International Institute of Information Technology
Bangalore

Functional Verification of SoCs

VL 701

Course Project - CDV of a MIPS processor integrated with Branch Prediction Logic using QuestaSim

Submitted by: *Chinmay Sultania* (IMT2021540)

Divyansh Singhal (IMT2021522)

Contents

1 The MIPS Processor

MIPS (Microprocessor without Interlocked Pipelined Stages) is a family of reduced instruction set computer (RISC) instruction set architectures (ISA) developed by MIPS Computer Systems, now MIPS Technologies, based in the United States. There are multiple versions of MIPS: including MIPS I, II, III, IV, and V; as well as five releases of MIPS32/64 (for 32- and 64-bit implementations, respectively). The early MIPS architectures were 32-bit; 64-bit versions were developed later.

- It includes the following instructions: RISC Principles: MIPS adheres to RISC design principles, emphasizing a small, highly optimized set of instructions that can execute within a single clock cycle.
- Pipeline Architecture: MIPS processors use a pipeline architecture, typically with stages like Instruction Fetch, Instruction Decode, Execute, Memory Access, and Write Back. This allows for high instruction throughput.
- Load/Store Architecture: Only load and store instructions can access memory, meaning all other instructions operate on CPU registers.
- Fixed Instruction Length: MIPS instructions are 32 bits long, which simplifies instruction decoding and pipelining.
- Register File: MIPS has 32 general-purpose registers (GPRs), each 32 bits wide in the case of MIPS32, or 64 bits wide in MIPS64.
- Simple Addressing Modes: MIPS supports a few straightforward addressing modes, including immediate, register, and base+offset.
- Branch Delay Slots: MIPS architecture employs branch delay slots, where the instruction immediately following a branch instruction is always executed, regardless of whether the branch is taken or not. This can optimize pipeline performance.

The MIPS instruction set is categorized into three main types:

- R-Type (Register): These instructions perform operations purely within registers.
- I-Type (Immediate): These instructions include an immediate value as part of the instruction
- J-Type (Jump): These instructions are used for jumping to a different part of the code.

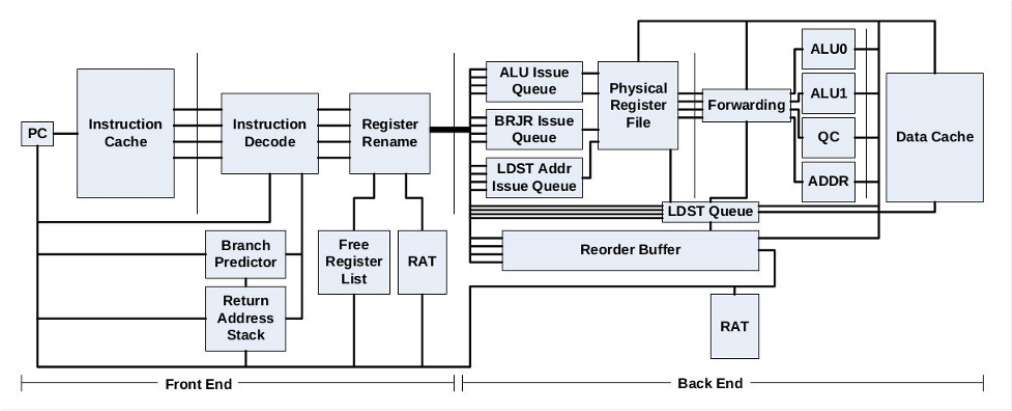

Figure 1: Block Diagram of MIPS processor

1.1 Overview of the Stages in the Processor

1. **Instruction Fetch stage (IF) -** The processor fetches instructions from the memory based on the program counter (PC). The PC points to the address of the next instruction to be fetched. In addition to fetching the next sequential instruction, the processor also predicts the outcome of branch instructions.

2. **Branch Prediction (BP) -** The branch predictor analyses the branch instructions encountered during the instruction fetch stage and predicts whether the branch will be taken or not taken. The branch predictor uses various techniques, such as branch history tables or branch target buffers, to make these predictions.

3. **Instruction Decode Stage (ID) -** In this stage, the fetched instruction is decoded. The control unit generates control signals based on the instruction opcode, which determines the operations to be performed in subsequent stages. The control unit also determines whether the branch prediction was correct or not.

4. **Execute Stage (EX) -** In this stage, the ALU (Arithmetic Logic Unit) performs the arithmetic and logical operations specified by the instruction. The ALU receives the necessary input operands, such as register values or immediate values, and produces the result. For branch instructions, the ALU also compares values to determine whether the branch should be taken based on the predicted outcome.

5. **Memory Access (MEM) -** This stage primarily handles load and store instructions. If the instruction is a load instruction, the memory stage retrieves the data from memory. If the instruction is a store instruction, the memory stage writes the data to memory. Other instructions that do not involve memory access proceed through this stage without any impact.

6. **Writeback stage (WB) -** IIn this final stage, the results of the executed instruction are written back to the register file. The results can be the ALU output, the data read from memory, or other computed values. The register file is updated with the new values, which can be used by subsequent instructions.
   If the branch prediction was incorrect, and the actual outcome of the branch instruction differs from the predicted outcome, a pipeline flush occurs. This means that the instructions following the branch instruction that were speculatively executed are discarded, and the pipeline is reloaded

with the correct instructions based on the actual branch outcome. This ensures that the processor maintains the correct program flow.

1.2 Overview of the Pipelines in the Processor

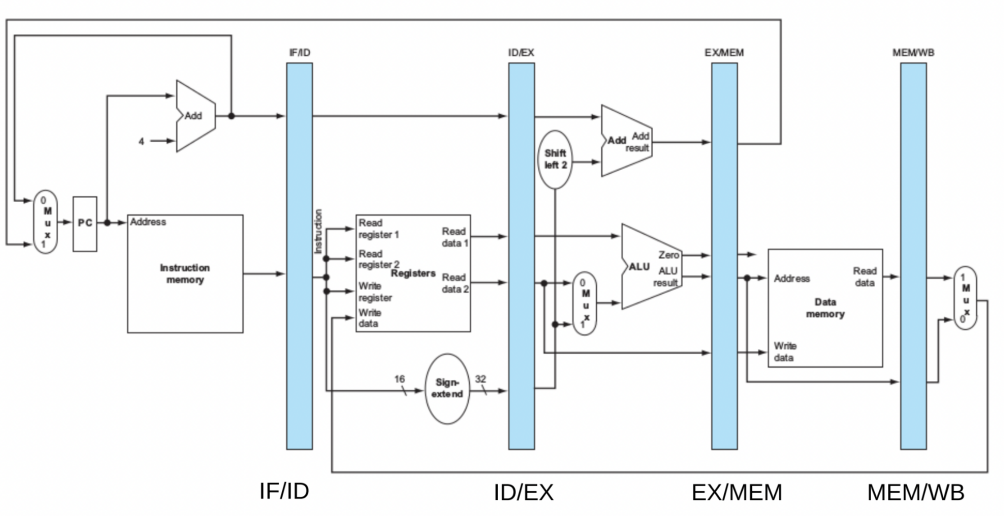

Figure 2: MIPS Pipeline

1. **IF/ID Pipeline -** It contains the data to be sent to the ID stage by the IF stage, like, fetched instruction, if the instruction is valid, etc.
2. **ID/EX Pipeline -** It contains the data to be sent to the EX stage, like the source register data, ALU Opcode, destination register address, etc.; or to be forwarded to the next instruction's ID stage like the source registers' data (in case of data hazards), etc.
3. **EX/MEM Pipeline -** It contains the data to be sent to the MEM stage, like the calculated address of memory or which memory needs to be accessed for storing/loading data.
4. **MEM/WB Pipeline -** It contains the data to be sent to the WB (LSU) stage by the EX stage, like, calculated data, calculated address, etc.

2 The Layered Testbench Approach for CDV

A layered testbench has been developed for the CDV approach. Randomized inputs were used instead of directed test vectors as the latter does not guarantee that all test-cases have been covered.

The following classes were used for constructing the layered test-bench:

- Transaction
- Generator
- Driver
- Monitor

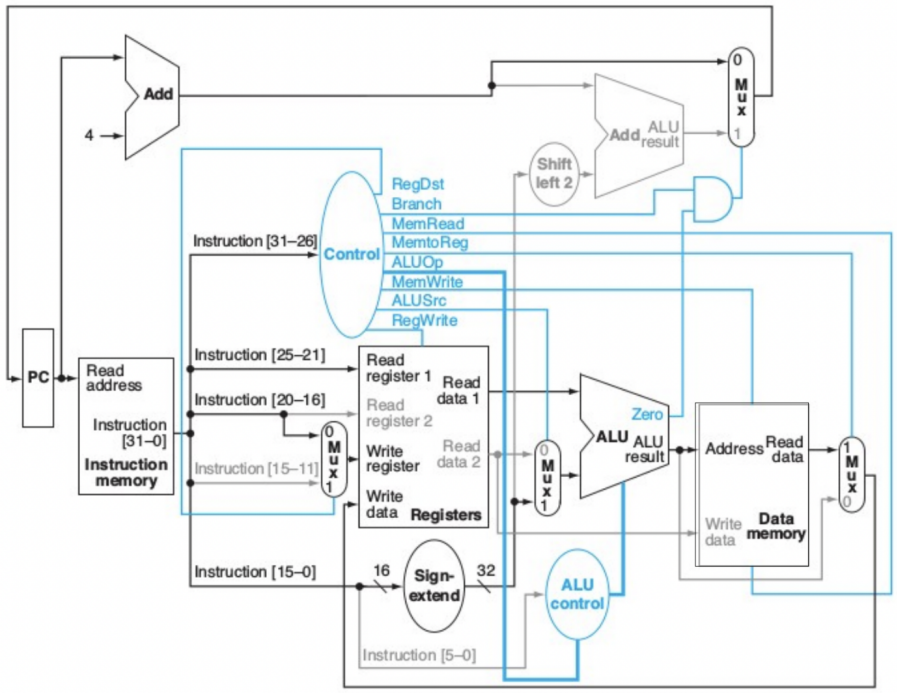

Figure 3: MIPS Control and Data path

- Scoreboard
- Environment
- Test (Program Block)
- Design Under Test (Verilog Module)
- Interface (Interface Data-type)
- Testbench-Top (Verilog Module)

1. **Transaction :** Consists of all the instructions and input-output variables that are needed by the Design Under Test (D.U.T.). These parameters are initialized as random variables of fixed length. Coverage is also measured here.
2. **Generator** : Generates random stimulus based on transaction class and given constraints. The data is sent to the driver via a mailbox
3. **Driver :** Receives the data from the Generator via Mailbox and converts it to the form of inputs to the DUT and passes it on to the interface
4. **Monitor :** Records all inputs-outputs passing into and coming out of the interface. Sends this data to the scoreboard.

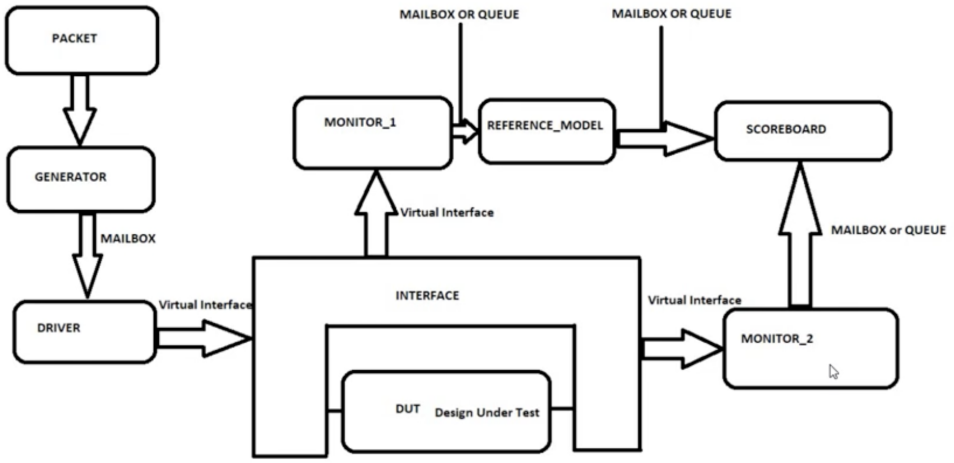

Figure 4: Layered Test-bench

5. **Scoreboard :** This implementation includes the reference model inside the scoreboard. It receives the data from the monitor via a mailbox and evaluates it according to the reference model.

6. **Environment :** All the above classes are instantiated here.

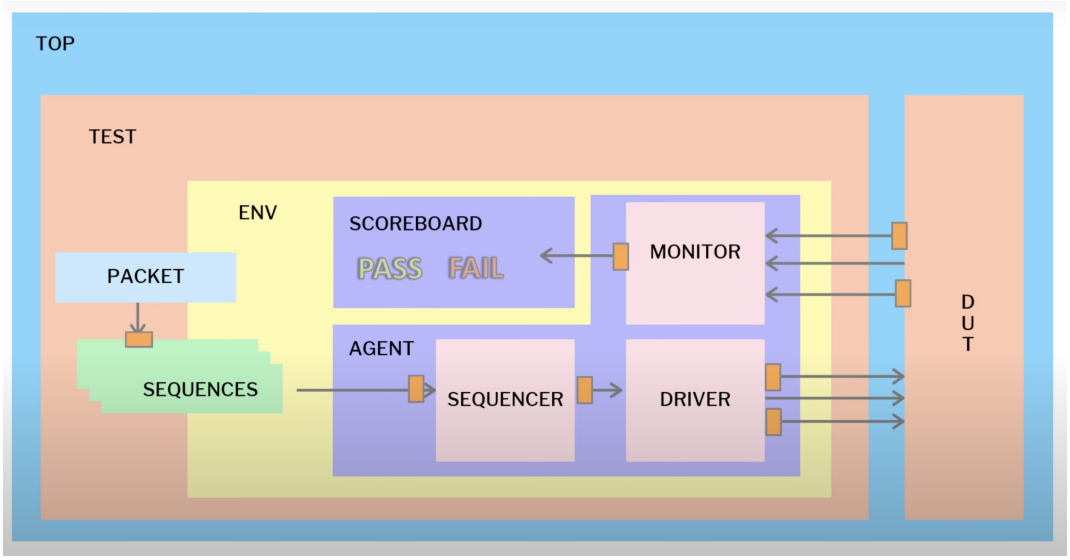

Figure 5: Layered Test-bench

3 Branch Prediction Mechanism

Branch prediction in a MIPS processor is a technique that predicts the outcome of a branch instruction before it is executed to improve the processor's performance. It does this by analysing the history of

previous branch instructions. There are several types of branch predictors, including static, dynamic, and hybrid predictors, which can be used to achieve better performance.

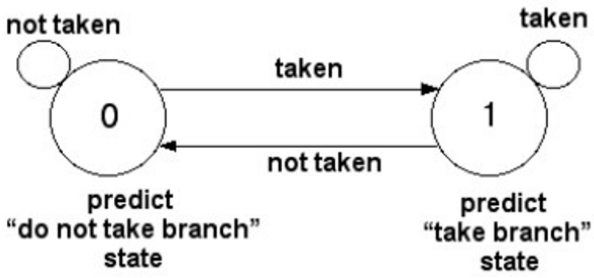

Figure 6: 1-Bit Branch predictor

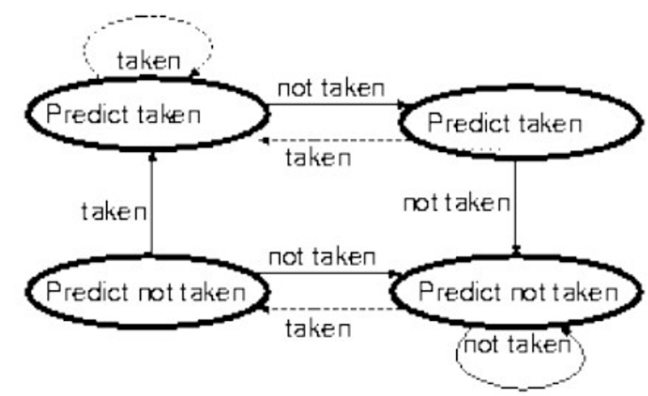

Figure 7: 2-Bit Branch Predictor

A 2nd order Markov branch predictor is a type of branch prediction mechanism that uses the history of the last two branches to predict the outcome of the current branch. It operates based on the assumption that the behaviour of a branch is influenced not only by the previous branch but also by the branch before it.

3.1 II order Markov Branch Predictor

- History Register: The predictor maintains a history register that stores the outcomes of past branches. The register can be represented as a sequence of 2-bit patterns, where each pattern corresponds to the outcome of a branch (e.g., "00" for not taken, "01" for taken, "10" for taken, etc.).
- Pattern Extraction: To predict the current branch, the predictor extracts the last two branches' outcomes from the history register. It determines the current pattern based on the previous two branches.
- Pattern Matching: The extracted pattern is compared with patterns stored in the history register

to find matches. The predictor searches for occurrences of the same pattern in the history register and records the outcomes of the branches that follow those patterns.

- Outcome Analysis: Based on the observed outcomes following each matching pattern, the predictor determines the probability of the current branch being taken or not taken. It calculates the ratio of taken branches to not-taken branches and uses this ratio to predict the current branch.
- Prediction: Using the calculated probability, the predictor makes a prediction for the current branch. If the ratio indicates a higher probability of the branch being taken, the predictor predicts a taken branch; otherwise, it predicts a not taken branch.
- Update: After the actual outcome of the current branch is known, the predictor updates the history register by shifting the patterns and incorporating the current outcome into the register. This helps in maintaining an updated history for future predictions.

A 2^{nd} order Markov branch predictor captures a higher degree of correlation between branches compared to simpler predictors like 1^{st} order Markov or single-bit predictors. By considering two previous branches, it can better capture patterns and dependencies in branch behaviour, leading to improved prediction accuracy

4 Verification flow

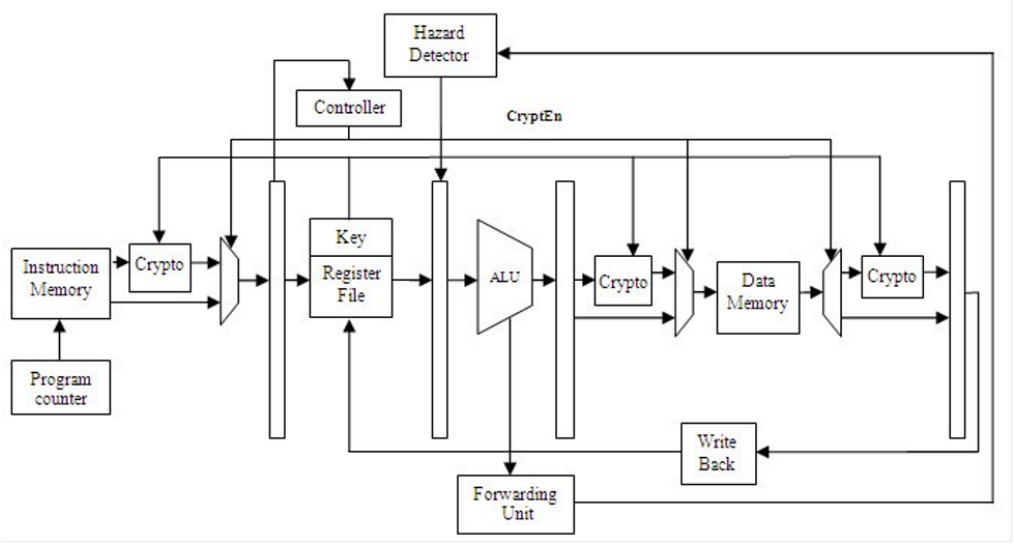

Figure 8:

These are the blocks used in our verification pipeline. We have explained it in terms of one file:

4.1 Generator:

- Creates a new instance of the Packet object.
- Randomises the values inside the "tr" object based on the constraints specified in the packet class.
- Samples the coverage group associated with the Packet object

- Puts the "tr" packet into the mailbox named "drv_mbx". A mailbox is used for communication between different threads or processes in a SystemVerilog simulation.
- Then, it waits for the "drv_done" event to be triggered by the driver.

4.2 Design:

- "Always" block triggers on the positive edge of the "clk" signal.
- If a hazard is detected ('hazardflag' is active), the 'ifidOpcode' register is cleared (set to all zeros) to flush the contents in case of a hazard.
- If no hazard is detected, the 'else' block is executed.
- Display line are used to display the current simulation times and the opcode being processed in the IF/ID stage.
- The 'opcode' input is assigned to the 'ifidOpcode' register, which holds the opcode for the next pipeline stage.
- In summary, the 'ifidPipeline' module captures the opcode from the instruction, handles hazards by flushing the contents if a hazard is detected, and provides the opcode to the next pipeline stage.

4.3 Driver:

- Driver continuously waits for incoming packets from the generator, synchronises the signals with the virtual interface, and triggers the "drv_done" event to signal completion.
- It assigns the values of tr.opcode, tr.hazardflag, tr.ifidOpcode, to their respective virtual interface signals.

4.4 Test:

. Test class creates an instance of the environment and sets up the communication between the test's mailbox and the environment's mailbox. It then runs the environment's "run" task, which triggers the execution of the test.

4.5 Environment:

- It declares the instances of the generator, driver, monitor and scoreboard modules as g0, d0, m0 and s0.
- Run function does the following tasks:-
- It assigns the virtual interface 'vif' to the driver and monitor modules
- It connects the 'drv_done' event and 'drv_mbx' mailbox between the driver and generator modules.
- It connects the 'scb_mbx' mailbox between the scoreboard and monitor modules.
- Inside a fork-join construct, it launches the execution of the driver, monitor, scoreboard, generator, and the display task of the generator in parallel.
- The 'join_any' statement ensures that the 'run' task completes when any of the forked tasks finishes execution.

4.6 Monitor:

The 'monitor' class represents a monitoring module in the ifid Pipeline of a MIPS processor. It continuously monitors the signals in the 'vif' interface, creates a 'Packet' object with the monitored values, prints the values using the print() function, and sends the transaction to the scoreboard for further analysis through the 'scb_mbx' mailbox.

4.7 Testbench:

- Declares the signals with their respective widths.
- Instantiates the DUT named "ifidPipeline" and connects the input and output signals.
- Marks the beginning of the initial block, which contains the testbench setup code.
- Declares an instance of the test class named "t0".
- Creates a new instance of the test class and assigns it to "t0".
- Connects the virtual interface of the test environment to the virtual interface of the testbench.
- Calls the "run" task of the test, initialising the test execution.
- Specified the dump file to be generated during simulation.

4.8 Scoreboard:

- First we declare two variables, 'tr' and 'ref_tr'. Of type 'packet'. These variables will hold the packets received from the mailbox.
- The 'scb_mbx.get(tr)' statement retrieves a packet from the 'scb_mbx' mailbox.
- The 'tr.print("Scoreboard ")' statement prints the contents of the received packet for debugging or monitoring purposes.
- The 'ref_tr' variable is then assigned the value of 'tr' using the 'copy()' method, creating a copy of the packet for comparison.
- If the 'hazardflag' field of the 'ref_tr' packet is set, indicating a hazard, the ifidOpcode field is cleared by setting it to all zeros (32'b0).
- If there is no hazard (the hazardflag is not set), the ifidOpcode field of ref_tr is set to the value of the opcode field of ref_tr.
- Finally, an if statement compares the ifidOpcode field of ref_tr with the ifidOpcode field of the original tr packet. If they are different, an error message is displayed, indicating a mismatch.

4.9 Transaction:

The Packet class represents a transaction or a packet in the ifid Pipeline of a MIPS processor. Here's an explanation of its components

- bit clk: Represents the clock signal associated with the transaction.

- rand bit[31:0] opcode: A randomizable 32-bit vector that represents the opcode field of the transaction. It holds the operation code for the MIPS instruction.
- rand bit hazardflag: A randomizable bit that represents the hazard flag field of the transaction. It indicates the presence of a hazard in the MIPS pipeline.
- bit[31:0] ifidOpcode: A 32-bit vector that represents the ifidOpcode field of the transaction. It holds the opcode value for the ifid stage of the pipeline.
- function void print(string tag=""): A function that prints the values of the transaction's fields. It displays the values of opcode, hazardflag, and ifidOpcode along with an optional tag parameter for identification.
- covergroup cg: A covergroup that defines coverage points and bins for the transaction. It enables coverage analysis for the transaction fields.
- opc: coverpoint opcode: Defines a coverage point for the opcode field.
- bins v1 = [0:65536], [65536:16777216], [16777216:1073741824], [1073741824:32'd4294967295]: Defines four bins to categorise the values of opcode. Each bin represents a specific range of values.
- hflag: coverpoint hazardflag: Defines a coverage point for the hazardflag field.
- ifidc: coverpoint ifidOpcode: Defines a coverage point for the ifidOpcode field. 11. bins v3 = [0:65536], [65536:16777216], [16777216:1073741824], [1073741824:32'd4294967295]: Defines four bins to categorize the values of ifidOpcode. Each bin represents a specific range of values.
- function void copy(Packet tr): A function that copies the values of another packet (tr) into the current packet object. It assigns the values of clk, opcode, hazardflag, and ifidOpcode from tr to the corresponding fields in the current object.
- function new(): A constructor function that initialises the covergroup cg for coverage analysis.

5 Verification Results

Coverage Reports

Coverage Report Screenshots for some modules: Rest are in their respective directories named as coverage.txt.

```
#
# Covergroup Coverage:
#     Covergroups                    1        na        na    78.57%
#         Coverpoints/Crosses        7        na        na        na
#             Covergroup Bins       10         7         3    70.00%
# ------------------------------------------------------------------------------------------------------------------------
# Covergroup                                                Metric       Goal       Bins    Status
#
# ------------------------------------------------------------------------------------------------------------------------
#  TYPE /testbench_sv_unit/hazardUnitPacket/cg             78.57%        100          -    Uncovered
#     covered/total bins:                                       7         10          -
#     missing/total bins:                                       3         10          -
#     % Hit:                                               70.00%        100          -
#     Coverpoint EXmr                                      50.00%        100          -    Uncovered
#         covered/total bins:                                   1          2          -
#         missing/total bins:                                   1          2          -
#         % Hit:                                           50.00%        100          -
#         bin auto[0]                                           0          1          -    ZERO
#         bin auto[1]                                           1          1          -    Covered
#     Coverpoint pcr                                       50.00%        100          -    Uncovered
#         covered/total bins:                                   1          2          -
#         missing/total bins:                                   1          2          -
#         % Hit:                                           50.00%        100          -
#         bin auto[0]                                           1          1          -    Covered
#         bin auto[1]                                           0          1          -    ZERO
#     Coverpoint hzf                                       50.00%        100          -    Uncovered
#         covered/total bins:                                   1          2          -
#         missing/total bins:                                   1          2          -
#         % Hit:                                           50.00%        100          -
#         bin auto[0]                                           1          1          -    Covered
#         bin auto[1]                                           0          1          -    ZERO
#     Coverpoint idRs                                     100.00%        100          -    Covered
#         covered/total bins:                                   1          1          -
#         missing/total bins:                                   0          1          -
#         % Hit:                                          100.00%        100          -
#         bin v1                                                1          1          -    Covered
#     Coverpoint idRt                                     100.00%        100          -    Covered
#         covered/total bins:                                   1          1          -
#         missing/total bins:                                   0          1          -
#         % Hit:                                          100.00%        100          -
#         bin v2                                                1          1          -    Covered
#     Coverpoint exRt                                     100.00%        100          -    Covered
#         covered/total bins:                                   1          1          -
#         missing/total bins:                                   0          1          -
#         % Hit:                                          100.00%        100          -
#         bin v3                                                1          1          -    Covered
#     Coverpoint ifidOp                                   100.00%        100          -    Covered
#         covered/total bins:                                   1          1          -
#         missing/total bins:                                   0          1          -
#         % Hit:                                          100.00%        100          -
#         bin v4                                                1          1          -    Covered
#
# COVERGROUP COVERAGE:
# ------------------------------------------------------------------------------------------------------------------------
# Covergroup                                                Metric       Goal       Bins    Status
#
# ------------------------------------------------------------------------------------------------------------------------
#  TYPE /testbench_sv_unit/hazardUnitPacket/cg             78.57%        100          -    Uncovered
#     covered/total bins:                                       7         10          -
#     missing/total bins:                                       3         10          -
#     % Hit:                                               70.00%        100          -
#     Coverpoint EXmr                                      50.00%        100          -    Uncovered
#         covered/total bins:                                   1          2          -
#         missing/total bins:                                   1          2          -
#         % Hit:                                           50.00%        100          -
#         bin auto[0]                                           0          1          -    ZERO
#         bin auto[1]                                           1          1          -    Covered
#     Coverpoint pcr                                       50.00%        100          -    Uncovered
#         covered/total bins:                                   1          2          -
#         missing/total bins:                                   1          2          -
#         % Hit:                                           50.00%        100          -
#         bin auto[0]                                           1          1          -    Covered
#         bin auto[1]                                           0          1          -    ZERO
#     Coverpoint hzf                                       50.00%        100          -    Uncovered
#         covered/total bins:                                   1          2          -
#         missing/total bins:                                   1          2          -
#         % Hit:                                           50.00%        100          -
#         bin auto[0]                                           1          1          -    Covered
#         bin auto[1]                                           0          1          -    ZERO
#     Coverpoint idRs                                     100.00%        100          -    Covered
#         covered/total bins:                                   1          1          -
#         missing/total bins:                                   0          1          -
#         % Hit:                                          100.00%        100          -
#         bin v1                                                1          1          -    Covered
#     Coverpoint idRt                                     100.00%        100          -    Covered
#         covered/total bins:                                   1          1          -
#         missing/total bins:                                   0          1          -
#         % Hit:                                          100.00%        100          -
#         bin v2                                                1          1          -    Covered
#     Coverpoint exRt                                     100.00%        100          -    Covered
#         covered/total bins:                                   1          1          -
#         missing/total bins:                                   0          1          -
#         % Hit:                                          100.00%        100          -
#         bin v3                                                1          1          -    Covered
#     Coverpoint ifidOp                                   100.00%        100          -    Covered
#         covered/total bins:                                   1          1          -
#         missing/total bins:                                   0          1          -
#         % Hit:                                          100.00%        100          -
#         bin v4                                                1          1          -    Covered
#
# TOTAL COVERGROUP COVERAGE: 78.57%  COVERGROUP TYPES: 1
#
```

Figure 9: Coverage Report for the Verification of the Hazard-Unit module

```
#
# Covergroup Coverage:
#     Covergroups                    1        na        na    62.50%
#         Coverpoints/Crosses        4        na        na        na
#             Covergroup Bins        5         3         2    60.00%
# ----------------------------------------------------------------------------------------------------------------
# Covergroup                                             Metric       Goal       Bins    Status
#
# ----------------------------------------------------------------------------------------------------------------
#  TYPE /testbench_sv_unit/Packet/cg                     62.50%        100          -    Uncovered
#     covered/total bins:                                    3          5          -
#     missing/total bins:                                    2          5          -
#     % Hit:                                            60.00%        100          -
#     Coverpoint ctrl                                   50.00%        100          -    Uncovered
#         covered/total bins:                                1          2          -
#         missing/total bins:                                1          2          -
#         % Hit:                                        50.00%        100          -
#         bin auto[0]                                        1          1          -    Covered
#         bin auto[1]                                        0          1          -    ZERO
#     Coverpoint in1                                   100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v1                                             1          1          -    Covered
#     Coverpoint in2                                    0.00%        100          -    ZERO
#         covered/total bins:                                0          1          -
#         missing/total bins:                                1          1          -
#         % Hit:                                        0.00%        100          -
#         bin v2                                             0          1          -    ZERO
#     Coverpoint out                                   100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v3                                             1          1          -    Covered
#
# COVERGROUP COVERAGE:
# ----------------------------------------------------------------------------------------------------------------
# Covergroup                                             Metric       Goal       Bins    Status
#
# ----------------------------------------------------------------------------------------------------------------
#  TYPE /testbench_sv_unit/Packet/cg                     62.50%        100          -    Uncovered
#     covered/total bins:                                    3          5          -
#     missing/total bins:                                    2          5          -
#     % Hit:                                            60.00%        100          -
#     Coverpoint ctrl                                   50.00%        100          -    Uncovered
#         covered/total bins:                                1          2          -
#         missing/total bins:                                1          2          -
#         % Hit:                                        50.00%        100          -
#         bin auto[0]                                        1          1          -    Covered
#         bin auto[1]                                        0          1          -    ZERO
#     Coverpoint in1                                   100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v1                                             1          1          -    Covered
#     Coverpoint in2                                    0.00%        100          -    ZERO
#         covered/total bins:                                0          1          -
#         missing/total bins:                                1          1          -
#         % Hit:                                        0.00%        100          -
#         bin v2                                             0          1          -    ZERO
#     Coverpoint out                                   100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v3                                             1          1          -    Covered
#
# TOTAL COVERGROUP COVERAGE: 62.50%  COVERGROUP TYPES: 1
#
# Total Coverage By Instance (filtered view): 62.50%
#
```

Figure 10: Coverage Report for the Verification of the Mux2to1 module

```
# WBregWrite=0 idRs=9 idRt=1 WbwriteReg=1c WBresult=79c66f9 idregA=0 idregB=0
# T=10000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=17 idRt=c WbwriteReg=1a WBresult=fe8b7fe1 idregA=0 idregB=0
# T=20000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=f idRt=17 WbwriteReg=2 WBresult=fb3a7f8 idregA=0 idregB=0
# T=30000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=1 idRt=3 WbwriteReg=1a WBresult=11e23b49 idregA=0 idregB=0
# T=40000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=14 idRt=0 WbwriteReg=1b WBresult=7bf0d8a9 idregA=0 idregB=0
# T=50000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=3 idRt=18 WbwriteReg=5 WBresult=fed82369 idregA=0 idregB=0
# T=60000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=1b idRt=1 WbwriteReg=8 WBresult=661eefb8 idregA=0 idregB=0
# T=70000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=13 idRt=13 WbwriteReg=8 WBresult=8934bbef idregA=0 idregB=0
# T=80000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=1e idRt=1a WbwriteReg=1d WBresult=96384e62 idregA=0 idregB=0
# T=90000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=13 idRt=1b WbwriteReg=11 WBresult=ff81b4a6 idregA=0 idregB=0
# T=100000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=12 idRt=18 WbwriteReg=0 WBresult=fc373ed0 idregA=0 idregB=0
# T=110000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=1f idRt=3 WbwriteReg=f WBresult=b2a18cc8 idregA=0 idregB=0
# T=120000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=1c idRt=f WbwriteReg=14 WBresult=c2b85d52 idregA=0 idregB=0
# T=130000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=8 idRt=14 WbwriteReg=12 WBresult=feb1eacd idregA=0 idregB=0
# T=140000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=1 idRs=13 idRt=2 WbwriteReg=c WBresult=bb7f2c0d idregA=0 idregB=0
# T=150000 [Driver] waiting for item ...
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# WBregWrite=0 idRs=0 idRt=0 WbwriteReg=0 WBresult=0 idregA=0 idregB=0
# Overall Coverage of ControlUnit Module = 100.000000
# coverage of covergroup = 100.000000
# coverage of coverpoint WBregWrite = 100.000000
# coverage of coverpoint idRs = 100.000000
# coverage of coverpoint idRt = 100.000000
# coverage of coverpoint WBwriteReg = 100.000000
# coverage of coverpoint WBresult = 100.000000
# coverage of coverpoint idregA = 100.000000
# coverage of coverpoint idregB = 100.000000
```

Figure 11: Coverage Report for the Verification of the RegisterFile module

```
#
# Covergroup Coverage:
#     Covergroups                     1        na         na    87.50%
#         Coverpoints/Crosses         4        na         na        na
#             Covergroup Bins         5         4          1    80.00%
#
----------------------------------------------------------------------------------------------------
-----
# Covergroup                                              Metric       Goal       Bins    Status
#
#
----------------------------------------------------------------------------------------------------
-----
#  TYPE /design_sv_unit/Packet/cg                        87.50%        100          -    Uncovered
#     covered/total bins:                                    4          5          -
#     missing/total bins:                                    1          5          -
#     % Hit:                                             80.00%        100          -
#     Coverpoint address                                100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v1                                             1          1          -    Covered
#     Coverpoint memWrite                                50.00%        100          -    Uncovered
#         covered/total bins:                                1          2          -
#         missing/total bins:                                1          2          -
#         % Hit:                                         50.00%        100          -
#         bin auto[0]                                        1          1          -    Covered
#         bin auto[1]                                        0          1          -    ZERO
#     Coverpoint writedata                              100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v2                                             1          1          -    Covered
#     Coverpoint readdata                               100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v3                                             1          1          -    Covered
#
# COVERGROUP COVERAGE:
#
----------------------------------------------------------------------------------------------------
-----
# Covergroup                                              Metric       Goal       Bins    Status
#
#
----------------------------------------------------------------------------------------------------
-----
#  TYPE /design_sv_unit/Packet/cg                        87.50%        100          -    Uncovered
#     covered/total bins:                                    4          5          -
#     missing/total bins:                                    1          5          -
#     % Hit:                                             80.00%        100          -
#     Coverpoint address                                100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v1                                             1          1          -    Covered
#     Coverpoint memWrite                                50.00%        100          -    Uncovered
#         covered/total bins:                                1          2          -
#         missing/total bins:                                1          2          -
#         % Hit:                                         50.00%        100          -
#         bin auto[0]                                        1          1          -    Covered
#         bin auto[1]                                        0          1          -    ZERO
#     Coverpoint writedata                              100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v2                                             1          1          -    Covered
#     Coverpoint readdata                               100.00%        100          -    Covered
#         covered/total bins:                                1          1          -
#         missing/total bins:                                0          1          -
#         % Hit:                                       100.00%        100          -
#         bin v3                                             1          1          -    Covered
#
# TOTAL COVERGROUP COVERAGE: 87.50%  COVERGROUP TYPES: 1
#
# Total Coverage By Instance (filtered view): 87.50%
```

Figure 12: Coverage Report for the Verification of the Data Memory module

```
#     Covergroups                     1        na        na    75.00%
#         Coverpoints/Crosses         5        na        na        na
#             Covergroup Bins         9         5         4    55.55%
# ------------------------------------------------------------------------------------------------------------------------
# Covergroup                                              Metric       Goal       Bins    Status
#
# ------------------------------------------------------------------------------------------------------------------------
#  TYPE /testbench_sv_unit/Packet/cg                      75.00%        100          -    Uncovered
#     covered/total bins:                                      5          9          -
#     missing/total bins:                                      4          9          -
#     % Hit:                                              55.55%        100          -
#     Coverpoint aluA                                    100.00%        100          -    Covered
#         covered/total bins:                                  1          1          -
#         missing/total bins:                                  0          1          -
#         % Hit:                                         100.00%        100          -
#         bin v1                                               1          1          -    Covered
#     Coverpoint aluB                                    100.00%        100          -    Covered
#         covered/total bins:                                  1          1          -
#         missing/total bins:                                  0          1          -
#         % Hit:                                         100.00%        100          -
#         bin v2                                               1          1          -    Covered
#     Coverpoint ctr                                      25.00%        100          -    Uncovered
#         covered/total bins:                                  1          4          -
#         missing/total bins:                                  3          4          -
#         % Hit:                                          25.00%        100          -
#         bin auto[0]                                          0          1          -    ZERO
#         bin auto[1]                                          0          1          -    ZERO
#         bin auto[2]                                          0          1          -    ZERO
#         bin auto[3]                                          1          1          -    Covered
#     Coverpoint EX                                      100.00%        100          -    Covered
#         covered/total bins:                                  1          1          -
#         missing/total bins:                                  0          1          -
#         % Hit:                                         100.00%        100          -
#         bin v3                                               1          1          -    Covered
#     Coverpoint ALUz                                     50.00%        100          -    Uncovered
#         covered/total bins:                                  1          2          -
#         missing/total bins:                                  1          2          -
#         % Hit:                                          50.00%        100          -
#         bin auto[0]                                          1          1          -    Covered
#         bin auto[1]                                          0          1          -    ZERO
#
# COVERGROUP COVERAGE:
# ------------------------------------------------------------------------------------------------------------------------
# Covergroup                                              Metric       Goal       Bins    Status
#
# ------------------------------------------------------------------------------------------------------------------------
#  TYPE /testbench_sv_unit/Packet/cg                      75.00%        100          -    Uncovered
#     covered/total bins:                                      5          9          -
#     missing/total bins:                                      4          9          -
#     % Hit:                                              55.55%        100          -
#     Coverpoint aluA                                    100.00%        100          -    Covered
#         covered/total bins:                                  1          1          -
#         missing/total bins:                                  0          1          -
#         % Hit:                                         100.00%        100          -
#         bin v1                                               1          1          -    Covered
#     Coverpoint aluB                                    100.00%        100          -    Covered
#         covered/total bins:                                  1          1          -
#         missing/total bins:                                  0          1          -
#         % Hit:                                         100.00%        100          -
#         bin v2                                               1          1          -    Covered
#     Coverpoint ctr                                      25.00%        100          -    Uncovered
#         covered/total bins:                                  1          4          -
#         missing/total bins:                                  3          4          -
#         % Hit:                                          25.00%        100          -
#         bin auto[0]                                          0          1          -    ZERO
#         bin auto[1]                                          0          1          -    ZERO
#         bin auto[2]                                          0          1          -    ZERO
#         bin auto[3]                                          1          1          -    Covered
#     Coverpoint EX                                      100.00%        100          -    Covered
#         covered/total bins:                                  1          1          -
#         missing/total bins:                                  0          1          -
#         % Hit:                                         100.00%        100          -
#         bin v3                                               1          1          -    Covered
#     Coverpoint ALUz                                     50.00%        100          -    Uncovered
#         covered/total bins:                                  1          2          -
#         missing/total bins:                                  1          2          -
#         % Hit:                                          50.00%        100          -
#         bin auto[0]                                          1          1          -    Covered
#         bin auto[1]                                          0          1          -    ZERO
#
# TOTAL COVERGROUP COVERAGE: 75.00%  COVERGROUP TYPES: 1
#
```

Figure 13: Coverage Report for the Verification of the Execute Unit module

6 Github link for the CDV codes

The CDV codes for the modules verified have been added to this Github Repository.

7 Challenges Faced

1. We couldn't include all the cases in our coverage report as there is a vast possibility of values the instruction set could take since it is a 32-bit architecture.
2. We were unable to decide on the appropriate bin placements to achieve maximum coverage results due to the state space being large. The scoreboards could have captured even more functionality as there were some modules that had predetermined values which had to be locally implemented.

8 Work Distribution

- **Chinmay** - Worked on CDV of GlobalBranchPredictor, controlUnit, , exempipe, forwardingUnit, ifidPipeline, idexPipeline, mux2to1_5bit, mux2to1_32bit.
- **Divyansh** - Worked on CDV of mux2to1, mux4to1, dataMemory, execute, registerFile, writeback, hazardUnit, processor.

9 Conclusion

We were able to achieve coverage driven verification methodology on the MIPS processor partly. The design proposed is a MIPS processor with a second order Markov chain predictor which is used to increase the timing performance of the processor by predicting the branches dynamically. We implemented a layered testbench in which the stimulus is generated randomly also known as the constrained random generation method. CDV is better and faster for more complex designs such as the MIPS processor but it is difficult in planning and estimating the verification completion.

10 References

1. Branch Prediction
2. MIPS Architecture
3. Layered Testbench Reading Material
4. Layered Testbench YT Resource