04 Computer systems

04.02 Instruction-set architecture

- Classification criteria
- Instruction set
- Execution model
- Addressing

Classification criteria

- Instruction set
  - Size
  - Instruction format
- Execution model
  - Stack, Mem-Mem, Mem-Reg, Reg-Reg
- Addressing mode
  - Ordering
  - Alignment
  - Address computation

Instruction-Set

- RISC
  - All instructions have the same size
  - The OpCode has a fixed position and size within the instruction
  - There are only a few different formats
  - There is a single format per OpCode
- CISC
  - Different instructions may have different size
  - The OpCode has variable position and size
  - There are many different formats

Execution model (1)

- STACK
  - A stack is a first-in-last-out queue
  - Internal registers are handled has a stack
  - Operands are always taken from the top of the stack
  - Results are always put at the top of the stack
  - Source and destination registers do not need to be specified

Execution model (2)

- MEM-MEM
  - A single instruction takes the operands from memory, computes the result and puts it back in memory
  - No internal (general-purpose) registers are required
- MEM-REG
  - At most one operand can be taken directly from memory
  - At least 1 general-purpose register is required
- REG-REG
  - Both operands and results are internal registers

Execution models comparison
Example

- Example: A ← B+C

  Where A, B and C are memory elements (i.e., variables)

MEM-MEM	MEM-REG	MEM-REG-1	REG-REG	STACK
ADD A,B,C	LOAD R1,B ADD R3,R1,C STORE A,R3	LOAD B ADD C STORE A	LOAD R1,B LOAD R2,C ADD R3,R1,R2 STORE A,R3	PUSH B PUSH C ADD POP A
A <-- B+C	R1 <-- B R3 <-- R1+C A <-- R3	R1 <-- B R1 <-- R1+C A <-- R1	R1 <-- B R2 <-- C R3 <-- B+C A <-- R3	

PUSH B	PUSH C	ADD	POP A
	C		
B	B	B+C	

Addressing

Byte ordering

- Memory addresses are assigned with bytes
- Handling single bits requires Boolean masks
- Words of 2, 4, 8 bytes can be handled as a whole (i.e., by a single instruction)
  - In this case a single address is specified for the entire word and a criterion is required to retrieve all the bytes
  - *Little Endian*: the word address is the address of the least significant byte of the word
  - *Big Endian*: the word address is the address of the most significant byte of the word

Addressing

Alignment

- 32-bit memories make 4 bytes addressable as a whole as long as the address is divisible by 4 (i.e., the word is *aligned* with the memory)
- Misaligned words require 2 bus cycles to be completely read/written

Little Endian

3	2	1	0	i
3	2	1	0	i+4
3	2	1	0	i+8
3	2	1	0	i+12
D31-24	D23-16	D15-8	D7-0	

Big Endian

i	0	1	2	3
i+4	0	1	2	3
i+8	0	1	2	3
i+12	0	1	2	3
	D31-24	D23-16	D15-8	D7-0

Addressing

Address computation

Direct

LD R1,var R1 ← M[var]

Indirect

LD R1,(R2) R1 ← M[R2]

Relative to registers

LD R1,var(R2) R1 ← M[R2+var]

Relative to registers, *indexed* and *scaled*

LD R1,var(R2)(RX)

R1 ← M[R2+var+RX*d]