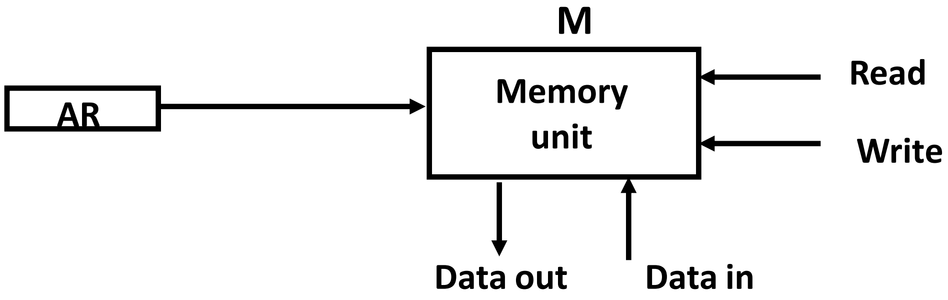
Memory Transfer

Memory is usually accessed in computer systems by putting the desired address in a special register, the Memory Address Register (MAR, or AR)



Memory Read

- **To read a value from a location in memory and load it into a register, the register transfer language notation looks like this:**

R1 ← M[MAR]

- **This causes the following to occur**
  1. **The contents of the MAR get sent to the memory address lines**
  2. **A Read (= 1) gets sent to the memory unit**
  3. **The contents of the specified address are put on the memory's output data lines**
  4. **These get sent over the bus to be loaded into register R1**

Memory Write

- **To write a value from a register to a location in memory looks like this in register transfer language:**

M[MAR] ← R1

- **This causes the following to occur**
  1. **The contents of the MAR get sent to the memory address lines**
  2. **A Write (= 1) gets sent to the memory unit**
  3. **The values in register R1 get sent over the bus to the data input lines of the memory**
  4. **The values get loaded into the specified address in the memory**

- Memory is used to store ………..

a) Data
b) Instructions
c) Both A and B
d) None of the above

SUMMARY OF R. TRANSFER MICROOPERATIONS

A ← B	**1.Transfer content of reg. B into reg. A**
AR ← DR(AD)	**2.Transfer content of AD portion of reg. DR into reg. AR**
A ← constant	**3.Transfer a binary constant into reg. A**
ABUS ← R1, R2 ← ABUS	**4.Transfer content of R1 into bus A and, at the same time, transfer content of bus A into R2**
AR	**5.Address register**
DR	**6.Data register**
M[R]	**7.Memory word specified by reg. R**
M	**8.Equivalent to M[AR]**
DR ← M	**9.Memory *read* operation: transfers content of memory word specified by AR into DR**
M ← DR	**10.Memory *write* operation: transfers content of DR into memory word specified by AR**

Overview

- Register Transfer Language
- Register Transfer
- Bus and Memory Transfers
- **Arithmetic Micro-operations**
- Logic Micro-operations
- Shift Micro-operations
- Arithmetic Logic Shift Unit

MICROOPERATIONS

Computer system microoperations are of four types:

- **Register transfer microoperations**
- **Arithmetic microoperations**
- **Logic microoperations**
- **Shift microoperations**

Arithmetic MICROOPERATIONS

- **The basic arithmetic microoperations are**
  - Addition
  - Subtraction
  - Increment
  - Decrement

- **The additional arithmetic microoperations are**
  - Add with carry
  - Subtract with borrow
  - Transfer/Load
  - etc. …

Summary of Typical Arithmetic Micro-Operations

R3 ← R1 + R2	Contents of R1 plus R2 transferred to R3
R3 ← R1 - R2	Contents of R1 minus R2 transferred to R3
R2 ← R2’	Complement the contents of R2
R2 ← R2’+ 1	2's complement the contents of R2 (negate)
R3 ← R1 + R2’+ 1	subtraction
R1 ← R1 + 1	Increment
R1 ← R1 - 1	Decrement

Binary Adder

- 4-bit Binary Adder : ***Fig. 4-6***
  - Full adder = 2-bits sum + previous carry
  - Binary adder = the arithmetic sum of two binary numbers of any length
  - c_0(input carry), c_4(output carry)

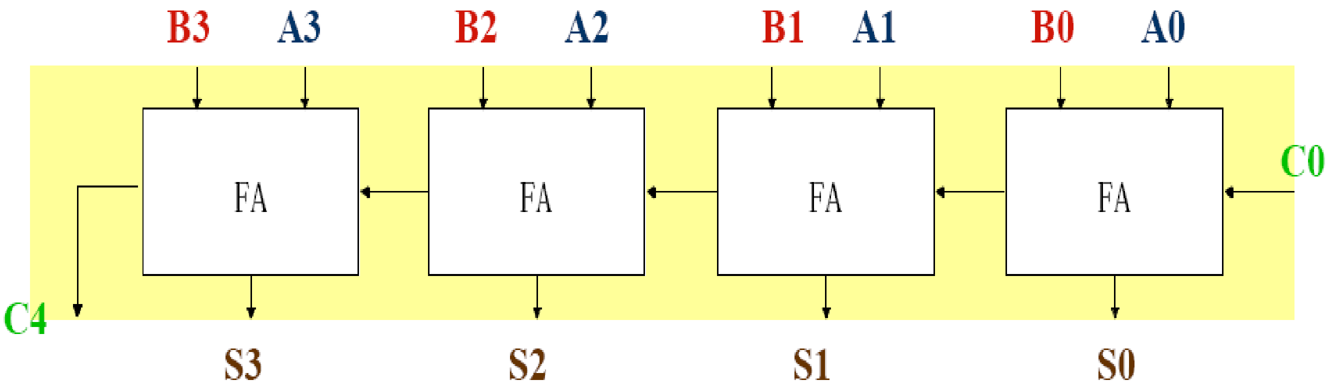
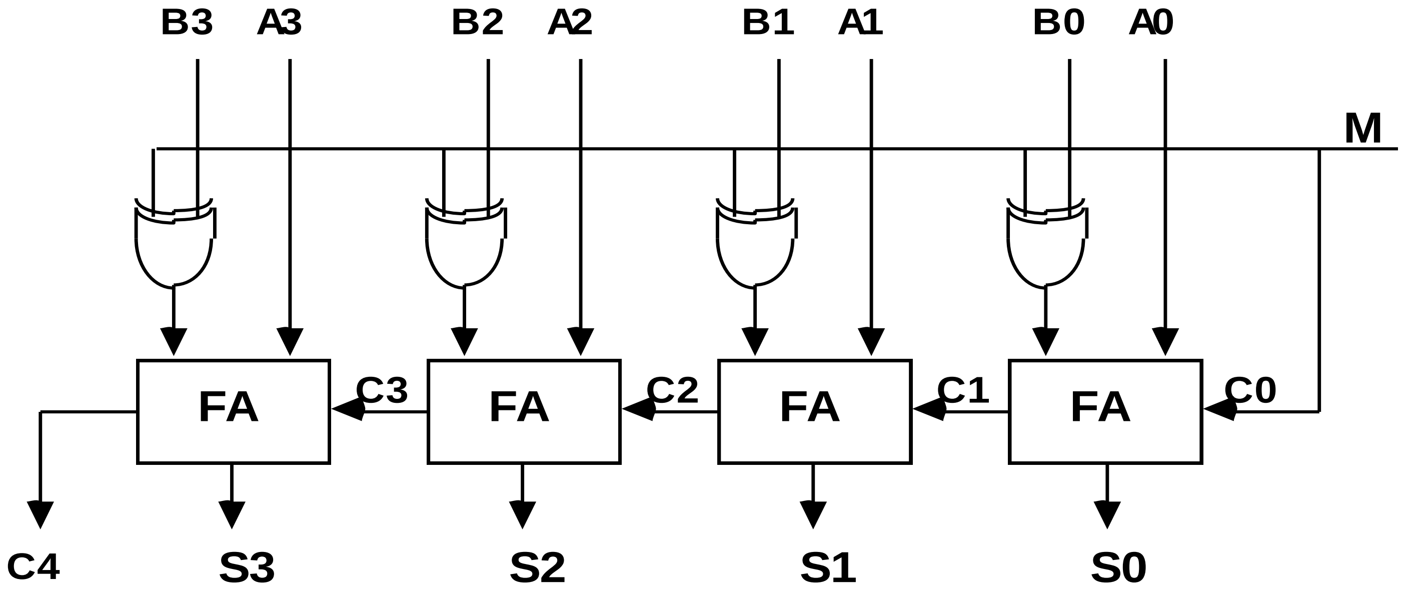


Figure 4-6. 4-bit binary adder

Binary Adder-Subtractor

Binary Adder-Subtractor



- ➢ Mode input M controls the operation
  - ➢ M=0 ---- adder
  - ➢ M=1 ---- subtractor

- Which of the following adder is used to perform the arithmetic sum of two binary numbers of any length?

A) Full adder

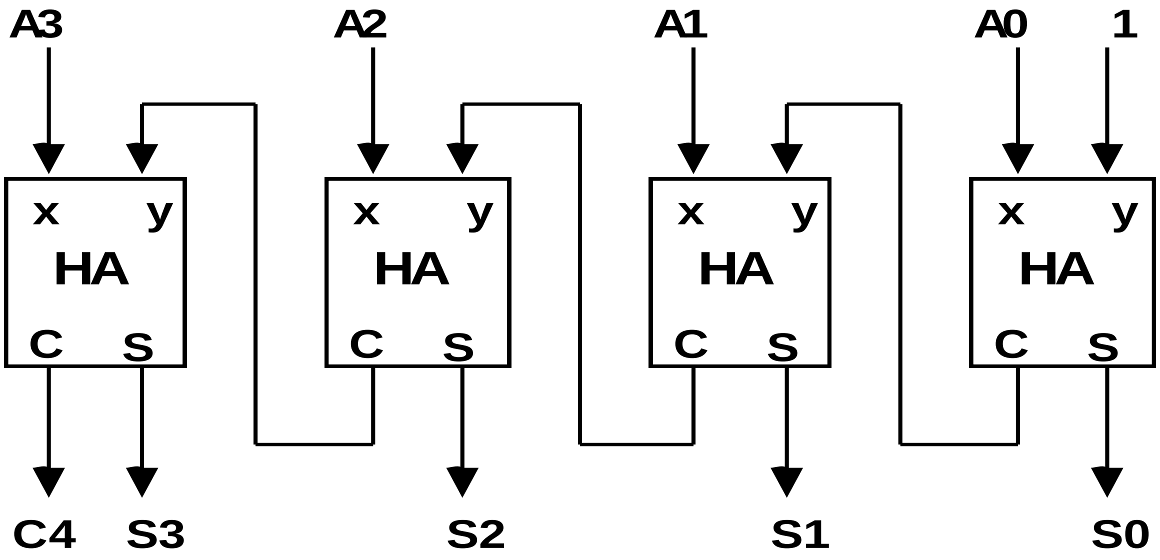
B) Half adder

C) Binary adder

D) All of the above

Binary Incrementer

Binary Incrementer


A3
A2
A1
A0
1
x
y
HA
C
S
x
y
HA
C
S
x
y
HA
C
S
x
y
HA
C
S
C4
S3
S2
S1
S0