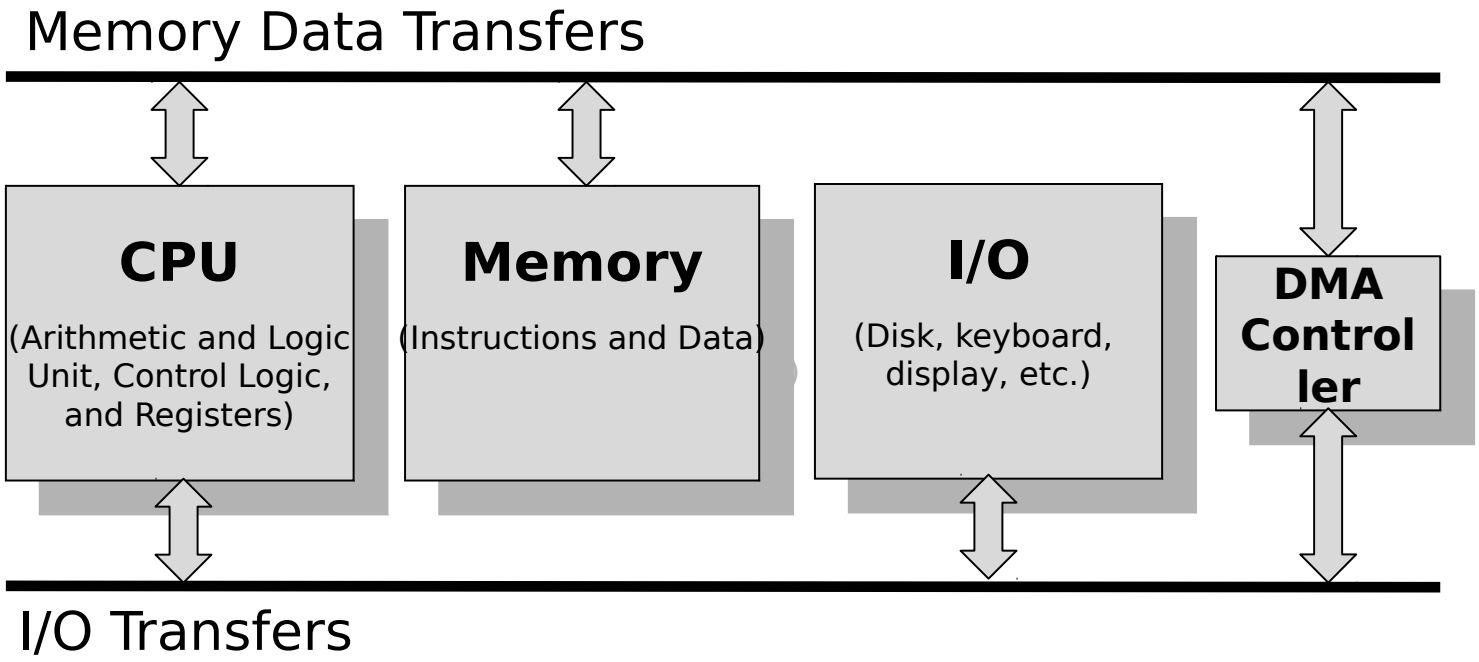
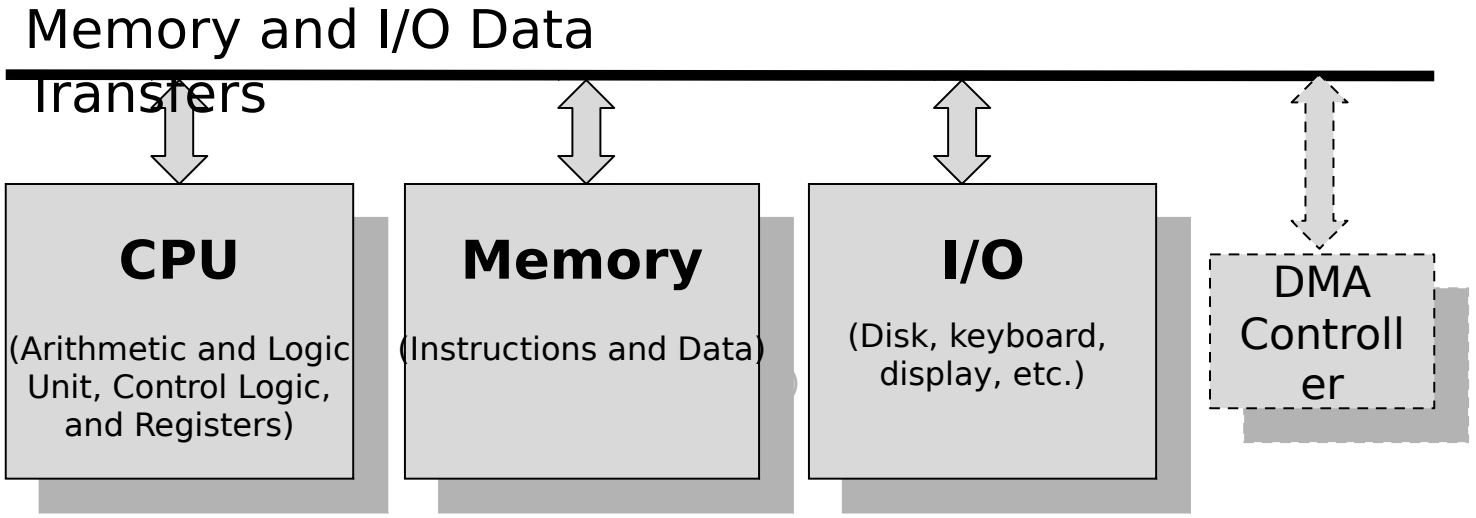
CHAPTER 5

Programming in Assembly
Part 1: Computer Organization

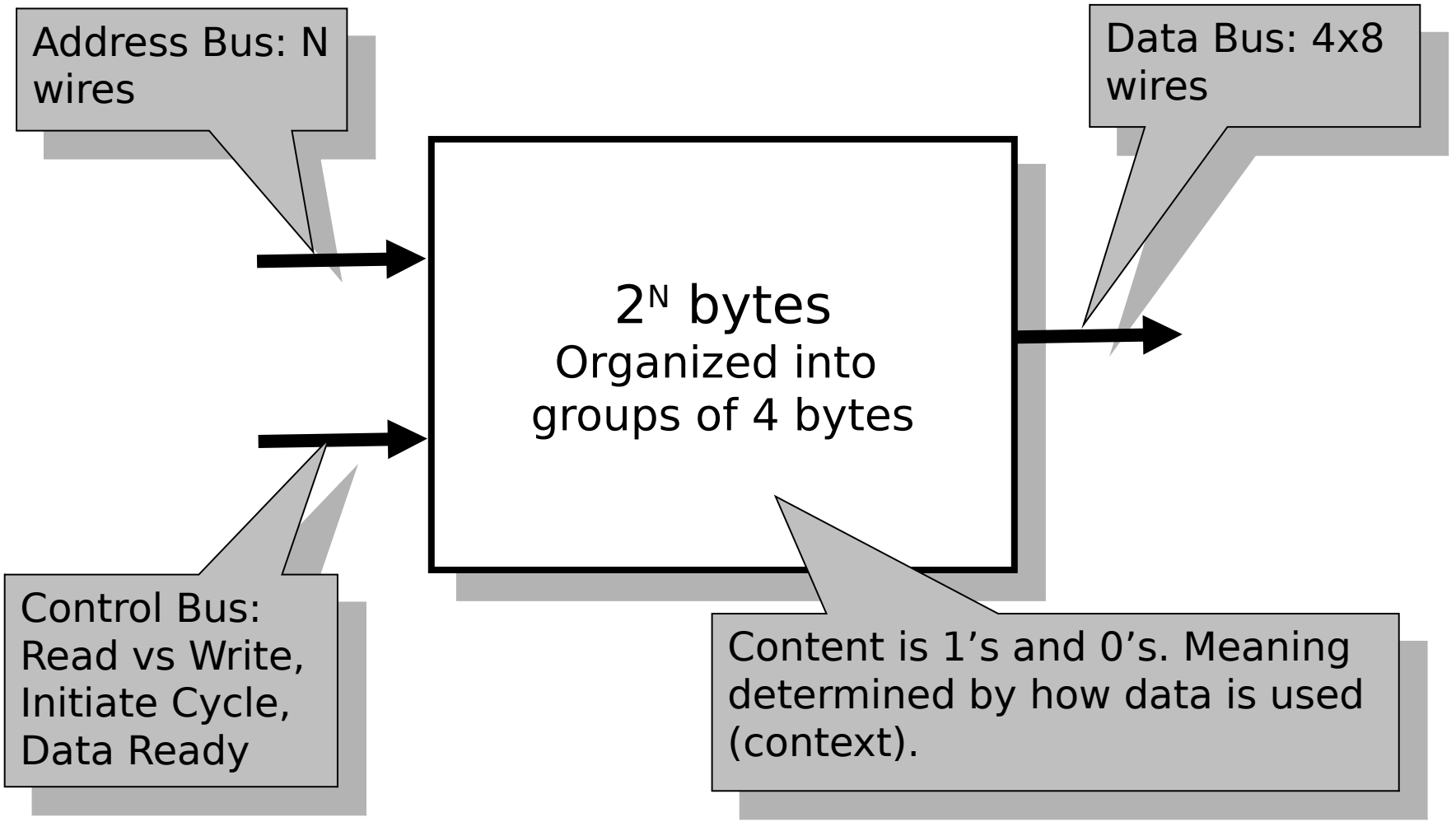
Separate Memory & I/O Busses



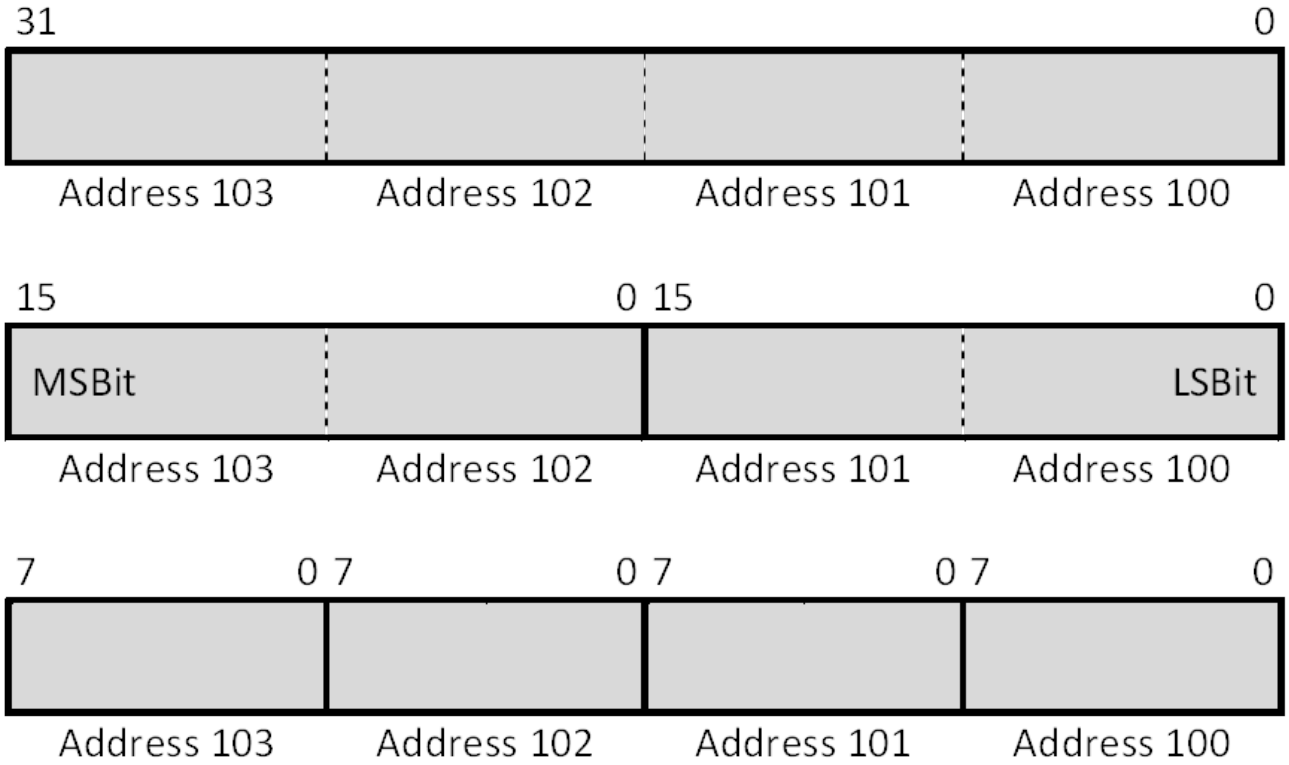
Memory-Mapped Peripherals



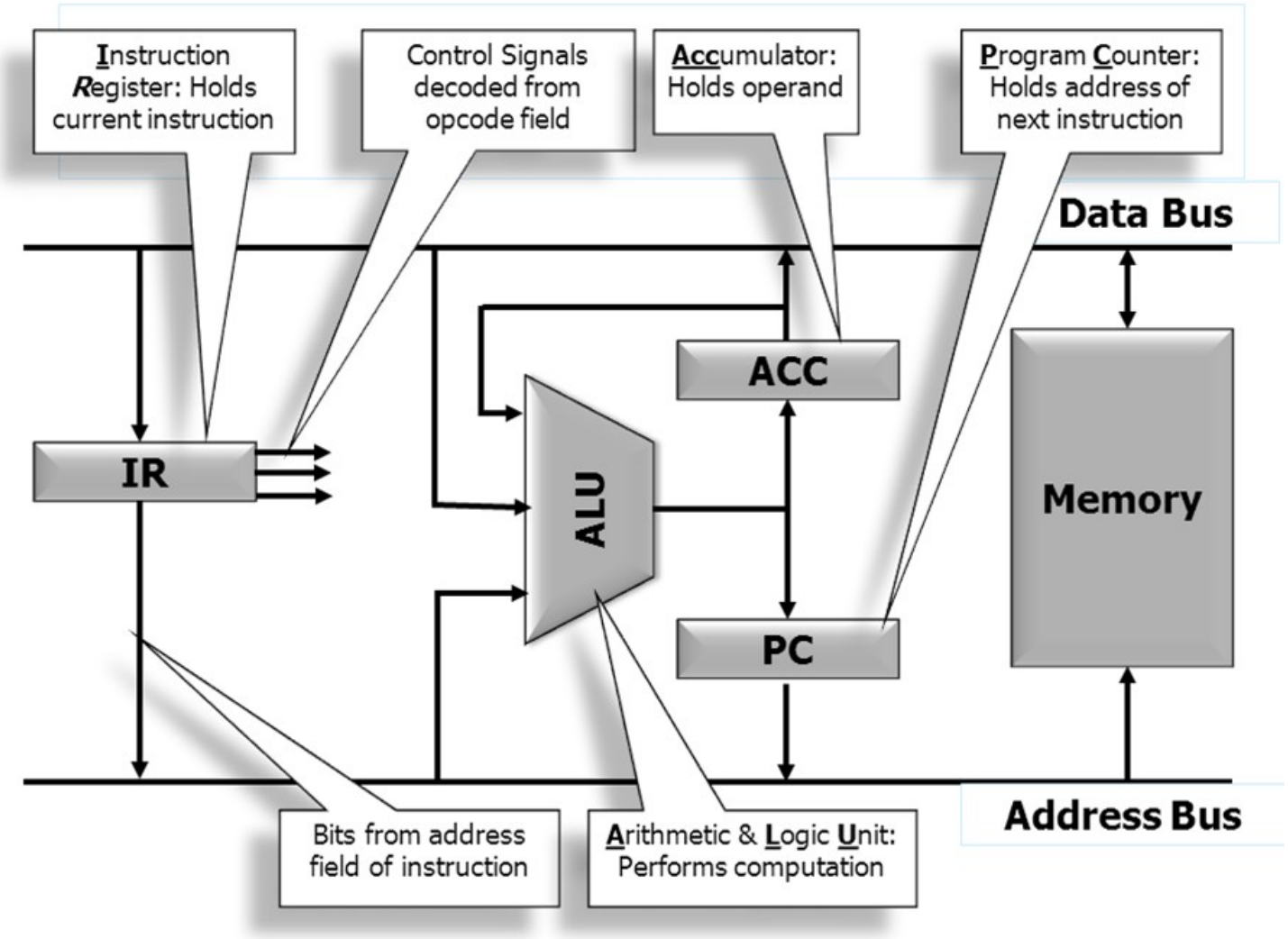
Memory


Address Bus: N wires
Data Bus: 4x8 wires
2^N bytes
Organized into groups of 4 bytes
Control Bus: Read vs Write, Initiate Cycle, Data Ready
Content is 1’s and 0’s. Meaning determined by how data is used (context).

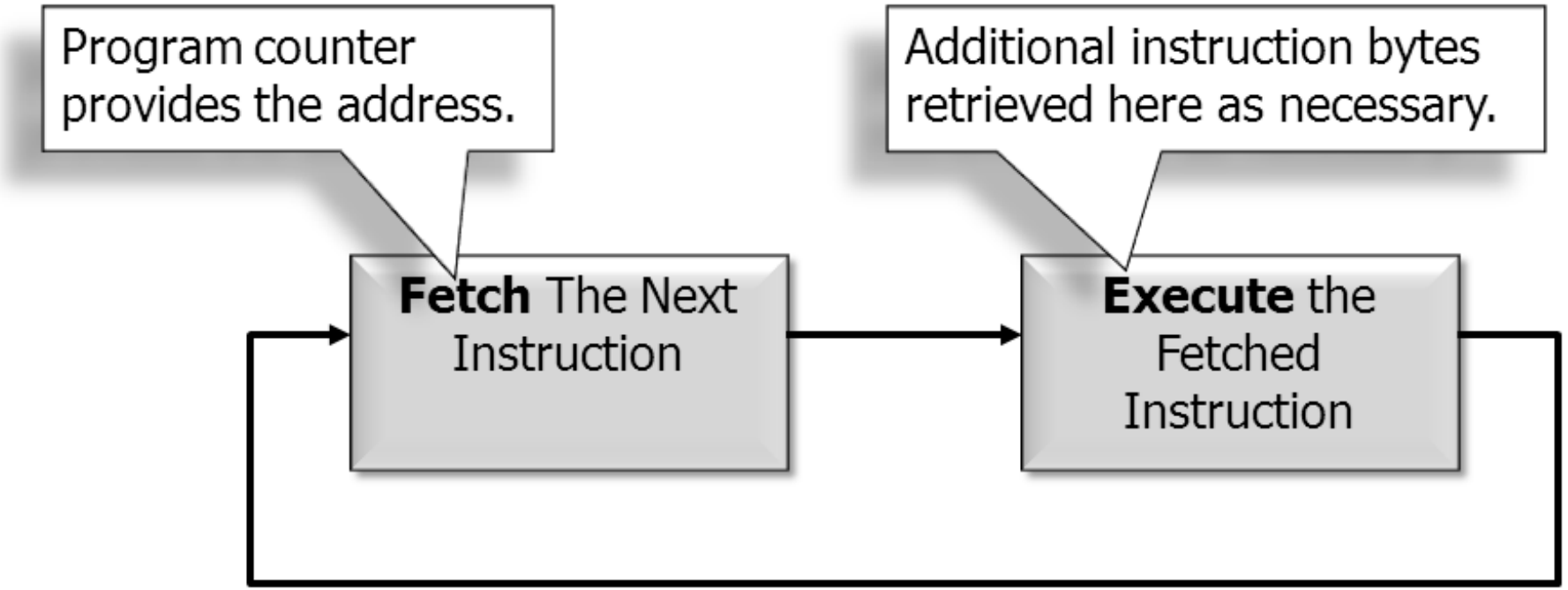
Memory
(Little Endian Ordering)



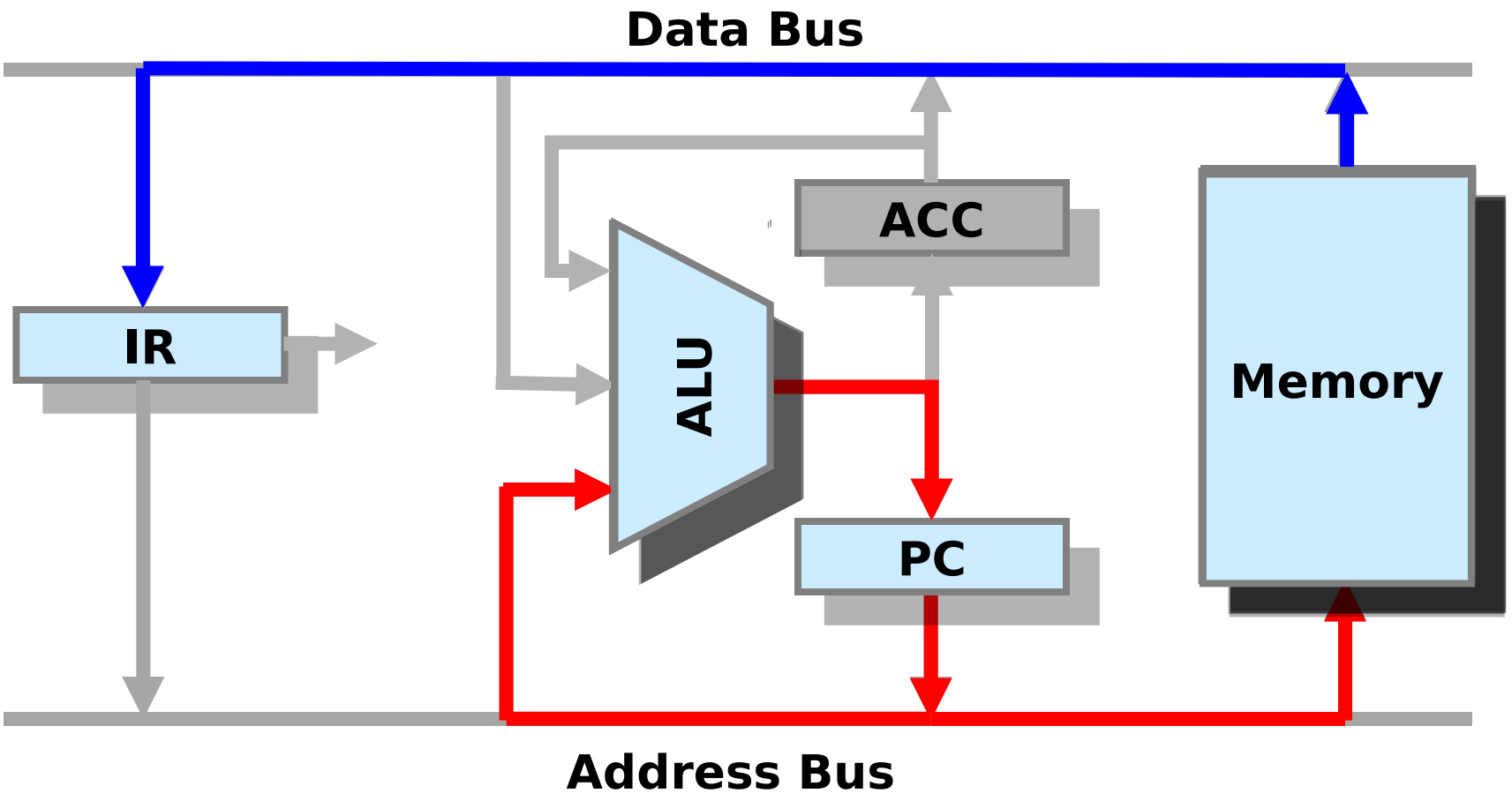
Single Accumulator Architecture



The Fetch-Execute Cycle



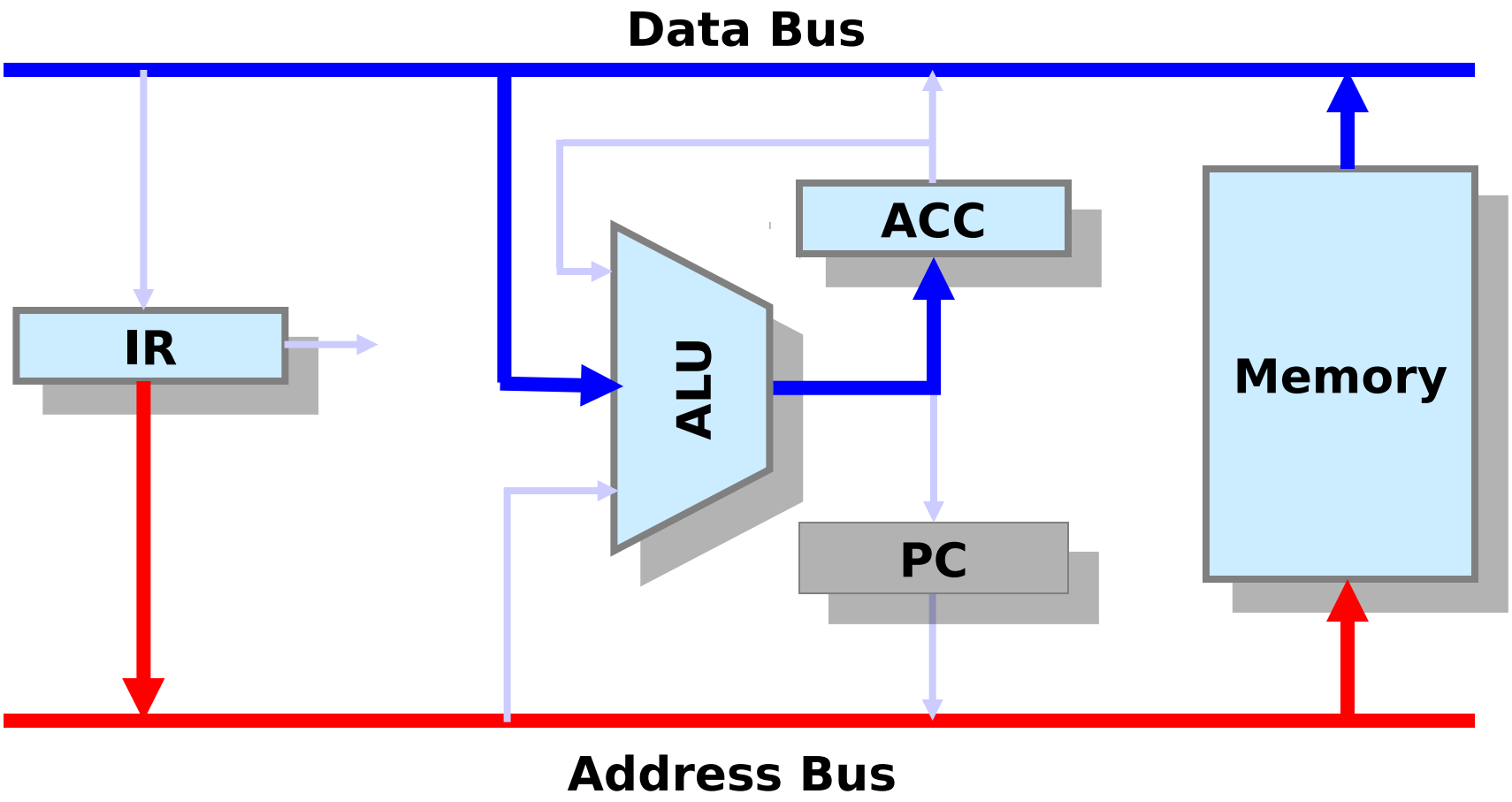
Instruction Fetch



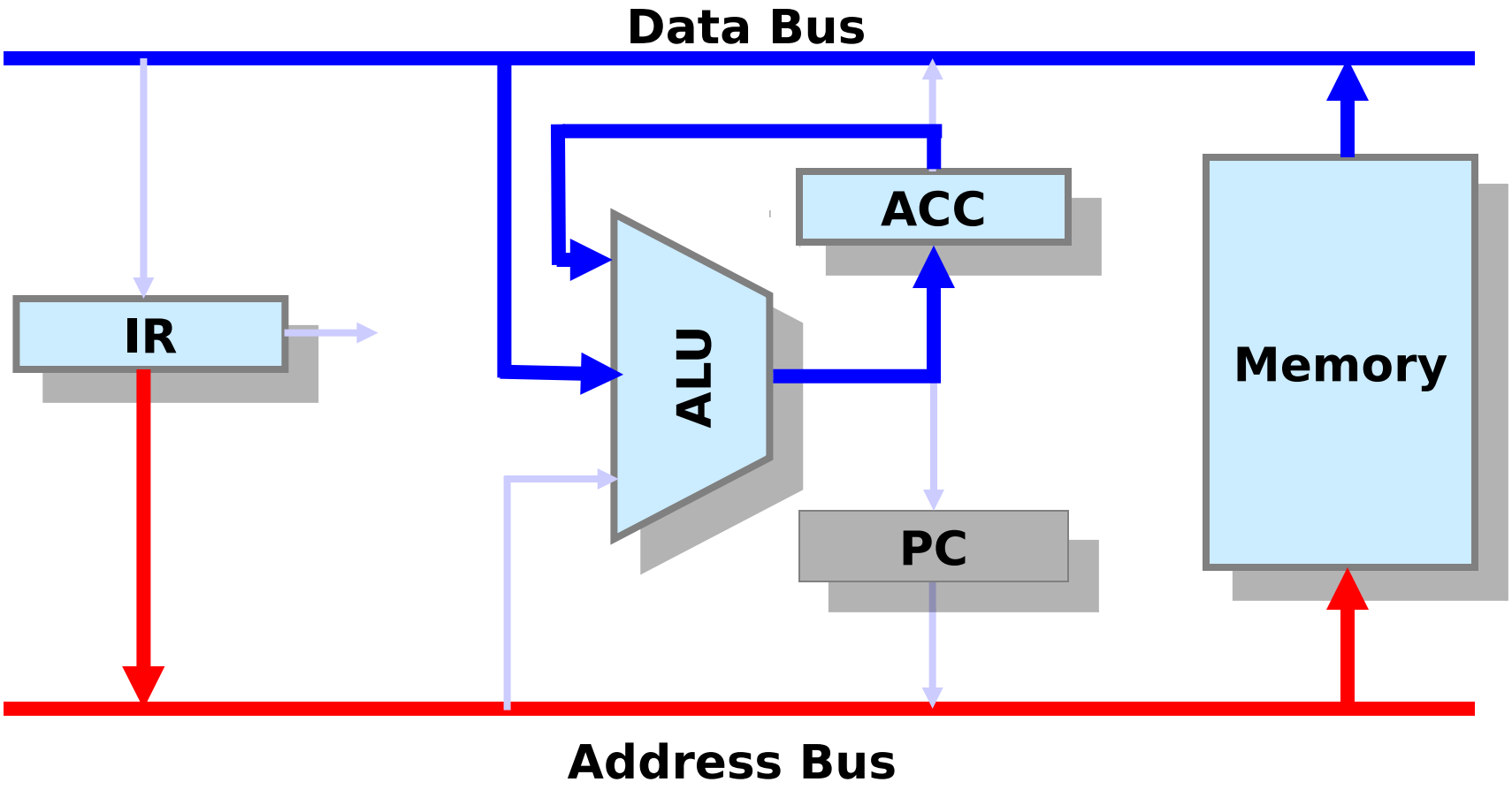
The Fetch Phase

1. Memory_Address_Bus ← Program_Counter
2. Start Memory Read Operation
3. Increment Program_Counter
4. Wait for Memory Read to Complete
5. Instruction_Register ← Memory_Data_Bus
6. Go to execute phase.

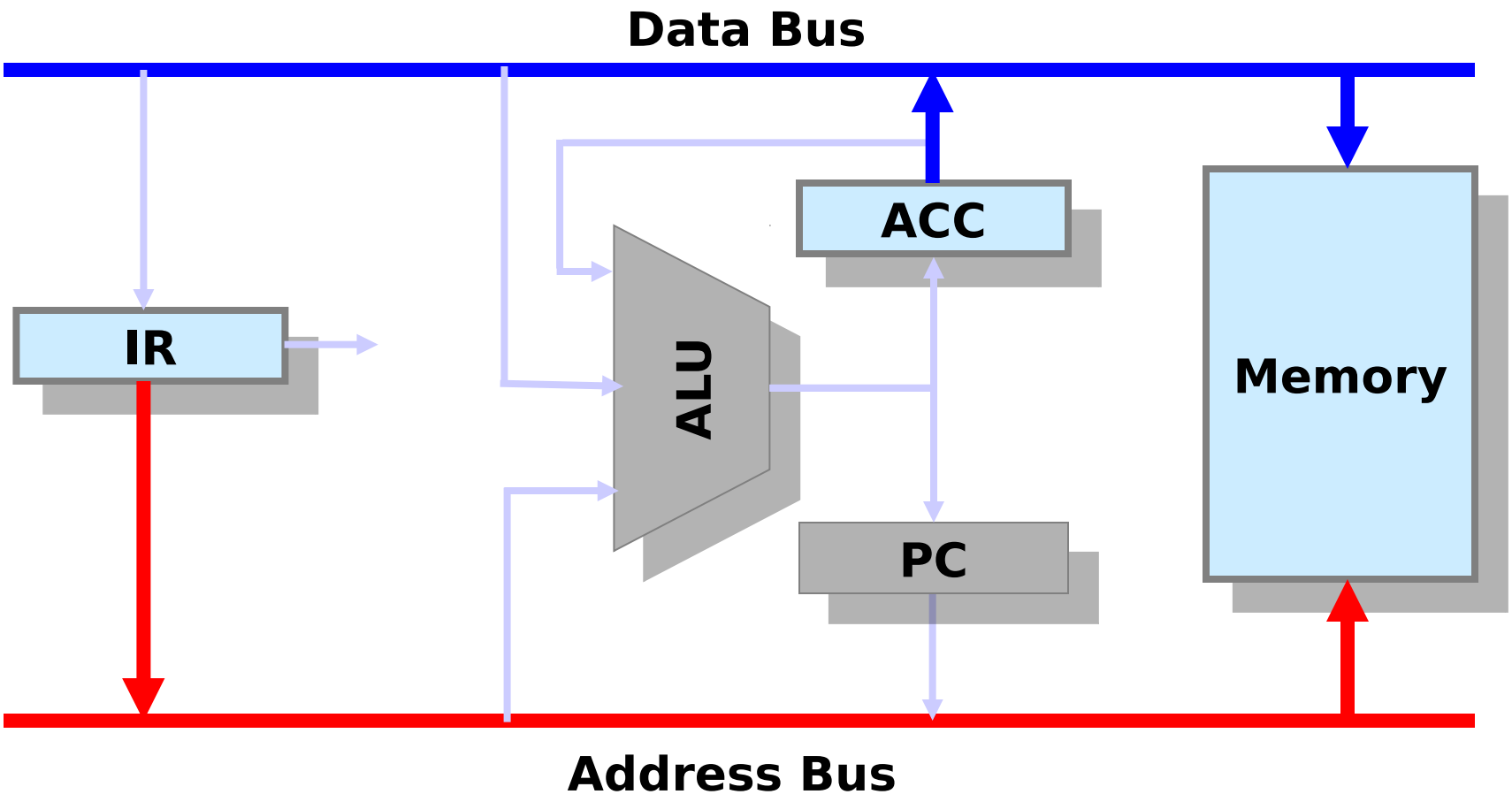
Execution Phase: Load ACC



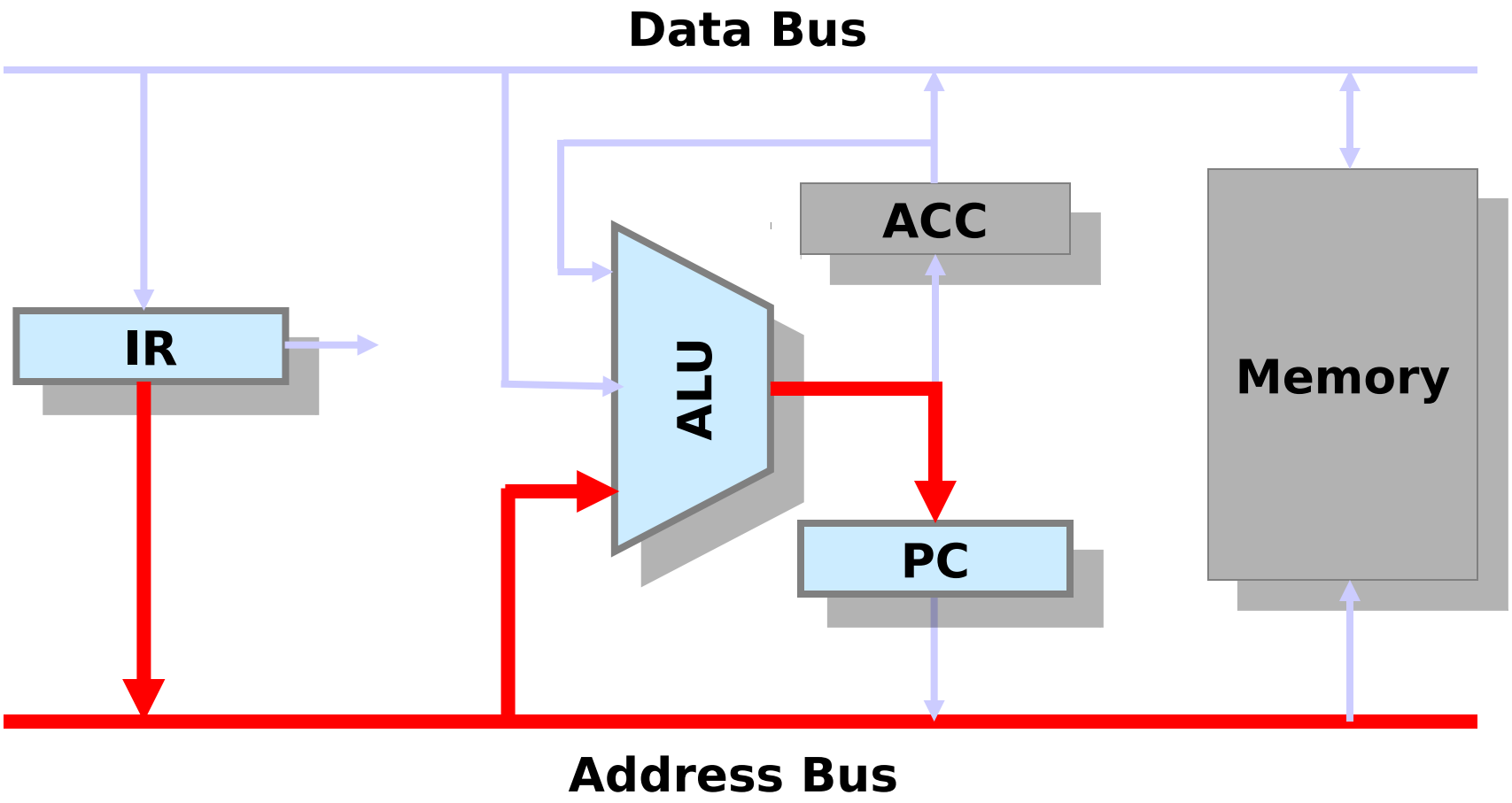
Execution Phase: ADD



Execution Phase: Store ACC



Execution Phase: Branch



result ← op1 + op2

Single Accumulator Machine:

ACC ← MEM[adrs_of_op1]
ACC ← ACC + MEM[adrs_of_op2]
MEM[adrs_of_result] ← ACC

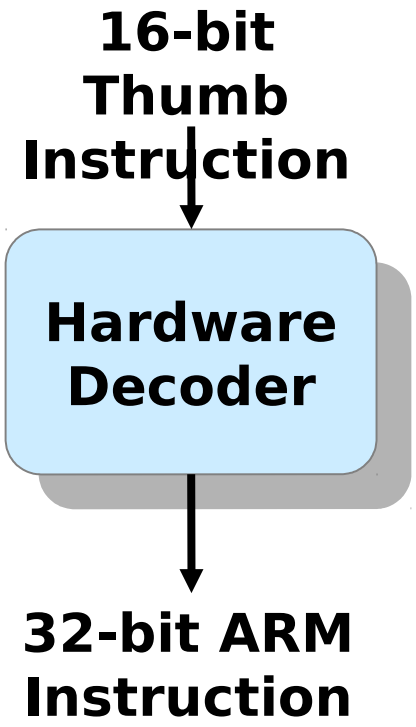
Register Machine:

REG[r] ← MEM[adrs_of_op1]
REG[r] ← REG[r] + MEM[adrs_of_op2]
MEM[adrs_of_result] ← REG[r]

The ARM Processor Family

Three Instruction Sets

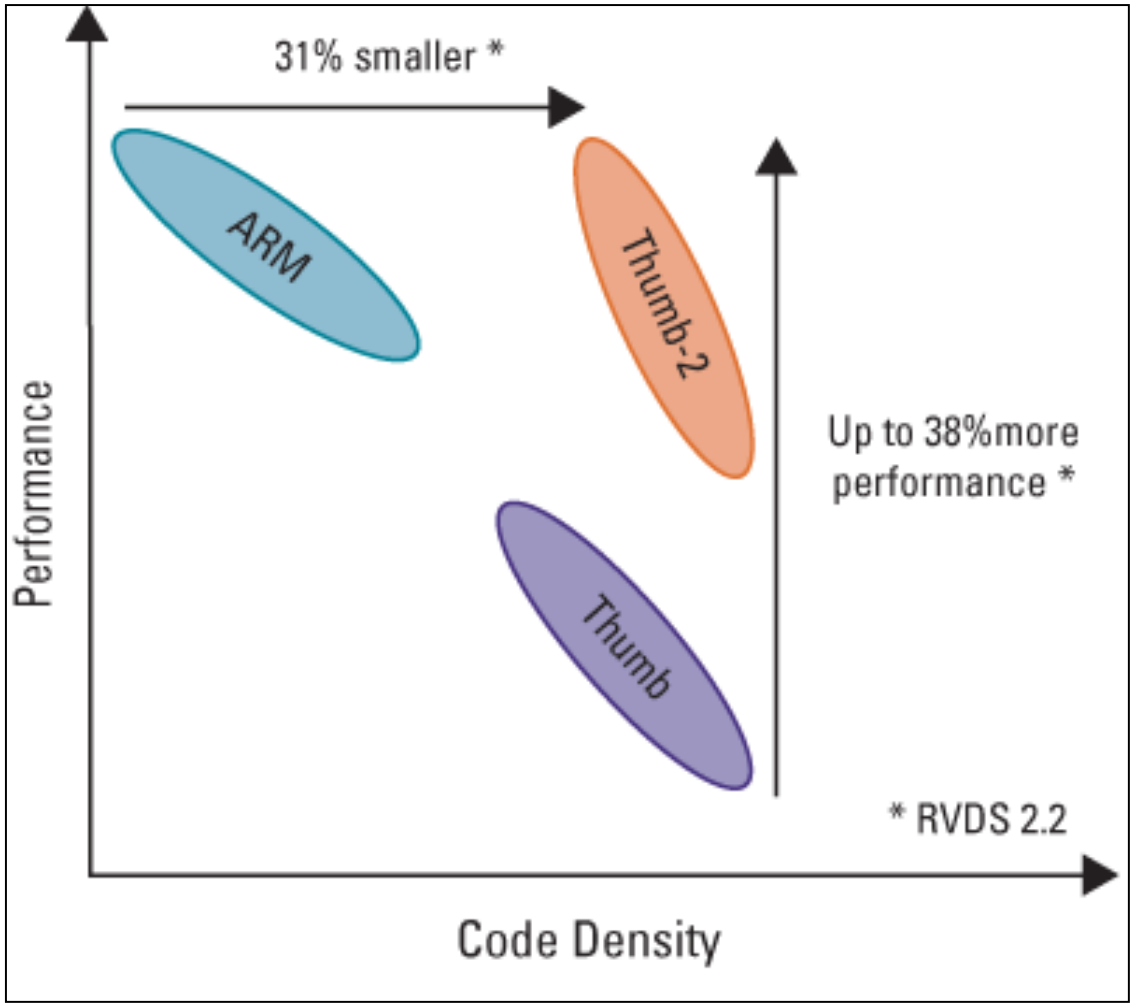
- ARM Instruction Set
  - Instructions are 32 bits wide
  - Original RISC (lots of parallelism)
  - “Load/Store” Architecture
- Thumb Instruction Set
  - Subset of ARM instructions, some restrictions
  - Instructions are 16 bits wide (more like CISC)
  - Intended for compilers
  - Less parallelism, longer instruction sequences
  - but total code size is 30% smaller
- Jazelle Instruction Set
  - Java byte codes



Three Instruction Sets

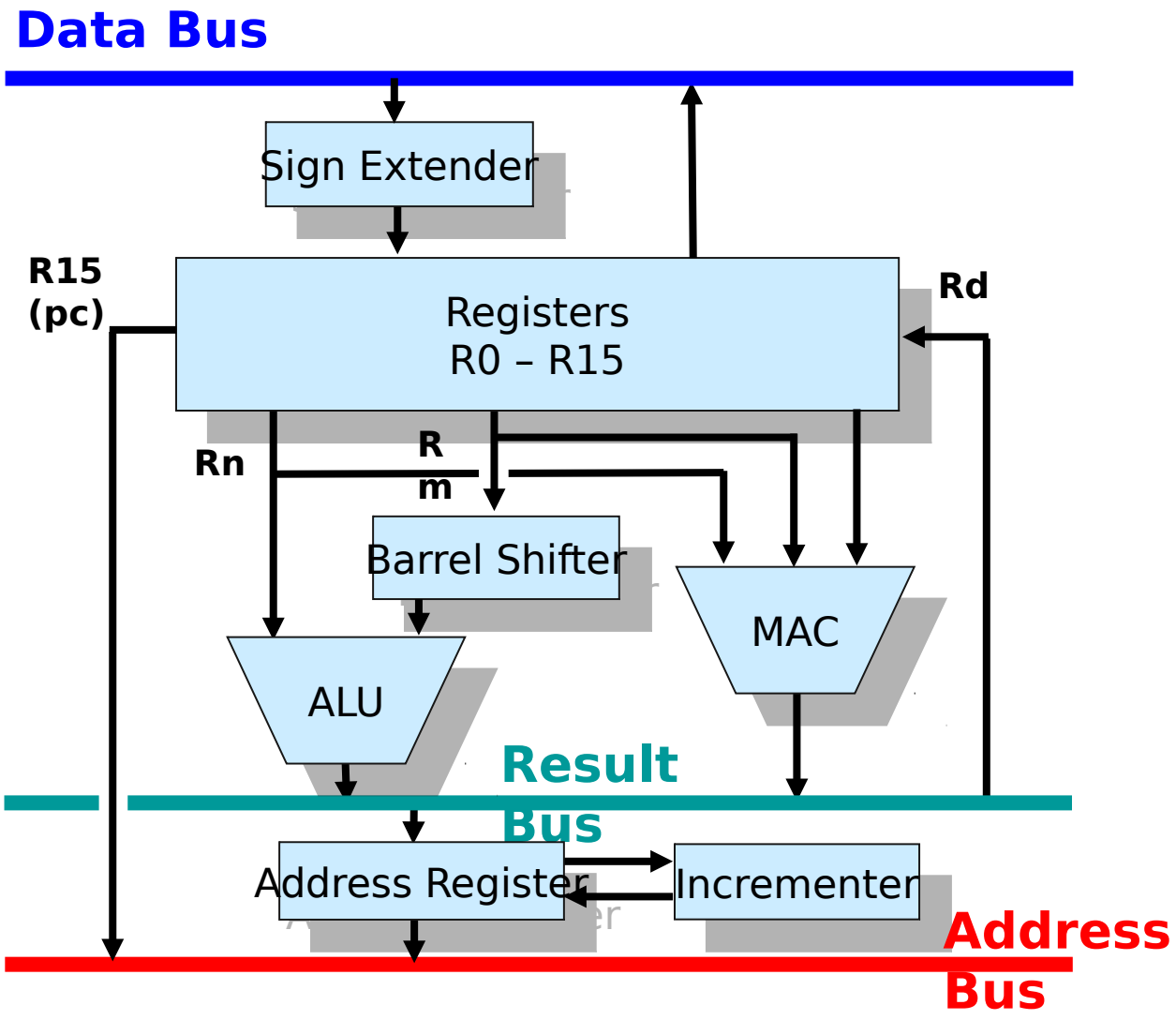
	ARM	Thumb*	Jazelle
Instruction Size	32 bits	16 bits	8 bits
Core instructions	58	30	> 60% of Java byte codes in hardware; rest in software
Conditional Execution	most	Only branch instructions or in an IT block	N/A
Data processing instructions	Access to barrel shifter and ALU	Separate barrel shifter and ALU instructions	N/A
Program status register	Read/write in privileged mode	No direct access	N/A
Register usage	15 general purpose registers + pc	8 general purpose registers + 7 high registers + pc	N/A

Code Density vs. Performance



The ARMv7-M “Cortex-M3” Architecture used in LM3S811

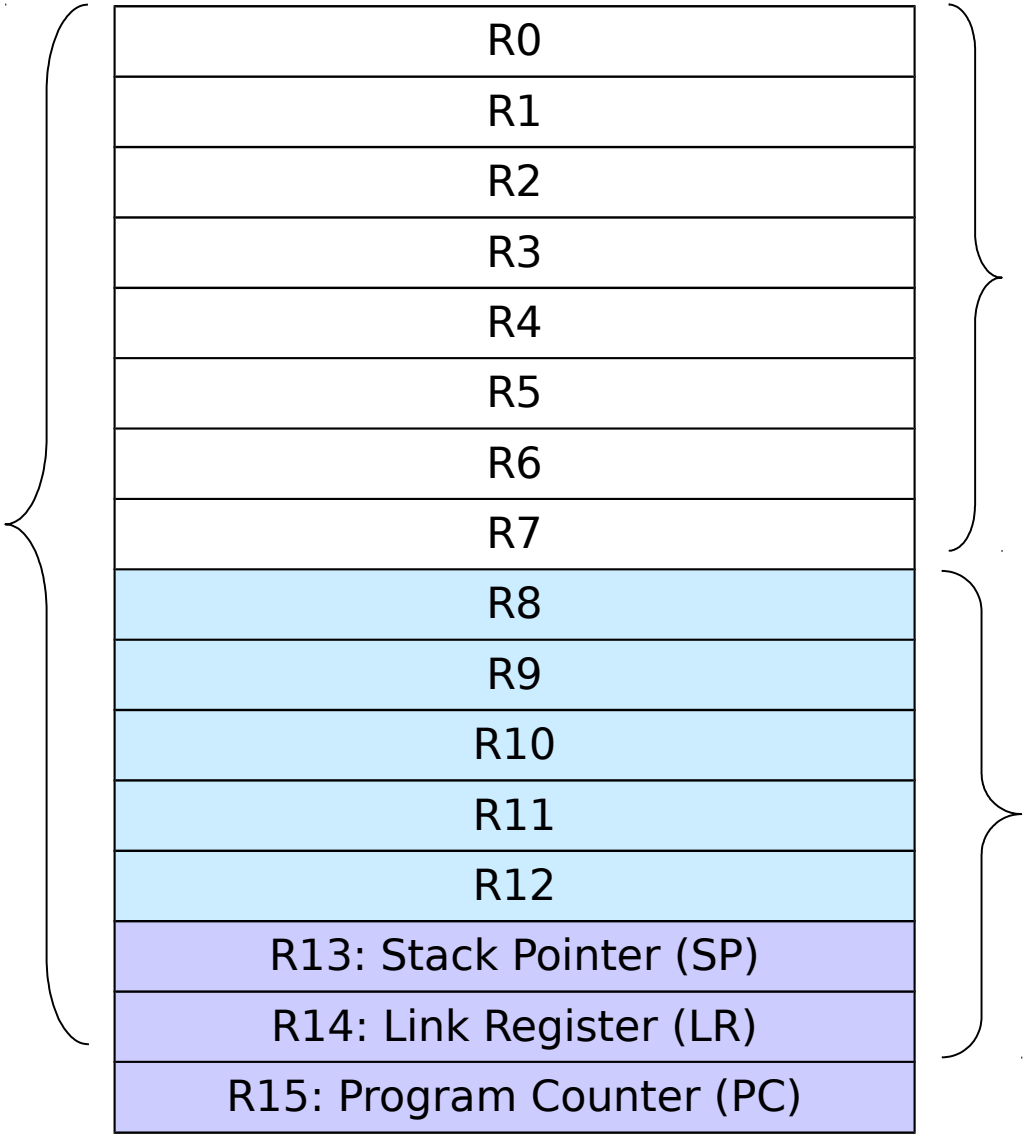
ARM Cortex-M3 CPU



ARM Registers

ARM Mode:

15 general purpose registers



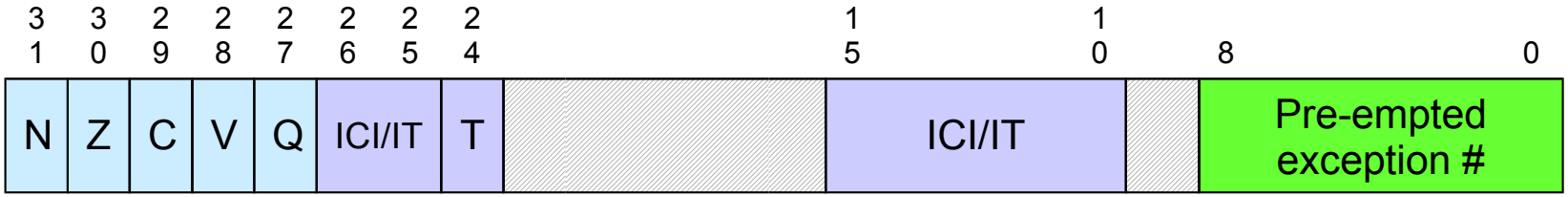
Thumb Mode:

8 general purpose registers

7 “high” registers

r8-R12 only accessible with MOV, ADD, or CMP

Processor Status Register (xPSR)



APSR
Application Program Status Register

EPSR
Execution Program Status Register

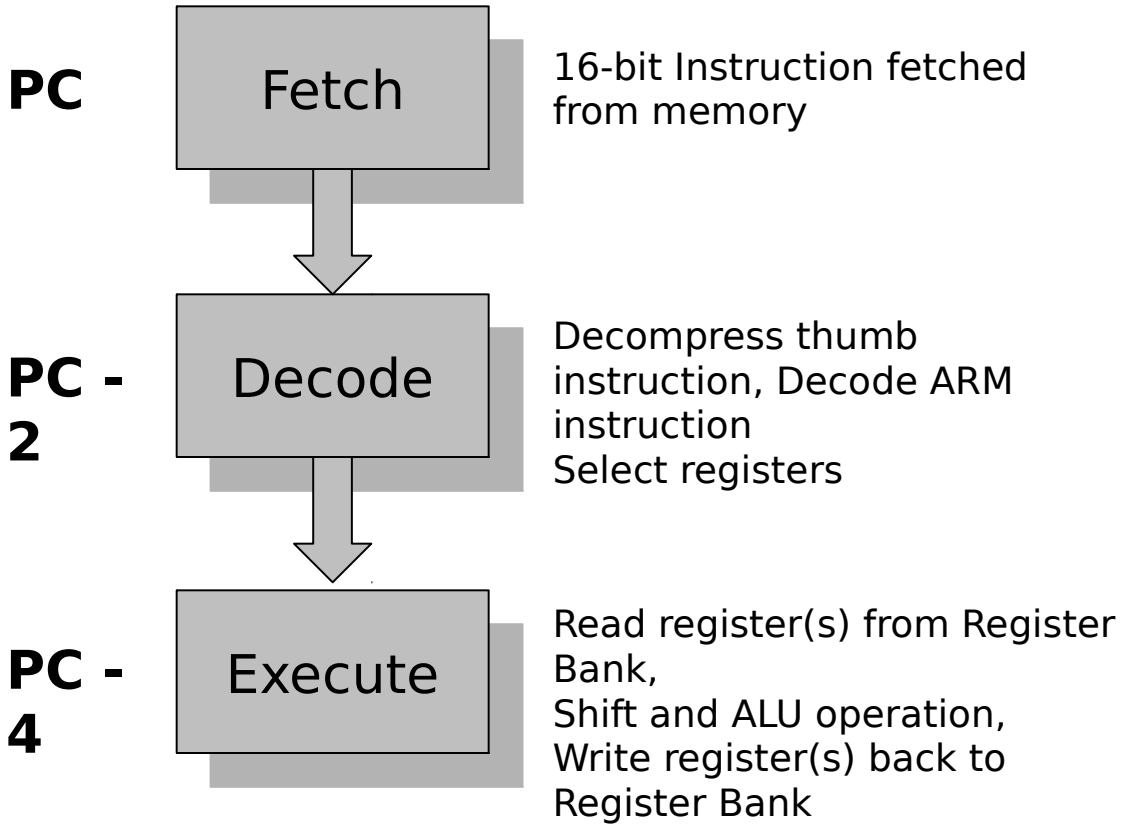
ICI: Used by LDM/STM
IT: Used by IT instr.
T: 1 = Thumb Mode

IPSR
Interrupt Program Status Register

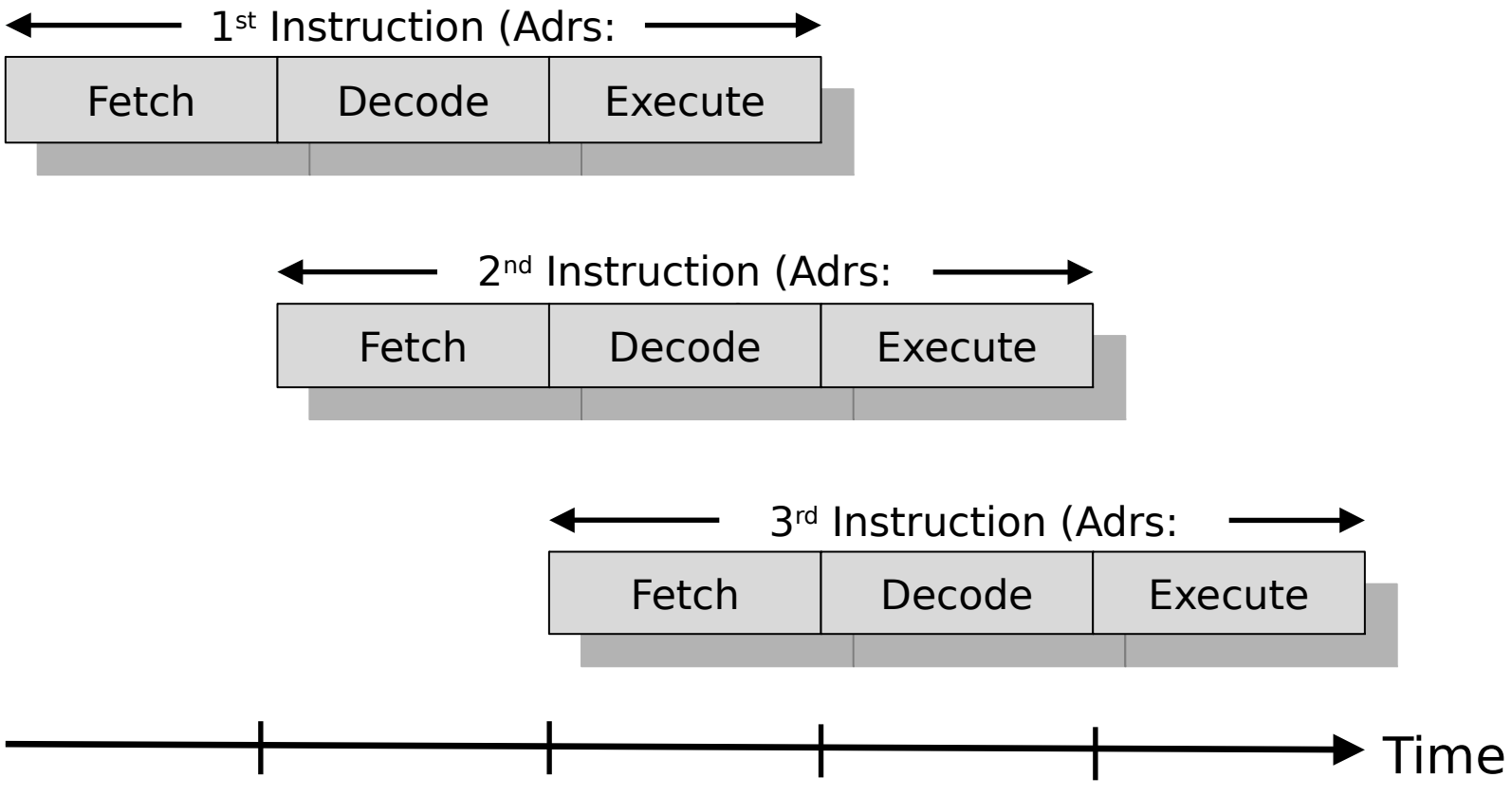
Bits	Name	Description
31	**N**	**Negative (or less than)**
30	**Z**	**Zero (or Equal)**
29	**C**	**Carry or Borrow Out**
28	**V**	**Overflow (2's complement)**
27	Q	Sticky Saturation Flag

Most important for application programming

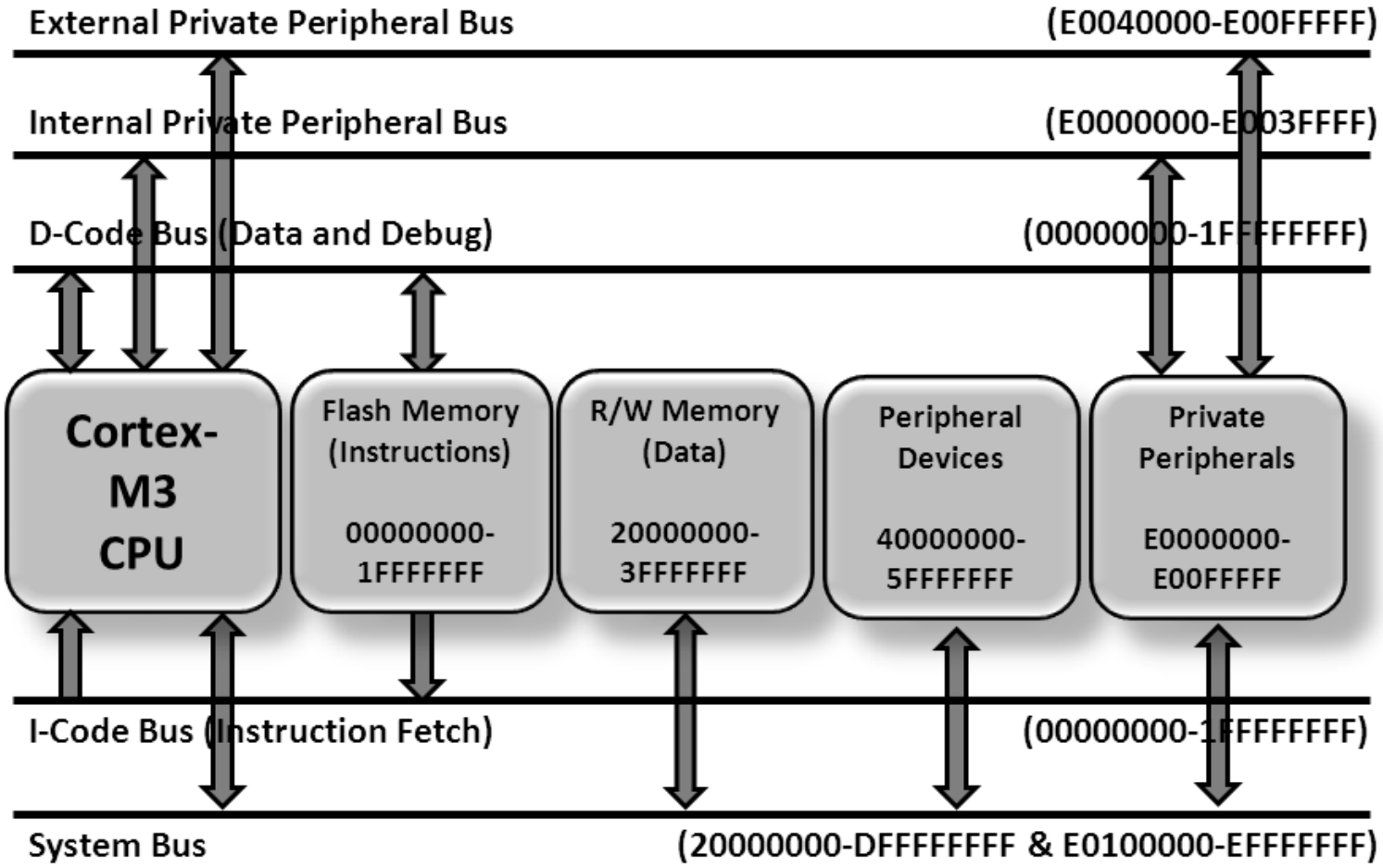
Pipelined Instruction Processing



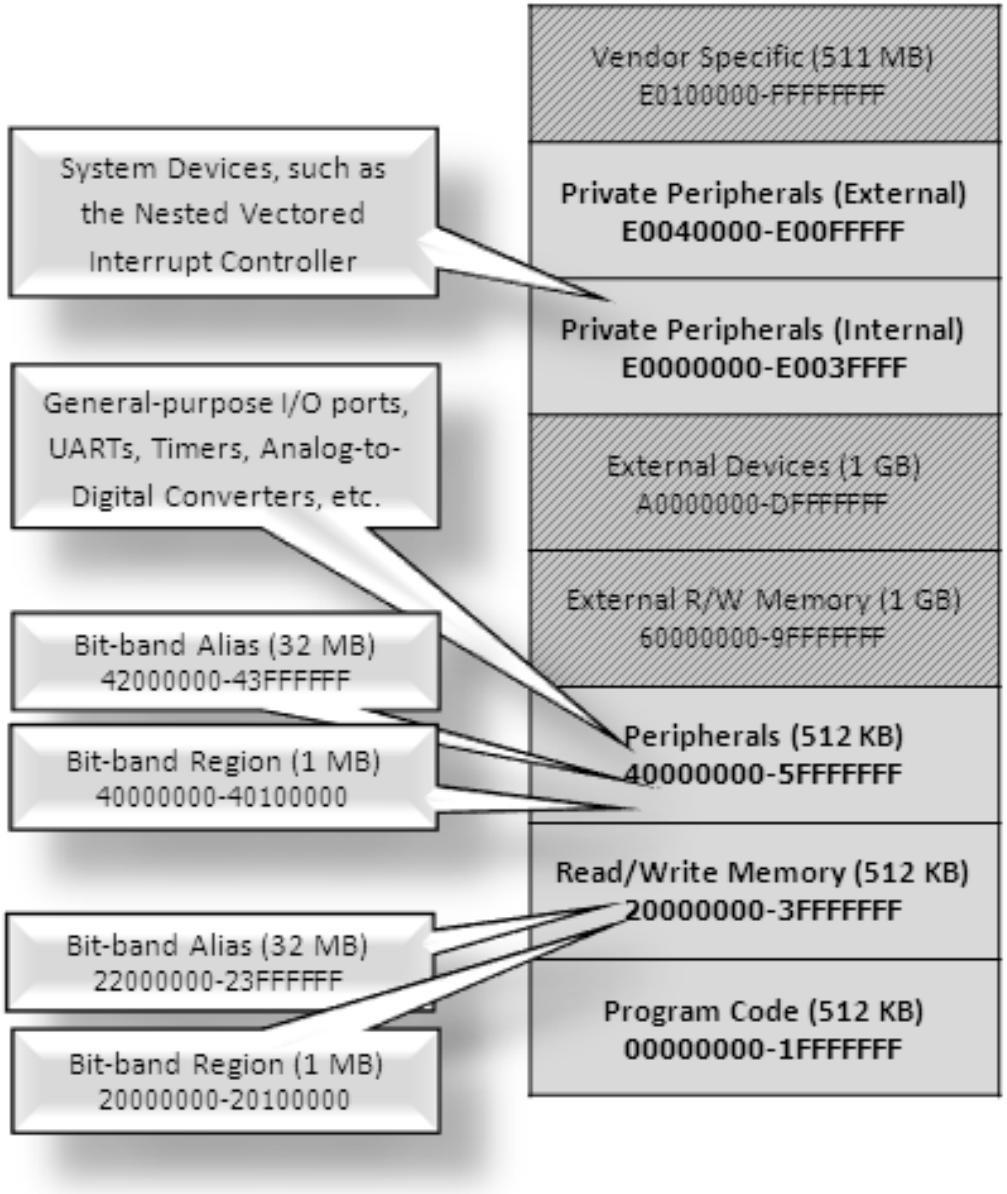
Pipelined Instruction Processing



ARM Cortex-M3 Bus Structure



ARM Cortex-M3 Memory



Bit-Banding

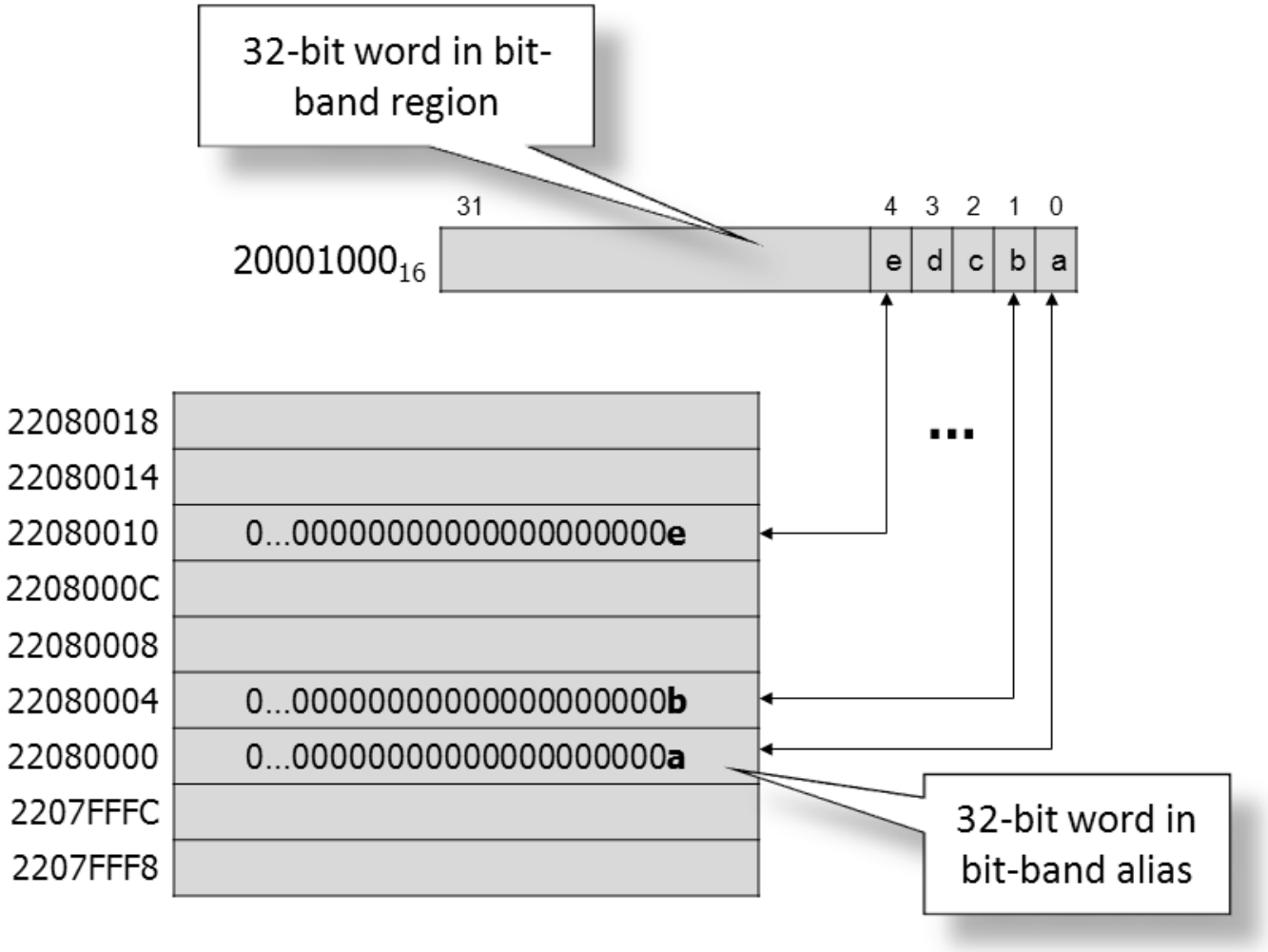
- 1 word (**4** bytes) of address space in the bit-band area = 1 bit in the corresponding primary area.
- bit-band alias = **bit-band base** + 128 × **word offset** + **4** × **bit number**
- For example, if bit **3** at address 2000**1000**$_{16}$ is to be modified, the bit-band alias is calculated as:

$$\mathbf{22000000}_{16} + 128_{10} \times \mathbf{1000}_{16} + \mathbf{4}_{10} \times \mathbf{3}_{10} = 2208000C_{16}$$

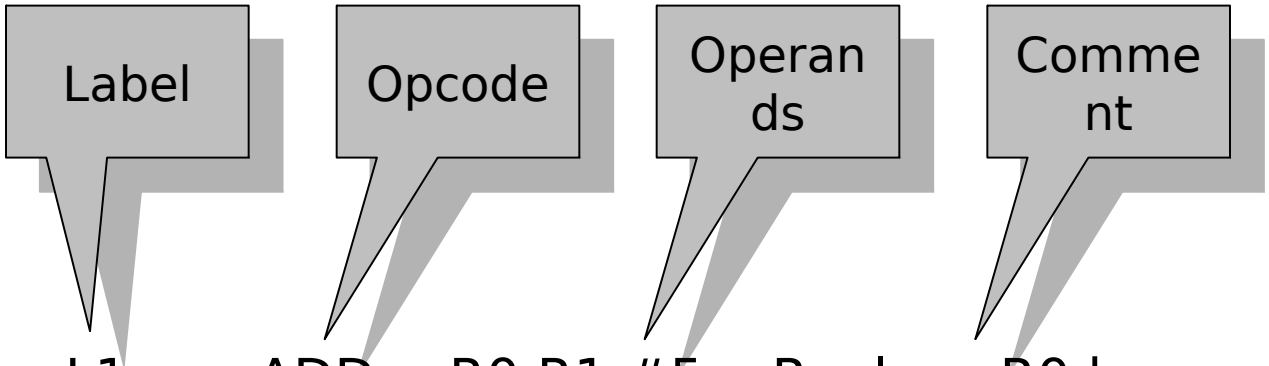
Address 2208000C allows direct access to only bit 3 of the byte at address 20001000:

- **Read from 2208000C: All other bits will be 0's**
- **Write to 2208000C: All other data bits are ignored; hardware performs a read-modify-write.**

ARM Cortex-M3 Bit-Banding



Assembler Syntax




```
L1:     ADD    R0,R1,#5  ; Replace R0 by sum of R1 and
5
```

Two-Pass Assembler