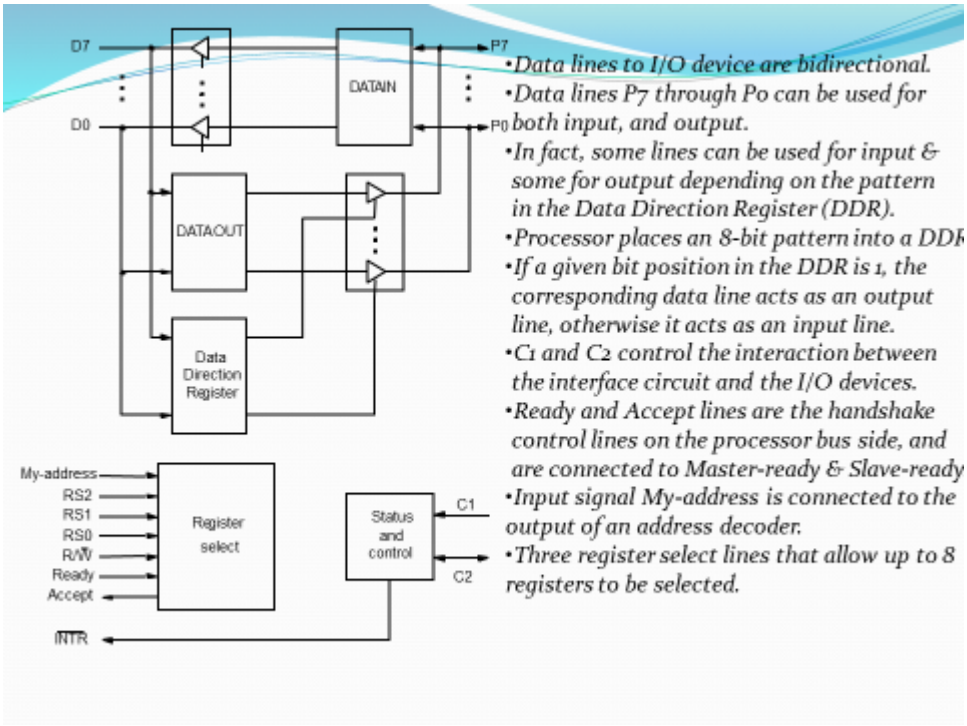
Internal Assesment Test 2 –SHEME & ANSWER KEY

Sub:	COMPUTER ORGANIZATION	Sub Code:	18CS34	Branch:	CSE

Answer **FIVE** FULL questions selecting AT LEAST ONE question **FROM EACH PART**

		MARKS	CO	RBT
	PART A			
1 (a)	With a neat diagram, explain general 8-bit parallel interface. (6+4)	[6+4]	CO3	L2
	OR			
2 (a)	With neat timing diagrams, explain synchronous bus. (6+4)	[6+4]	CO3	L2
	PART B			
3 (a)	Explain the tree structure of USB with split bus operation in detail. Diagram 4 m + 6 m	[10]	CO3	L2
	OR			
4 (a)	Consider sixteen words and each word having eight bits. Draw internal organization of the specified RAM memory chip. (7+3)	[7+3]	CO4	L3
	PART C			
5 (a)	With a neat diagram explain addition subtraction logic circuit .Also with a neat diagram explain sequential circuit multiplication. (5+5)	[5+5]	CO5	L2
	OR			
6 (a)	Perform multiplication for -13 and -14 using Booth algorithm.	[5]	CO5	L3
(b)	Perform multiplication for +13 and -17 using Bit-pair recoding.	[5]	CO5	L3
	PART D			
7 (a)	Perform multiplication of 101101 and 111111 using Carry-save Addition.	[10]	CO5	L3
	OR			
8 (a)	With a figure, explain circuit arrangement for binary division. (4+6)	[4+6]	CO5	L2
	PART E			
9 (a)	Explain Asynchronous DRAM for 2M*8 with the help of a block diagram.	[6]	CO4	L2
(b)	Short note on a) Cache memory b)Memory Access time c)Memory Cycle time d)Fast page mode	[4]	CO4	L1
	OR			
10 (a)	Explain 16-bit carry look-ahead adder.	[10]	CO5	L2

1. With a neat diagram, explain general 8-bit parallel interface.



2. With neat timing diagrams, explain synchronous bus.

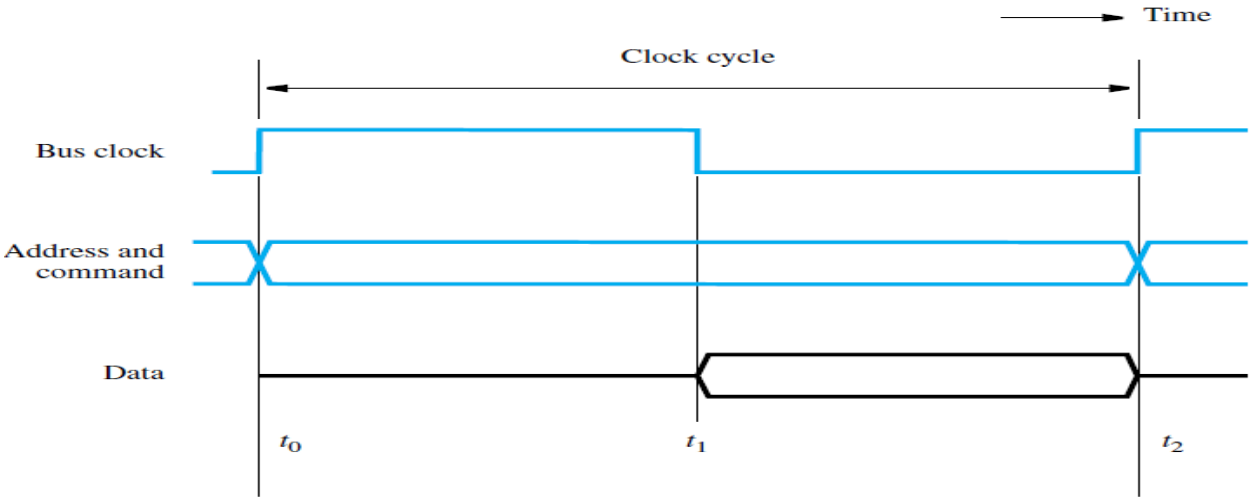


Figure 7.3 Timing of an input transfer on a synchronous bus.

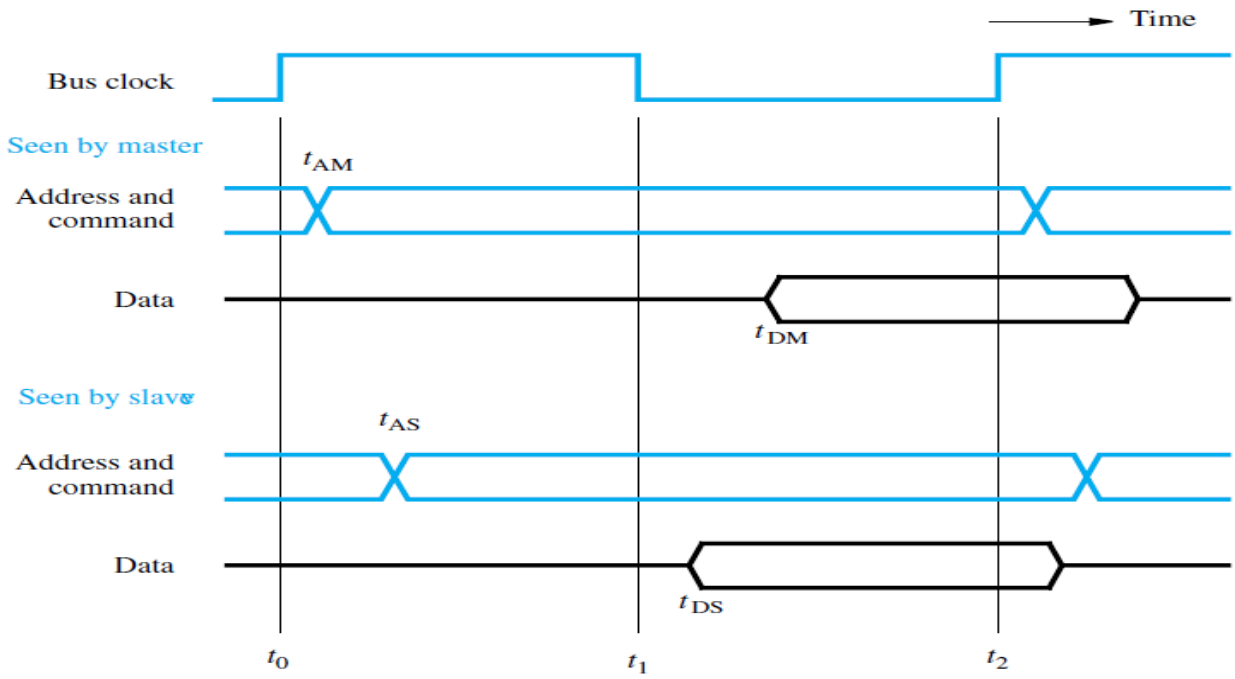


Figure 7.4 A detailed timing diagram for the input transfer of Figure 7.3.

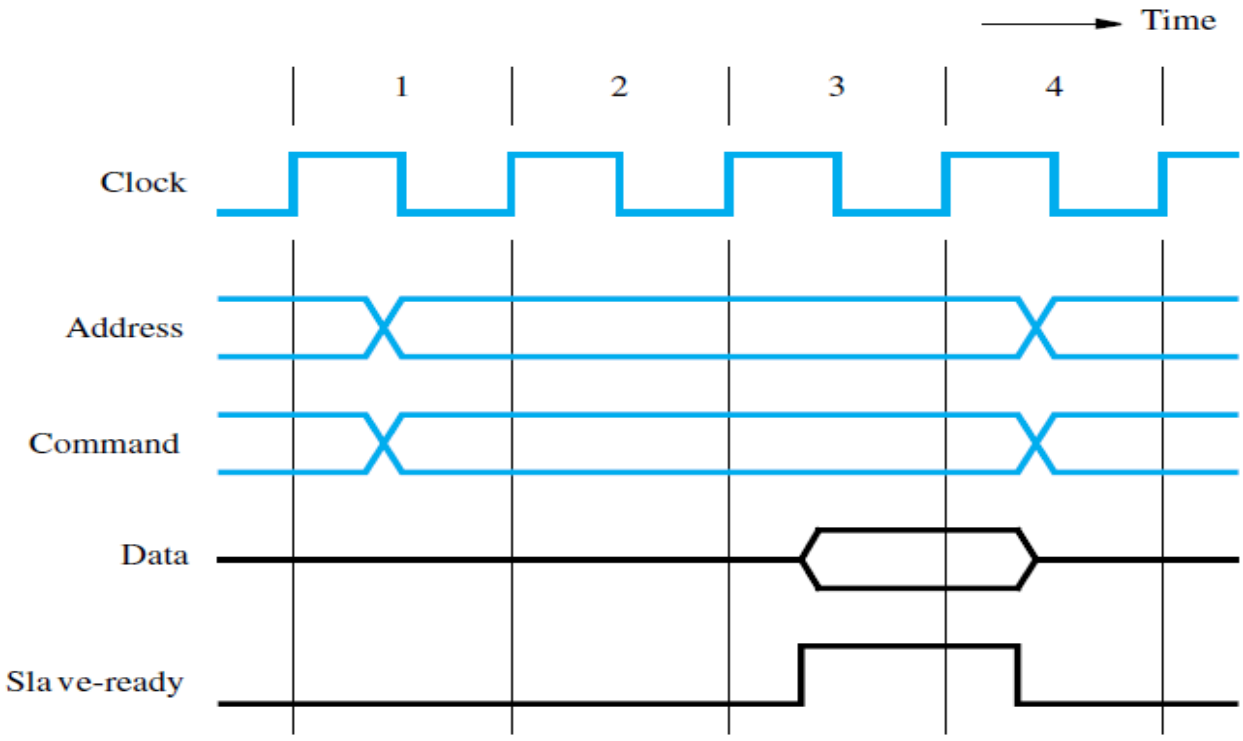
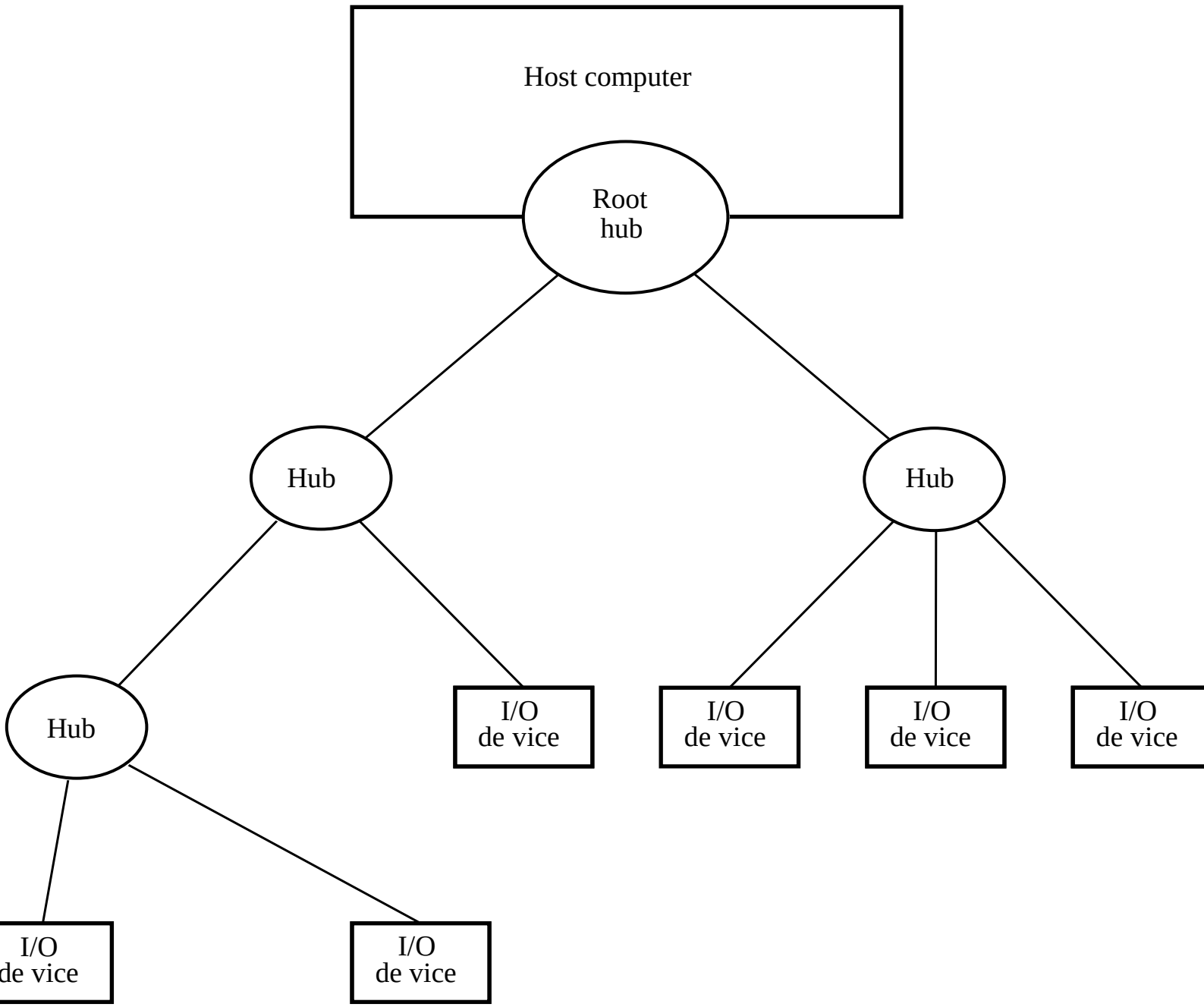


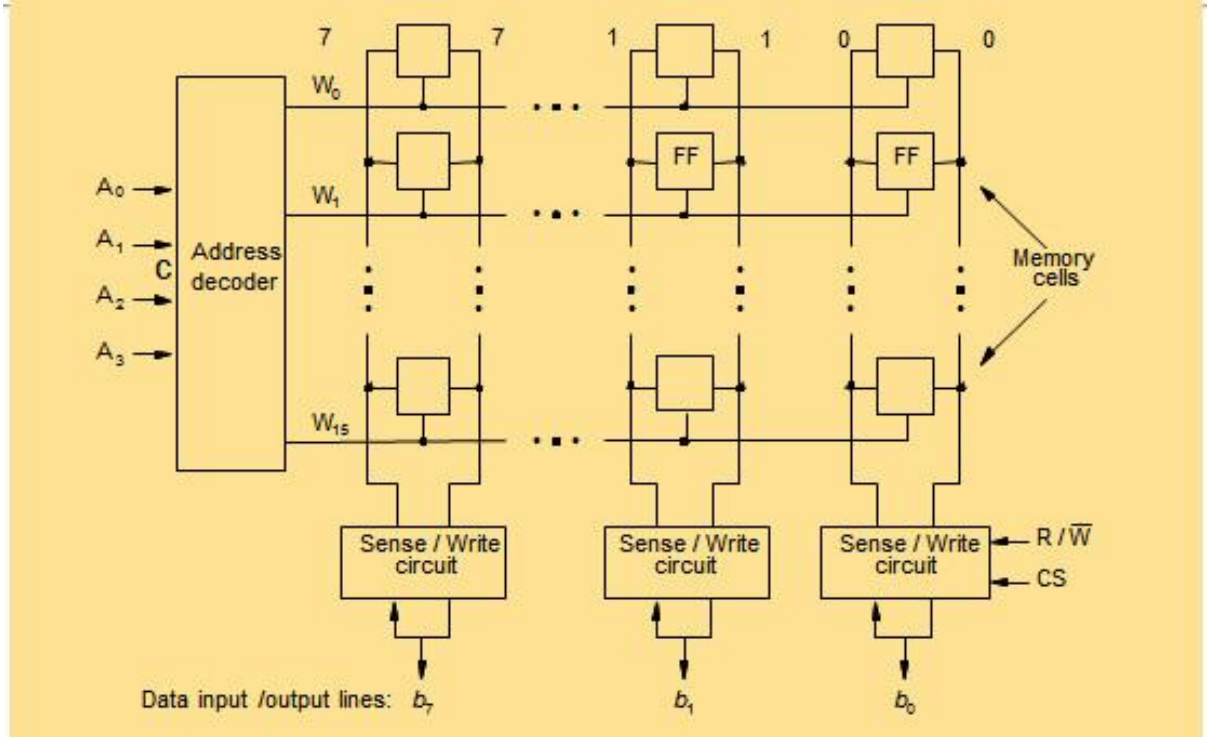
Figure 7.5 An input transfer using multiple clock cycles.

3. Explain the tree structure of USB with split bus operation in detail.



- To accommodate a large number of devices that can be added or removed at any time, the USB has the tree structure as shown in the figure.
- Each node of the tree has a device called a hub, which acts as an intermediate control point between the host and the I/O devices. At the root of the tree, a root hub connects the entire tree to the host computer. The leaves of the tree are the I/O devices being served (for example, keyboard, Internet connection, speaker, or digital TV)
- In normal operation, a hub copies a message that it receives from its upstream connection to all its downstream ports. As a result, a message sent by the host computer is broadcast to all I/O devices, but only the addressed device will respond to that message. However, a message from an I/O device is sent only upstream towards the root of the tree and is not seen by other devices. Hence, the USB enables the host to communicate with the I/O devices, but it does not enable these devices to communicate with each other.

4. Consider sixteen words and each word having eight bits. Draw internal organization of the specified RAM memory chip.



The above figure is an example of a very small memory chip consisting of 16 words of 8 bits each. This is referred to as a 16×8 organization. The data input and the data output of each Sense/Write circuit are connected to a single bidirectional data line that can be connected to the data bus of a computer. Two control lines, R/W (Read/ Write) input specifies the required operation, and the CS (Chip Select) input selects a given chip in a multichip memory system. The memory circuit given above stores 128 and requires 14 external connections for address, data and control lines. Of course, it also needs two lines for power supply and ground connections. Consider now a slightly larger memory circuit, one that has a 1k (1024) memory cells. For a 1k×1 memory organization, the representation is given next. The required 10-bit address is divided into two groups of 5 bits each to form the row and column addresses for the cell array. A row address selects a row of 32 cells, all of which are accessed in parallel. However, according to the column address, only one of these cells is connected to the external data line by the output multiplexer and input de-multiplexer.

5. With a neat diagram explain addition subtraction logic circuit.

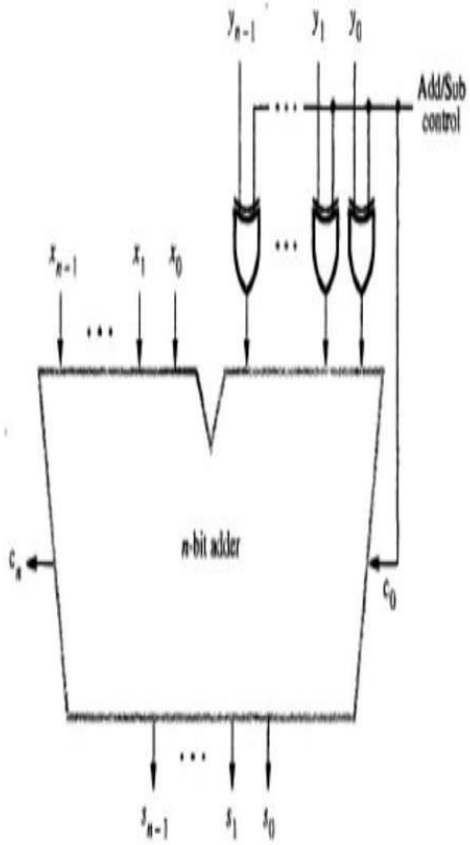
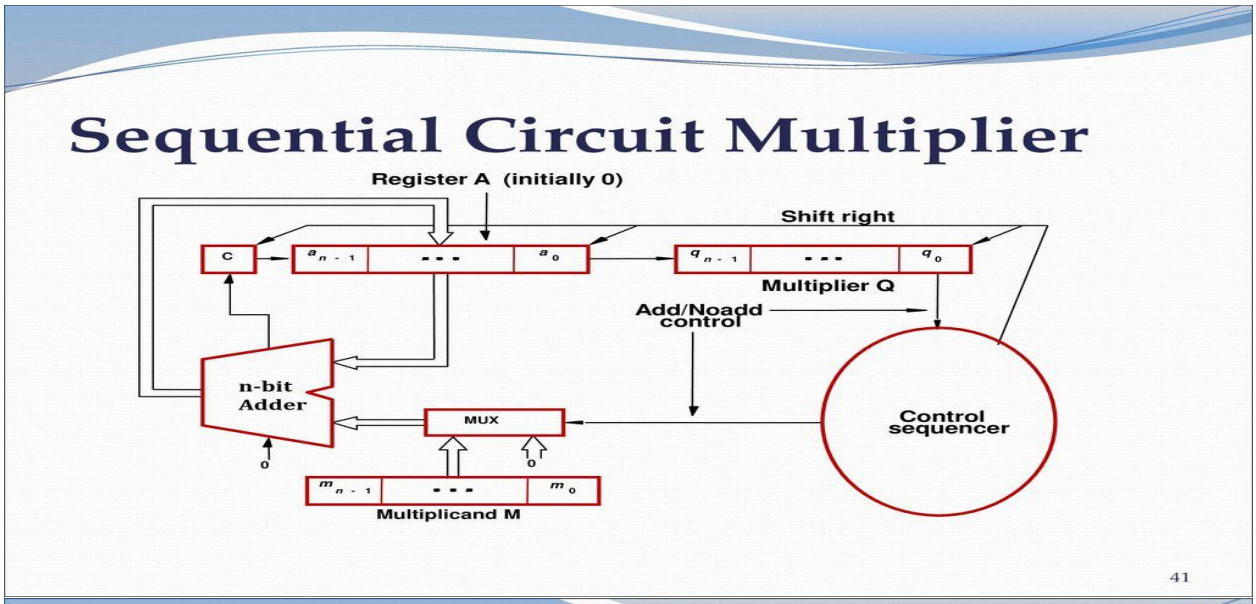


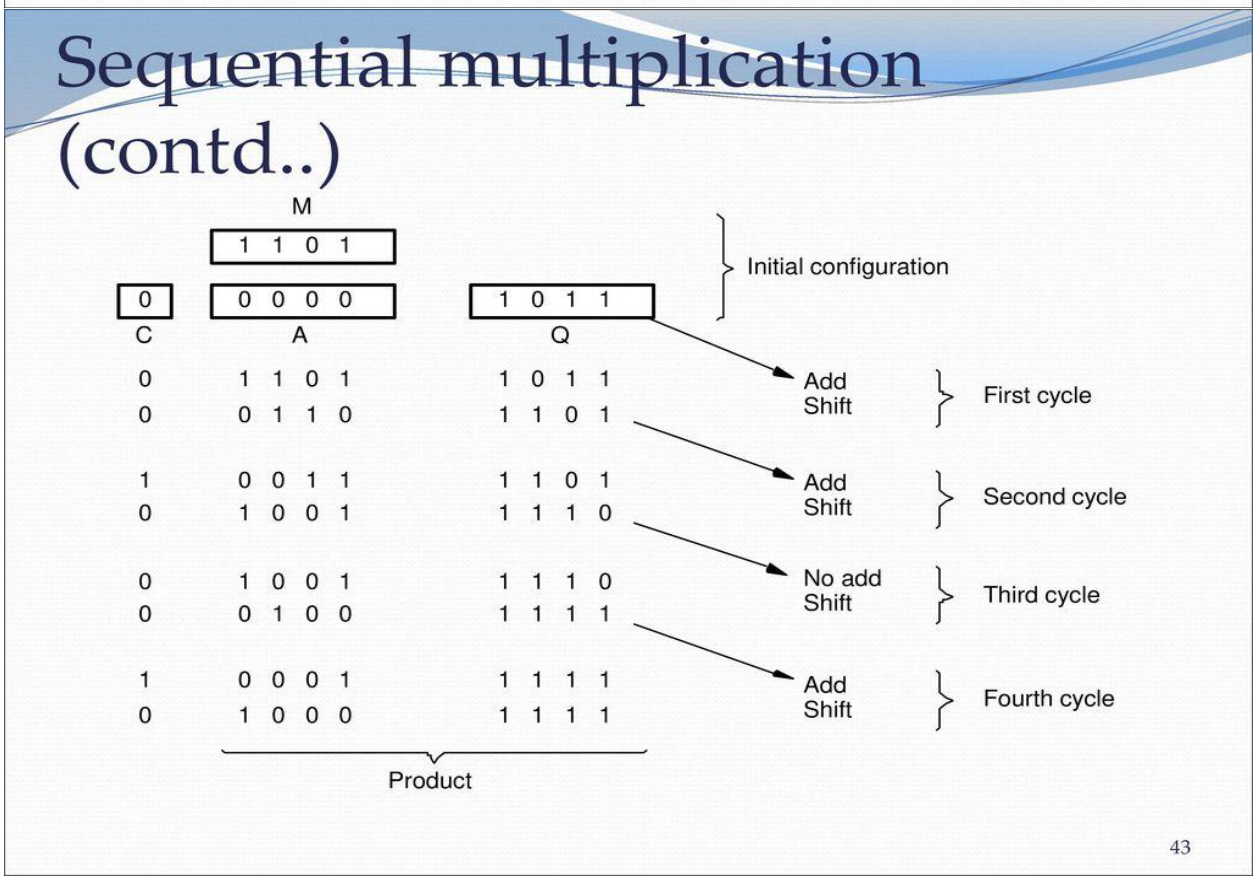
Figure 6.3 Binary addition-subtraction logic network.

The n-bit adder can be used to add 2's complement numbers X and Y.

- Overflow can only occur when the signs of the 2 operands are the same.
- In order to perform the subtraction operation X-Y on 2's complement numbers X and Y; We form the 2's complement of Y and add it to X.
- Addition or subtraction operation is done based on value applied to the Add/Sub input Control-line.
- Control-line=0 for addition, applying the Y vector unchanged to one of the adder inputs. Control-line=1 for subtraction, the Y vector is 2's complemented

6. Also with a neat diagram explain sequential circuit multiplication.





• Registers A and Q combined hold PPi(partial product) while the multiplier bit qi generates the signal Add/Noadd.
• The carry-out from the adder is stored in flip-flop C.

• Procedure for multiplication:

1) Multiplier is loaded into register Q, Multiplicand is loaded into register M and C & A are cleared to 0.
2) If q0=1, add M to A and store sum in A. Then C, A and Q are shifted right one bit-position. If q0=0, no addition performed and C, A & Q are shifted right one bit-position.
3) After n cycles, the high-order half of the product is held in register A and the low-order half is held in register Q.

7. Perform multiplication for -13 and -14 using Booth algorithm

10011 (-13)
10010 (-14)

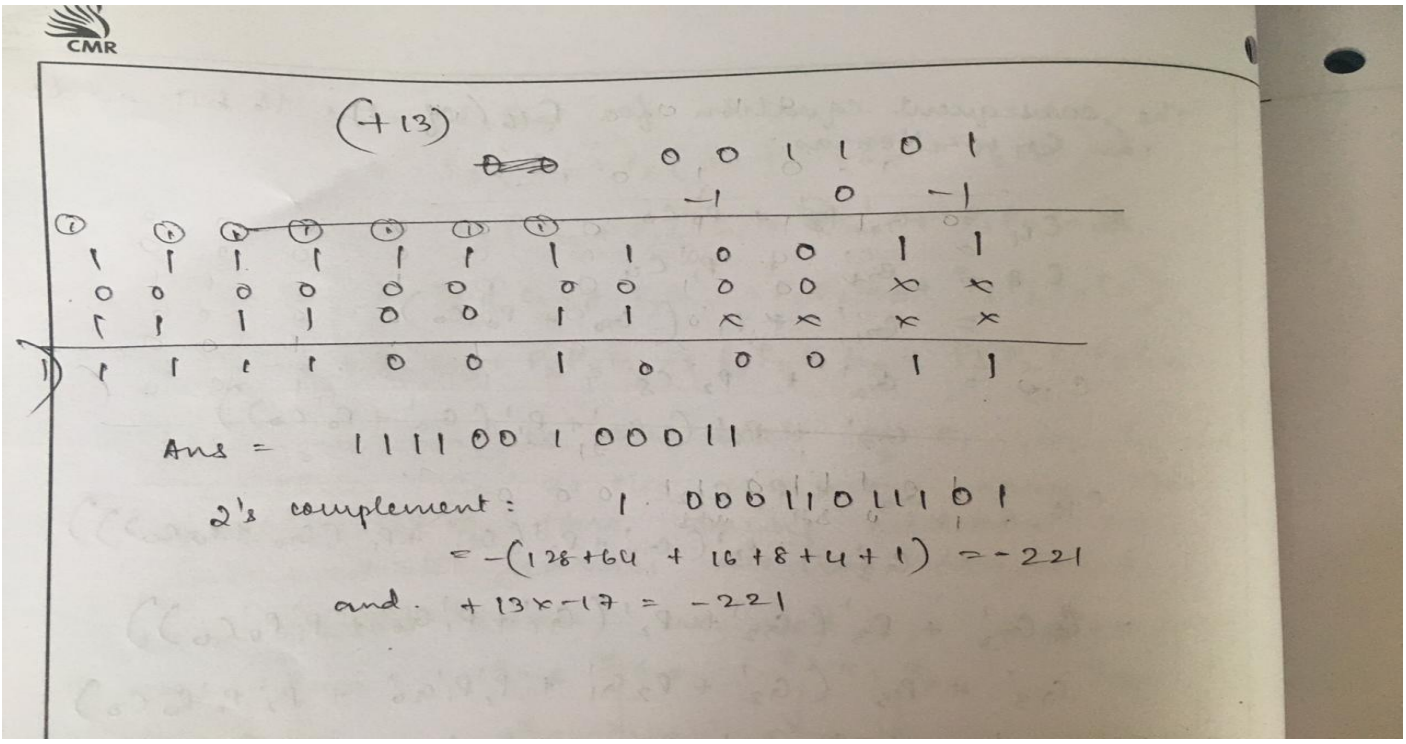
Booths recoding for -14(multiplier) : +1 0 -1 +1 0

```
                  1 0  0  1  1
                 +1 0 -1 +1  0
      ---------------------------------------
         0 0 0 0 0 0 0 0 0 0
         1 1 1 1 1 0 0 1 1
         0 0 0 0 1 1 0 1
         0 0 0 0 0 0 0
         1 1 0 0 1 1
      ----------------------------------------
   1     1 1 0 1 0 0 1 0 1 0
```

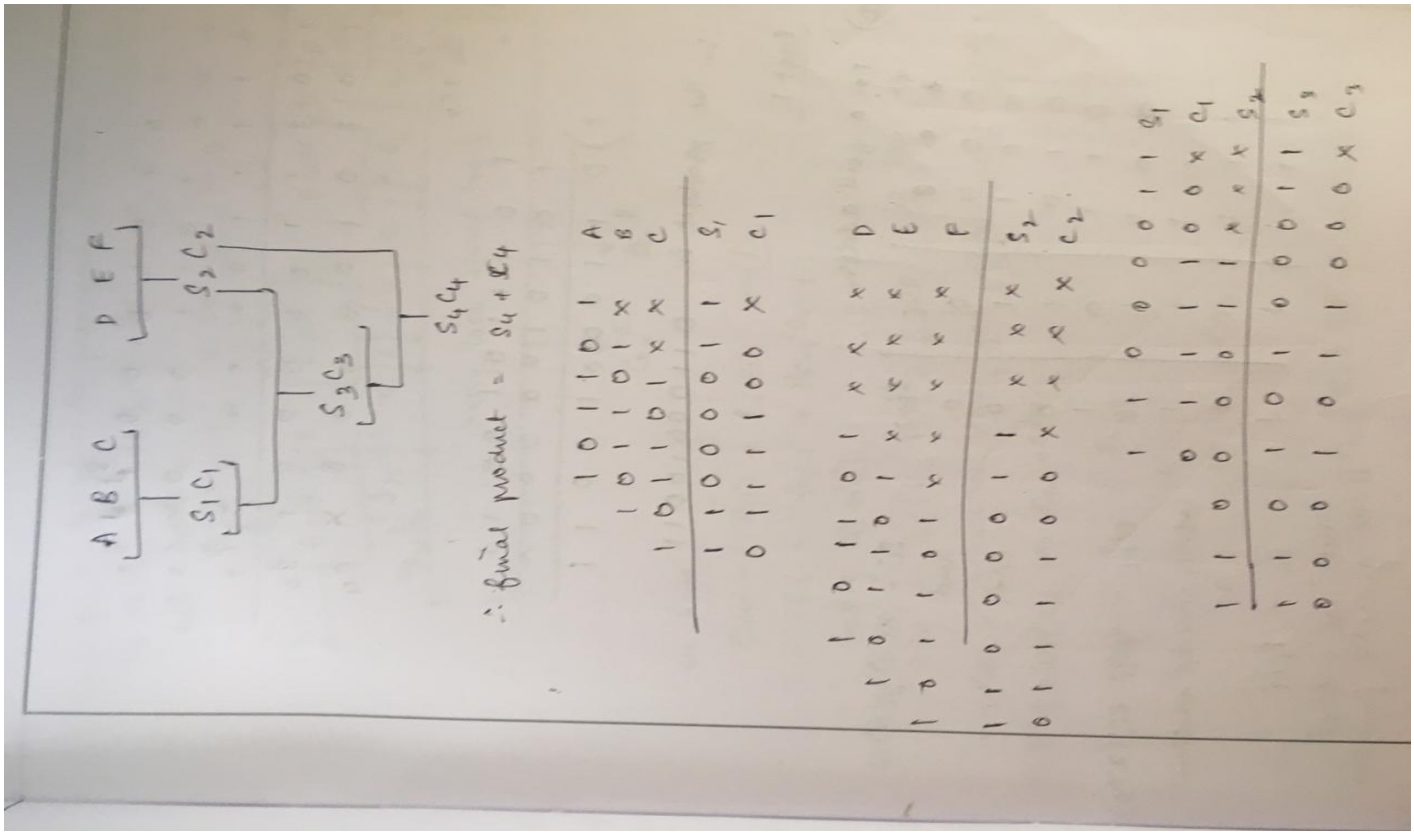
Take 2's complement of result 0 0 1 0 1 1 0 1 0 1
+ 1

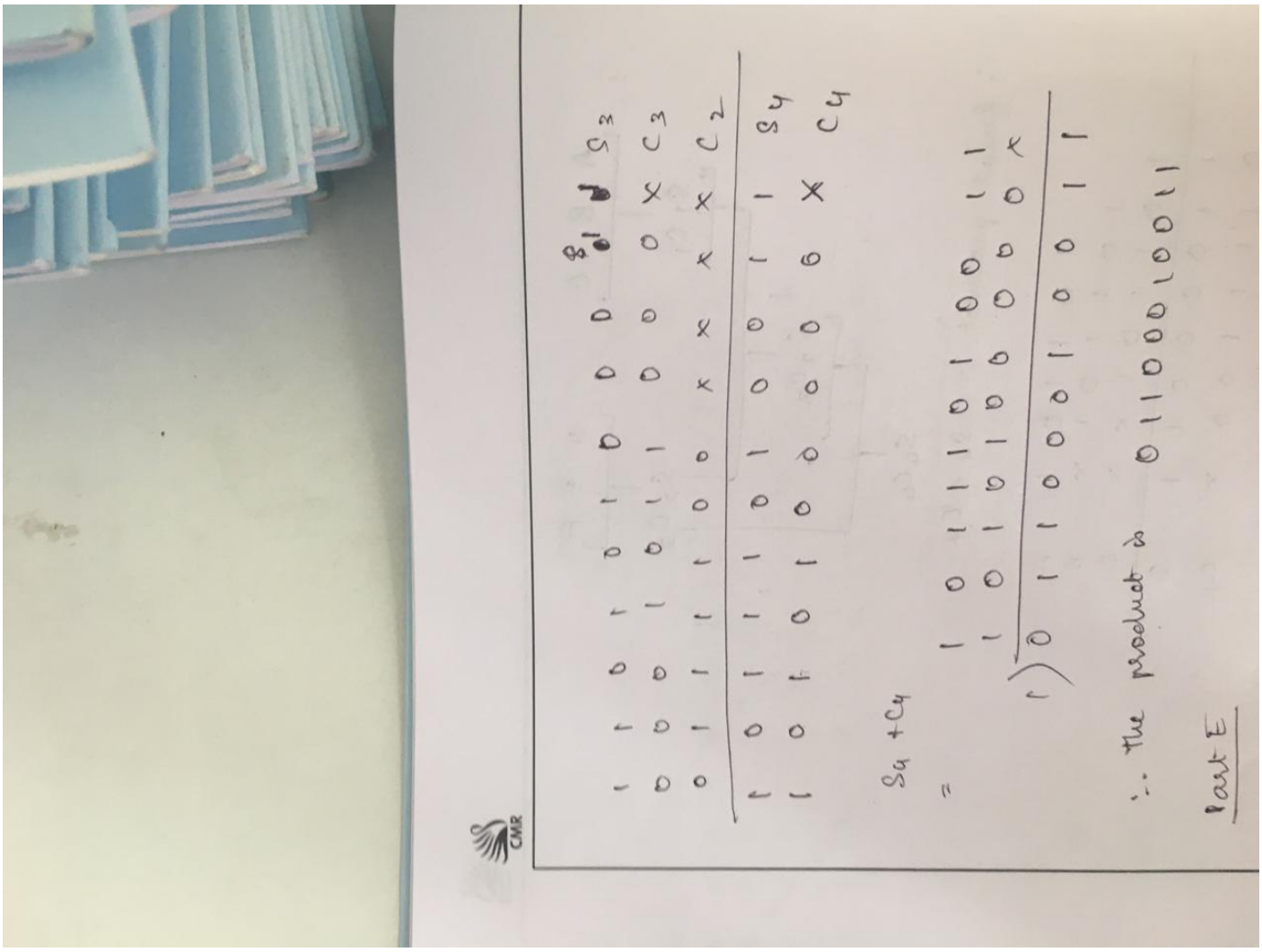
0 0 1 0 1 1 0 1 1 0
1282+ 16+4+2 =182

8. Perform multiplication for +13 and -17 using Bit-pair recoding.



9. Perform multiplication of 101101 and 111111 using Carry-save Addition.





10. With a figure, explain circuit arrangement for binary division.

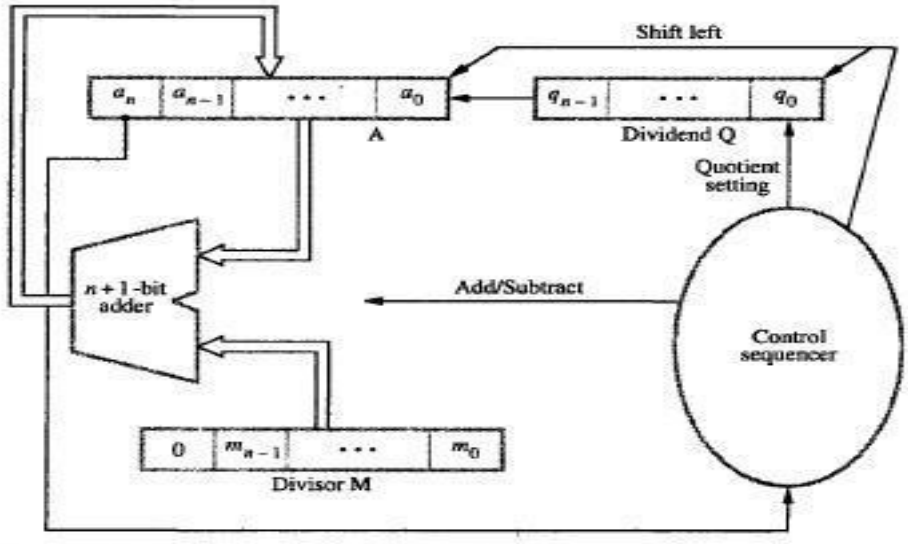


Figure 6.21 Circuit arrangement for binary division.

• An n-bit positive-divisor is loaded into register M. An n-bit positive-dividend is loaded into register Q at the start of the operation. Register A is set to 0
• After division operation, the n-bit quotient is in register Q, and the remainder is in register A.

• Procedure:
step 1:

Do the following n times

i) If the sign of A is 0, shift A and Q left one bit position and subtract M from A; otherwise, shift A and Q left and add M to A .

ii) Now, if the sign of A is 0, set q0 to 1; otherwise set q0 to 0.

Step 2:

If the sign of A is 1, add M to A (restore).

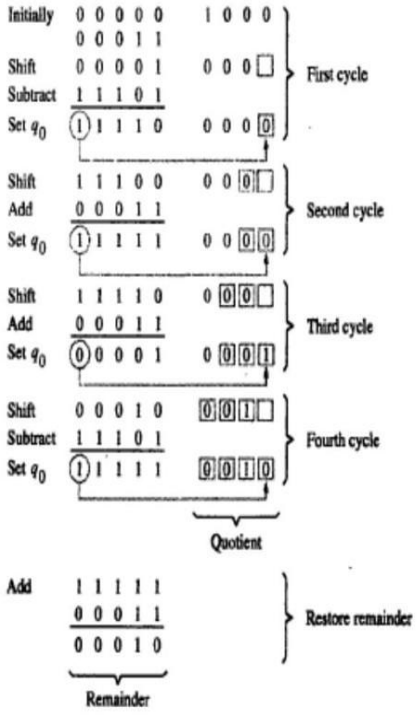
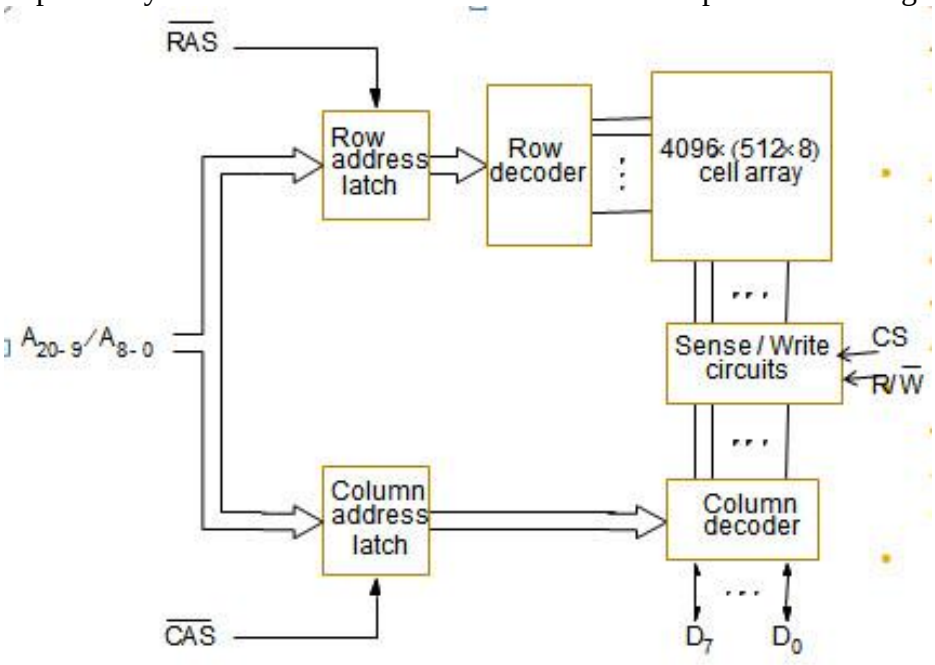


Figure 6.23 A nonrestoring-division example.

---

11. Explain Asynchronous DRAM for 2M*8 with the help of a block diagram.



- Each row can store 512 bytes. 12 bits to select a row, and 9 bits to select a group In a row. Total of 21 bits.
- First apply the row address; RAS signal latches the row address. Then apply the Column address, CAS signal latches the address.
- Timing of the memory unit is controlled by a specialized unit which generates RAS and CAS.
- This is asynchronous DRAM

12. Short note on a) Cache memory b)Memory Access time c)Memory Cycle time d)Fast page mode

Memory Access Times: -

It is a useful measure of the speed of the memory unit. It is the time that elapses between the initiation of an operation and the completion of that operation (for example, the time between READ and MFC).

Memory Cycle Time :-

It is an important measure of the memory system. It is the minimum time delay required between the initiations of two successive memory operations (for example, the time between two successive READ operations). The cycle time is usually slightly longer than the access time.

Cache Memory:-

This is a small and fast memory that is inserted between the larger, slower main memory and the CPU. This holds the currently active segments of a program and its data. Because of the locality of address references, the CPU can, most of the time, find the relevant information in the cache memory itself (cache hit) and infrequently needs access to the main memory (cache miss) with suitable size of the cache memory, cache hit rates of over 90% are possible leading to a cost-effective increase in the performance of the system.

Fast Page Mode

1. Suppose if we want to access the consecutive bytes in the selected row.
2. This can be done without having to reselect the row.
3. Add a latch at the output of the sense circuits in each row.
4. All the latches are loaded when the row is selected.
5. Different column addresses can be applied to select and place different bytes on the data lines.

13. bit carry look-ahead adder

CARRY-LOOKAHEAD ADDITIONS

• The logic expression for si(sum) and ci+1(carry-out) of stage i are

si= xi+yi+ci ------(1)

ci+1=xiyi+xici+yici ------(2)

• Factoring (2) into
ci+1=xiyi+(xi+yi)ci
we can write ci+1=Gi+ PiCi where Gi=xiyi and Pi=xi+yi
• The expressions Gi and Pi are called generate and propagate functions .
• If Gi=1, then ci+1=1, independent of the input carry ci. This occurs when both xi and yi are 1. Propagate function means that an input-carry will produce an output-carry when either xi=1 or yi=1.
• All Gi and Pi functions can be formed independently and in parallel in one logic-gate delay.
• Expanding ci terms of i-1 subscripted variables and substituting into the ci+1 expression, we obtain
ci+1= Gi+PiGi-1+PiPi-1Gi-2. . . . . .+P1G0+PiPi-1 . . . P0c0

• Conclusion: Delay through the adder is 3 gate delays for all carry-bits &
4 gate delays for all sum-bits.
• Consider the design of a 4-bit adder. The carries can be implemented as

c1=G0+P0c0
c2=G1+P1G0+P1P0c0
c3=G2+P2G1+P2P1G0+P2P1P0c0
c4=G3+P3G2+P3P2G1+P3P2P1G0+P3P2P1P0c0
• The carries are implemented in the block labeled carry-look ahead logic. An adder implemented in this form is called a *carry-look ahead adder*.
• Limitation: If we try to extend the carry-look ahead adder for longer operands, we run into a problem of gate fan-in constraints.

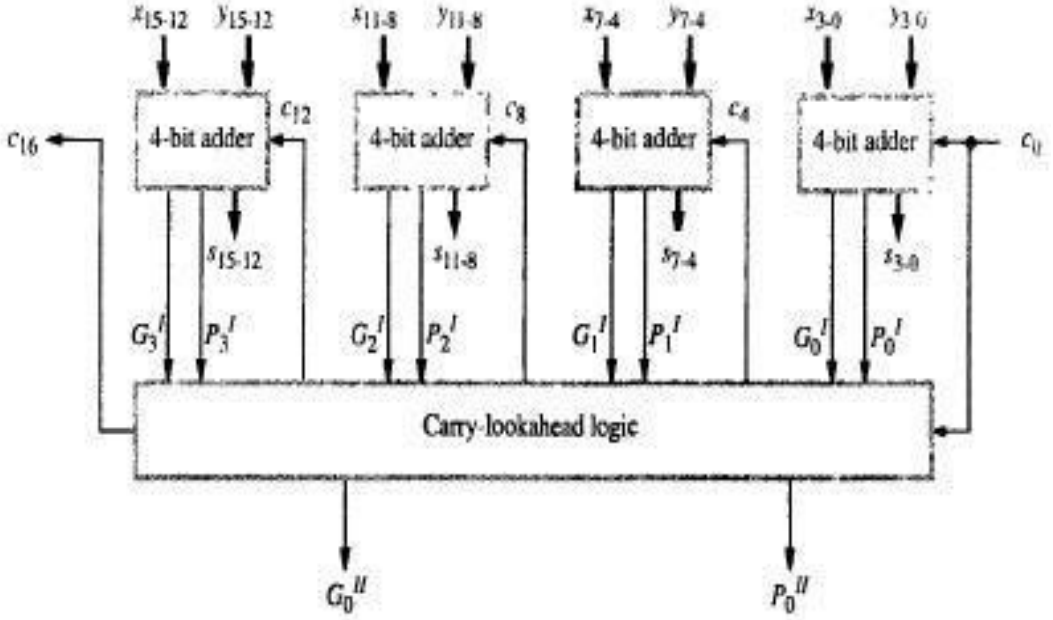


Figure 6.5 16-bit carry-lookahead adder built from 4-bit adders (see Figure 6.4b).