AMERICAN INTERNAIONAL UNIVERSITY – BANGLADESH

Where leaders are created

Lecture-1 (Final)
Processor Logic Design

Department of EEE

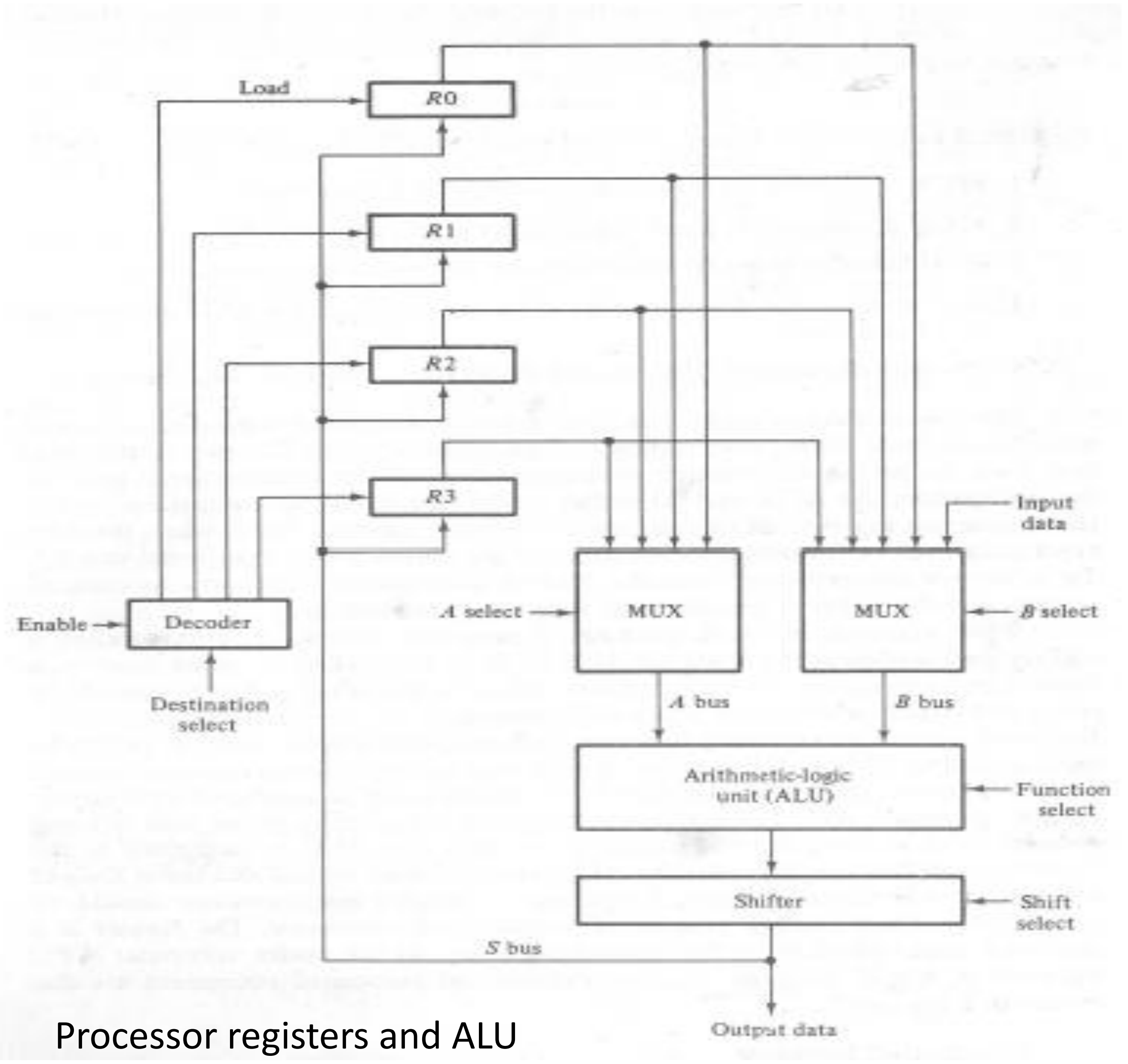
Introduction

- A processor unit is that part of a digital system that implements the operations in the system.
- It is comprised of a number of registers and the digital functions that implement arithmetic, logic, shift and transfer micro-operations.
- Central processing unit (CPU) consists of MEMORY (Register banks), ARITHMETIC LOGIC UNIT (ALU) and CONTROL UNIT (CU).

Bus Organization

- To perform the micro-operation:
  - R1 = R2 + R3
- The control unit must provide binary selection variables to the following selector inputs –

1. MUX A Selector: to place contents of R2 onto bus A
2. MUX B Selector: to place contents of R3 onto bus B
3. ALU Function Selector: to provide the arithmetic operation A + B
4. Shift Selector: for direct transfer from the output of the ALU onto output bus S (no shift)
5. Decoder Destination Selector: to transfer the contents of bus S into R1



Processor registers and ALU connected through Common Bus

AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Bus Organization

- The five control selection variables must be generated simultaneously and must be available during one common clock pulse interval.
- The binary information from the two source registers propagates through the combinational gates in the multiplexers, the ALU, and the shifter, to the output bus and into the inputs of the destination register all during one clock pulse interval.
- Then, when the next clock pulse arrives, the binary information on the output bus is transferred to R1.

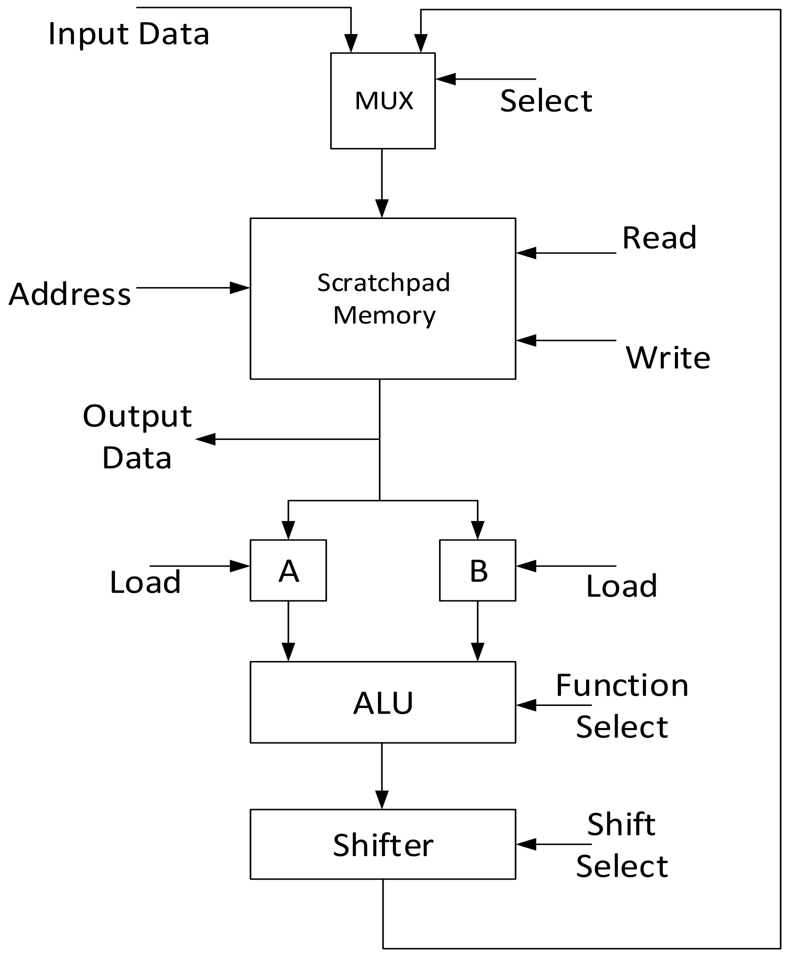
SCRATCHPAD MEMORY

- The registers in a processor unit can be enclosed within a small memory unit. When included in a processor unit, a small memory is sometimes called a scratchpad memory.
- The use of a small memory is a cheaper alternative to connecting processor registers through a bus system. The difference between the two systems is the manner in which information is selected for transfer into the ALU.
- In the bus system, the information transfer is selected by the multiplexers that form the buses.
- On the other hand, a single register in a group of registers organized as a small memory must be selected by means of an address to the memory unit.

SCRATCHPAD MEMORY

- To perform the operation R1=R2+R3, the control must provide binary selection variables to perform the following sequence of three microoperations:
    - T1: A ← M[010] Read R2 into register A
    - T2: B ← M[011] Read R3 into register B
    - T3: M [001] ← A+B Perform operation in ALU and transfer result to R1

- The reason for a sequence of three microoperations, instead of just one as in a bus-organized processor, is due to the limitation of the memory unit. Since the memory unit has only one set of address terminals but two source registers are to be accessed.

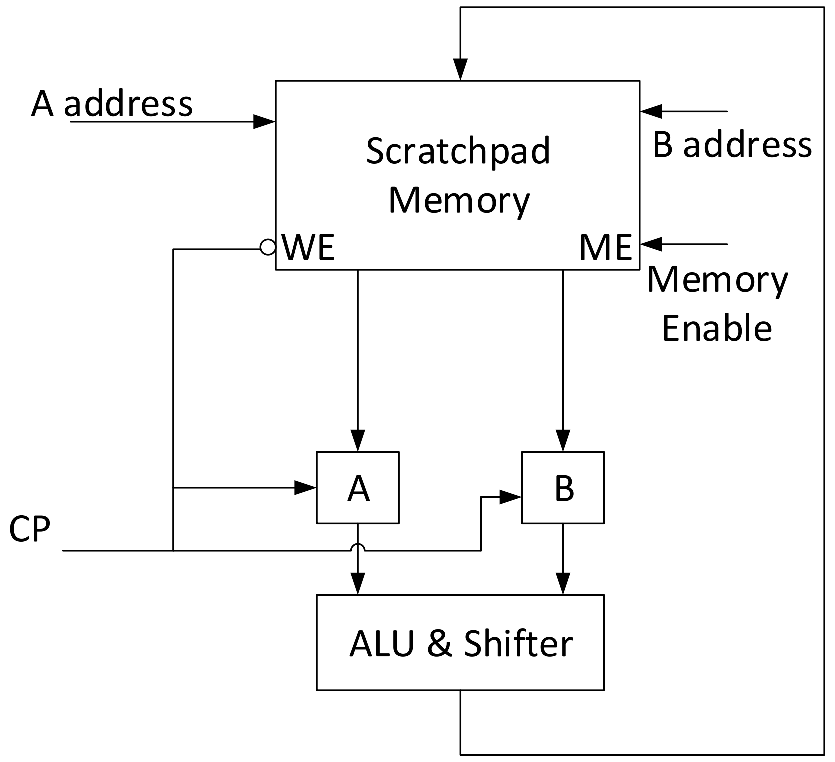


Processor Unit Employing a Scratchpad Memory

AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Processor Unit with two port memory

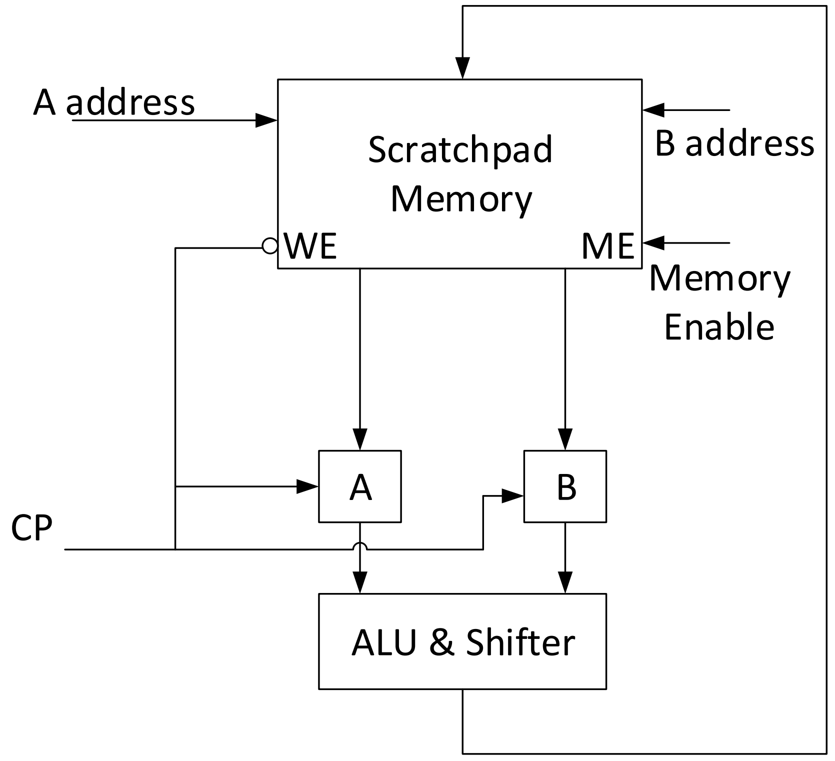
- Some processors employ a 2-port memory in order to overcome the delay caused when reading two source registers.
- If the destination register is the same as one of the source register, then the entire microoperation can be done within one clock pulse period.



Processor unit with 2-port memory

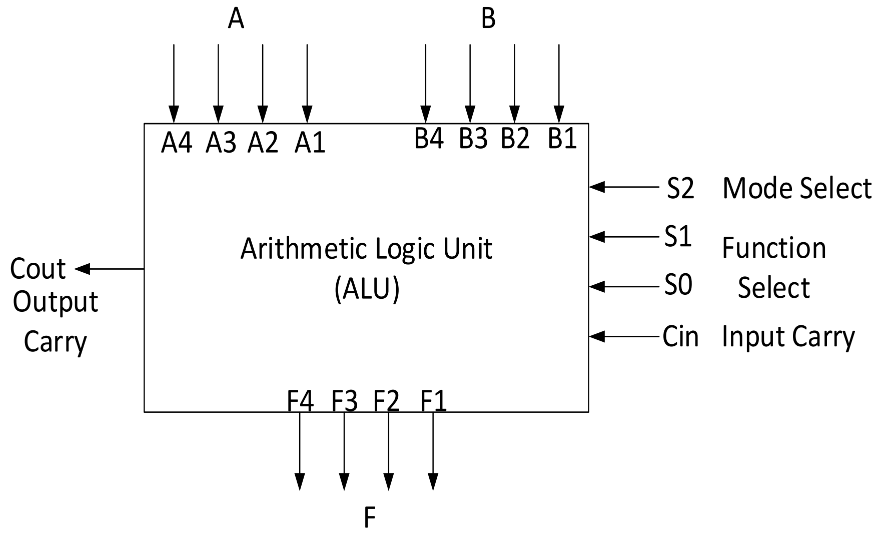
Processor Unit with two port memory

- When Clock = 1, A and B latches are open and accept information coming from memory. The WE' is also in 1 state and disables write operation and enables read operation. Thus when Clock =1, words selected by A and B addresses are read from memory and placed in A and B registers.
- When Clock = 0, the latches are closed and retains the last data entered. ALU operation is performed and if write is enabled since Clock = 0, and also ME input is enabled, the result of the micro-operation is written into memory word defined by B address.



ARITHMETIC LOGIC UNIT (ALU)

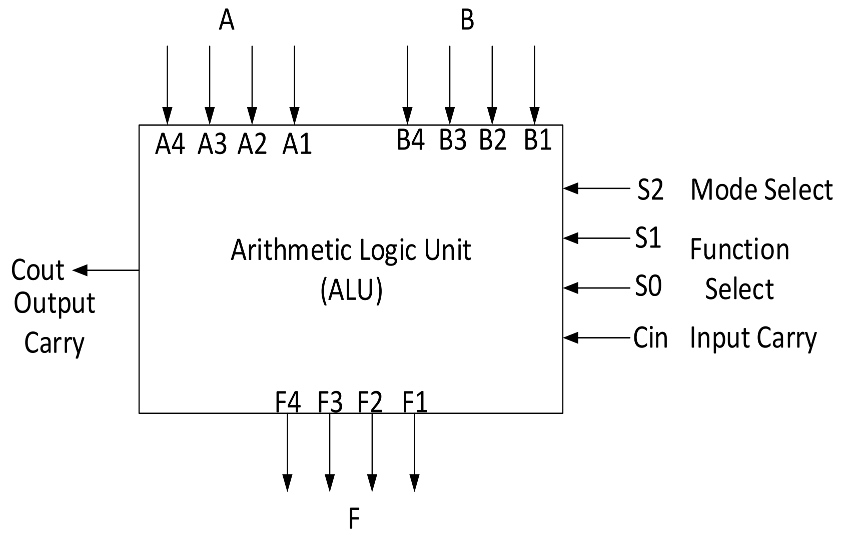
- An ARITHMETIC LOGIC UNIT (ALU) is a multi-operation, combinational logic digital function.
- It can perform a set of basic arithmetic operations and a set of logic operations.
- The ALU has a number of selection lines to select a particular operation in the unit.
- The selection lines are decoded within the ALU so that k selection variables can specify upto 2^K distinct operations.



AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

ARITHMETIC LOGIC UNIT (ALU)

1. The mode select **s2** distinguishes between arithmetic and logic operations.
2. The two function select inputs **s1 and s0** specify the particular arithmetic and logic operation to be generated.
3. The input and output carry (Cin and Cout) have meanings only during arithmetic operations.
4. The input carry **Cin** in the least significant position of an ALU is quite often used as fourth selection variable that can double the number of arithmetic operations. Thus a total of 8 arithmetic operations and 4 logical operations can be performed.

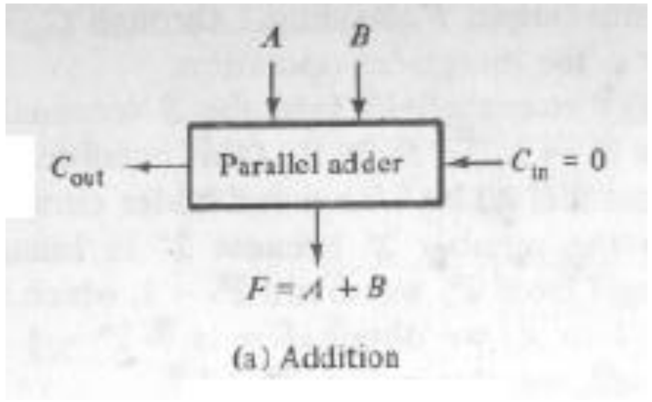


Design of Arithmetic Circuit

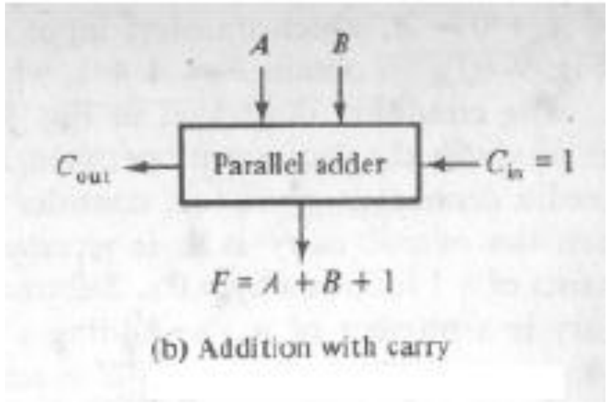
- The basic component of the arithmetic section of an ALU is the parallel adder.
- A parallel adder is constructed with a number of full adder circuits cascaded.
- By controlling the data inputs to the parallel adder, it is possible to obtain different types of arithmetic operations.

Operations obtained by controlling one set of inputs to a parallel adder

S_1S_0 = 01

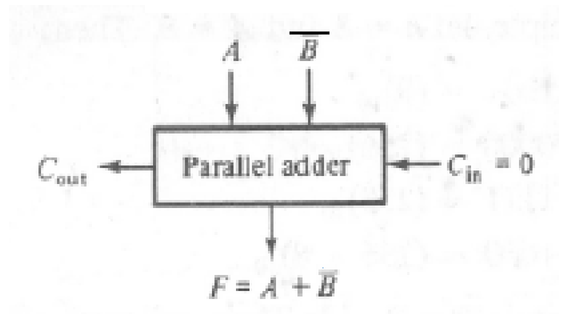


(a) Addition

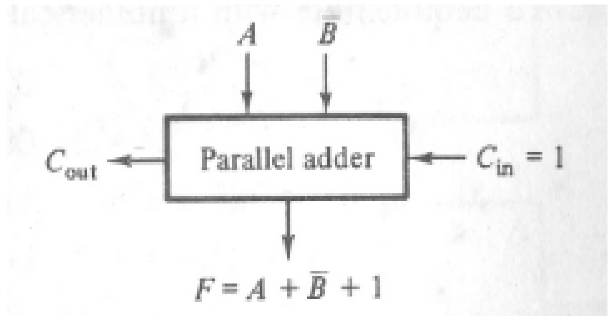


(b) Addition with carry

S_1S_0 = 10



(c) A plus 1's complement of B

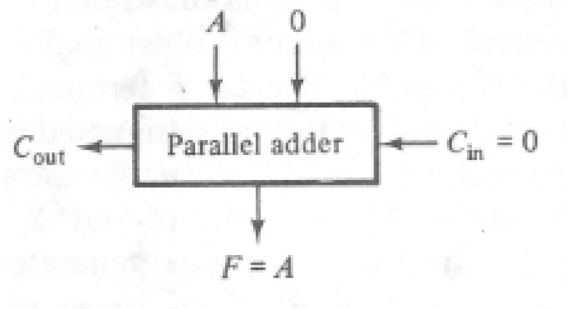


(d) Subtraction

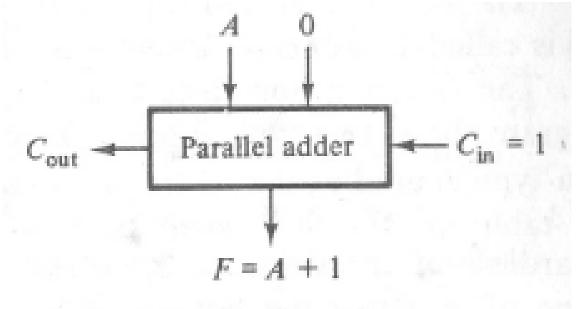
Fig. 9.6

Operations obtained by controlling one set of inputs to a parallel adder

S_1S_0 = 00

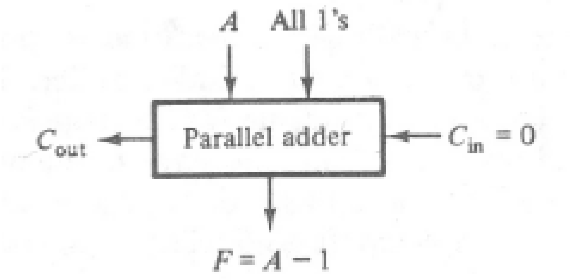


(e) Transfer *A*

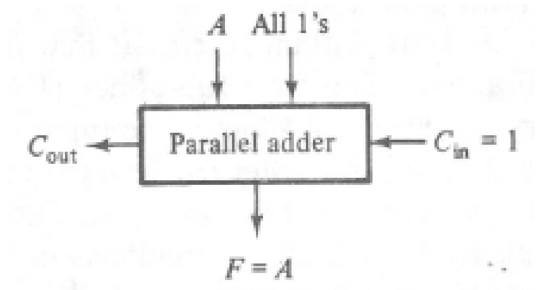


(f) Increment *A*

S_1S_0 = 11



(g) Decrement *A*



(h) Transfer *A*

AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

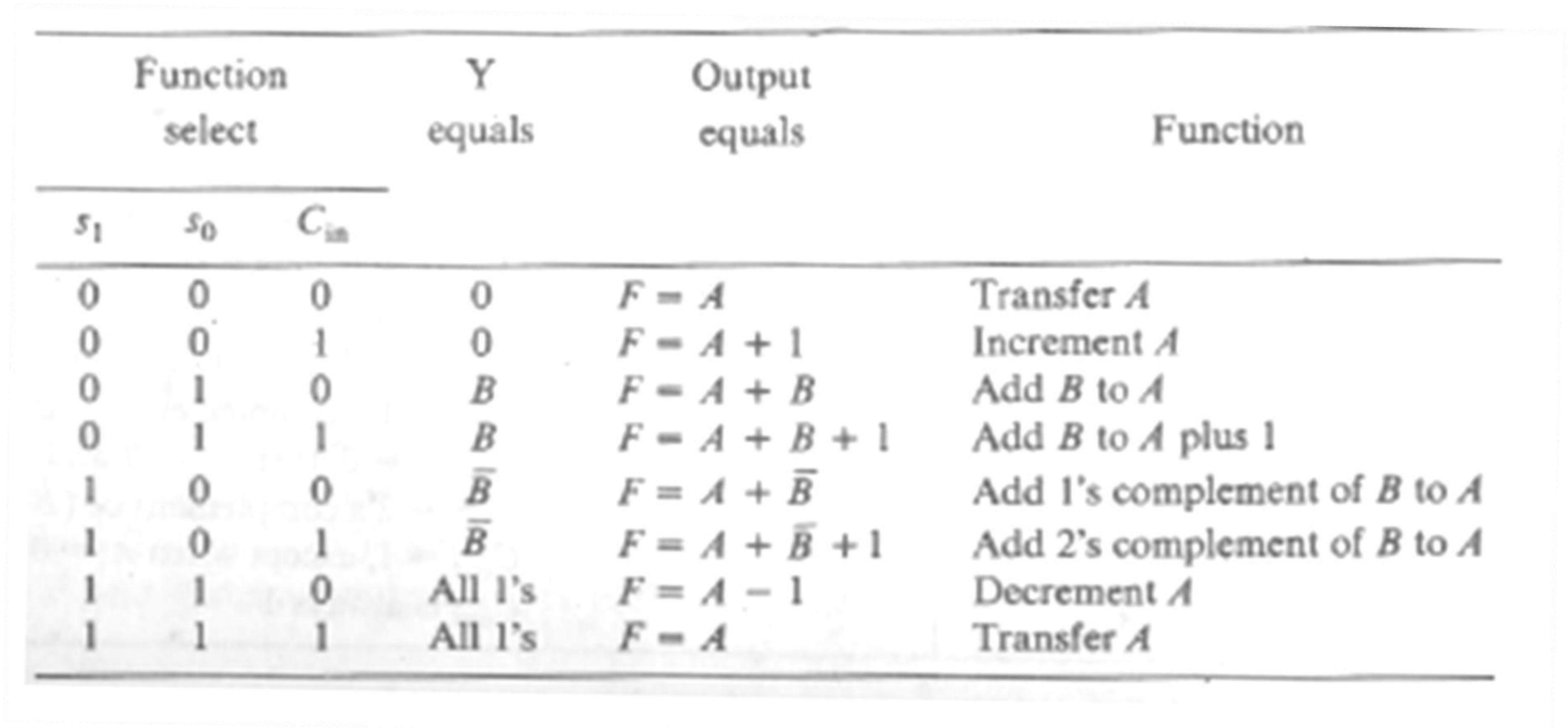
The condition illustrated in Fig. 9-6(g) inserts all 1's into the B terminals. This produces the decrement operation $F = A - 1$. To show that this condition is indeed a decrement operation, consider a parallel adder with n full-adder circuits. When the output carry is 1, it represents the number 2^n because 2^n in binary consists of a 1 followed by n 0's. Subtracting 1 from 2^n, we obtain $2^n - 1$, which in binary is a number of n 1's. Adding $2^n - 1$ to A, we obtain $F = A + 2^n - 1 = 2^n + A - 1$. If the output carry 2^n is removed, we obtain $F = A - 1$.

To demonstrate with a numerical example, let $n = 8$ and $A = 9$. Then:

$$\begin{array}{rlll} A = & \ 0000 & 1001 & = (9)_{10} \\ 2^n = & 1\ 0000 & 0000 & = (256)_{10} \\ 2^n - 1 = & \ 1111 & 1111 & = (255)_{10} \\ A + 2^n - 1 = & 1\ 0000 & 1000 & = (256 + 8)_{10} \end{array}$$

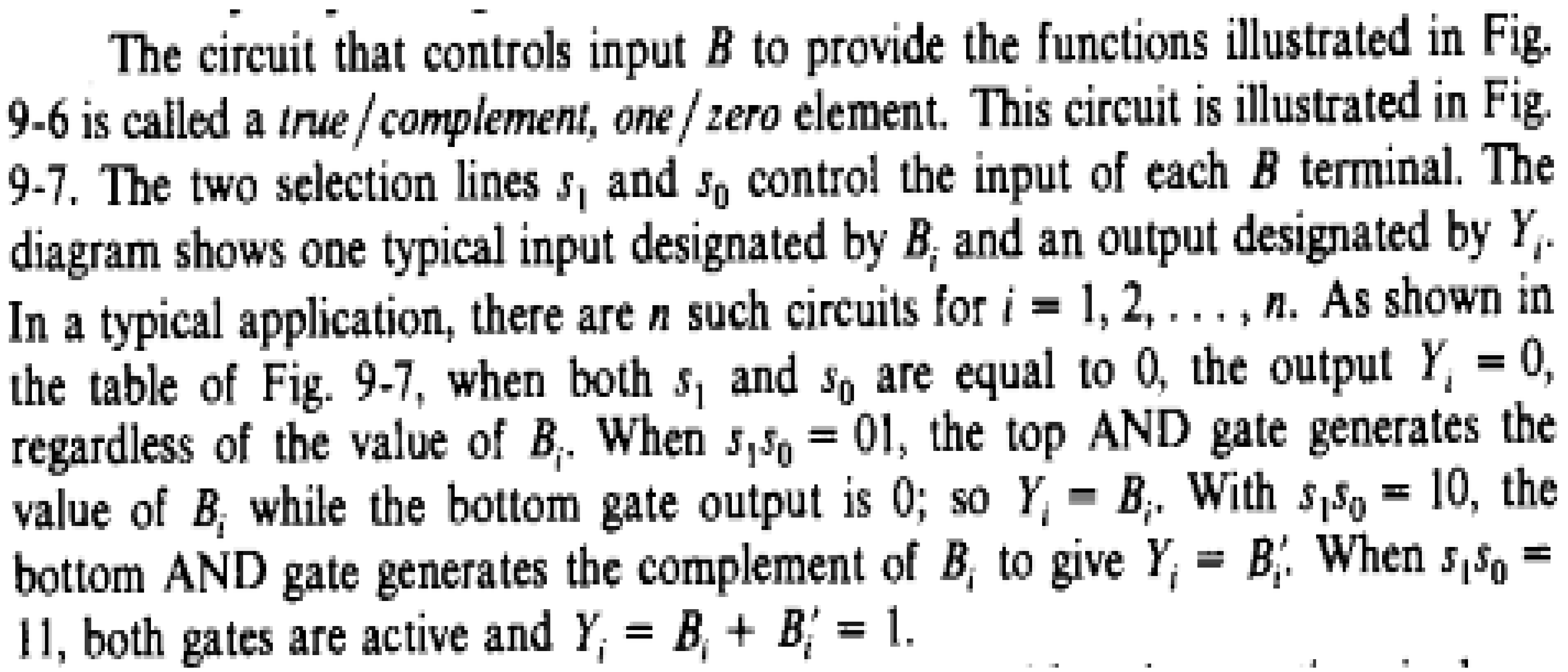
Removing the output carry $2^n = 256$, we obtain $8 = 9 - 1$. Thus, we have decremented A by 1 by adding to it a binary number with all 1's.

Function table for arithmetic Circuit



Function select			Y equals	Output equals	Function
s_1	s_0	C_{in}			
0	0	0	0	$F = A$	Transfer A
0	0	1	0	$F = A + 1$	Increment A
0	1	0	B	$F = A + B$	Add B to A
0	1	1	B	$F = A + B + 1$	Add B to A plus 1
1	0	0	$\bar{B}$	$F = A + \bar{B}$	Add 1's complement of B to A
1	0	1	$\bar{B}$	$F = A + \bar{B} + 1$	Add 2's complement of B to A
1	1	0	All 1's	$F = A - 1$	Decrement A
1	1	1	All 1's	$F = A$	Transfer A

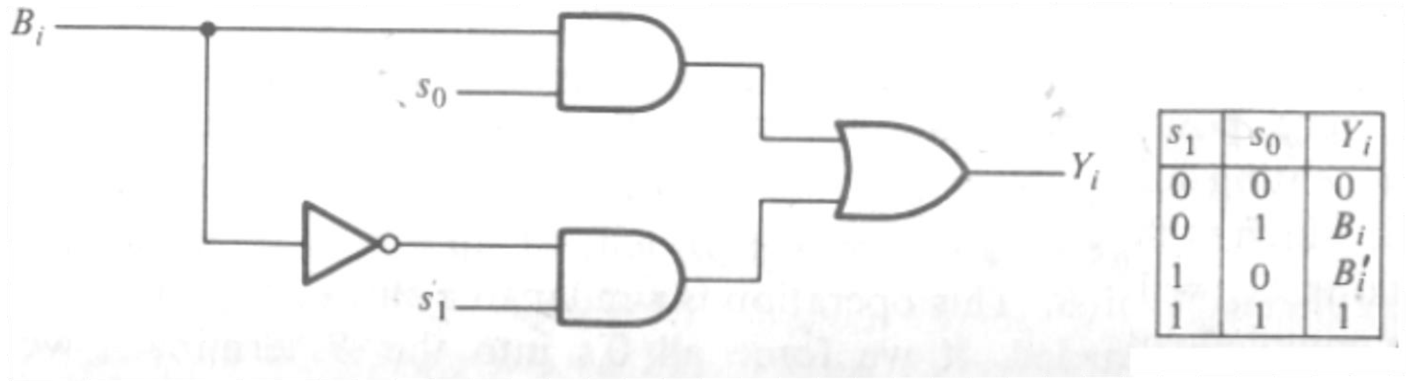
AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH



The circuit that controls input B to provide the functions illustrated in Fig. 9-6 is called a *true/complement, one/zero* element. This circuit is illustrated in Fig. 9-7. The two selection lines s_1 and s_0 control the input of each B terminal. The diagram shows one typical input designated by B_i and an output designated by Y_i. In a typical application, there are n such circuits for $i = 1, 2, \ldots, n$. As shown in the table of Fig. 9-7, when both s_1 and s_0 are equal to 0, the output $Y_i = 0$, regardless of the value of B_i. When $s_1s_0 = 01$, the top AND gate generates the value of B_i while the bottom gate output is 0; so $Y_i = B_i$. With $s_1s_0 = 10$, the bottom AND gate generates the complement of B_i to give $Y_i = B_i'$. When $s_1s_0 = 11$, both gates are active and $Y_i = B_i + B_i' = 1$.

True/Complement, one/zero Circuit

- The values of the Y inputs to the full-adder circuits are a function of selection variables S_1 and S_0 .



s_1	s_0	Y_i
0	0	0
0	1	B_i
1	0	B_i'
1	1	1

Fig. 9.7

- The arithmetic circuit needs a combinational circuit in each stage specified by Boolean functions –

$$X_i = A_i$$

$$Y_i = B_i S_0 + B_i' S_1, \ i = 1,2,3,\ldots,n$$

AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Logic Diagram of Arithmetic Circuit

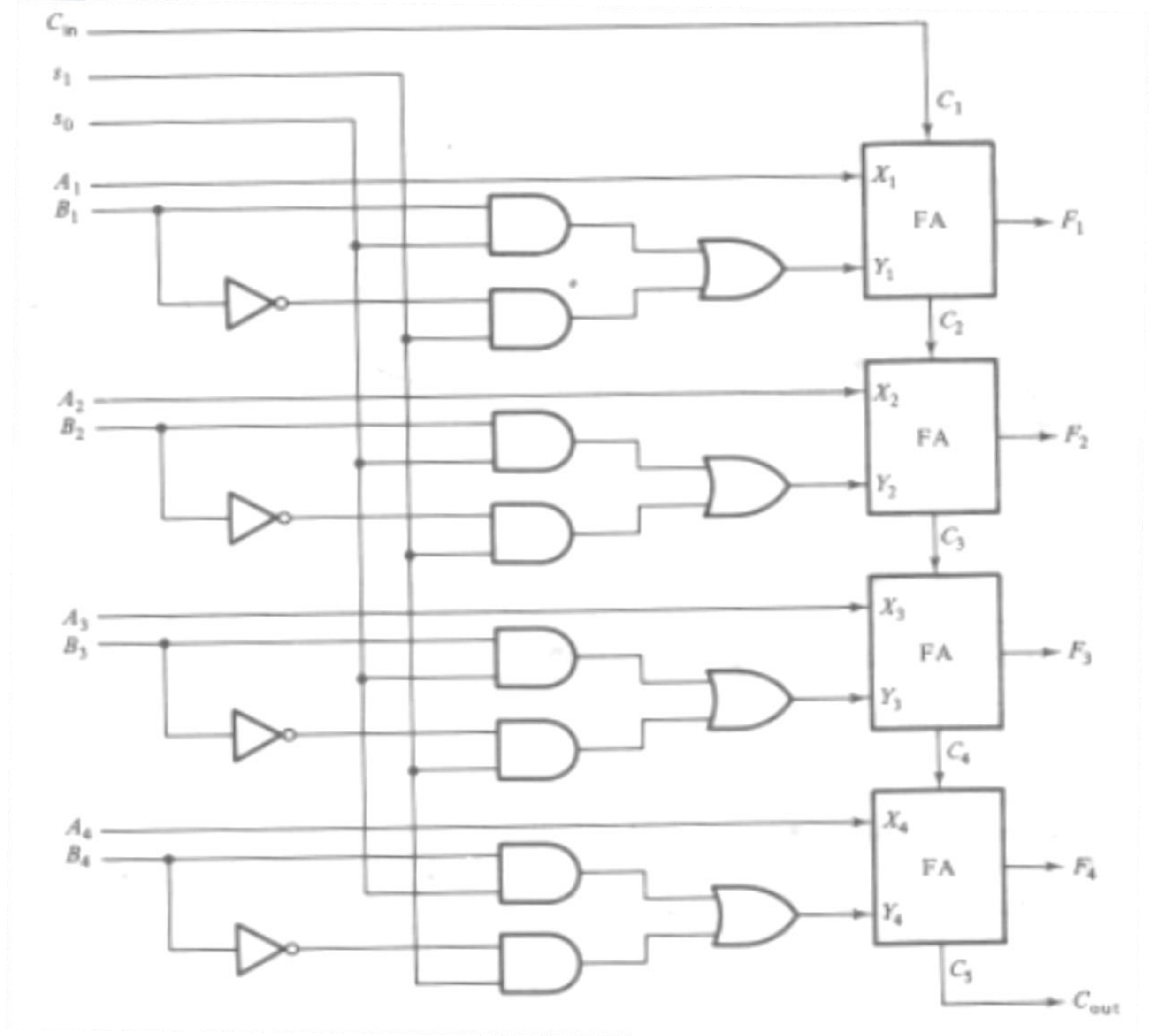


Figure 9-8

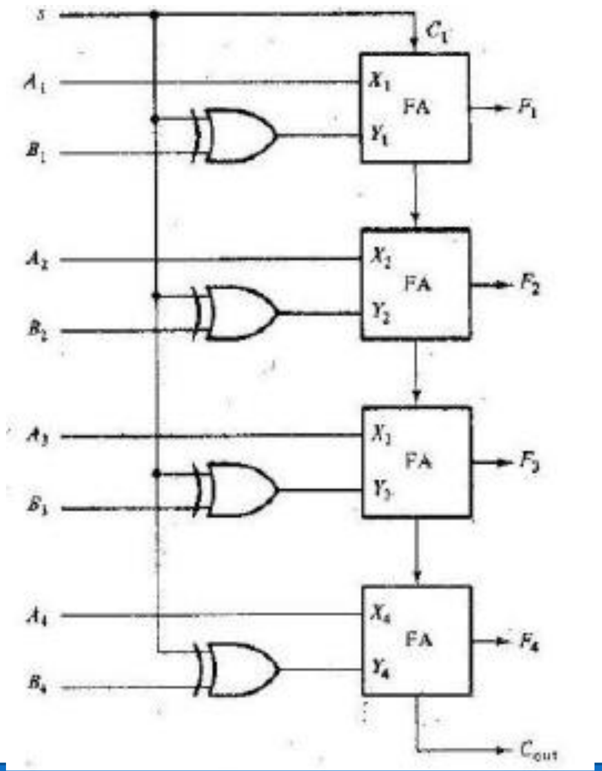
AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Exercise (Home Work)

- Design an adder/subtractor circuit with one selection variable s and two inputs A and B: when s= 0 the circuit performs A+B. When S=1 the circuit performs A-B by taking the 2’s complement of B.

$X_i = A_i$
$Y_i = B_i$ XOR S
Cin = S

S	F	X	Y	C_{in}
0	A+B	A	B	0
1	A-B	A	B’	1



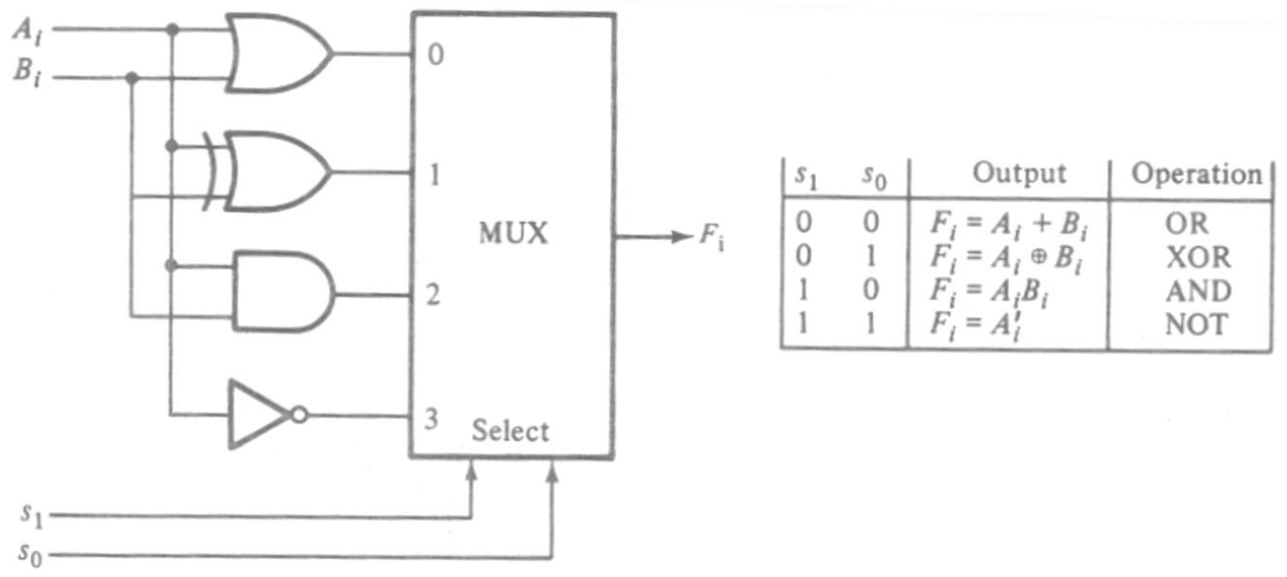
INPUT		OUTPUT
A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

Truth Table

AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Design of Logic Circuit

- The logic microoperations manipulate the bits of the operands separately and treat each bit as a binary variable.
- All logic operations can be obtained by means of **AND, OR and NOT** operations.
- For 3 operations, we need 2 selection variables but 2 selection lines can choose between 4 logic operations and hence XOR operation was also incorporated in the logical circuit.



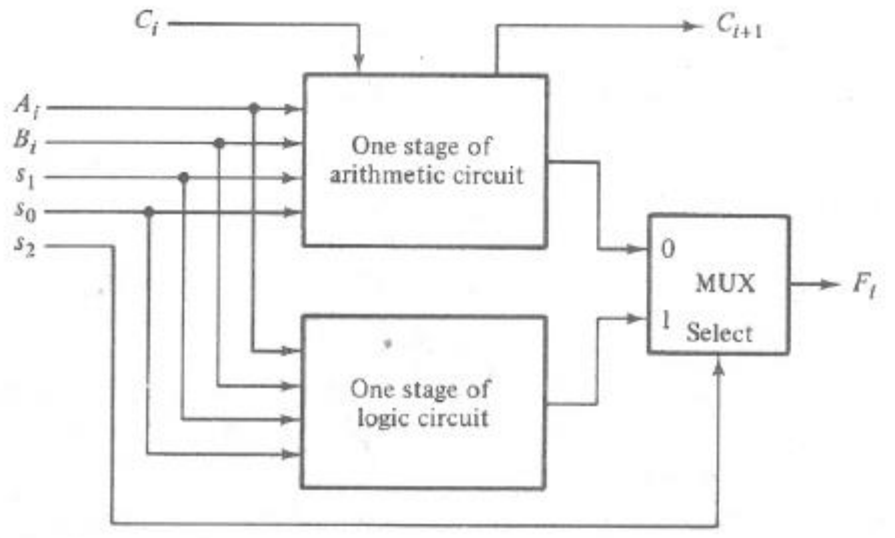
s_1	s_0	Output	Operation
0	0	$F_i = A_i + B_i$	OR
0	1	$F_i = A_i \oplus B_i$	XOR
1	0	$F_i = A_i B_i$	AND
1	1	$F_i = A_i'$	NOT

(a) Logic diagram (b) Function table

Figure 9-11 One stage of logic circuit

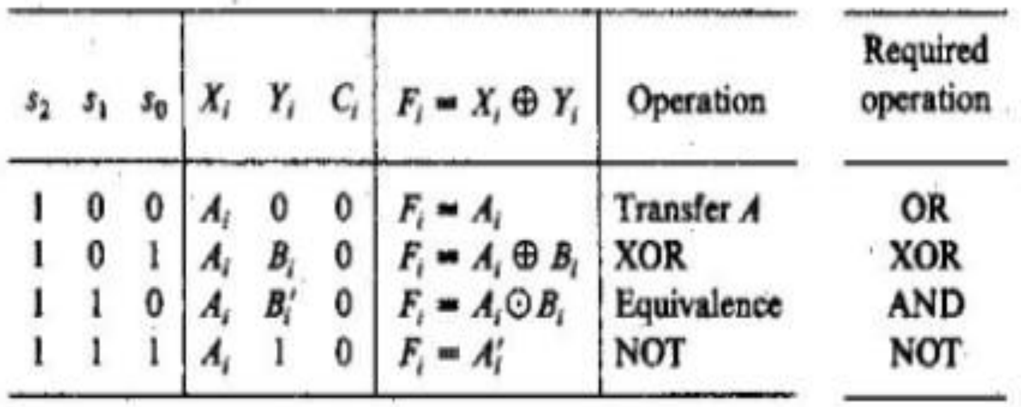
Combining logic and arithmetic Circuits

- The logic circuit can be combined with the arithmetic circuit to produce one arithmetic logic unit.
- Selection variables s1 and s0 can be made common to both sections, provided we use a third variable s2 to differentiate between the two.



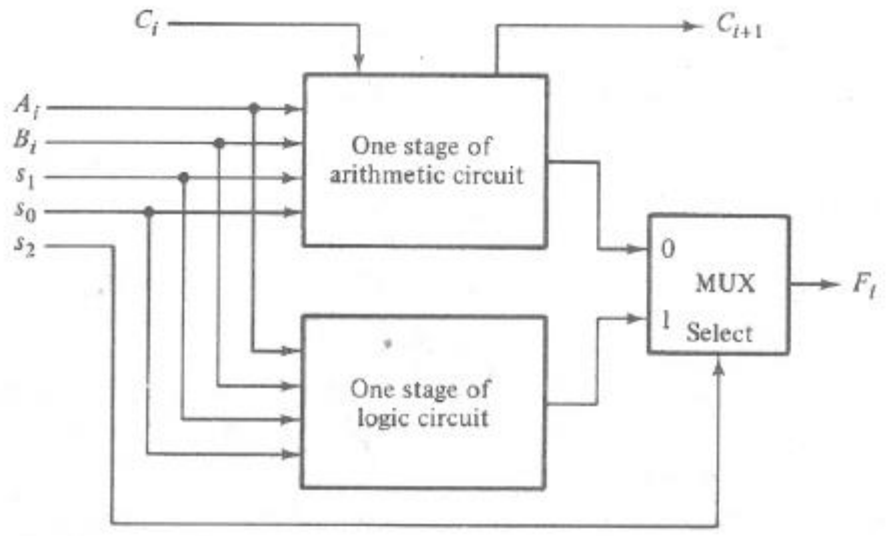
AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Logic Operations in one stage of Arithmetic Circuit



s_2	s_1	s_0	X_i	Y_i	C_i	$F_i = X_i \oplus Y_i$	Operation	Required operation
1	0	0	A_i	0	0	$F_i = A_i$	Transfer A	OR
1	0	1	A_i	B_i	0	$F_i = A_i \oplus B_i$	XOR	XOR
1	1	0	A_i	B_i'	0	$F_i = A_i \odot B_i$	Equivalence	AND
1	1	1	A_i	1	0	$F_i = A_i'$	NOT	NOT

Table 9-3



AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

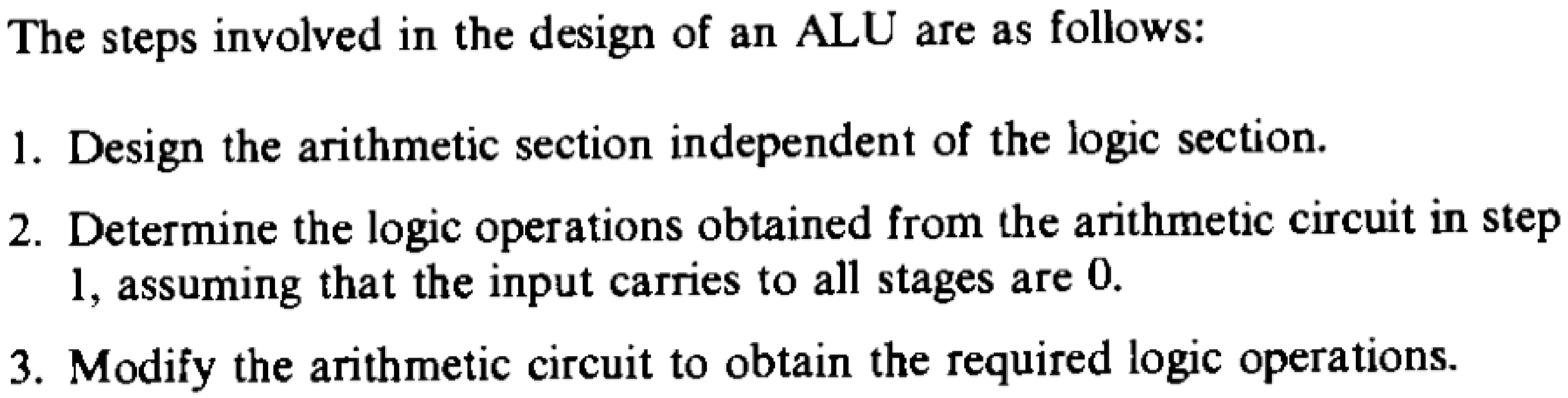
The value of X_i is always equal to input A_i. Table 9-3 shows the four logic operations obtained when a third selection variable $s_2 = 1$. This selection variable forces C_i to be equal to 0 while s_1 and s_0 choose a particular value for Y_i. The four logic operations obtained by this configuration are transfer, exclusive-OR, equivalence, and complement. The third entry is the equivalence operation because:

$$A_i \oplus B_i' = A_iB_i + A_i'B_i' = A_i \odot B_i$$

The last entry in the table is the NOT or complement operation because:

$$A_i \oplus 1 = A_i'$$

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.

The solution to the first design step is shown in Fig. 9-8. The solution to the second design step is presented in Table 9-3. The solution of the third step is carried out below.

From Table 9-3, we see that when $s_2 = 1$, the input carry C_i in each stage must be 0. With $s_1 s_0 = 00$, each stage as it stands generates the function $F_i = A_i$. To change the output to an OR operation, we must change the input to each full-adder circuit from A_i to $A_i + B_i$. This can be accomplished by ORing B_i and A_i when $s_2 s_1 s_0 = 100$.

The other selection variables that give an undesirable output occur when $s_2 s_1 s_0 = 110$. The unit as it stands generates an output $F_i = A_i \odot B_i$, but we want to generate the AND operation $F_i = A_i B_i$. Let us investigate the possibility of ORing each input A_i with some Boolean function K_i. The function so obtained is then used for X_i when $s_2 s_1 s_0 = 110$:

$$F_i = X_i \oplus Y_i = (A_i + K_i) \oplus B_i' = A_i B_i + K_i B_i + A_i' K_i' B_i'$$

Careful inspection of the result reveals that if the variable $K_i = B_i'$, we obtain an output:

$$F_i = A_i B_i + B_i' B_i + A_i B_i B_i' = A_i B_i$$

Two terms are equal to 0 because $B_i B_i' = 0$. The result obtained is the AND operation as required. The conclusion is that, if A_i is ORed with B_i' when $s_2 s_1 s_0 = 110$, the output will generate the AND operation.

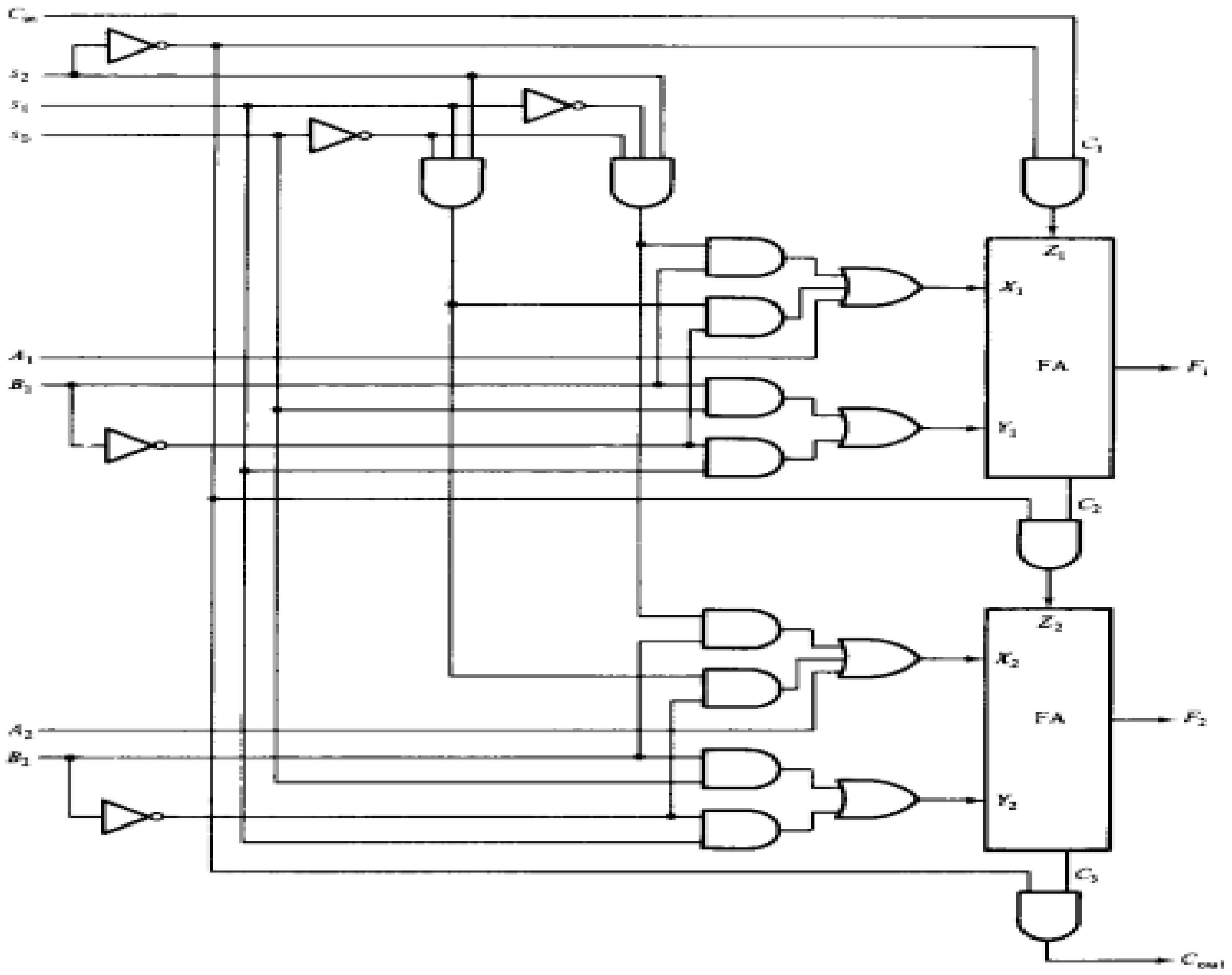


Figure 9-13 Logic diagram of arithmetic logic unit (ALU)

The final ALU is shown in Fig. 9-13. Only the first two stages are drawn, but the diagram can be easily extended to more stages. The inputs to each full-adder circuit are specified by the Boolean functions:

$$X_i = A_i + s_2 s_1' s_0' B_i + s_2 s_1 s_0' B_i'$$
$$Y_i = s_0 B_i + s_1 B_i'$$
$$Z_i = s_2' C_i$$

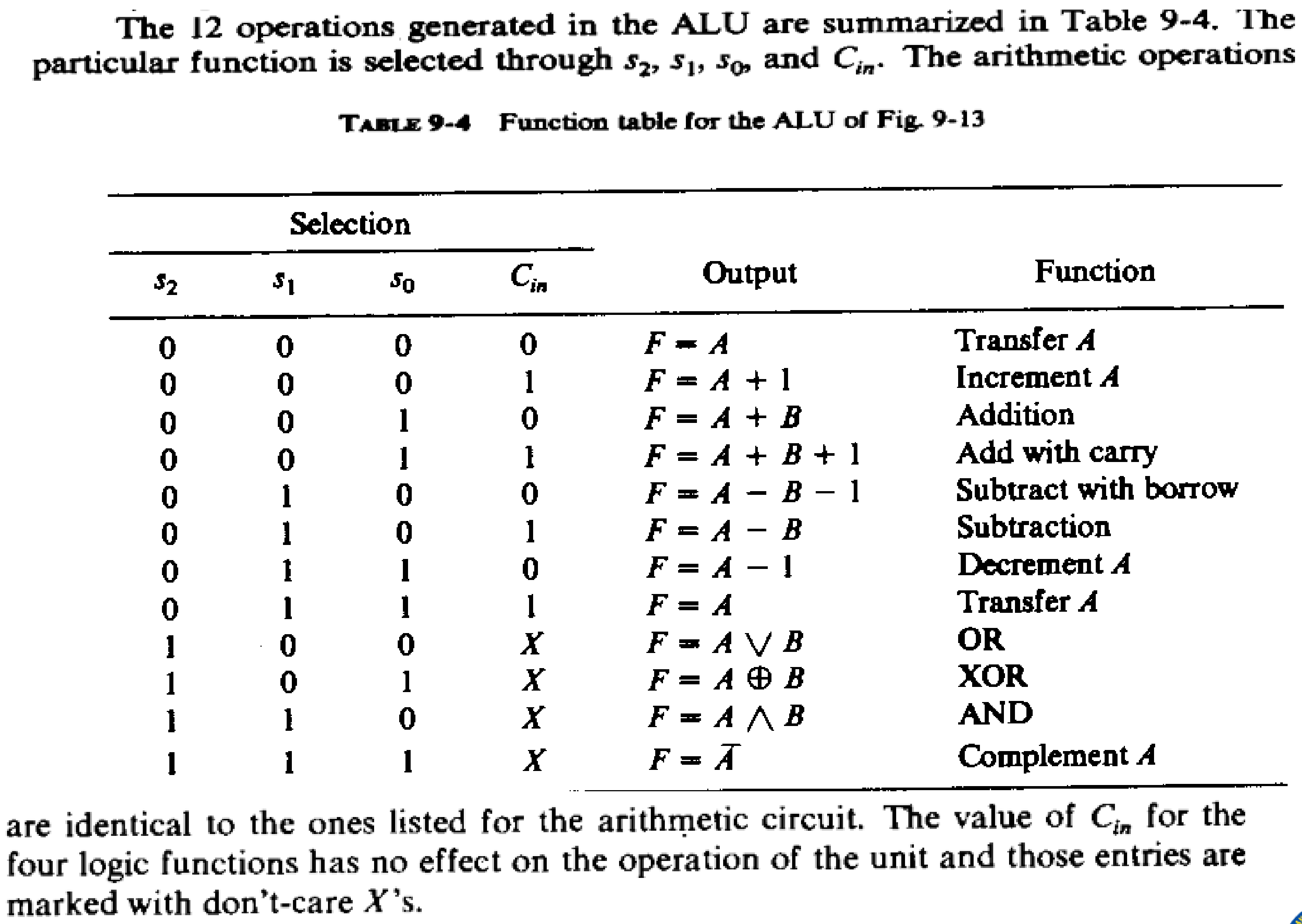
When $s_2 = 0$, the three functions reduce to:

$$X_i = A_i$$
$$Y_i = s_0 B_i + s_1 B_i'$$
$$Z_i = C_i$$

which are the functions for the arithmetic circuit of Fig. 9-8. The logic operations are generated when $s_2 = 1$. For $s_2 s_1 s_0 = 101$ or 111, the functions reduce to:

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

Output F_i is then equal to $X_i \oplus Y_i$ and produces the exclusive-OR and complement operations as specified in Table 9-3. When $s_2 s_1 s_0 = 100$, each A_i is ORed with B_i to provide the OR operation as discussed above. When $s_2 s_1 s_0 = 110$, each A_i is ORed with B_i' to provide the AND operation as explained previously.



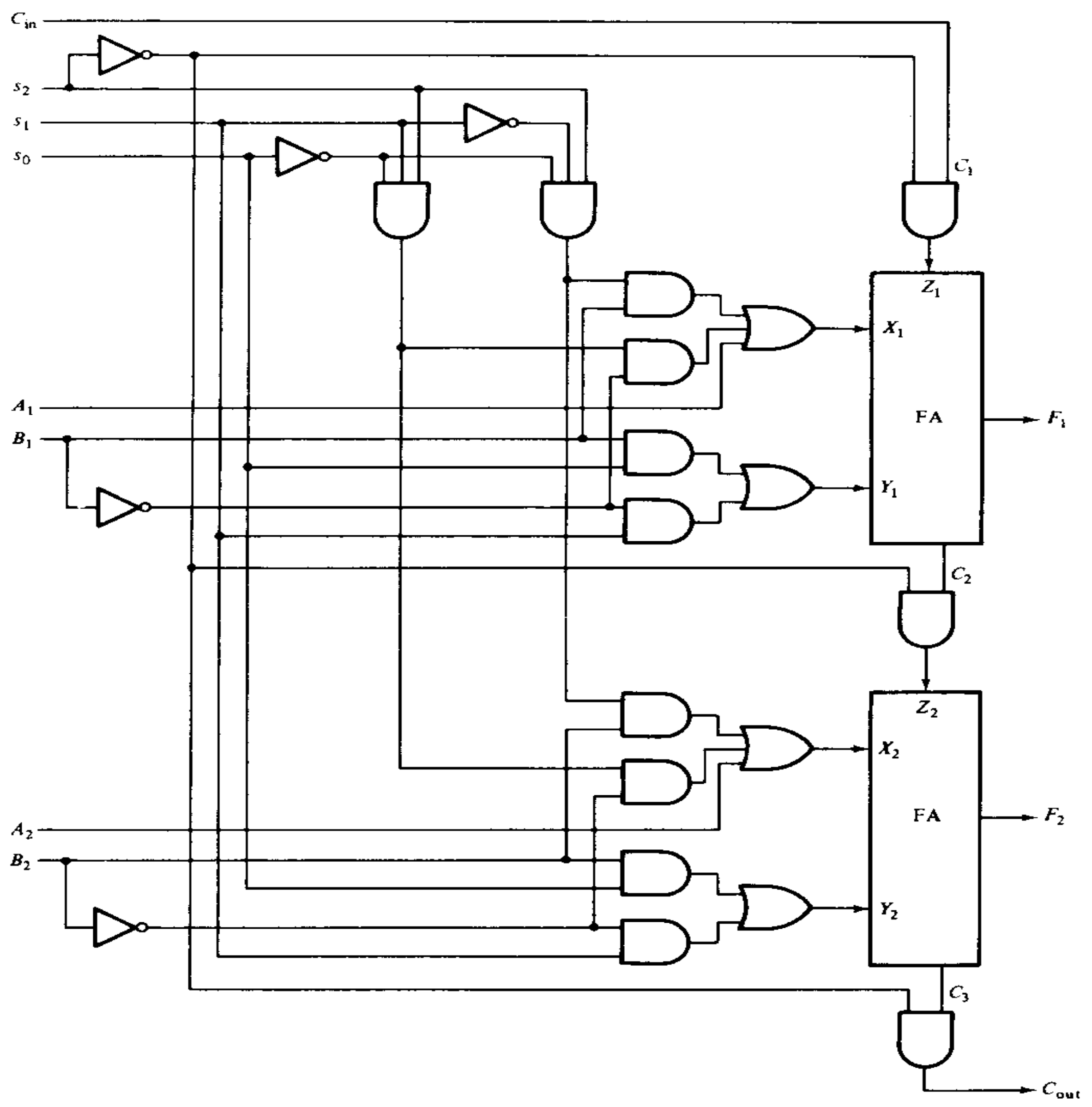
The 12 operations generated in the ALU are summarized in Table 9-4. The particular function is selected through s_2, s_1, s_0, and C_{in}. The arithmetic operations

TABLE 9-4 Function table for the ALU of Fig. 9-13

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

are identical to the ones listed for the arithmetic circuit. The value of C_{in} for the four logic functions has no effect on the operation of the unit and those entries are marked with don't-care X's.

Logic Diagram of ALU (Only first two stage are drawn)



Next Class.....

- Status Register
- Shifter
- Design of processor Unit with control variables
- Micro operations for processor