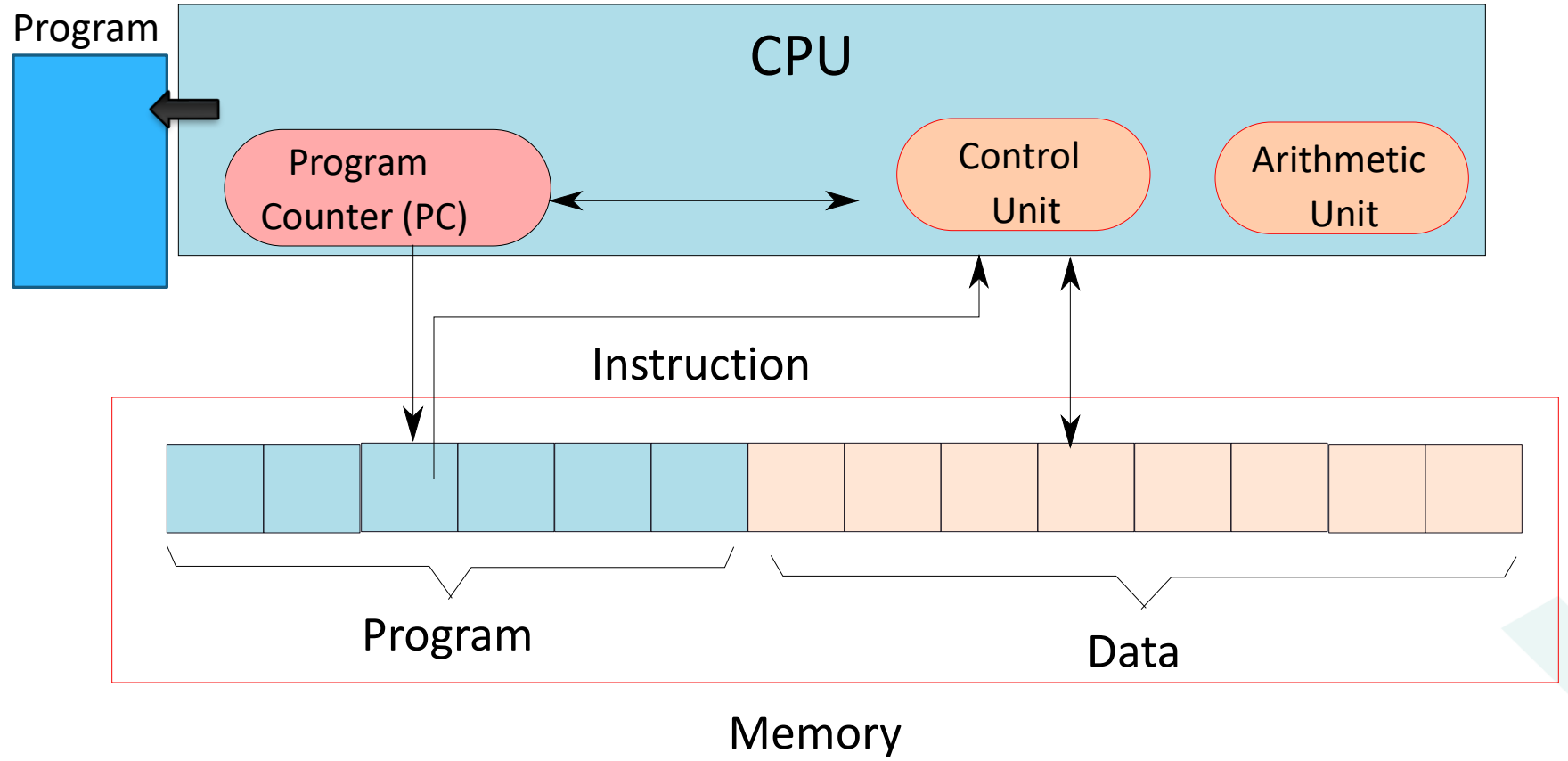
CSE 112: Computer Organization

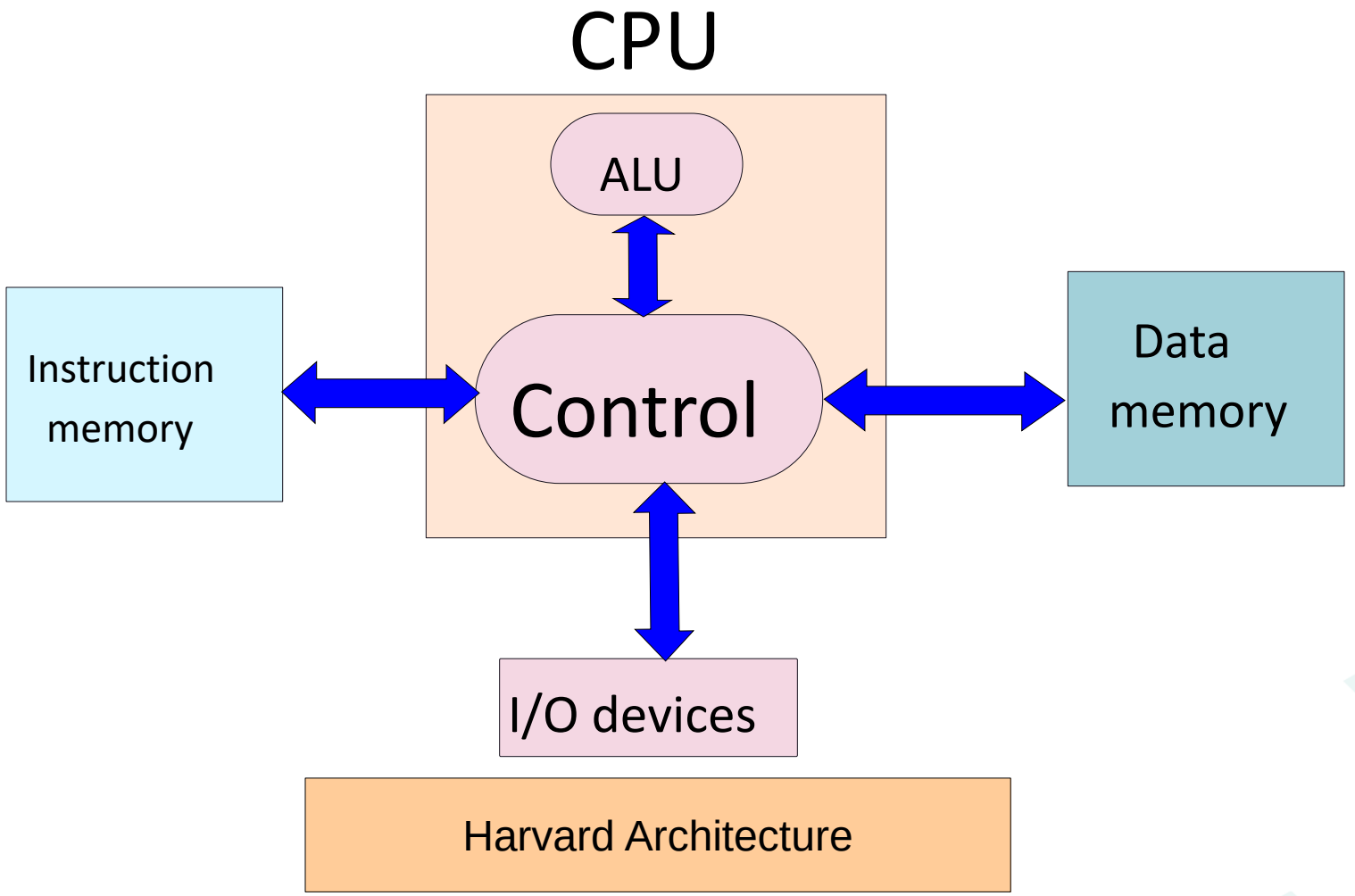
Instructor: Sujay Deb

Lecture 6

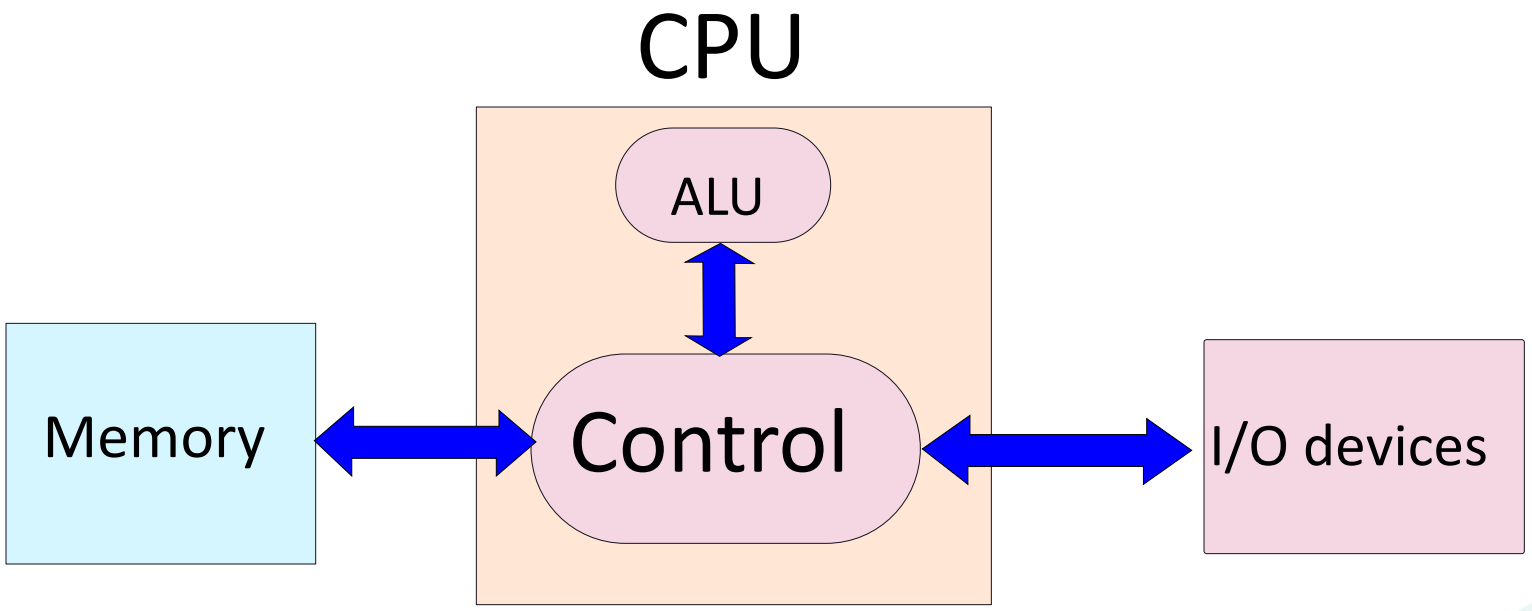
Elements of a Computer



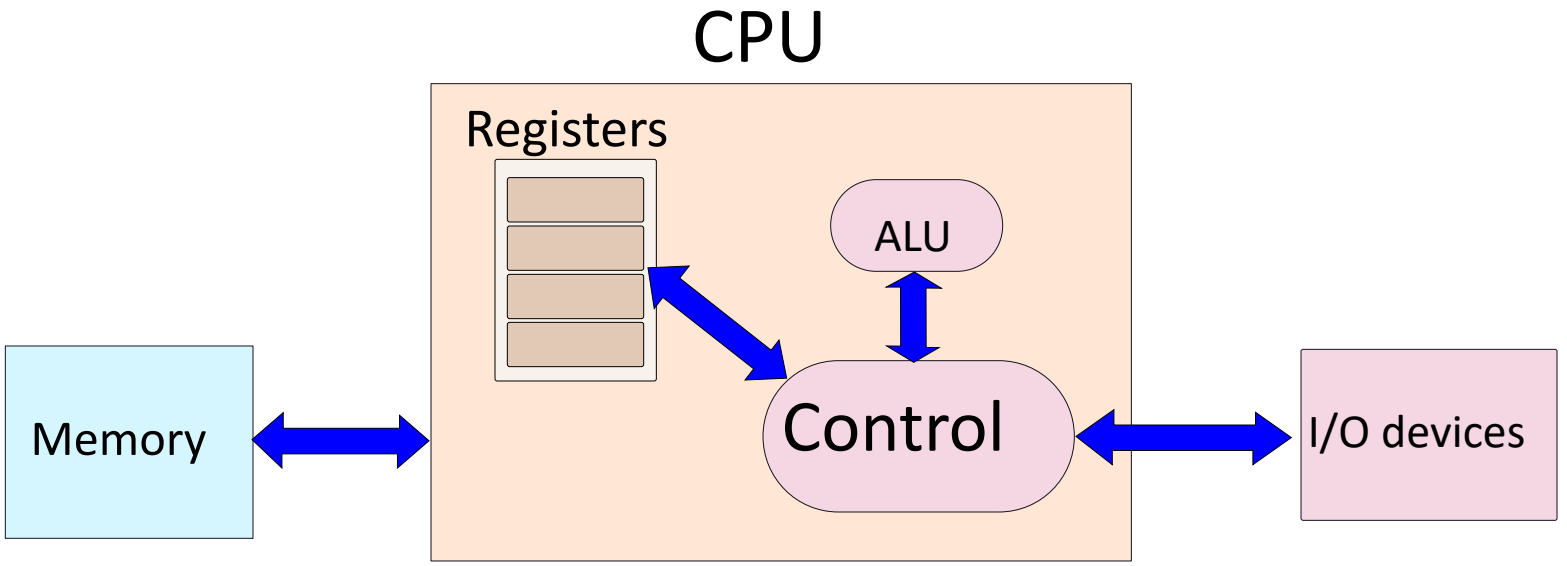
Designing Practical Machines



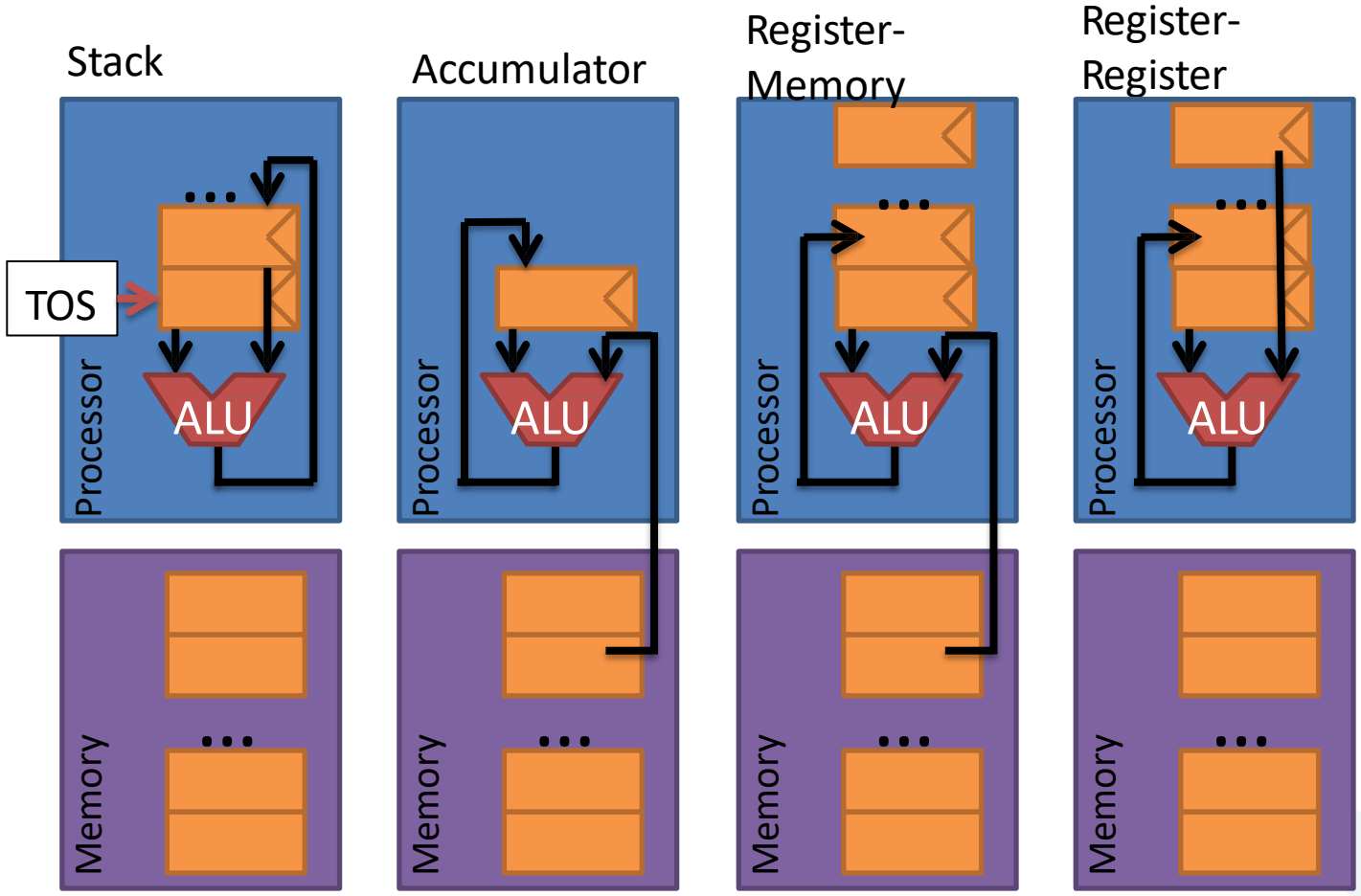
Von-Neumann Architecture



Machine with Registers

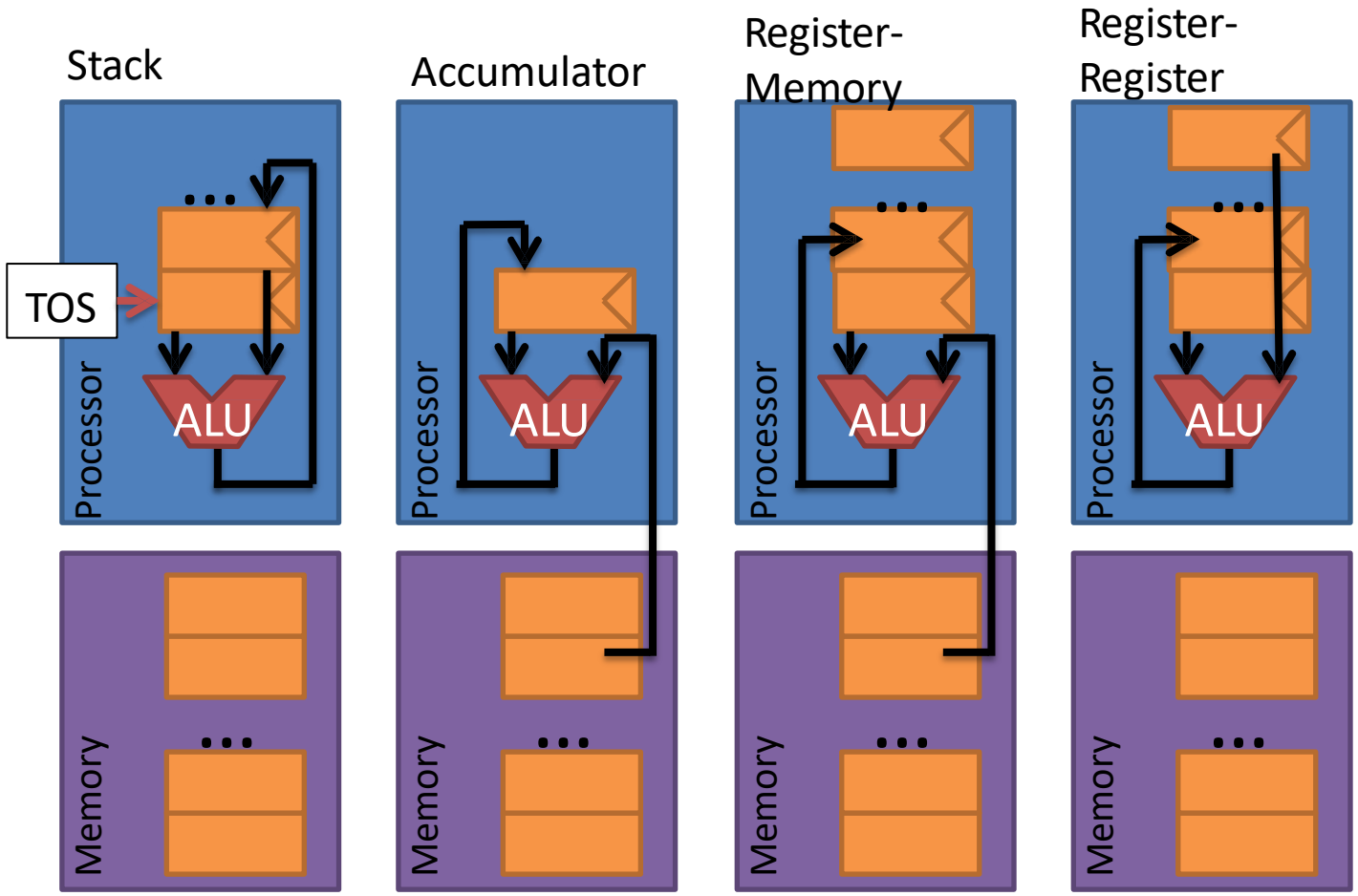


Where Do Operands Come from And Where Do Results Go?



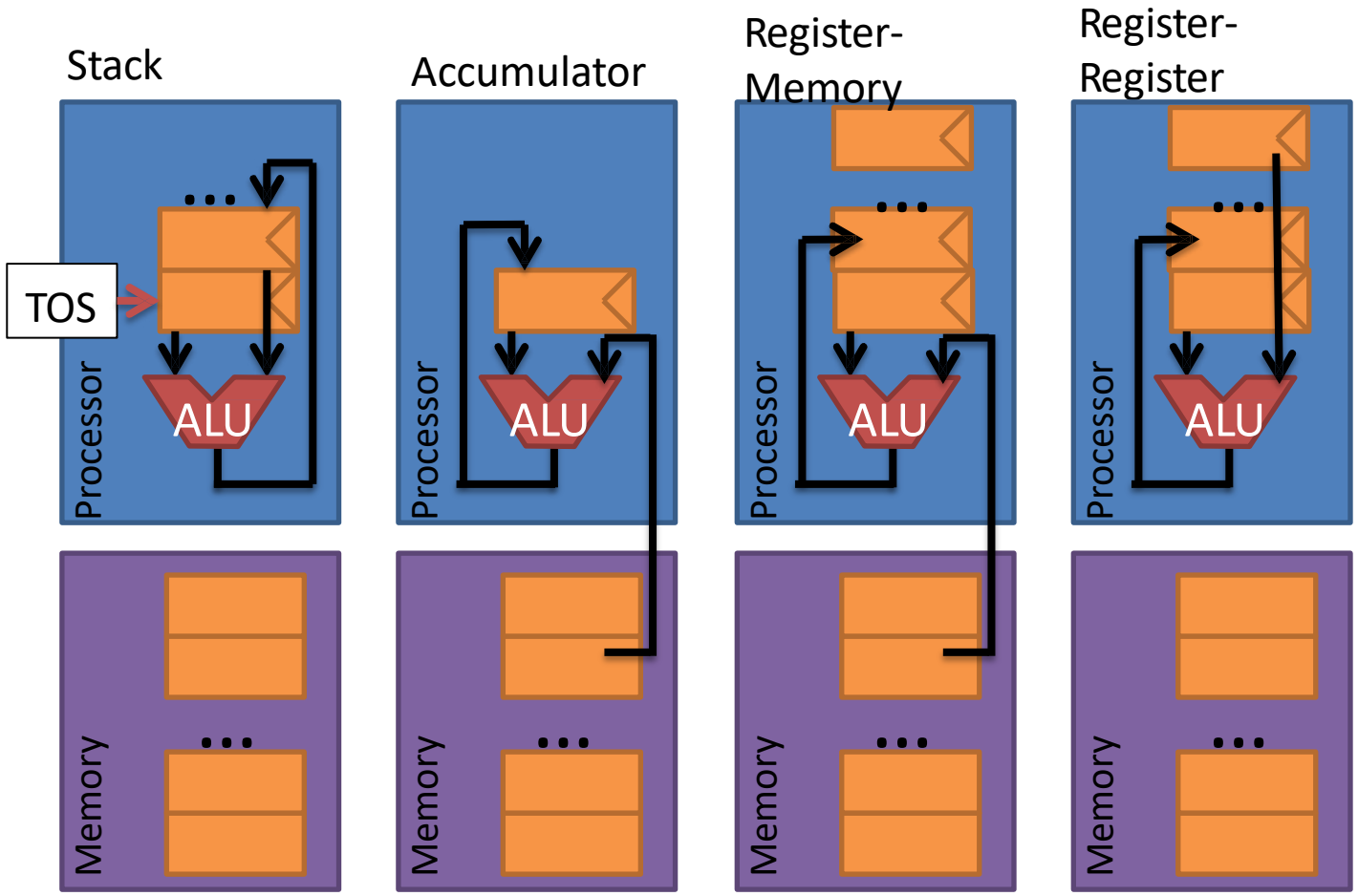
Number Explicitly Named Operands:	0	1	2 or 3	2 or 3

Machine Model Summary



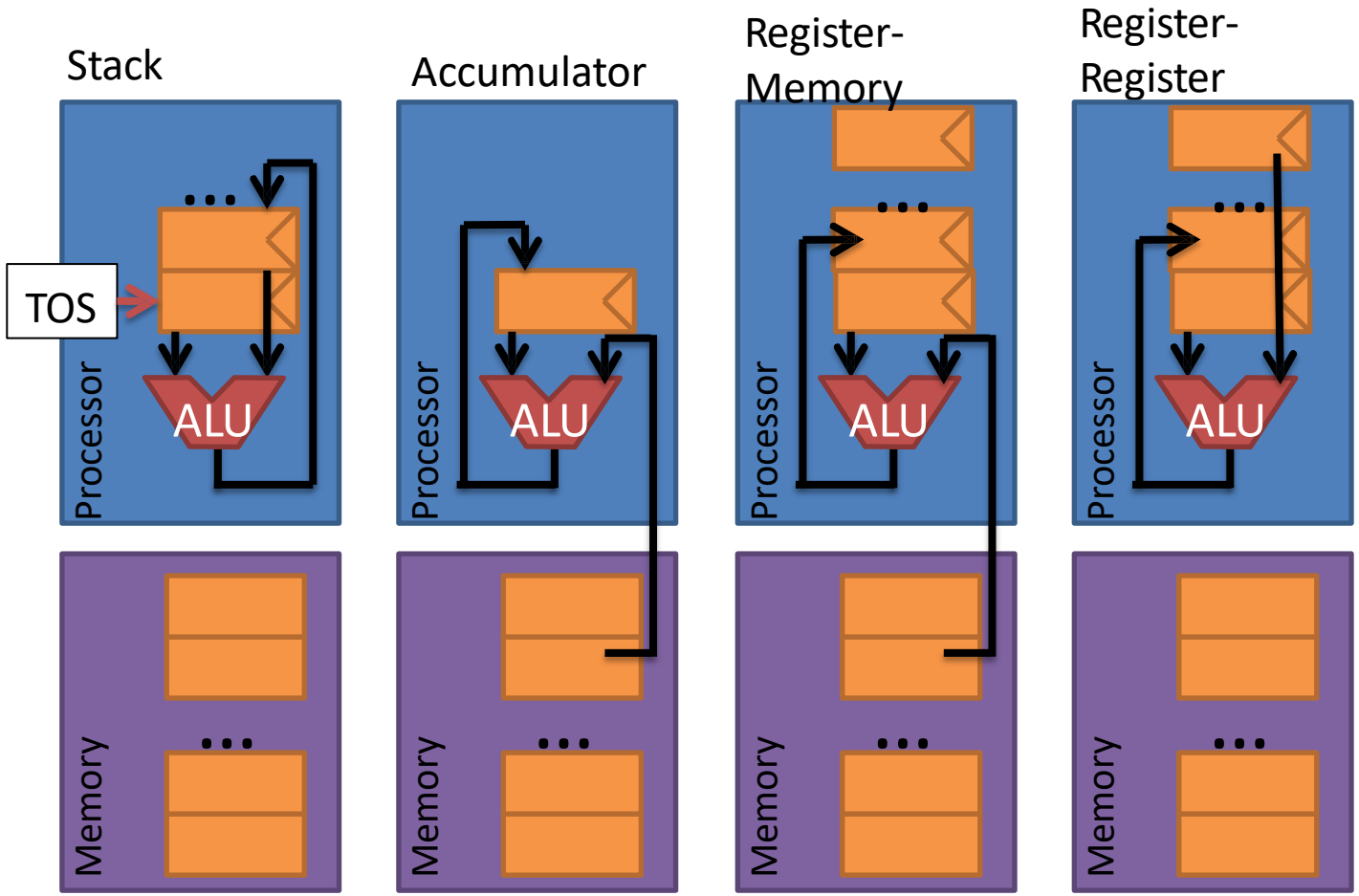
Machine Model Summary

C = A + B



Machine Model Summary

C = A + B



Classes of Instructions

- Data Transfer
  - LW, SW
- ALU
  - ADD, SUB, AND, OR, XOR
- Control Flow
  - BEQ, BNE
- Floating Point
  - ADD.D, SUB.S, MUL.D

Addressing Modes: How to Get Operands from Memory

Addressing Mode	Instruction	Function
Register	Add R4, R3, R2	Regs[R4] <- Regs[R3] + Regs[R2] **
Immediate	Add R4, R3, #5	Regs[R4] <- Regs[R3] + 5 **
Displacement	Add R4, R3, 100(R1)	Regs[R4] <- Regs[R3] + Mem[100 + Regs[R1]]
Register Indirect	Add R4, R3, (R1)	Regs[R4] <- Regs[R3] + Mem[Regs[R1]]
Absolute	Add R4, R3, (0x475)	Regs[R4] <- Regs[R3] + Mem[0x475]
PC relative	Add R4, R3, 100(PC)	Regs[R4] <- Regs[R3] + Mem[100 + PC]

** May not actually access memory!

Class Interaction #7

Data Types and Sizes

- Types
  - Binary Integer
  - Binary Coded Decimal (BCD)
  - Floating Point
    - IEEE 754
    - Cray Floating Point
    - Intel Extended Precision (80-bit)
  - Packed Vector Data
  - Addresses
- Width
  - Binary Integer (8-bit, 16-bit, 32-bit, 64-bit)
  - Floating Point (32-bit, 40-bit, 64-bit, 80-bit)
  - Addresses (16-bit, 24-bit, 32-bit, 48-bit, 64-bit)

ISA Encoding

Fixed Width: Every Instruction has same width

- Easy to decode

(RISC Architectures: RISC V, MIPS, PowerPC, SPARC, ARM..)

Ex: RISC V, every instruction 4-bytes

Variable Length: Instructions can vary in width

- Takes less space in memory and caches

(CISC Architectures: IBM 360, x86, Motorola 68k, VAX…)

Ex: x86, instructions 1-byte upto 18 bytes

Mostly Fixed or Compressed:

- Ex: MIPS16, THUMB (only two formats 2 and 4 bytes)
- PowerPC and some VLIWs (Store instructions compressed, decompress into Instruction Cache

(Very) Long Instruction Word:

- Multiple instructions in a fixed width bundle
- Ex: Multiflow, HP/ST Lx, TI C6000

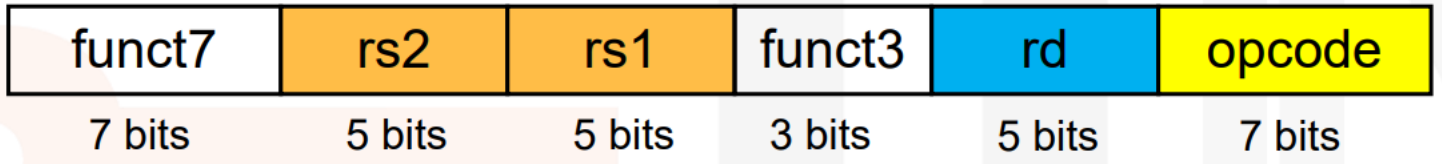
x86 (IA-32) Instruction Encoding

Instruction Prefixes	Opcode	ModR/M	Scale, Index, Base	Displacement	Immediate
Up to four Prefixes (1 byte each)	1,2, or 3 bytes	1 byte (if needed)	1 byte (if needed)	0,1,2, or 4 bytes	0,1,2, or 4 bytes

x86 and x86-64 Instruction Formats

Possible instructions 1 to 18 bytes long

RISC V Instruction Encoding



Real World Instruction Sets

Arch	Type	# Oper	# Mem	Data Size	# Regs	Addr Size	Use
Alpha	Reg-Reg	3	0	64-bit	32	64-bit	Workstation
ARM	Reg-Reg	3	0	32/64-bit	16	32/64-bit	Cell Phones, Embedded
MIPS	Reg-Reg	3	0	32/64-bit	32	32/64-bit	Workstation, Embedded
SPARC	Reg-Reg	3	0	32/64-bit	24-32	32/64-bit	Workstation
TI C6000	Reg-Reg	3	0	32-bit	32	32-bit	DSP
IBM 360	Reg-Mem	2	1	32-bit	16	24/31/64	Mainframe
x86	Reg-Mem	2	1	8/16/32/ 64-bit	4/8/24	16/32/64	Personal Computers
VAX	Mem-Mem	3	3	32-bit	16	32-bit	Minicomputer
Mot. 6800	Accum.	1	1/2	8-bit	0	16-bit	Microcontroler

Why the Diversity in ISAs?

Technology Influenced ISA

- Storage is expensive, tight encoding important
- Reduced Instruction Set Computer
  - Remove instructions until whole computer fits on die
- Multicore/Manycore
  - Transistors not turning into sequential performance

Application Influenced ISA

- Instructions for Applications
  - DSP instructions
- Compiler Technology has improved