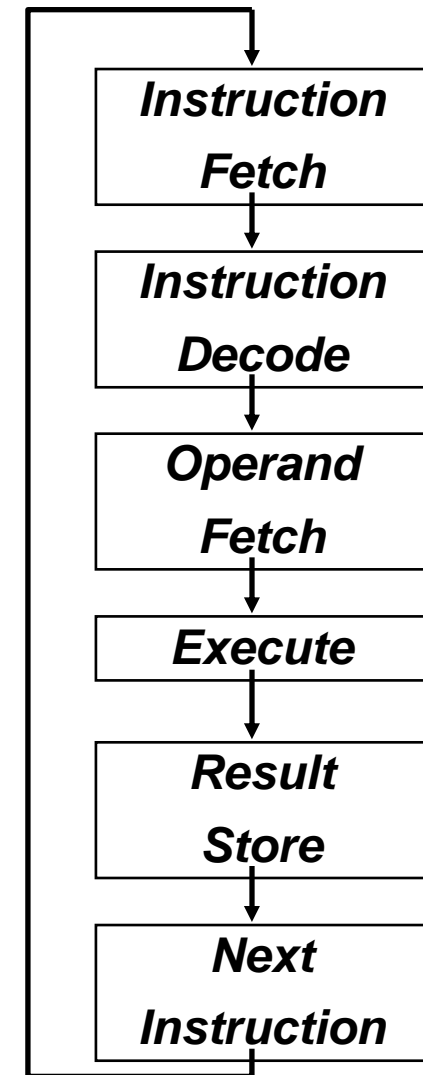
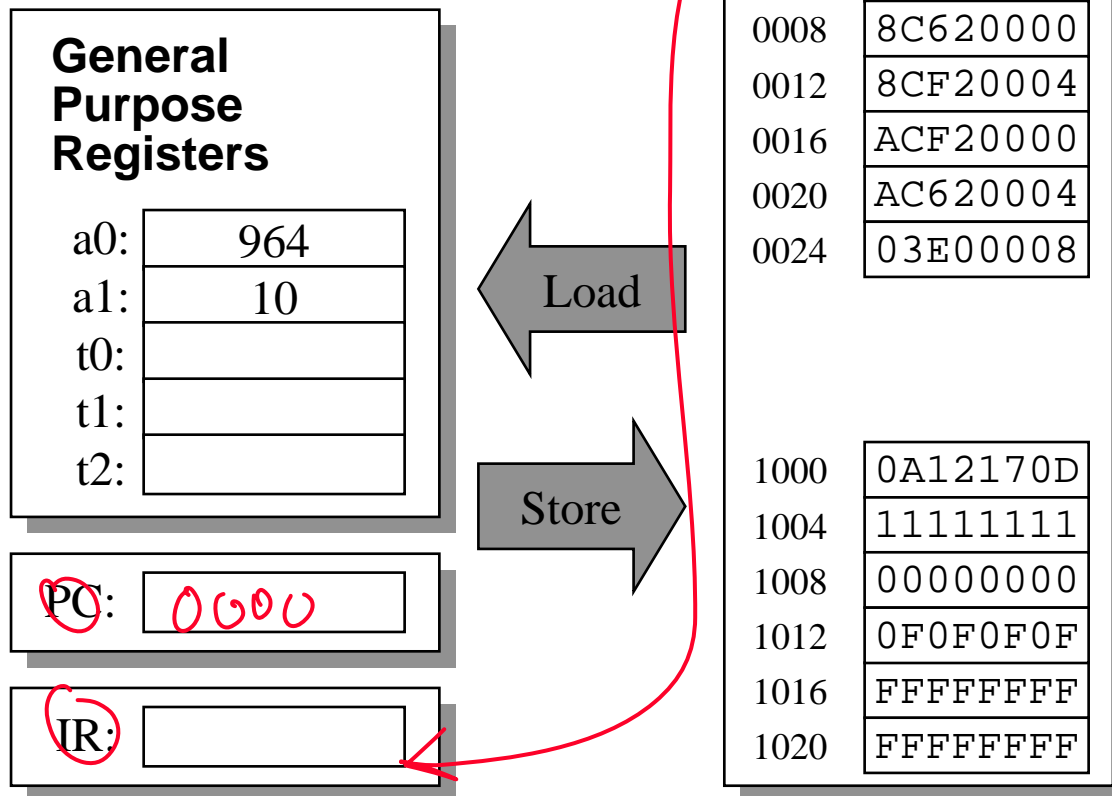


Execution Cycle Example

PC: Program Counter ←

IR: Instruction Register →



Control Flow

Jumps – GOTO different next instruction

```
j 25          # go to 100: PC = 25*4 (instructions are 32-bit)
jr $ra        # go to address in $ra: PC = value of $ra
```

Branches – GOTO different next instruction if condition is true

2 register: beq (==), bne (!=)

```
beq $t0, $t1, FOO      # if $t0 == $t1 GOTO FOO: PC = FOO
```

1 register: bgez (>=0), bgtz (>0), blez (<=0), bltz (<0)

```
bgez $t0, FOO          # if $t0 >= 0 GOTO FOO: PC = FOO
```

```
if (a == b)           # $a0 = a, $a1 = b, $a2 = c
    a = a + 3;         →      bne    $a0, $a1, ELSEIF      # branch if a!=b
else                  →      addi    $a0, $a0, 3;          # a = a + 3
    b = b + 7;         →      j DONE;                    # avoid else
c = a + b;            →      ELSEIF:
                        addi    $a1, $a1, 7;              # b = b + 7
                        →      DONE:
                        add     $a2, $a0, $a1;            # c = a + b
```

Loop Example

Compute the sum of the values 1...N-1

```
int sum = 0;
for (int I = 0; I != N; I++) {
    sum += I;
}
```

add \$t1, \$zero, \$zero
add \$t2, \$2, \$2

beq \$t0, \$t2, END

: Loop:

↓

j Loop

—

: END:

↓
\$t0 = N, \$t1 = sum, \$t2 = I

add \$t1, \$zero, \$zero

add \$t2, \$zero, \$zero

add \$t1, \$t1, \$t2 ←

addi \$t2, \$t2, 1

beq \$t2, \$t0, END

j Loop

bne \$t0, \$t2, Loop

Comparison Operators

For logic, want to set a register TRUE (1) / FALSE(0) based on condition

```
slt $t0, $t1, $t2    # if ($t1 < $t2) $t0 = 1 else $t0 = 0;
```

if (a >= b)
 c = a + b;
a = a + c;

Handwritten notes:
- Red checkmark and question mark next to the C code.
- Red arrow from the C code to the MIPS code.
- Red text: `addi $t4, $zero, 1` *a >= b*
- Red text: *a < b*
- Red text: `# $t0 = a, $t1 = b, $t2 = c`
- Red text: `slt $t3, $t0, $t1` *// (a < b)?*
- Green text: `bne $t3, $zero, Foo`
- Red text: `beg $t3, $t4, Foo`
- Red text: `add $t2, $t0, $t1` *// c = a + b*
- Red text: `add $t0, $t0, $t2` *// a = a + c*
- Red text: *Foo:*