# Modified Single-Cycle MIPS Architecture with DSP Unit

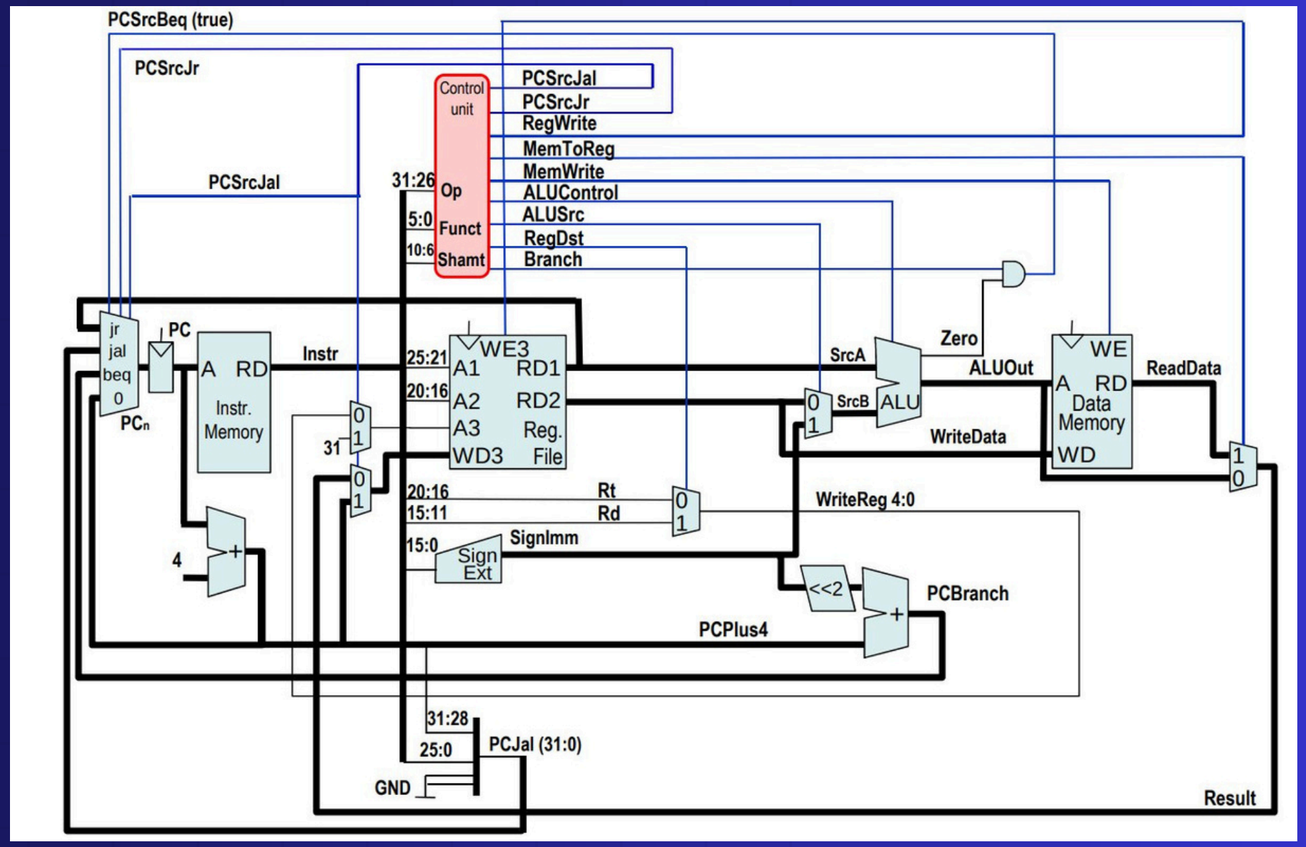
Presented by: Mohaned Asker, Mohamed Alaa, Marcelino Emad, Sara Abdelnasser, Hana Nasef

# What is a Single-Cycle Processor?

A Single-Cycle CPU completes each instruction in exactly one clock cycle.

**Features**:

- One instruction = one clock cycle.
- Simple design.
- Long clock cycle (to accommodate slowest instruction).
- Easier to understand and implement.

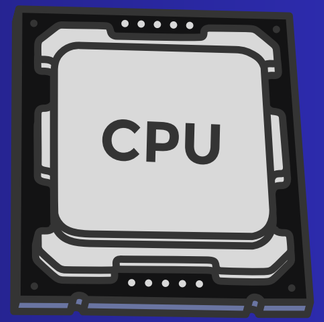


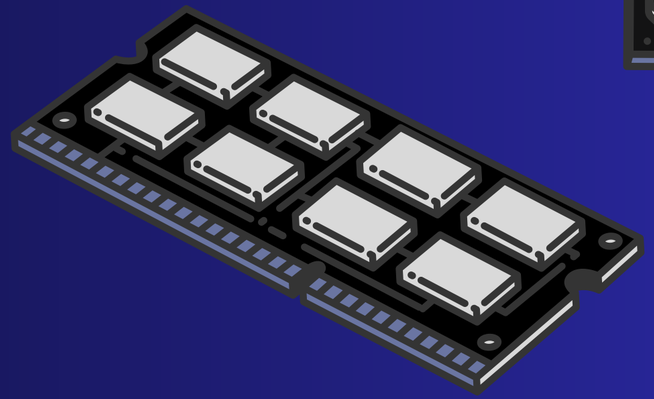
# What is DSP (Digital Signal Processing)?

**Digital Signal Processing** is the use of mathematical operations to manipulate signals (like audio, images, or sensor data) in digital form.

A DSP unit is a specialized hardware block inside a processor that:
Performs operations like:

- Filtering (FIR, IIR)
- Convolution
- FFT (Fourier transform)
- Multiply-Accumulate (MAC)



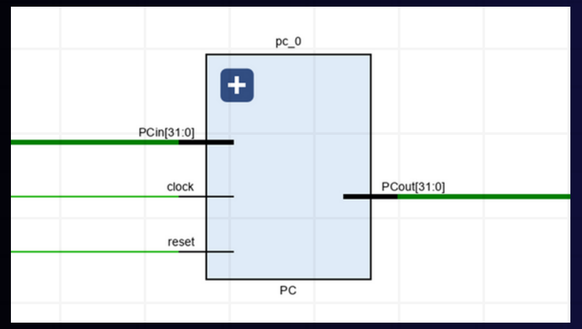



**Why DSP?**

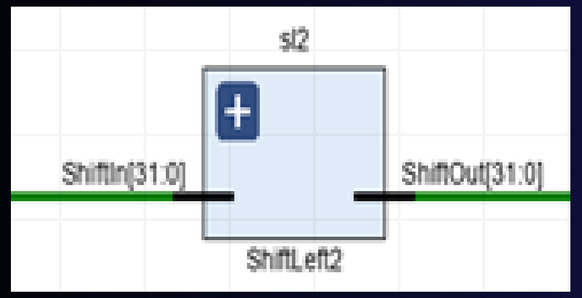
- Real-time signal handling
- Reduces CPU load
- Improves performance

# Phase 1 – Standard MIPS CPU Implementation

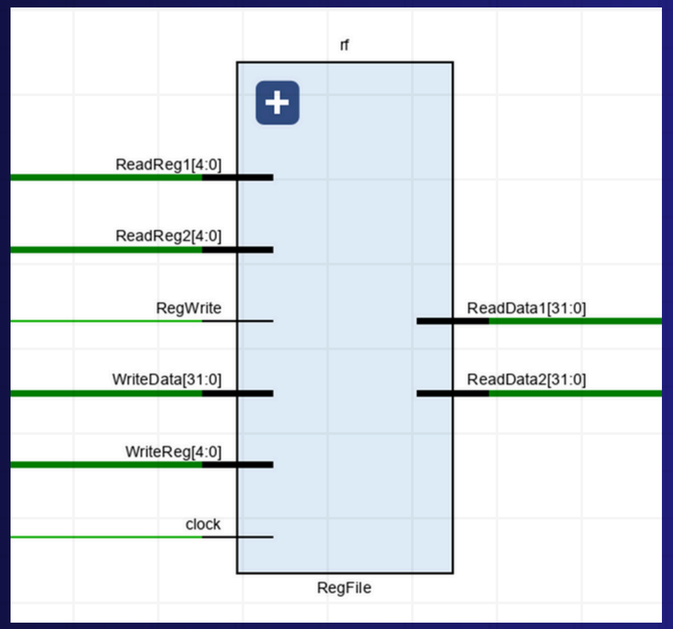
## 1. Datapath:



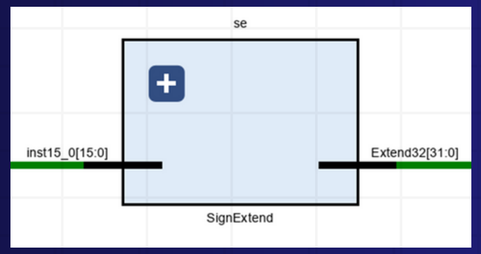

PC



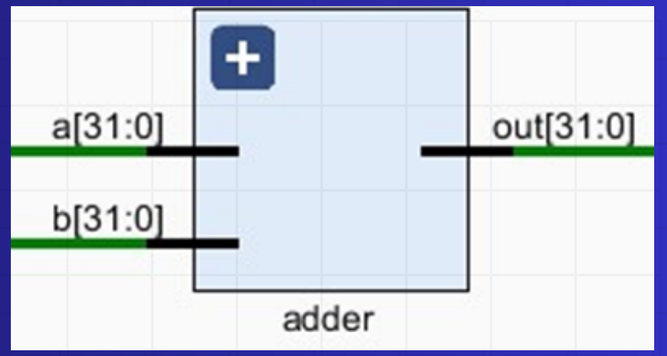

Shift Left 2



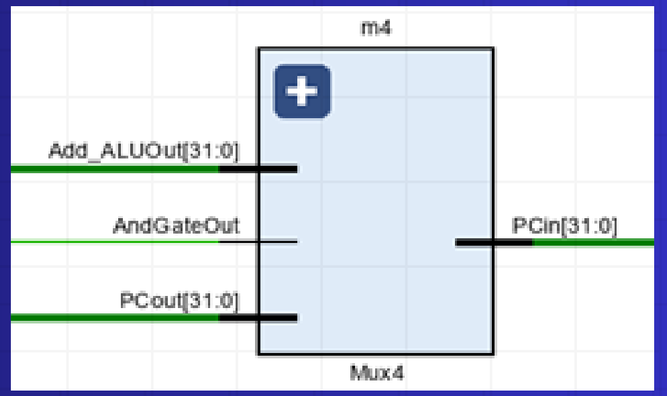

Register File



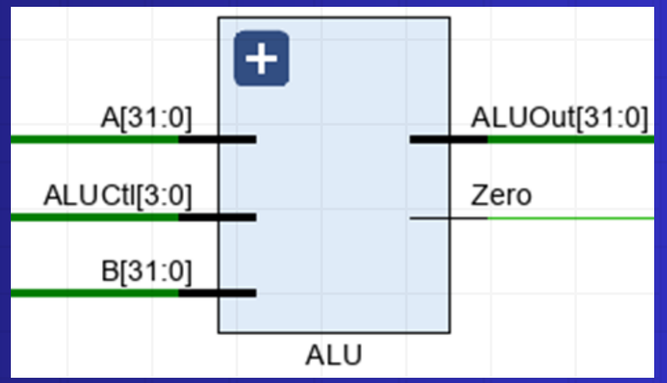

Sign Extend




Adder



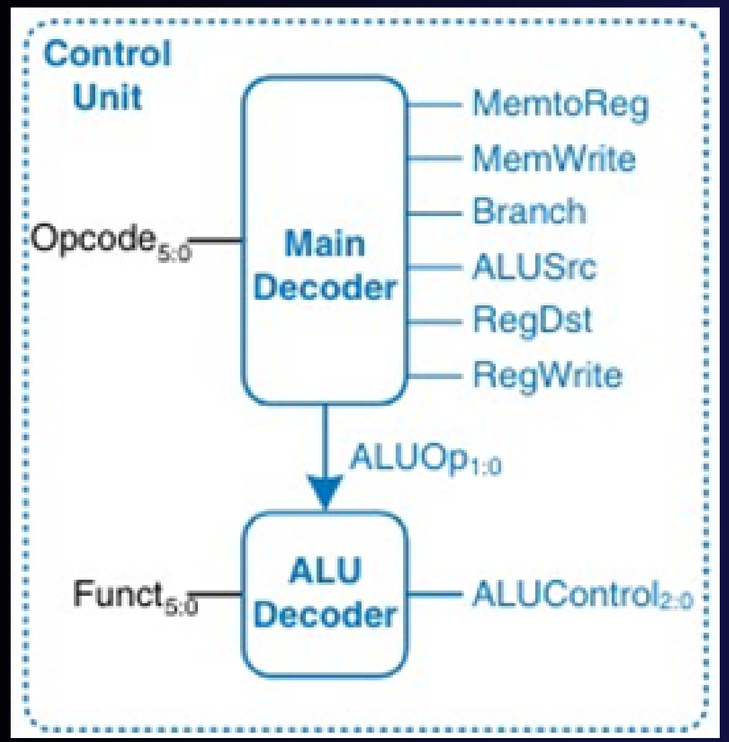

Multiplexer




ALU

# Phase 1 – Standard MIPS CPU Implementation

## 2. Control Unit:




MIPS Control Unit

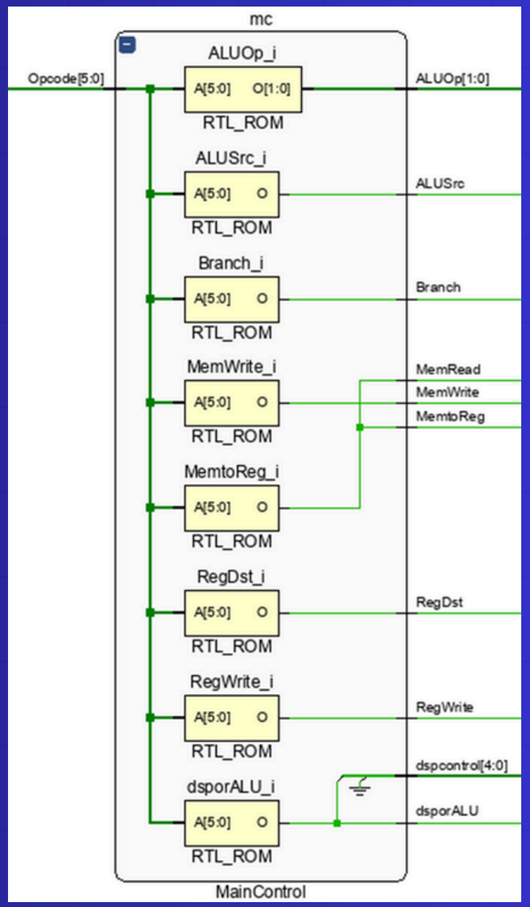
| Instructio | Opcode | RegWrite | RegDst | ALUSrc | Branch | MemWrit | MemtoRe | ALUOp | Jump |
|---|---|---|---|---|---|---|---|---|---|
| R-type | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 10 | 0 |
| lw | 100011 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| sw | 101011 | 0 | X | 1 | 0 | 1 | X | 0 | 0 |
| beq | 100 | 0 | X | 0 | 1 | 0 | X | 1 | 0 |
| addi | 1000 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| j | 10 | 0 | X | X | X | 0 | X | XX | 1 |

Main Decoder's Truth Table

| Op | Funct | ALUControl |
|---|---|---|
| 0 | XXXXXX | 010 (add) |
| 1 | XXXXXX | 110 (sub) |
| 1X | 100000 | 010 (add) |
| 1X | 100010 | 110 (sub) |
| 1X | 100100 | 000 (and) |
| 1X | 100101 | 001 (or) |
| 1X | 101010 | 111 (slt) |

ALU Decoder

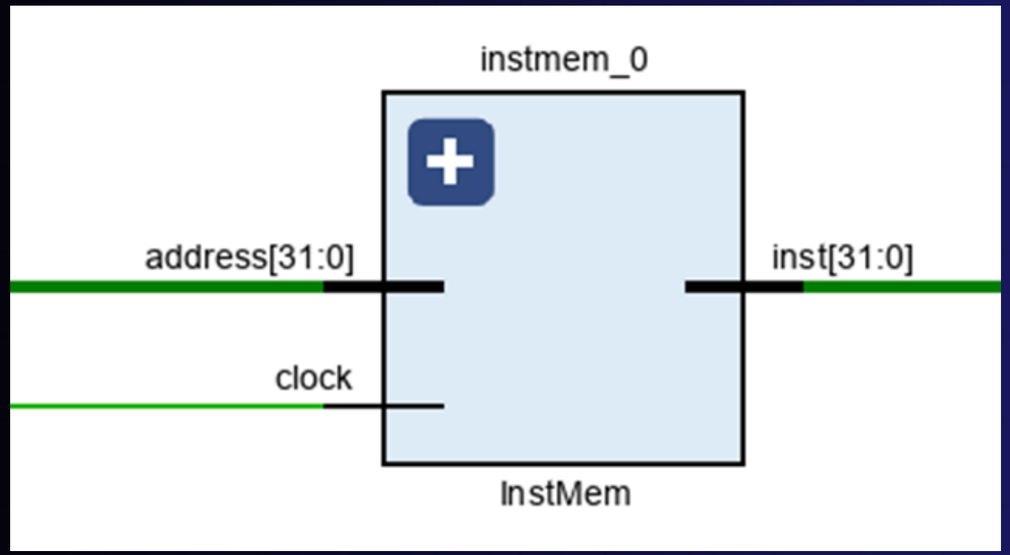
After implementing the main decoder and the ALU decoder



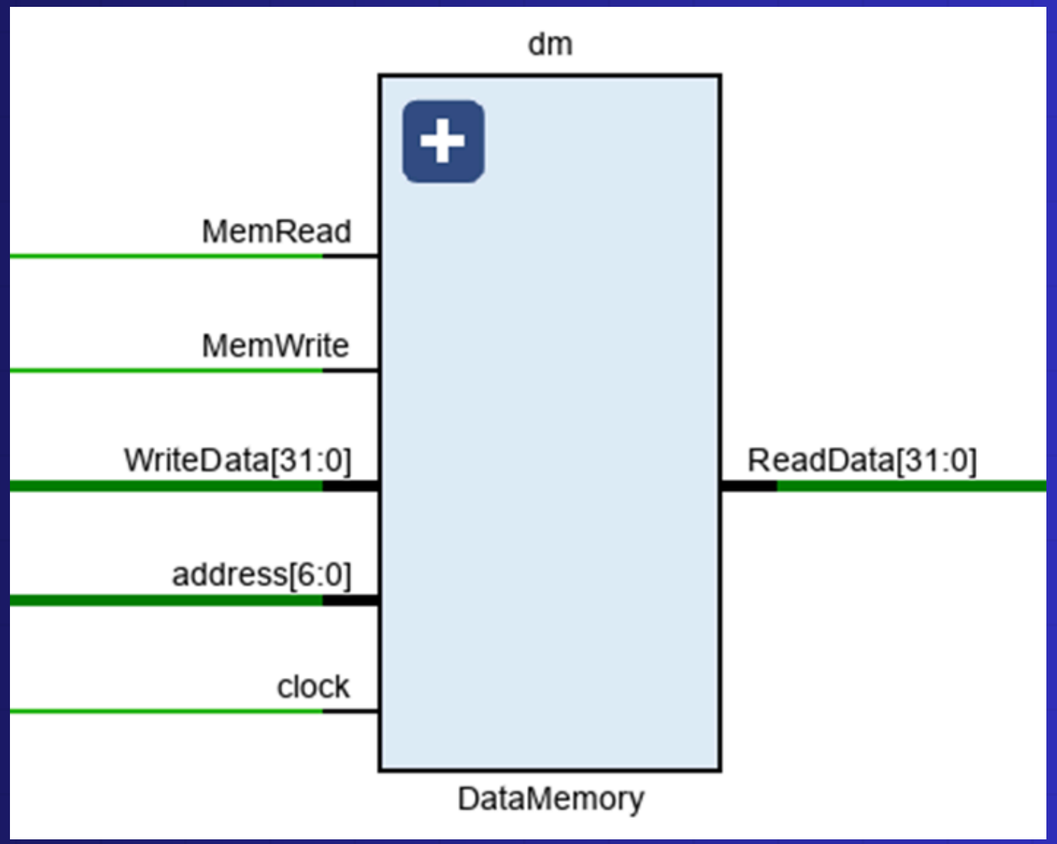

Control Unit

# Phase 1 – Standard MIPS CPU Implementation

## 3. External Memory:




Instruction Memory




Data Memory

# Phase 2 – DSP Unit Implementation

To implement a 3rd-order moving average FIR filter inside the DSP unit to smooth noisy signals.

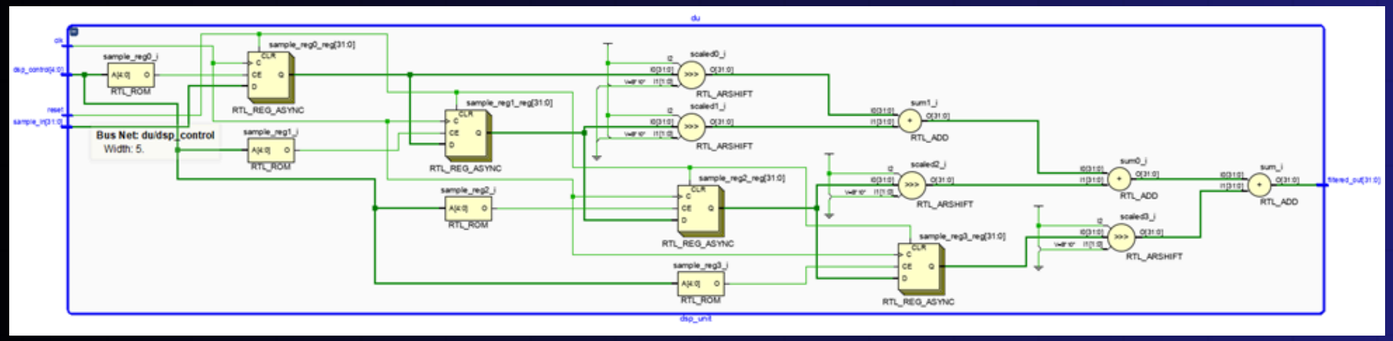
**How It Works:**

For each new input sample x[n], the filter averages it with the previous 3 samples:

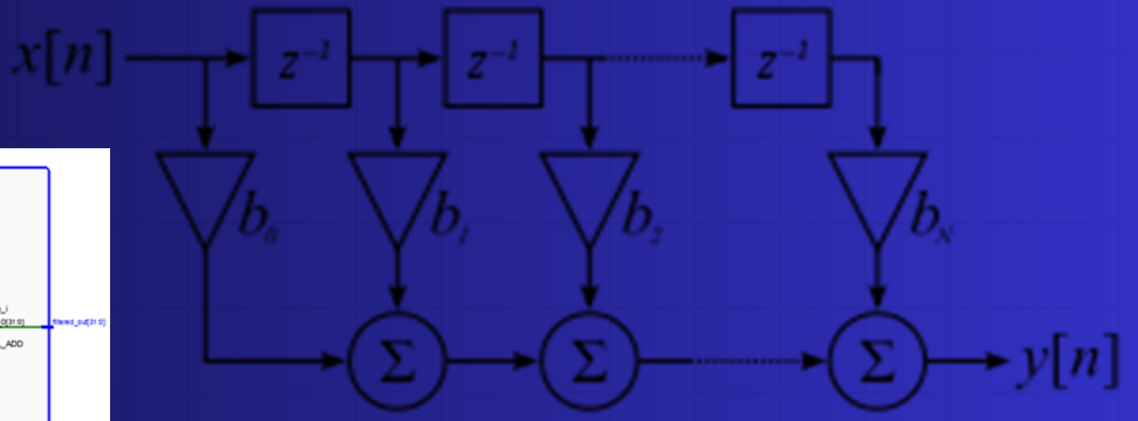
y[n] = (x[n]+x[n−1]+x[n−2]+x[n−3])/4

For simplicity, coefficients are all equal and fixed at:

$b$ = 1/N = 1/4= 0.25




Moving Average FIR Filter RTL Schematic




General Diagram for FIR Filters

# Phase 3 – Modified Architecture with DSP Unit

## 1. Modified R-type instruction format:

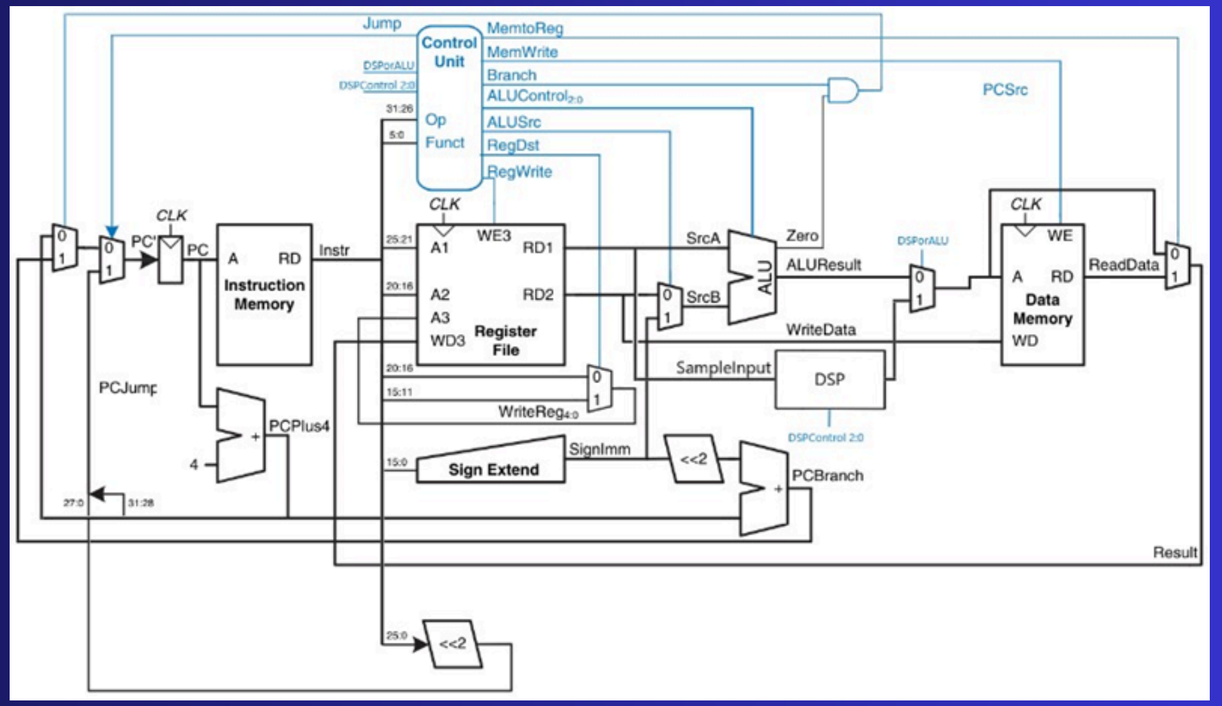
Old R-type Format:

| opcode | rs | rt | rd | shamt | funct |
|---|---|---|---|---|---|
| 6-bits | 5-bits | 5-bits | 5-bits | 5-bits | 6-bits |

DSP format based on R-type

| opcode | rs | rt | rd | dsp | funct |
|---|---|---|---|---|---|
| 6-bits | 5-bits | 5-bits | 5-bits | 5-bits | 6-bits |

## 2. Modified Datapath:




## 3. Modified Control Unit:

| Instruct | Opcode | RegWrit | RegDst | ALUSrc | Branch | DSPorA | MemWr | Memtol | Op | Jump |
|---|---|---|---|---|---|---|---|---|---|---|
| R-type | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 10 | 0 |
| lw | 100011 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| sw | 101011 | 0 | X | 1 | 0 | 0 | 1 | X | 0 | 0 |
| beq | 100 | 0 | X | 0 | 1 | 0 | 0 | X | 1 | 0 |
| addi | 1000 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| j | 10 | 0 | X | X | X | X | 0 | X | XX | 1 |
| dsp | 1 | 1 | 1 | X | 0 | 1 | 0 | 0 | 11 | 0 |

Modified Main Decoder Truth Table

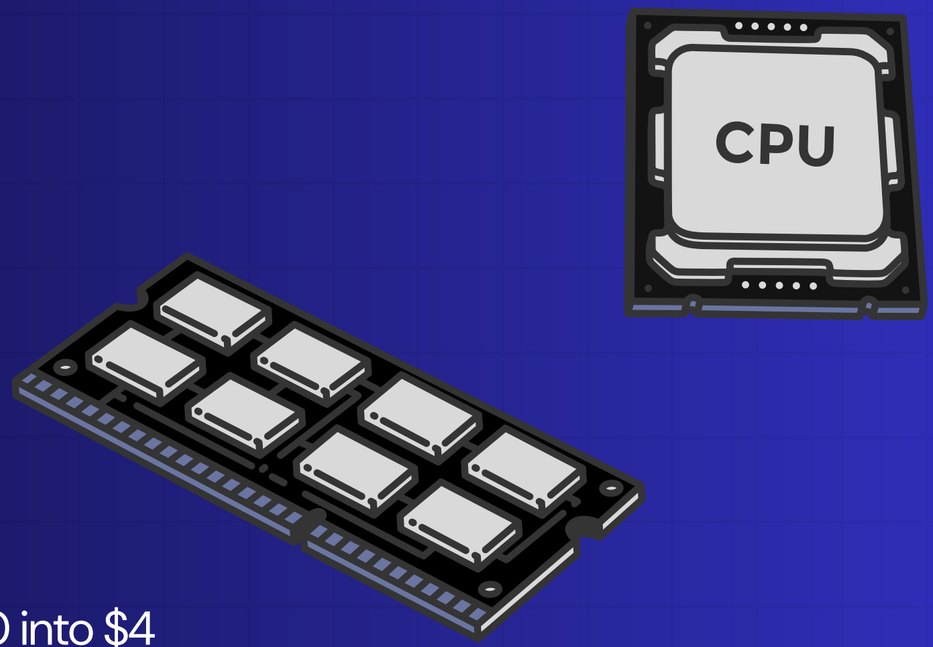
| Op | Funct | ALUControl | DSP | DSPControl |
|---|---|---|---|---|
| 0 | XXXXXX | 010 (add) | XXXXX | XXX |
| 1 | XXXXXX | 110 (sub) | XXXXX | XXX |
| 10 | 100000 | 010 (add) | XXXXX | XXX |
| 10 | 100010 | 110 (sub) | XXXXX | XXX |
| 10 | 100100 | 000 (and) | XXXXX | XXX |
| 10 | 100101 | 001 (or) | XXXXX | XXX |
| 10 | 101010 | 111 (slt) | XXXXX | XXX |
| 11 | XXXXXX | XXX | 1 | 0 |

Modified Secondary Decoder Truth Table

# Testing:

## Testbench Setup:

- Initialized registers with 32 noisy samples
- DSP instruction called for each sample
- Results stored back into registers




## Code:

20000000 // ADDI $0, $0, 0

8C040000 // LW $4, 0($0) – Load value from memory address 0 into $4

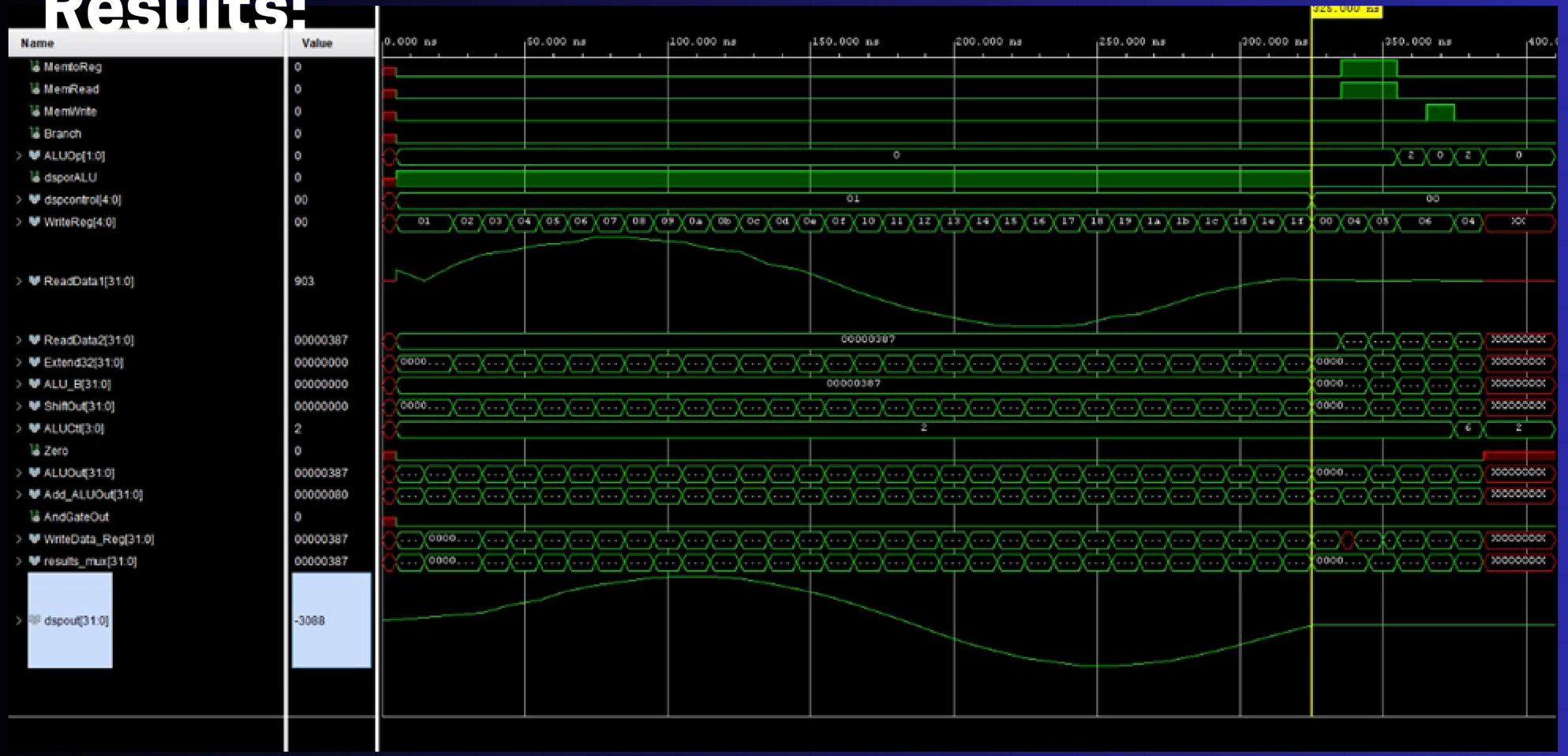
8C050004 // LW $5, 4($0) – Load value from memory address 4 into $5

00853020 // ADD $6, $4, $5 – Add $4 and $5, store in $6

AC060008 // SW $6, 8($0) – Store $6 into memory address 8

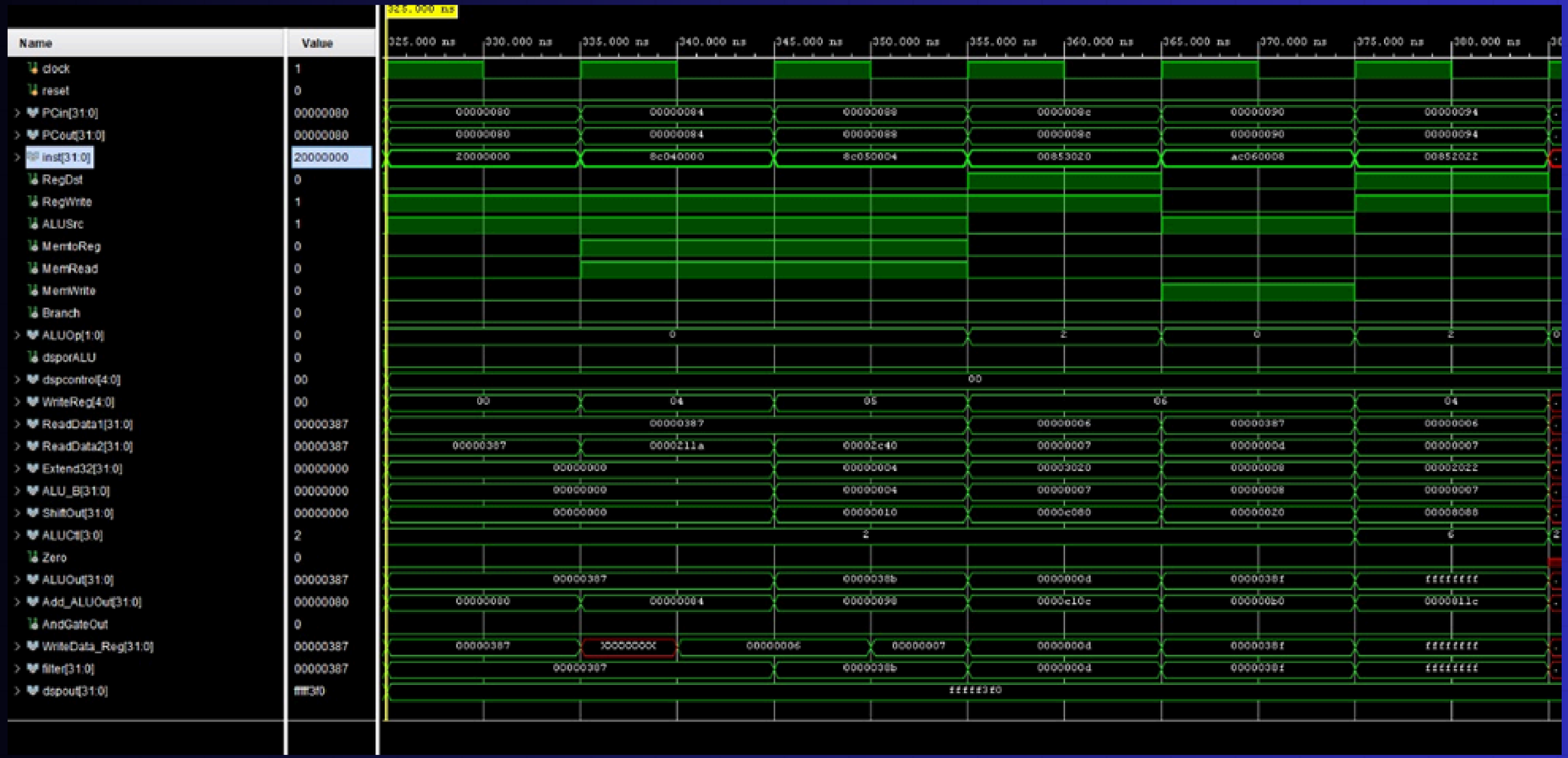
00852022 // SUB $7, $4, $5 – Subtract $5 from $4, store in $7

# Results:




Waveform showing successful the DSP unit

# Results:




Waveform for multiple R-type I-type Instructions

# Results:

**Design Timing Summary**

| Setup | | Hold | | Pulse Width | |
|---|---|---|---|---|---|
| Worst Negative Slack (WNS): | 2.841 ns | Worst Hold Slack (WHS): | 0.052 ns | Worst Pulse Width Slack (WPWS): | 8.750 ns |
| Total Negative Slack (TNS): | 0.000 ns | Total Hold Slack (THS): | 0.000 ns | Total Pulse Width Negative Slack (TPWS): | 0.000 ns |
| Number of Failing Endpoints: | 0 | Number of Failing Endpoints: | 0 | Number of Failing Endpoints: | 0 |
| Total Number of Endpoints: | 1014 | Total Number of Endpoints: | 1014 | Total Number of Endpoints: | 265 |

**All user specified timing constraints are met.**

Synthesis timing report

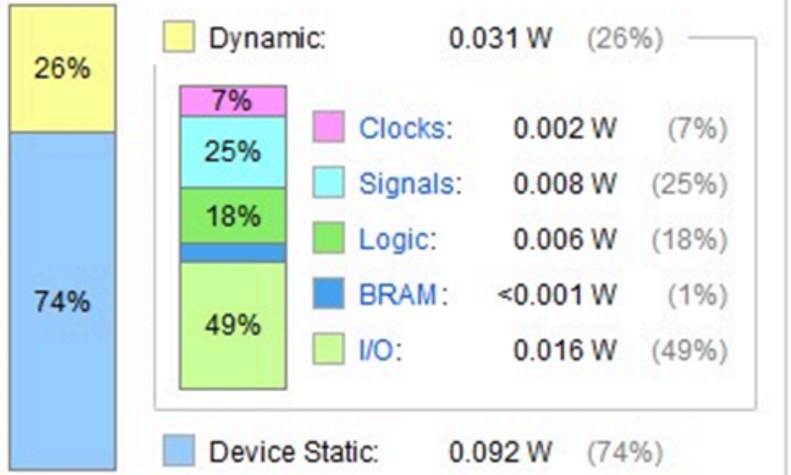
**Summary**

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

| | |
|---|---|
| **Total On-Chip Power:** | **0.123 W** |
| **Design Power Budget:** | **Not Specified** |
| **Power Budget Margin:** | **N/A** |
| **Junction Temperature:** | **26.4°C** |
| Thermal Margin: | 58.6°C (5.0 W) |
| Effective ϑJA: | 11.5°C/W |
| Power supplied to off-chip devices: | 0 W |
| Confidence level: | Medium |

Launch Power Constraint Advisor to find and fix invalid switching activity

**On-Chip Power**




Synthesis power report

# THANK YOU