Micro-Operation

- Computer system micro-operations are of four types:
  - Register transfer micro-operations
  - Arithmetic micro-operations
  - Logic micro-operations
  - Shift micro-operations

Arithmetic Micro-operations

Summary of Typical Arithmetic Micro-Operations

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

Logical Micro-operations

Boolean function	Microoperation	Name
$F_0 = 0$	$F \leftarrow 0$	Clear
$F_1 = xy$	$F \leftarrow A \wedge B$	AND
$F_2 = xy'$	$F \leftarrow A \wedge \overline{B}$	
$F_3 = x$	$F \leftarrow A$	Transfer A
$F_4 = x'y$	$F \leftarrow \overline{A} \wedge B$	
$F_5 = y$	$F \leftarrow B$	Transfer B
$F_6 = x \oplus y$	$F \leftarrow A \oplus B$	Exclusive-OR
$F_7 = x + y$	$F \leftarrow A \vee B$	OR
$F_8 = (x + y)'$	$F \leftarrow \overline{A \vee B}$	NOR
$F_9 = (x \oplus y)'$	$F \leftarrow \overline{A \oplus B}$	Exclusive-NOR
$F_{10} = y'$	$F \leftarrow \overline{B}$	Complement B
$F_{11} = x + y'$	$F \leftarrow A \vee \overline{B}$	
$F_{12} = x'$	$F \leftarrow \overline{A}$	Complement A
$F_{13} = x' + y$	$F \leftarrow \overline{A} \vee B$	
$F_{14} = (xy)'$	$F \leftarrow \overline{A \wedge B}$	NAND
$F_{15} = 1$	$F \leftarrow$ all 1's	Set to all 1's

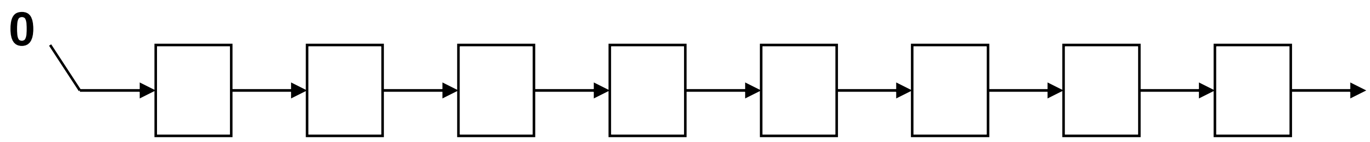
Shift Micro-operations

- $R \leftarrow shl\ R$ — Shift-left register R
- $R \leftarrow shr\ R$ — Shift-right register R
- $R \leftarrow cil\ R$ — Circular shift-left register R
- $R \leftarrow cir\ R$ — Circular shift-right register R
- $R \leftarrow ashl\ R$ — Arithmetic shift-left R
- $R \leftarrow ashr\ R$ — Arithmetic shift-right R

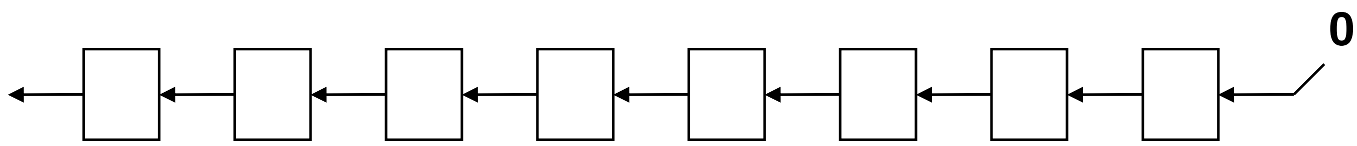
LOGICAL SHIFT

- In a logical shift the serial input to the shift is a 0.

- A right logical shift operation:



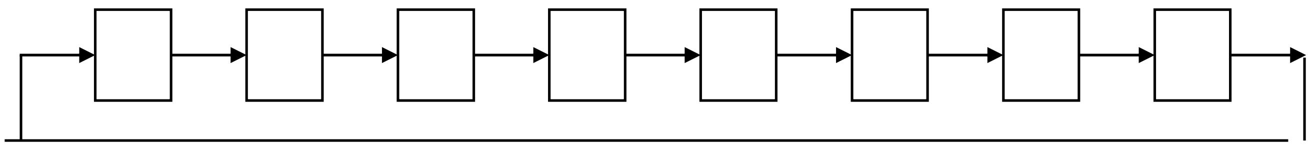
- A left logical shift operation:



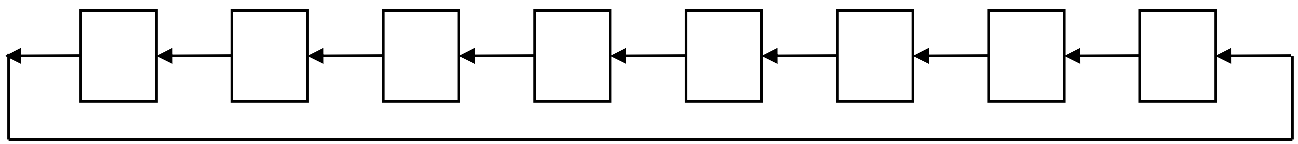
- In a Register Transfer Language, the following notation is used
  - *shl* for a logical shift left
  - *shr* for a logical shift right

CIRCULAR SHIFT

- In a circular shift the serial input is the bit that is shifted out of the other end of the register.
- A right circular shift operation:



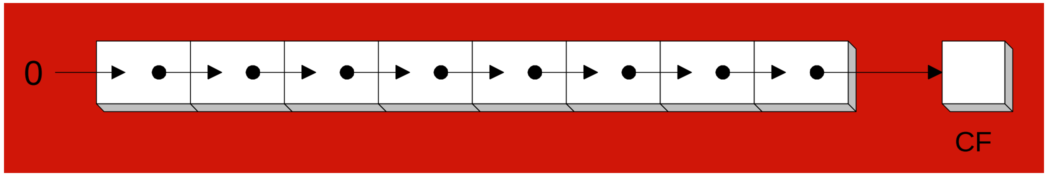
- A left circular shift operation:



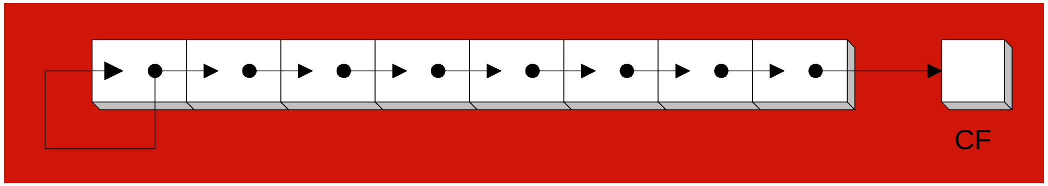
- In a RTL, the following notation is used
  - *cil* for a circular shift left
  - *cir* for a circular shift right

Logical versus Arithmetic Shift

- A logical shift fills the newly created bit position with zero:



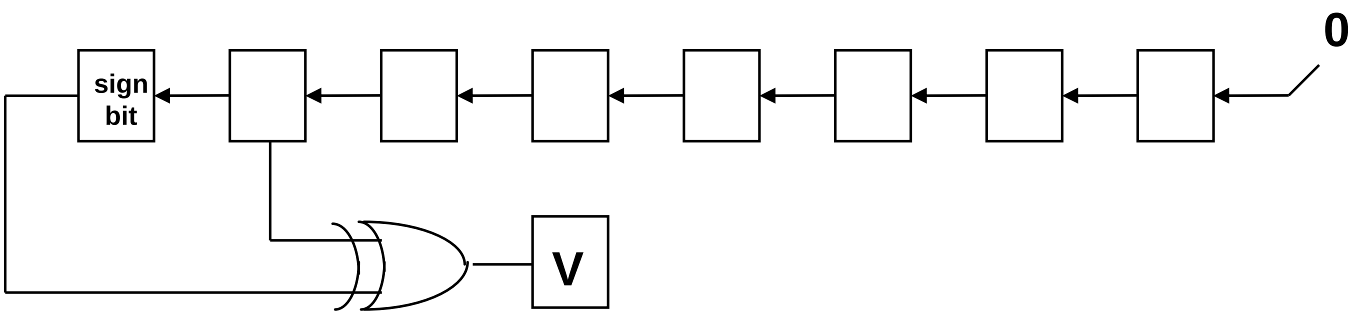
- An arithmetic shift fills the newly created bit position with a copy of the number's sign bit:



Arithmetic Right Shift Process

ARITHMETIC SHIFT

- An left arithmetic shift operation must be checked for the overflow



Before the shift, if the leftmost two bits differ, the shift will result in an overflow

- **In a RTL, the following notation is used**
  - ***ashl*** **for an arithmetic shift left**
  - ***ashr*** **for an arithmetic shift right**

APPLICATIONS OF LOGIC MICROOPERATIONS

- Logic micro-operations can be used to manipulate individual bits or a portions of a word in a register.
- Consider the data in a register A. In another register B is bit data that will be used to modify the contents of A.

SELECTIVE SET

- In a selective set operation, the bit pattern in B is used to *set* certain bits in A.

1 1 0 0 A			
1 0 1 0 B	$(A \leftarrow A + B)$	OR Operation
1 1 1 0 A			

SELECTIVE COMPLEMENT

- In a selective complement operation, the bit pattern in B is used to *complement* certain bits in A.

$$
\begin{array}{ll}
1\,1\,0\,0 & A \\
1\,0\,1\,0 & B \\
\hline
0\,1\,1\,0 & A
\end{array}
\qquad (A \leftarrow A \oplus B)
$$

XOR Operation

SELECTIVE CLEAR

- In a selective clear operation, the bit pattern in B is used to *clear* certain bits in A.

$$\begin{array}{ll} 1\,1\,0\,0 & A \\ 1\,0\,1\,0 & B \\ \hline 0\,1\,0\,0 & A \end{array} \qquad (A \leftarrow A \cdot B')$$

MASK OPERATION

- The mask operation is similar to selective-clear operation except that the bits of A are cleared only where there are corresponding 0's in B.

$$
\begin{array}{ll}
1\,1\,0\,0 & A \\
1\,0\,1\,0 & B \\
\hline
1\,0\,0\,0 & A
\end{array}
\qquad (A \leftarrow A \cdot B) \qquad \text{AND Operation}
$$

CLEAR OPERATION

- The clear operation compares the words in A & B and produces an all 0's results if the two numbers are equal.

$$\begin{array}{rl} 1\,0\,1\,0 & A \\ 1\,0\,1\,0 & B \\ \hline 0\,0\,0\,0 & A \end{array} \qquad (A \leftarrow A \oplus B)$$

XOR operation

INSERT OPERATION

- An insert operation inserts a new value into a group of bits. This is done by first masking the bits and then ORing them with the required value.
- Example: A register contains eight bits 0110 1010. To replace the four leftmost bits by the value 1001, we first mask the four unwanted bits.

```
    0110 1010          A before
    0000 1111          B (mask)
  ─────────────
    0000 1010          A after masking  AND Operation
```

And then insert the new value

```
    0000 1010          A before
    1001 0000          B (insert)
  ─────────────
    1001 1010          A after insertion OR Operation
```

Question

- Register A holds the 8-bit binary 11011001. Determine the B operation and the logic microoperation to be performed in order to change the value in A to : 01101101, 11111101