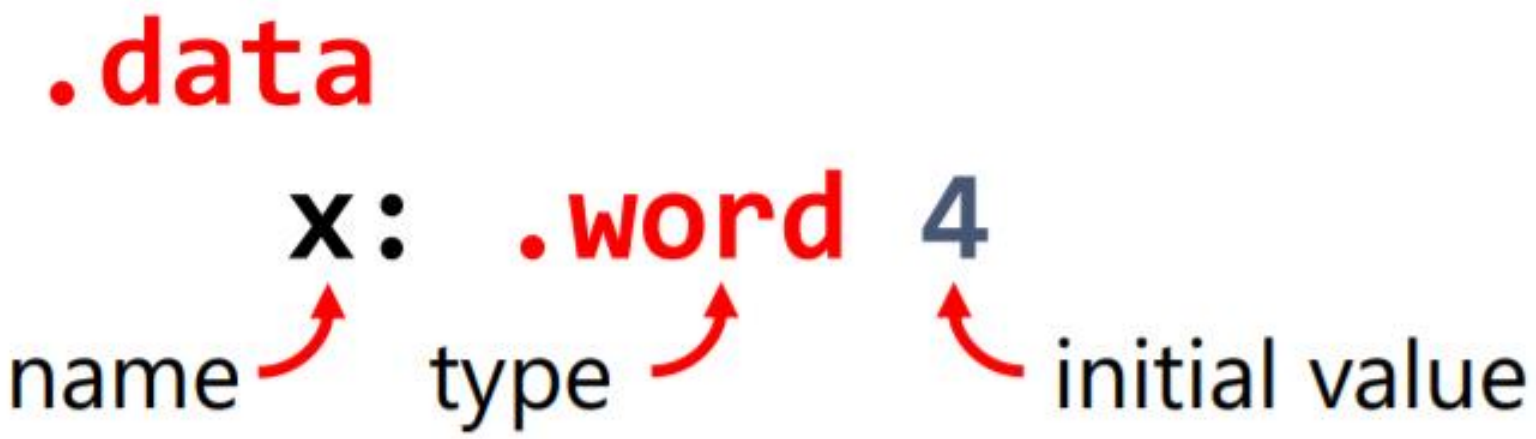
Lab 2

MIPS Variables



```
.data
    x: .word 4
```

Type of variable:

- word (4 Bytes)
- half (2 Bytes)
- byte (1 Byte)

Load and Store Instructions

- **Load** : Memory -> Register
- **Store**: Register -> Memory

Load

- lw
  - copy word (4 bytes) at source memory address to destination register

```
a:    .word 0
lw t0, a
```

  - lw can be replaced with lb or lh depending on the type of variable that is being loaded
    - lb or lbu if variable is of type *byte* (1 Byte)
    - lh or lhu if variable is of type *half* (or half word = 2 Bytes)

EXPAND VALUE

- if you load a **byte...**

10010000

31			0
00000000	00000000	00000000	00000000

If the byte is **signed...** what *should* it become?

31			0
11111111	11111111	11111111	10010000

`lb` does sign extension.

If the byte is **unsigned...** what *should* it become?

31			0
00000000	00000000	00000000	10010000

`lbu` does zero extension.

Load address

- `la t0, var1`
  - Copy memory address of var1 (a variable defined in the program) into register t0

- Indirect Addressing
  - `lw t2, (t0)`
    - load word at memory address contained in t0 into t2
  - `sw t2, (t0)`
    - store word in register t2 into memory at address contained in t0

Exercise

- If memory address of variable x is 0x10010001,

```
la t0, x

lw t1, 16(t0)

lw t1, -12(t0)
```

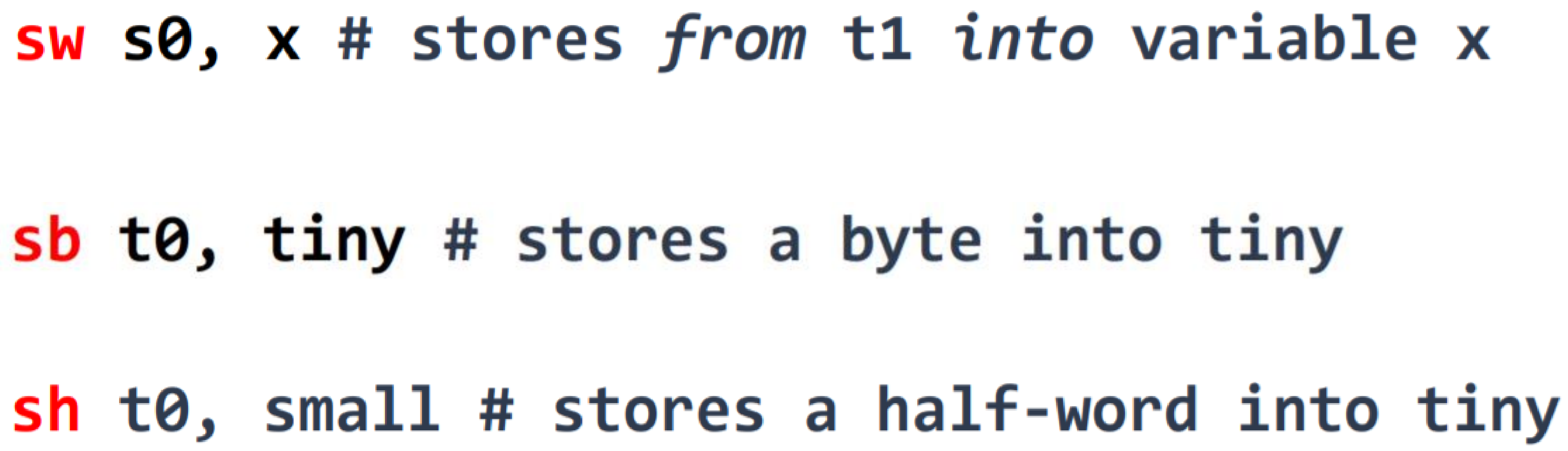
t1 stores the value at which memory address?

Effective address = value of `register_address + offset`

Load Immediate

- `li  register_destination, value`
    - Loads immediate value in destination register

Store Instructions



```
sw s0, x # stores from t1 into variable x

sb t0, tiny # stores a byte into tiny

sh t0, small # stores a half-word into tiny
```

https://luisfnqoliveira.gitlab.io/cs447_s2021/slides/05-Memory-and-Addresses.pdf

Lab Overview