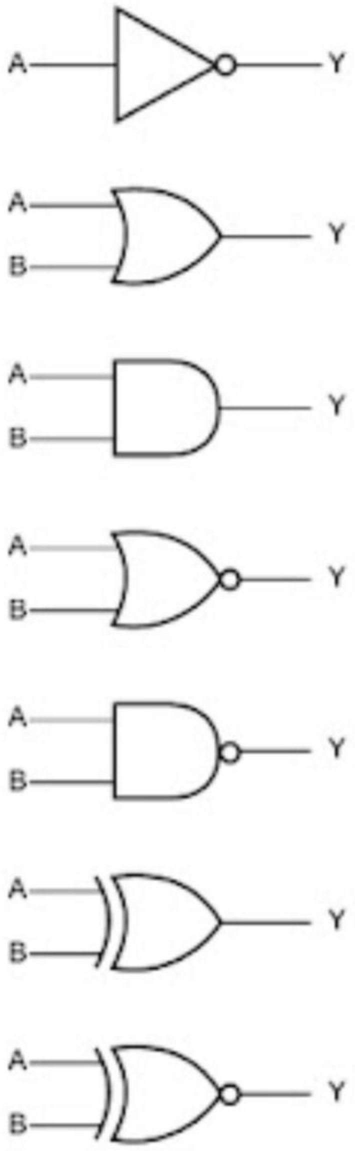
$$\begin{tikzcd} \text{เลขฐานสิบหก} \arrow[r, leftrightarrow] & \text{เลขฐานสอง} \arrow[r, leftrightarrow] \arrow[d, leftrightarrow] & \text{เลขฐานสิบ} \arrow[d, leftrightarrow] \\ & \text{เลขฐานแปด} & \text{เลข BCD} \end{tikzcd}$$

Output

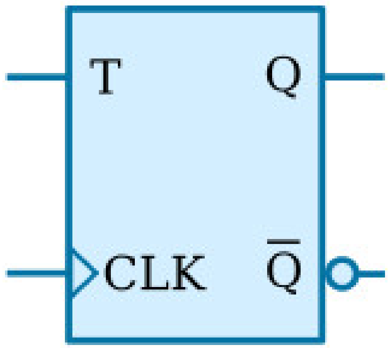
Inputs

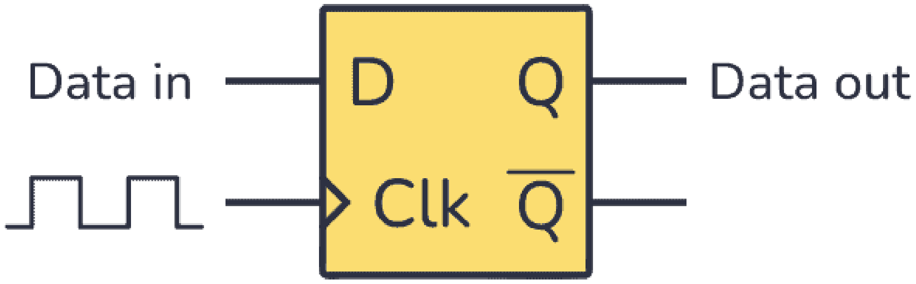
A	B	x
0	0	1
0	1	0
1	0	1
1	1	0

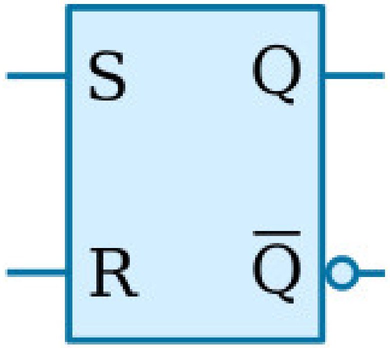

A
Y
A
B
Y
A
B
Y
A
B
Y
A
B
Y
A
B
Y
A
B
Y

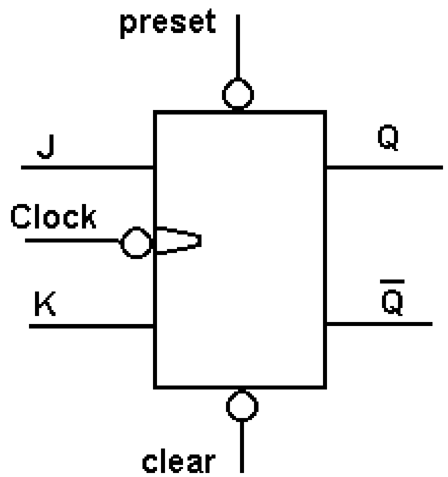
Name	AND form	OR form
Identity law	$1A = A$	$0 + A = A$
Null law	$0A = 0$	$1 + A = 1$
Idempotent law	$AA = A$	$A + A = A$
Inverse law	$A\overline{A} = 0$	$A + \overline{A} = 1$
Commutative law	$AB = BA$	$A + B = B + A$
Associative law	$(AB)C = A(BC)$	$(A + B) + C = A + (B + C)$
Distributive law	$A + BC = (A + B)(A + C)$	$A(B + C) = AB + AC$
Absorption law	$A(A + B) = A$	$A + AB = A$
De Morgan's law	$\overline{AB} = \overline{A} + \overline{B}$	$\overline{A + B} = \overline{A}\,\overline{B}$

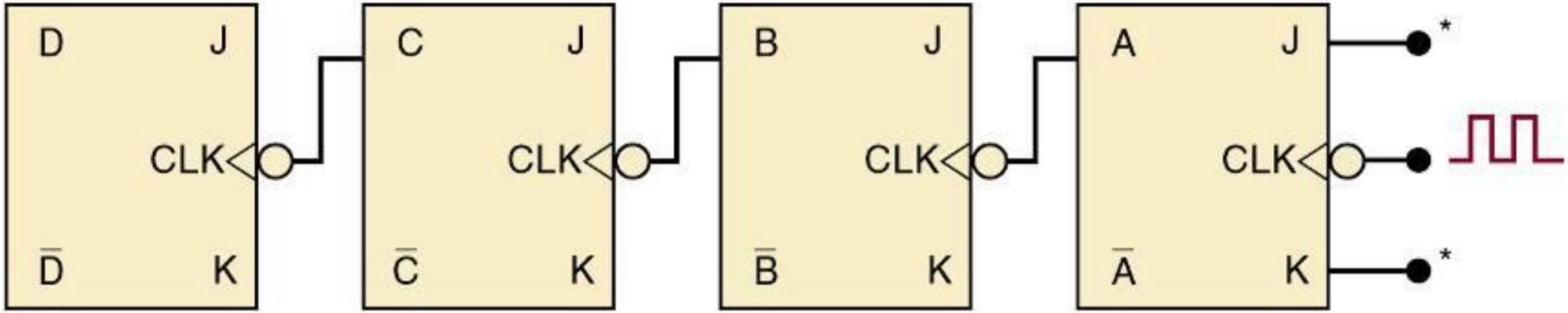
AB \ CD	00	01	11	10
00	0: X	1: X	3: 0	2: X
01	4: 1	5: 1	7: 1	6: 0
11	12: 0	13: 1	15: 0	14: 1
10	8: 1	9: 0	11: 1	10: 1


T
Q
CLK
$\overline{Q}$

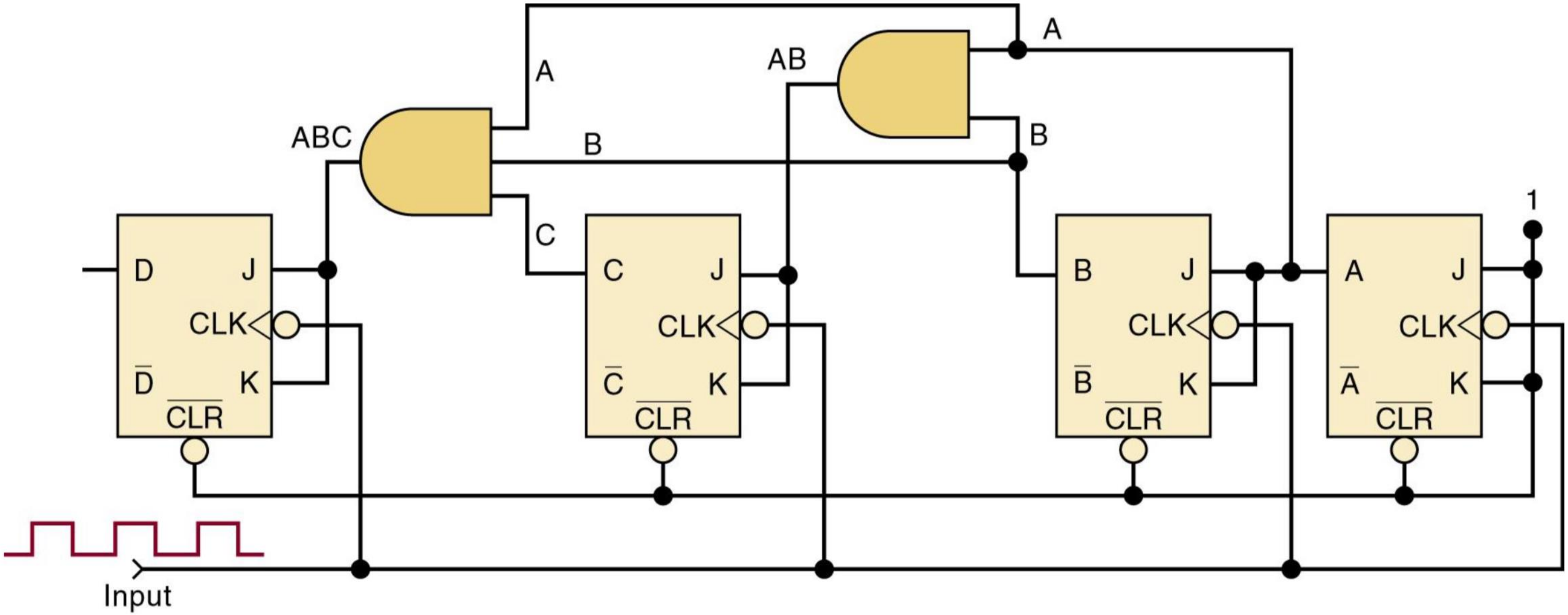

Data in
D
Q
Data out
Clk
$\overline{Q}$


S
Q
R
$\overline{Q}$


preset
J
Q
Clock
K
$\overline{Q}$
clear



*All J and K inputs assumed to be 1.


A
A
AB
ABC
B
B
C
1
D
J
CLK
$\overline{D}$
K
$\overline{CLR}$
C
J
CLK
$\overline{C}$
K
$\overline{CLR}$
B
J
CLK
$\overline{B}$
K
$\overline{CLR}$
A
J
CLK
$\overline{A}$
K
$\overline{CLR}$
Input

Before Clock	*After Clock*	*Before Clock*	
Q	Q	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

P.S.			N.S.		
Q_C	Q_B	Q_A	Q_C	Q_B	Q_A
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

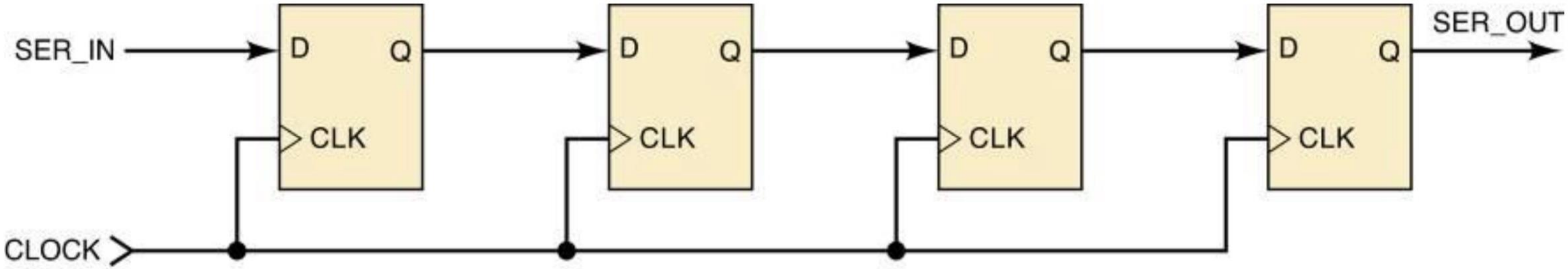
P.S.			N.S.			C		B		A	
Q_C	Q_B	Q_A	Q_C	Q_B	Q_A	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	1	1	1	X	0	X	0	1	X
1	1	1	0	0	0	X	1	X	1	X	1

J_C

C \ BA	00	01	11	10
0	0	0	1	0
1	X	X	X	X

K_C

C \ BA	00	01	11	10
0	X	X	X	X
1	0	0	1	0


SER_IN
D
Q
CLK
D
Q
CLK
D
Q
CLK
D
Q
CLK
SER_OUT
CLOCK

ตัวตั้ง		ตัวบวก	ผลบวก	ตัวทด
0	+	0	0	0
0	+	1	1	0
1	+	0	1	0
1	+	1	0	1

ตัวทด	1	1	1					ตรวจสอบ
ตัวตั้ง		1	1	1	1	0		30_{10}
							+	+
ตัวบวก			1	1	0	0		12_{10}
ผลรวม	1	0	1	0	1	0		42_{10}

$$\begin{array}{rr} & A \\ + & B \\ \hline Co & Sum \end{array}$$

Input		Output	
A (ตัวตั้ง)	B (ตัวบวก)	C_O (ตัวทดออก)	Sum (ผลรวม)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$\begin{array}{lr} & C_{in} \\ & A \\ + & B \\ \hline Co & Sum \end{array}$$

Input			Output	
C_{in}	A	B	C_o	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

ตัวตั้ง		ตัวลบ	ผลต่าง	ตัวยืม
0	-	0	0	0
0	-	1	1	1
1	-	0	1	0
1	-	1	0	0

ตัวอย่าง จงลบ 1001_2 จาก 10011_2

						ตรวจสอบ
ตัวยืม		10				
ตัวตั้ง	~~1~~	0	0	1	1	19_{10}
					-	-
ตัวลบ		1	0	0	1	9_{10}
ผลต่าง	0	1	0	1	0	10_{10}

$$\begin{array}{rr} & A \\ + & B \\ \hline Bo & Sum \end{array}$$

Input		Output	
A (ตัวตั้ง)	B (ตัวลบ)	B_O (ตัวยืมออก)	Sum (ผลต่าง)
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

Bor_{out} Bor_{in}

$$\begin{array}{r} A \\ -\ B \\ \hline \text{Diff} \end{array}$$

Input			Output	
Bor_i	A	B	Bor_o	Diff
0	0	0	0	0
0	0	1	1	1
0	1	0	0	1
0	1	1	0	0
1	0	0	1	1
1	0	1	1	0
1	1	0	0	0
1	1	1	1	1

1’s complement ของเลขฐานสอง เป็นการนำเลขฐานสองนั้นมาทำการกลับค่าของแต่ละบิตให้มีค่าตรงกันข้าม (0 เป็น 1 และ 1 เป็น 0) บิตต่อบิต เช่น

1’s complement	**ของ**	**1011101**
	คือ	**0100010**

ขั้นตอนการลบเลขฐานสองโดยวิธี 1’s complement

1. เปลี่ยนตัวลบให้เป็น 1’s complement ของมัน

2. นำตัวตั้งมาบวกกับ 1’s complement ของตัวลบที่ได้จาก 1. แล้วพิจารณาผลลัพธ์ที่ได้

2.1 ถ้าเกิดตัวทดจากการบวกหลักซ้ายสุดแสดงว่าคำตอบมีค่าเป็นบวก เราหาคำตอบได้โดยนำบิตตัวทดที่เกิดขึ้นไปบวกกับผลที่ได้จากการบวกในตอนแรก (End-around carry / EAC)

2.2 ถ้าไม่มีตัวทดเกิดขึ้นจากการบวกหลักซ้ายสุดแสดงว่าคำตอบมีค่าเป็นลบ (ติดลบ) เราหาคำตอบได้โดยนำผลที่ได้ไปแปลงเป็น 1’s complement อีกครั้ง

จงแสดงวิธีลบเลข $11001_2 - 10001_2$ ด้วยวิธี 1's complement

$$
\begin{array}{r}
1\;1\;0\;0\;1 \\
1\;0\;0\;0\;1\;- \\
\hline
\end{array}
\quad \Rightarrow \quad
\begin{array}{r}
1\;1\;0\;0\;1 \\
0\;1\;1\;1\;0\;+ \\
\hline
1\quad 0\;0\;1\;1\;1
\end{array}
$$

Overflow

EAC

$$
\begin{array}{r}
0\;0\;1\;1\;1 \\
1\;+ \\
\hline
+\;1\;0\;0\;0 \\
\hline
\end{array}
$$

ตรวจคำตอบ

25 – 17 = 8

จงแสดงวิธีลบเลข $101_2 - 11000_2$ ด้วยวิธี 1’s complement

$$
\begin{array}{r}
1\ 0\ 1 \\
1\ 1\ 0\ 0\ 0 \; - \\
\hline
\end{array}
\qquad
\begin{array}{r}
1\ 0\ 1 \\
0\ 0\ 1\ 1\ 1 \; + \\
\hline
0\ 1\ 1\ 0\ 0
\end{array}
$$

No Overflow

1’s complement

$$
\begin{array}{r}
-1\ 0\ 0\ 1\ 1 \\
\hline
\end{array}
$$

ตรวจคำตอบ

5 – 24 = -19

2’s complement ของเลขฐานสอง เป็นการนำเลขฐานสองนั้นมาทำ 1’s complement แล้วบวกผลที่ได้จากการทำ 1’s complement ด้วย 1

2’s complement ของ 1011101

คือ 1’s complement ของ 1011101	**=**	**0100010**
บวกด้วย 1 ซึ่งเท่ากับ		**0100011**

ขั้นตอนการลบเลขฐานสองโดยวิธี 2’s complement

1 เปลี่ยนตัวลบให้เป็น 2’s complement ของมัน

2. นำตัวตั้งมาบวกกับ 2’s complement ของตัวลบที่ได้จาก 1.

พิจารณาผลลัพธ์ที่ได้

I. ถ้าเกิดตัวทดจากการบวกแสดงว่าคำตอบมีค่าเป็นบวก เราหาคำตอบได้โดยนำผลบวกที่ได้มาเป็นคำตอบ บิตตัวทดที่เกิดขึ้นให้ทิ้งไป

II. ถ้าผลที่ได้ไม่มีตัวทดแสดงว่าคำตอบมีค่าเป็นลบเป็นลบ (ติดลบ) หาคำตอบได้โดยนำผลบวกที่ได้ไปแปลงเป็น 2’s complement อีกครั้ง

จงแสดงวิธีลบเลข $1011_2 - 100_2$ ด้วยวิธี 2's complement

$$\begin{array}{r} 1\;0\;1\;1 \\ 1\;0\;0^{-} \\ \hline \end{array} \longrightarrow \begin{array}{r} 1\;0\;1\;1 \\ 1\;1\;0\;0^{+} \\ \hline 1\quad 0\;1\;1\;1 \end{array}$$

Overflow

Ignore the Overflow

$$\begin{array}{r} +\;0\;1\;1\;1 \\ \hline \end{array}$$

1's complement = 1 0 1 1

2's complement = 1 1 0 0

ตรวจคำตอบ

11 – 4 = 7

จงแสดงวิธีลบเลข $10010_2 - 11000_2$ ด้วยวิธี 2's complement

$$\begin{array}{r} 1\ 0\ 0\ 1\ 0 \\ -\ \ 1\ 1\ 0\ 0\ 0 \\ \hline \end{array} \longrightarrow \begin{array}{r} 1\ 0\ 0\ 1\ 0 \\ +\ \ 0\ 1\ 0\ 0\ 0 \\ \hline 1\ 1\ 0\ 1\ 0 \end{array}$$

2's complement

No Overflow

1's complement = 0 0 1 1 1

2's complement = 0 1 0 0 0

$$\begin{array}{r} 0\ 0\ 1\ 0\ 1 \\ +\ \ 1 \\ \hline -\ \ \ 1\ 1\ 0 \\ \hline \end{array}$$

ตรวจคำตอบ

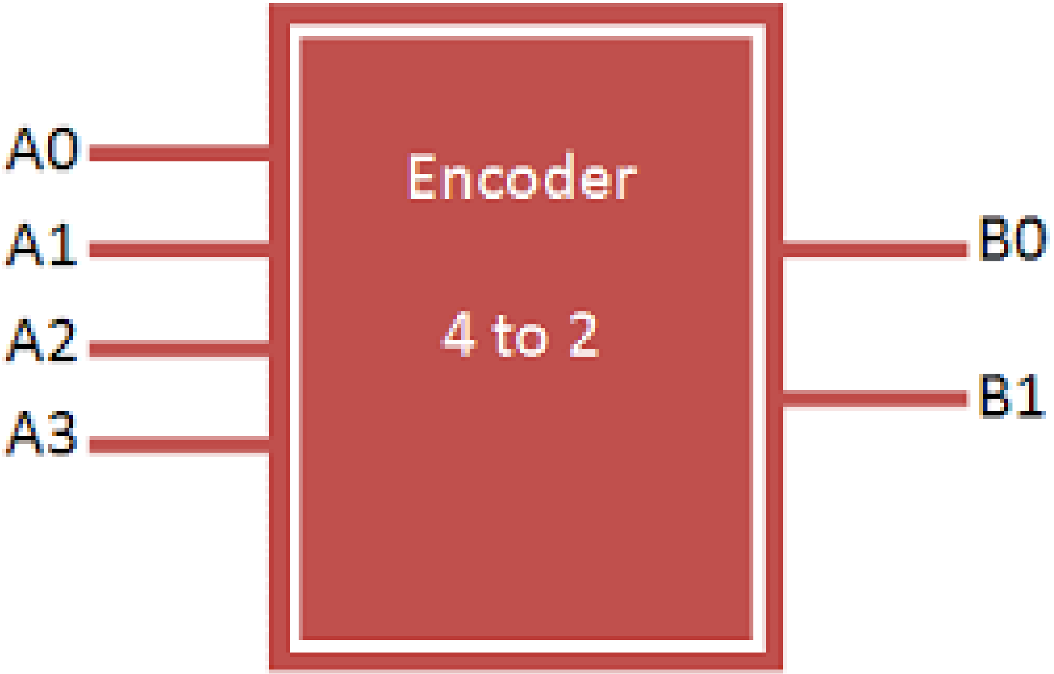
18 – 24 = -6

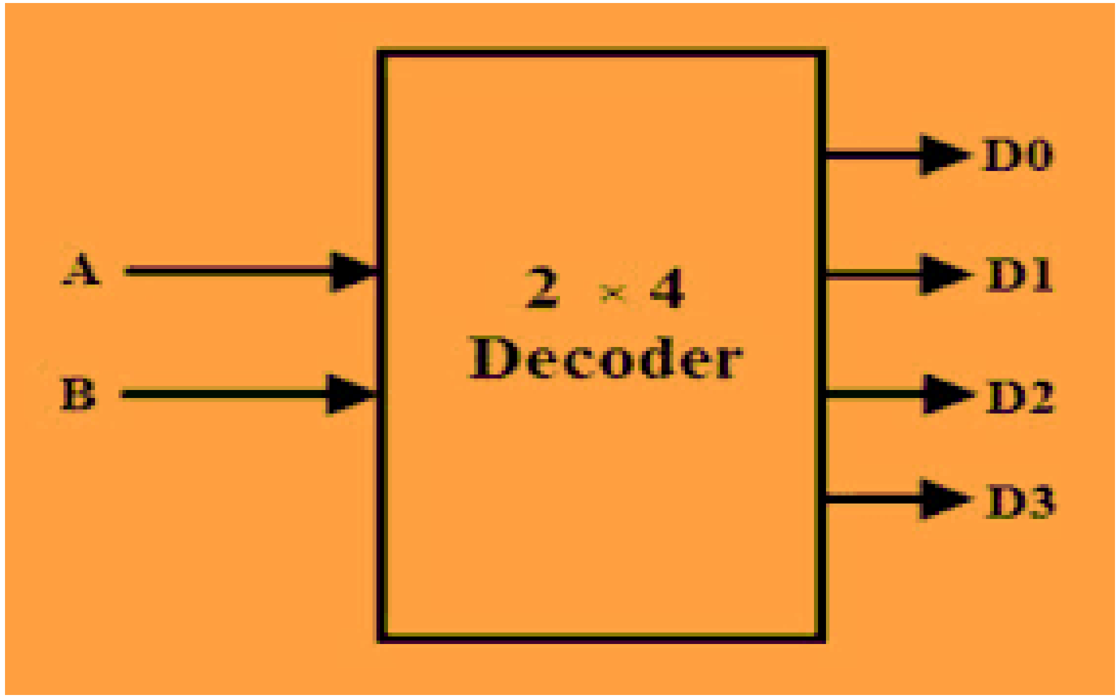
การทำ 2's Complement วิธีลัด

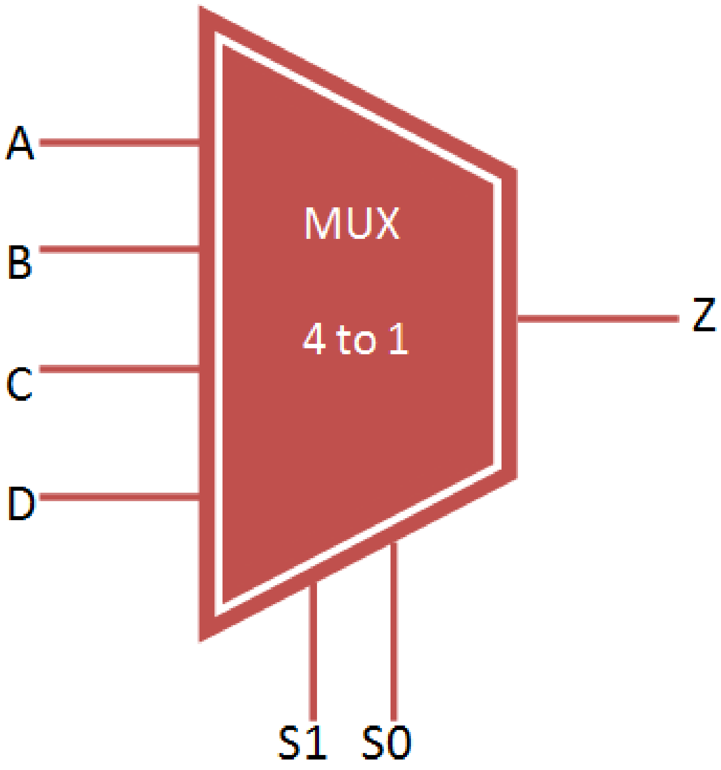
110001110

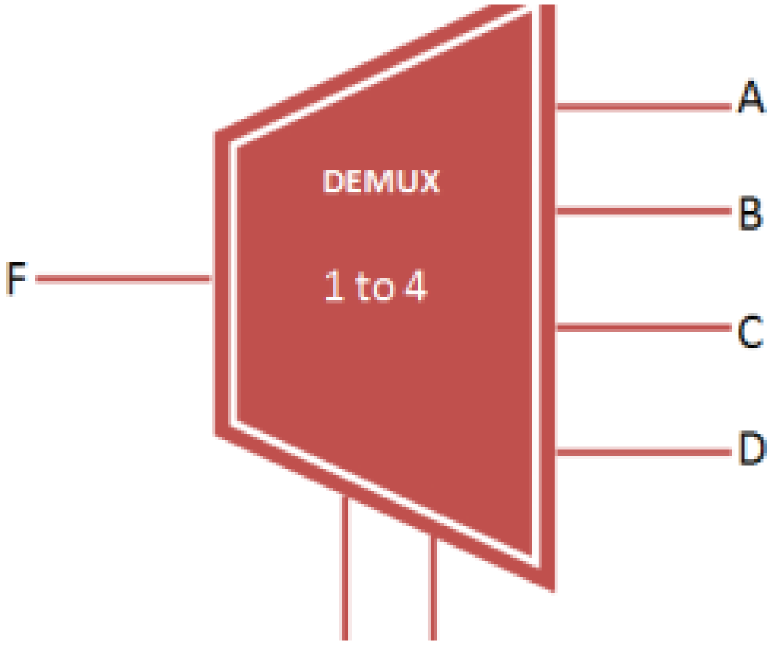
10110001

111000100


Encoder
4 to 2
A0
A1
A2
A3
B0
B1


2 × 4
Decoder
A
B
D0
D1
D2
D3


A
B
C
D
MUX
4 to 1
Z
S1
S0


F
DEMUX
1 to 4
A
B
C
D

ANYTHING ELSE?