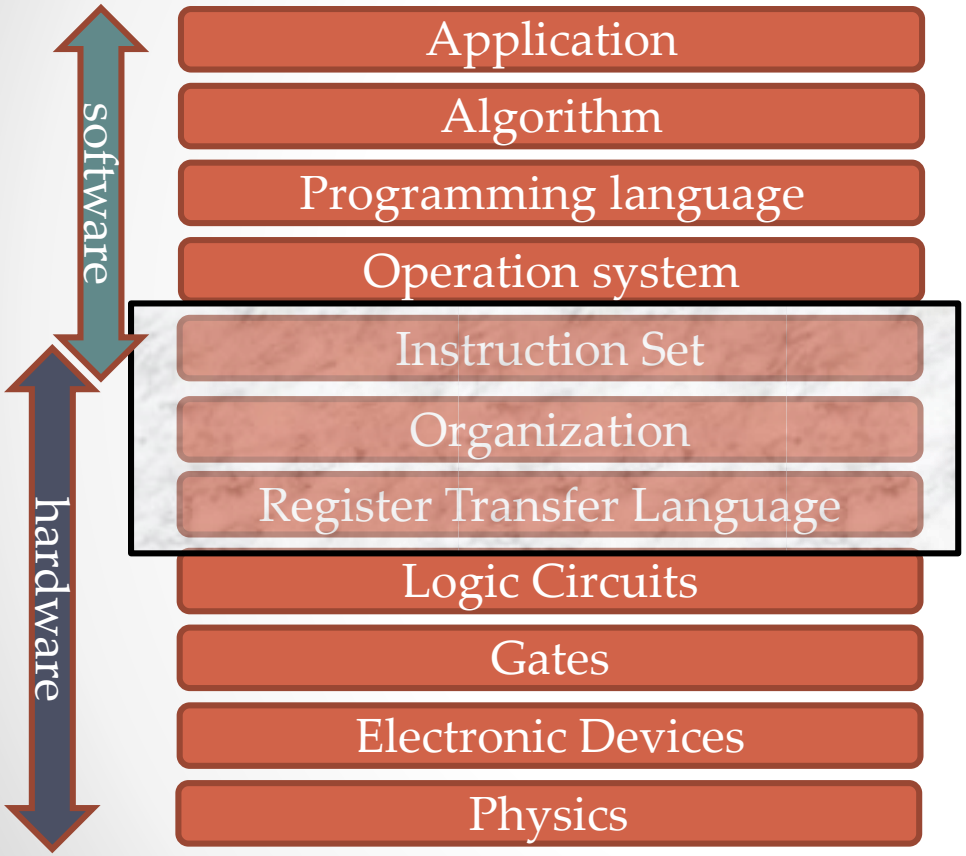
Computer Architecture

Dr. Esti Stein
(Partly taken from Dr. Alon Schclar slides)

Based on slides by:
Prof. Myung-Eui Lee
Korea University of Technology & Education
Department of Information & Communication

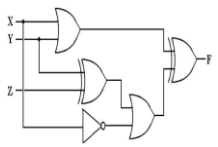
Taken from: **M. Mano/Computer Design and Architecture 3rd Ed.**

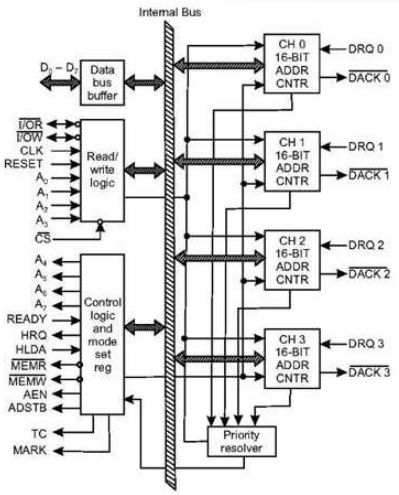
Logic Circuits

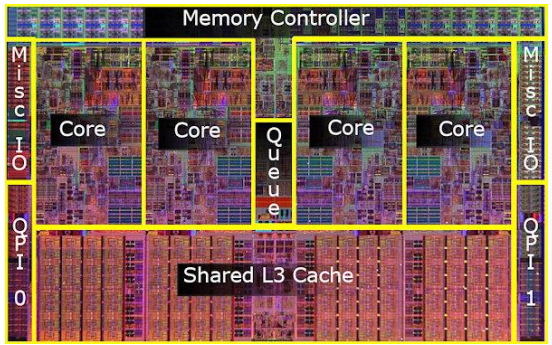


Logic Circuits

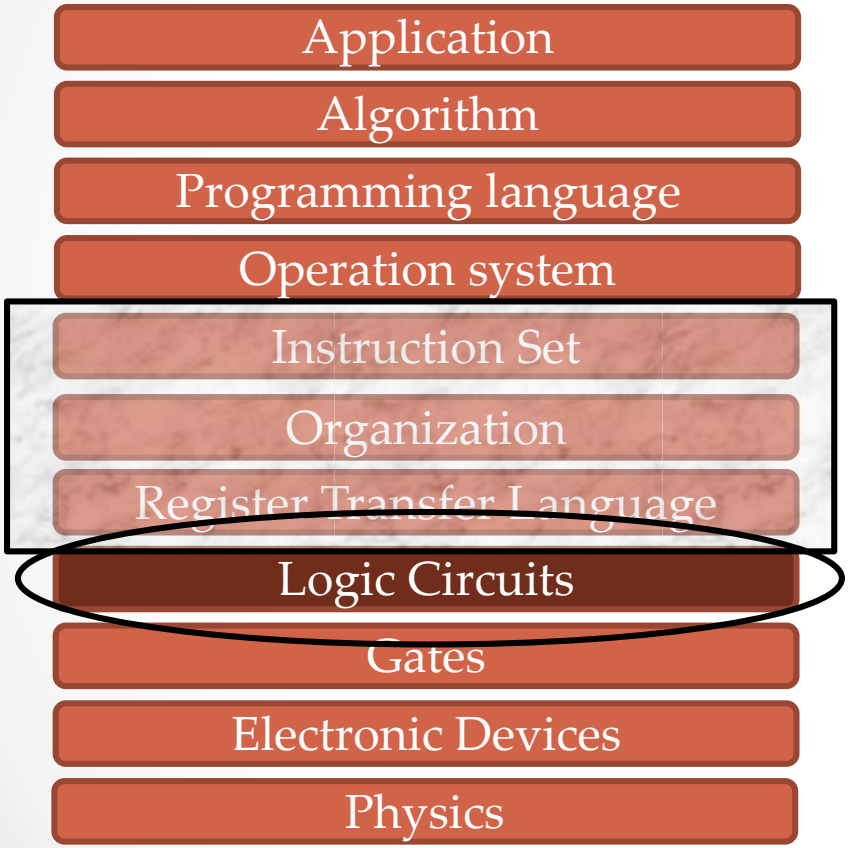
- Application
- Algorithm
- Programming language
- Operation system
- Instruction Set
- Organization
- Register Transfer Language
- **Logic Circuits**
- Gates
- Electronic Devices
- Physics







Logic Circuits (LCs)



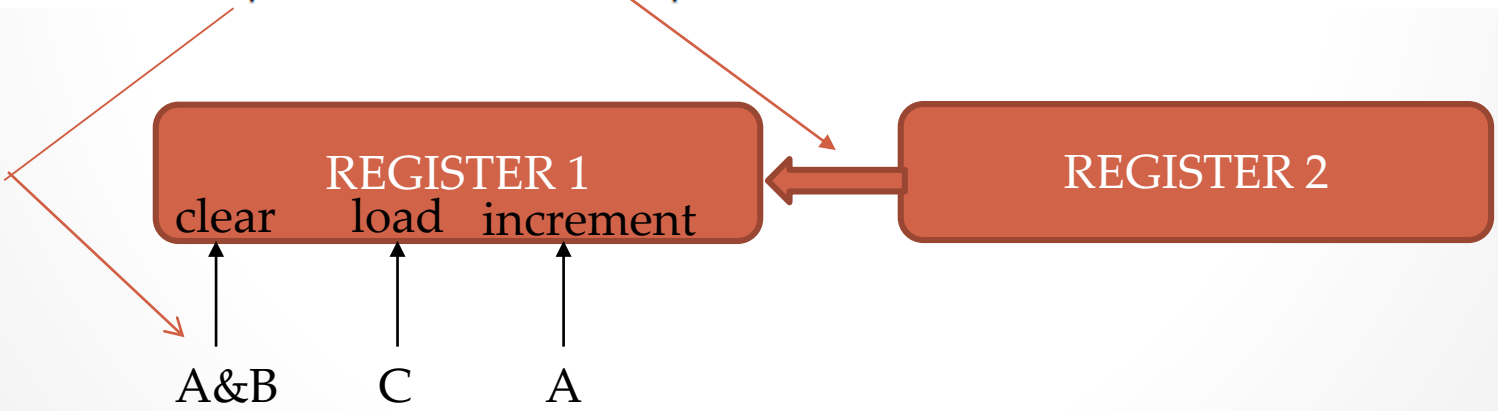
Best defined by:

- their registers
- the data operations on the registers

Microoperations

How to describe LCs

- 4-1 Register Transfer Language
  - Microoperation
    - The operations executed on data stored in registers *(shift, clear, load, count)*
  - Internal H/W Organization *(best defined by specifying)*
    1. The set of registers
    2. The sequence of microoperations
    3. The sequence control of microoperations



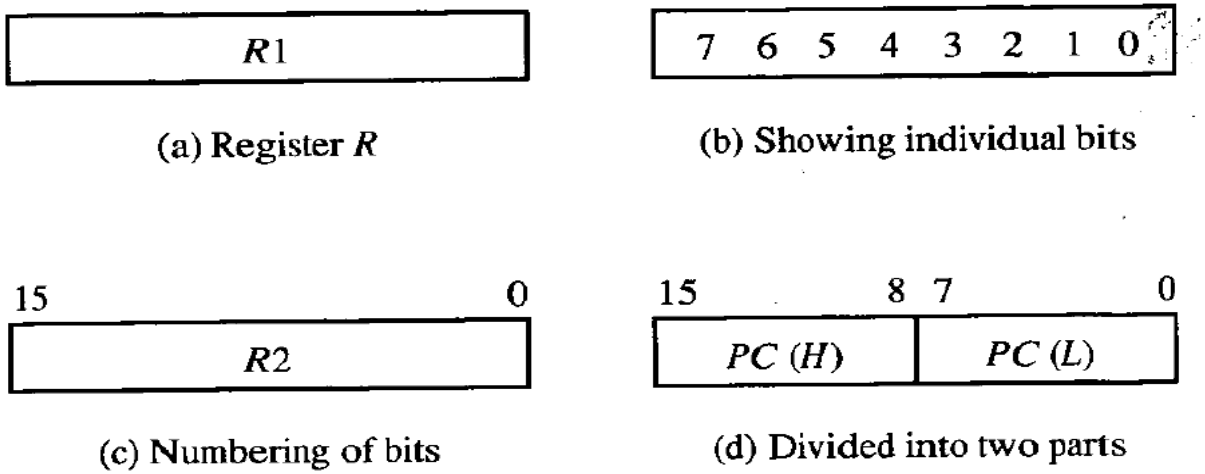
The RTL Language

- Register Transfer Language
  - The symbolic notation used to describe the microoperation transfer among registers
    - » The use of ***symbols*** instead of a ***narrative explanation*** provides an organized and concise manner
  - A convenient tool for describing the internal organization of digital computers in concise and precise manner

Registers

- Registers : ***Fig. 4-1***
  - Designated by Capital Letter(*sometimes followed by numerals*) : MAR(Memory Address Register), PC(Program Counter), IR(Instruction Register), R1(Processor Register)

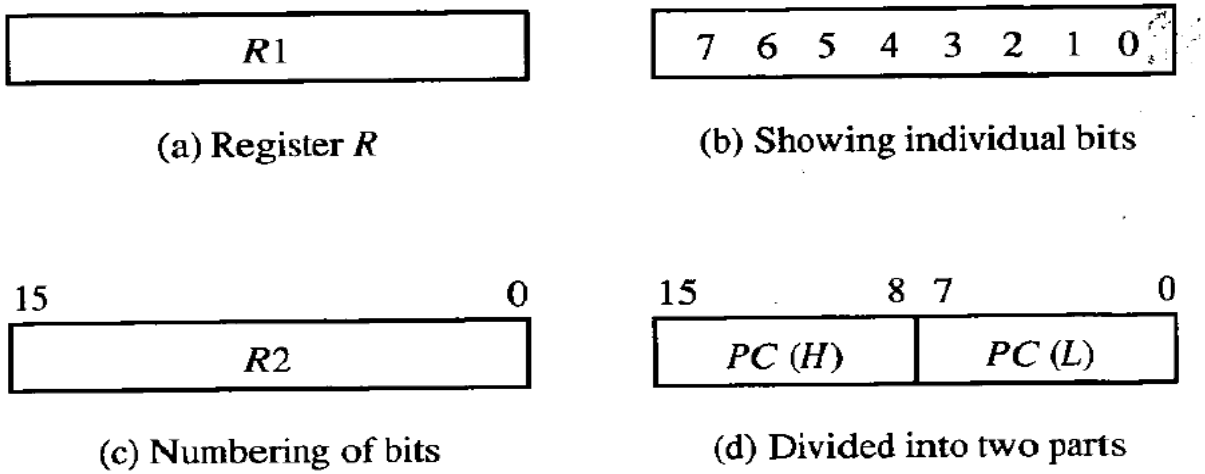
Figure 4-1 Block diagram of register.



Registers

- The individual F/Fs in an n-bit register : numbered in sequence from 0(*rightmost position*) through n-1
- The numbering of bits in a 16-bit register : marked on top of the box
- A 16-bit register partitioned into two parts : bit 0-7(*symbol "L" Low byte*), bit 8-15(*symbol "H" High byte*)

Figure 4-1 Block diagram of register.



RTL Explanation

- Register Transfer : *Information transfer from one register to another*
  - $R2 \leftarrow R1$ (***transfer of the content of register R1 into register R2***)
    - » The content of the source register R1 does not change after the transfer

- Control Function : *The transfer occurs only under a predetermined control condition*
  - The transfer operation is executed by the hardware only if P=1 : ***Fig. 4-2***

$$\left.\begin{array}{l} if\,(P=1)\ then\ (R2 \leftarrow R1) \\ P: R2 \leftarrow R1 \end{array}\right\} =$$

  - A comma is used to separate two or more operations(*Executed at the same time*)

$$T: R2 \leftarrow R1,\ R1 \leftarrow R2$$

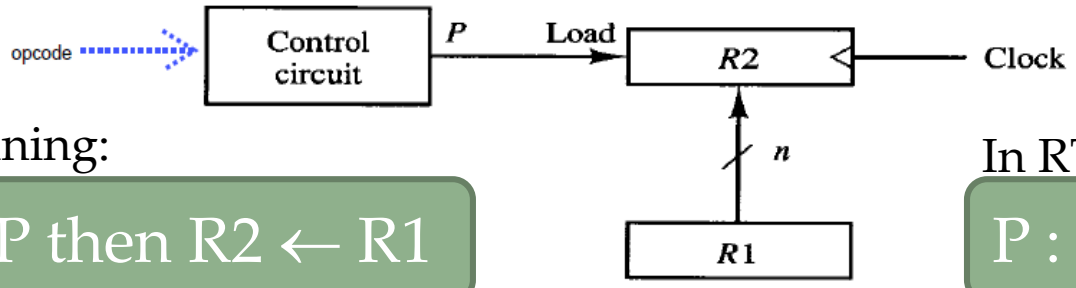
RTL Definition

- Basic Symbols for Register Transfer : Tab. 4-1

Symbol	Description	Examples
Letters(and numerals)	Denotes a register	MAR, R2
Parentheses ( )	Denotes a part of a register	R2(0-7), R2(L)
Arrow <—	Denotes transfer of information	R2 <— R1
Comma ,	Separates two microoperations	R2 <-- R1, R1 <— R2

Example:

Figure 4-2 Transfer from $R1$ to $R2$ when $P = 1$.



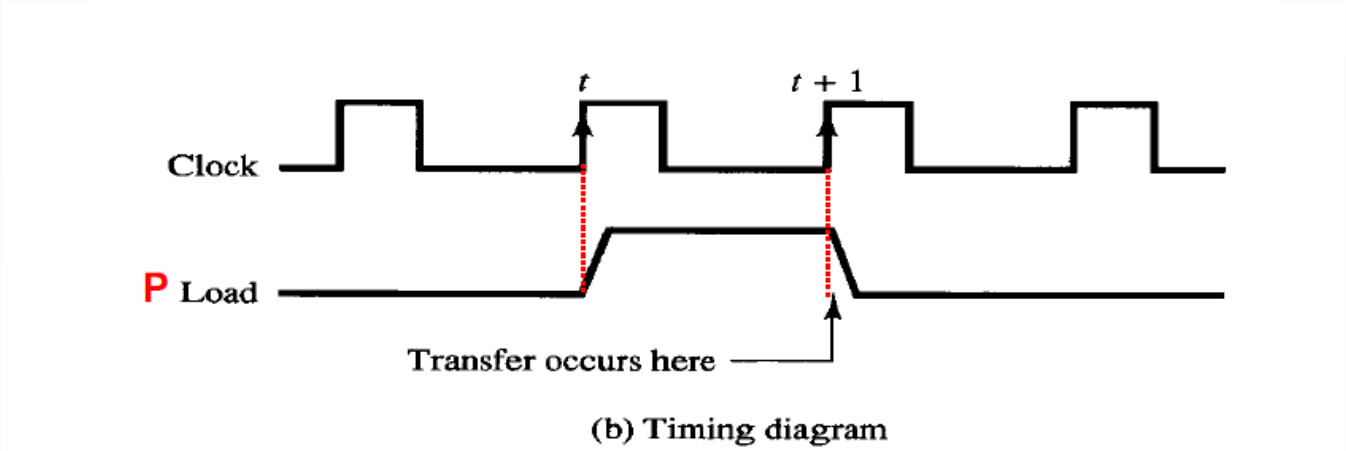
Meaning:

If P then R2 ← R1

In RTL:

P : R2 ← R1

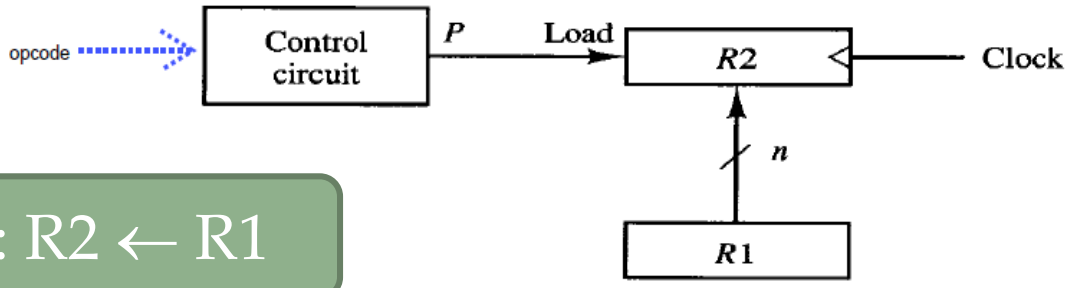
RTL meaning



(b) Timing diagram

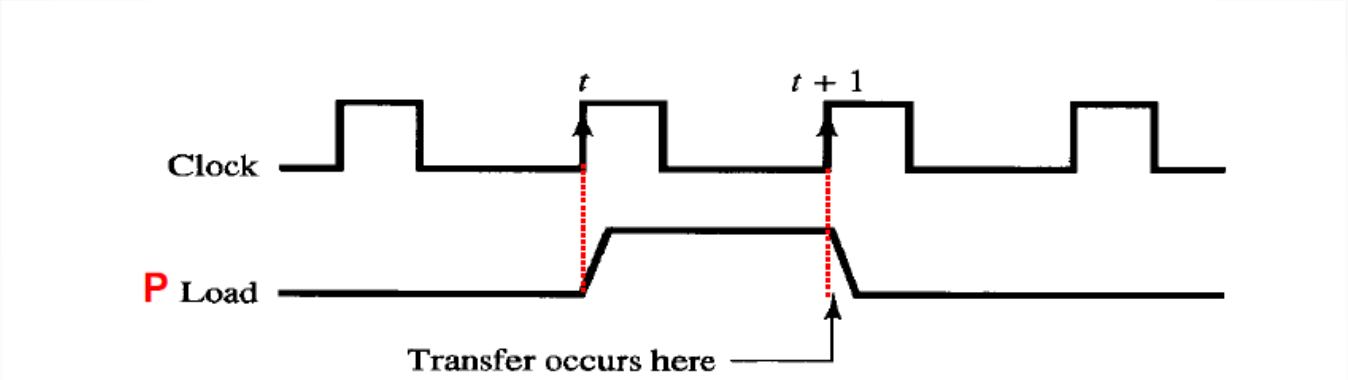
Example:

Figure 4-2 Transfer from $R1$ to $R2$ when $P = 1$.



P : R2 ← R1

RTL timing



- The clock is not included as a variable in the register transfer statements
  - It is assumed that all transfers occur during a clock edge transition.
- The control condition such as ***P*** becomes active just after time ***t***
- The actual transfer occurs when the register is triggered by the next positive transition of the clock at time ***t + 1***.

Reminder

An n-bit counter with parallel load

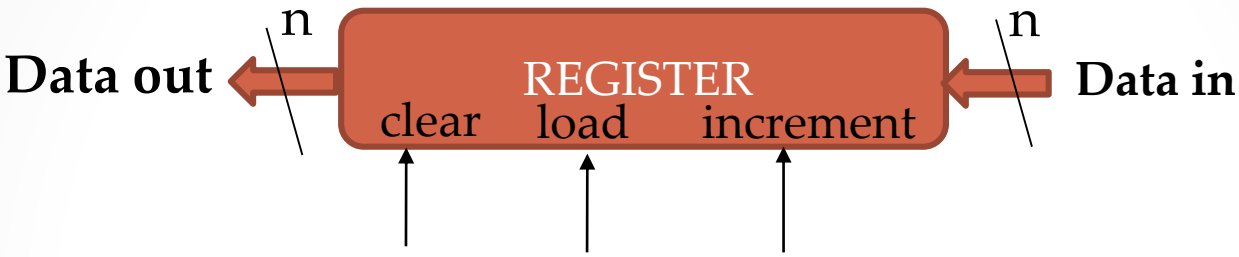


TABLE 2-5 Function Table for the Register of Fig. 2-11

Clock	Clear	Load	Increment	Operation
↑	0	0	0	No change
↑	0	0	1	Increment count by 1
↑	0	1	×	Load inputs I_0 through I_3
↑	1	×	×	Clear outputs to 0

Another Reminder

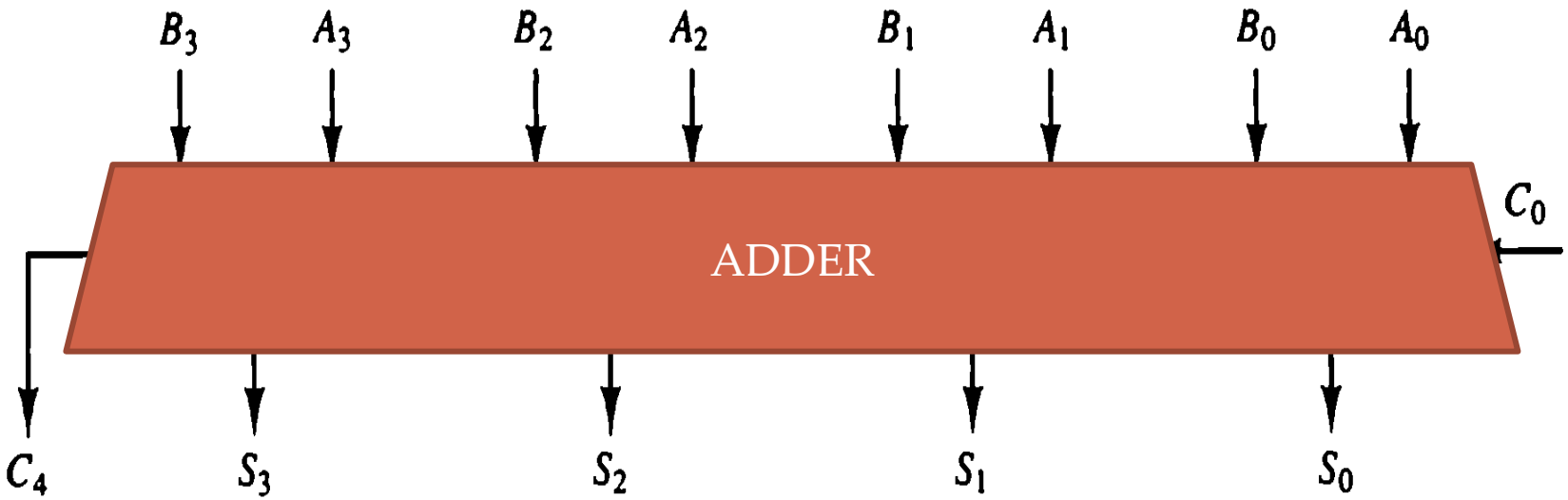


Figure 4-6 4-bit binary adder.

Another Reminder

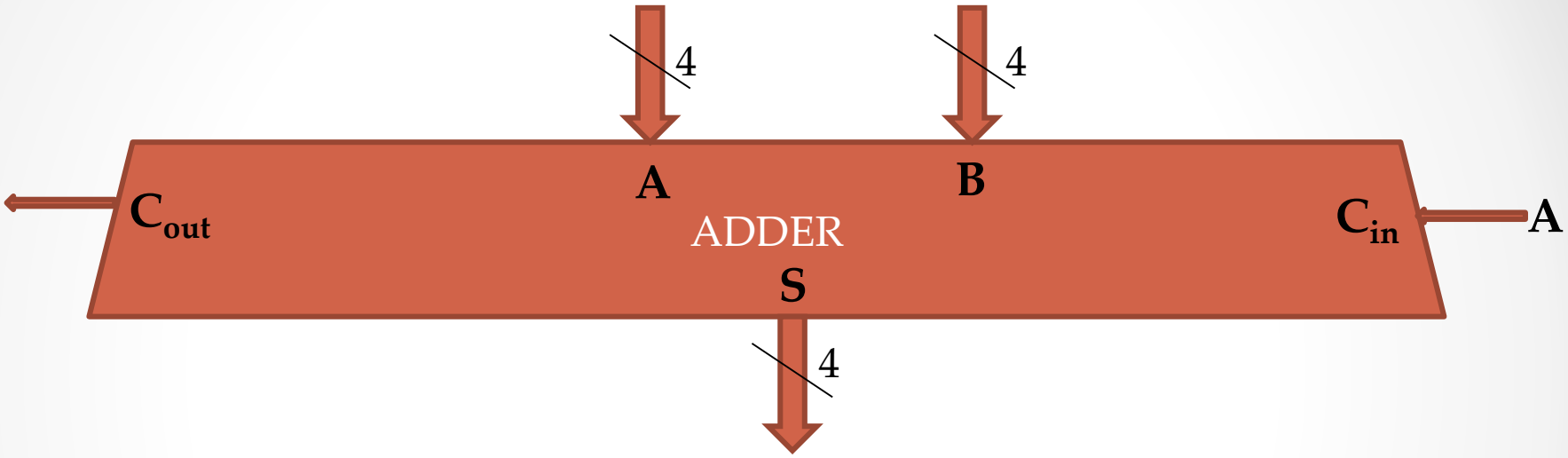
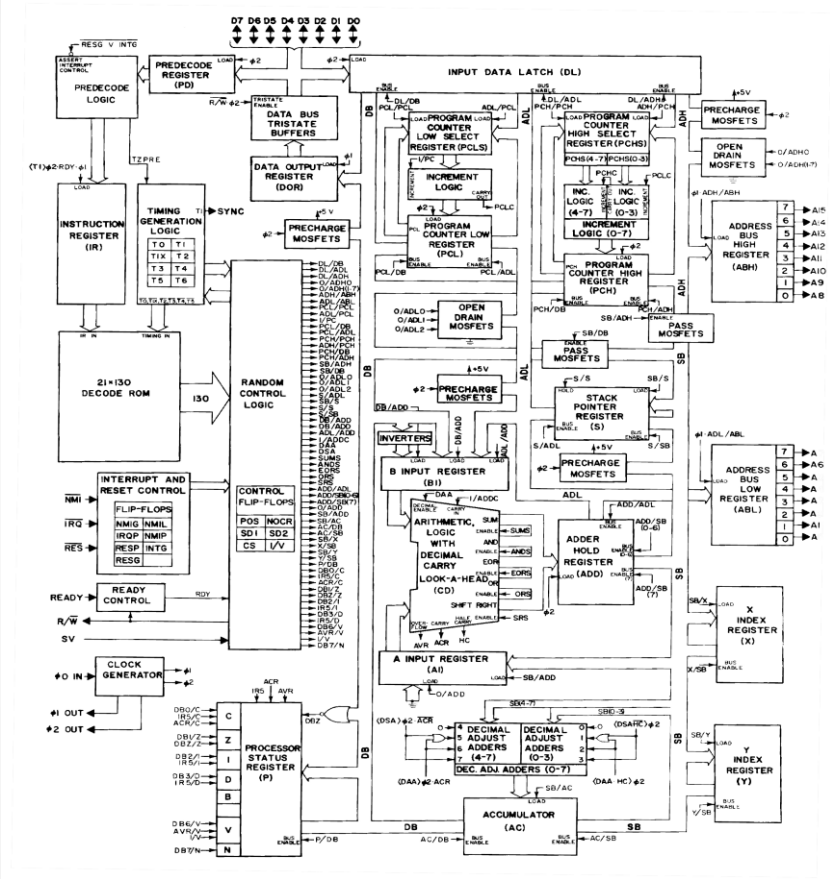


Figure 4-6 4-bit binary adder.

The BUS



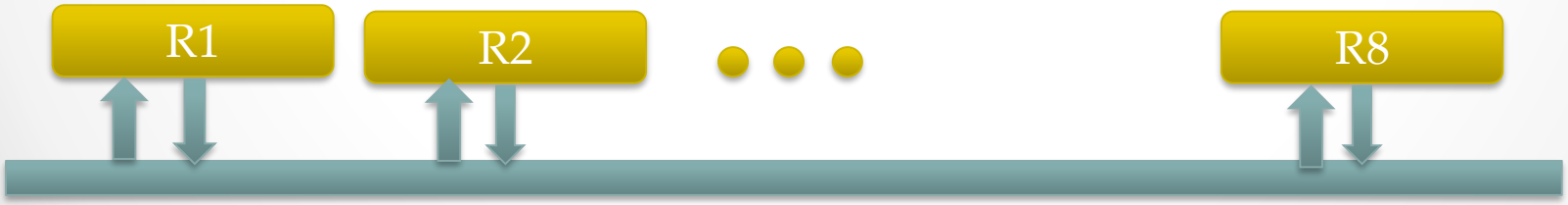
The 8-bit Z-80 processor was very popular in the late 1970s and early 1980s

A structure consists of set of common lines (one for each bit of a register) through which data is transferred among them, **from one source** register at a time.

The BUS

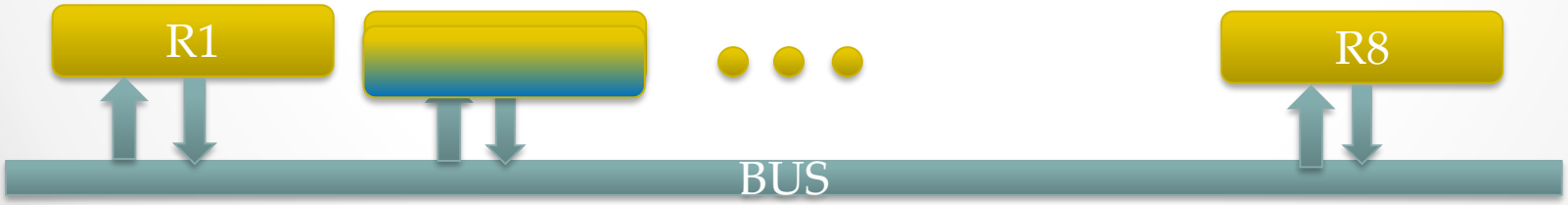
Control signals determine which source register is selected during each particular register transfer

A structure consists of set of common lines (one for each bit of a register) through which data is transferred among them, **from one source** register at a time.



The BUS

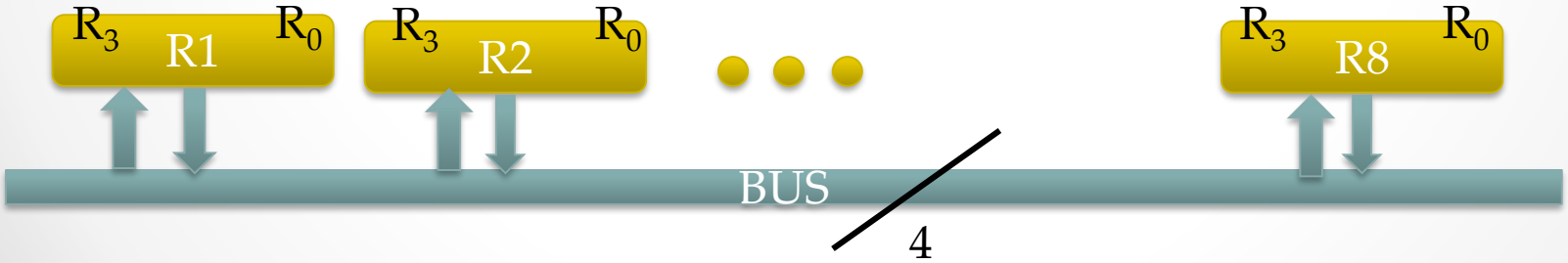
Control signals determine which source register is selected during each particular register transfer



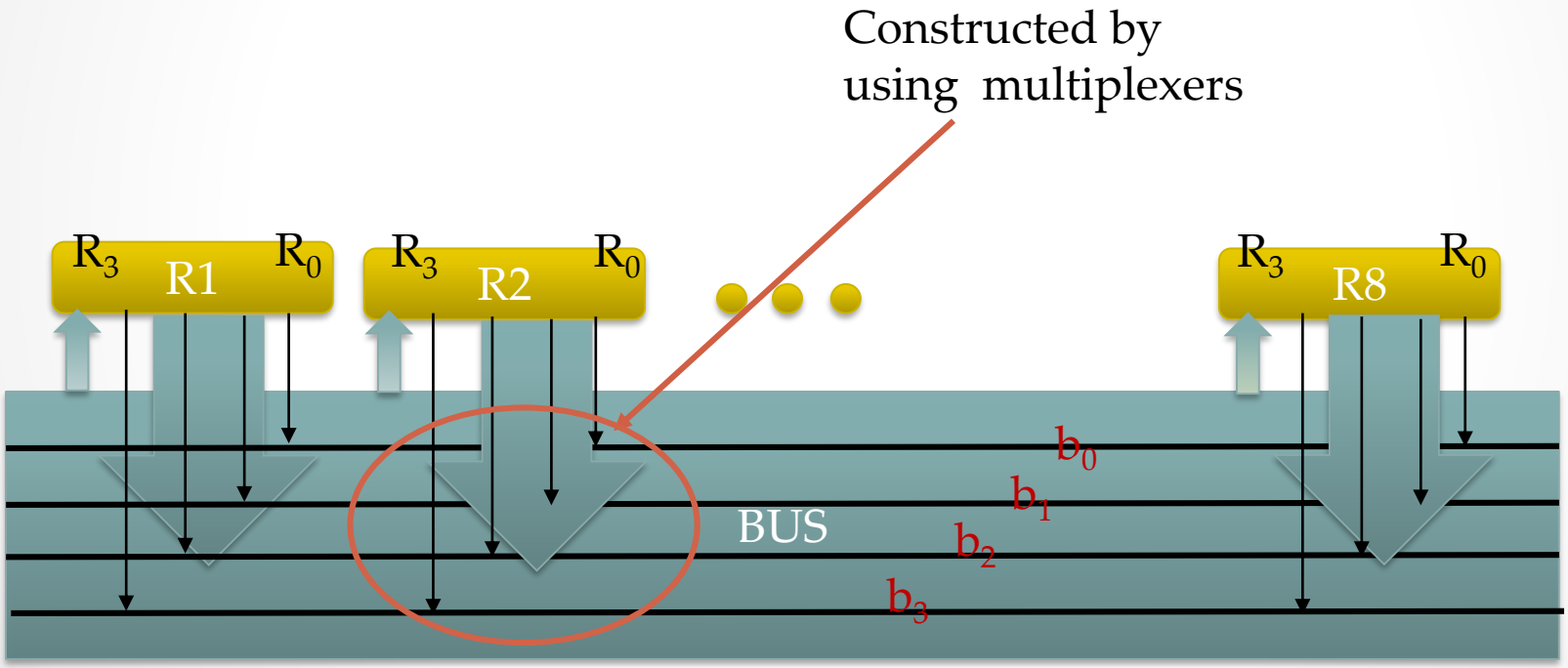
$R8 \leftarrow R2$

The BUS

Control signals determine which source register is selected during each particular register transfer

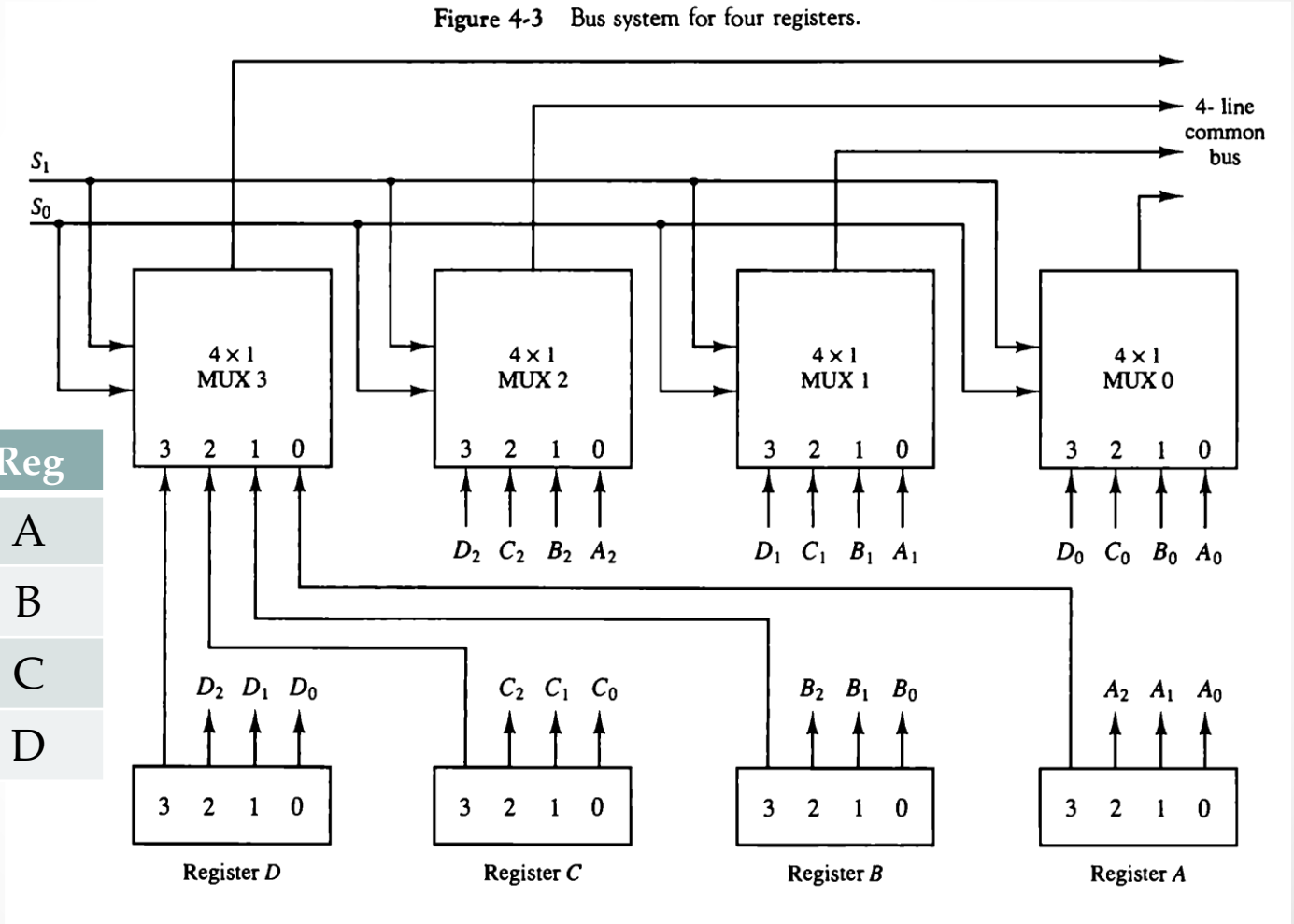


The BUS


Constructed by
using multiplexers
R3
R1
R0
R3
R2
R0
R3
R8
R0
BUS
b0
b1
b2
b3

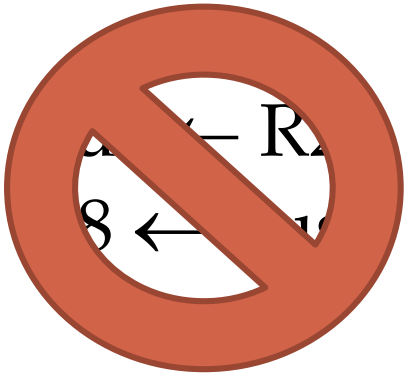
The BUS - MUX

S1	S0	Reg
0	0	A
0	1	B
1	0	C
1	1	D

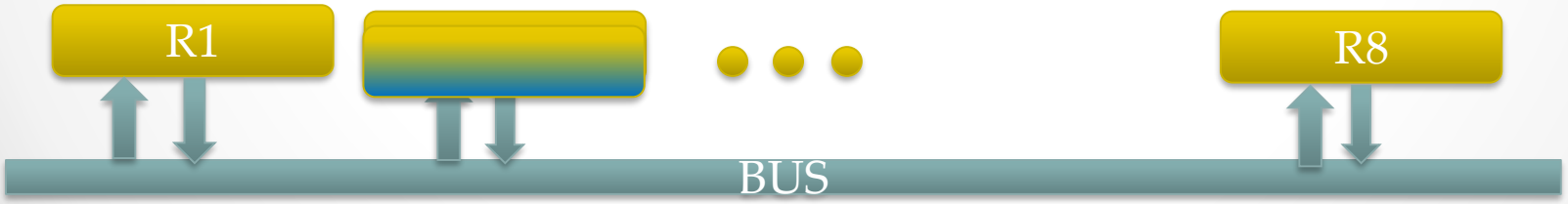


The BUS Transfer in RTL

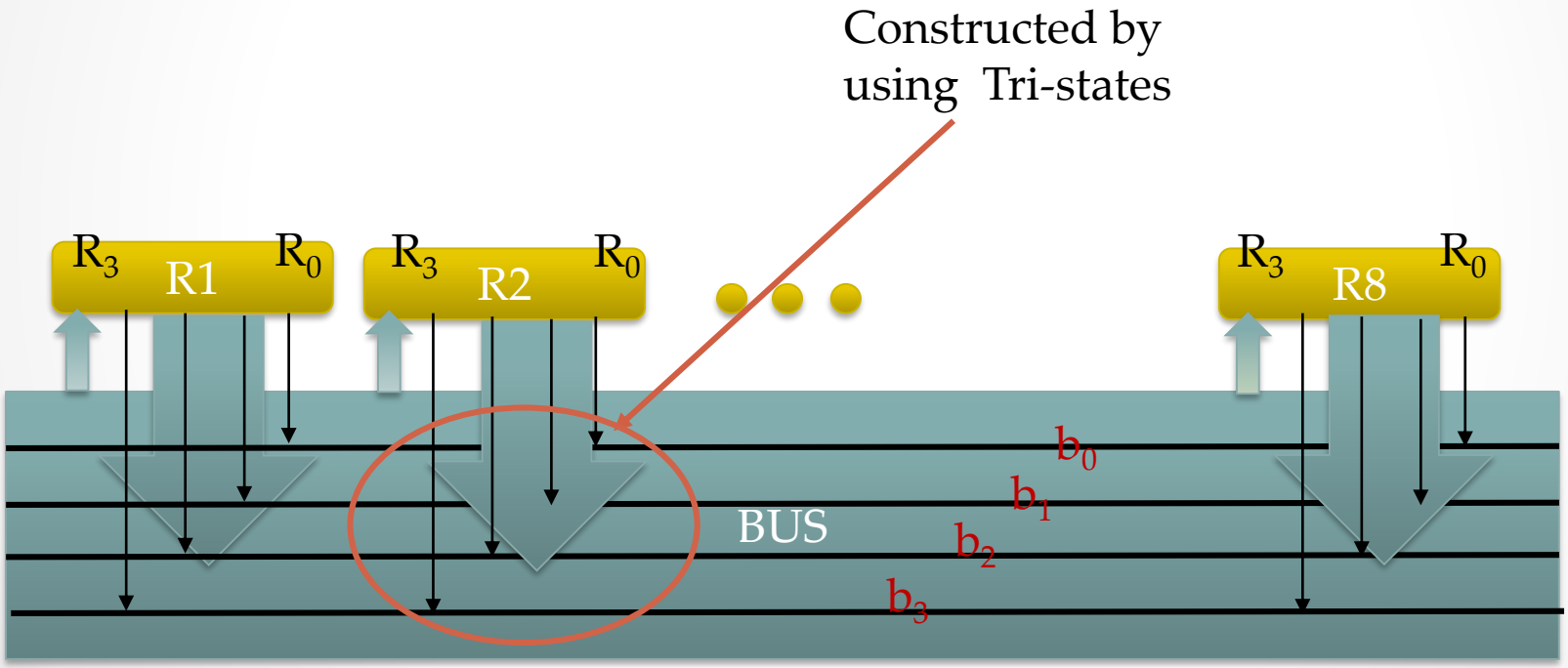
- The content of register C is placed on the bus, and the content of the bus is loaded into register R1 by activating its load control input



Simply R8 ← R2



The BUS


Constructed by
using Tri-states
R3
R1
R0
R3
R2
R0
R3
R8
R0
BUS
b0
b1
b2
b3

The BUS – Tristate

- Tri-state buffer gate : Fig. 4-4
  - When control input =1 : The output is enabled(output Y = input A) ***2 logic states***
  - When control input =0 : The output is disabled(output Y = high-impedance)

If C=1, Output Y = A

If C=0, Output = High-impedance

A	C	Output
0	0	high imp.
0	1	0
1	0	high imp.
1	1	1

The BUS - MUX

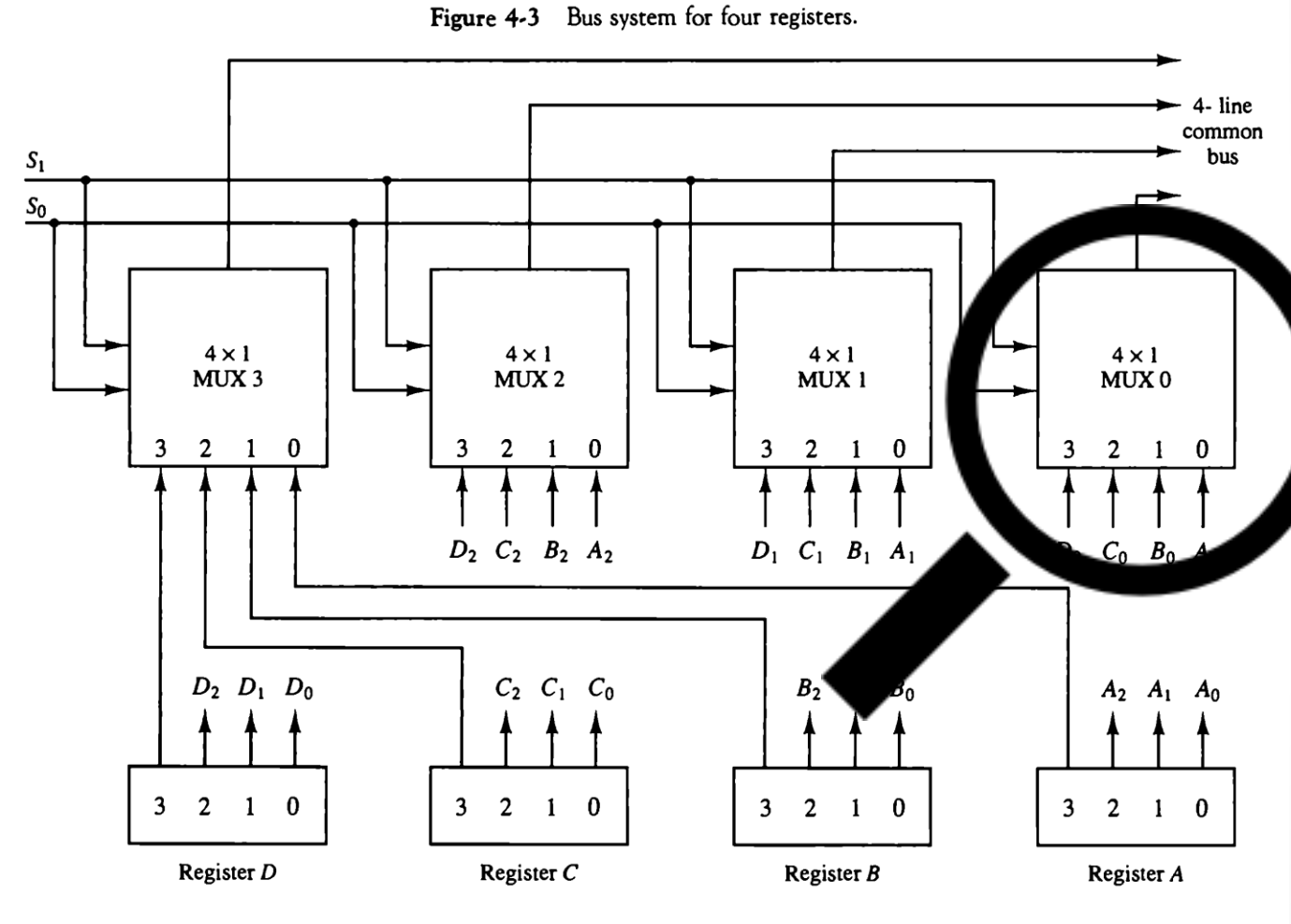
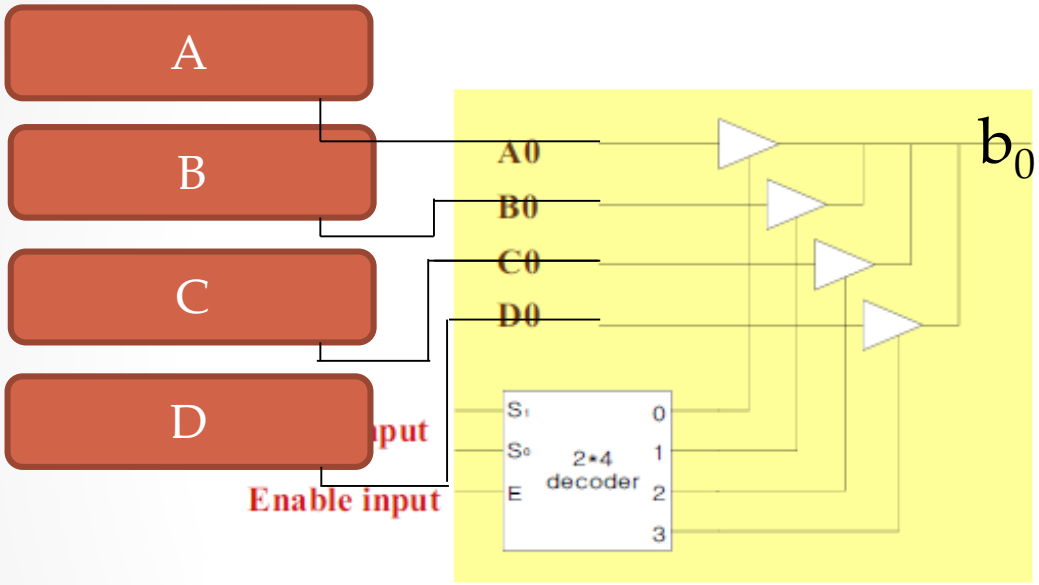


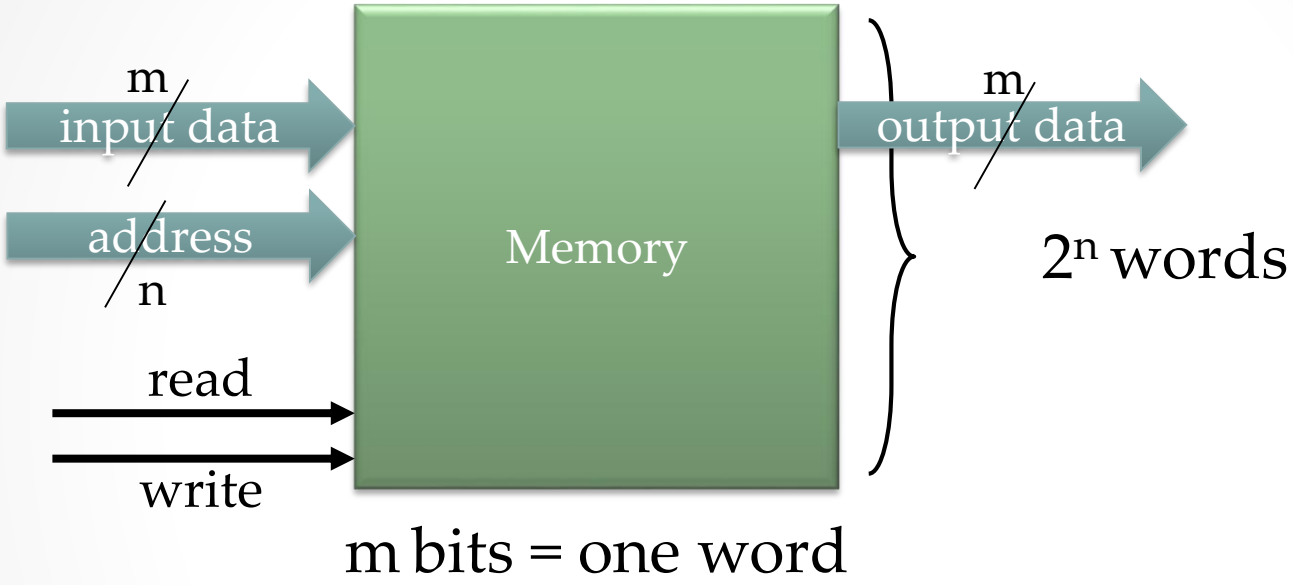
Figure 4-3 Bus system for four registers.
4- line common bus
S_1
S_0
4 × 1 MUX 3
4 × 1 MUX 2
4 × 1 MUX 1
4 × 1 MUX 0
3 2 1 0
D_2 C_2 B_2 A_2
D_1 C_1 B_1 A_1
C_0 B_0
D_2 D_1 D_0
C_2 C_1 C_0
B_2
A_2 A_1 A_0
Register D
Register C
Register B
Register A

Construct a MUX with a Tristate

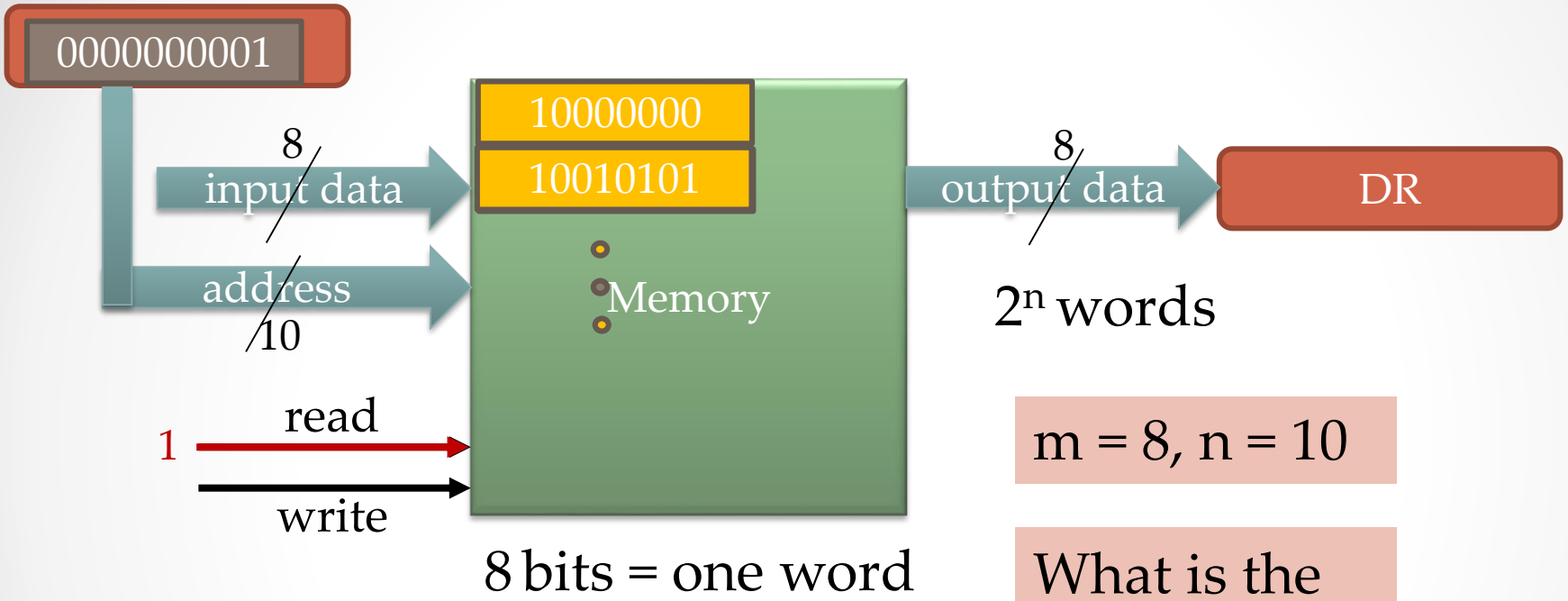


A bus line for one bit

The MEMORY


m
input data
address
n
read
write
Memory
m
output data
2n words
m bits = one word

The MEMORY - read

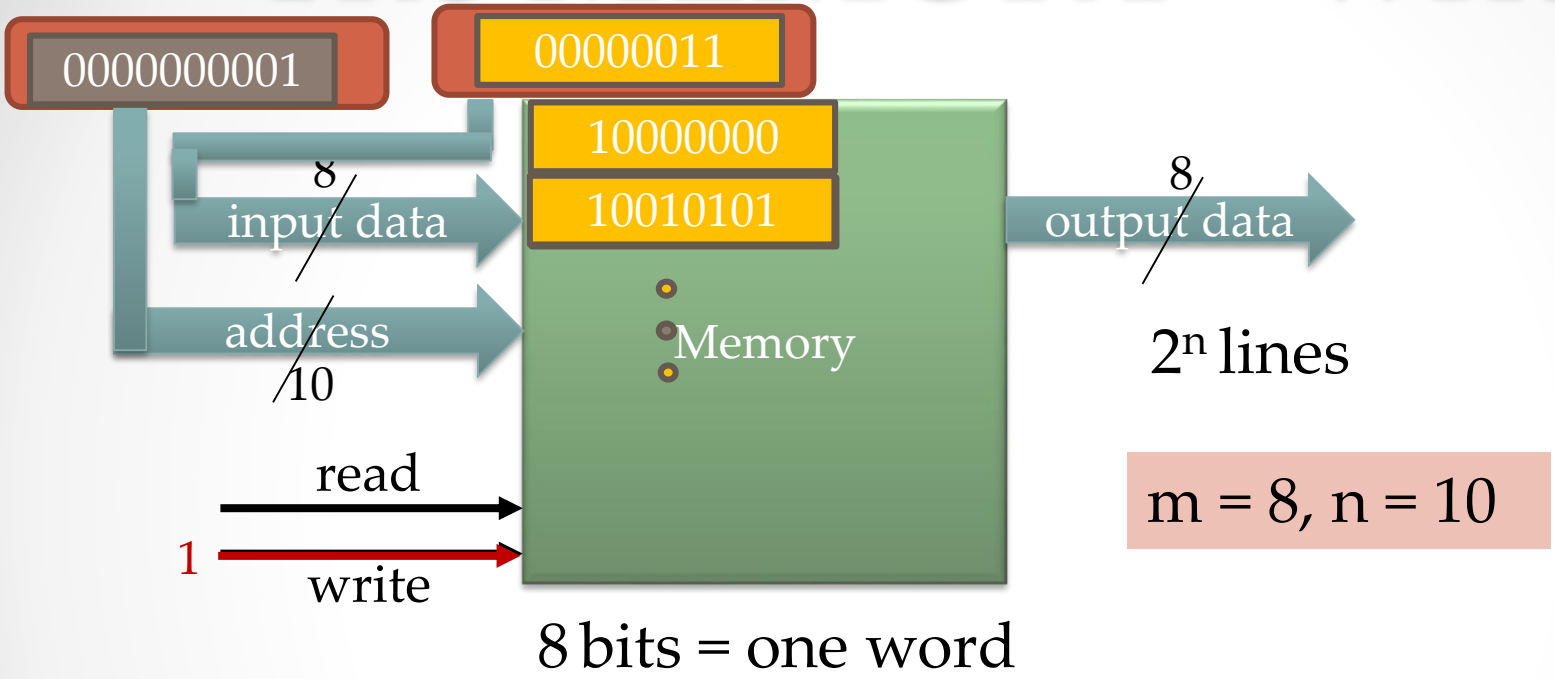


m = 8, n = 10

What is the size of the memory in bits? $2^{10} \times 8$

Read in RTL is DR ← M[AR]

The MEMORY - write



m = 8, n = 10

write in RTL is M[AR] ← R1