Instructions: Data Transfer

load and store architecture

MIPS: can access memory only to transfer data to or from registers

CISC: may allow, for example, add to memory location, store in register

load: copy the data from memory into a register.

store: copy the data from a register to memory.

`lw  $rt, offset($rs)`	Load word from memory location to register
`sw  $rt, offset($rs)`	Store word from register to memory location

Offset is a 16-bit 2C value (immediate); all are I-type

semantics of `lw`

$\texttt{Addr <-- R[s] + }(\texttt{IR}_{15})^{16}\texttt{::IR}_{15\text{-}0}$

$\texttt{R[t] <-- M}_4\texttt{[Addr]}$

compute address

- add the contents of register s (base) to the sign-extended offset
- offset is immediate
- non-aligned address: hardware exception

get data

- copy 4 bytes located at memory address starting at `Addr` to register t.
- CPU fetches the four bytes based on the endianness of the machine

`sw` similar to `lw` but the 4 byte quantity is copied from the register to memory.

$\texttt{M}_4\texttt{[Addr] <-- R[t]}$ stored in the endianness of the machine

Instructions: Data Transfer

Byte operations

`lb  $rt, offset($rs)`	Load sign-extended byte from memory location to register
`lbu $rt, offset($rs)`	Load zero-extended (unsigned) byte from memory location to register
`sb  $rt, offset($rs)`	Store the least significant byte of a register to memory location

What about sbu?

For `lb`, the address is computed the same way as `lw`,

but the address does not have to be word aligned.

$\texttt{Addr <-- R[s] + } (\texttt{IR}_{15})^{16}\texttt{::IR}_{15\text{-}0}$

$\texttt{R[t] <-- } (\texttt{M}_1\texttt{[Addr]}_7)^{24}\texttt{::M}_1\texttt{[Addr]}$

Since the value is interpreted as 2C, the fetched byte is sign-extended to 32 bits.

`lbu` is just like `lb` except the byte is zero-extended in the register

sb is similar to `sw`:

$\texttt{M}_1\texttt{[Addr] <-- R[t]}_{7\text{-}0}$

The least significant byte of register t is copied to the address in memory.

Do we really need to have separate instructions to load, store bytes?

Instructions: Data Transfer

machine code

```
lw  $rt, offset($rs)
```

100101	01000	01001	00000 00000 100000
b_{31-26}	b_{25-21}	b_{20-16}	b_{15-0}
opcode	$rs	$rt	immediate

opcodes:	100 xxx (load)
	101 xxx (store)
$rs:	base address
$rt:	target register

immediate: offset

Instructions: Data Transfer

Halfword operations (short int)

`short int` is usually 16 bits

`lh $rt, offset($rs)`	Load halfword from memory location to register
	Data is sign-extended in register
`lhu $rt, offset($rs)`	Data is zero-extended in register
`sh $rt, offset($rs)`	Store halfword from register to memory location

Loading constant

`lui $rt, immed`

Semantics:

$R[t] = IR_{15-0}\ 0^{16}$

load the lower halfword of `immed` into upper halfword of `$rt`

lower bits of `$rt` are set to 0

`$rs` is ignored

Data transfer: summary

		R-type	I-type
Load	Word		`lw`
	Halfword		`lh`
	Halfword unsigned		`lhu`
	Byte		`lb`
	Byte unsigned		`lbu`
	Constant		`lui`
Store	Word		`sw`
	Halfword		`sh`
	Byte		`sb`

Instruction Types

Arithmetic	✓
Logical	✓
Data Transfer	✓
Compare/Branch	
Jump	