RISC V Architecture

Mahesh Awati
Department of Electronics and Communication Engg.

RISC V ARCHITECTURE

UNIT 2 – Instructions: The Language of Computer

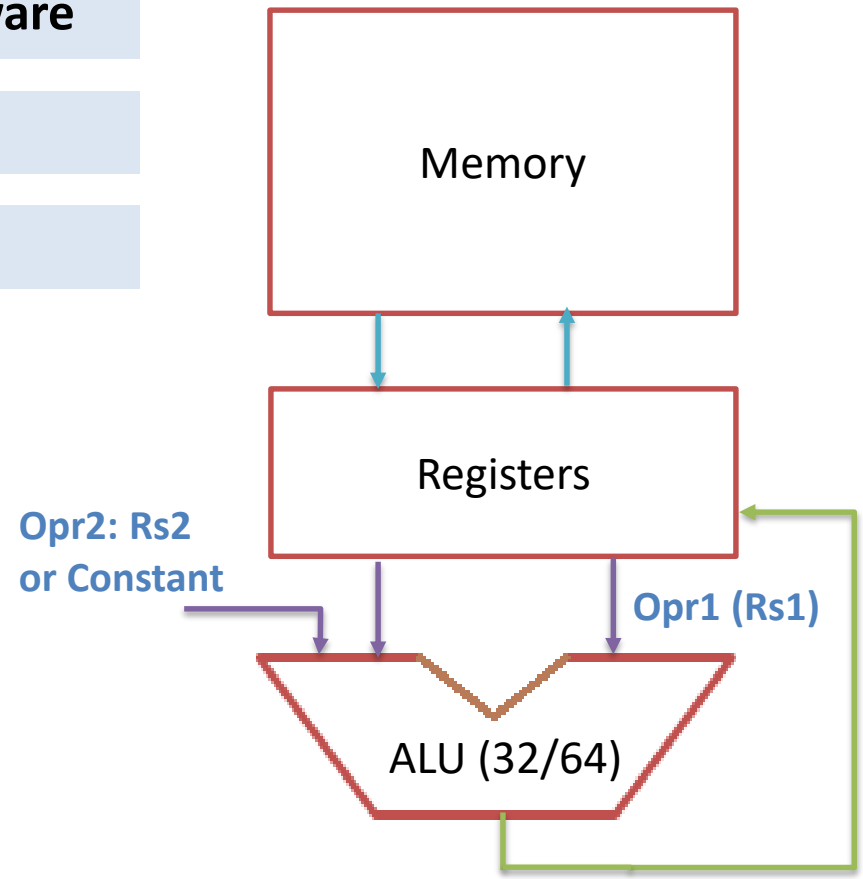
Mahesh Awati
Department of Electronics and Communication Engineering

Instructions – Language of Computer

Operands of the Computer Hardware

Operand location: physical location in computer

1. Registers – Special Location built directly in Hardware
2. Memory Operands
3. Constants (also called immediate)



Instructions – Language of Computer

Operands of the Computer Hardware

1. Registers – Special Location built directly in Hardware

- Programming languages have **simple variables that contain single/few** data elements
- Operands in arithmetic instructions must be **limited number** of **special locations built directly in hardware called registers.**
- Registers are faster than memory

In RISC V

- The size of a register in the RISC-V architecture is 64 bits; known as double-word or 32 bit size known as word.
- Variables in programming Language can be of any number but **registers in a hardware are always limited**.
- Ex: In RISC V 64 bit processor – There are 32 , 64 bit registers.
- Ex: In RISC V 32 bit processor – There are 32 , 32 bit registers.

Instructions – Language of Computer

Operands of the Computer Hardware

Why there is a limit on number of registers ?

Design Principle 2: "Smaller is faster".

1) A very large number of registers may **increase the clock cycle time simply because it takes electronic signals longer time when they must travel farther.**
2) **The number of bits it would take in the instruction format to address a register.**
   Example: If number of registers is 32, then the instruction format will have 5 bit field to address the registers. i.e., if there are 3 operands then the Instruction format will have 3, 5 bit fields reserved.
   Increasing Number of registers means increase in bits required by these fields. This intern will increase size of Instructions

Design Principle 1: Simplicity favours Regularity.

Design Principle 2: "Smaller is faster".

R type Instruction Format

The RISC-V convention is **x followed by the number of the register**, except for a few register names that we will cover later

Instructions – Language of Computer

Operands of the Computer Hardware

RISC V Registers

Name	Register Number	Usage
zero	x0	Constant value 0
ra	x1	Return address
sp	x2	Stack pointer
gp	x3	Global pointer
tp	x4	Thread pointer
t0–2	x5-7	Temporaries
s0/fp	x8	Saved register / Frame pointer
s1	x9	Saved register
a0–1	x10-11	Function arguments / return values
a2–7	x12-17	Function arguments
s2–11	x18-27	Saved registers
t3–6	x28-31	Temporaries

- Registers can be used either name (i.e., ra, zero) or x0, x1, etc. Using name is preferred
- Registers used for specific purposes:
  - **Zero Register:** always holds the constant value 0.
  - **Saved Registers, s0-s11**: used to hold variables
  - **Temporary registers, t0-t6 :**used to hold intermediate values during a larger computation
  - **Function arguments/return values**
  - **Pointers** – Stack, Global, Thread

Instructions – Language of Computer

Operands of the Computer Hardware

Why there is a limit on number of registers ?

Design Principle 2: "Smaller is faster".

Compiling a Complex C Assignment into RISC-V

Example

A somewhat complicated statement contains the five variables `f`, `g`, `h`, `i`, and `j`:

```
f = (g + h) −(i + j);
```

What might a C compiler produce?

```
add x5, x20, x21
add x6, x22, x23
sub x19, x5, x6
```

Saved Registers	x8, x9, x18-x27
Temporary registers	x5-x7, x28-x31

Instructions – Language of Computer

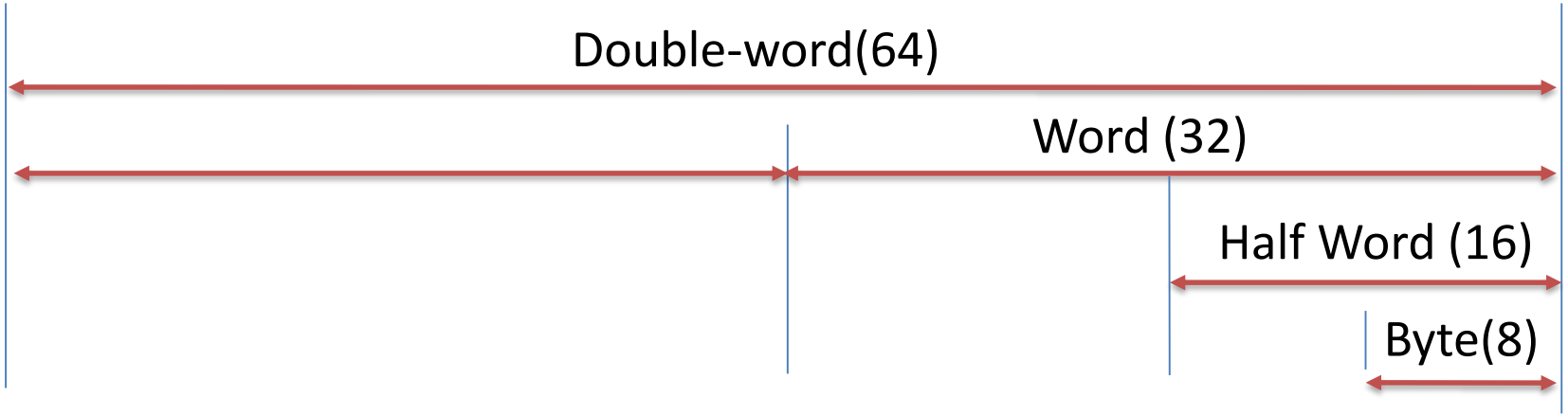
Operands of the Computer Hardware

Data types – Double word, Word, Half word and byte

0x0F	0x0E	0x0D	0x0C	0x0B	0x0A	0x09	0x08
15	14	13	12	11	10	9	8
63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0

← Address in Hexadecimal

← Address in decimal



THANK YOU

Mahesh Awati
Department of Electronics and Communication

mahesha@pes.edu

+91 9741172822