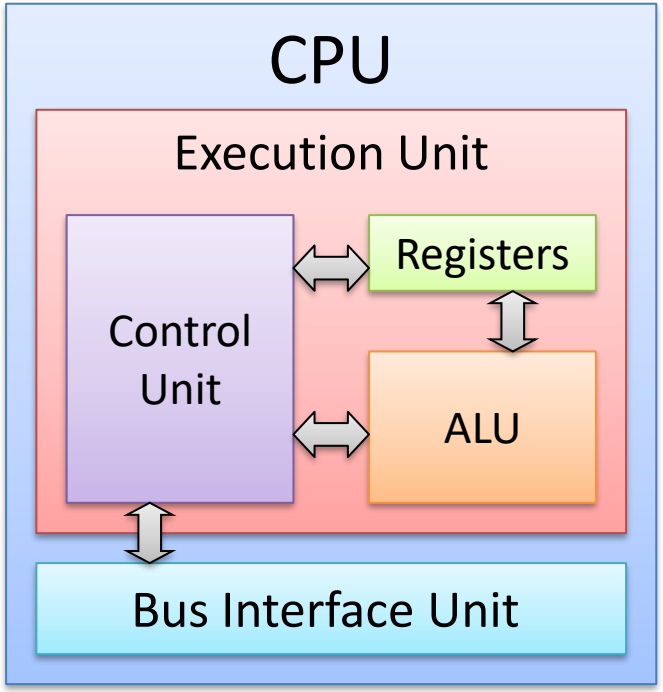
# Microprocessors

Embedded Systems

Wolfgang Neff

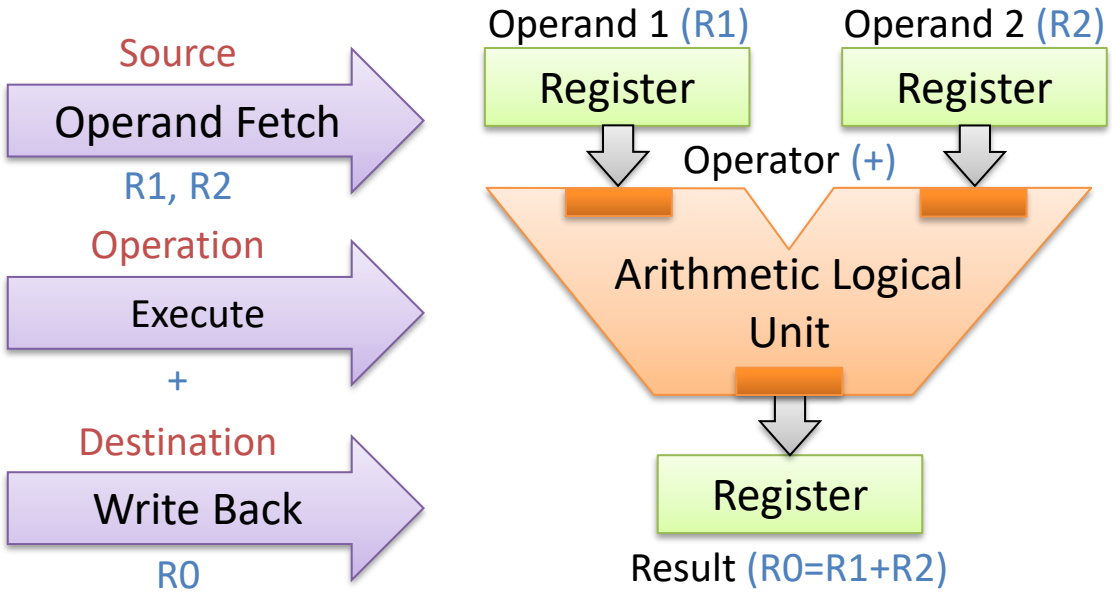
# Central Processing Unit (1)

- Block diagram
  - Execution Unit
    - Control Unit
    - Registers
    - Arithmetic logic unit
      - ADD, SUB etc.
      - NOT, AND etc.
  - Bus Interface Unit



# Central Processing Unit (2)

- Arithmetic logic unit

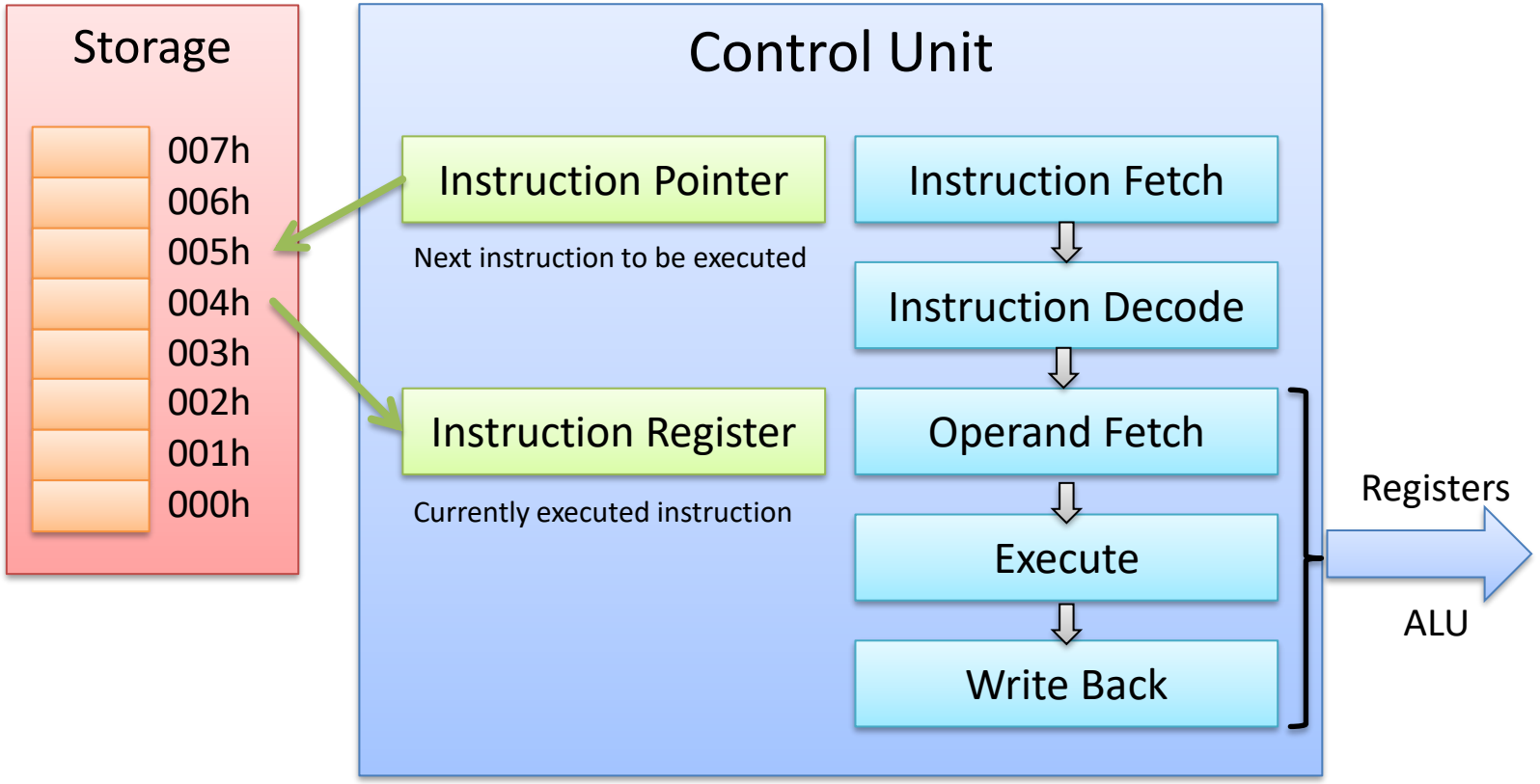



Result = Operand1 *Operator* Operand2

Destination = Source1 *Operator* Source2

# Central Processing Unit (3)

- Structure of Control Unit



# Central Processing Unit (4)

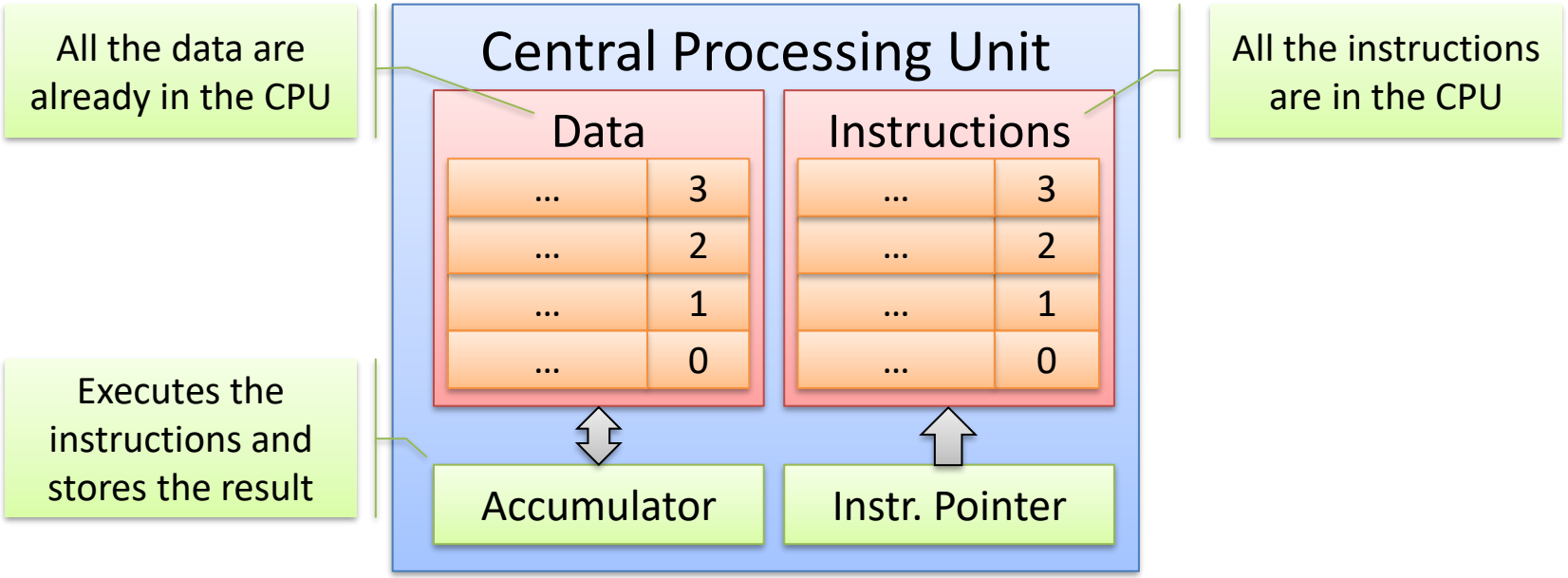
- Instruction pipeline

| Instruction | Pipeline stage | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | IF | ID | OF | EX | WB | | |
| 2 | | IF | ID | OF | EX | WB | |
| 3 | | | IF | ID | OF | EX | WB |
| 4 | | | | IF | ID | OF | EX |
| 5 | | | | | IF | ID | OF |
| **Clock** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |

Parallel execution of five instructions

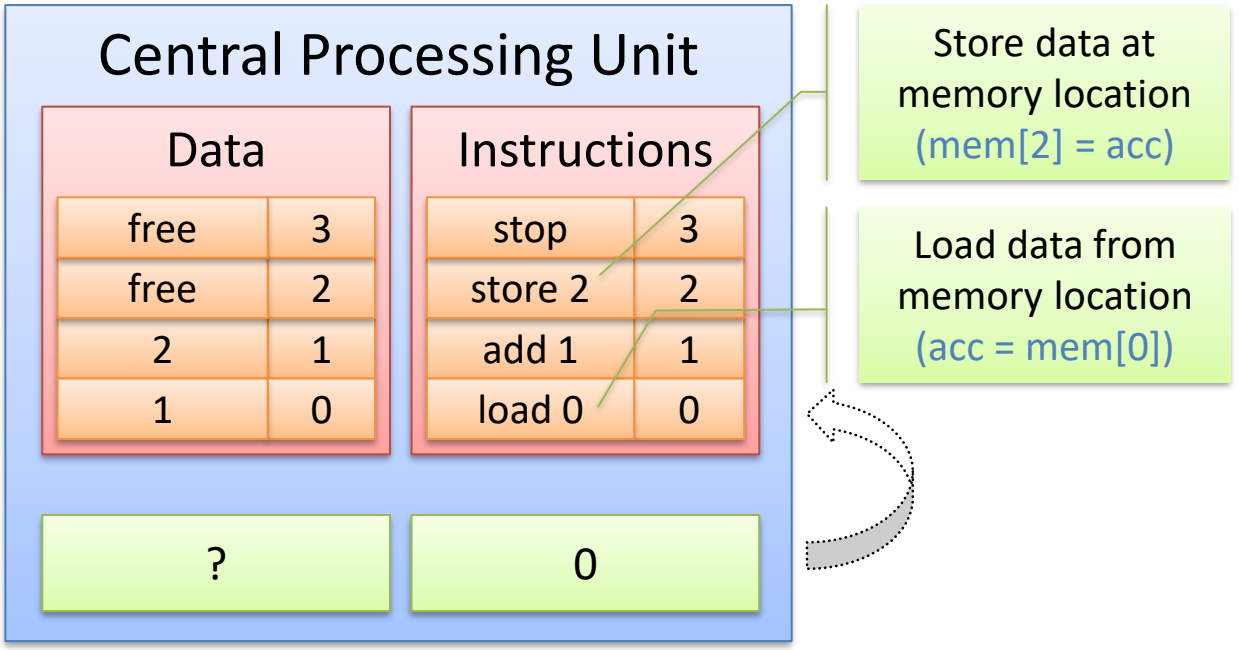
# Register Machines (1)

- Theoretical model



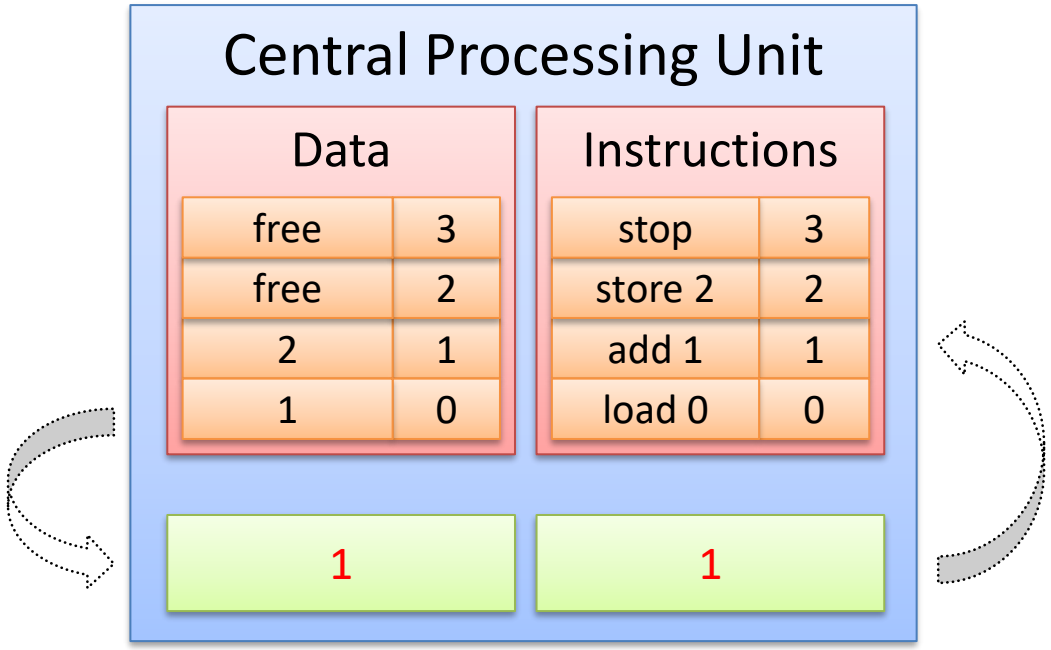
# Register Machines (2)

- Program Execution



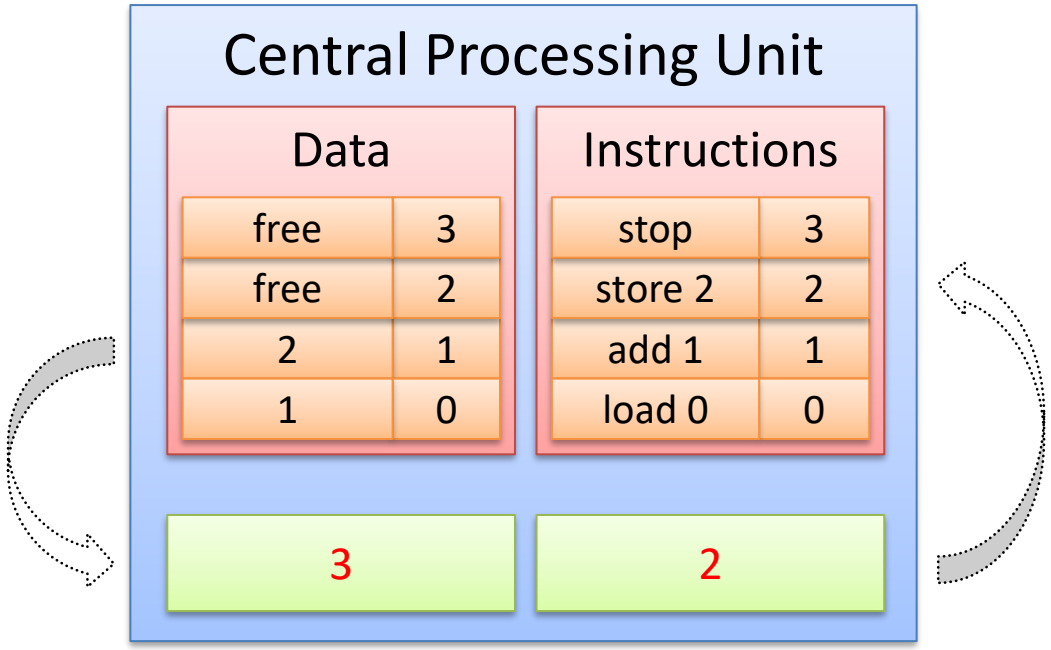
# Register Machines (2)

- Program Execution (continued)



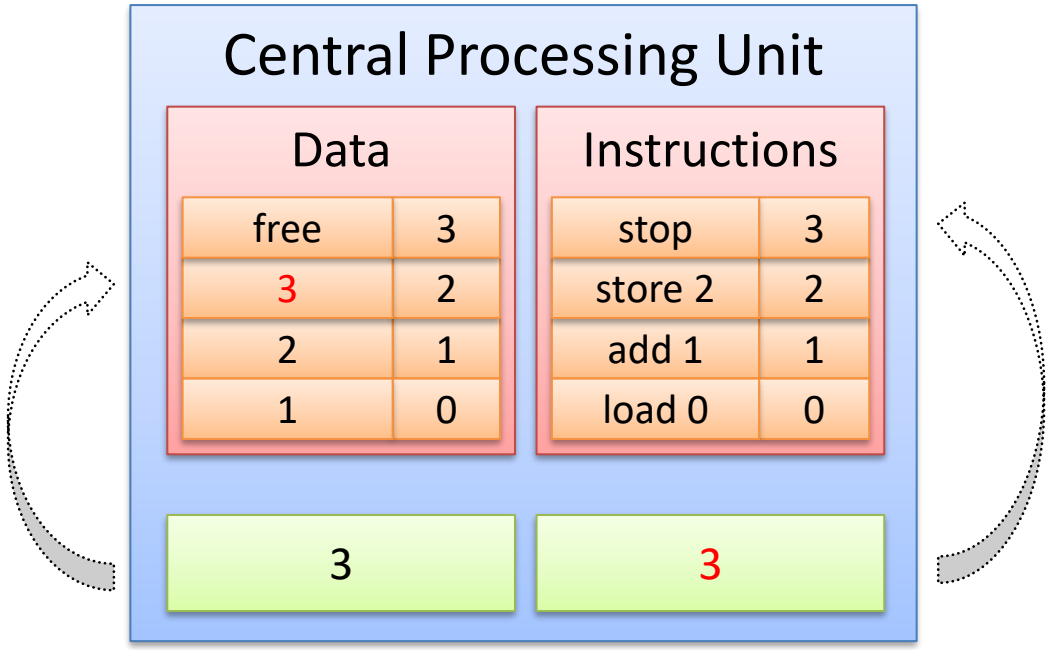
# Register Machines (2)

- Program Execution (continued)



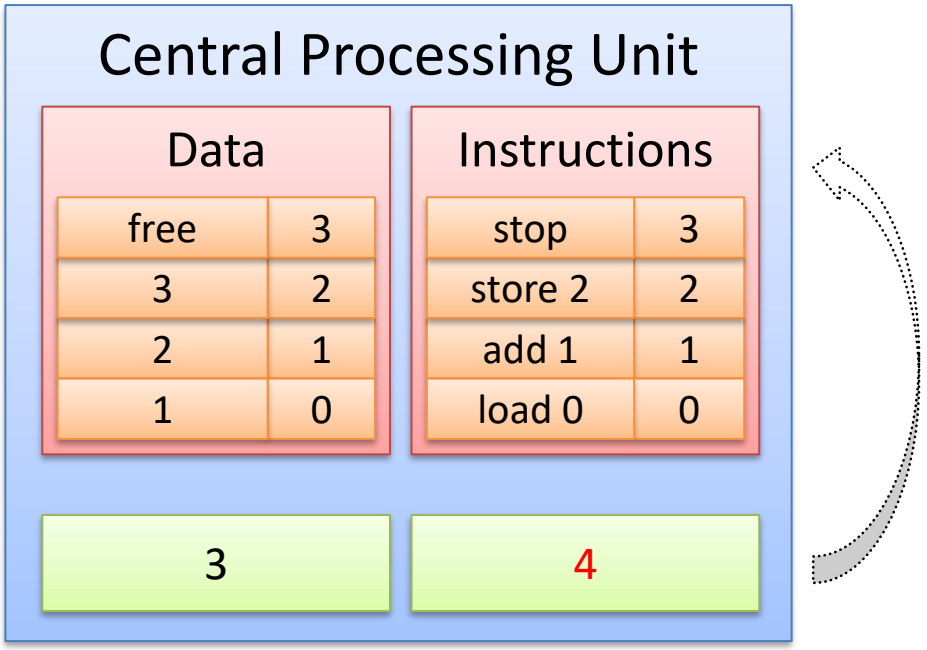
# Register Machines (2)

- Program Execution (continued)



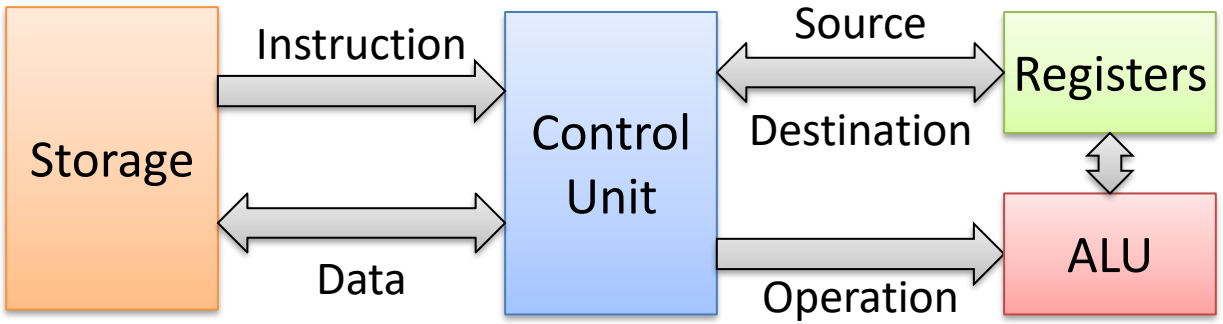
# Register Machines (2)

- Program Execution (finished)



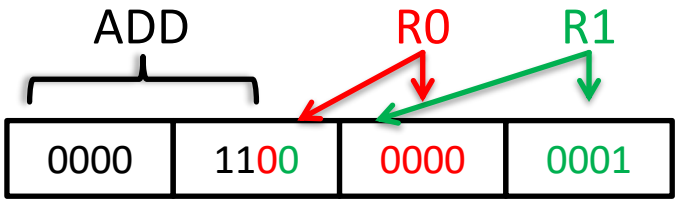
# Instruction Set (1)

- Instructions
  - Statements of a program
  - Stored storage
  - Specify operation, source and destination



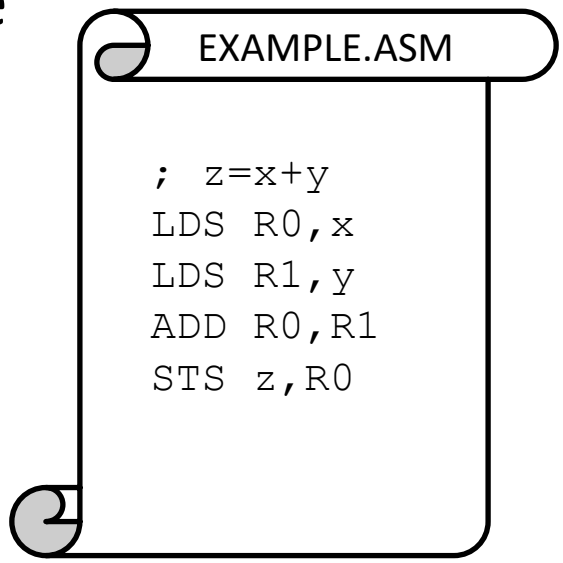
# Instruction Set (2)

- Machine Language
  - Specific for every microprocessor
  - Set of instructions understood by these processors
  - Represented by numeric operation codes
  - Example:
    - Operation: ADD
    - Source: R0, R1
    - Destination: R0
    - Opcode: $0C01_{hex}$



# Instruction Set (3)

- Assembly Language
  - Opcodes replaced by textual mnemonics
  - Source code is readable
  - Has to be compiled
  - Major benefits:
    - Easier to read
    - Comments allowed
    - Variables names
    - Labels

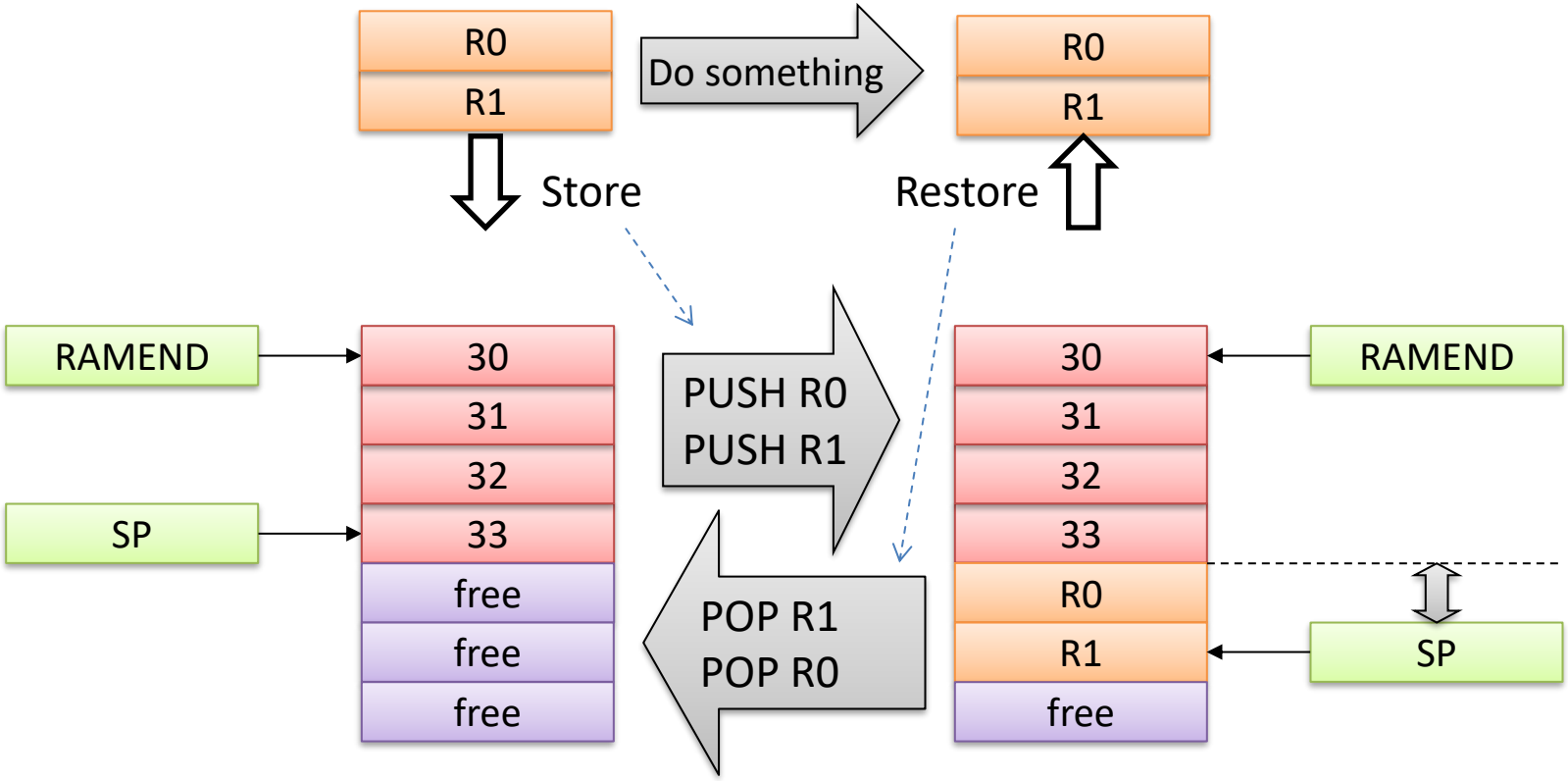


# Instruction Set (4)

- Instruction Set
  - Instructions for the ALU
    - Arithmetic Instructions: *ADD*, *SUB*, *MUL*, *CP*, …
    - Logical Instructions: *AND*, *OR*, *EOR*, *COM*, …
    - Shift and Rotate Instructions: *LSL*, *LSR*, *ROL*, *ROR*, …
    - Bit Manipulation Instructions: *SBI*, *CBI*, *CLI*, *SEI*, …
  - Instructions for the Control Unit
    - Data Movement Instructions: *MOV*, *LD*, *ST*, *PUSH*, …
    - Branch Instructions: *RCALL*, *RET*, *BRcc*, …

# Stack (1)

- Operating mode



# Stack (2)

- Summary
  - Used to store temporary data
  - Is a last-in-first-out (LIFO) memory
  - Grows from up to down
  - Stack pointer points to the last element
  - Two special instructions
    - PUSH: Decrement Stack pointer, put element on Stack
    - POP: Get element from Stack, increment Stack pointer
    - PUSH/POP: Pre-decrement / post-increment