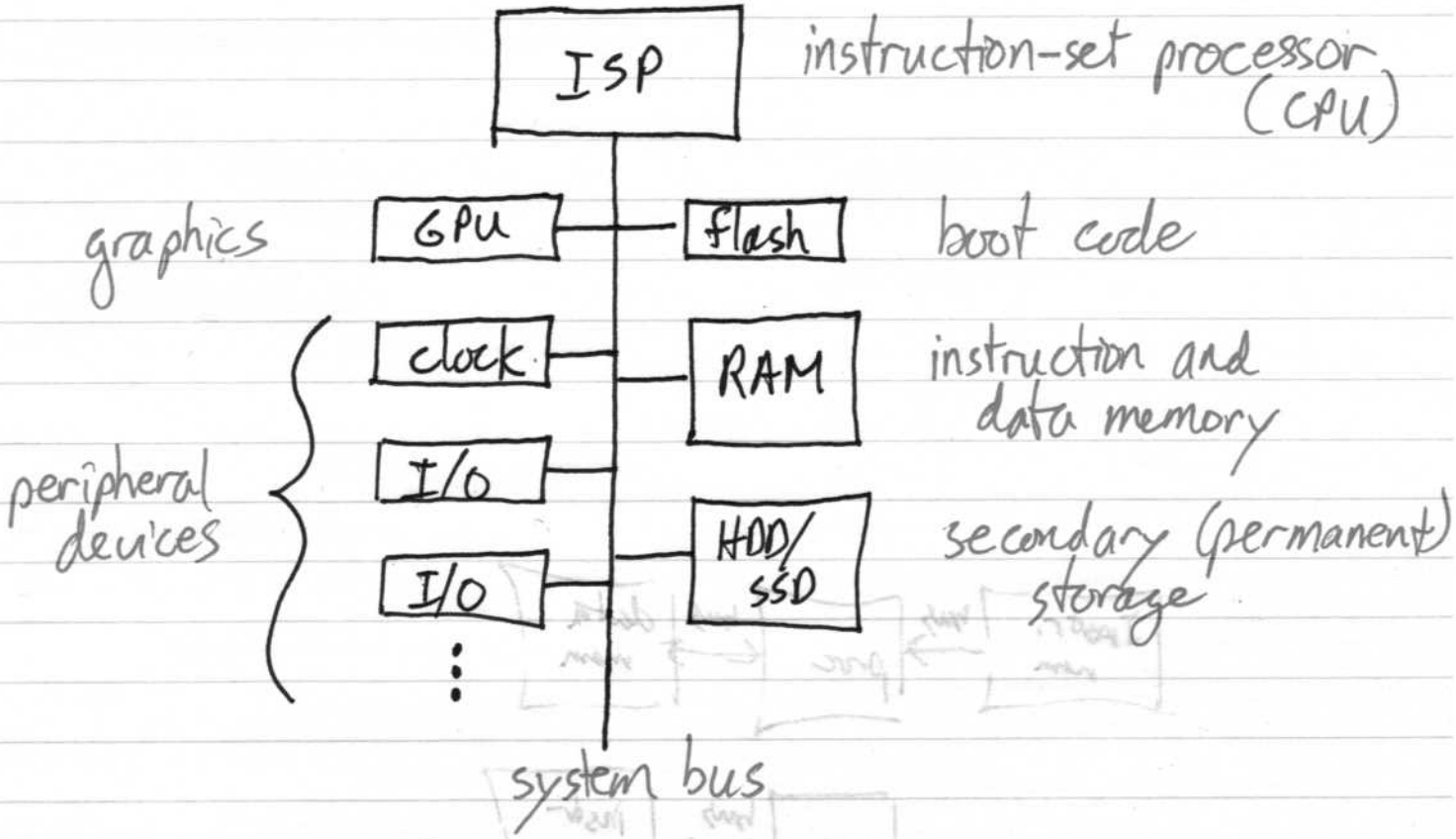
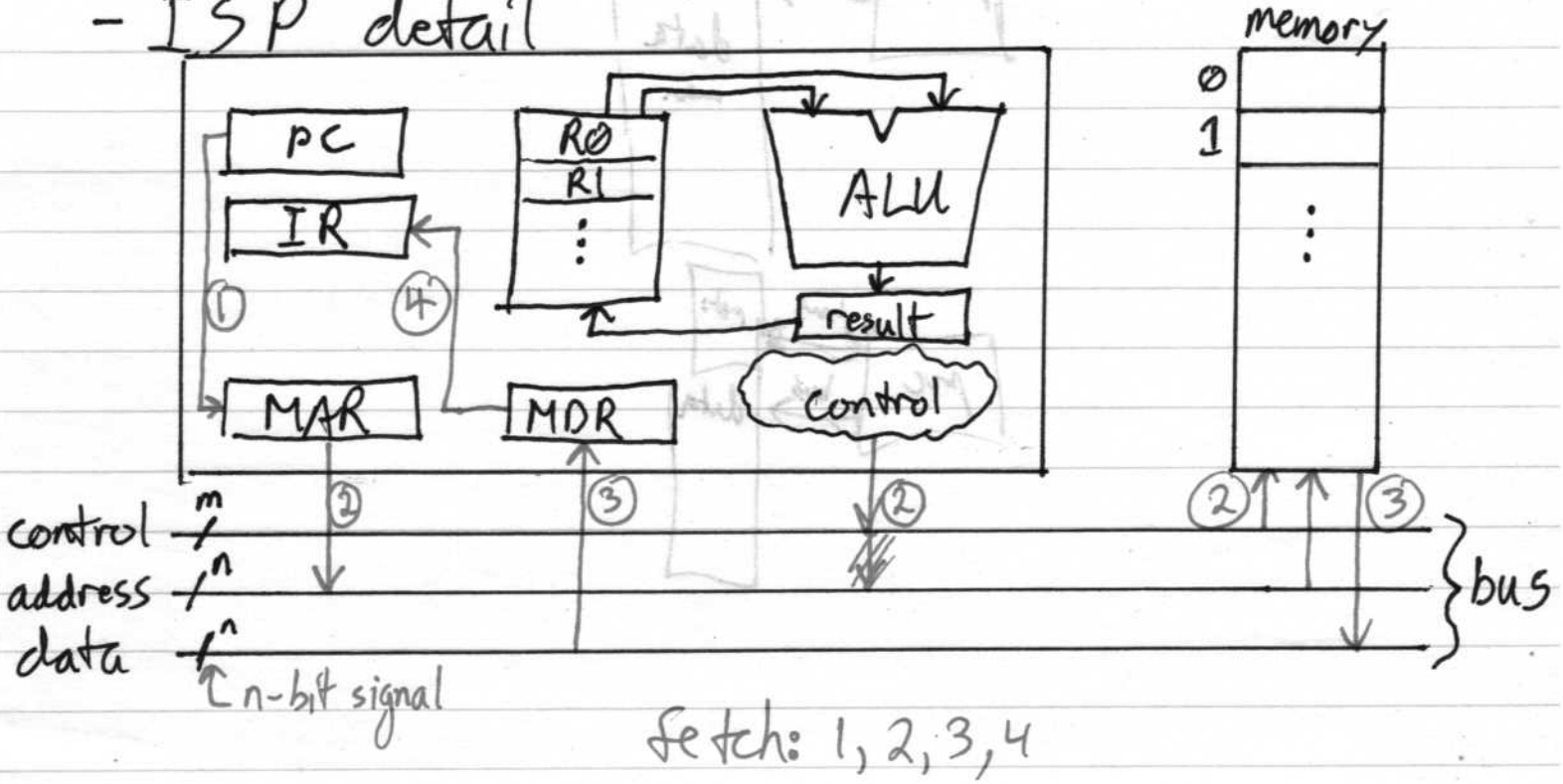
Computer Systems

1) Structure

[3.2, 3.4]



- ISP detail



fetch: 1, 2, 3, 4

PC = Program Counter
- stores next instruction address

IR = Instruction Register
- holds current instruction for decoding

MAR = Memory Address Register
- outputs address to memory

MDR = Memory Data Register
- holds data to/from memory

Harvard Architecture vs Von Neumann Architecture
- separate memories for instructions and data
- one memory for instructions and data

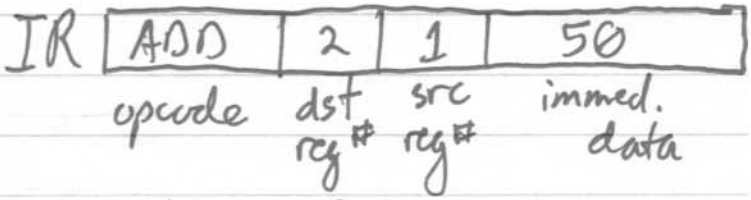
- instruction execution

destination register
source operand register
immediate operand

e.g. ADD R2, R1, #50

F: fetch (instruction) and increment PC
MAR ← PC, bus read, IR ← MDR
PC ← PC+4

D: decode (figure out what to do)



C: compute
result ← R1 + 50

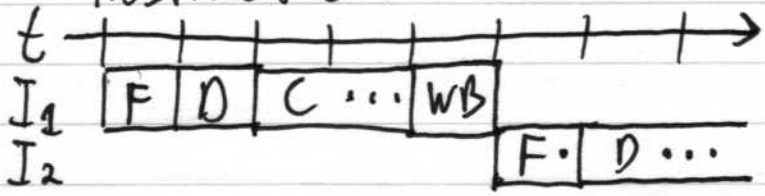
WB: write back (result to a register)
R2 ← result

- ISA : Instruction Set Architecture,
  - describes processor functionality (instruction encoding, memory model, register set, exception handling)
  - abstraction layer between hw and sw

- ISA types

CISC

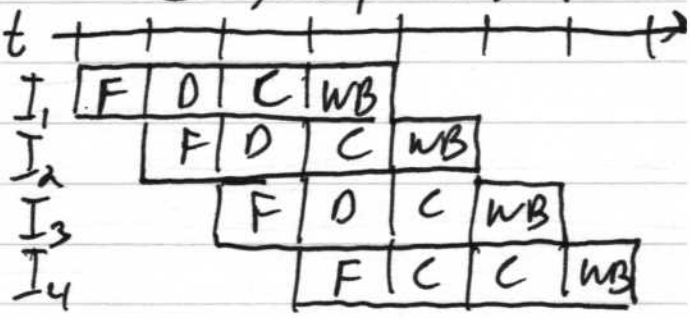
- complex instruction-set computer
- popular 1960s - mid 1980s
- variable-length instructions
- each instruction can do a lot e.g. string copy
- shorter programs
- multiple cycles per instruction



e.g. x86

RISC

- reduced instr.-set comp.
- introduced mid-80s
- fixed-length instructions
- simple instructions
- longer programs
- pipelined execution ⇒ 1 cycle per instr.



e.g. ARM 3-stage
IBM Power 9 12-stages

2) Memory

[3.3]

- size prefixes

SI (~ systeme internationale)		magnitude	IEC (~ international electrotechnical commission)		magnitude
k	kilo	10^3	ki	kibi	$2^{10} = 1024$
M	mega	10^6	Mi	mebi	2^{20}
G	giga	10^9	Gi	gibi	2^{30}
T	tera	10^{12}	Ti	tebi	2^{40}
P	peta	10^{15}	Pi	pebi	2^{50}
E	exa	10^{18}	Ei	exbi	2^{60}

- memory is sized in powers of 2 but frequently (and inaccurately) reported with SI prefixes
  e.g. 1 kB should be 1 kiB = 1024 bytes
- hard drives (HDD) are sized in powers of 10 and (accurately) reported with SI prefixes
  e.g. 1 TB HDD has 10^{12} bytes (not 2^{40} bytes)
- ordering of multi-byte data in memory

  e.g. 0x 12 34 AB CD
  - 12: most significant byte (MSB)
  - CD: least significant byte (LSB)

Big-Endian

0	12 (MSB)
1	34
2	AB
3	CD (LSB)
	...

Little-Endian

0	CD (LSB)
1	AB
2	34
3	12 (MSB)
	...

2.1) Storage (Memory) Technology

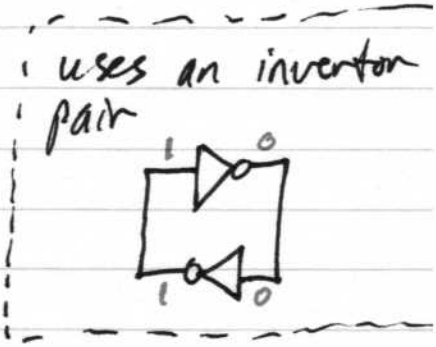
- non-volatile memory retains data when powered off

e.g. ROM read-only memory
- contents are programmed at the foundry
- used for startup code (BIOS)

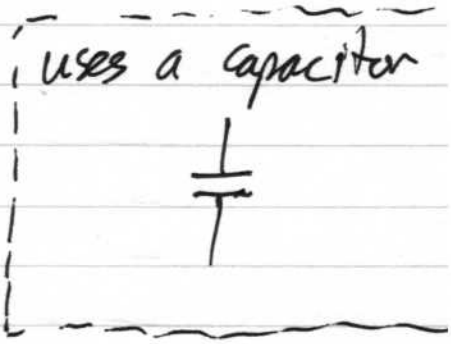
e.g. Flash reprogrammable memory
- can only be reprogrammed in blocks of 16-128 kiB
- also used for startup code (e.g. in LPC1768)

- volatile memory loses data when powered off

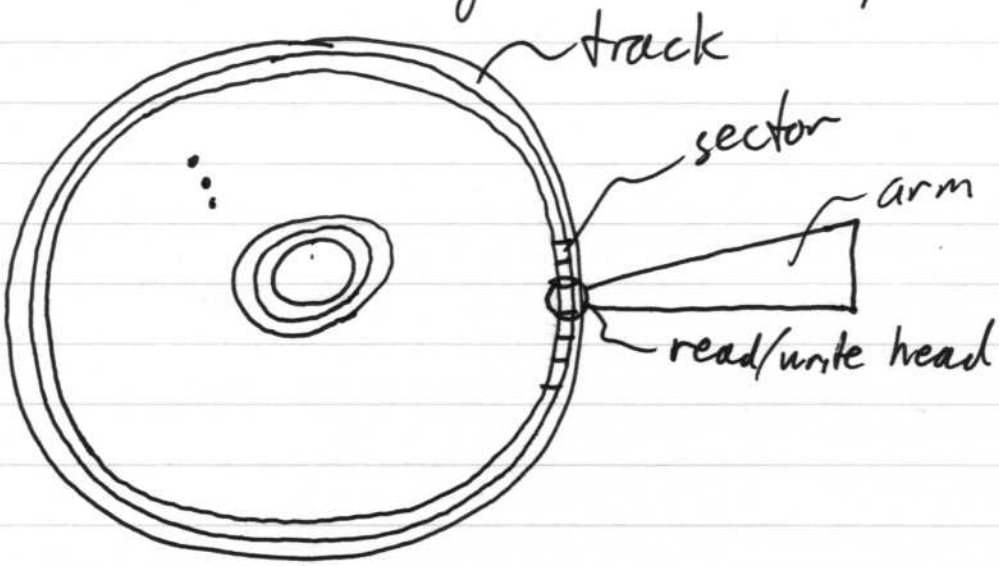
e.g. SRAM static RAM
- a fast, expensive memory
- used in caches



e.g. DRAM dynamic RAM
- a slower, cheaper memory
- used for main memory



- hard disk drives (HDD)
  - data is stored on spinning magnetic platters
  - the read/write head seeks to the correct track, then waits for the right sector to spin under it

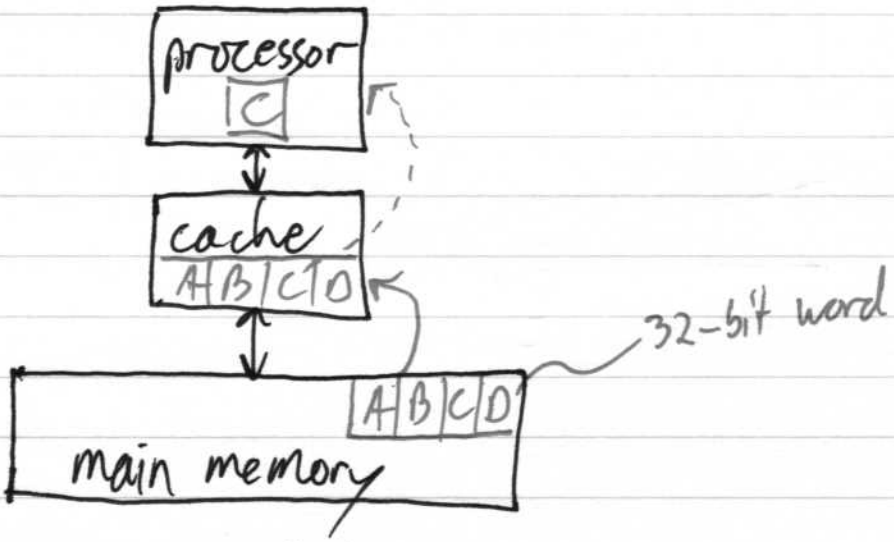


  - data is stored in sectors of 512 B or 4 kiB
  - HDDs are very slow (~10 ms latency) but have large capacity and high bandwidth

2.2) Cache Memories [19]

- main memory is large but has high latency (~ 100s clock cycles)
- caches are smaller, faster memories storing a subset of data from main memory

e.g.



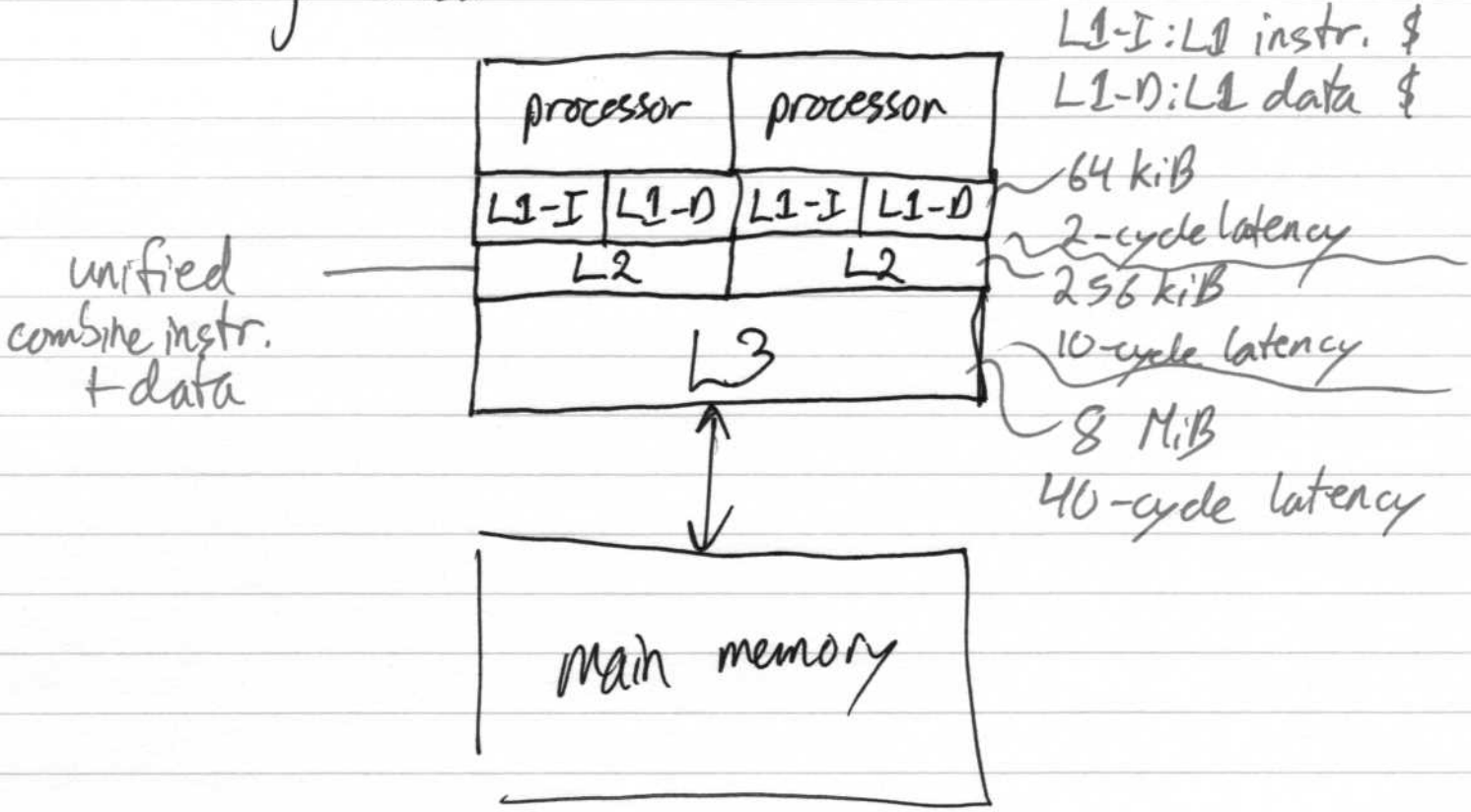
- based on the idea that if a processor needs data from memory:
  - a) it will need the same data again soon (temporal locality)
  - b) it will need other data nearby (spatial locality)

- operation:
  - stores recently used data
  - retrieves data from memory in blocks (this takes advantage of memory's high bandwidth)

- cache blocks are "tagged" with their memory address
- for every processor memory access:
  - it will check the tags for a matching address
    - if found "hit" - returns data
    - if not found "miss" - retrieve block from memory, then return the data

- there are often multiple levels of cache
  e.g. L1, L2, L3
  - smaller caches are faster (L1)
  - larger caches have higher hit rates (L3)

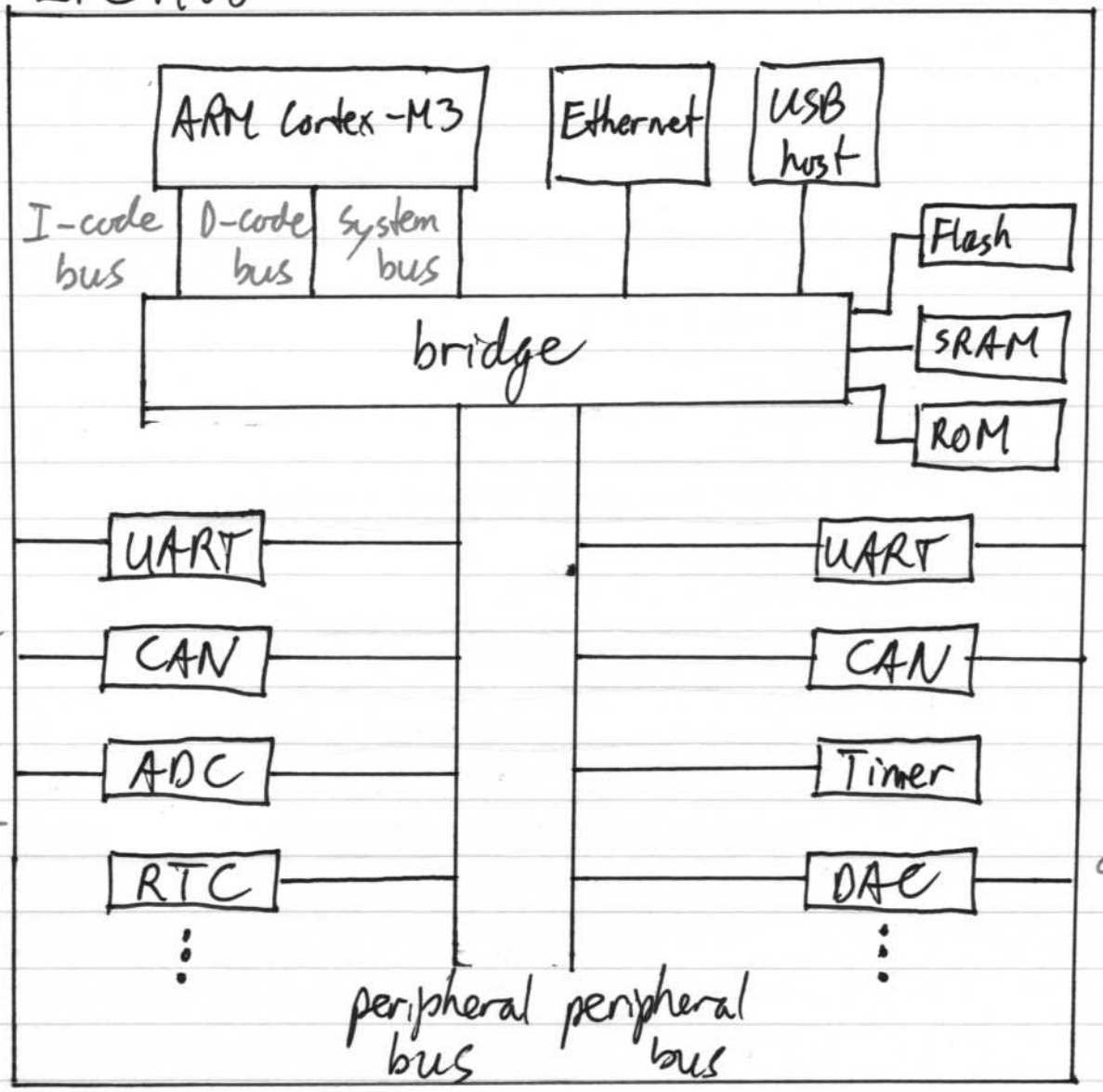
e.g. dual-core



3) Microcontroller Unit (MCU)

- an MCU integrates the processor, memory, real-time clock (RTC), and I/O devices on one chip

e.g. LPC1768



- I-code bus: for instructions (connects to Flash) } Harvard Architecture
- D-code bus: for data (connects to SRAM) } Harvard Architecture
- System bus: for I/O

- ARM Cortex-M3 processor
  - word size = 32 bits (all registers are 32 bits)
  - it can address 4 GiB of memory (2^{32} = 4Gi)
  - it has 16 general-purpose registers (R0-R15); R15 is the PC
  - it can do 8-bit, 16-bit and 32-bit data transfers
  - no cache