Lecture One
Instructions: Language of the Computer

CSE-2204: Computer Architecture and Organization

Course Teacher: Dr. Mosarrat Jahan
Associate Professor
CSE, DU.

MIPS

- MIPS (Microprocessor without Interlocked Pipelined Stages) is Reduced Instruction Set Computer (RISC) Instruction Set Architecture (ISA).
- MIPS was developed by MIPS Computer Systems (an American company now known as MIPS Technologies).
- It has the following characteristics:
  - Streamlined instruction.
  - Load/store architecture.
  - Single clock cycle execution.

Why MIPS?

- An elegant example of instruction set produced since 1980.
- Once learned, helps the designer to pick up other popular instruction sets:
  - ARM 7 (Advanced RISC Machine/Acron RISC Machine)
  - Intel x86
  - ARM8

Operations of the Computer Hardware

- MIPS arithmetic instruction performs only one operation and must always have exactly three variables.
- Example: add a, b, c
- Keeping exactly three operands for every instruction, makes the hardware very simple. Hardware for a variable number of operands is more complicated.
- Design Principle 1: ***Simplicity favors regularity.***

Example

High level instruction: f = (g+h) – (i+j)

MIPS instruction: add t0, g, h $\longrightarrow$ t0 = g + h

: add t1, i, j $\longrightarrow$ t1 = i + j

: sub f, t0, t1 $\longrightarrow$ f= t0 – t1

With MIPS register notation: add \$t0, \$s1, \$s2 $\longrightarrow$ \$t0 $\leftarrow$ g + h

: add \$t1, \$s3, \$s4 $\longrightarrow$ \$t1 $\leftarrow$ i + j

: sub \$s0, \$t0, \$t1 \$s0 $\leftarrow$ f

Operands of Computer Hardware

Two types: 1. Registers 2. Memory operands

- **Register:** Special locations built directly into the hardware.
- The size of registers: 32 bits (MIPS architecture)
- Word size: 32 bits (MIPS architecture)
- Three operands of MIPS instruction must each be chosen from one of the 32 registers.
- A very large number of registers may increase the clock cycle time simply because it takes electronic signals longer when they travel farther.
- Design Principle 2: ***Smaller is faster.***
- MIPS registers: $s0, $s1,… for variables in C and $t0, $t1, … for temporary variables.

Memory Operands

- ✓ Registers can contain few data values. The remaining data values and the complex data structures are kept in the memory.
- ✓ **Data Transfer Instruction:** Arithmetic operations are performed only on registers. So, data transfer instruction are needed to transfer data between memory and registers.
- ✓ Load Instruction: Used to fetch data from memory into a register.
- ✓ Format of Load Instruction:

lw	Dest Register	Constant/offset	Base Register

Memory address= constant + Base register

Example

1.High level instruction: g = h + A[8]

MIPS instruction:

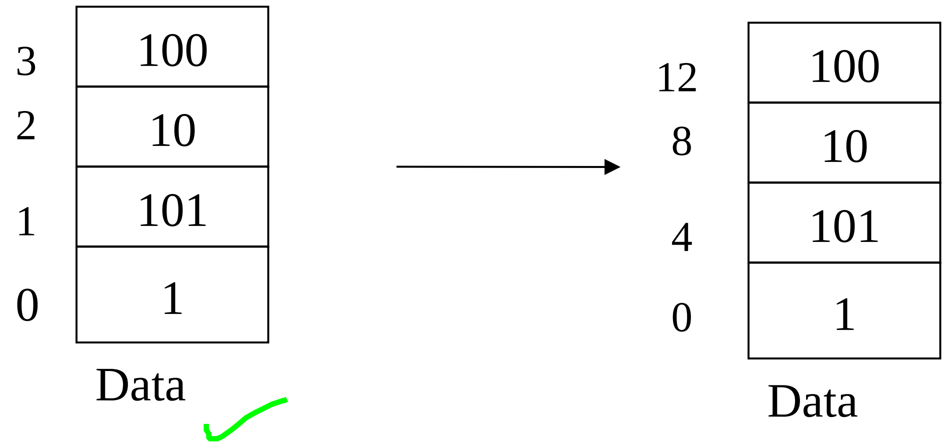
```
lw $t0, 8($s3)
add $s1, $s2, $t0
```

2. High level instruction: A [12] = h + A[8]

MIPS instruction:

```
lw $t0, 32 ($s3)
add $t0, $s2, $t0
sw $t0, 48($s3)
```

Memory Representation



- In MIPS, words must start at addresses that are multiples of 4. (Alignment restriction)
- MIPS follow Big Endian scheme.
- Store instruction: used to store data from register to memory location.
- Format of Store Instruction:
-

sw	Source Register	Constant	Base Register

Constant and Immediate Operand

- ✓ By adding constant operand in the instruction we can avoid load operation.
- ✓ Example: addi $s3, $s3, 4
- ✓ Constant operands occur frequently. By including constants inside arithmetic instruction, they are much faster than if constants were loaded from memory.
- ✓ Design Principle 3: ***Make the common case fast.***

MIPS Instruction Format

Op	rs	rt	rd	shamt	funct
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

Figure: R- type or R-format Instruction

- Data and instruction are represented as binary number in computer memory.
- 5 bits are required for register address because there are 32 registers.
- Mapping of register to number:

$s0------ $s7	16………….23
$t0------ $t7	8……………15

Representing Instruction in the Computer

Example: add $t0, $s1, $s2

Decimal Representation:

0	17	18	8	0	32

Binary Representation:

000000	1001	10010	01000	00000	100000
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

MIPS Instruction Format

Op	rs	rt	Constant or address
6 bits	5 bits	5 bits	16 bits

Figure: I- format or I- type instruction (used by immediate and data transfer instruction)

- rs=base register
- Design Principle 4: ***Good design demands good compromise.***
- We can reduce hardware complexity by keeping the formats similar.
- Formats are distinguished by values in the first field.

MIPS Instruction Encoding

Instruc -tion	**Format**	**Op**	**rs**	**rt**	**rd**	**shamt**	**funct**	**address**
add	R	0	reg	reg	reg	0	32	n.a
sub	R	0	reg	reg	reg	0	34	n.a
addi	I	8	reg	reg	n.a	n.a	n.a	constant
lw	I	35	reg	reg	n.a	n.a	n.a	address
sw	I	43	reg	reg	n.a	n.a	n.a	address

Example

A [300] = h + A [300]

MIPS instructions:

1. lw $t0, 1200($t1) # $t0 := A[300]
2. add $t0, $s2, $t0 #$t0 : = h + A[300]
3. sw $t0, 1200($t1) #A[300] : = A[300] + h

Decimal Representation:

op	**rs**	**rt**	**rd**	**address/ shamt**	**funct**
35	9	8	1200		
0	18	8	8	0	32
43	9	8	1200		

Chapter 2 of textbook.