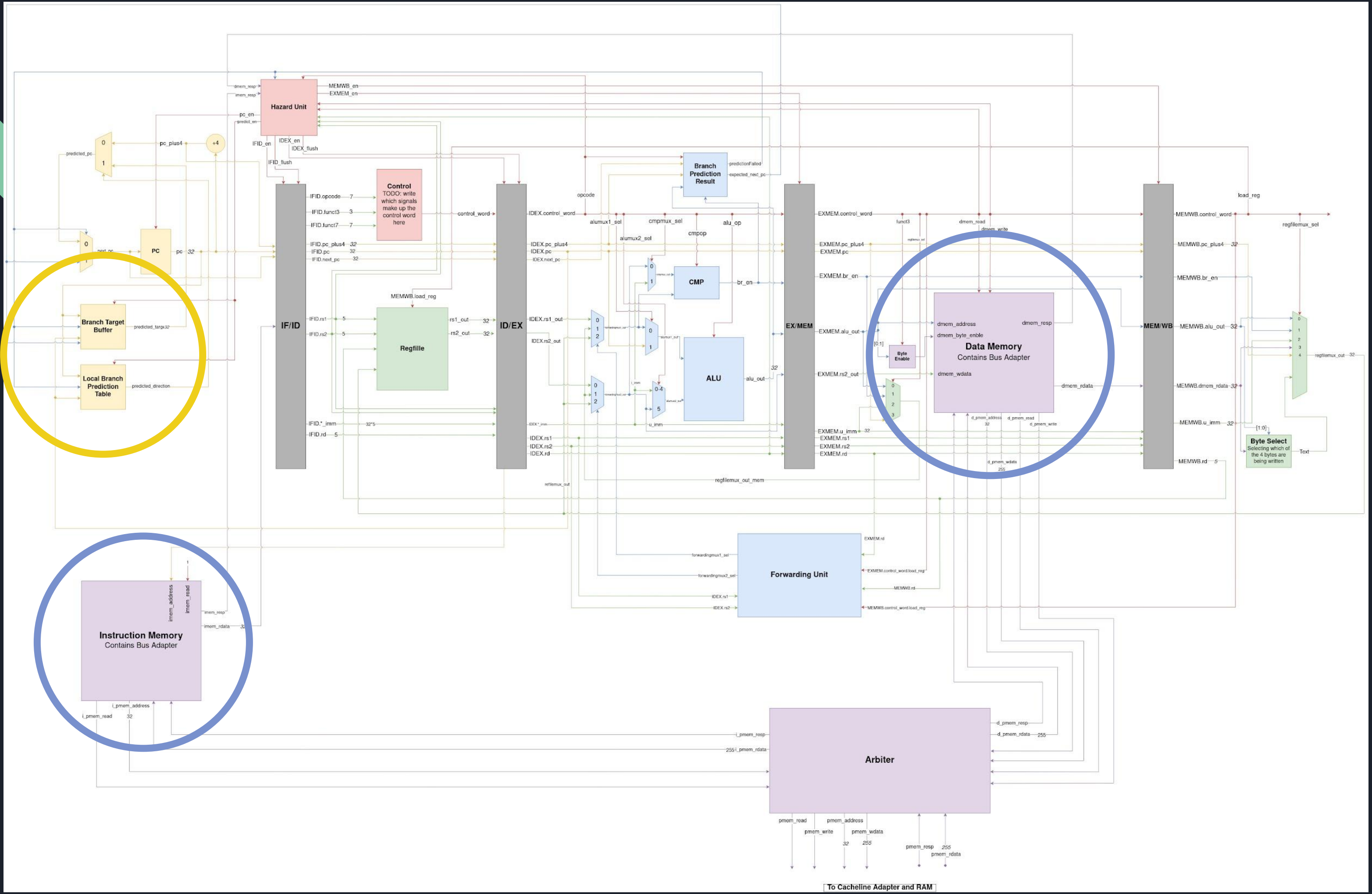
ECE 411 - MP4 Final Presentation

Team: LMP
Lakshya, Mridul, Peter

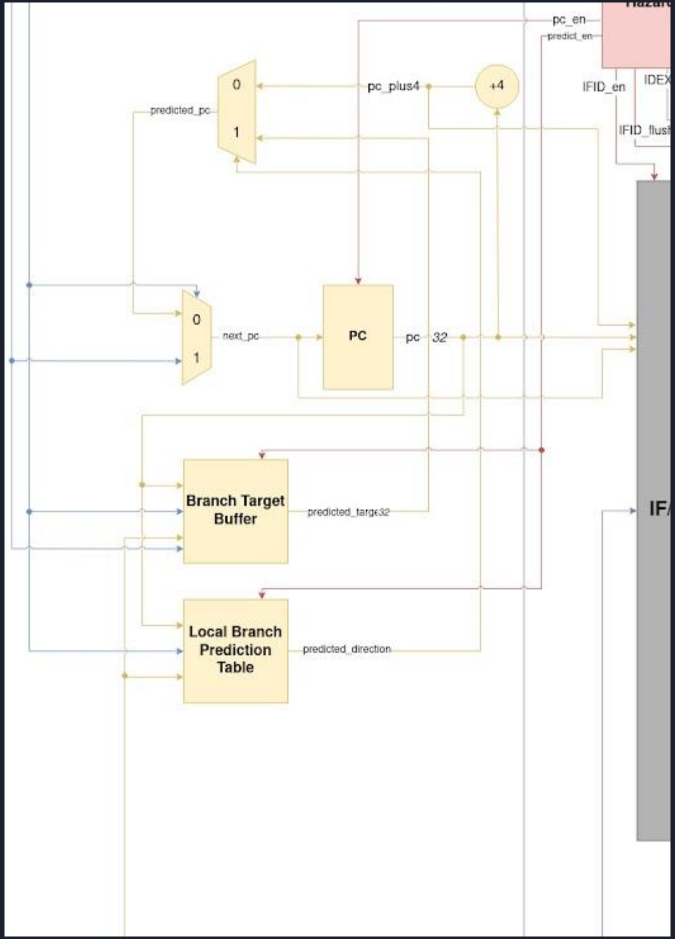
Feature Overview

- 5-stage Pipelined CPU with Hazard Detection and Forwarding
- Parameterized Local Branch Prediction Table and Branch Target Buffer
- 2-stage Pipelined & Parameterized Split Cache up to 64 Sets and 4 Ways
- RISC-V M-Extension with Wallace Tree Multiplier and Non-Storing Divider
- L2 Cache System (Not included in the Competition)
- Global Branch History Register (Not included in Competition)


Hazard Unit
MEMWB_en
EXMEM_en
dmem_resp
imem_resp
pc_en
predict_en
IFID_en
IDEX_en
IDEX_flush
IFID_flush
pc_plus4
+4
predicted_pc
next_pc
PC
pc
32
Branch Target Buffer
predicted_target
Local Branch Prediction Table
predicted_direction
IF/ID
IFID.opcode
IFID.funct3
IFID.funct7
IFID.pc_plus4
IFID.pc
IFID.next_pc
IFID.rs1
IFID.rs2
IFID.*_imm
IFID.rd
Control
TODO: write which signals make up the control word here
control_word
MEMWB.load_reg
Regfille
rs1_out
rs2_out
ID/EX
IDEX.control_word
IDEX.pc_plus4
IDEX.pc
IDEX.next_pc
IDEX.rs1_out
IDEX.rs2_out
IDEX.*_imm
IDEX.rs1
IDEX.rs2
IDEX.rd
opcode
alumux1_sel
alumux2_sel
cmpmux_sel
cmpop
alu_op
Branch Prediction Result
predictionFailed
expected_next_pc
CMP
br_en
ALU
alu_out
u_imm
EX/MEM
EXMEM.control_word
EXMEM.pc_plus4
EXMEM.pc
EXMEM.br_en
EXMEM.alu_out
EXMEM.rs2_out
EXMEM.u_imm
EXMEM.rs1
EXMEM.rs2
EXMEM.rd
funct3
dmem_read
dmem_write
Byte Enable
Data Memory
Contains Bus Adapter
dmem_address
dmem_byte_enble
dmem_wdata
dmem_resp
dmem_rdata
d_pmem_address
d_pmem_read
d_pmem_write
d_pmem_wdata
MEM/WB
MEMWB.control_word
MEMWB.pc_plus4
MEMWB.br_en
MEMWB.alu_out
MEMWB.dmem_rdata
MEMWB.u_imm
MEMWB.rd
load_reg
regfilemux_sel
regfilemux_out
Byte Select
Selecting which of the 4 bytes are being written
regfilemux_out_mem
Forwarding Unit
forwardingmux1_sel
forwardingmux2_sel
IDEX.rs1
IDEX.rs2
EXMEM.rd
EXMEM.control_word.load_reg
MEMWB.rd
MEMWB.control_word.load_reg
Instruction Memory
Contains Bus Adapter
imem_address
imem_read
imem_resp
imem_rdata
i_pmem_address
i_pmem_read
Arbiter
i_pmem_resp
i_pmem_rdata
d_pmem_resp
d_pmem_rdata
255
pmem_read
pmem_write
pmem_address
pmem_wdata
pmem_resp
pmem_rdata
To Cacheline Adapter and RAM

Feature #1: Parameterized Local Branch Prediction Table and Branch Target Buffer

- Single-cycle
  - Tried to use BRAM but this affected functionality due to 2-cycle latency
- Parametrized for Number of Sets
  - Tested for 128-512 sets
- **Added: Checking for BR/Jumps**
- Hashed using lower order bits of PC
  - Tried using other bits
  - 7-bit Global History Register
    - Goal: increased contextual information for BR/Jump
    - Too much ‘randomness’ in indexing



Feature #1 Results

Metric	**Basic Prediction**		**Checking Instruction Opcode**	
	comp1.s	comp3.s	comp1.s	comp3.s
Clock Cycles	55,885	72,665	49,537	68,835
BR Accuracy	84.0 %	56.6 %	89.3 %	67.5 %
J Accuracy	99.9 %	~100.0 %	99.9 %	~100.0 %
Total Accuracy	**87.8 %**	**75.1 %**	**91.9 %**	**81.8 %**

About 4.65% and 8.95% improvement respectively

Feature #2: 2-stage Pipelined & Parameterized Split Cache up to 64 Sets and 4 Ways

- Pipelining
  - Modified the state machine to be able to receive new memory requests while responding to the previous requests
  - Stores previous request’s address and data into registers
  - No need to stall for consecutive hits
- Parameterized Cache
  - Parameterized the number of index bits, cacheline size, and number of ways
  - Used the Pseudo LRU design from lecture
  - Tested with up to 64 sets and 4 ways
- BRAM
  - Used BRAM for reduced power and resource utilization

Feature #2 Results

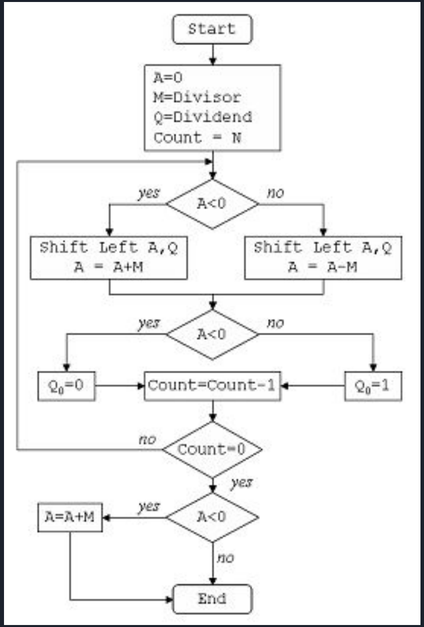
Metric	**8 Sets, 2 Ways**			**16 Sets, 4 Ways**			**32 Sets, 4 Ways**		
	comp1.s	comp2_i.s	comp3.s	comp1.s	comp2_i.s	comp3.s	comp1.s	comp2_i.s	comp3.s
Clock Cycles	50,377	385,383	341,533	50,057	237,319	72,665	50,057	119,570	72,665
I-Cache Hit Rates	99.9 %	91.0%	82.8%	99.9%	95.6%	99.9%	99.9%	99.9%	99.9%
D-Cache Hit Rates	99.7 %	98.9%	89.6%	99.7 %	99.8%	95.2%	99.7%	99.5%	95.2%

Up to 79% improvement in execution time

Feature #3: RISC-V M-Extension with Wallace Tree Multiplier and Non-Storing Divider

- Wallace Tree Multiplier
  - Implemented a 2-cycle 32 bit unsigned Wallace Tree Multiplier
  - Used 2-cycle because the delay is too long for single-cycle

- Non-Storing Divider
  - Implemented an 8-cycle unsigned Non-Storing Divider
  - The division originally takes 32 cycles, but we fit in 4 division cycles within 1 clock cycle



Divider Logic

Feature #3 Results

Metrics	comp2_i.s	comp2_m.s
Clock Cycles	119,570	25,924
Execution Time	1,195,755 ns	259,295 ns
# of Instructions	116,080	11,082
Cycles per Instruction	**1.03**	**2.34**

About 78% improvement in execution time

Overall Performance

- 32 sets, 4-way I- and D-cache
- BPT and BTB 128 sets with opcode checking in Fetch stage
- M-Extension

	comp1.s	comp2_m.s	comp3.s
FMax (MHz)	91.11		
Scaled Time (ns)	543,766	287,098	755,576
Power (mW)	607.17	608.16	641.41*
Score	9.7625E-11	1.4391E-11	2.7668E-10
Geometric Mean	**7.2981E-11**		

Utilization:
Logic - 20,366 (56%)
Registers - 8181 (23%)
Total Pins - 165 (57%)
M9K Blocks - 68 (21%)

*This is an upper estimate from Quartus since the VCD file could not be generated due to space limitations.

Challenges & Optimizations

- Challenges Faced
  - Multiplier causing too much delay
  - Larger BTB led too high resource utilization
  - Weren’t able to pipeline BTB to be able to use BRAM
    - BTB was cause for high power and resource utilization
- Tricks to improve performance
  - Removing Priority Encoding (If-Else constructs)
  - Quartus based optimizations
    - Fitter effort → set for Performance
    - Register Retiming
    - Physical Synthesis

Future Developments

- More sophisticated Branch Prediction
  - Default taken predictions for BR/JAL
  - Tournament Predictor with global history
  - Associative BPT and BTB
    - Remove some aliasing
- 8-Way associative cache
  - 2 → 4 way gave a large improvement