William Stallings
Computer Organization
and Architecture
7th Edition

Chapter 17
Micro-programmed Control

Key points

- Microprogram is an *alternative* implementation of control unit
- Microprogrammed control unit is capable of ***sequencing*** through microinstructions and ***generating*** control signals
- Control signals are used to cause *register transfer* and *ALU operations*

History of microprogram

- First coined by M.V.Wilkes in the ealy 1950s
- Aim to *avoid the complexities* of hardwired implementation of control unit
- *Limited by speed and price* of control memory
- Reviewed in 1964
- IBM’s system/360
- Popular since then

17.1 Basic concepts

- Microinstructions
- Micro-programmed control unit
- Wilkes control
- Advantages and disadvantages

Microinstructions

- Micro-program: a sequence of *microinstructions*
- Use sequences of instructions to control complex operations
- Control signals are generated by *micro-operations*
- *Microinstructions* describes micro-operations

Control signals

- Control unit generates a set of control signals
- ✓ Each control signal is *on or off*
- ✓ Represent each control signal by *a bit*

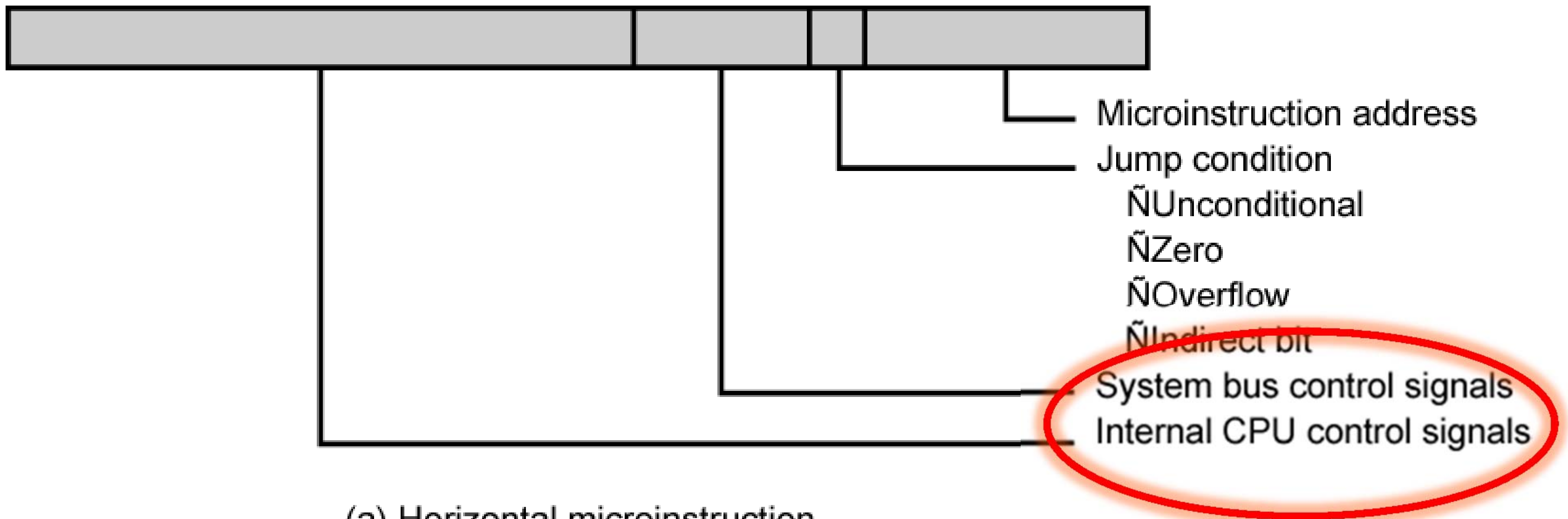
Implementation of micro-instruction

- Have a *control word* for each micro-operation
- Have *a sequence of control words* for each machine code instruction
- Put control words in a *memory*, each with a unique *address*
- Add an address to specify the *next* micro-instruction, depending on conditions

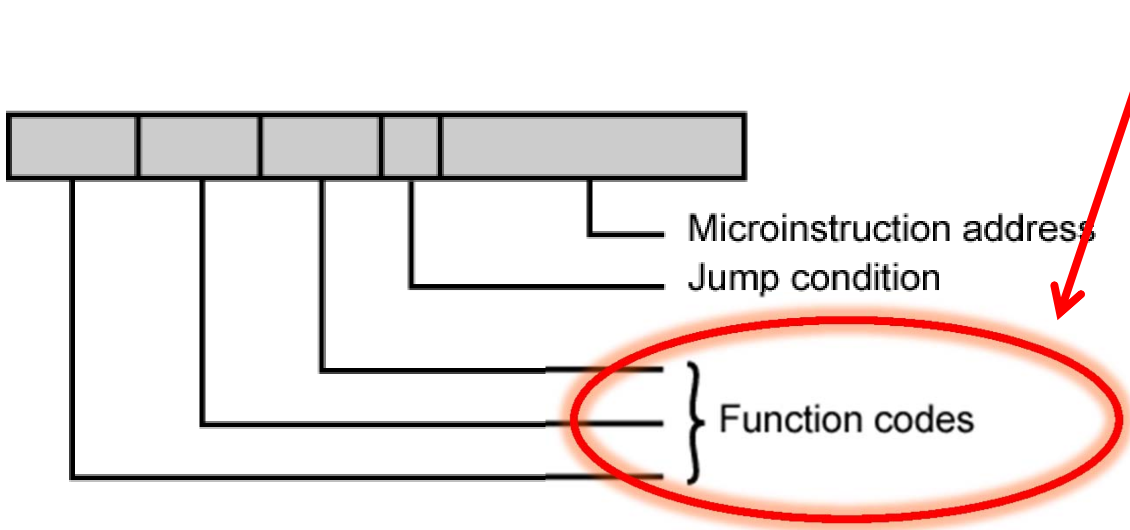
Micro-instruction formats

- ***Horizontal***: Each micro-instruction specifies many different micro-operations to be performed in parallel
    - —*One bit* for each internal processor control line
    - —*One bit* for each system bus control line
    - —A condition field
    - —A field with the address to be executed next when a branch is taken
- ***Vertical***: Each micro-instruction specifies single (or few) micro-operations to be performed
    - —*Function codes*
    - —A condition field (*the same to Horizontal*)
    - —A field with the address to be executed next when a branch is taken (*the same to Horizontal*)

Typical Microinstruction Formats

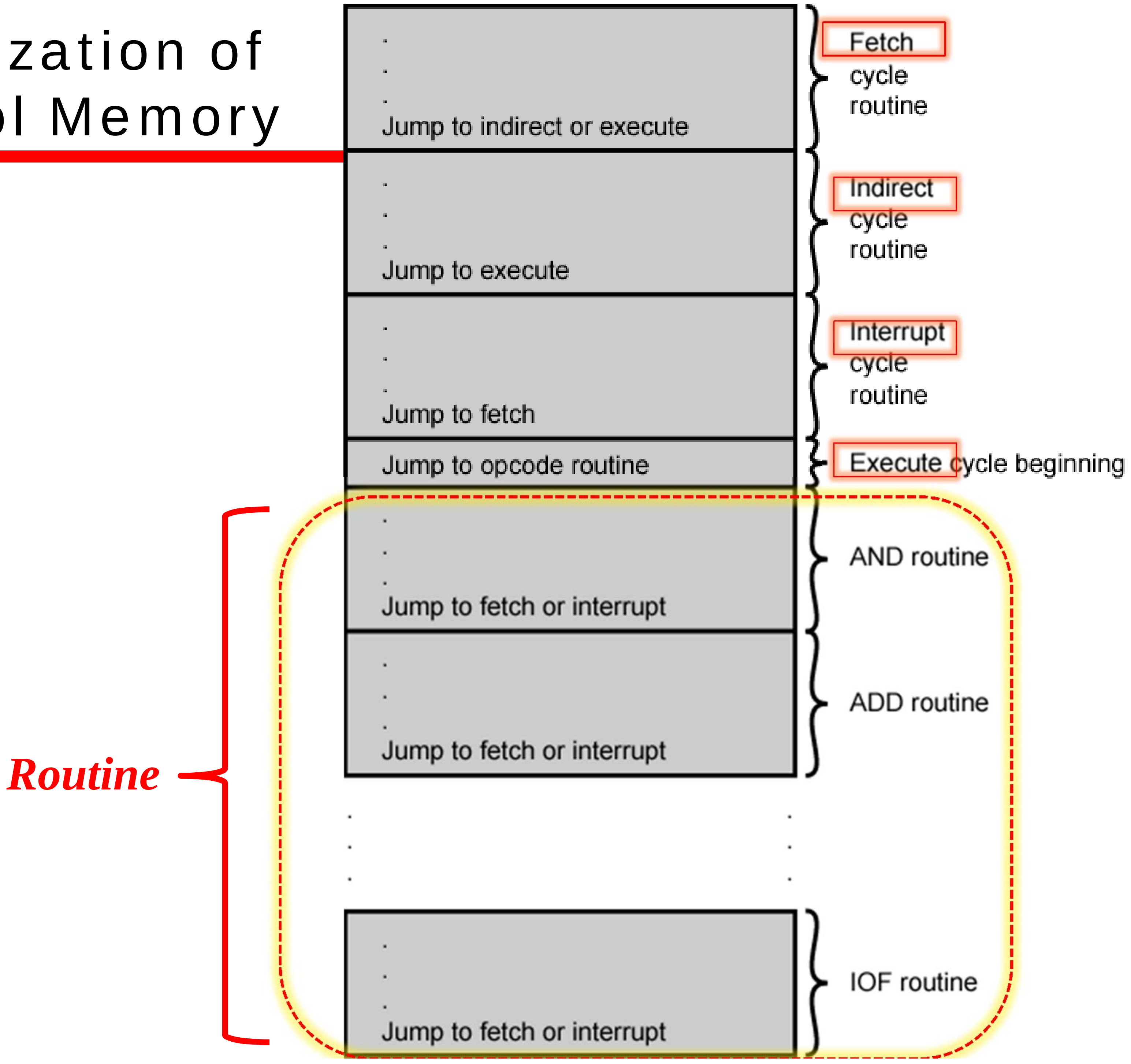


(a) Horizontal microinstruction

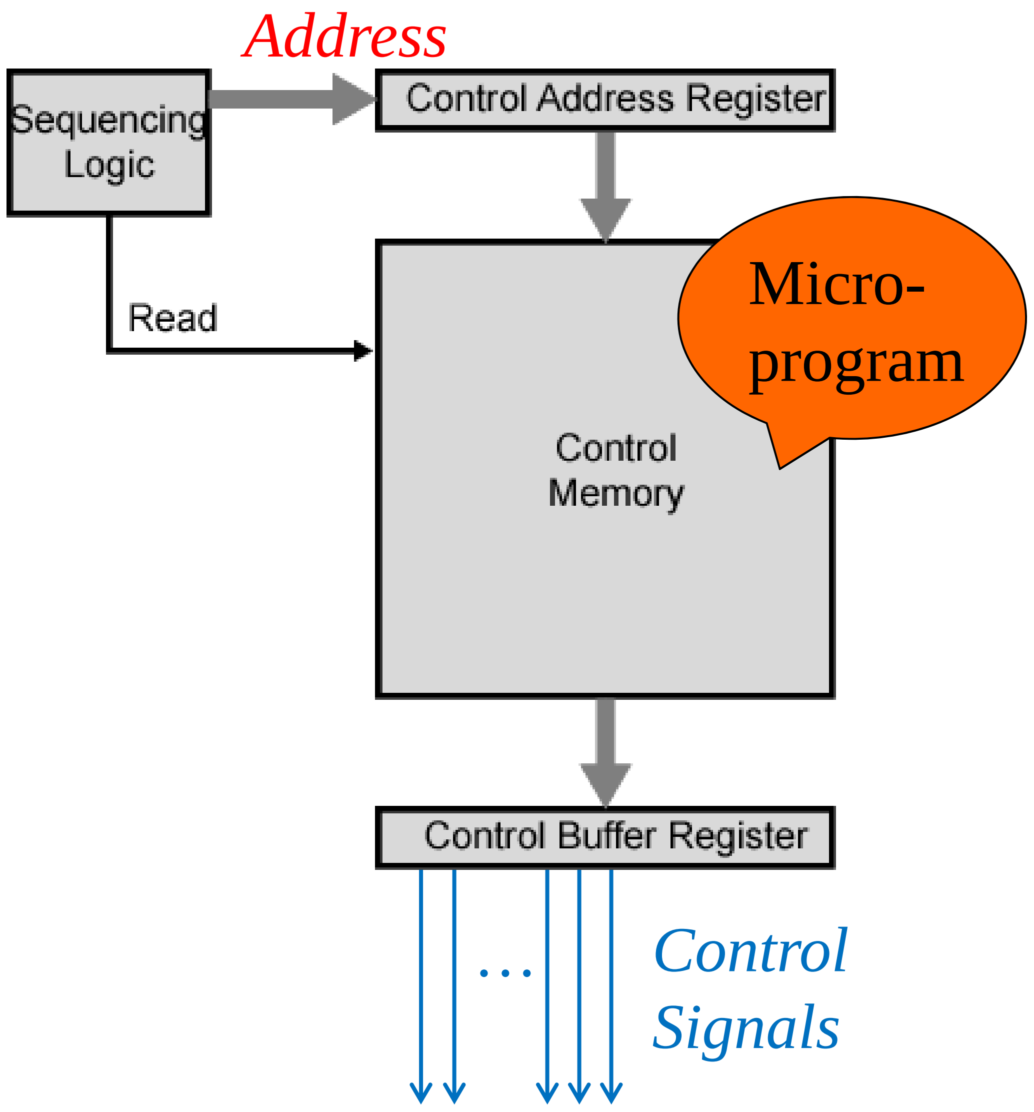


(b) Vertical microinstruction

Organization of Control Memory



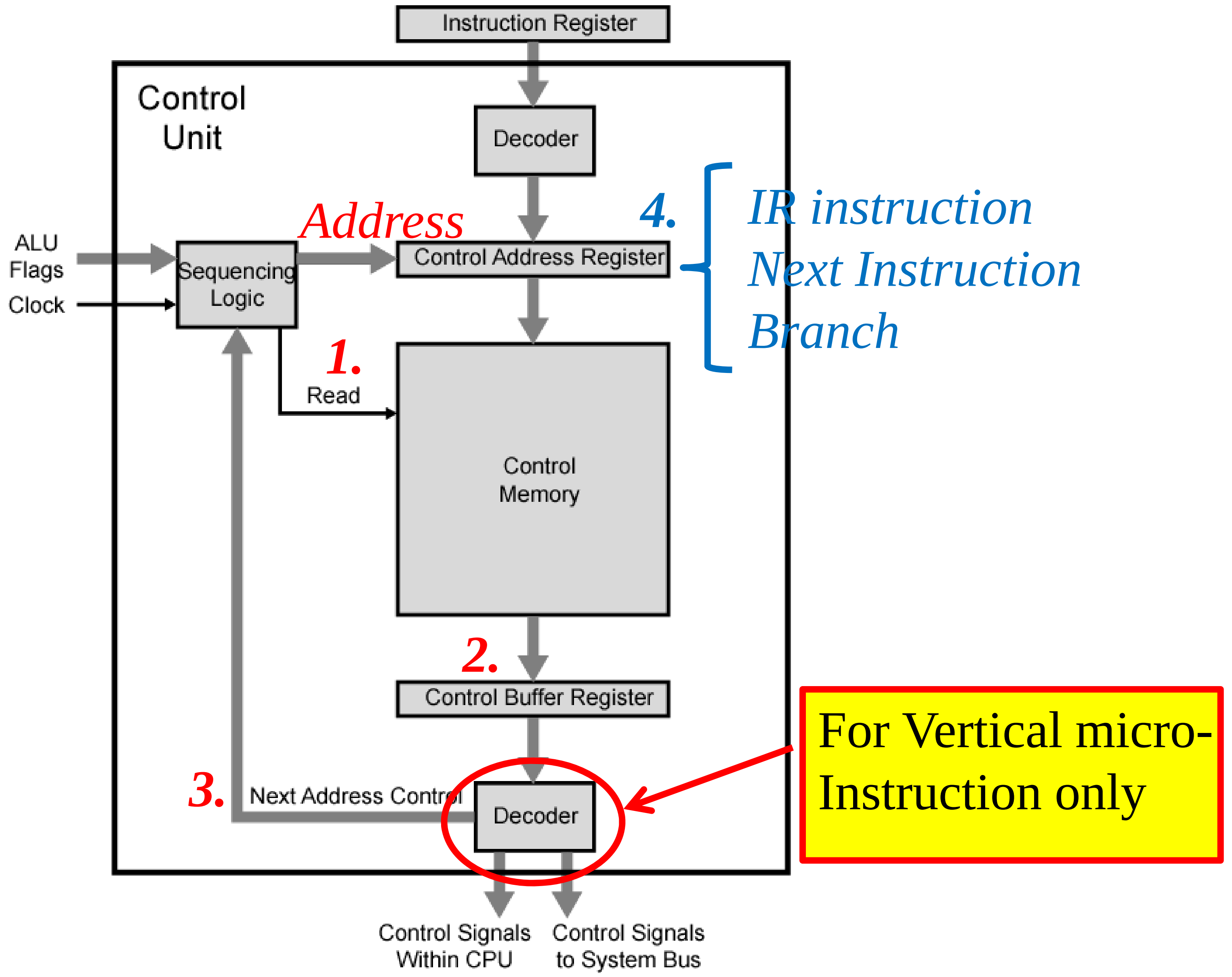
Microprogrammed Control Unit



Next Address Decision

- Depending on *ALU flags* and *control buffer register*
    - Get next instruction
        - Add 1 to control address register
    - Jump to new routine based on jump microinstruction
        - Load address field of control buffer register into control address register
    - Jump to machine instruction routine
        - Load control address register based on opcode in IR

Functioning of Microprogrammed Control Unit

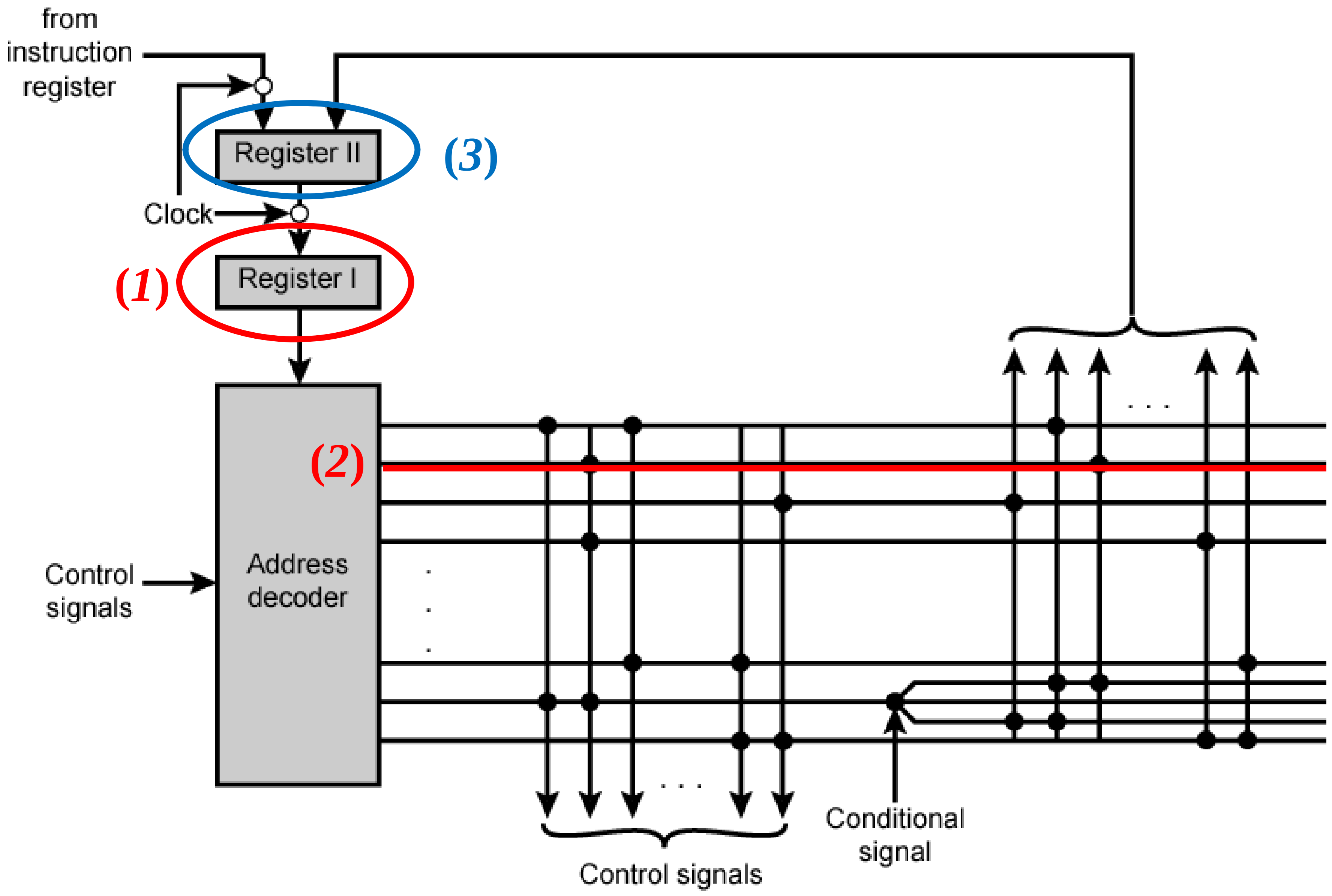


For Vertical micro-Instruction only

Wilkes Control

- 1951
- Matrix partially filled with *diodes*
- During cycle, one row activated
  - Generates signals where diode present
  - First part of row generates *control signal*
  - Second generates *address* for next cycle

Wilkes's Microprogrammed Control Unit



Advantages and Disadvantages of Microprogramming

- Simplifies design of control unit
- ➢ *Decoder & Sequencing* mainly compose of logic units.
  - —Cheaper
  - —Less error-prone
- Slower

➢ CISC: micro-programmed control;

➢ RISC: hard-wired control;

17.2 Microinstruction sequencing

- Two basic tasks of microprogrammed control unit
  - Microinstruction *sequencing*:
    - Get the next microinstruction from the control memory
  - Microinstruction *execution*:
    - Generate the control signals needed to execute the microinstruction

Design Considerations

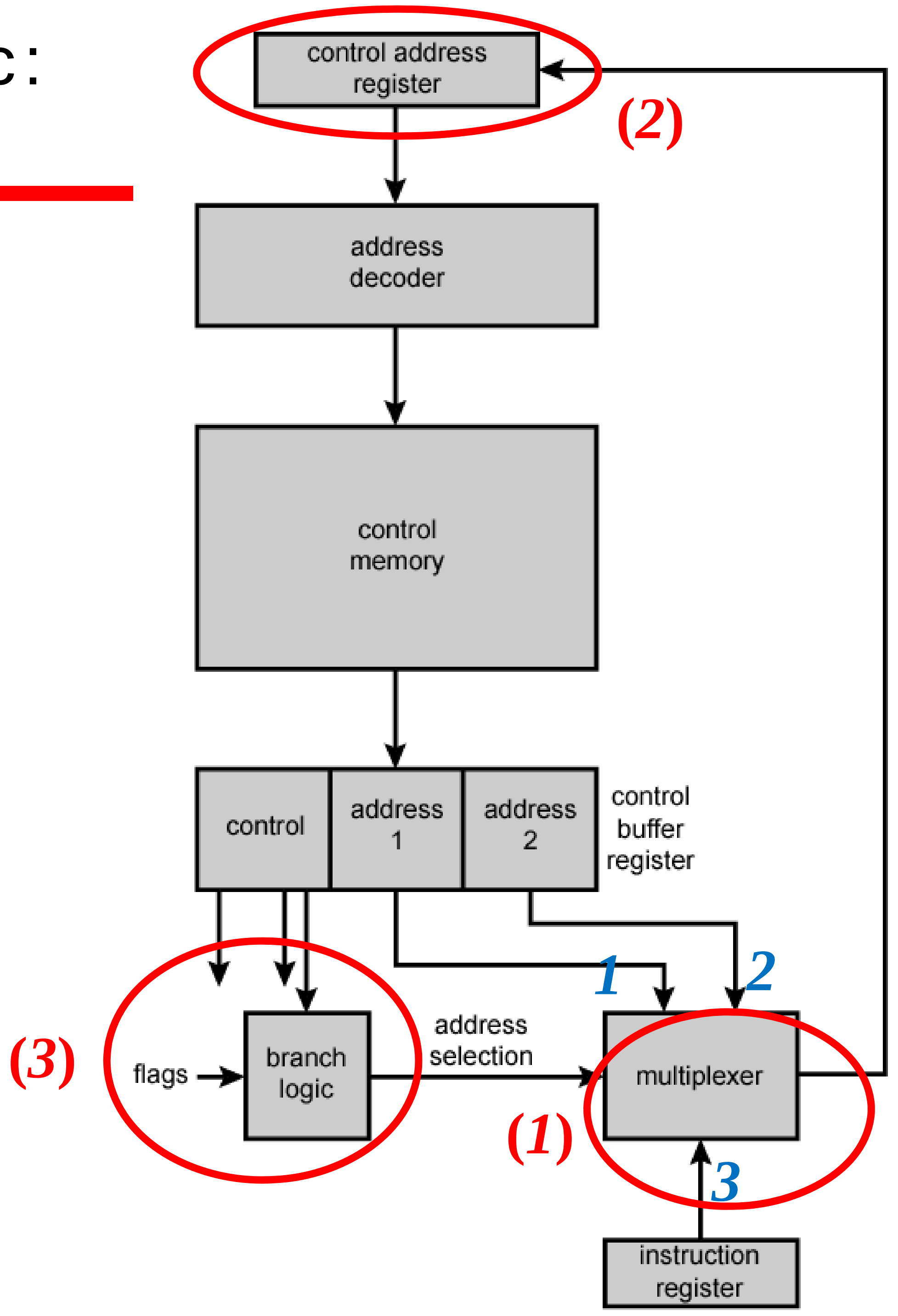
- Size of microinstructions *-> memory size*
- Address generation time *-> a.s.a.p.*
  - Determined by instruction register
    - Once per cycle, after instruction is fetched
  - Next sequential address
    - Common in most designed
  - Branches
    - Both conditional and unconditional
    - Highly frequently, One out of 3 or 4 could be a branch

Sequencing Techniques

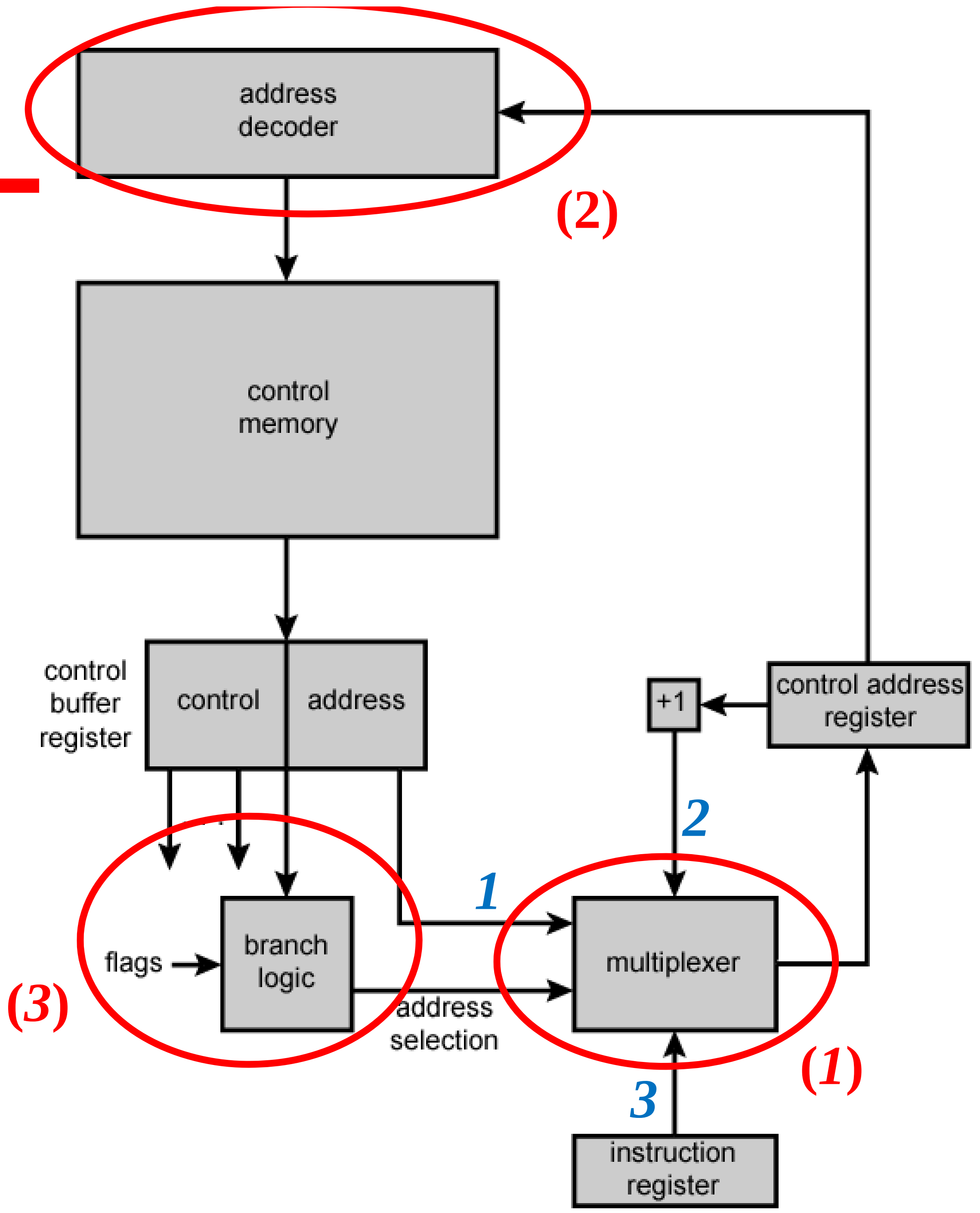
- Sources: Based on *current microinstruction*, *condition flags*, *contents of IR*, control memory address must be generated for the next microinstruction

- Tech: Based on format of address information
  - —*Two* address fields
  - —*Single* address field
  - —*Variable* format

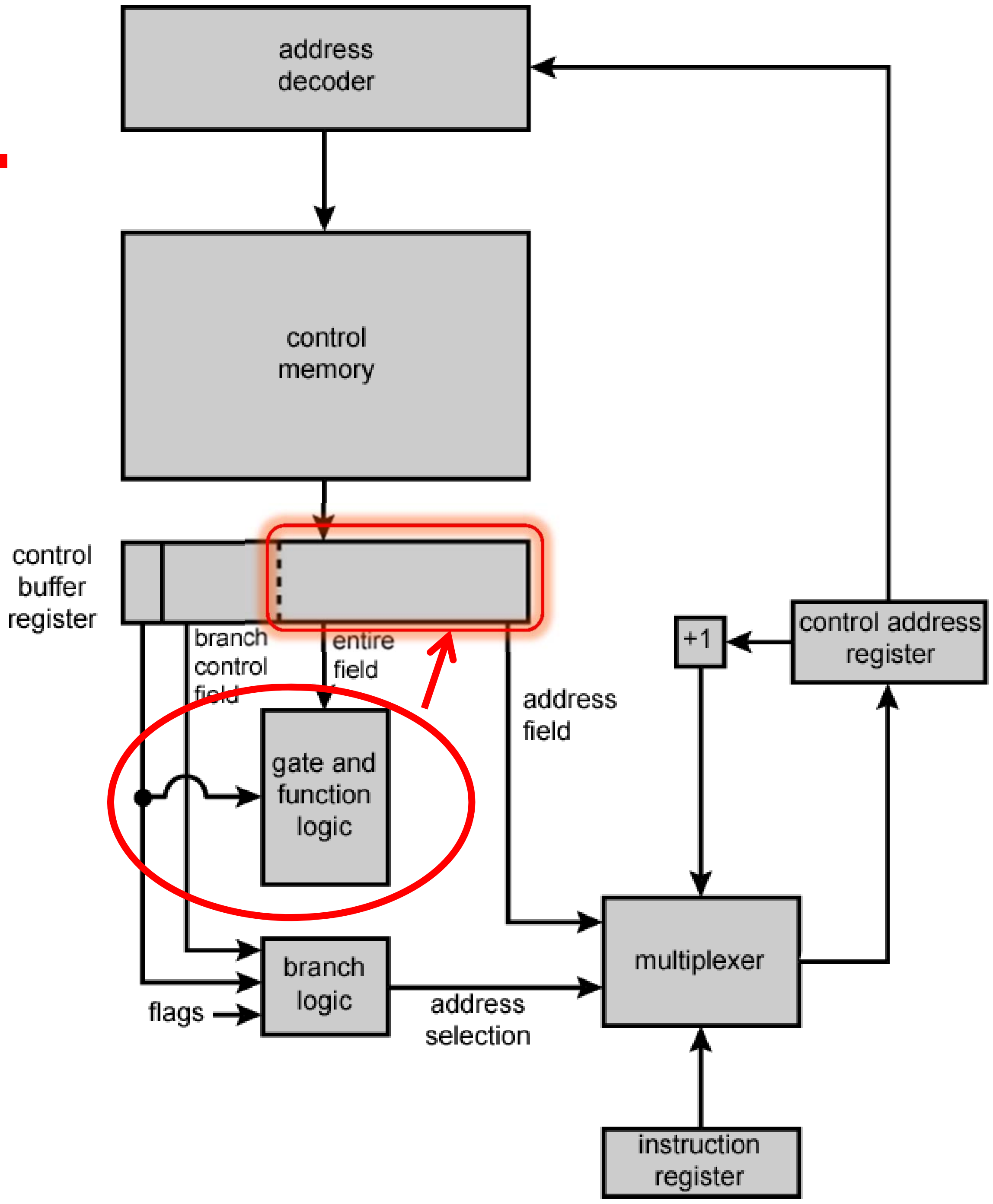
Branch Control Logic: Two Address Fields



Branch Control Logic: Single Address Field



Branch Control Logic: Variable Format



Address Generation

Explicit	***Implicit***
Two-field	Mapping *(inst. To micro-inst.)*
Unconditional Branch	Addition *(combination of two addresses)*
Conditional branch	Residual control *(temp. address saved before)*

Explicit techniques

- Addresses are available in the microinstruction
- Two-field approach: Two alternative addresses are available
- Single address field format: branch instruction
- Variable format: branch instruction
- Information for branch:
  - ALU flags
  - Part of the opcode or address mode fields
  - Parts of a selected register
  - Status bits within the control unit

Implicit techniques

- Require *additional logic* to generate the addresses
- *Mapping*
  - Opcode portion mapped into address
- *Combining or adding*
- *Residual control*
  - Use previously saved microinstruction address

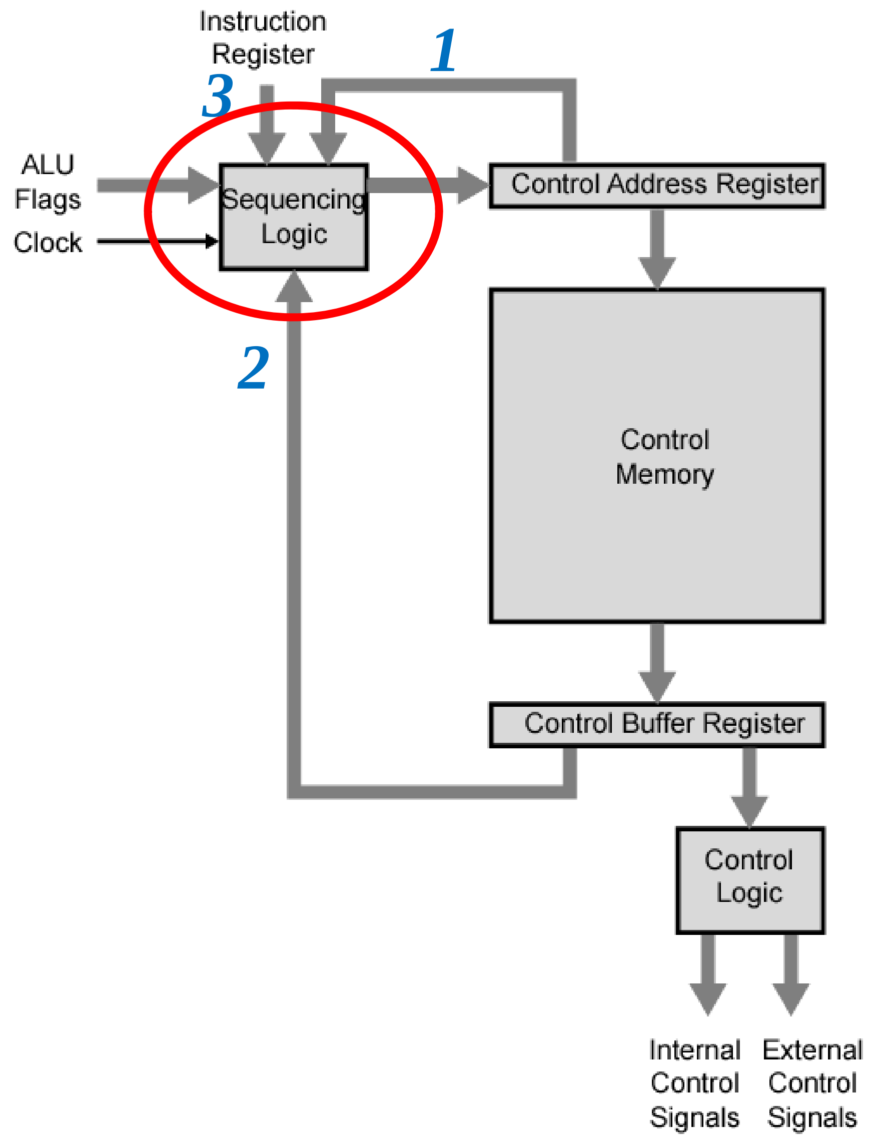
17.3 Microinstruction execution

- A taxonomy of microinstructions
- Microinstruction encoding
- LSI-11 microinstruction execution
- IBM 3033 Microinstruction execution

Execution

- The *microinstruction cycle* is the basic event on a microprogrammed processor
- Each cycle is made up of two events
  - ***Fetch***
    - Determined by generation of microinstruction address
  - ***Execute***
    - To generate control signals
    - Some control points *internal* to processor
    - Rest go to *external* control bus or other interface
    - And generate the *next* micro-instruction address

Control Unit Organization



A Taxonomy of Microinstructions

- Vertical/horizontal
- Packed/unpacked
- Hard/soft microprogramming
- Direct/indirect encoding

Improvements over Wilkes

- Wilkes had *each bit* directly produced a control signal or directly produced one bit of next address
- More complex address sequencing schemes, using *fewer microinstruction bits*, are possible
  - ➢ Require *more complex sequencing* logic module
  - ➢ Control word bits can be saved by *encoding* and subsequently *decoding* control information

How to Encode

- Assume K different internal and external control signals
- Wilkes's:
  - *K bits* dedicated
  - *2^K combinations for control signals* during instruction cycle
- ***Not all combinations are used***
  - Two sources cannot be gated to same destination
  - Register cannot be source and destination at same time
  - Only one pattern presented to ALU at a time
  - Only one pattern presented to external control bus at a time
- *Require all necessary $Q < 2^K$ which can be encoded with $\log_2 Q < K$ bits*
- Not done
  - As difficult to program as pure decoded (Wilkes) scheme
  - Requires complex slow control logic module
- ➢ Compromises
  - *More* bits than necessary used
  - Some combinations that are physically allowable are not possible to encode

Vertical/horizontal

- To the relative width of microinstructions
- Vertical
  - ✓ Width is narrow, 16-40bits
  - ✓ n control signals encoded into $\log_2 n$ bits
  - X Limited ability to express *parallelism*
  - X Considerable *encoding* of control information requires external memory word *decoder* to identify the exact control line being manipulated
- Horizontal
  - X Width 40-100bits
  - ✓ High degree of parallel operations possible
  - ✓ Little encoding of control information

For example, 8 bits control signals...

Packed/unpacked

- Packet
  - ✓ A given number of bits contains more information
  - — Encoding

Hard/soft programming

- The degree of closeness to the underlying control signals and hardware layout
- Hard: *fixed and committed* to read-only memory
- Soft: *changeable and suggestive* of user microprogramming

Specific *Encoding* Techniques

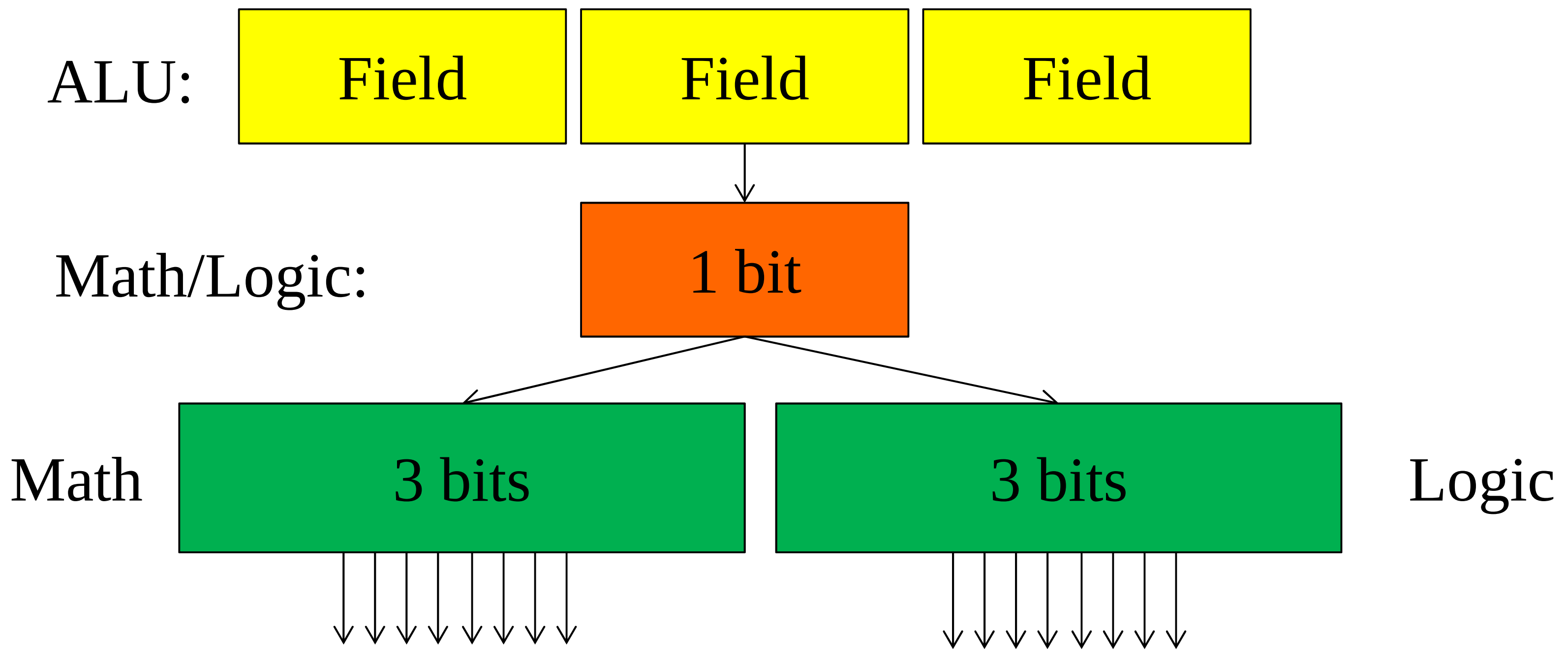
- Microinstruction organized as *set of fields*
- Each field contains a code
- A code activates one or more control signals
- Design of an encoded microinstruction format in simple terms
  - Organize format into *independent* fields
    - Each field depicts a set of actions (pattern of control signals)
    - Actions from different fields can occur *simultaneously*
  - Alternative actions that can be specified by a field are mutually exclusive
    - Only one action specified for field could occur at a time

Functional encoding/resource encoding

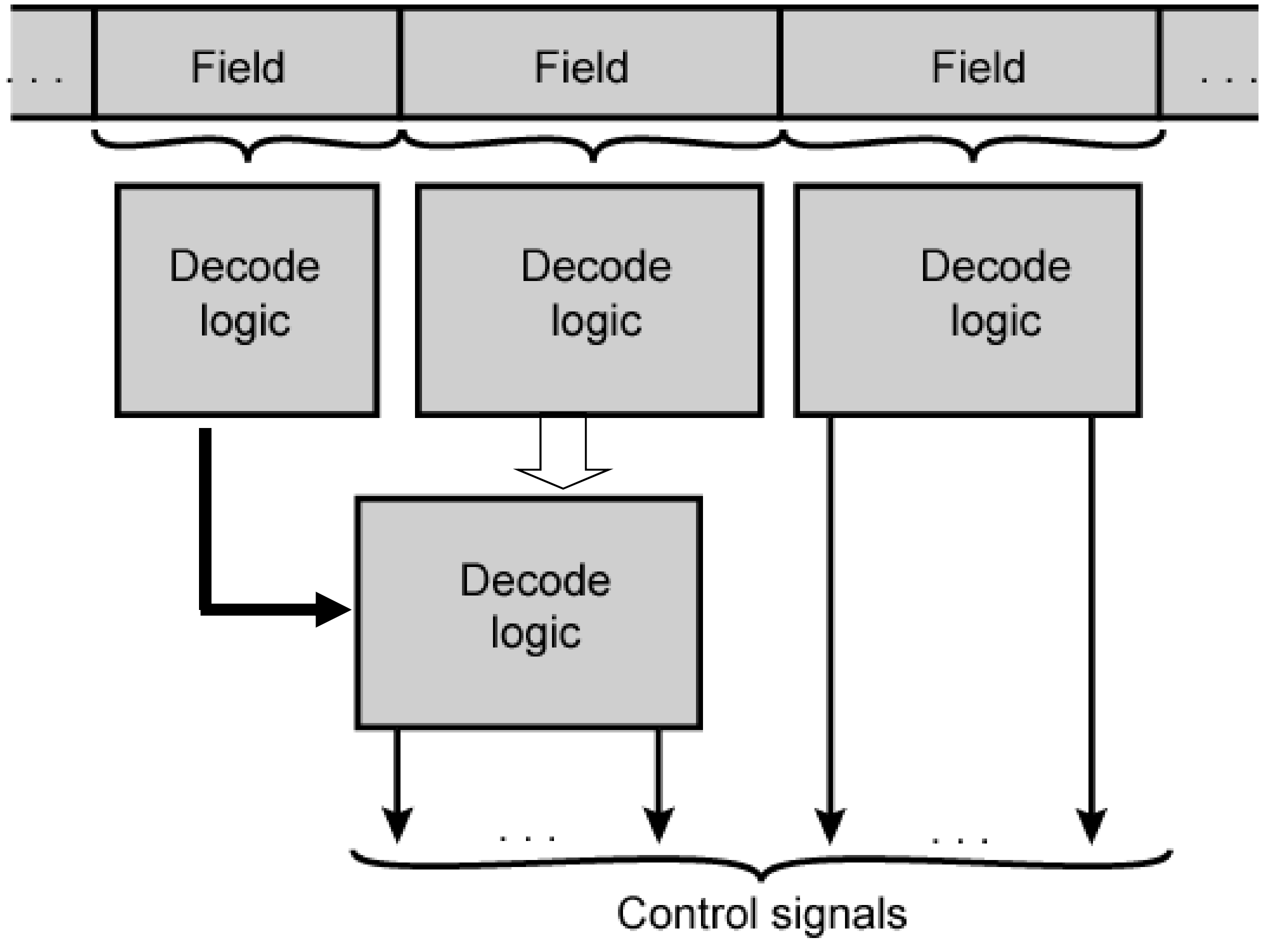
- Functional encoding
  - —One field for a function (*for example, “load to AC”*), and each code for a resource
- Resource encoding
  - —One field for a resource (*for example, “I/O, memory, ALU, etc”*)

Direct/indirect encoding

- Indirect
  - *One field* is used to determine the interpretation of *another field*
  - Two levels of decoding
  - slow

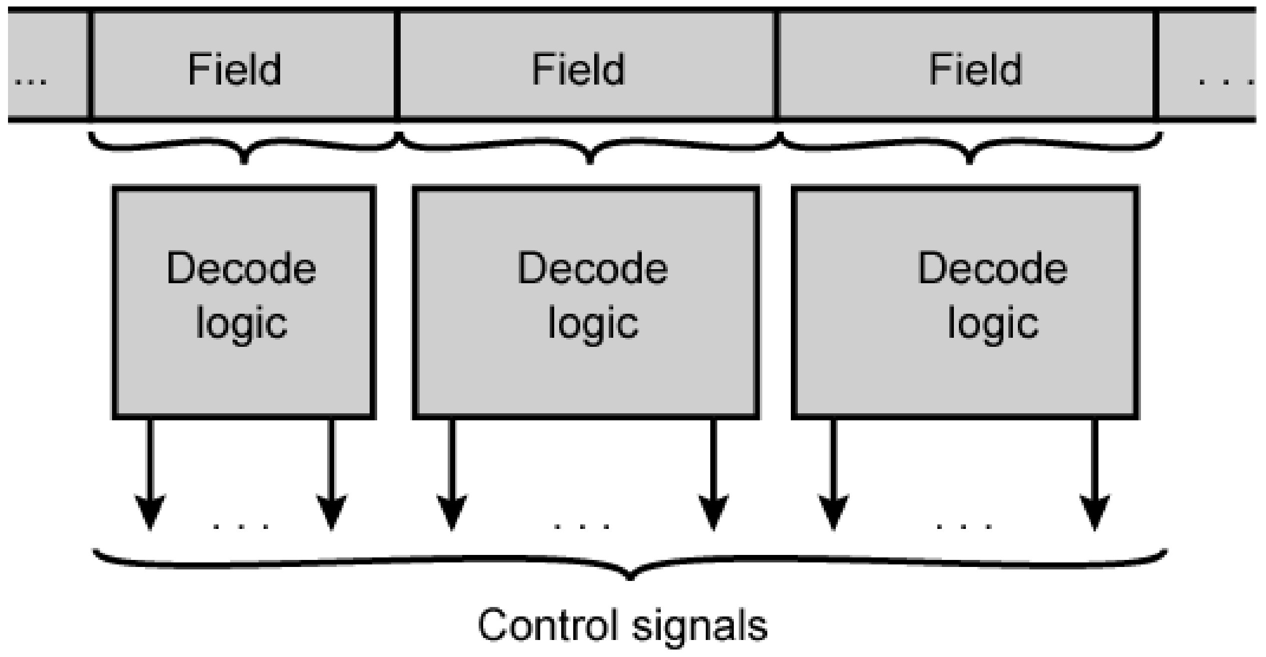


Microinstruction Encoding
Indirect Encoding



(b) Indirect encoding

Microinstruction Encoding
Direct Encoding



(a) Direct encoding

Home work

- Reading chapter 17
- Problems: 3, 4, 6, 7

- **Problems 17.3**

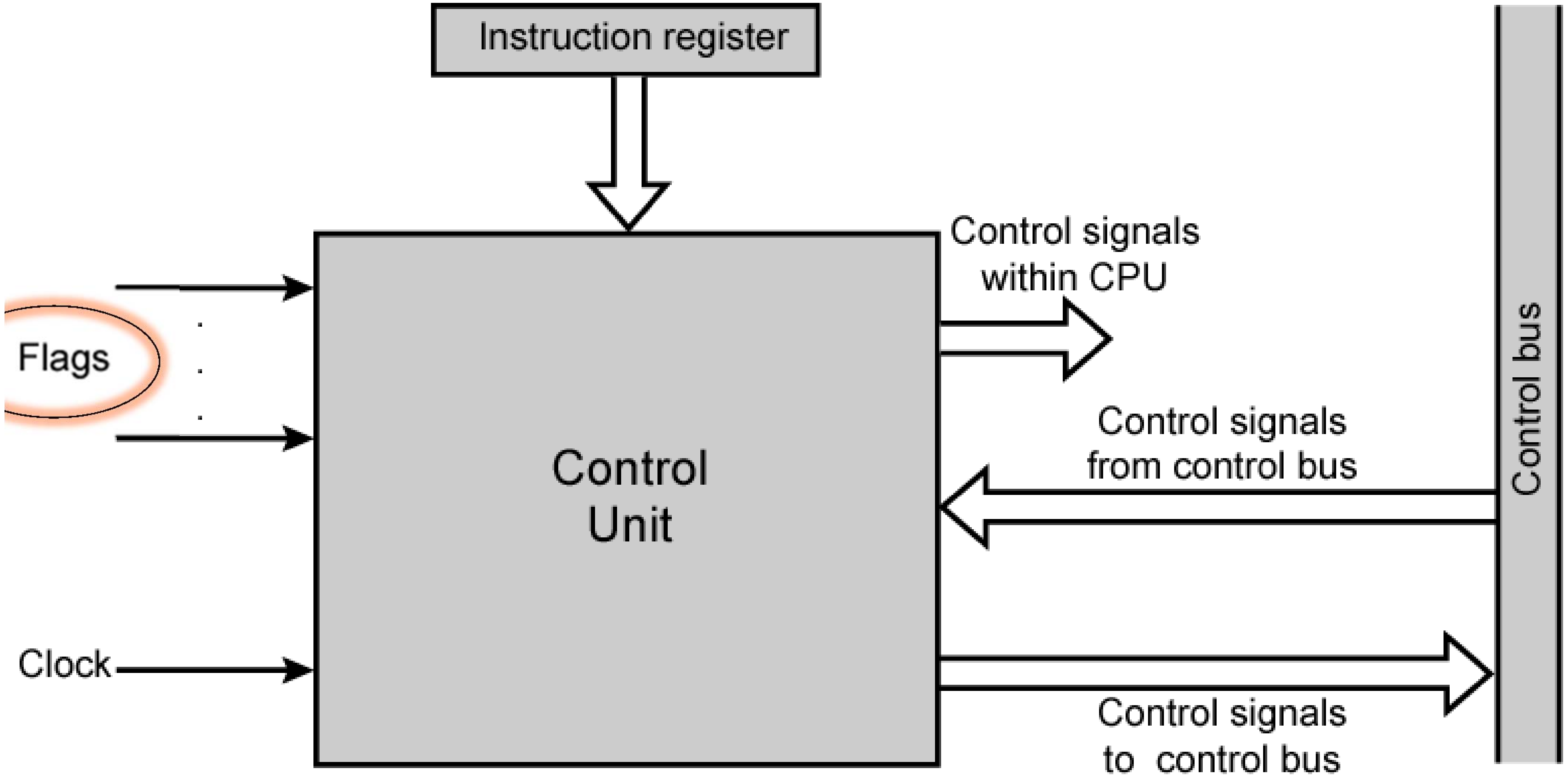
A simple processor has four major phases to its instruction cycle: fetch, indirect, execute, and interrupt. Two 1-bit flags designate the current phase in a hardwired implementation.

- A. why are these flags needed?
- B. why are they not needed in a micro-programmed control unit?

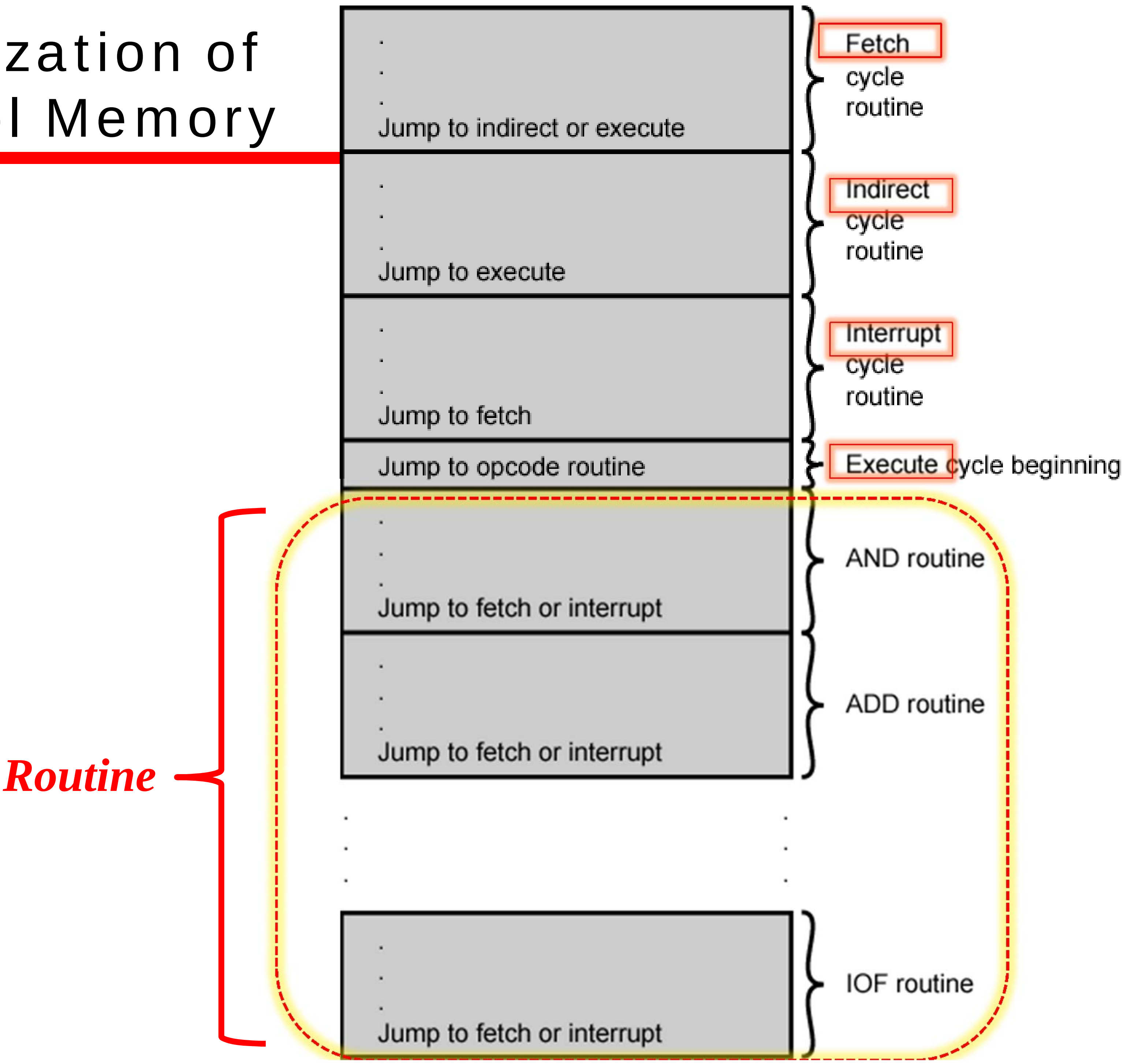
Control Signals - inputs

- Clock
  - One micro-instruction (or set of parallel micro-instructions) per clock cycle
- Instruction register
  - Op-code for current instruction
  - Determines which micro-instructions are performed
- Flags
  - ***State of CPU***
  - Results of previous operations
- From control bus
  - Interrupts
  - Acknowledgements

Model of Control Unit


Instruction register
Flags
Clock
Control Unit
Control signals within CPU
Control signals from control bus
Control signals to control bus
Control bus

Organization of Control Memory

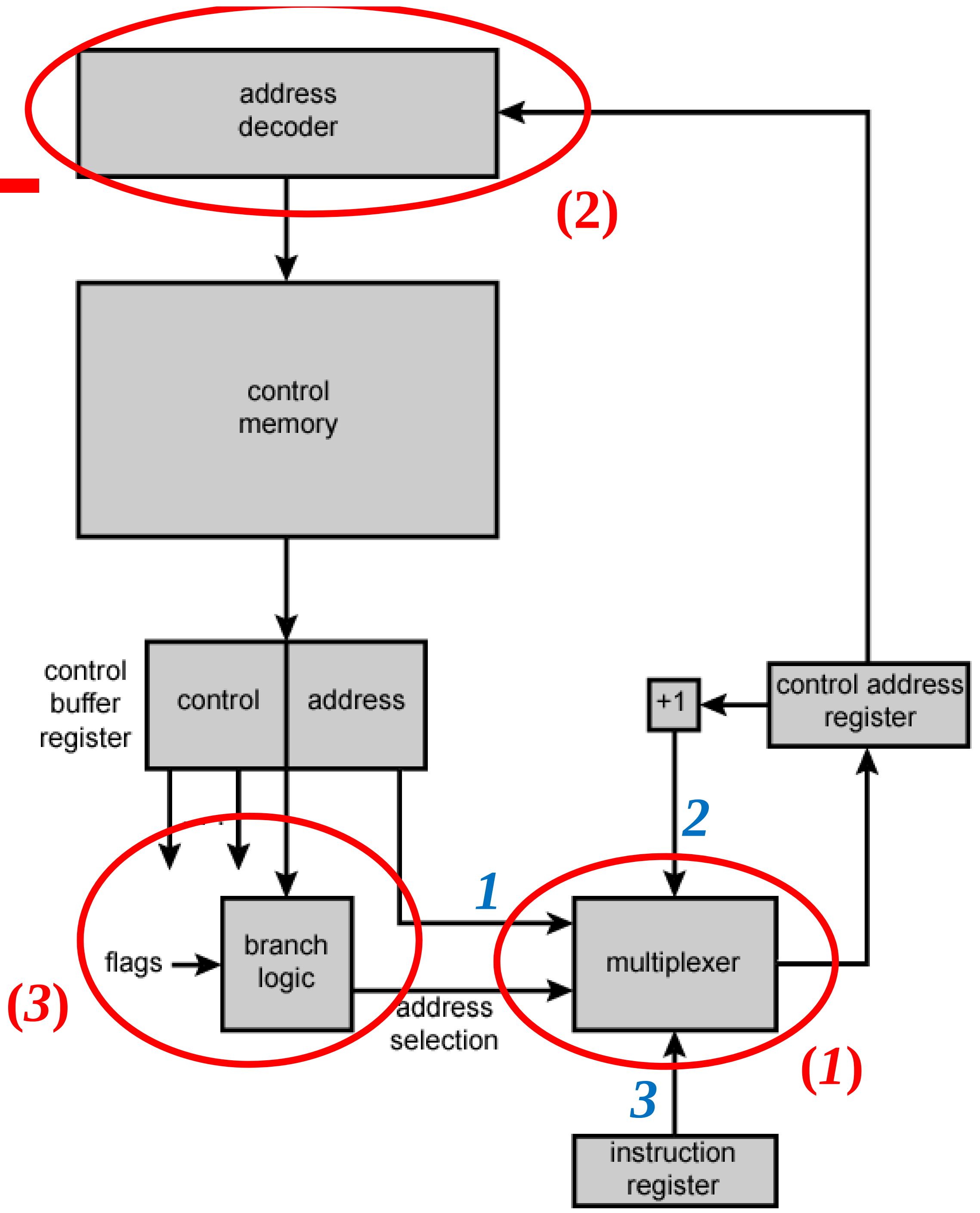


- **Problems 17.4**

Consider the control unit of Fig. 17.7. Assume that the control memory is 24 bits wide. The control portion of the microinstruction format is divided into two fields. A micro-operation field of 13 bits specifies the micro-operations to be performed. An address selection field specifies a condition, based on the flags, that will cause a microinstruction branch. There are eight flags.

- A. how many bits are in the address selection field?
- B. how many bits are in the address field?
- C. what is the size of the control memory?

Branch Control Logic: Single Address Field

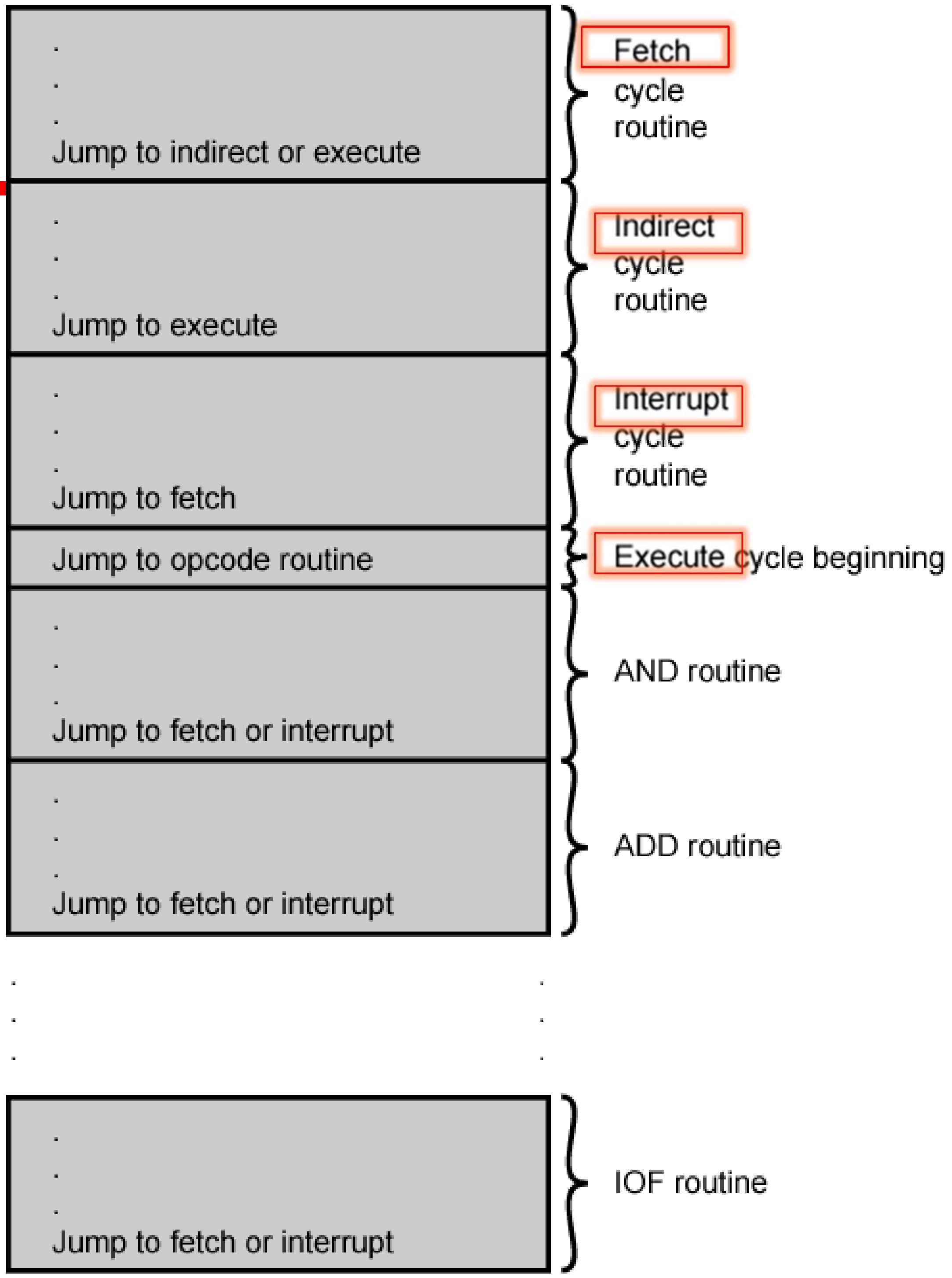


- **Problems 17.6**

We wish to provide *8 control words* for each machine instruction routine. Machine instruction opcodes have *5 bits*, and control memory has *1024 words*. Suggest a mapping from the instruction register to the control address register.

Organization of Control Memory

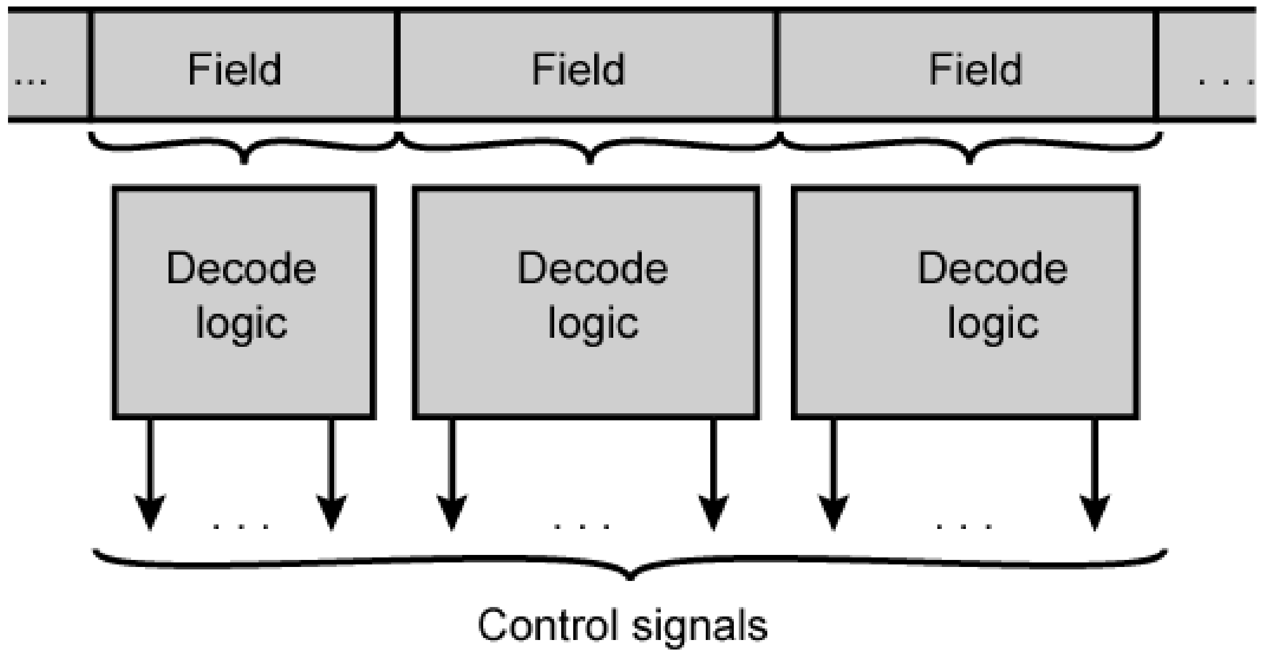
Routine <=> Opcode



- **Problems 17.7**

An encoded microinstruction format is to be used. Show how a 9-bit micro-operation field can be divided into subfields to specify 46 different actions.

Microinstruction Encoding
Direct Encoding



(a) Direct encoding