Digital Electronics & Logic Design

(EC 207)

Dr. Vivek Garg

Department of Electronics Engineering

S. V. National Institute of Technology (SVNIT)
Surat

email: vivekg@eced.svnit.ac.in; vivekgarg0101@gmail.com

Course Outline

- **PN DIODE AND TRANSITOR** **(04 Hours)**

  PN Diode Theory, PN Characteristic and Breakdown Region, PN Diode Application as Rectifier, Zener Diode Theory, Zener Voltage Regulator, Diode as Clamper and Clipper, Photodiode Theory, LED Theory, 7 Segment LED Circuit Diagram and Multi Colour LED, LASER Diode Theory and Applications, Bipolar Junction Transistor Theory, Transistor Symbols And Terminals, Common Collector, Emitter and Base Configurations, Different Biasing Techniques, Concept of Transistor Amplifier, Introduction to FET Transistor And Its Feature.

- **WAVESHAPING CIRCUITS AND OPERATIONAL AMPLIFIER** **(06 Hours)**

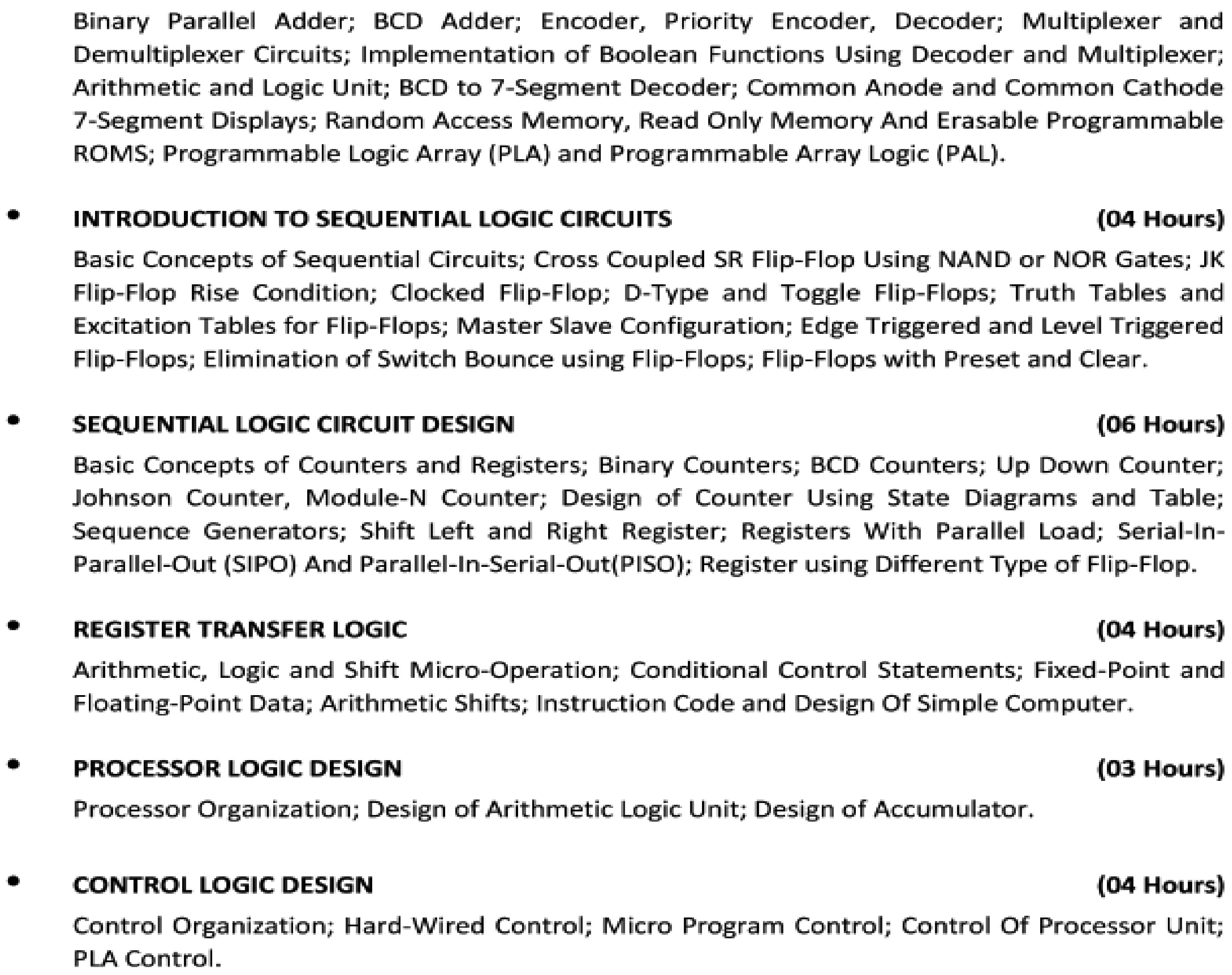
  Linear Wave Shaping Circuits, RC High Pass and Low Pass Circuits, RC Integrator and Differentiator Circuits, Nonlinear Wave Shaping Circuits, Two Level Diode Clipper Circuits, Clamping Circuits, Operational Amplifier OP-AMP with Block Diagram, Schematic Symbol of OP-AMP, The 741 Package Style and Pinouts, Specifications of Op-Amp, Inverting and Non-Inverting Amplifier, Voltage Follower Circuit, Multistage OP-AMP Circuit, OP-AMP Averaging Amplifier, OP-AMP Subtractor.

- **BOOLEAN ALGEBRA AND SWITCHING FUNCTIONS** **(04 Hours)**

  Basic Logic Operation and Logic Gates, Truth Table, Basic Postulates and Fundamental Theorems of Boolean Algebra, Standard Representations of Logic Functions- SOP and POS Forms, Simplification of Switching Functions-K-Map and Quine-Mccluskey Tabular Methods, Synthesis of Combinational Logic Circuits.

- **COMBINATIONAL LOGIC CIRCUIT USING MSI INTEGRATED CIRCUITS** **(07 Hours)**

Course Outline



Binary Parallel Adder; BCD Adder; Encoder, Priority Encoder, Decoder; Multiplexer and Demultiplexer Circuits; Implementation of Boolean Functions Using Decoder and Multiplexer; Arithmetic and Logic Unit; BCD to 7-Segment Decoder; Common Anode and Common Cathode 7-Segment Displays; Random Access Memory, Read Only Memory And Erasable Programmable ROMS; Programmable Logic Array (PLA) and Programmable Array Logic (PAL).

- **INTRODUCTION TO SEQUENTIAL LOGIC CIRCUITS** **(04 Hours)**

  Basic Concepts of Sequential Circuits; Cross Coupled SR Flip-Flop Using NAND or NOR Gates; JK Flip-Flop Rise Condition; Clocked Flip-Flop; D-Type and Toggle Flip-Flops; Truth Tables and Excitation Tables for Flip-Flops; Master Slave Configuration; Edge Triggered and Level Triggered Flip-Flops; Elimination of Switch Bounce using Flip-Flops; Flip-Flops with Preset and Clear.

- **SEQUENTIAL LOGIC CIRCUIT DESIGN** **(06 Hours)**

  Basic Concepts of Counters and Registers; Binary Counters; BCD Counters; Up Down Counter; Johnson Counter, Module-N Counter; Design of Counter Using State Diagrams and Table; Sequence Generators; Shift Left and Right Register; Registers With Parallel Load; Serial-In-Parallel-Out (SIPO) And Parallel-In-Serial-Out(PISO); Register using Different Type of Flip-Flop.

- **REGISTER TRANSFER LOGIC** **(04 Hours)**

  Arithmetic, Logic and Shift Micro-Operation; Conditional Control Statements; Fixed-Point and Floating-Point Data; Arithmetic Shifts; Instruction Code and Design Of Simple Computer.

- **PROCESSOR LOGIC DESIGN** **(03 Hours)**

  Processor Organization; Design of Arithmetic Logic Unit; Design of Accumulator.

- **CONTROL LOGIC DESIGN** **(04 Hours)**

  Control Organization; Hard-Wired Control; Micro Program Control; Control Of Processor Unit; PLA Control.

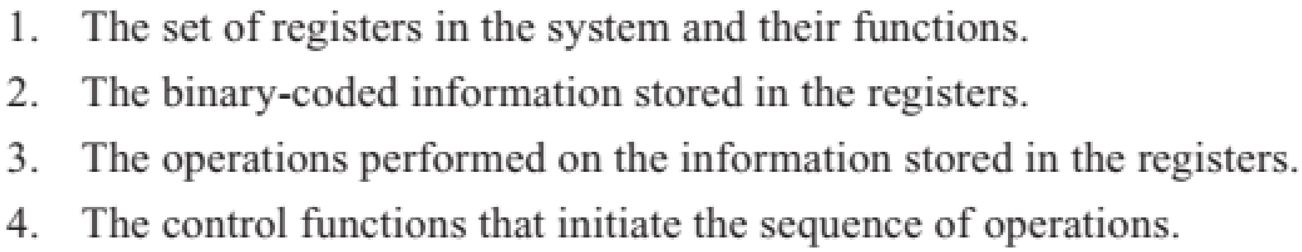
Course Text and Materials

1. Schilling Donald L. and Belove E., "Electronics Circuits- Discrete and Integrated", 3rd Ed., McGraw-Hill, 1989, Reprint 2008.
2. Millman Jacob, Halkias Christos C. and Parikh C., "Integrated Electronics", 2nd Ed., McGraw-Hill, 2009.
3. Taub H. and Mothibi Suryaprakash, Millman J., "Pulse, Digital and Switching Waveforms", 2nd Ed., McGraw-Hill, 2007.
4. Mano Morris, “Digital Logic and Computer Design”, 5th Ed., Pearson Education, 2005.
5. Lee Samual, “Digital Circuits and Logic Design”, 1st Ed., PHI, 1998.

Register Transfer Logic

Arithmetic, Logic and Shift Micro operations

- The operation of digital are described by specifying:



1. The set of registers in the system and their functions.
2. The binary-coded information stored in the registers.
3. The operations performed on the information stored in the registers.
4. The control functions that initiate the sequence of operations.

- **Register**: Shift Register, Counter, and memory units.
- **Binary Information**: Binary number, Binary Coded Decimal, Control Information, Alpha Numeric Number, etc.
- The **operation** performed on the data stored in the registers are called as micro operations. Ex, Add, Shift Left, Shift Right,…
- **Control Functions:** that initiate the sequence of operations consist of timing signals that sequence the operations one at a time.

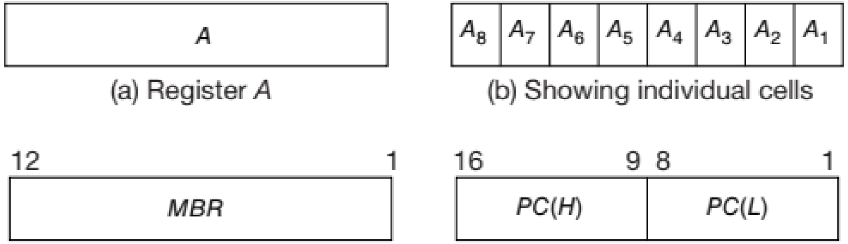
Register Transfer Logic

Arithmetic, Logic and Shift Micro operations

- Micro operations in digital systems can be broadly classified into four categories:

1. *Interregister-transfer* microoperations do not change the information content when the binary information moves from one register to another.
2. *Arithmetic* microoperations perform arithmetic on numbers stored in registers.
3. *Logic* microoperations perform operations such as AND and OR on individual pairs of bits stored in registers.
4. *Shift* microoperations specify operations for shift registers.

- **Inter Register Transfer:**
  Memory address register (MAR)
  A, B, R1, R2, IR,



(a) Register *A*

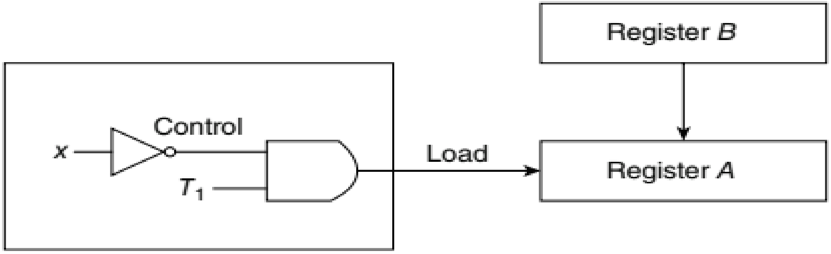
(b) Showing individual cells

(c) Numbering of cells

(d) Portions of a register

$$A \leftarrow B$$

$$x'T_1: A \leftarrow B$$



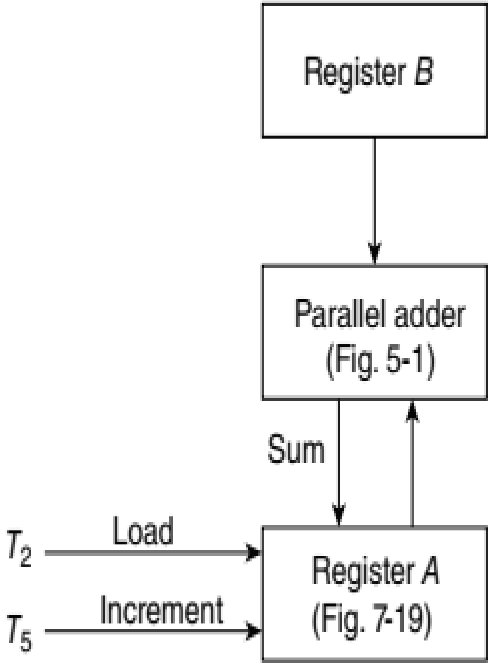
Register Transfer Logic

Arithmetic Micro operations

The arithmetic microoperation defined by the statement:

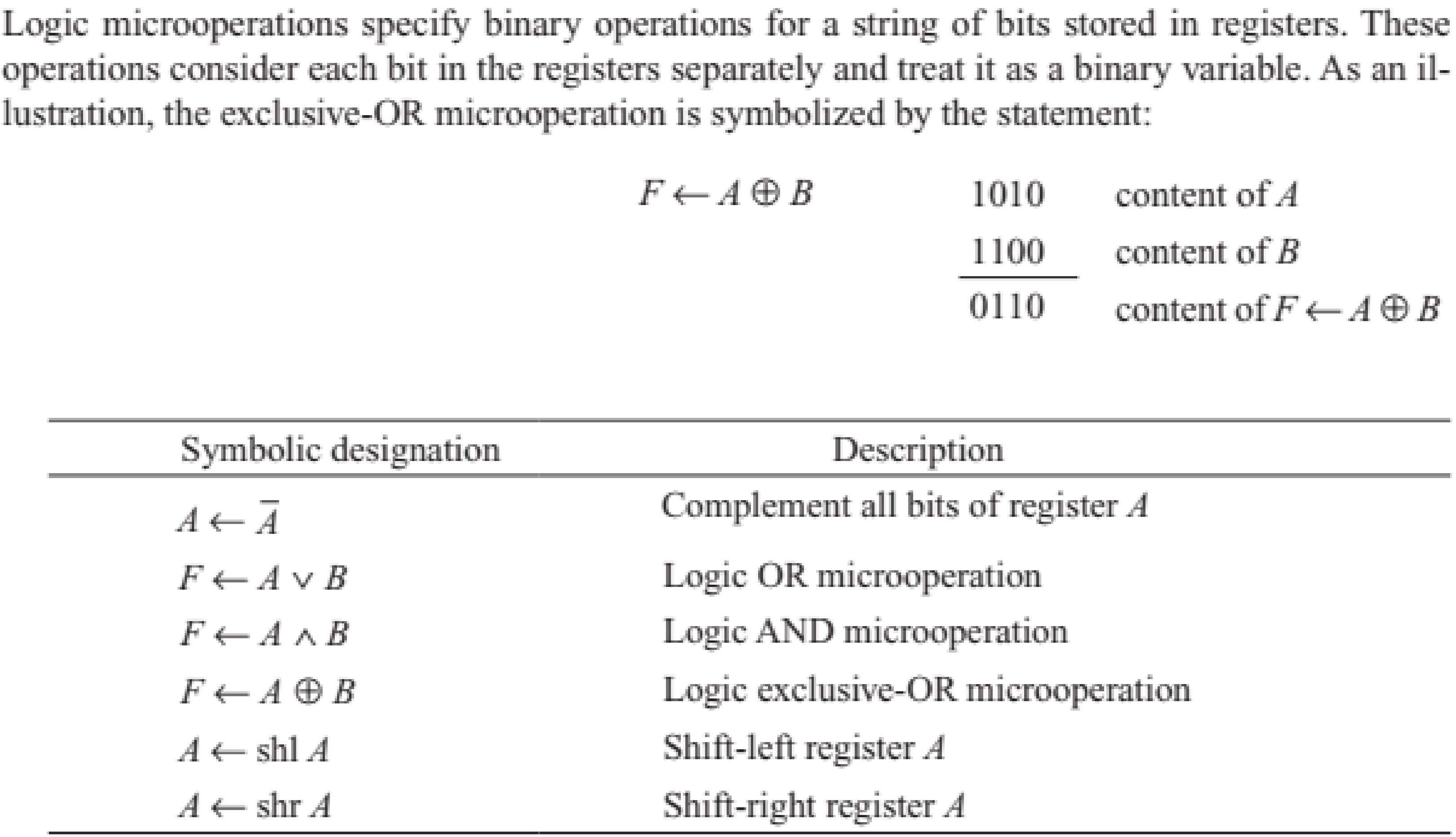
$$F \leftarrow A + B \qquad \begin{array}{l} T_2 \colon A \leftarrow A + B \\ T_5 A \leftarrow A + 1 \end{array}$$

Symbolic designation	Description
$F \leftarrow A + B$	Contents of A plus B transferred to F
$F \leftarrow A - B$	Contents of A minus B transferred to F
$B \leftarrow \overline{B}$	Complement register B (1's complement)
$B \leftarrow \overline{B} + 1$	Form the 2's complement of the contents of register B
$F \leftarrow A + \overline{B} + 1$	A plus the 2's complement of B transferred to F
$A \leftarrow A + 1$	Increment the contents of A by 1 (count up)
$A \leftarrow A - 1$	Decrement the contents of A by 1 (count down)



Register Transfer Logic

Logic Micro operations



Logic microoperations specify binary operations for a string of bits stored in registers. These operations consider each bit in the registers separately and treat it as a binary variable. As an illustration, the exclusive-OR microoperation is symbolized by the statement:

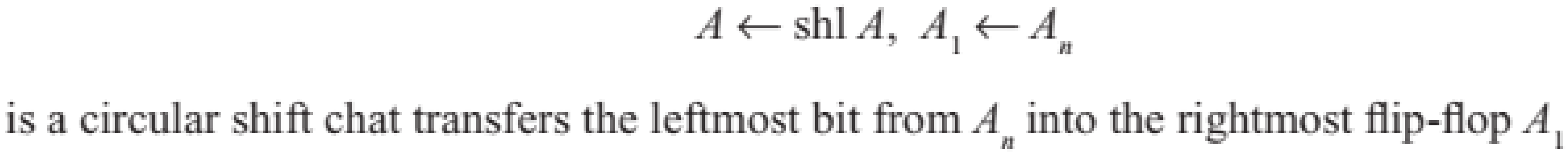
$$F \leftarrow A \oplus B \qquad \begin{array}{rl} 1010 & \text{content of } A \\ \underline{1100} & \text{content of } B \\ 0110 & \text{content of } F \leftarrow A \oplus B \end{array}$$

Symbolic designation	Description
$A \leftarrow \overline{A}$	Complement all bits of register A
$F \leftarrow A \vee B$	Logic OR microoperation
$F \leftarrow A \wedge B$	Logic AND microoperation
$F \leftarrow A \oplus B$	Logic exclusive-OR microoperation
$A \leftarrow \text{shl } A$	Shift-left register A
$A \leftarrow \text{shr } A$	Shift-right register A

Register Transfer Logic

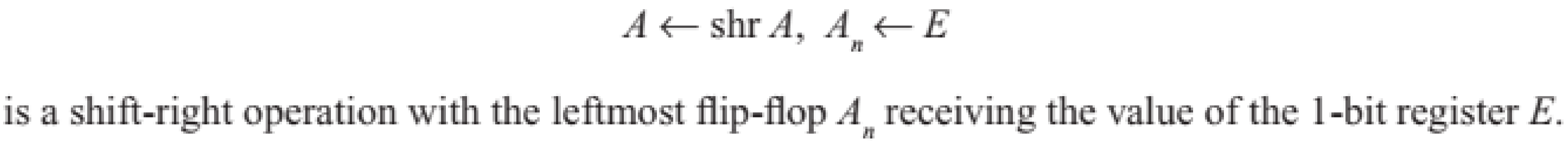
Shift Micro operations

$$A \leftarrow \text{shl } A, \qquad B \leftarrow \text{shr } B$$



$$A \leftarrow \text{shl } A, \;\; A_1 \leftarrow A_n$$

is a circular shift chat transfers the leftmost bit from A_n into the rightmost flip-flop A_1

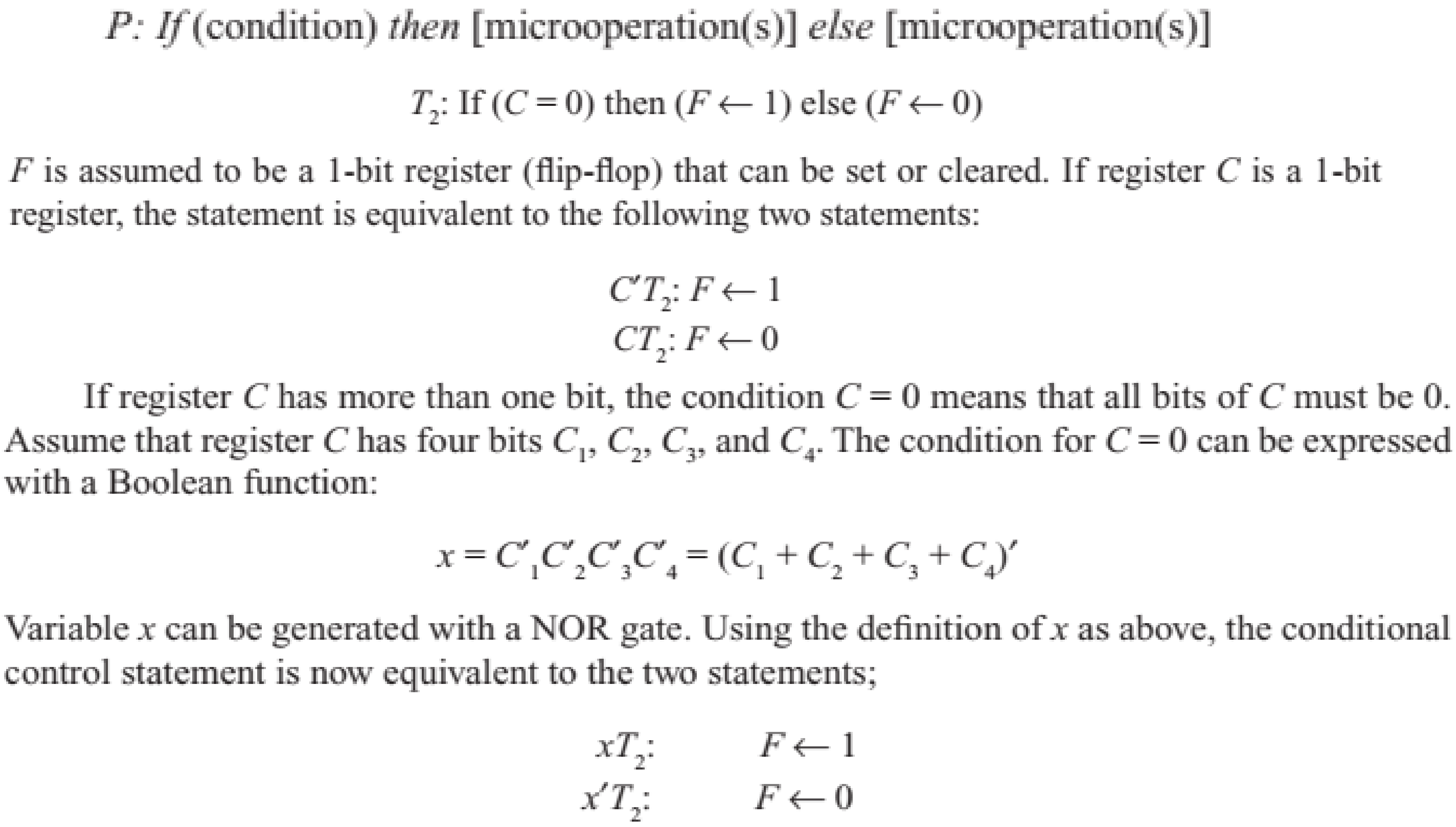


$$A \leftarrow \text{shr } A, \;\; A_n \leftarrow E$$

is a shift-right operation with the leftmost flip-flop A_n receiving the value of the 1-bit register E.

Register Transfer Logic

Conditional Control Statements



$$P\text{: } \textit{If} \text{ (condition) } \textit{then} \text{ [microoperation(s)] } \textit{else} \text{ [microoperation(s)]}$$

$$T_2\text{: If } (C = 0) \text{ then } (F \leftarrow 1) \text{ else } (F \leftarrow 0)$$

F is assumed to be a 1-bit register (flip-flop) that can be set or cleared. If register C is a 1-bit register, the statement is equivalent to the following two statements:

$$C'T_2\text{: } F \leftarrow 1$$
$$CT_2\text{: } F \leftarrow 0$$

If register C has more than one bit, the condition $C = 0$ means that all bits of C must be 0. Assume that register C has four bits C_1, C_2, C_3, and C_4. The condition for $C = 0$ can be expressed with a Boolean function:

$$x = C'_1C'_2C'_3C'_4 = (C_1 + C_2 + C_3 + C_4)'$$

Variable x can be generated with a NOR gate. Using the definition of x as above, the conditional control statement is now equivalent to the two statements;

$$xT_2\text{:} \qquad F \leftarrow 1$$
$$x'T_2\text{:} \qquad F \leftarrow 0$$

Register Transfer Logic

Fixed Point Data

- **Signed Binary Number**

1. Sign-magnitude.
2. Sign-1's complement.
3. Sign-2's complement.

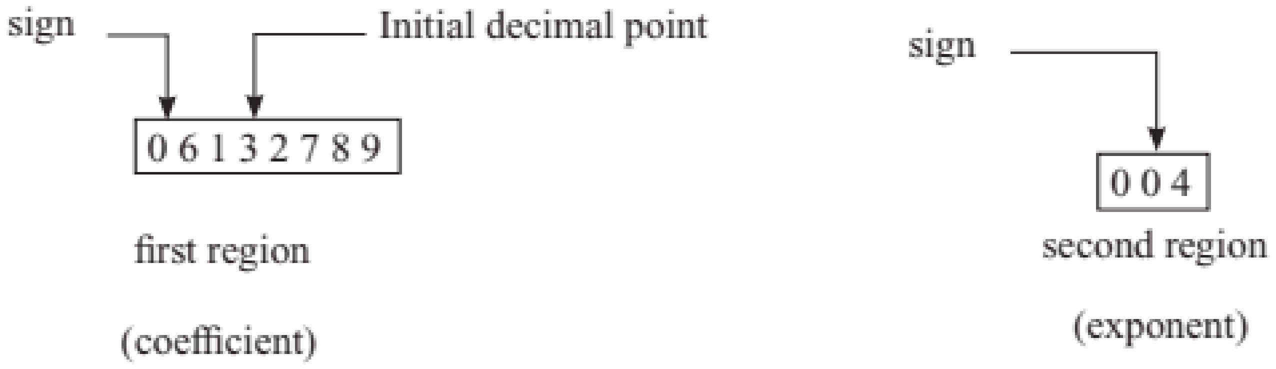
	+ 9	− 9
Sign-magnitude	0 001001	1 001001
Sign-1's complement	0 001001	1 110110
Sign-2's complement	0 001001	1 110111

$$\begin{array}{rrl} + & 6 & 0\,000110 \\ & & \qquad\quad + \\ + & 9 & 0\,001001 \\ \hline + & 15 & 0\,001111 \end{array} \qquad \begin{array}{rrl} - & 6 & 1\,111010 \\ & & \qquad\quad + \\ + & 9 & 0\,001001 \\ \hline + & 3 & 0\,000010 \end{array} \qquad \begin{array}{rrl} + & 6 & 0\,000110 \; + \\ & & \\ - & 9 & 1\,110111 \\ \hline - & 3 & 1\,111101 \end{array} \qquad \begin{array}{rrl} - & 9 & 1\,110111 \\ & & \qquad\quad + \\ - & 9 & 1\,110111 \\ \hline - & 18 & 1\,101110 \end{array}$$

$$\begin{array}{rrl} + & 6 & 0\,000110 \\ & & \qquad\quad + \\ + & 9 & 0\,001001 \\ \hline + & 15 & 0\,001111 \\ & & \\ \hline \end{array} \qquad \begin{array}{rrr} - & 6 & 1\,111001 \\ & & + \\ + & 9 & 0\,001001 \\ \hline & & 10\,000010 \\ & & + \\ & & \longrightarrow 1 \\ \hline + & 3 & 0\,000011 \end{array}$$

Floating Point Data

+6132.789



sign

Initial decimal point

0 6 1 3 2 7 8 9

first region

(coefficient)

sign

0 0 4

second region

(exponent)

$+.2601000 \times 10^{-4} = +.000026010000$

0 2 6 0 1 0 0 0

coefficient

1 0 4

exponent

$-.2601000 \times 10^{12} = -260100000000$

1 2 6 0 1 0 0 0

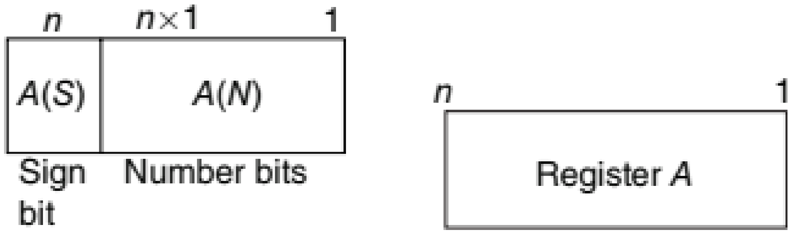
coefficient

0 1 2

exponent

Register Transfer Logic

Arithmetic Shift



$A(N) \leftarrow \text{shr } A(N)$, $A_{n-1} \leftarrow 0$ for sign-magnitude

$A \leftarrow \text{shr } A$, $A(A) \leftarrow A(S)$ for sign-1's or sign-2's complement

Positive number	+12:	0 01100	+6:	0 00110
Sign-magnitude	−12:	1 01100	−6:	1 00110
Sign-1's complement	−12:	1 10011	−6:	1 11001
Sign-2's complement	−12:	1 10100	−6:	1 11010

$A(N) \leftarrow \text{shl } A(N)$, $A_1 \leftarrow 0$ for sign-magnitude

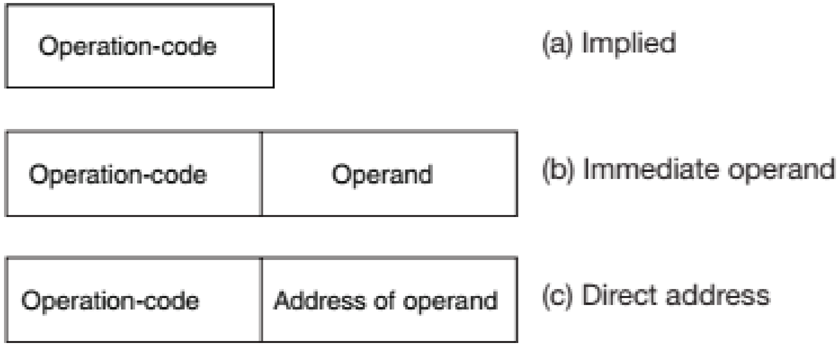
$A \leftarrow \text{shl } A$, $A_1 \leftarrow A(S)$ for sign-1's complement

$A \leftarrow \text{shl } A$, $A_1 \leftarrow 0$ for sign-2's complement

Positive number	0 01100	0 11000
Sign-magnitude	1 01100	1 11000
Sign-1's complement	1 10011	1 00111
Sign-2's complement	1 10100	1 01000

Instruction Code and Design of Simple Computer

- An *instruction code* is a group of bits that tell the computer to perform a specific operation.



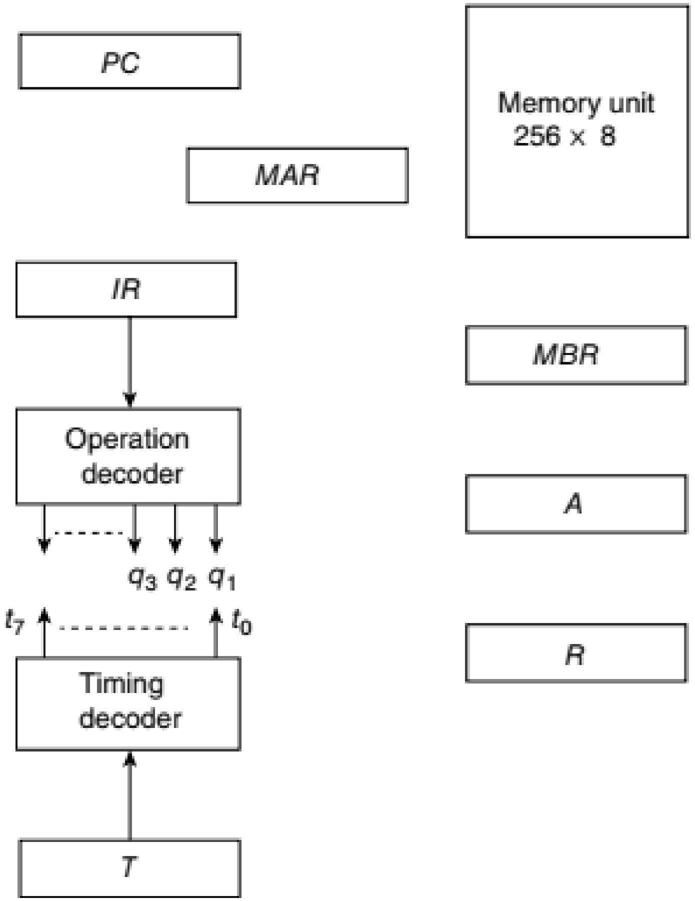
Operation-code — (a) Implied

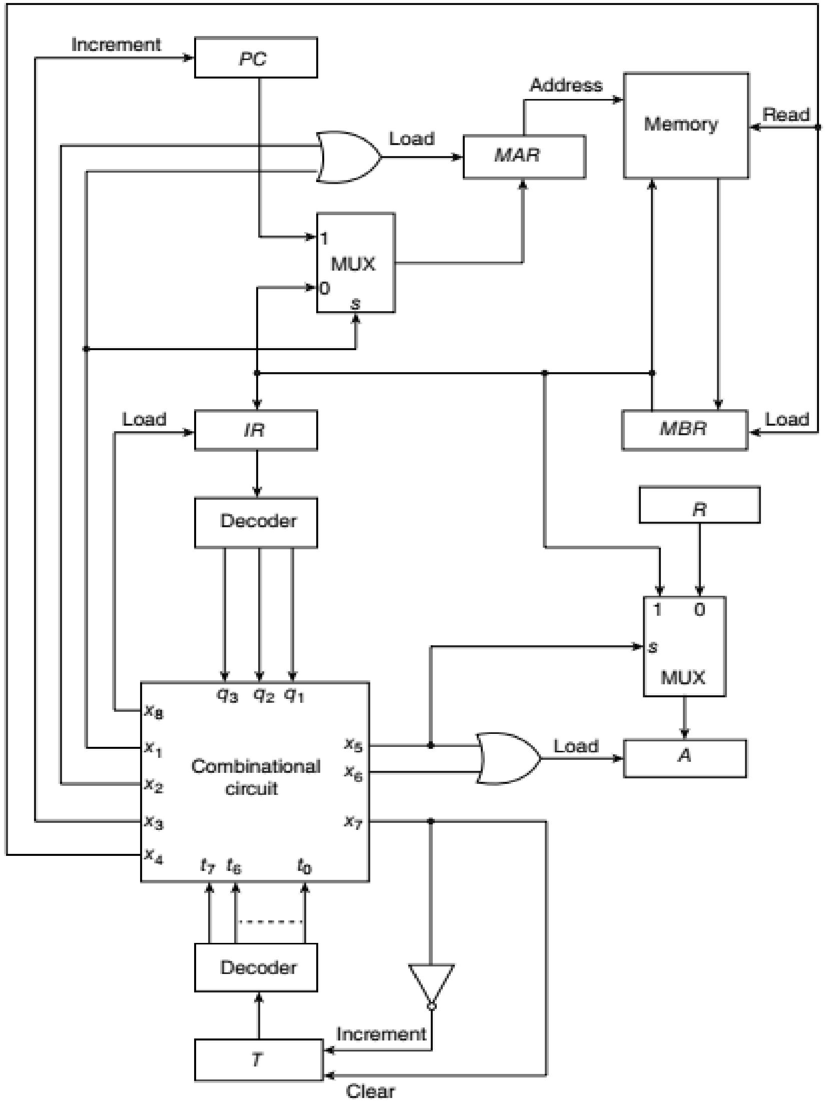
Operation-code | Operand — (b) Immediate operand

Operation-code | Address of operand — (c) Direct address

Register Transfer Logic

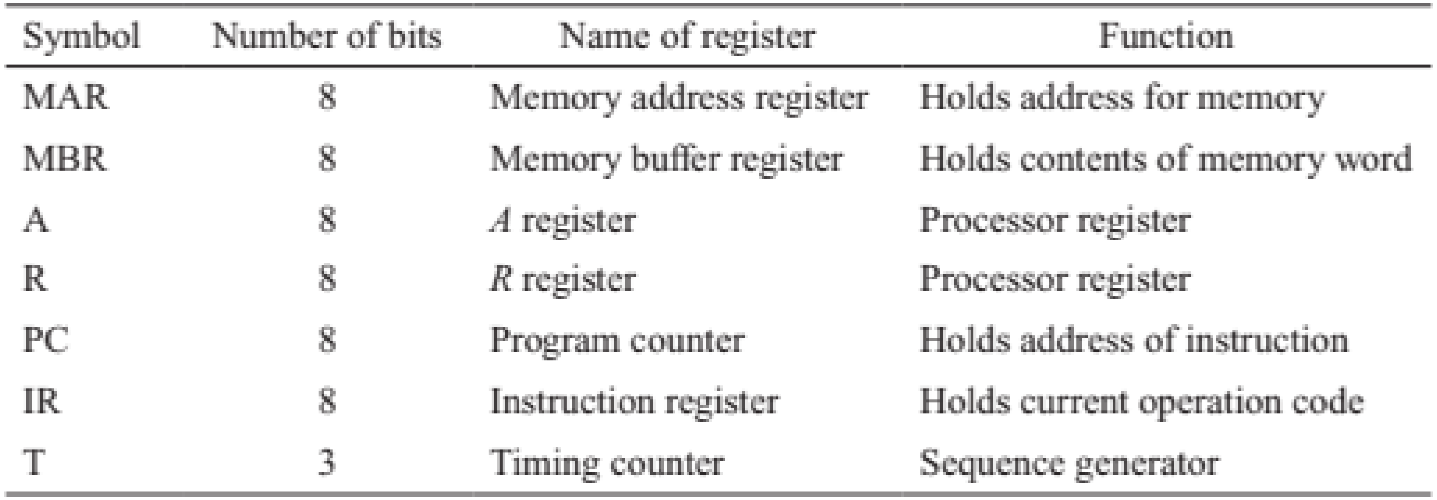
Design of Simple Computer



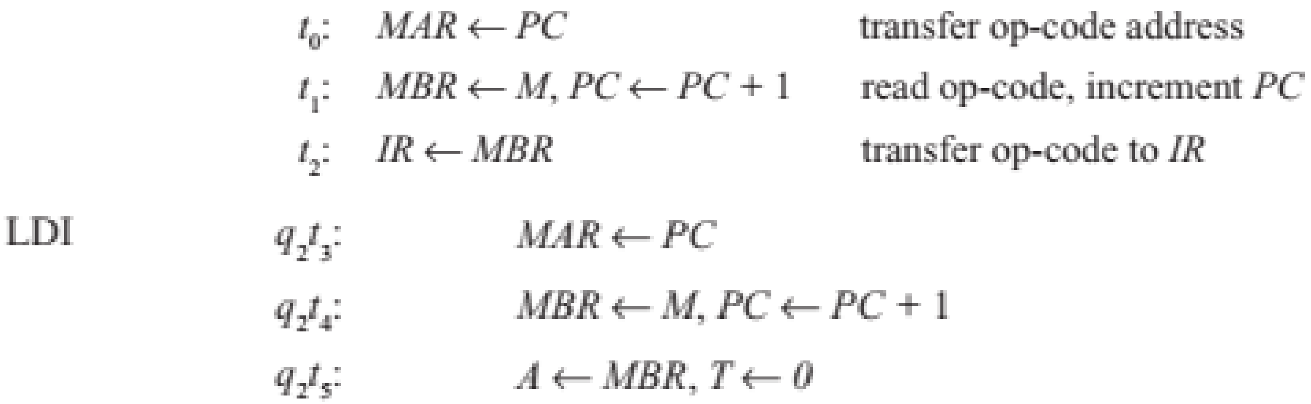


Register Transfer Logic

Instruction Code and Design of Simple Computer



Symbol	Number of bits	Name of register	Function
MAR	8	Memory address register	Holds address for memory
MBR	8	Memory buffer register	Holds contents of memory word
A	8	*A* register	Processor register
R	8	*R* register	Processor register
PC	8	Program counter	Holds address of instruction
IR	8	Instruction register	Holds current operation code
T	3	Timing counter	Sequence generator



t_0: $MAR \leftarrow PC$ transfer op-code address

t_1: $MBR \leftarrow M, PC \leftarrow PC + 1$ read op-code, increment *PC*

t_2: $IR \leftarrow MBR$ transfer op-code to *IR*

LDI q_2t_3: $MAR \leftarrow PC$

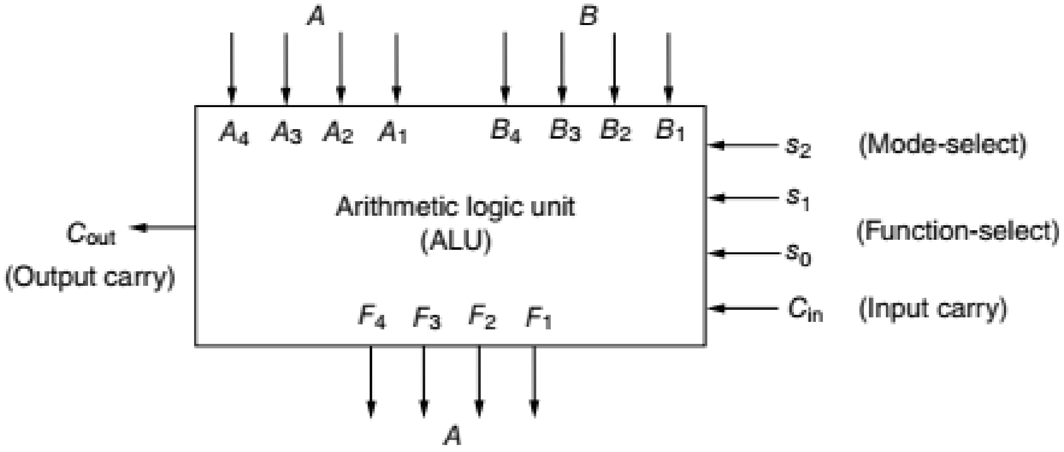
q_2t_4: $MBR \leftarrow M, PC \leftarrow PC + 1$

q_2t_5: $A \leftarrow MBR, T \leftarrow 0$

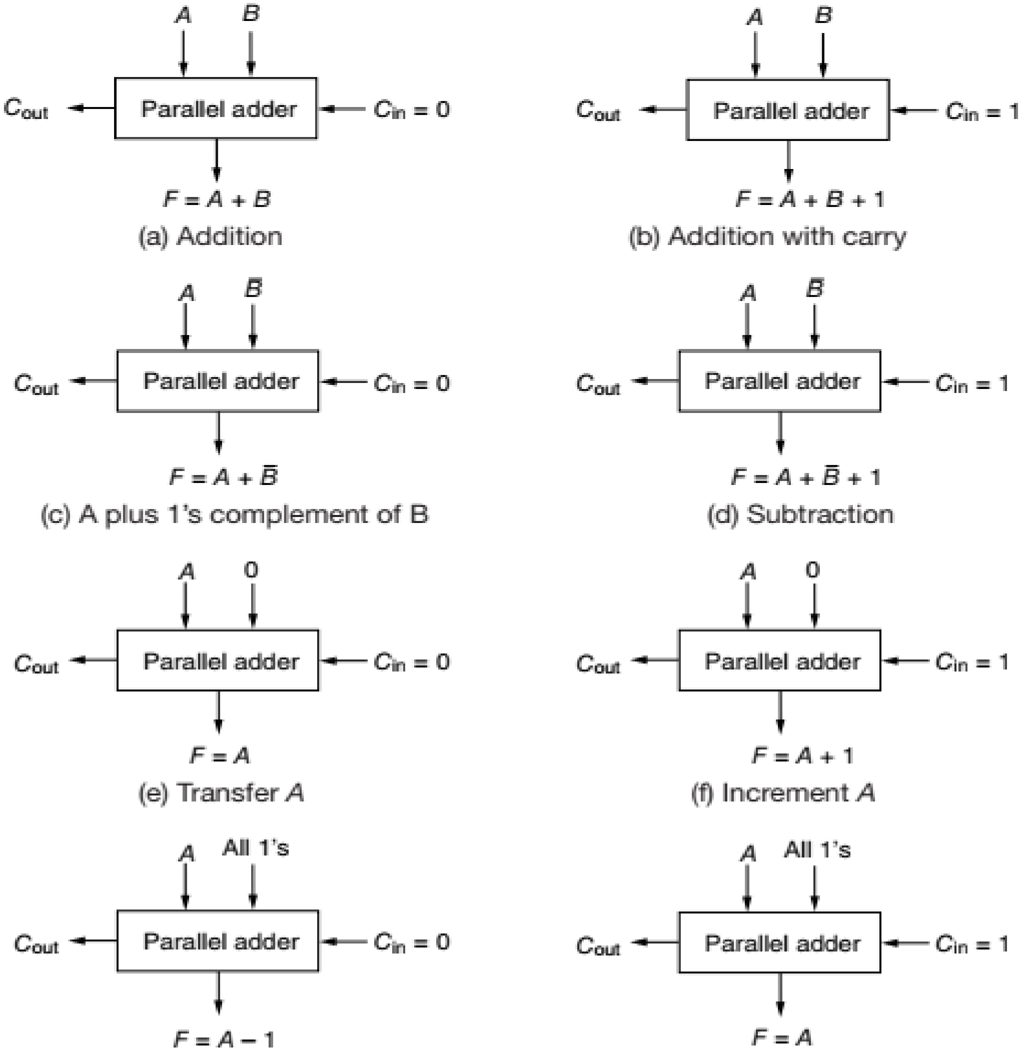
Design of Arithmetic Logic Unit

The steps involved in the design of an ALU are as follows:

1. Design the arithmetic section independent of the logic section.
2. Determine the logic operations obtained from the arithmetic circuit in step 1, assuming that the input carries to all stages are 0.
3. Modify the arithmetic circuit to obtain the required logic operations.



Design of Arithmetic Logic Unit



(a) Addition

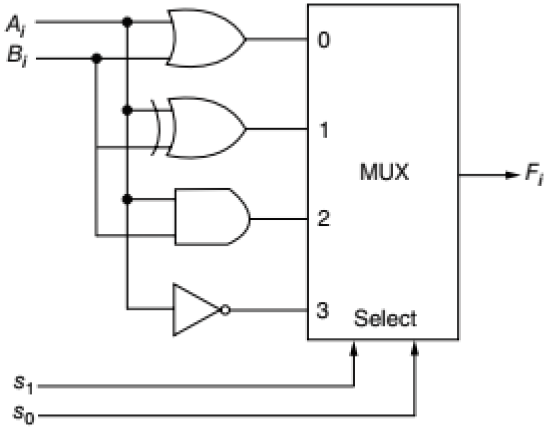
(b) Addition with carry

(c) A plus 1's complement of B

(d) Subtraction

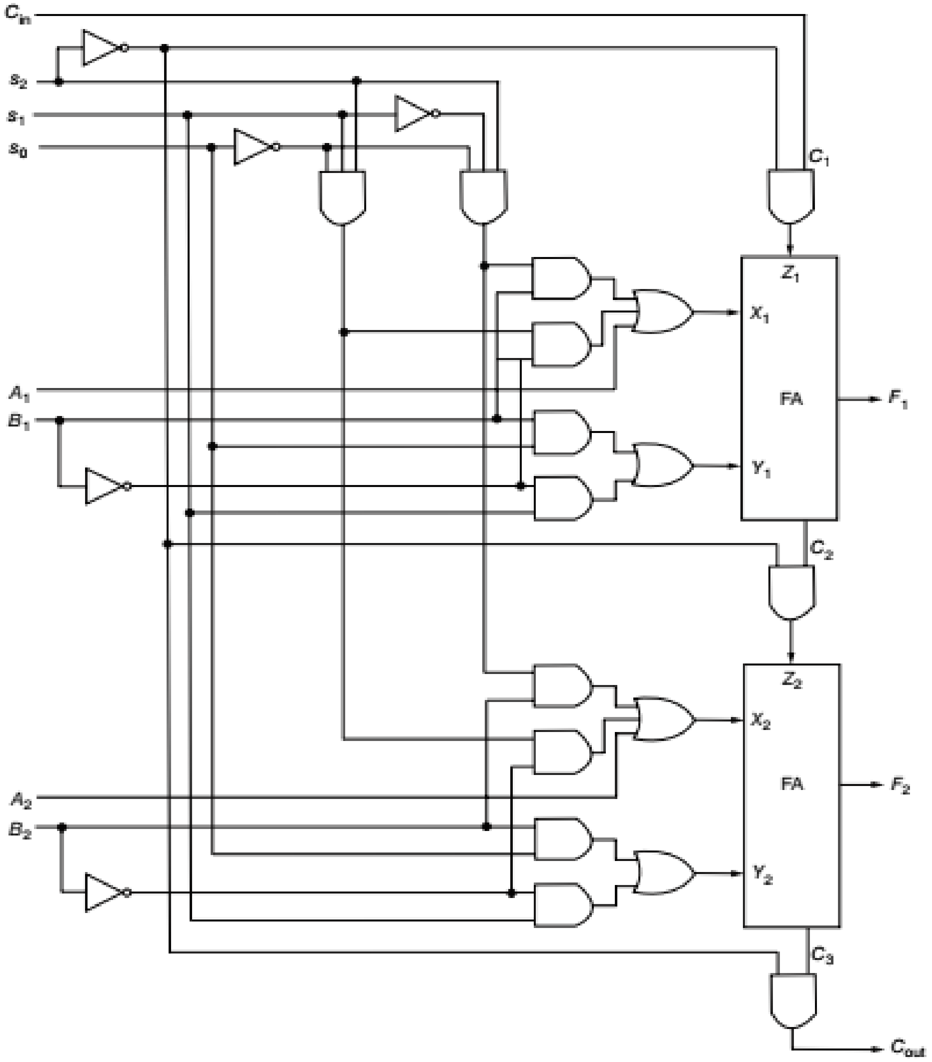
(e) Transfer *A*

(f) Increment *A*



s_1	s_0	Output	Operation
0	0	$F_i = A_i + B_i$	OR
0	1	$F_i = A_i \oplus B_i$	XOR
1	0	$F_i = A_i B_i$	AND
1	1	$F_i = A'_i$	NOT

Design of Arithmetic Logic Unit



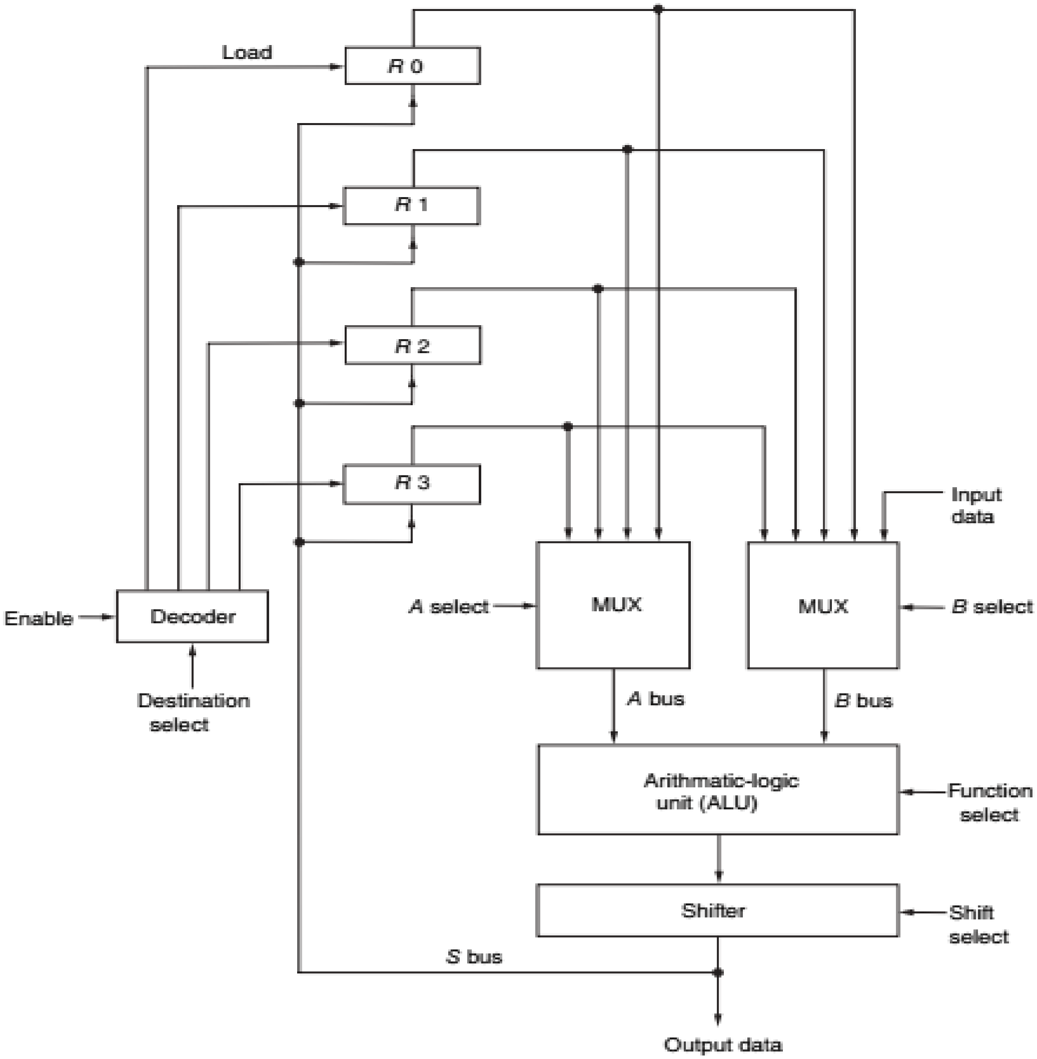
Selection				Output	Function
s_2	s_1	s_0	C_{in}		
0	0	0	0	$F = A$	Transfer A
0	0	0	1	$F = A + 1$	Increment A
0	0	1	0	$F = A + B$	Addition
0	0	1	1	$F = A + B + 1$	Add with carry
0	1	0	0	$F = A - B - 1$	Subtract with borrow
0	1	0	1	$F = A - B$	Subtraction
0	1	1	0	$F = A - 1$	Decrement A
0	1	1	1	$F = A$	Transfer A
1	0	0	X	$F = A \vee B$	OR
1	0	1	X	$F = A \oplus B$	XOR
1	1	0	X	$F = A \wedge B$	AND
1	1	1	X	$F = \bar{A}$	Complement A

$$X_i = A_i + s_2 s'_1 s'_0 B_i + s_2 s_1 s'_0 B'_i$$
$$Y_i = s_0 B_i + s_1 B'_i$$
$$Z_i = s_2 C'_i$$

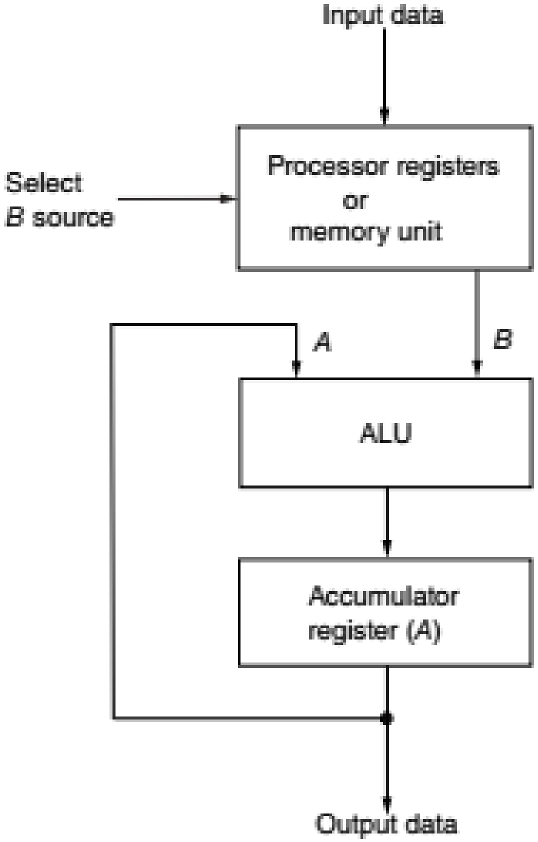
$$X_i = A_i$$
$$Y_i = s_0 B_i + s_1 B'_i$$
$$Z_i = C'_i$$

$$X_i = A_i$$
$$Y_i = s_0 B_i + s_1 B'_i$$
$$C_i = 0$$

Processor Organization



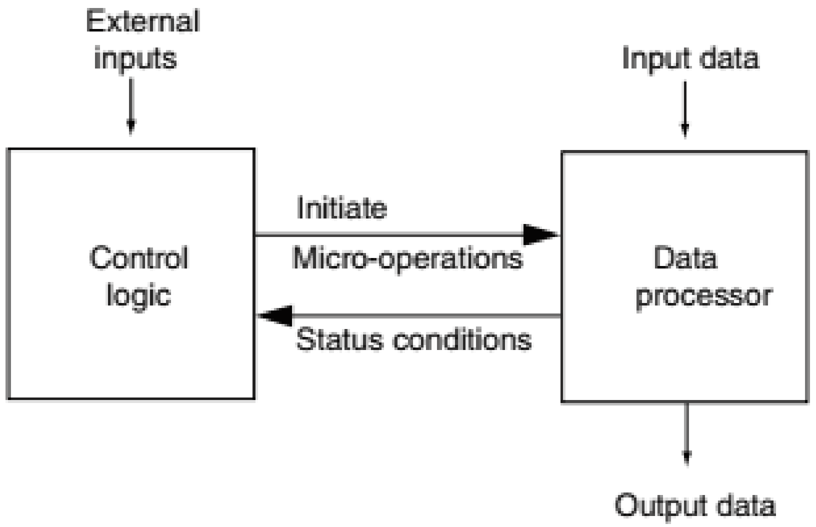
Accumulator Register



T_1:	$A \leftarrow 0$	clear A
T_2:	$A \leftarrow A + R1$	transfer $R1$ to A
T_3:	$A \leftarrow A + R2$	add $R2$ to A

Control variable	Microoperation	Name
p_1	$A \leftarrow A + B$	Add
p_2	$A \leftarrow 0$	Clear
p_3	$A \leftarrow \overline{A}$	Complement
p_4	$A \leftarrow A \wedge B$	AND
p_5	$A \leftarrow A \vee B$	OR
p_6	$A \leftarrow A \oplus B$	Exclusive-OR
p_7	$A \leftarrow \text{shr } A$	Shift-right
p_8	$A \leftarrow \text{shl } A$	Shift-left
p_9	$A \leftarrow A + 1$	Increment
	If $(A = 0)$ then $(Z = 1)$	Check for zero

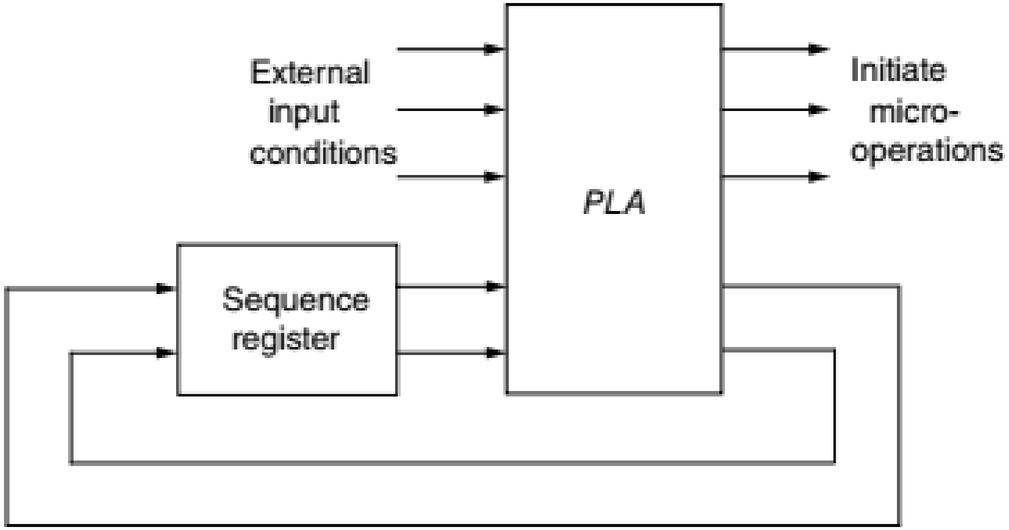
Control Logic Design



- PLA Control
- Micro Program Control
- Hardwired control
- Control of Processor Unit

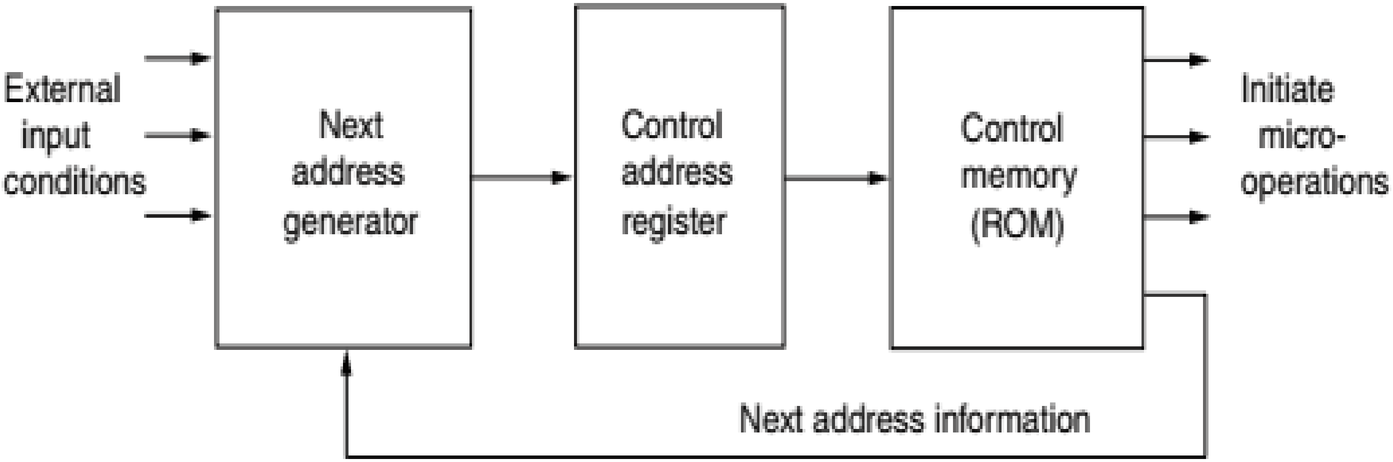
Control Logic Design

- PLA Control



Control Logic Design

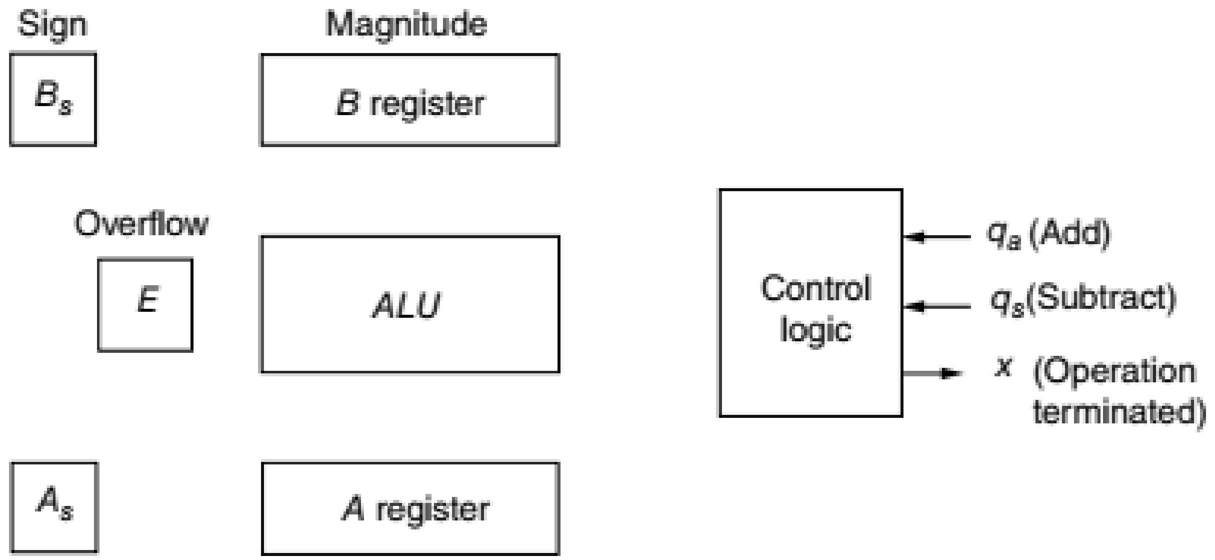
- Micro Program Control



Control Logic Design

- Hard Wired Control

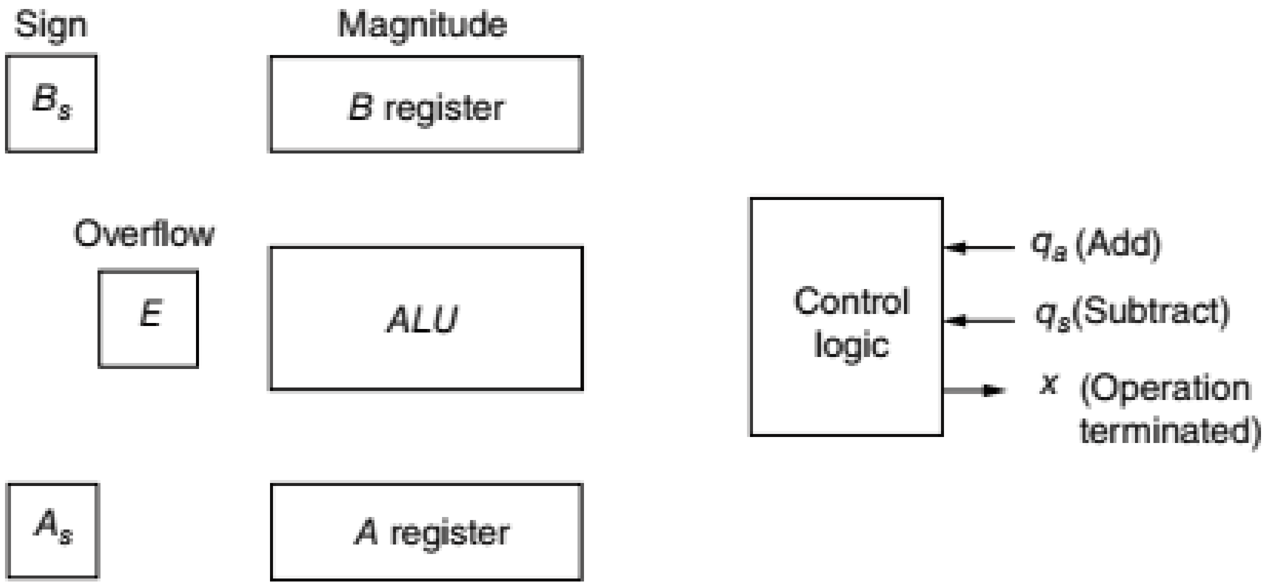
1. The problem is stated.
2. An initial equipment configuration is assumed.
3. An algorithm is formulated.
4. The data-processor part is specified.
5. The control logic is designed.



Control Logic Design

1. Problem Statement

2. Equipment Configuration



Control Logic Design

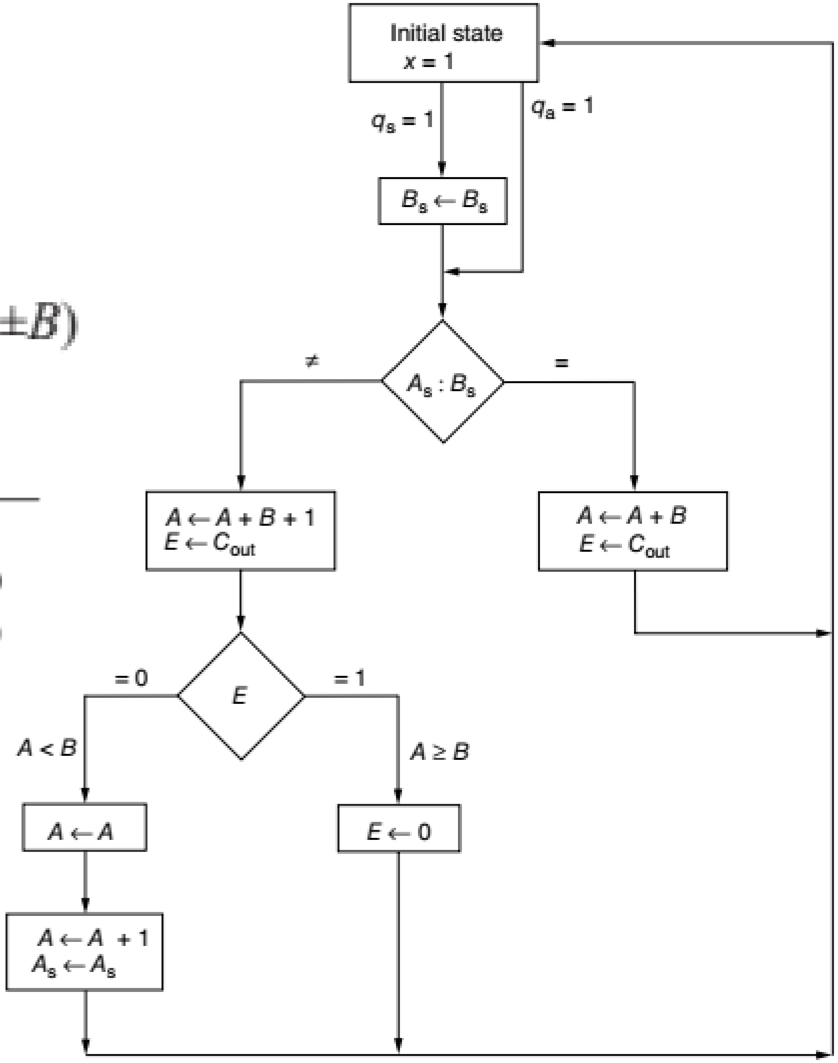
4. Algorithm Formulation

$$(\pm A) \pm (\pm B)$$

$$(\pm A) - (+B) = (\pm A) + (-B)$$
$$(\pm A) - (-B) = (\pm A) + (+B)$$

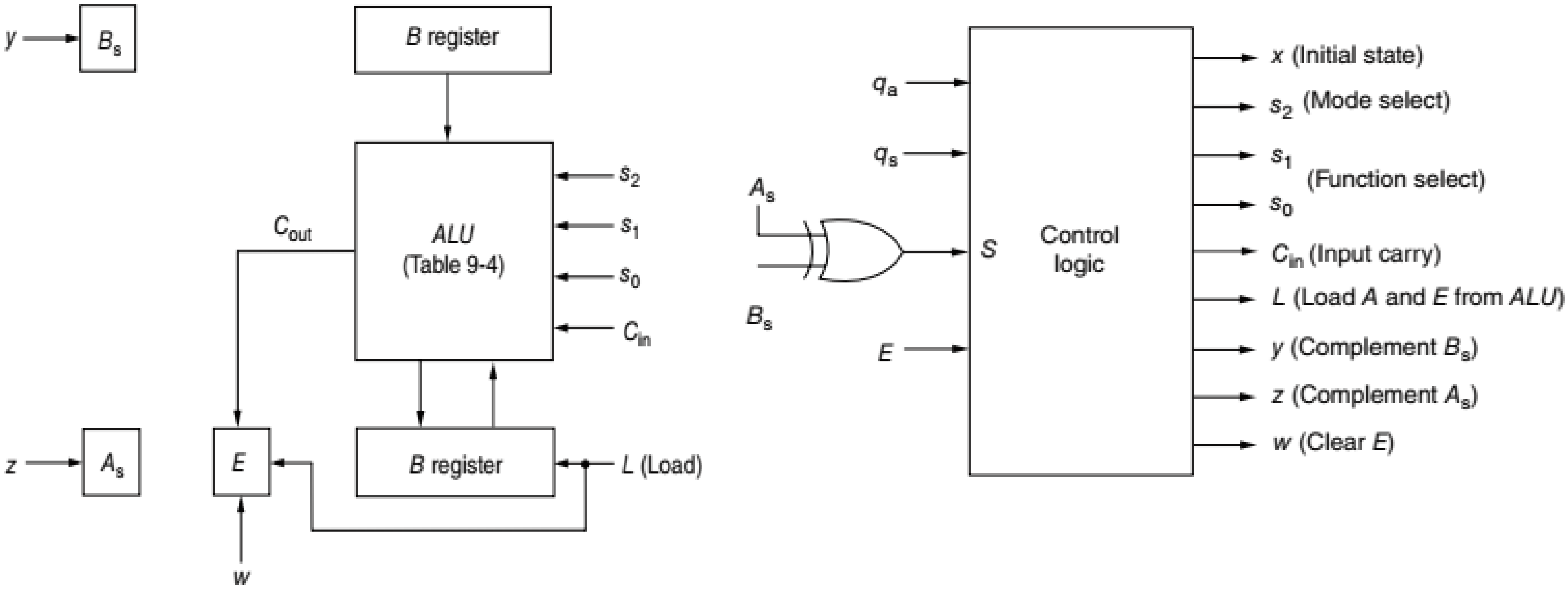
$$(\pm A) + (\pm B)$$

	If $A \geqslant B$	If $A < B$
$(+A) + (+B) = +(A + B)$		
$(+A) + (-B) =$	$+(A - B) =$	$-(B - A)$
$(-A) + (+B) =$	$-(A - B) =$	$+(B - A)$
$(-A) + (-B) = -(A + B)$		



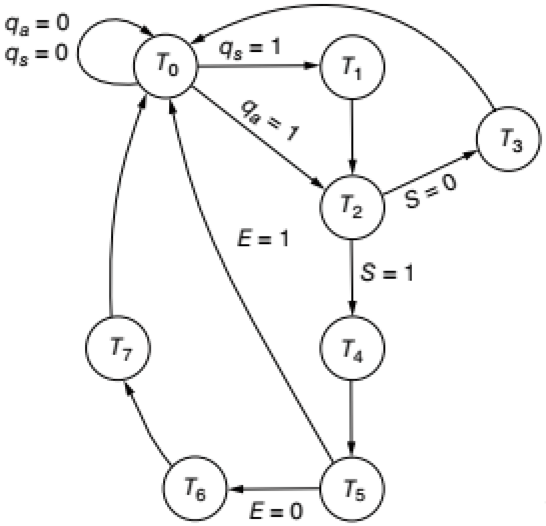
Control Logic Design

4. Data Processor Specification



Control Logic Design

4. Control State Diagram

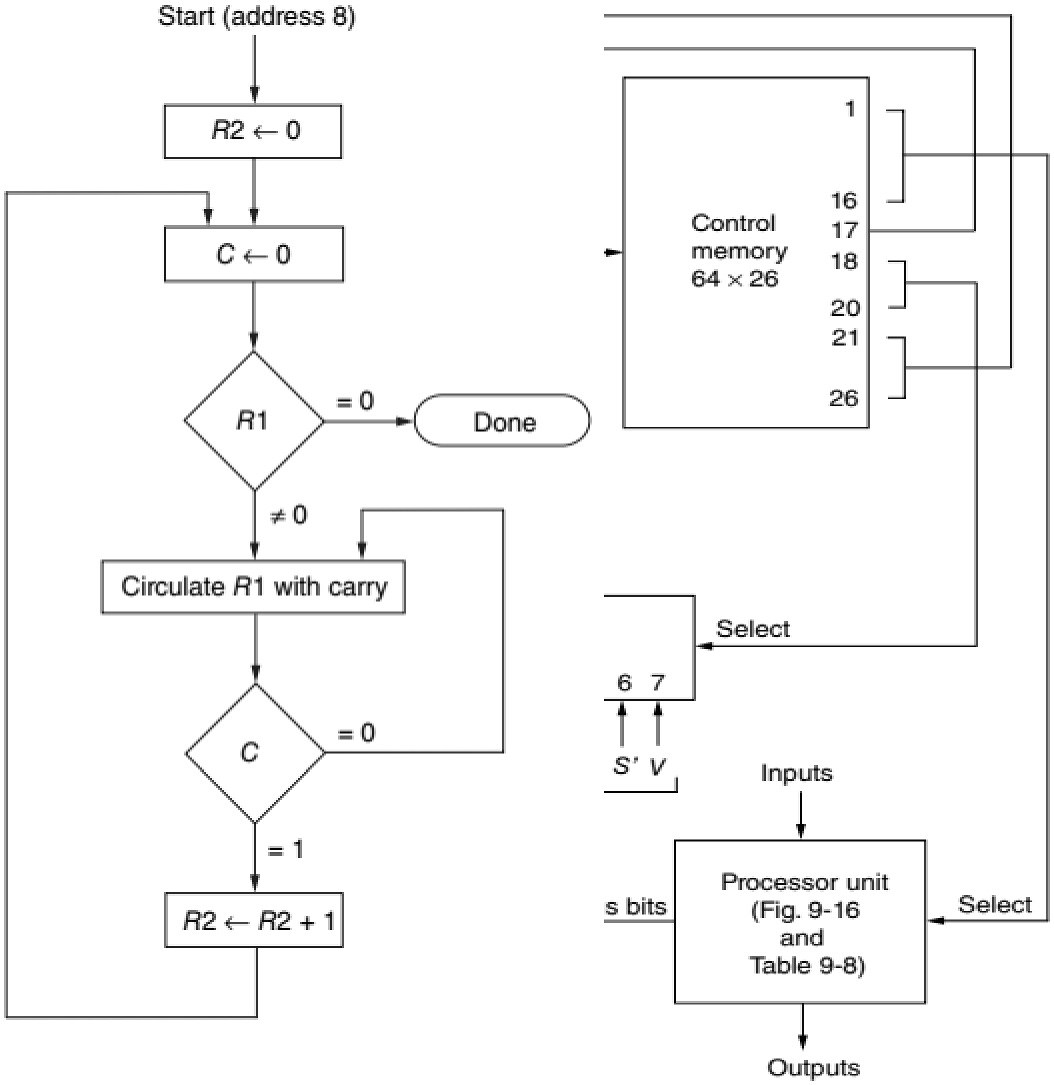


q_a	Add
q_s	Subtract
$S = 0$	Signs alike
$S = 1$	Signs unlike
E	Output carry

	Control outputs								
	x	s_2	s_1	s_0	C_{in}	L	y	z	w
T_0: Initial state $x = 1$	1	0	0	0	0	0	0	0	0
T_1: $B_s \leftarrow \overline{B}_s$	0	0	0	0	0	0	1	0	0
T_2: nothing	0	0	0	0	0	0	0	0	0
T_3: $A \leftarrow A + B$, $E \leftarrow C_{out}$	0	0	0	1	0	1	0	0	0
T_4: $A \leftarrow A + \overline{B} + 1$, $E \leftarrow C_{out}$	0	0	1	0	1	1	0	0	0
T_5: $E \leftarrow 0$	0	0	0	0	0	0	0	0	1
T_6: $A \leftarrow \overline{A}$	0	1	1	1	0	1	0	0	0
T_7: $A \leftarrow A + 1$, $A_s \leftarrow \overline{A}_s$	0	0	0	0	1	1	0	1	0

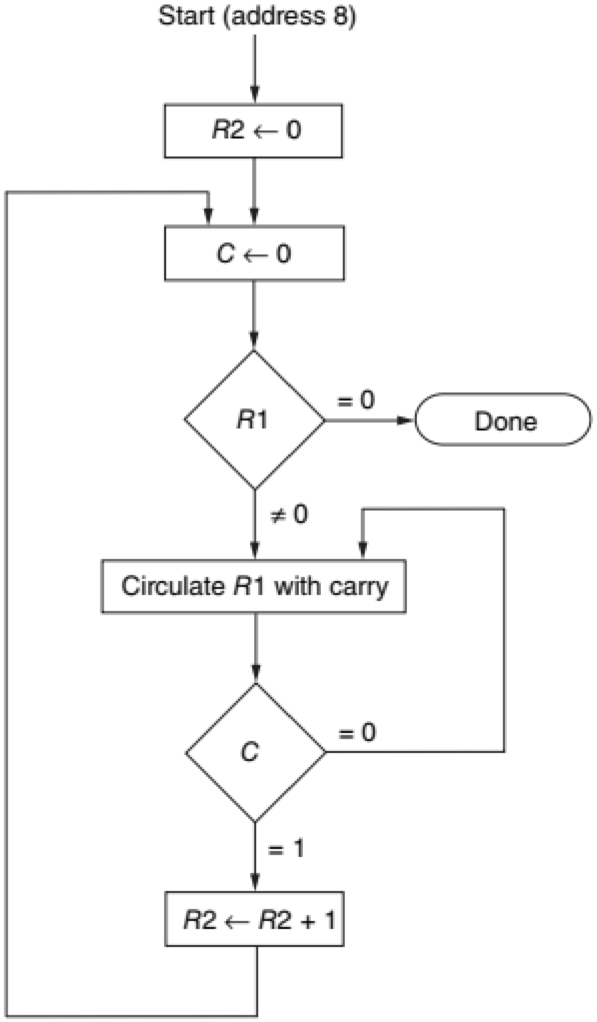
Control Logic Design

Control of Processor Unit



Control Logic Design

Control of Processor Unit



ROM address	Microinstruction	Comments
8	$R2 \leftarrow 0$	Clear $R2$ counter
9	$R1 \leftarrow R1$, $C \leftarrow 0$	Clear C, set status bits
10	If ($Z = 1$) then (go to external address)	Done if $R1 = 0$
11	$R1 \leftarrow$ crc $R1$	Circulate $R1$ right with carry
12	If ($C = 0$) then (go to 11)	Circulate again if $C = 0$
13	$R2 \leftarrow R2 + 1$, go to 9	Carry = 1, increment $R2$