CSC 220: Computer Organization

Unit 11
Datapath Design

Prepared by:
Md Saiful Islam, PhD

Updated by:
Isra Al-Turaiki, PhD

Department of Computer Science

College of Computer and Information Sciences

Overview

- Register file
- Datapath Representation
- Accessing RAM
- The Control Word

Chapter-8

M. Morris Mano, Charles R. Kime and Tom Martin, **Logic and Computer Design Fundamentals**, Global (5th) Edition, Pearson Education Limited, 2016. ISBN: 9781292096124

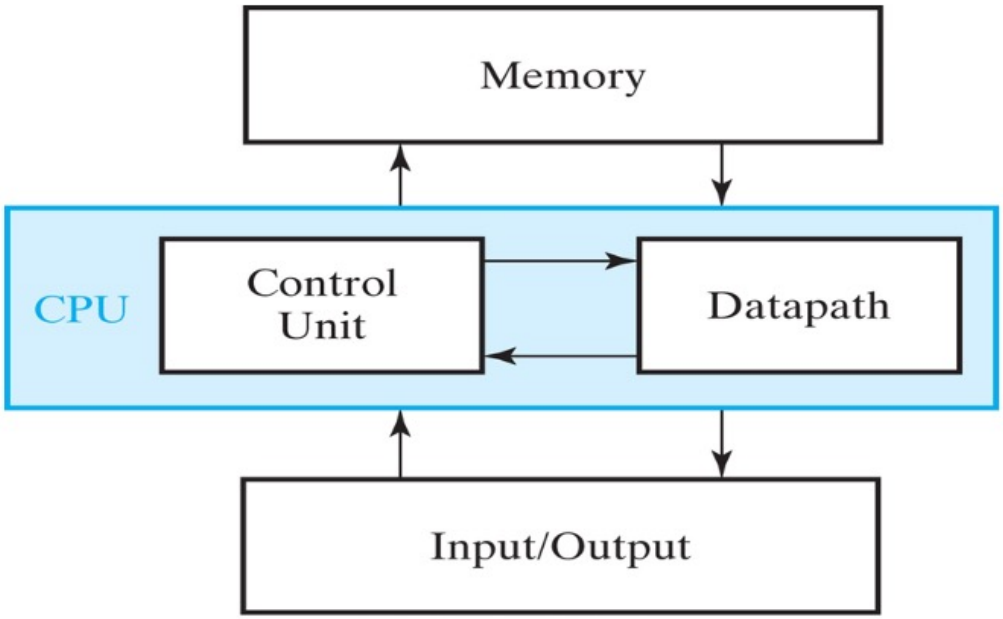
Introduction

- Computer Specification
  - Instruction Set Architecture (ISA) - the specification of a computer's appearance to a programmer at its lowest level.
  - Computer Architecture - a high-level description of the hardware implementing the computer derived from the ISA

Introduction

The architecture, for a simple computer, is typically divided into:

- Datapath for performing operations.
- Control unit for controlling datapath operations.



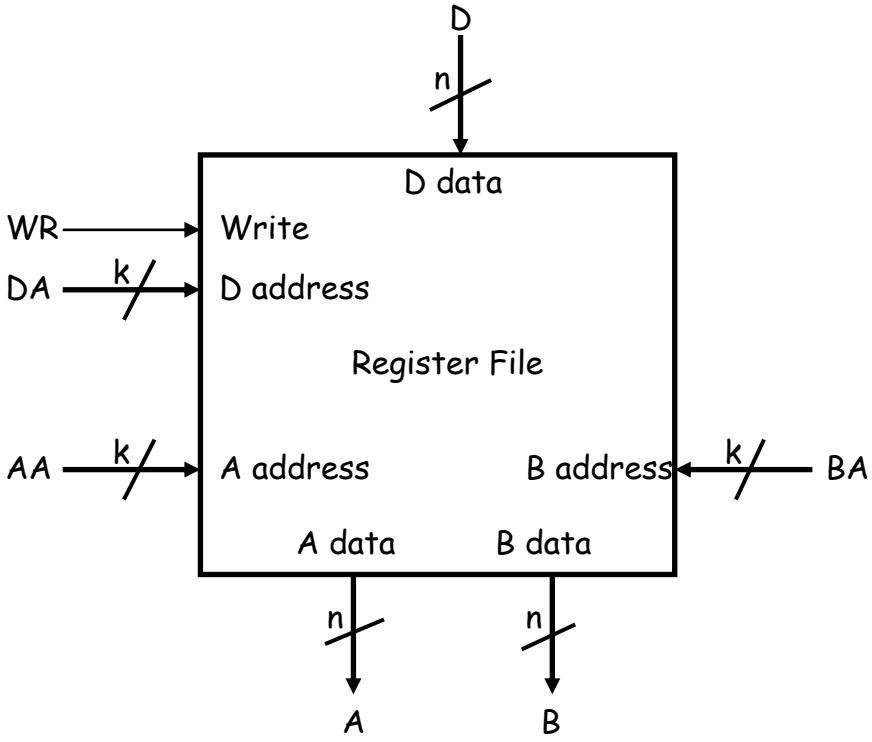
Datapath Design

A datapath is specified by:

- A set of registers with common access resources called a *register file*.
- One or more shared resources for implementing microoperations:
  - Buses - shared transfer paths.
  - Arithmetic-Logic Unit (ALU)
  - Shifter
- A control interface: signals coming into datapath.
- The datapath completes a single microoperation each clock cycle.

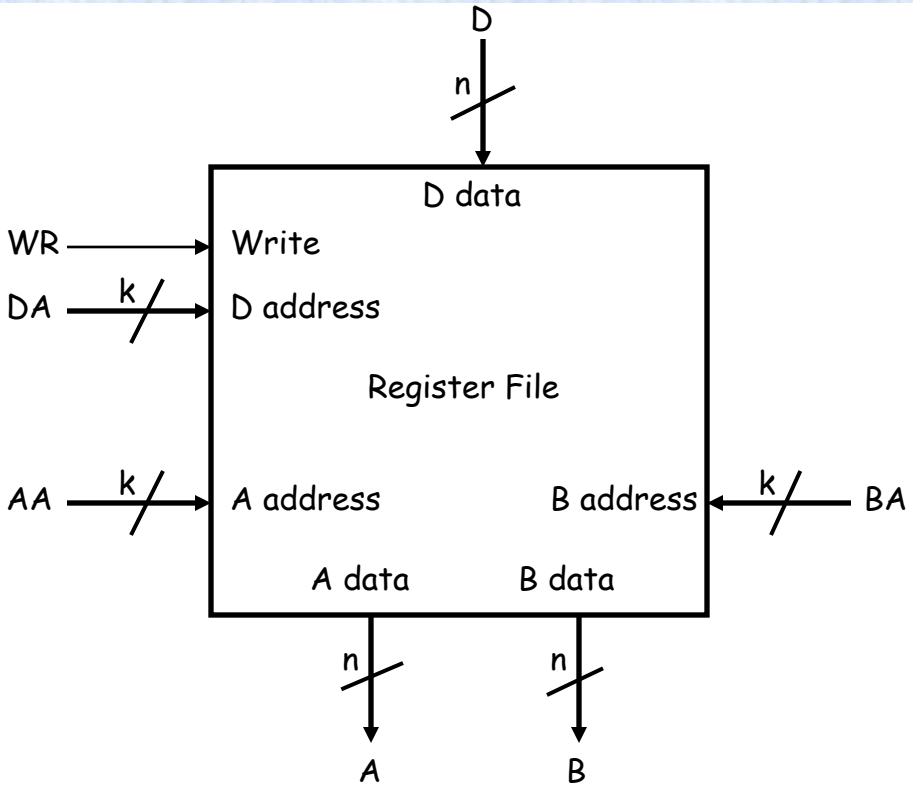
Register File

- Modern processors contain an array of fast registers grouped together in a register file.
- The register file appears like a memory based on clocked flip-flops (the clock is not shown)
- Much like words stored in a RAM, individual registers are identified by an address.
- Here is a block symbol for a 2^k x n register file.
  - There are 2^k registers, so register addresses are k bits long.
  - Each register holds an n-bit word, so the data inputs and outputs are n bits wide.



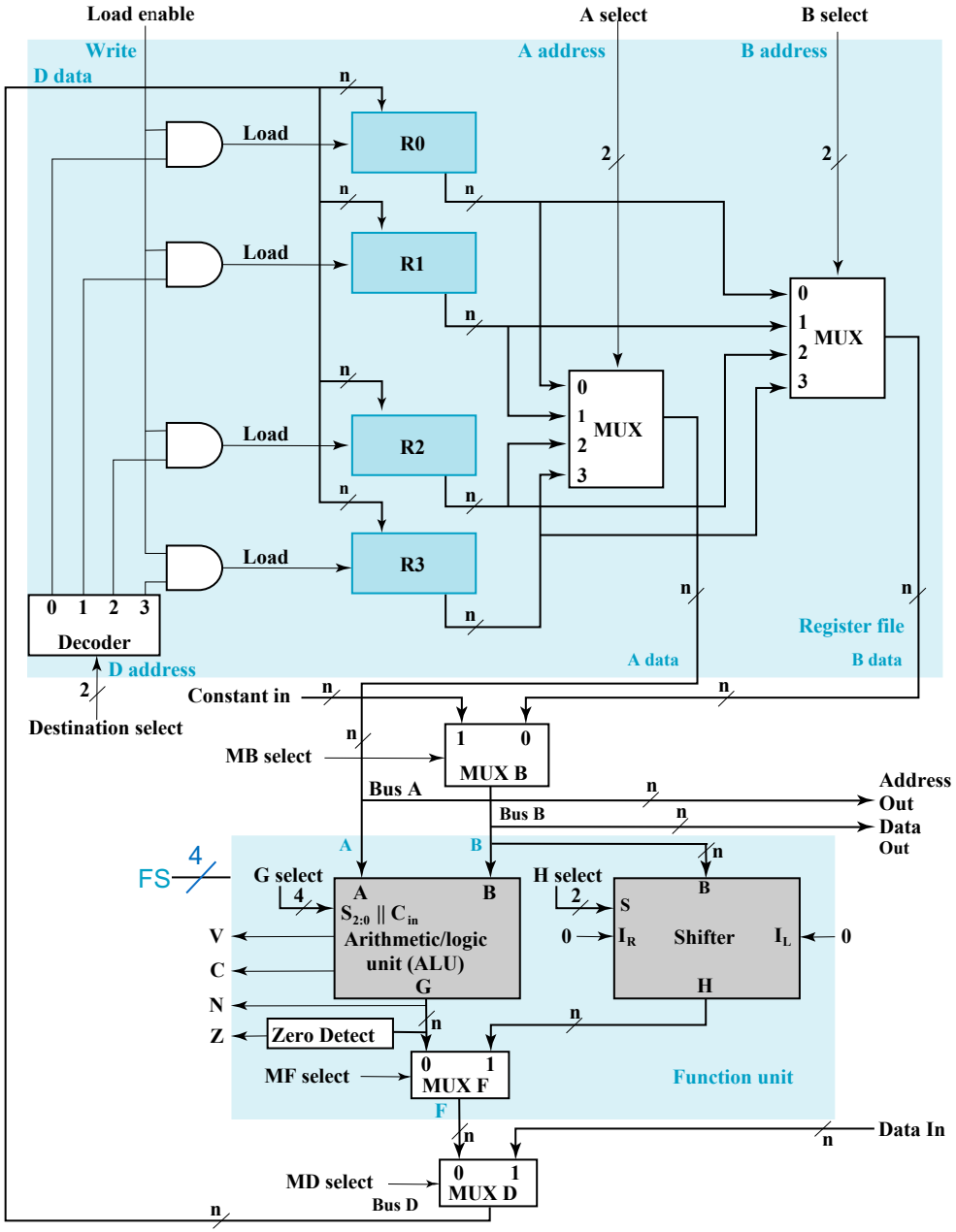
Accessing the Register File

- You can read two registers at once by supplying the AA and BA inputs. The data appears on the A and B outputs.
- You can write to a register by using the DA and D inputs, and setting WR = 1.
- These are registers so there must be a clock signal, even though we usually don't show it in diagrams.



Datapath Example

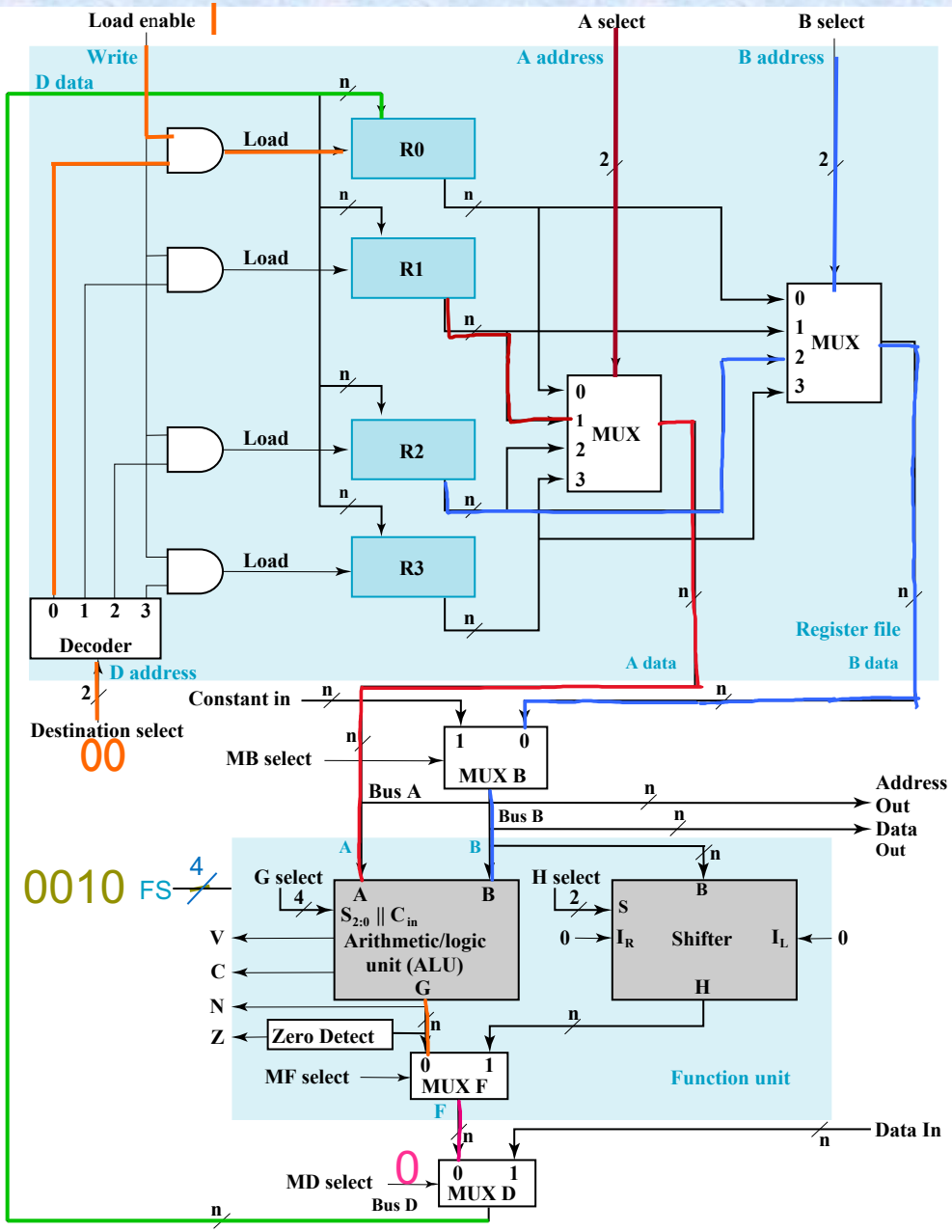
- Four parallel-load registers.
- Two mux-based register selectors.
- Register destination decoder.
- Mux B for external constant input.
- Buses A and B with external address and data outputs.
- ALU and Shifter with Mux F for output select.
- Mux D for external data input.
- Logic for generating status bits V, C, N, Z.



Datapath Example

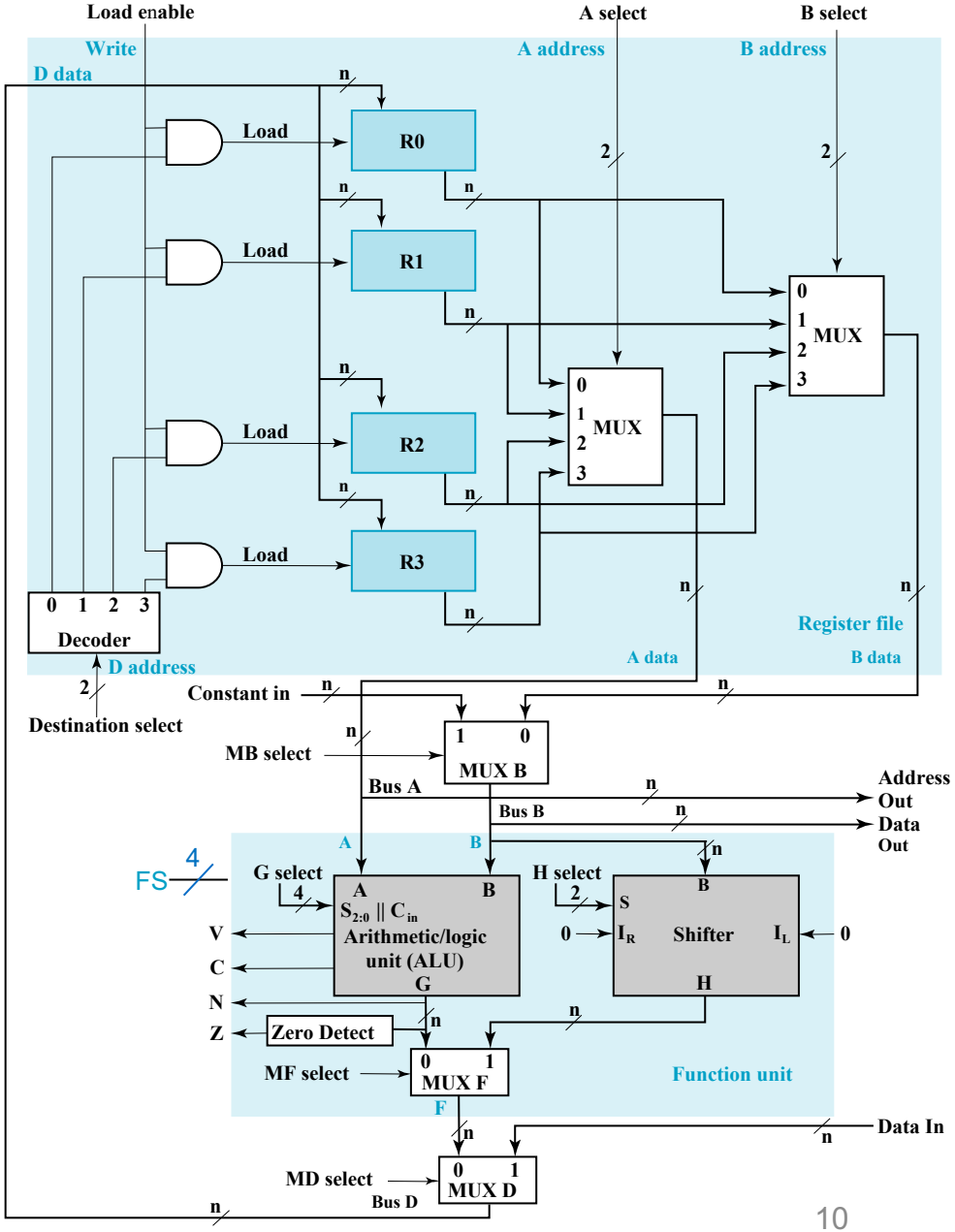
Microoperation: R0 ← R1 + R2

- Apply 01 to A select to place contents of R1 onto Bus A
- Apply 10 to B select to place contents of R2 onto B data
- Apply 0 to MB select to place B data on Bus B
- Apply 0010 to FS select to perform addition G = Bus A + Bus B
- Apply 0 to MD select to place the value of G onto BUS D
- Apply 00 to Destination select to enable the Load input to R0
- Apply 1 to Load Enable to force the Load input to R0 to 1 so that R0 is loaded on the clock pulse (not shown)
- The overall microoperation requires 1 clock cycle



Datapath Example: Key Control Actions for Microoperation Alternatives

- Perform a shift microoperation – apply 1 to MF select
- Use a constant in a micro-operation using Bus B – apply 1 to MB select
- Provide an address and data for a memory or output write microoperation – apply 0 to Load enable to prevent register loading
- Provide an address and obtain data for a memory or output read microoperation – apply 1 to MD select
- For some of the above, other control signals become don't cares



Simplifying the Data Path Representation

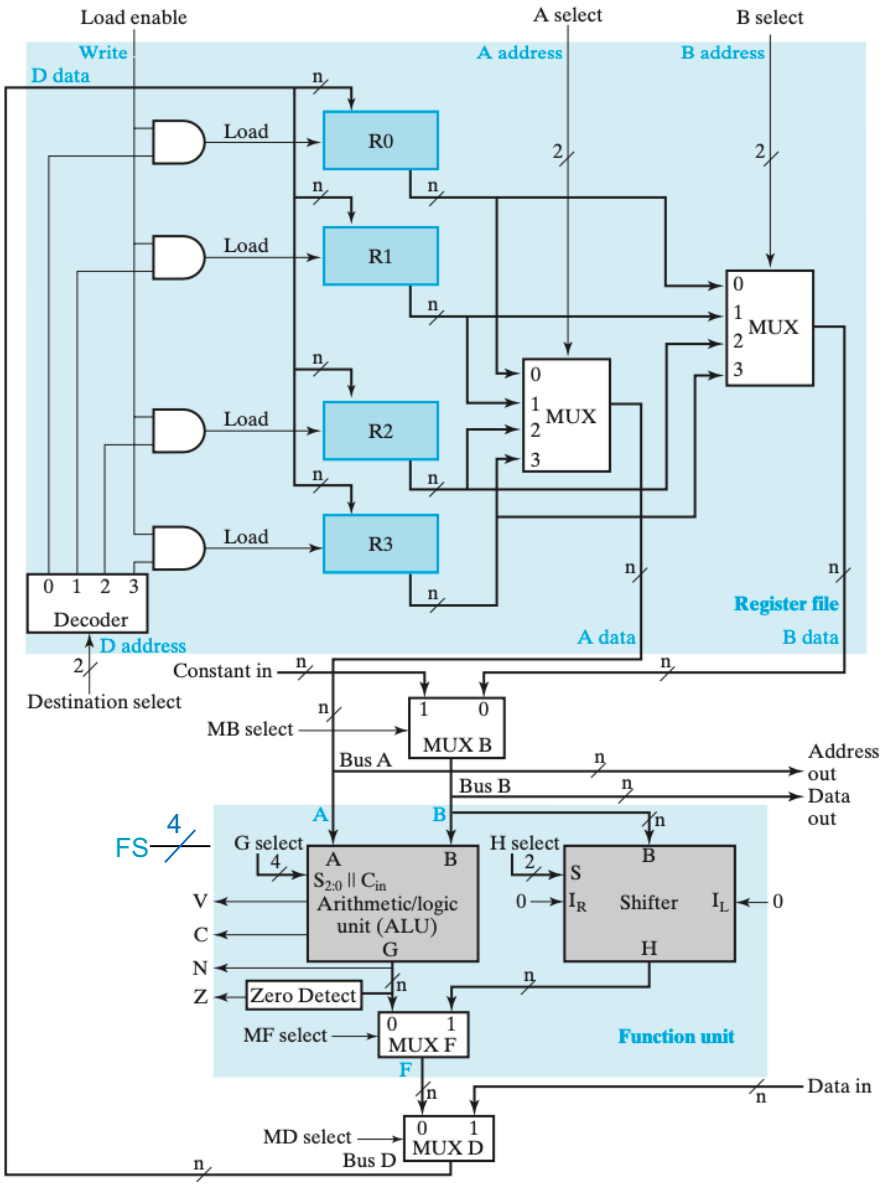


FIGURE 8-1
Block Diagram of a Generic Datapath

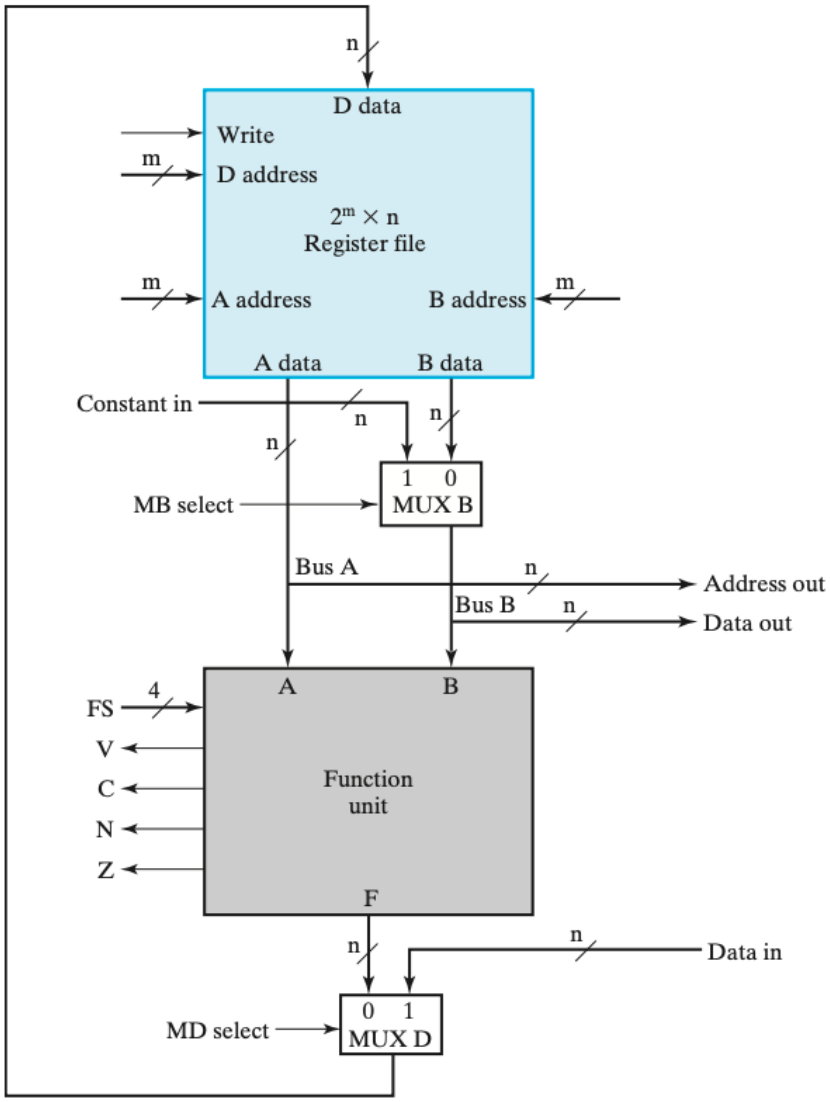


FIGURE 8-10
Block Diagram of Datapath Using the Register File and Function Unit

Definition of Function Unit Select (FS) Codes

FS	MF Select	G Select	H Select	Microoperation
0000	0	0000	XX	$F \leftarrow A$
0001	0	0001	XX	$F \leftarrow A + 1$
0010	0	0010	XX	$F \leftarrow A + B$
0011	0	0011	XX	$F \leftarrow A + B + 1$
0100	0	0100	XX	$F \leftarrow A + \overline{B}$
0101	0	0101	XX	$F \leftarrow A + \overline{B} + 1$
0110	0	0110	XX	$F \leftarrow A - 1$
0111	0	0111	XX	$F \leftarrow A$
1000	0	1X00	XX	$F \leftarrow A \wedge B$
1001	0	1X01	XX	$F \leftarrow A \vee B$
1010	0	1X10	XX	$F \leftarrow A \oplus B$
1011	0	1X11	XX	$F \leftarrow \overline{A}$
1100	1	XXXX	00	$F \leftarrow B$
1101	1	XXXX	01	$F \leftarrow \text{sr } B$
1110	1	XXXX	10	$F \leftarrow \text{sl } B$

Boolean Equations:

$MF = FS_3\ FS_2$

$G_3 = FS_3$

$G_2 = FS_2$

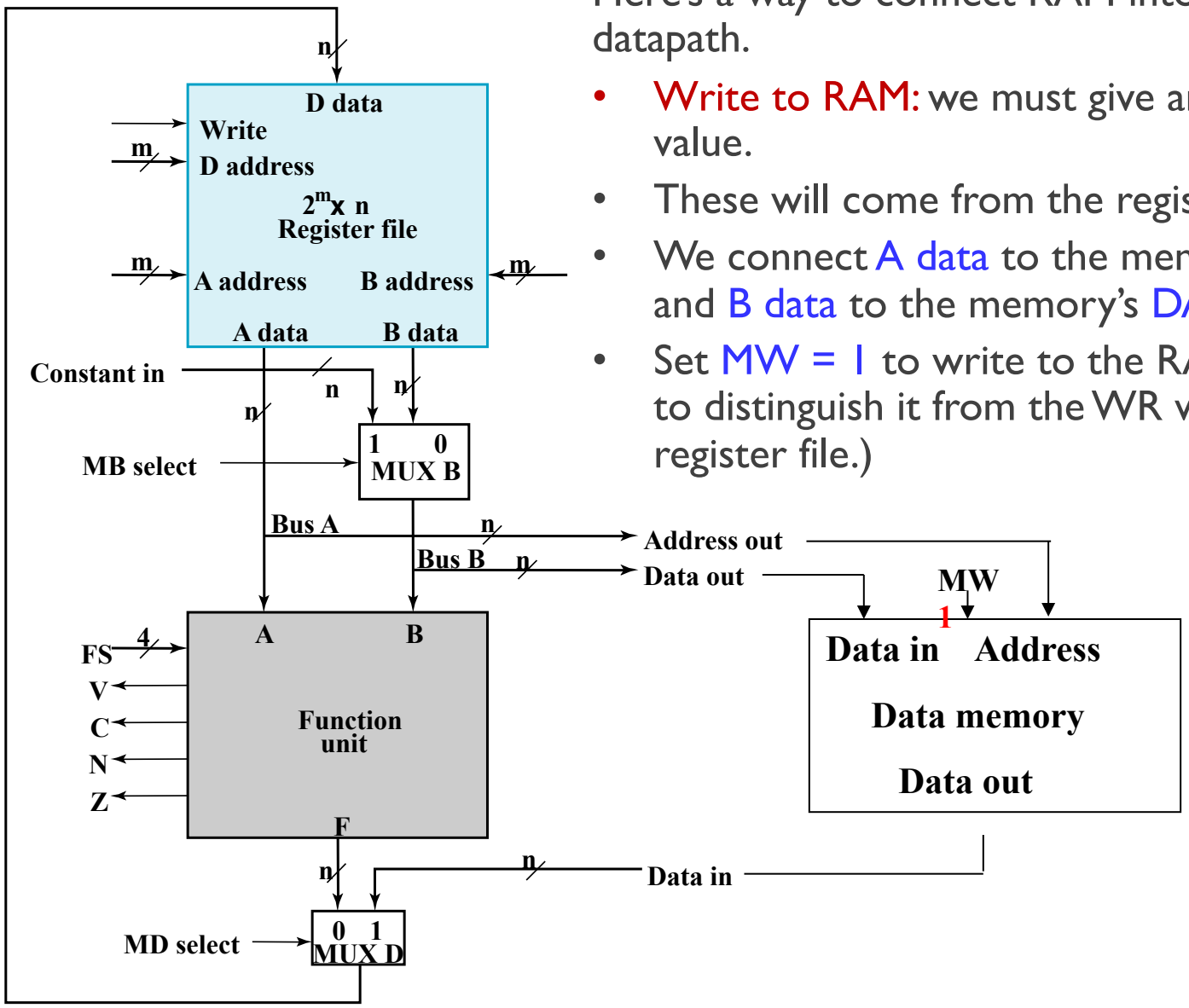
$G_1 = FS_1$

$G_0 = FS_0$

$H_1 = FS_1$

$H_0 = FS_0$

Access RAM

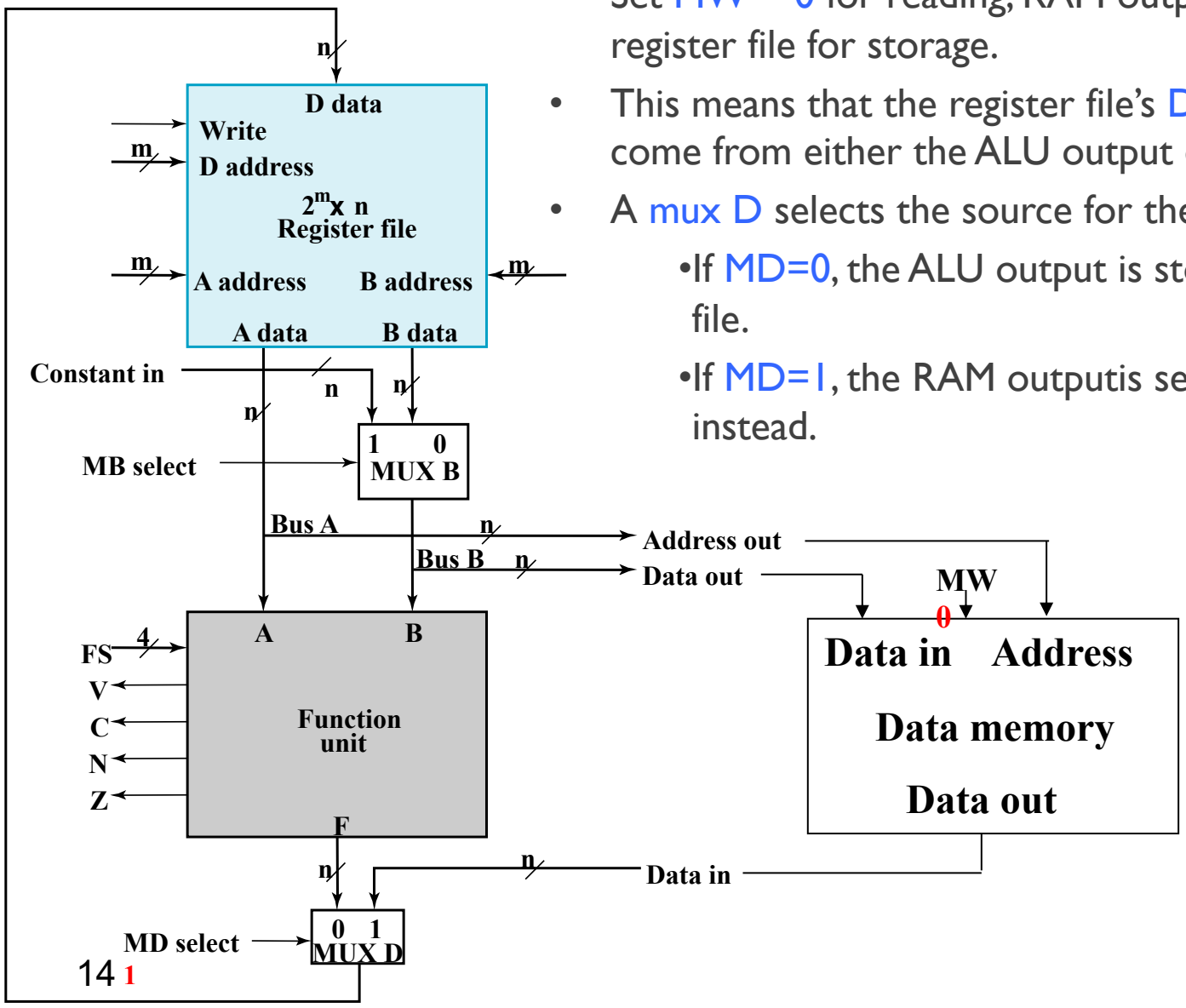


Here's a way to connect RAM into our existing datapath.

- Write to RAM: we must give an address and a data value.
- These will come from the register file.
- We connect A data to the memory's ADRS input, and B data to the memory's DATA input.
- Set MW = 1 to write to the RAM. (It's called MW to distinguish it from the WR write signal on the register file.)

Access RAM

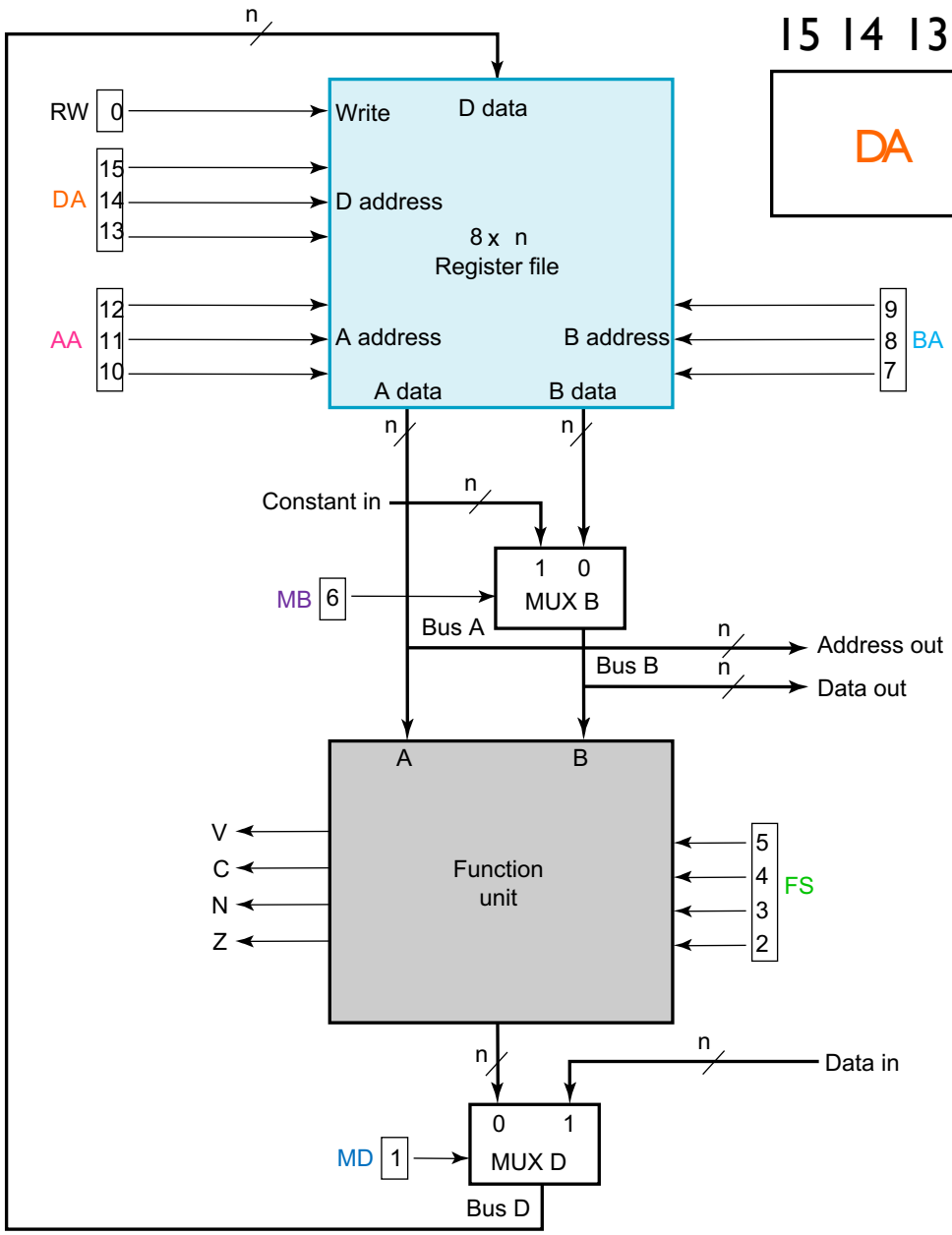
- Read from RAM: A data must supply the address.
- Set MW = 0 for reading, RAM output should be sent to register file for storage.
- This means that the register file's D data input could come from either the ALU output or the RAM.
- A mux D selects the source for the register file.
  - If MD=0, the ALU output is stored in the register file.
  - If MD=1, the RAM outputis sent to the registers instead.

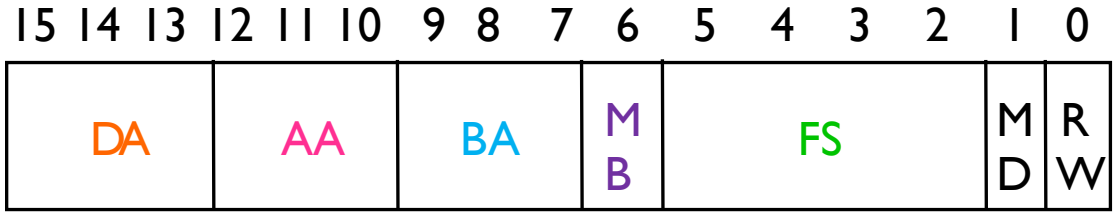


The Control Word

- The datapath has many control inputs
- The signals driving these inputs can be defined and organized into a *control word*
- To execute a microinstruction, we apply control word values for a clock cycle.
  - Addresses for the data read from register file.
  - Function performed.
  - Addresses for the data written to register file.
  - External data.

The Control Word Fields

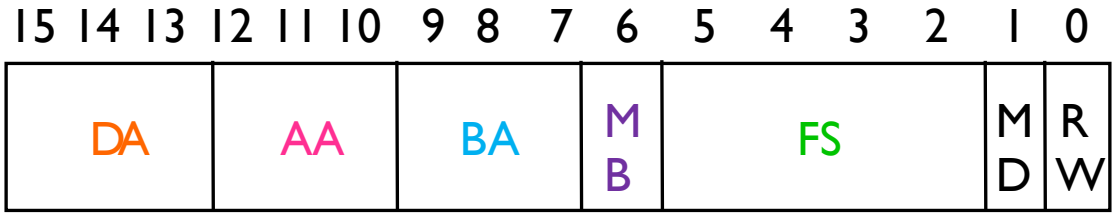




Control word

- **The Control Words Fields**
  - DA – D Address (destination)
  - AA – A Address
  - BA – B Address
  - MB – Mux B (constant/source)
  - FS – Function Select
  - MD – Mux D
  - RW – Register Write

The Control Word



Control word

- Fields
  - DA – D Address
  - AA – A Address
  - BA – B Address
  - MB – Mux B
  - FS – Function Select
  - MD – Mux D
  - RW – Register Write

3 bits each. Means we have up to 8 registers.

2nd parameter to function from register (0) or a constant (1).

Load to register external data or result of function.

Encoding The Control Word

TABLE 8-5
Encoding of Control Word for the Datapath

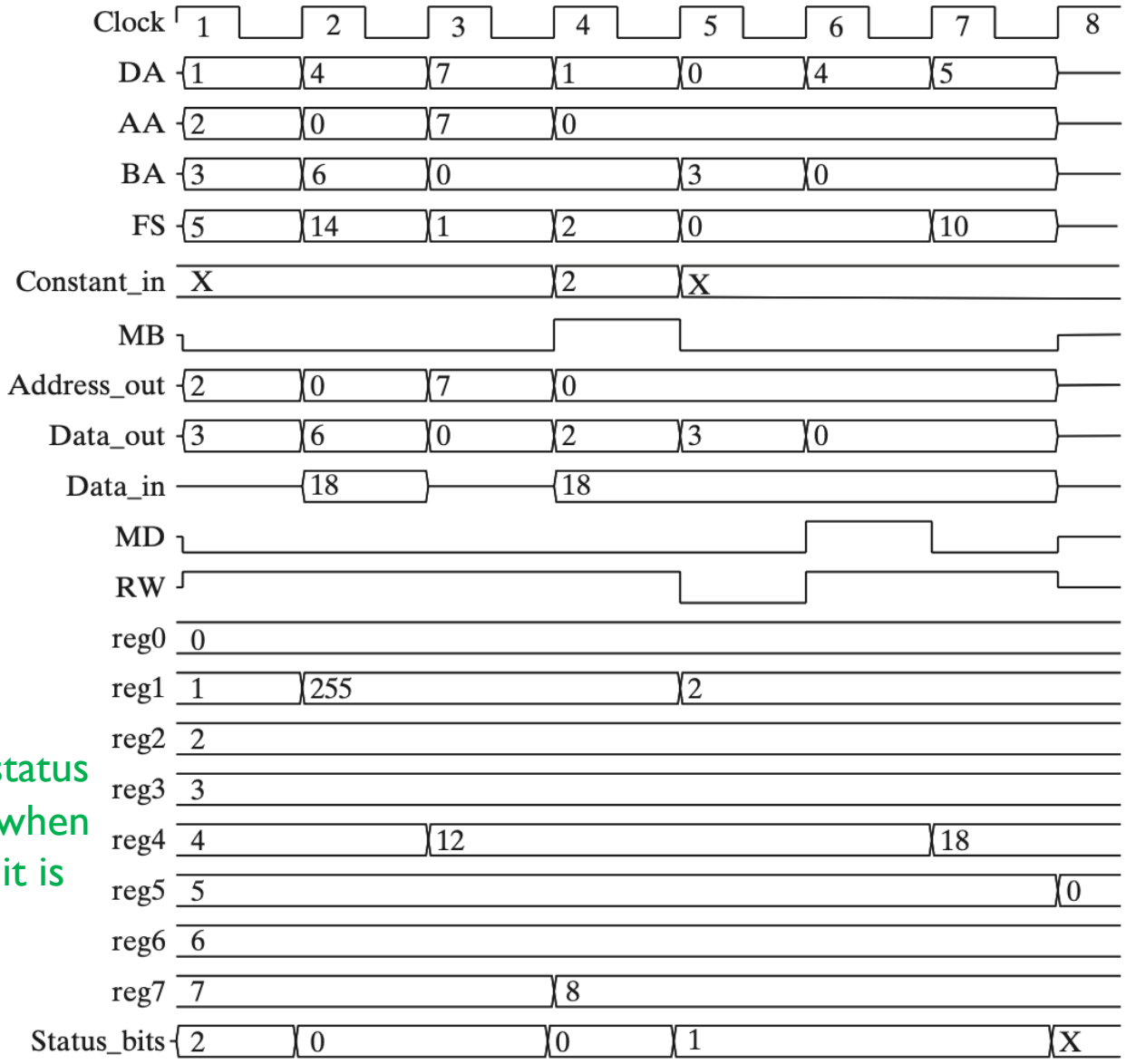
DA, AA, BA		MB		FS		MD		RW	
Function	Code	Function	Code	Function	Code	Function	Code	Function	Code
$R0$	000	Register	0	$F = A$	0000	Function	0	No Write	0
$R1$	001	Constant	1	$F = A + 1$	0001	Data in	1	Write	1
$R2$	010			$F = A + B$	0010				
$R3$	011			$F = A + B + 1$	0011				
$R4$	100			$F = A + \overline{B}$	0100				
$R5$	101			$F = A + \overline{B} + 1$	0101				
$R6$	110			$F = A - 1$	0110				
$R7$	111			$F = A$	0111				
				$F = A \wedge B$	1000				
				$F = A \vee B$	1001				
				$F = A \oplus B$	1010				
				$F = \overline{A}$	1011				
				$F = B$	1100				
				$F = \text{sr } B$	1101				
				$F = \text{sl } B$	1110				

Example Microoperations for the Datapath

Symbolic & Binary Representation

Micro-operation	DA	AA	BA	MB	FS	MD	RW
$R1 \leftarrow R2 - R3$	$R1$ 001	$R2$ 010	$R3$ 011	Register 0	$F = A + \overline{B} + 1$ 0101	Function 0	Write 1
$R4 \leftarrow$ sl R6	$R4$ 100	— xxx	$R6$ 110	Register 0	$F =$ sl B 1110	Function 0	Write 1
$R7 \leftarrow R7 + 1$	$R7$ 111	$R7$ 111	— xxx	— x	$F = A + 1$ 0001	Function 0	Write 1
$R1 \leftarrow R0 + 2$	$R1$ 001	$R0$ 000	— xxx	Constant 1	$F = A + B$ 0010	Function 0	Write 1
Data out $\leftarrow R3$	—— xxx	—— xxx	$R3$ 011	Register 0	— xxxx	—— x	No Write 0
$R4 \leftarrow$ Data in	$R4$ 100	—— xxx	—— xxx	—— x	— xxxx	Data in 1	Write 1
$R5 \leftarrow 0$	$R5$ 101	$R0$ 000	$R0$ 000	Register 0	$F = A \oplus B$ 1010	Function 0	Write 1

Simulation of the Microoperation Sequence



the value of the status bits are relevant when the Function unit is used.

(V, C, N, Z)

FIGURE 8-12
Simulation of the Microoperation Sequence in Table 8-7