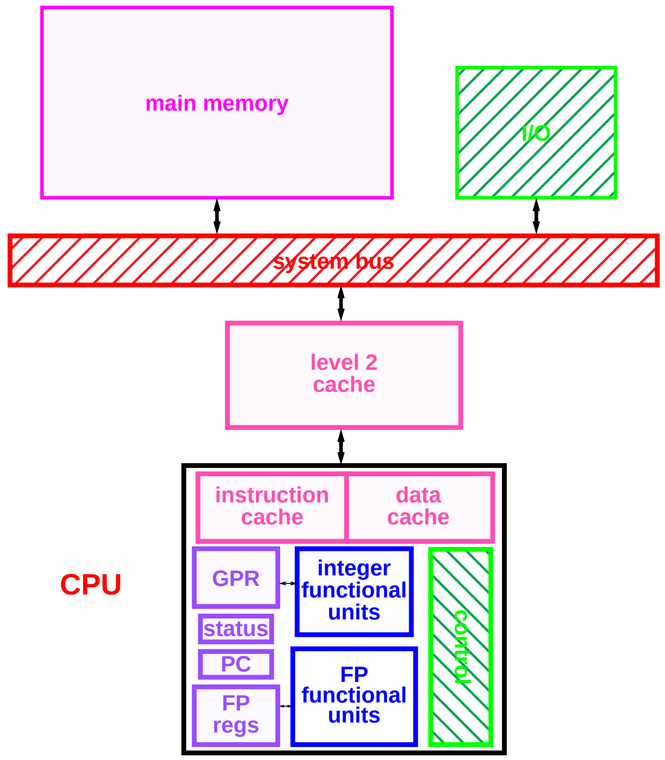
A Simplified Machine Model



MIPS is a RISC Architecture

RISC = **R**educed **I**nstruction **S**et **C**omputer

- simple instructions
- arithmetic instructions do some operation on 2 source registers & put the results in a destination register
    - $\mathtt{reg}_c = \mathtt{reg}_b \ \mathtt{op} \ \mathtt{reg}_a$
- separate data transfer instructions
    - **load** from memory into $\mathtt{reg}_a$ & $\mathtt{reg}_b$
    - **store** from $\mathtt{reg}_c$ to memory
- called a **load/store architecture**

Instructions

An instruction tells the CPU

- the **operation** to be performed: **opcode**
- the **operands** (0 or more) on which to perform the operation

The **instruction set architecture** (**ISA**) specifies:

- what the opcode means
- how many operands it requires
- what is the type of the operands

An operand can be a:

- register
- memory address
- constant

Registers

High-speed storage for instruction operands

One set for integer values; one set for floating point values

Naming convention:

- named by their location in the register file & use
  for example, **$s7, $f14**

General purpose registers (**GPRs**)

- all registers are alike: any value can be in any register
- not quite true
  - $0 is hardwired to 0
  - some have a special use by software convention *(MIPS details later; see p. A-22)*

Usually 32 registers in the ISA (of each type: integer, FP)

- seems a good trade-off between having enough high-speed storage for “live” values and a short register file access time
- only takes 5 bits to represent a register in an instruction
- there are other registers, but they are hidden from the programmer

Information Units

Basic unit is a **bit** (stores a 0 or 1)

Bits are grouped to form other information units

- **byte** = 8 bits
- **word** = 32 bits
- **double word** = 64 bits

Word size is “the size of the machine”, e.g., a 32-bit processor has a 32-bit word size

- determines several characteristics of the architecture or implementation
  - size of the registers = “word size” bits
  - number of integer values that can be stored is $2^{\text{word size}}$
  - size of addressable memory (memory locations from 0 to $2^{\text{word size}}$-1)

Memory

Memory is an array of **information units** (the logical organization)

- each unit is the same size
- each unit has a unique address
- address & contents are ***different***

address	content
0	123
1	-17
2	
3	
$2^{\text{wordsize}}-1$	

Memory

Most computers have **byte addressable** memory

- each address is a byte location
- on a 32-bit processor can address 2^{32} different values
  → 4G**B** address space
- when address words, they must be on 4-byte boundaries
- today 64-bit architectures are being built (e.g., Alpha, UltraSPARC, R12000, Pentium Pro)

Memory Hierarchy

The memory system is a hierarchy of “memories”

the closer they are to the CPU, the smaller, faster, more expensive they are

The **hierarchy**:

- registers
- caches (one or more levels)
- main memory (primary memory)
- disks (secondary memory)

Memory Level	Capacity (bytes)	Speed
Registers	10’s - low 100’s	< nanosecond
Cache: level 1	8 - 128KB on-chip	~2 nanoseconds
Cache: level 2	1 - 16MB off-chip	10’s of nanoseconds
Memory	10MB - 2GB	10’s to 100’s of nanoseconds
Disk	1GB - 10GB	10’s of milliseconds

Execution Cycle

The CPU executes a program by repeating this cycle:

1. fetch an instruction
2. execute the instruction
3. compute the address of the next instruction
4. go back to 1. until you run out of instructions

Now we’re ready to dive in!