RISC-V SoC Microarchitecture Design & Optimization

Design Review #2

Group 23
Instructor & Sponsor: Weikang Qian
Group Member: Li Shi, Jian Shi, Yichao Yuan, Yiqiu Sun, Zhiyuan Liu

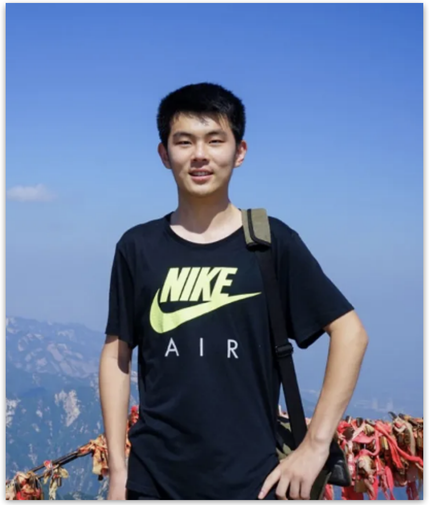
Team Members

Zhiyuan Liu

Jian Shi

Yichao Yuan

Yiqiu Sun



Li Shi

Overview

- Introduction
- Concept Generation & Decision Making
  - Instruction Set Architecture
  - Overall Microarchitecture Design
  - Local Design #1, #2
- First Version Design
- Conclusion

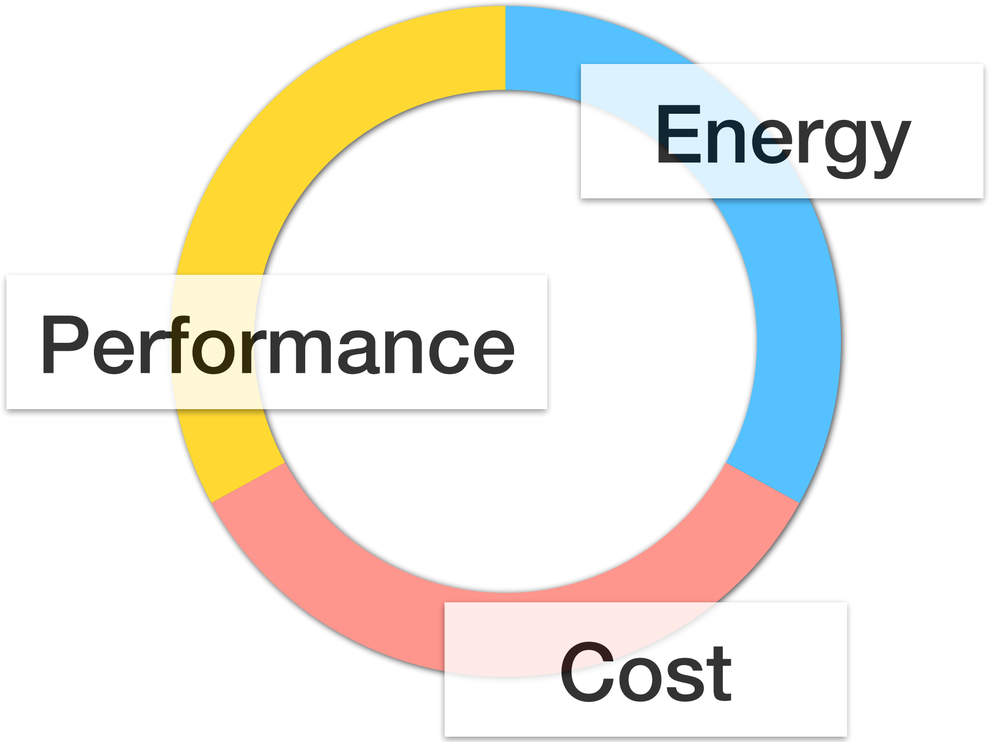
1. Introduction

Introduction

Design Problems

Domain Specific Optimization: Future of Computing

Fig 1. Tesla self-driving cars.
Source: www.businessinsider.com/tesla-autopilot-full-self-driving-subscription-early-2021-elon-musk-2020-12



Introduction

Customer Requirement (CR)

General

- Be compatible with RISC-V apps
- Have detailed docs for reference
- Is inexpensive

Performance

- Run fast for normal arithmetic programs
- Run fast for machine learning (ML) apps
- Run fast for memory-bound apps

Embedded-System

- Be easy to configure for various parameters
- Have good support for multiple I/O devices
- Save power
- Respond quickly

Introduction

Engineering Specifications (ES)

	Unit	Target Value
Support RV32G instruction set architecture (ISA)	-	Yes
Core frequency on FPGA test platform	MHz	100
Number of pipeline stages	-	9
Instructions executed per clock cycle (IPC)	-	0.5
Support instruction dynamic scheduling	-	Yes
Typical total cache size	KB	32
Number of function units	-	6
Average response time to a request for service	ms	10
Usage of look-up tables (LUT) on FPGA	k	120
Usage of block RAM (BRAM) on FPGA	-	50
Usage of digital signal processor (DSP) on FPGA	-	30
Power consumption on target FPGA test platform	W	5
Operations processed within unit energy	MOp/J	25
Number of flexibly-configured modules	-	10
Number of I/O device types	-	3
User guide and programmers manual	-	Yes

Introduction

Project Plan

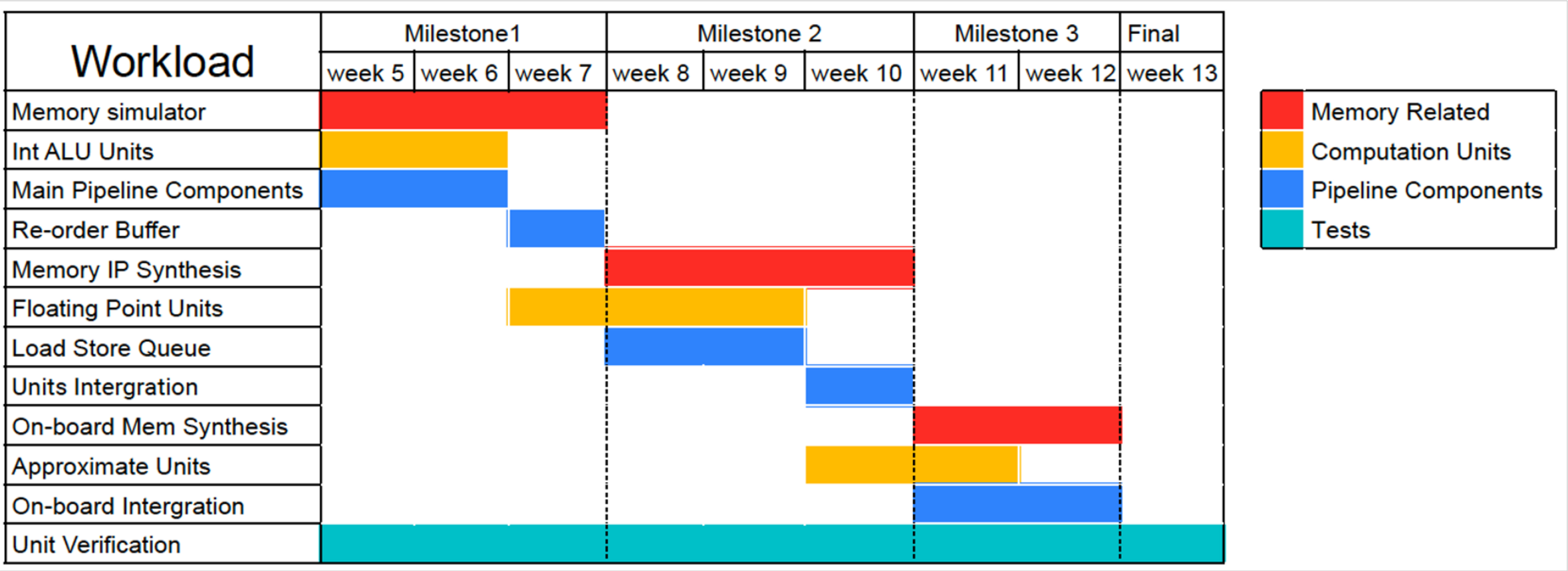


Figure 2. Gantt chart.

2. Concept Generation & Decision Making

Concept Generation & Decision Making

Instruction Set Architecture (ISA)

	ARMv7 ISA	RISC-V 32G ISA
Pro	Support complicated instructions	Custom design space Easy to implement
Con	No custom design space Difficult to implement	Only support a small number of instructions

Source:

1. A. Armstrong, et al. “ISA Semantics for ARMv8-a, RISC-v, and CHERI-MIPS." *Proceedings of ACM on Programming Languages*, no. POPL (2019): 1-31.
2. ARM Limited. www.arm.com
3. RISC-V International. www.riscv.org

Concept Generation & Decision Making

Instruction Set Architecture (ISA) - Decision

RISC-V®

RISC-V 32G ISA

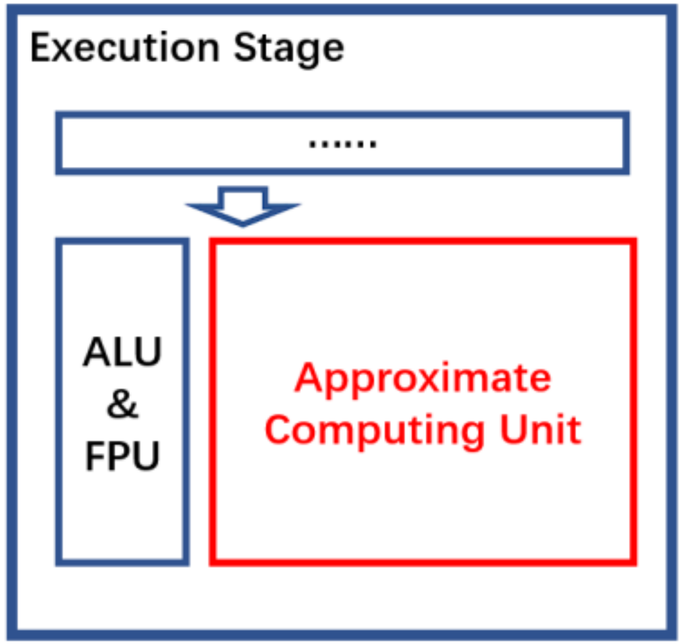
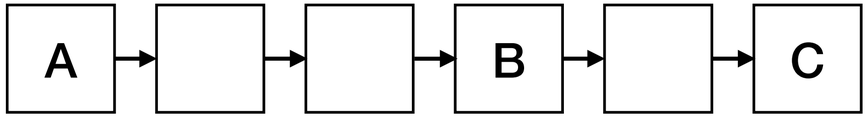
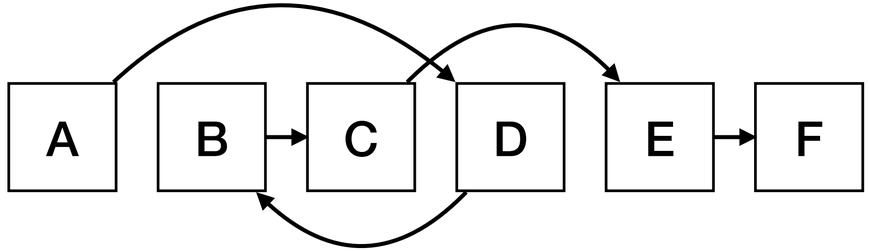


Fig 3. Execution stage with approximate computing unit.

Concept Generation & Decision Making

Microarchitecture Design: Instruction Scheduling

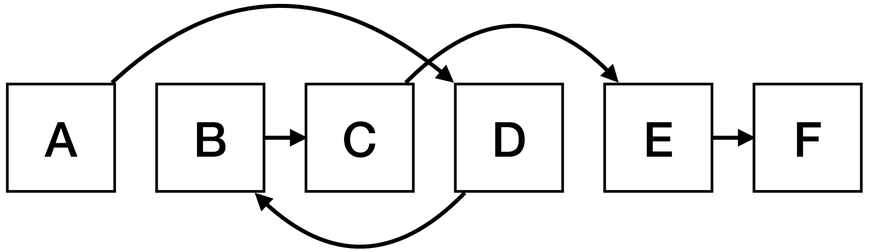




	Static (In-order)	Dynamic (Out-of-order)
Pro	Easy to implement Little overhead	Better performance
Con	Limited performance	Hard to implement Extra overhead

Concept Generation & Decision Making

Microarchitecture Design: Instruction Scheduling - Decision



Dynamic (Out-of-order)

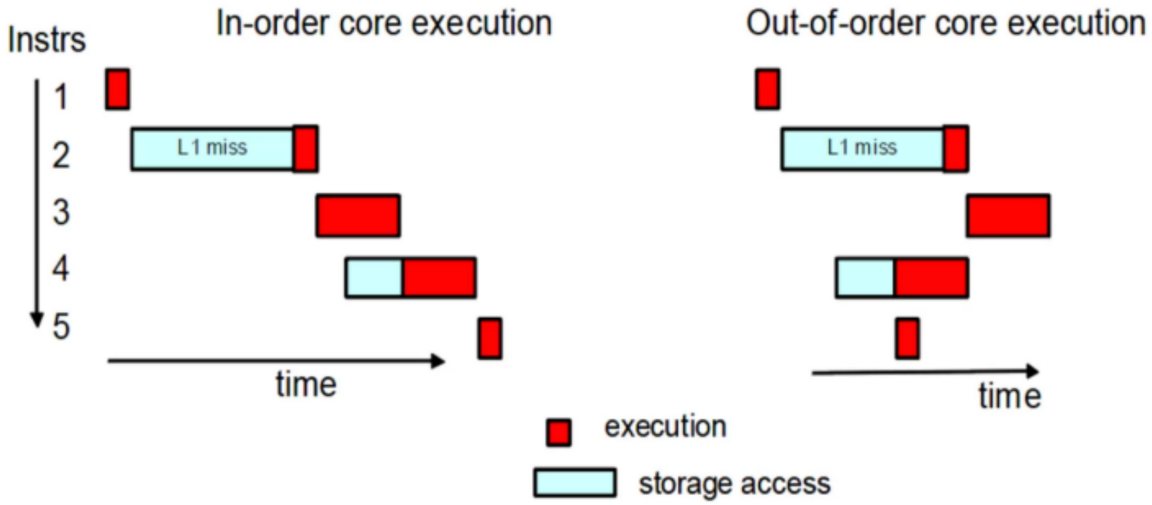
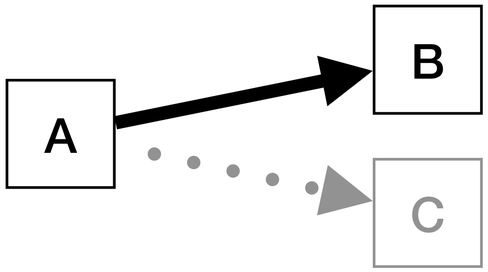
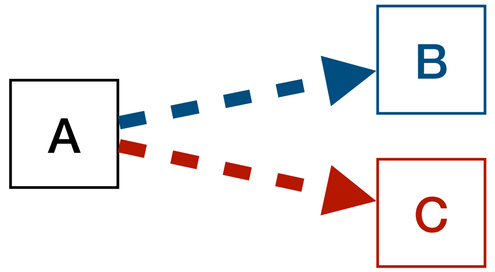


Fig 4. In-order execution vs. out-of-order execution.
Source: Anatomy of a CPU, https://www.techspot.com/article/2000-anatomy-cpu/

Concept Generation & Decision Making

Local Design #1: Branch Prediction



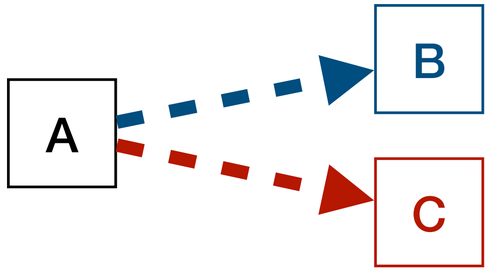


	Static (Compile Time)	Dynamic (Runtime)
Pro	Simple to implement	Better prediction accuracy
Con	Low accuracy (30%-40%*)	More hardware cost

*Data source: Patterson and Hennessy. *Computer Organization and Design*, ARM Edition.

Concept Generation & Decision Making

Local Design #1: Branch Prediction - Decision



Dynamic (Runtime)

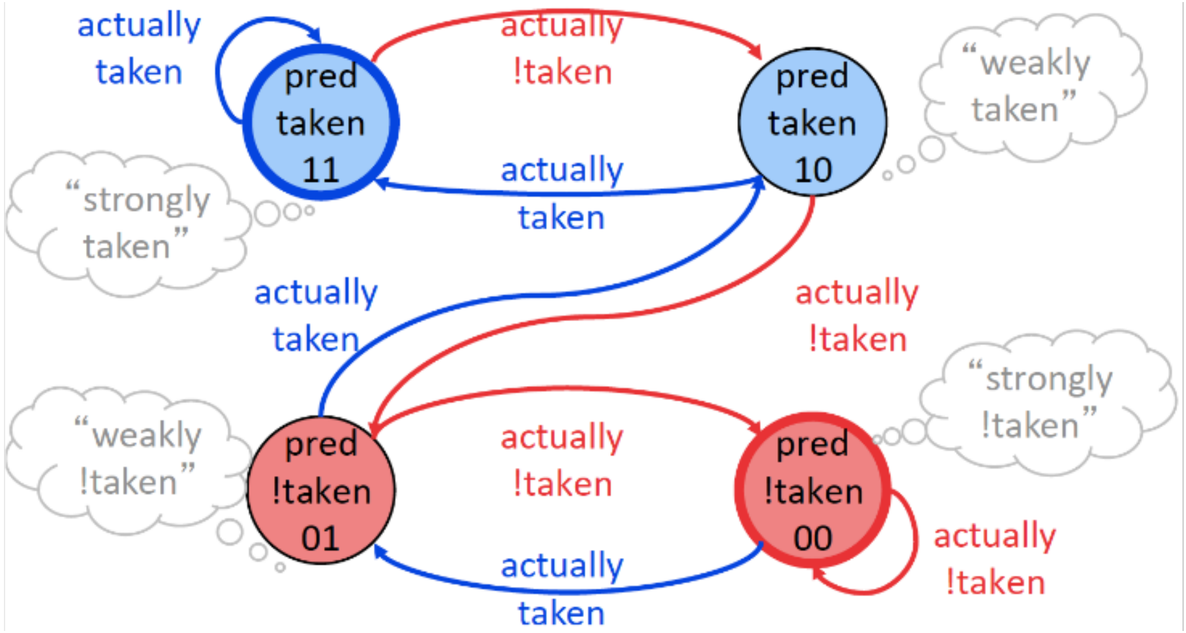


Fig 5. State machine for 2-bit saturating counter.
Source: Patterson and Hennessy. *Computer Organization and Design*, ARM Edition.

Concept Generation & Decision Making

Local Design #2: Floating Point Execution Units

	Floating Point Unit (FPU)	Approx. Comp. Unit (ACU)
	3.4*2.8 = 9.52	3.4*2.8 ≈ 9.5
Pro	Accurate	Fast Low power usage
Con	Slow High power usage	Inaccurate

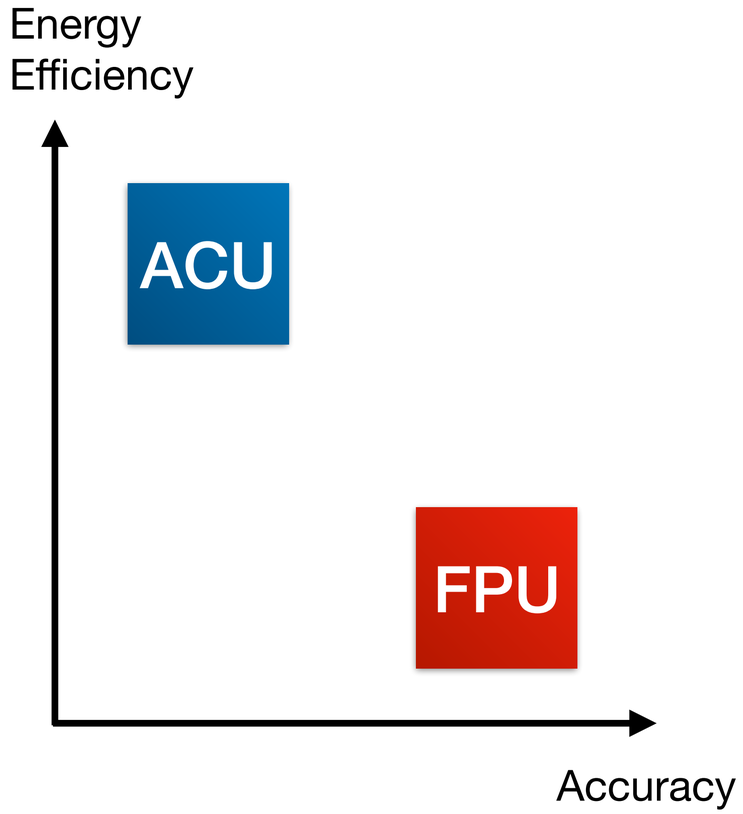


Fig 6. ACU & FPU comparison.

Concept Generation & Decision Making

Local Design #2: Floating Point Execution Units - Decision

3.4*2.8 = 9.52

Floating Point Unit (FPU)

3.4*2.8 ≈ 9.5

Approx. Comp. Unit (ACU)

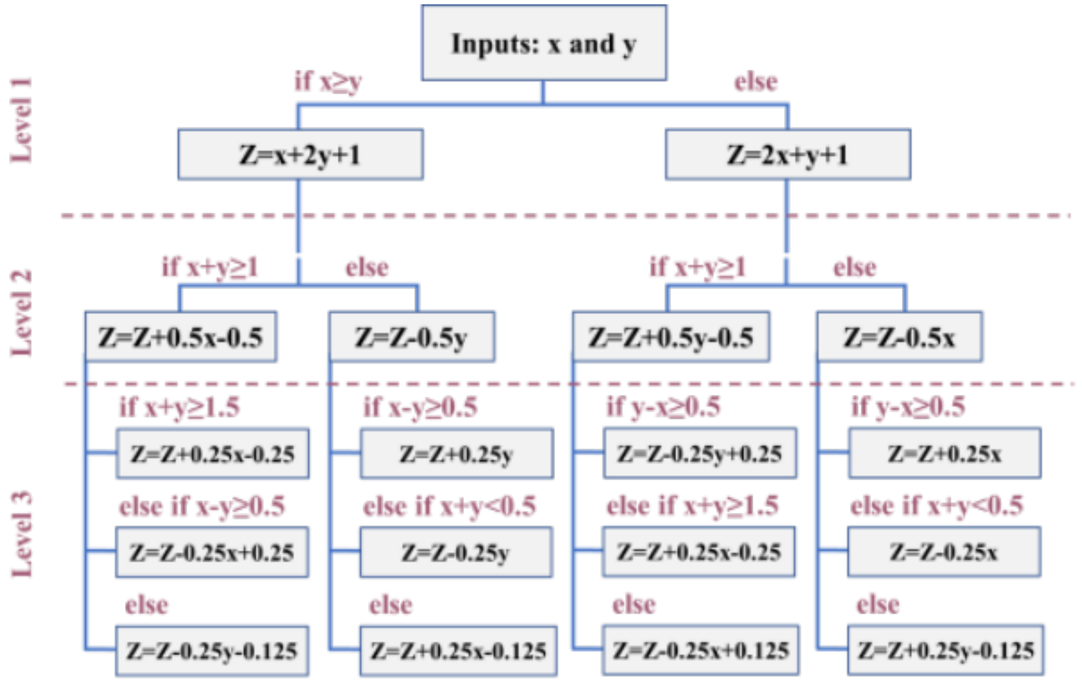


Fig 7. Flow of the approximate multiplier.

Source: C. Chen, S. Yang, et al. "Optimally Approximated and Unbiased Floating-Point Multiplier with Runtime Configurability," *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2020, pp. 1-9.

Decision Matrix

Design Criterion	Weight Factor	Unit	Branch Prediction				Floating Point Ex Unit					
			Static		Dynamic		FPU + APU		FPU		Approx	
			Score	Rating	Score	Rating	Score	Rating	Score	Rating	Score	Rating
Compability	0.25	%	10	2.5	10	2.5	4	1	7	1.75	2	0.5
Performance (Latency per instruction)	0.35	%	10	3.5	9	3.15	9	3.15	2	0.7	10	3.5
Hardware Complexity	0.18	%	10	1.8	6	1.08	7	1.26	9	1.62	8	1.44
Accuracy	0.22	%	4	0.88	9	1.98	9	1.98	10	2.2	8	1.76
				8.68		**8.71**		**7.39**		6.27		7.2

3. First Version Design

First Version Design

Simplified Pipeline Diagram (9 Stages)

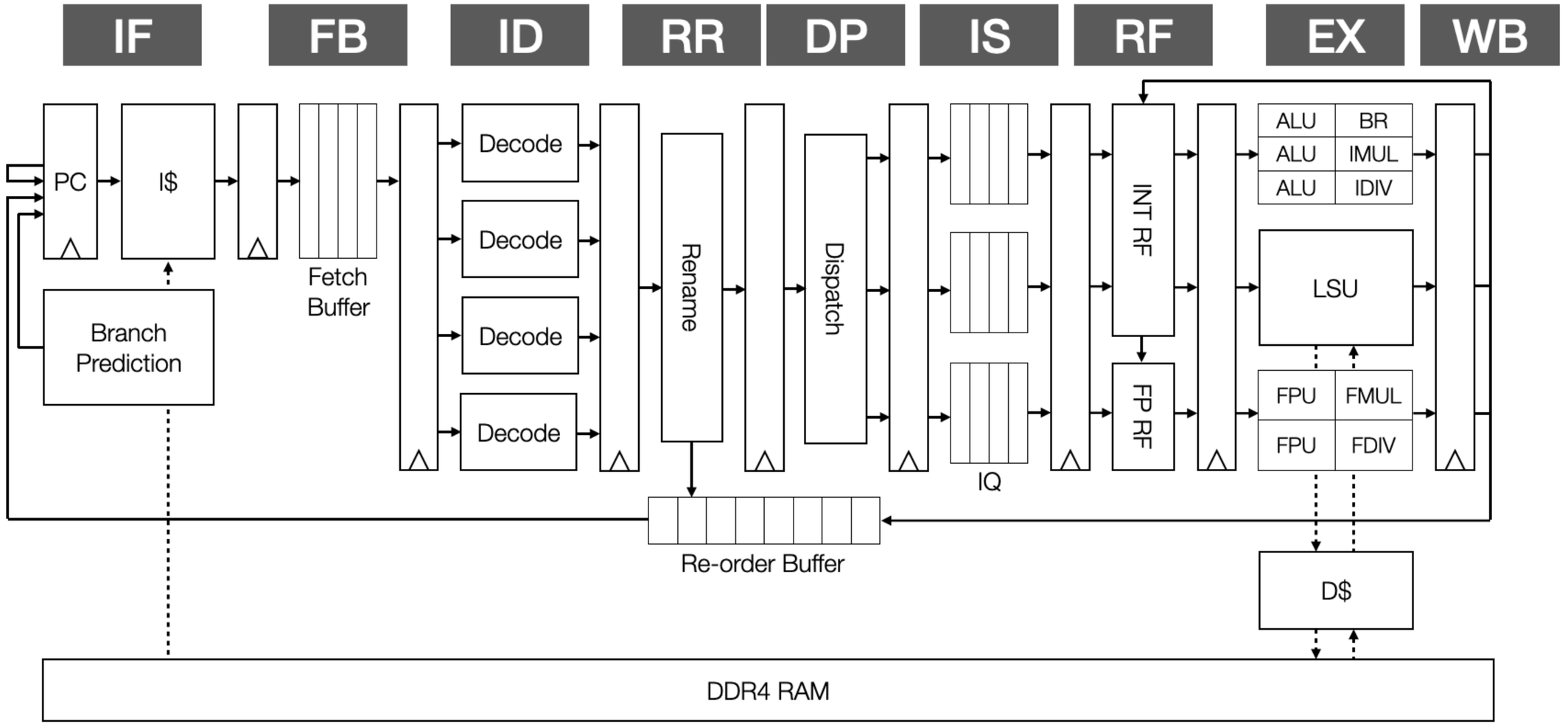
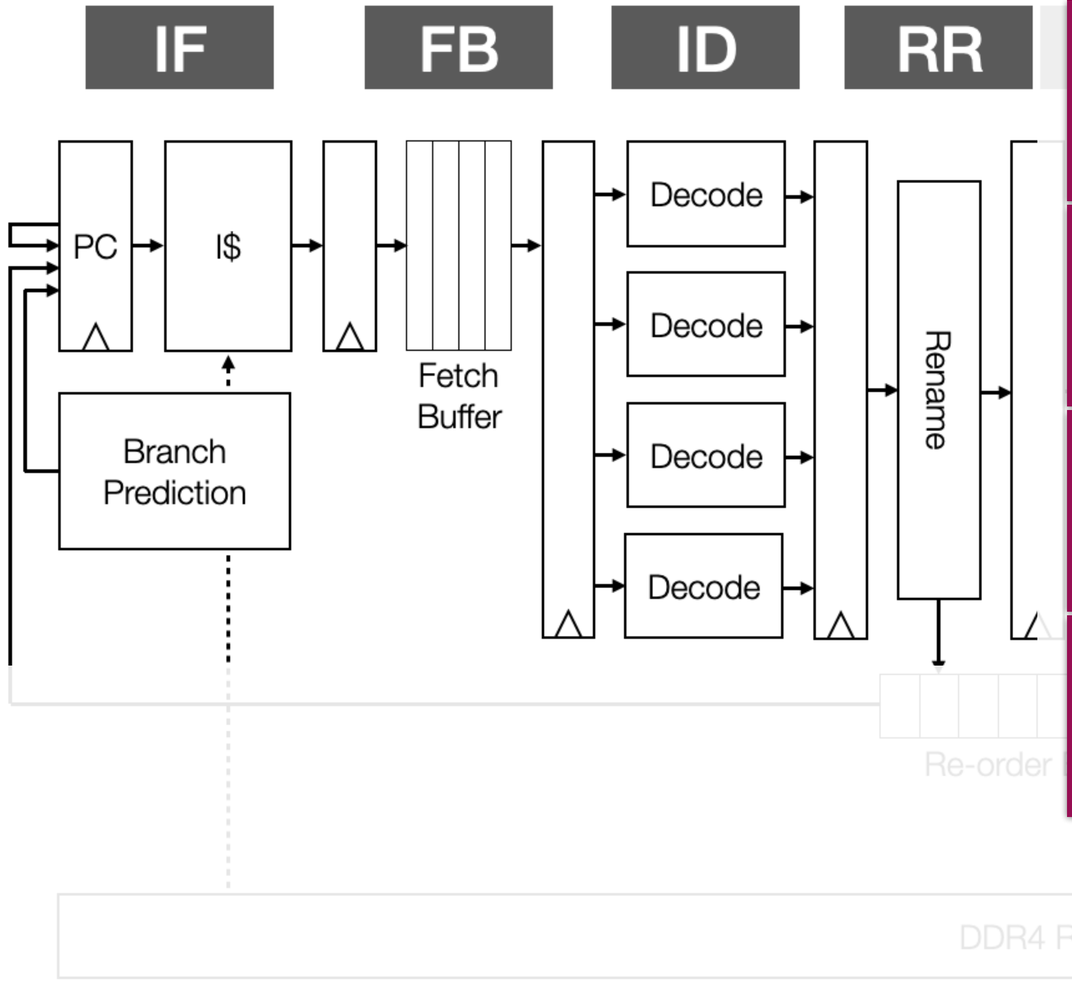


Fig 8. Simplified pipeline diagram of first version processor design.

First Version Design

Frontend Design



IF (Instruction Fetch)
Fetch instructions from memory

FB (Fetch Buffer)
Temporarily store instructions

ID (Instruction Decode)
Transform instructions into micro-ops

RR (Register Renaming)
Rename the destination operand

First Version Design

Backend Design

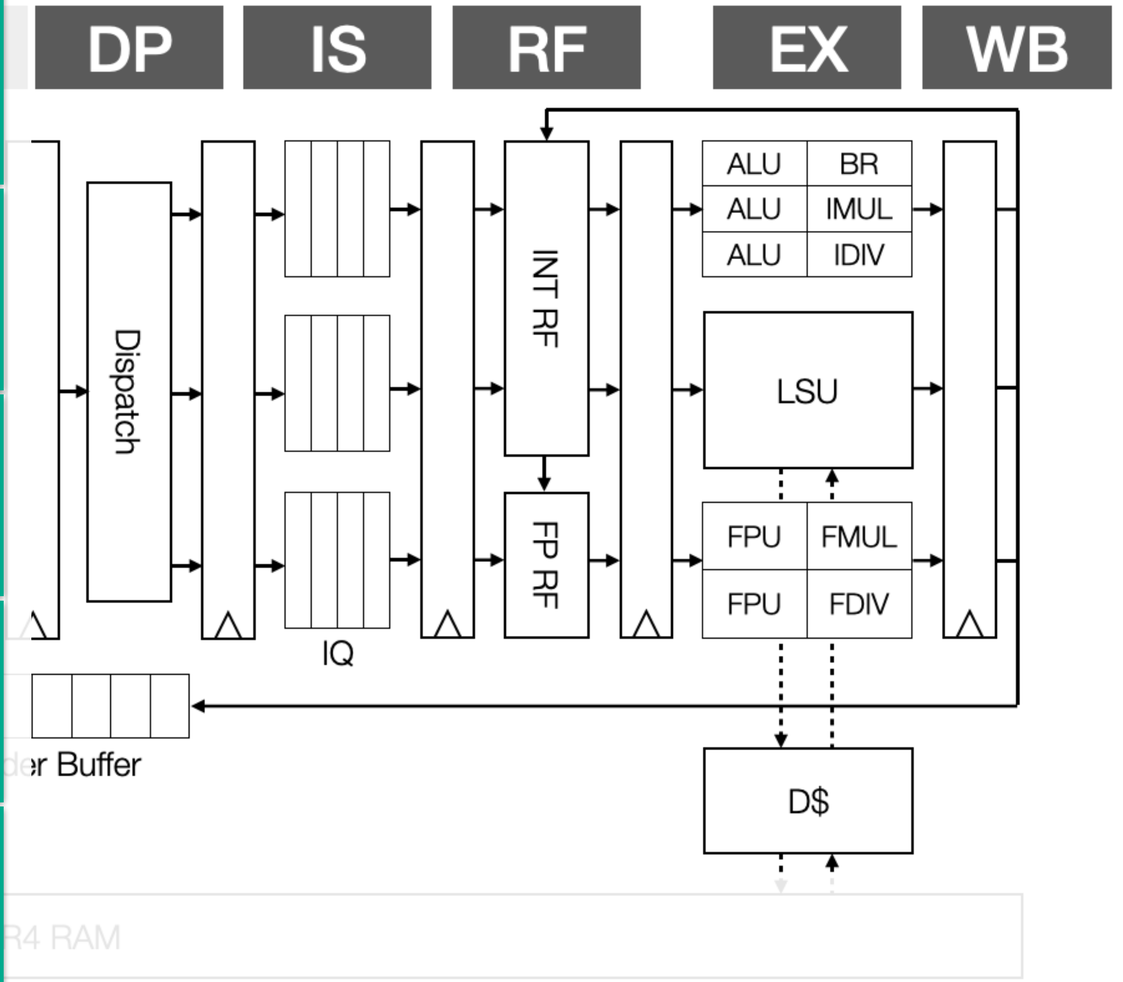
DP (Dispatch)
Dispatch instructions to backend

IS (Issue)
Issue instructions to execution units

RF (Register File)
Read values from register file

EX (Execution)
Execute the instruction

WB (Write Back)
Write the result to register file



4. Conclusion

Conclusion

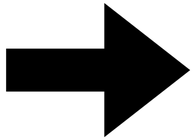
Instruction Set Architecture	ARMv7	RISC-V 32G
Instruction Scheduling	Static (In-Order)	Dynamic (Out-of-order)
Branch Prediction	Static (Compile Time)	Dynamic (Runtime)
Floating Point Execution Units	Floating Point Unit (FPU)	Approx. Comp. Unit (ACU)

Conclusion

Fast

Inexpensive

Energy-efficient



RISC-V

Out-of-order Design

Domain-specific Optimization

Thank you!

RISC-V SoC Microarchitecture Design & Optimization

Group 23
Instructor & Sponsor: Weikang Qian
Group Member: Li Shi, Jian Shi, Yichao Yuan, Yiqiu Sun, Zhiyuan Liu