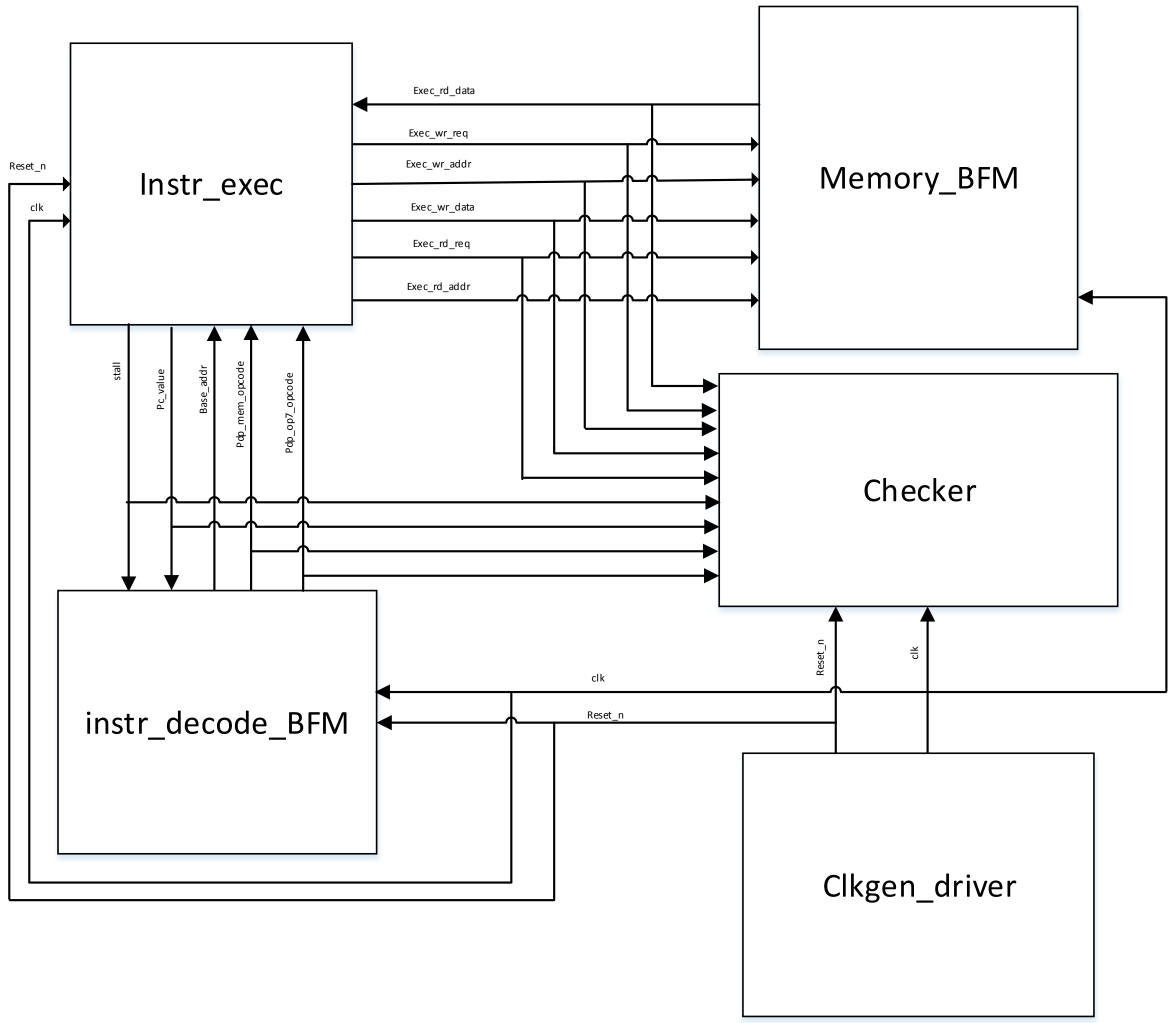


FIGURE 4: UNIT LEVEL TEST BENCH: EXECUTION UNIT

Stimulus:

- PDP_OPCODE from INSTR_DECODE_BFM
- EXEC_RD_DATA from MEMORY_BFM
- RESET_N from CLK_GEN

Checks to perform:

- Check if result is computed correctly
- Check if result computed is written to memory for certain instructions
- Check if LINK bit is correctly computed
- Check if STALL is asserted after beginning to execute each opcode
- Check if STALL is deasserted after completion of execution of each opcode
- Check if illegal opcodes are executed as NOP

Checks to be performed for each state transition:

State	**Instruction response check**	**State transition**	**Number of cycles**
CLA_CLL	Clears Acc and link register	Unstall	1
Mem_RD_REQ	Gets the address from the instruction and read request is sent to that ADDRESS	Data_rcvd	1
Data_rcvd	Data received from the memory and latched. Present value of ACC and linker are stored in a temp reg	Jumps to different states based on the instructions/opcodes	1
ADD_Acc_mem	Data from the memory location from the instruction is added with the accumulator. Carry is obtained in the link which is complemented. TAD is done here	Unstall	1
AND_Acc_mem	Data from the memory is ANDed with the data in the accumulator, AND operation is done here.	Unstall	1
ISZ_Wr_Req	Data from memory is incremented. Write add is calculated here	ISZ_UPDT_PC	1
ISZ_UPDT_PC	PC is incremented here conditionally ,ISZ is done here	Unstall	1
DCA	Write request is asserted here and it gets the address location from the opcode. Write data is the data in the Acc	CLA	1

CLA	ACC is cleared here	Unstall	1
JMS_WR_REQ	Write request is asserted here and it gets the address location from the opcode. Write data is the PC value	JMS_UPDT_PC	1
JMS_UPDT_PC	PC is updated with the value from the PC and JMS is done here	Unstall	1
JMP	PC is updated with the value from the PC and JMP is done here	Unstall	1
NOP	-	Unstall	1
Unstall	Stall is cleared Write request is cleared and PC is incremented internally	Stall	1

Coverage:

– State coverage, Arc coverage, Path Coverage

Chip level testbench

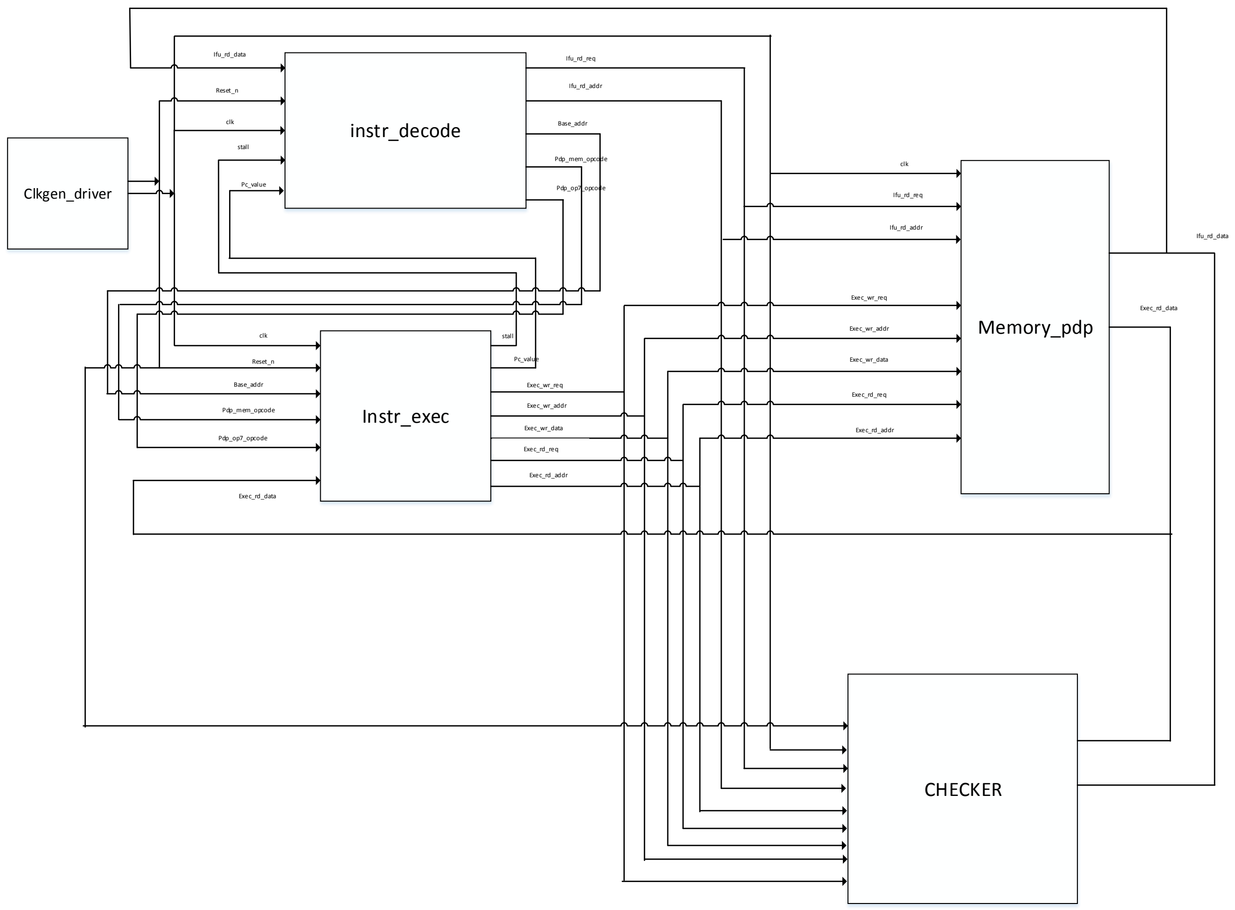


FIGURE 5: FULL CHIP TEST BENCH

Stimulus:

- Stimulus: Assembly code loaded in memory
- Stimulus should ensure that result is written to memory after each operation so that checker checks the result
- Stimulus should cover all possible opcodes
- Stimulus should also cover illegal opcodes

Checks to perform:

Tap memory interface with IFD/EXEC for

- Instruction fetched
- Operand fetched

Compute Golden Result and check if Actual Result matches Golden Result

Coverage:

State coverage, Path coverage, Arc coverage