10/4: Lecture Topics

- Instruction encoding
  - slides from 10/1
- Addressing modes
- High-level concepts in procedure call

Addressing Modes

- Each ISA specifies a number of *addressing modes*
- MIPS supports very few:
  - register mode
  - displacement mode
  - immediate mode
  - PC-relative mode
  - Pseduodirect mode

Register Mode

- The operand is a register
- Takes 5 bits
- Example: `add`

```
add $s0, $s1, $s2
```

- Nearly all instructions use at least one register

Displacement Mode

- The operand is at a memory location
- The location is the sum of a register and a constant
- Takes 16+5 = 21 bits
- Example: `lw`

```
lw $t0, 0($t1)
```

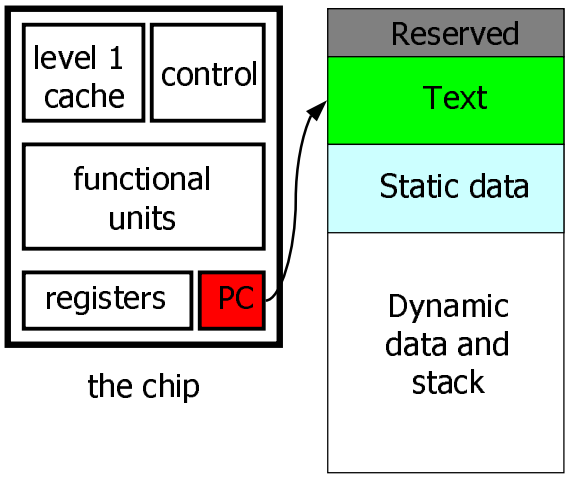
Immediate Mode

- The operand is a constant within the instruction
- Takes 16 bits
- Example: `addi`

```
addi $t0, $t0, 2
```

The Program Counter

- PC = program counter
- Keeps track of the next instruction to execute



PC-Relative Mode

- The operand is at a memory location
- The address is the sum of the program counter (PC) and a constant in the instruction
- Takes 16 bits
- Example: bne

```
bne $t0, $t1, loop
```

Pseudodirect Mode

- The operand is at a memory location
- The address is the upper bits of the PC concatenated with the jump address
- Takes 26 bits
- Example: j

```
j  L0
```

Pseudodirect Mode, cont.

- Current PC:

  `0x0684b40c`

- Jump address:

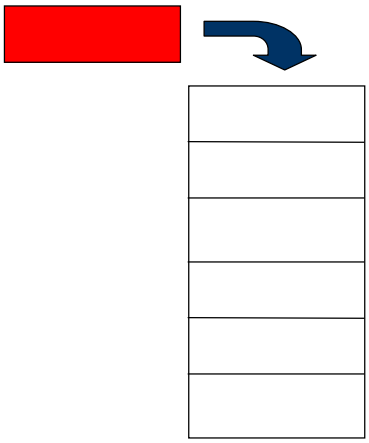
  `0x3497808`

Procedure Call

- More than just a branch and a return
- Data goes in
  - *arguments, parameters*
- Data goes back out
  - *return value*
- What makes this possible?
  - the stack

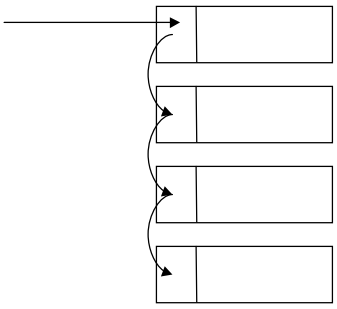
Review of Stacks

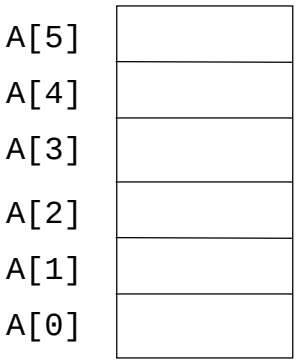
- Two operations:
  - *push* an item onto the stack
  - *pop* an item off the stack



Stack Implementations

- Pretty easy to do with a linked list
  - You probably saw this in 373



A Stack in an Array

- Linked lists are nice if you have them
- Arrays are a lot faster