Digital Electronic Circuits
Section 1 (EE, IE)

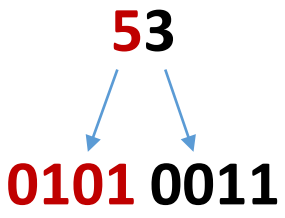
Lecture 17

Class Test 2:
27-10-2020 (TUE): Starting 10:15 AM (as was suggested by students, this is additional holiday after Dusserah as per institute holiday list)

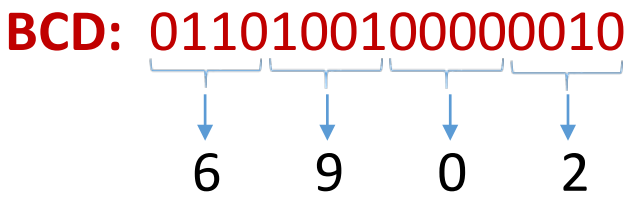
Syllabus: Logic families (not covered in CT1) and primarily post CT1, shall include concept dealt in pre-CT1 part which forms the pre-requisite.

Binary Coded Decimal (BCD)

0: 0000
1: 0001
2: 0010
3: 0011
4: 0100
5: 0101
6: 0110
7: 0111
8: 1000
9: 1001



Decimal	BCD
10	0001 0000
11	0001 0001
99	1001 1001
100	0001 0000 0000
101	0001 0000 0001
487	0100 1000 0111
…	…



Addition of BCD

- Result is valid BCD when 4-bit sum is less than or equal to 9.

```
0101        0011
0010        0110
------      ------
0111        1001
  ⇩           ⇩
  7           9
```

Binary addition

$A_3A_2A_1A_0$
$B_3B_2B_1B_0$

C_3 $S_3S_2S_1S_0$

- Result is not valid when 4-bit sum is more than 9. Addition of 6 gives valid BCD.

```
0100        1011        1001        (000)1 0001
0111        0110        1000              0110
------      ------      ------      ------
1011        (000)1 0001 (000)1 0001 (000)1 0111
  ⇩           ⇩  ⇩        ⇩  ⇩        ⇩  ⇩
Not valid BCD 1  1        1  1        1  7
```

Logic for Addition of 6

Decimal result	C_3	S_3	S_2	S_1	S_0	Y (Add 6)
0 - 9	--	--	--	--	--	0
10	0	1	0	1	0	1
11	0	1	0	1	1	1
12	0	1	1	0	0	1
13	0	1	1	0	1	1
14	0	1	1	1	0	1
15	0	1	1	1	1	1
16	1	0	0	0	0	1
17	1	0	0	0	1	1
18	1	0	0	1	0	1

BCD Addition result is between 0 and 18.

Y = 1 when Addition of 6 is needed
Y = 1 when addition result is between 10 and 18.
i.e. $C_3S_3S_2S_1S_0$: 01010 to 10010

Y = 0 when $C_3S_3S_2S_1S_0$: 00000 to 01001
Y = X (don't care) when $C_3S_3S_2S_1S_0 > 10010$

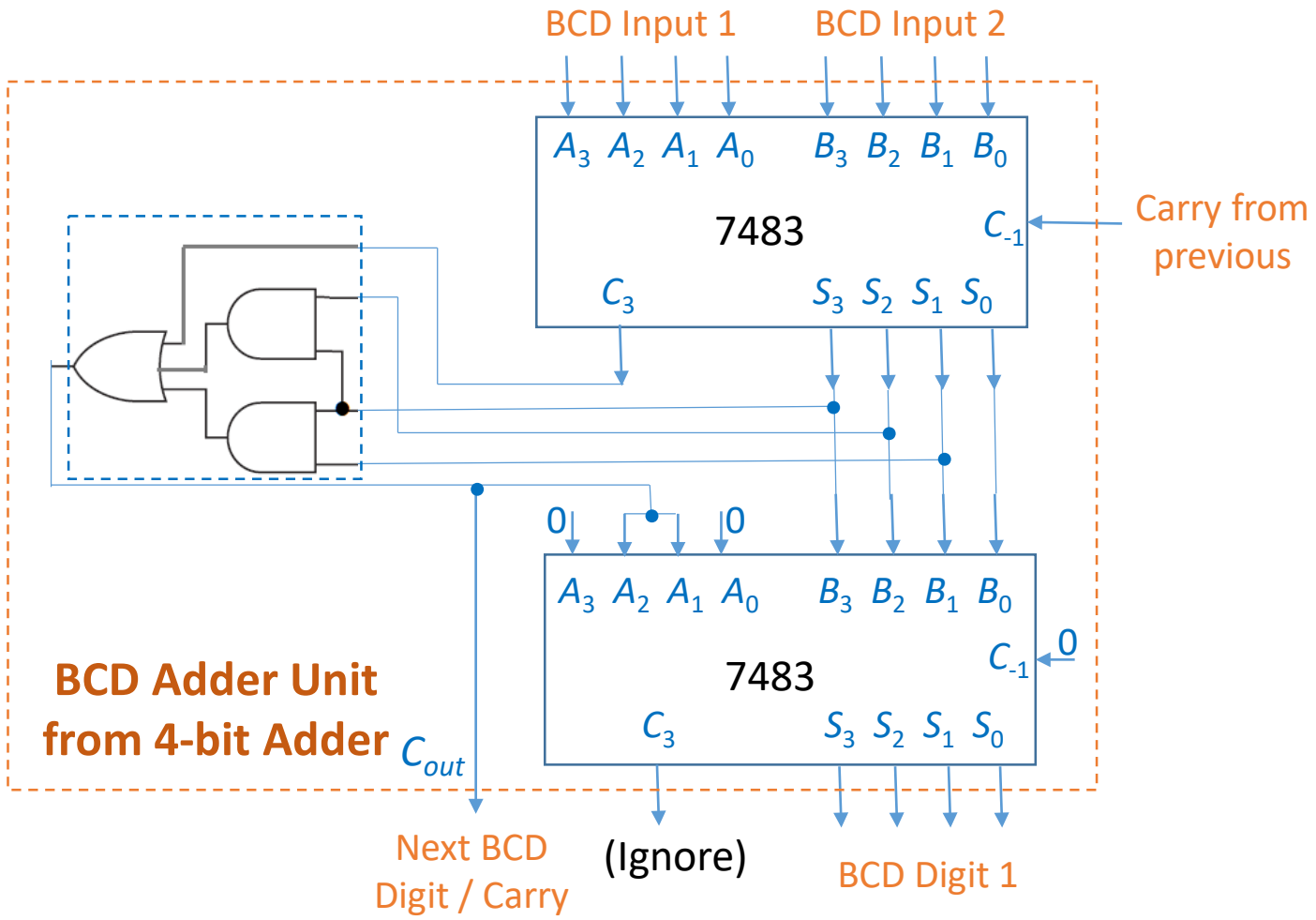
This Truth Table on simplification gives

$$Y = C_3 + S_3S_2 + S_3S_1$$

BCD Adder Unit

49	0100	1001
+ 58	0101	1000

	1	0001
		0110
	------	------
	1010	0111
	0110	
	-------	------
(000)1	0000	0111
	⇩	⇩ ⇩
	1	0 7

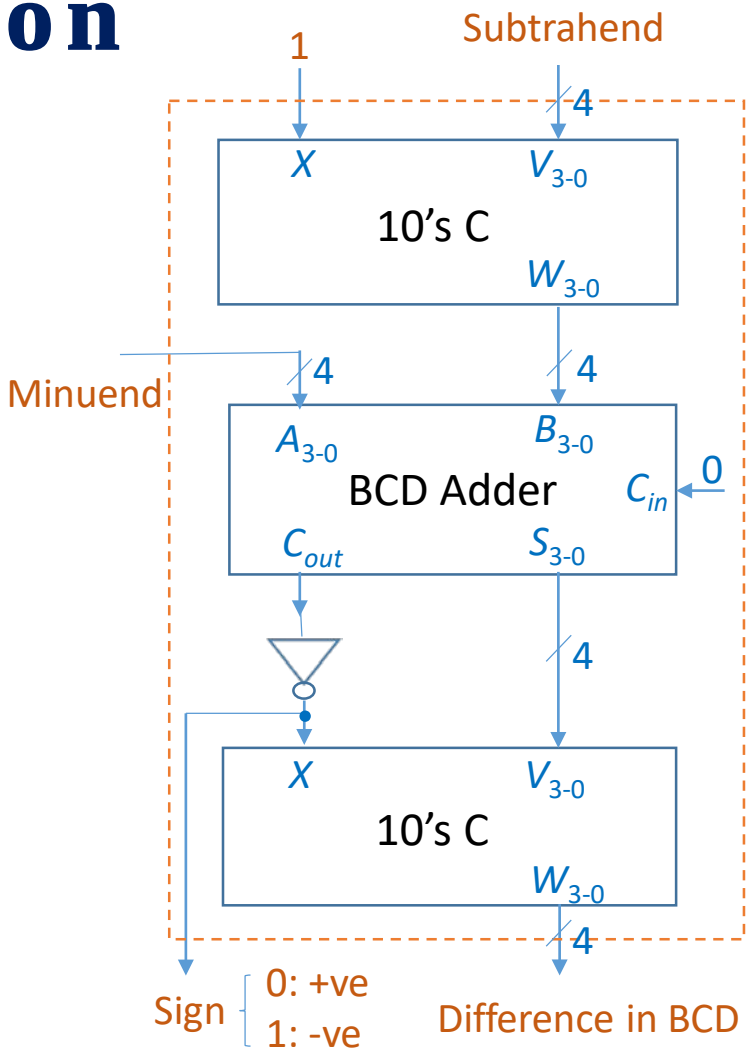


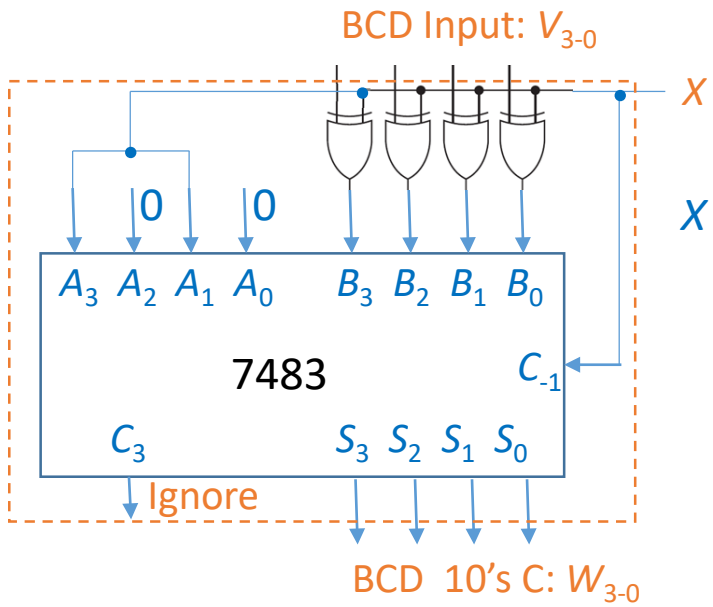
BCD Subtraction

Subtraction using 10's C: Decimal
(Similar to 2's C subtraction in binary)

Smaller from larger: 7 – 4
10's C of 4 = 10 – 4 = 6
7 + 6 = 13
Carry present: Result +ve
Ignore carry
Difference = 3 (+ve)

Larger from smaller: 4 – 7
10's C of 7 = 10 – 7 = 3
4 + 3 = 7
Carry absent: Result -ve
Difference = 10's C of 7 (-ve)
= 10 – 7 = 3 (-ve)





X =1 if 10's C
else, 0

10's C Generator Unit

BCD Subtractor Unit

9's Complement and 1's Complement

Decimal Subtraction using 9's C:

Smaller from larger: 7 – 4
9's C of 4 = 9 – 4 = 5
7 + 5 = 12
Carry present: Result +ve
Add carry to get result
Difference = 2 + 1 = 3 (+ve)

Larger from smaller: 4 – 7
9's C of 7 = 9 – 7 = 2
4 + 2 = 6
Carry absent: Result -ve
Difference = 9's C of 6 (-ve)
= 9 – 6 = 3 (-ve)

Binary Subtraction using 1's C:

Smaller from larger: 0111 – 0100
1's C of 0100 = 1011
0111 + 1011 = 10010
Carry present: Result +ve
Add carry to get result
Difference = 0010 + 1 = 0011 (+ve)

Larger from smaller: 0100 – 0111
1's C of 0111 = 1000
0100 + 1000 = 1100
Carry absent: Result -ve
Difference = 1's C of 1100 (-ve)
= 0011 (-ve)

Subtraction Revisited

0000 :	0	1000 :	-8
0001 :	1	1001 :	-7
0010 :	2	1010 :	-6
0011 :	3	1011 :	-5
0100 :	4	1100 :	-4
0101 :	5	1101 :	-3
0110 :	6	1110 :	-2
0111 :	7	1111 :	-1

Range: -8 to +7

$A_3A_2A_1A_0$
$B_3B_2B_1B_0$

$[C_3]$ $S_3S_2S_1S_0$

0111 (7)
1110 (-2)

$\pm$ 0101 (5)

0011 (3)
1101 (-3)

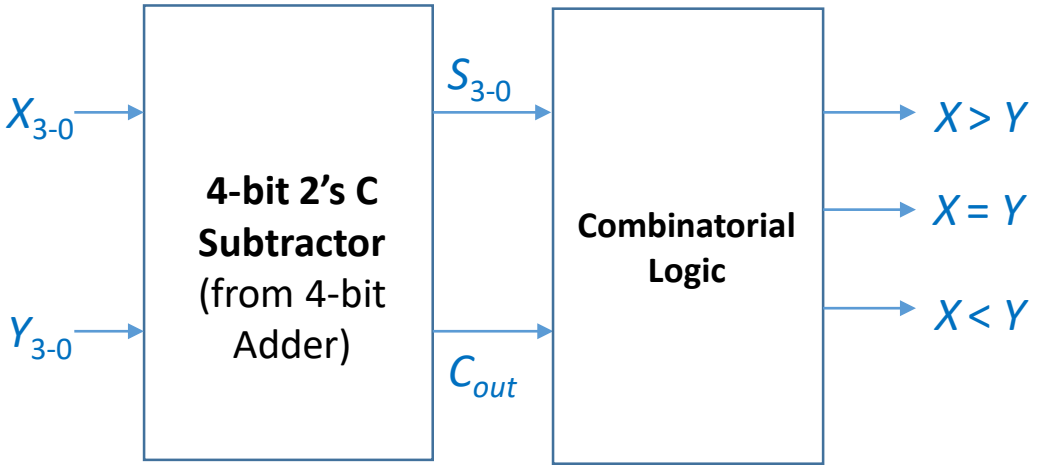
$\pm$ 0000 (0)

0010 (2)
1001 (-7)

1011 (-5)

Magnitude Comparison by Subtraction

- Magnitude comparison finds if (i) one number is same as the other number (identity / equality) or (ii) greater / smaller than the other number.
- An adder circuit performing 2's C subtraction ($X - Y$) can be used for this.
  - If $X = Y$ and $X > Y$ then $C_{out} = 1$.
    If $C_{out} = 0$, then $X < Y$.
  - if C_{out} is 1 and any of $S_i = 1$ then $X > Y$, else $X = Y$.

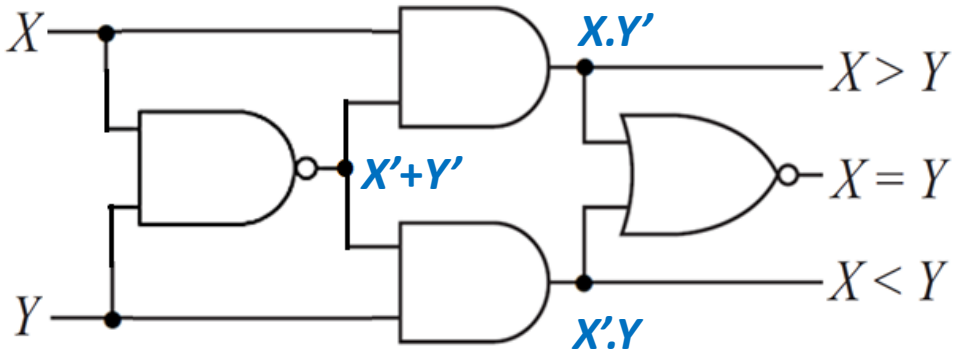


$(X < Y) = C_{out}'$ $(X = Y) = C_{out}.S_3'.S_2'.S_1'.S_0'$ $(X > Y) = C_{out}.(S_3+S_2+S_1+S_0)$

1-bit Magnitude Comparison

- Magnitude comparison can be done without generating addition / subtraction result.

Input		Output		
X	*Y*	*X* > *Y*	*X* = *Y*	*X* < *Y*
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0



$(X > Y)$: $G = X.Y'$

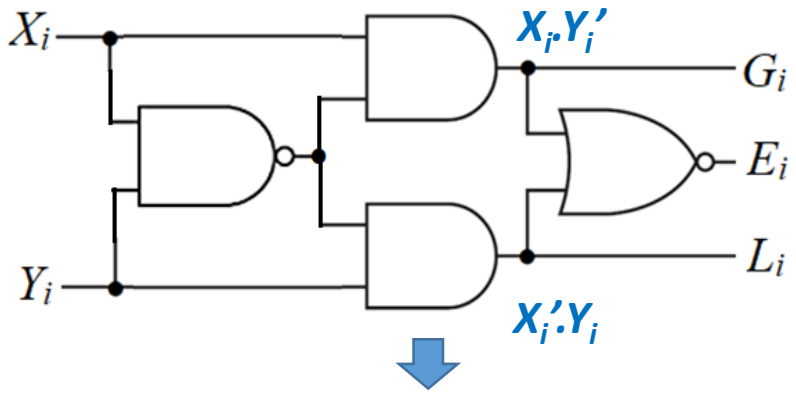
$(X = Y)$: $E = X'.Y' + X.Y$

$(X < Y)$: $L = X'.Y$

(If NAND gate is not used, two NOT gates will be required to get *X'* and *Y'*)

$$X'.Y' + X.Y = (X' + Y).(X + Y')$$
$$= (X.Y')'.(X'.Y)'$$
$$= [(X.Y') + (X'.Y)]'$$

2-bit Magnitude Comparison



Numbers to be compared: $X: X_1X_0$ and $Y: Y_1Y_0$

Define:

$G_0 = X_0.Y_0'$
$E_0 = X_0'.Y_0' + X_0.Y_0$
$L_0 = X_0'.Y_0$

$G_1 = X_1.Y_1'$
$E_1 = X_1'.Y_1' + X_1.Y_1$
$L_1 = X_1'.Y_1$

$X > Y$:

- If $X_1 > Y_1$
- If $X_1 = Y_1$ and $X_0 > Y_0$

$(X > Y) = G_1 + E_1G_0$

$X = Y$:

- If $X_1 = Y_1$ and $X_0 = Y_0$

$(X = Y) = E_1.E_0$

$X < Y$:

- If $X_1 < Y_1$
- If $X_1 = Y_1$ and $X_0 < Y_0$

$(X < Y) = L_1 + E_1L_0$

Other than two 1-bit comparators, a logic circuit to generate final outputs.

n-bit Magnitude Comparison

Numbers to be compared: $X : X_{n-1}X_{n-2} \ldots X_1X_0$ and $Y : Y_{n-1}Y_{n-2} \ldots Y_1Y_0$

$$(X > Y) = G_{n-1} + E_{n-1}G_{n-2} + E_{n-1}E_{n-2}G_{n-3} + \ldots + E_{n-1}E_{n-2} \ldots E_1G_0$$

$$(X = Y) = E_{n-1}E_{n-2} \ldots E_1E_0$$

$$(X < Y) = L_{n-1} + E_{n-1}L_{n-2} + E_{n-1}E_{n-2}L_{n-3} + \ldots + E_{n-1}E_{n-2} \ldots E_1L_0$$

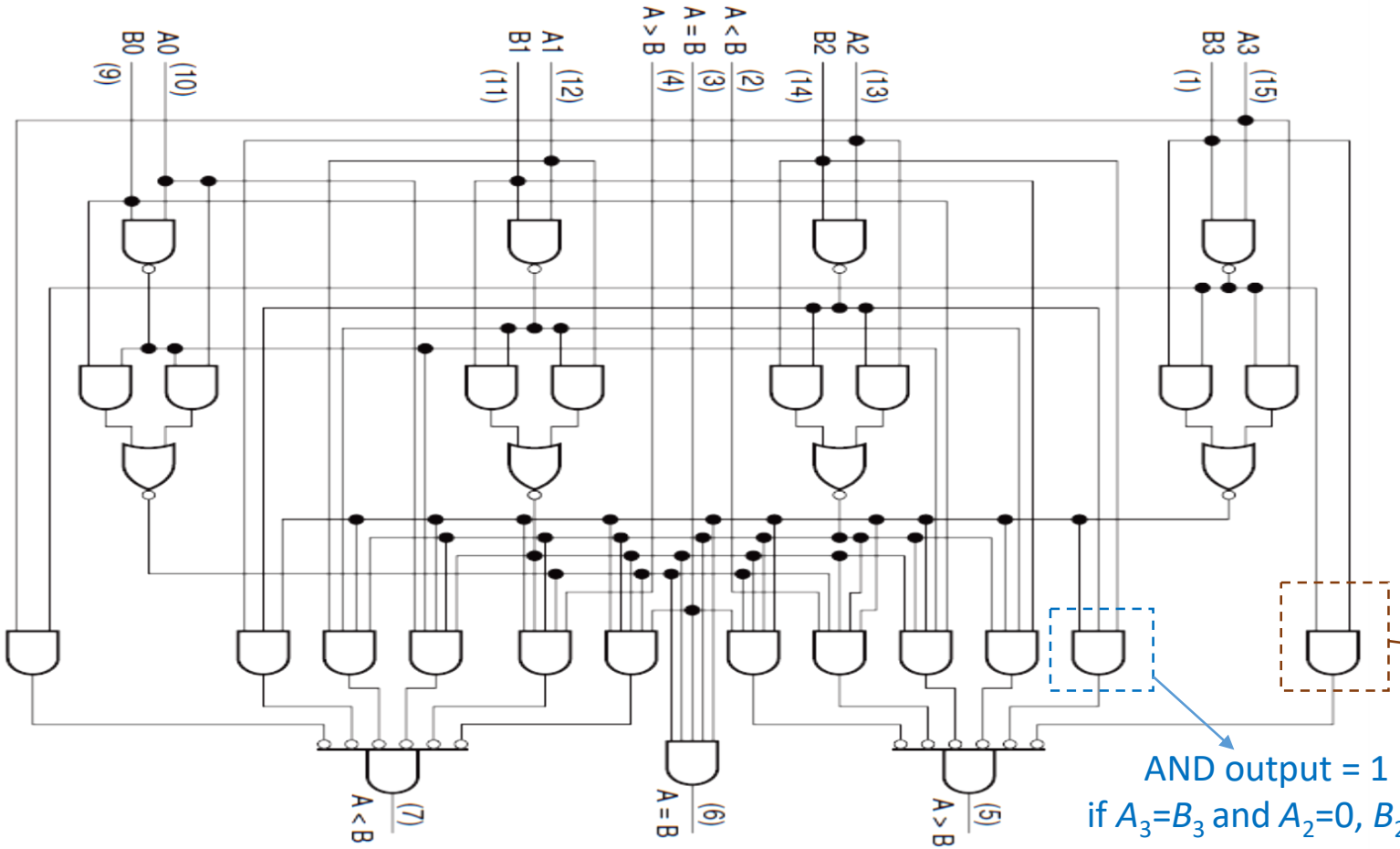
With cascading inputs coming from another block with lower significant bits:

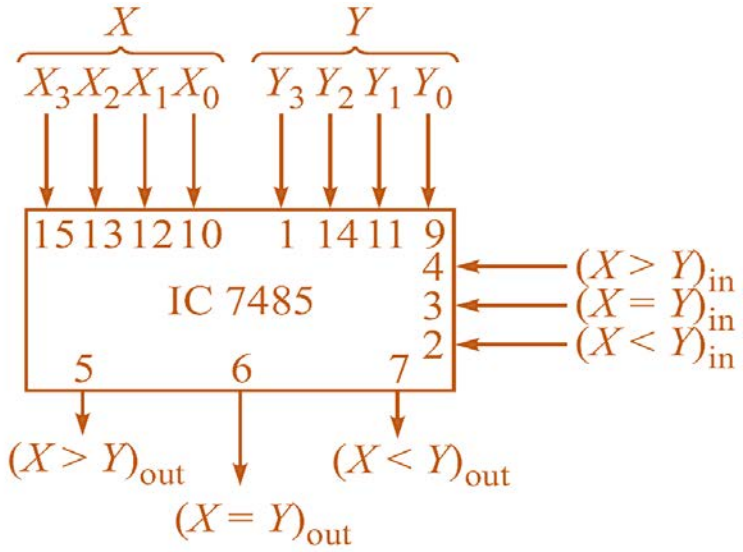
$$(X > Y)_{out} = G_{n-1} + E_{n-1}G_{n-2} + \ldots + E_{n-1}E_{n-2} \ldots E_1G_0 + E_{n-1}E_{n-2} \ldots E_0.(X > Y)_{in}$$

$$(X = Y)_{out} = E_{n-1}E_{n-2} \ldots E_1E_0.(X = Y)_{in}$$

$$(X < Y)_{out} = L_{n-1} + E_{n-1}L_{n-2} + \ldots + E_{n-1}E_{n-2} \ldots E_1L_0 + E_{n-1}E_{n-2} \ldots E_0.(X < Y)_{in}$$

4-bit Comparator IC



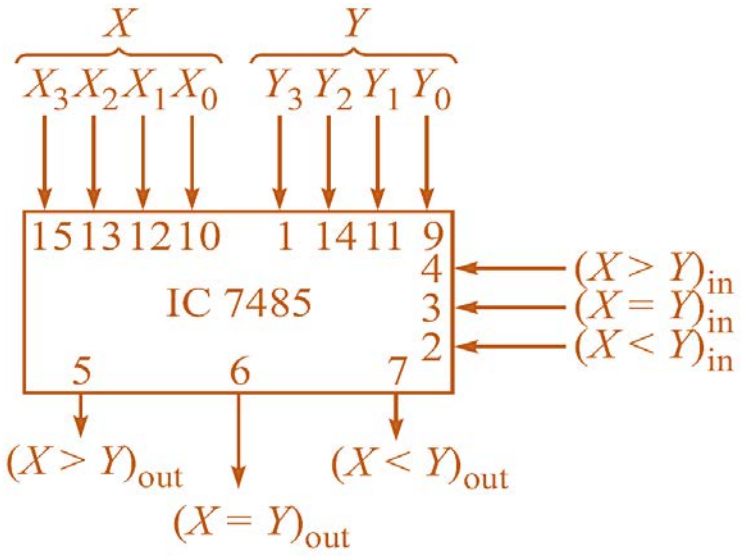


AND output = 1
if A_3=0, B_3=1

AND output = 1
if A_3=B_3 and A_2=0, B_2=1

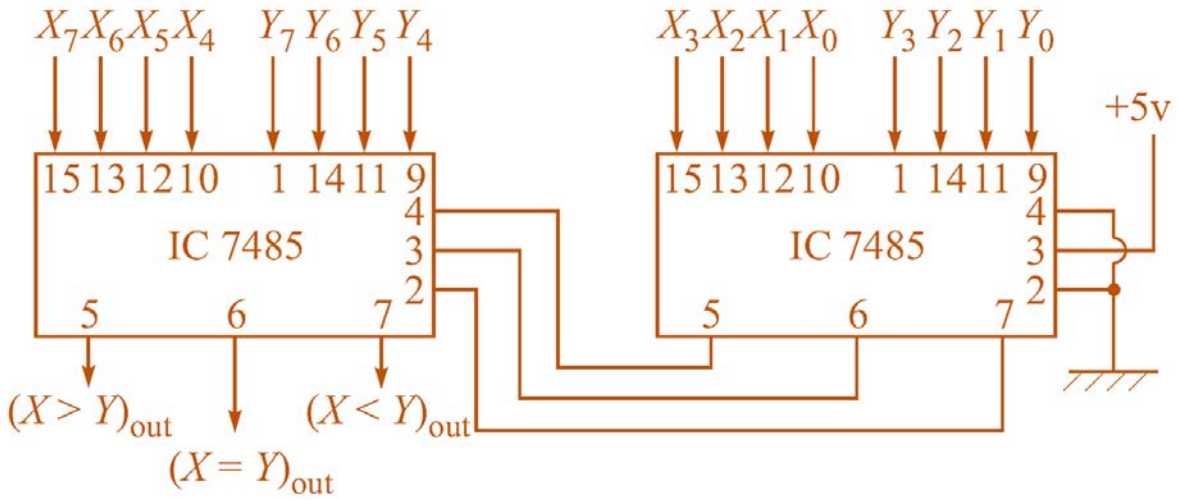
IC 7485 Function Table

COMPARING INPUTS				CASCADING INPUTS			OUTPUTS		
A3, B3	A2, B2	A1, B1	A0, B0	A>B	A<B	A=B	A>B	A<B	A=B
A3>B3	X	X	X	X	X	X	H	L	L
A3<B3	X	X	X	X	X	X	L	H	L
A3=B3	A2>B2	X	X	X	X	X	H	L	L
A3=B3	A2<B2	X	X	X	X	X	L	H	L
A3=B3	A2=B2	A1>B1	X	X	X	X	H	L	L
A3=B3	A2=B2	A1<B1	X	X	X	X	L	H	L
A3=B3	A2=B2	A1=B1	A0>B0	X	X	X	H	L	L
A3=B3	A2=B2	A1=B1	A0<B0	X	X	X	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	H	L	L	H	L	L
A3=B3	A2=B2	A1=B1	A0=B0	L	H	L	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	H	H	L	L	L	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	L	H	H	L
A3=B3	A2=B2	A1=B1	A0=B0	X	X	H	L	L	H



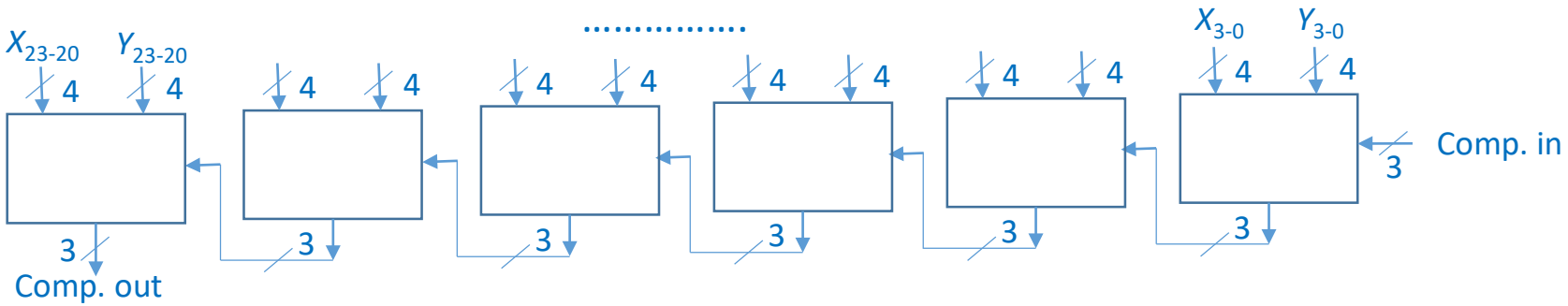
Cascaded Comparison

8-bit comparison

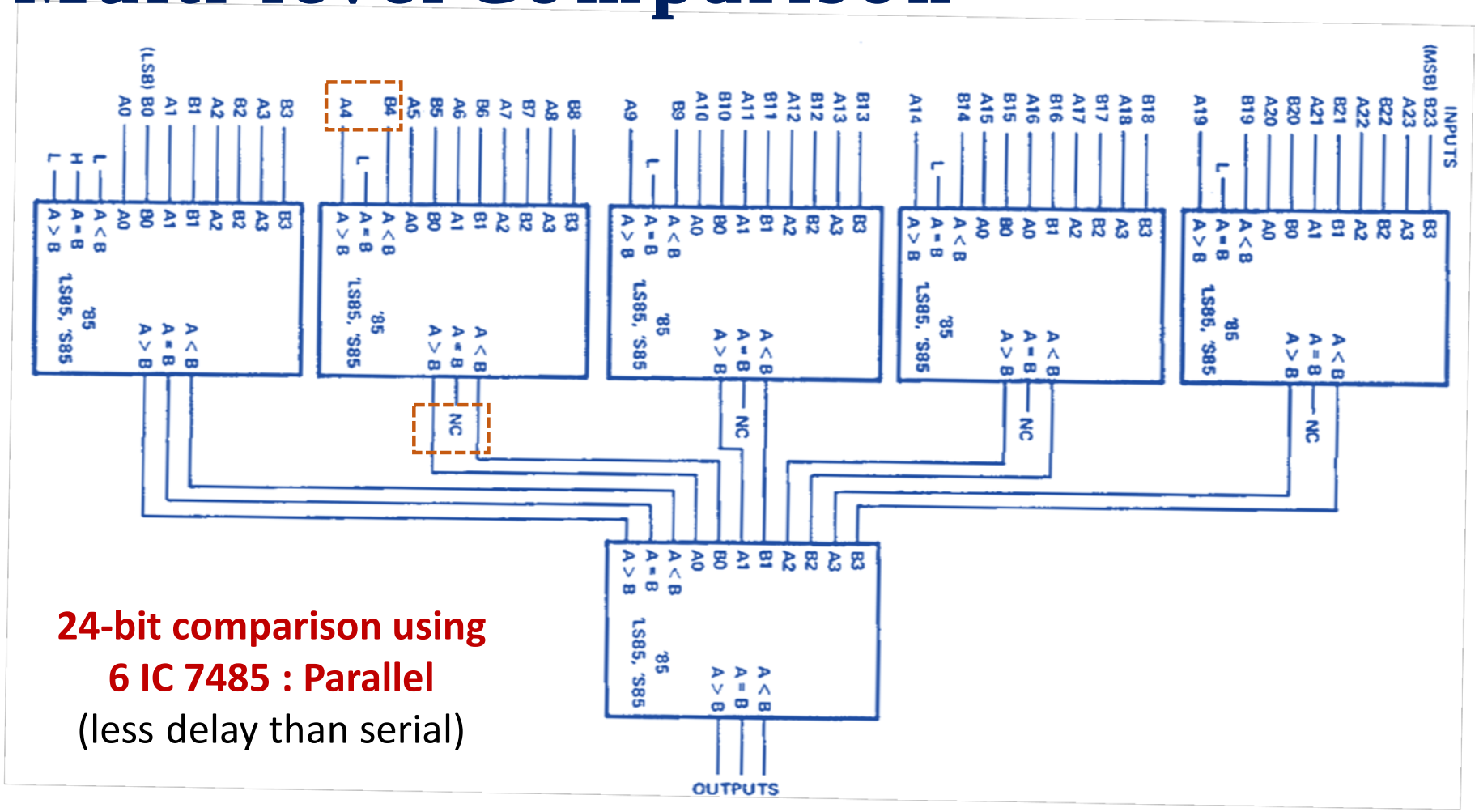


Here, generation of comparator output of one stage delays the output generation of next stage to which it is fed.

24-bit comparison



Multi-level Comparison



24-bit comparison using
6 IC 7485 : Parallel
(less delay than serial)

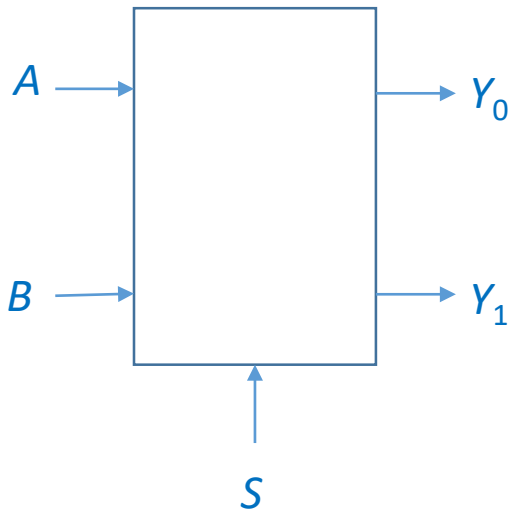
CASCADING INPUTS			OUTPUTS		
A>B	A<B	A=B	A>B	A<B	A=B
H	H	L	L	L	L
L	L	L	H	H	L

A3=B3 | A2=B2 | A1=B1 | A0=B0

Treated equal in next level

Arithmetic Logic Unit

Arithmetic Logic Unit (ALU) is a versatile unit that can perform arithmetic or logic operation on operands as per control input.



If $S = 0$,

$Y_0 = (A.B)'$

$Y_1 = (A + B)'$

If $S = 1$,

$Y_0 = A'.B + A.B'$

(sum bit of H.A.)

$Y_1 = A.B$

(carry bit of H.A.)

$Y_0 = S'.(A.B)' + S.(A'.B + A.B')$

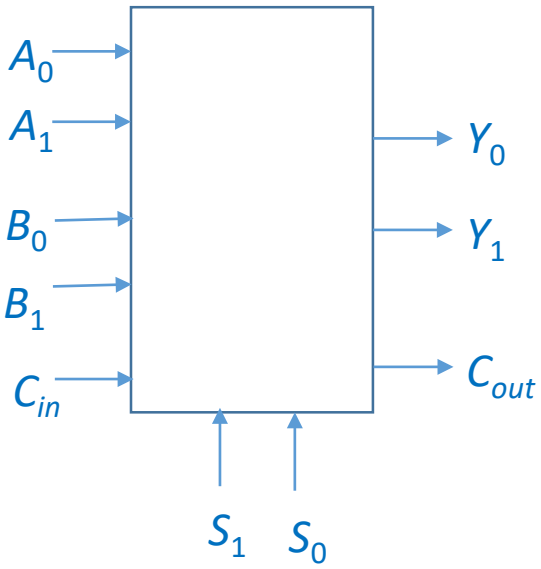
… simplification …

$= S'.A' + A'.B + A.B'$

$Y_1 = S'.(A + B)' + S.A.B$

$= S'.A'.B' + S.A.B$

Arithmetic Logic Unit



Function Table

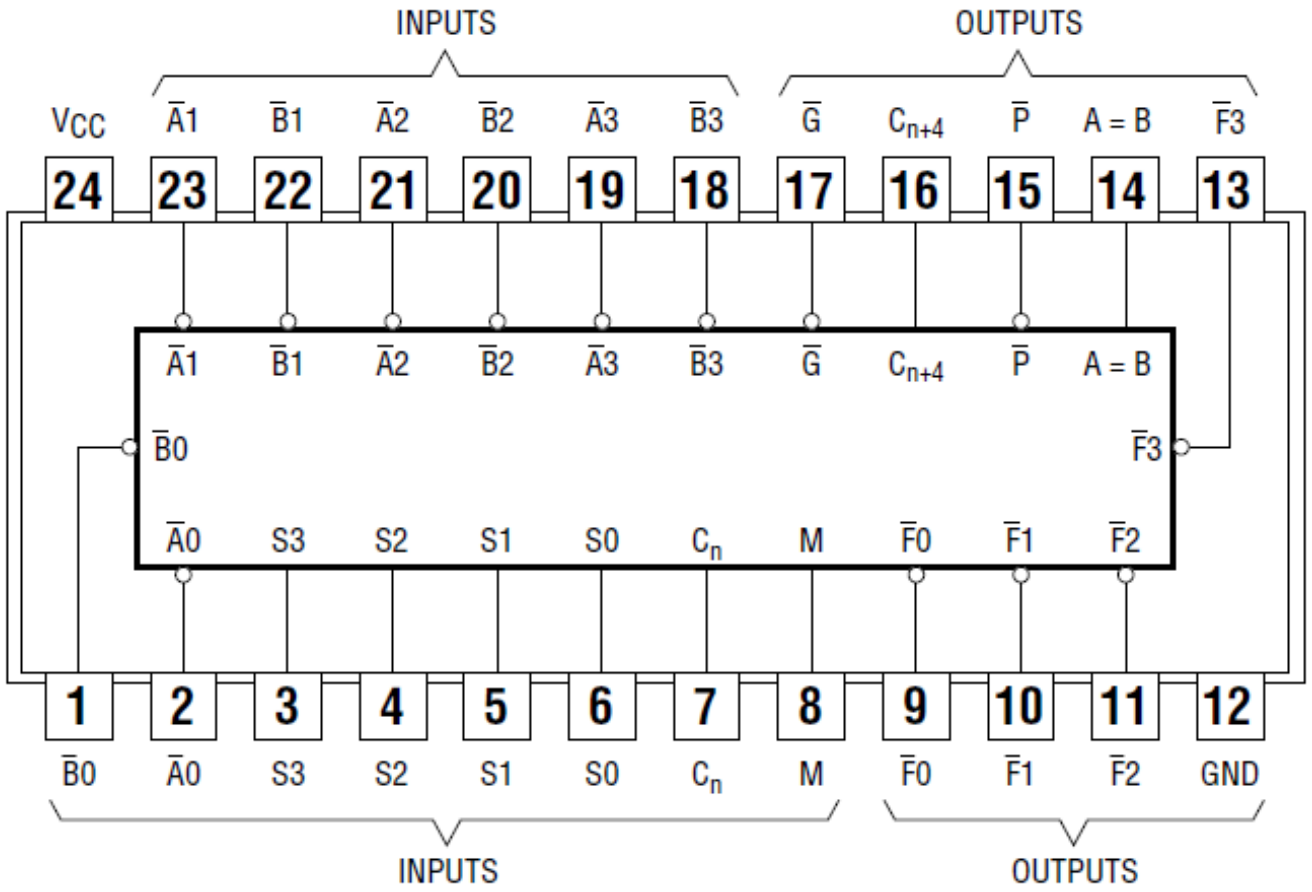
S_1	S_0	C_{in}	Output
0	0	X	$Y = (A.B)'$
0	1	X	$Y = (A + B)'$
1	0	0	$Y = A$
1	0	1	$Y = A$ PLUS 1
1	1	0	$Y = A$ PLUS B
1	1	1	$Y = A$ PLUS B PLUS 1

Logic operation: Bitwise
Arithmetic operation: 2-bit number

	$\overline{A}_0\overline{B}_0$	$\overline{A}_0B_0$	A_0B_0	$A_0\overline{B}_0$
$\overline{S}_1\overline{S}_0$	1	1	0	1
$\overline{S}_1S_0$	1	0	0	0
S_1S_0	0	1	0	1
$S_1\overline{S}_0$	0	0	1	1

Y_0
for $C_{in} = 0$

IC 74181



M: Mode L: Arithmetic operation
H: Logic operation

Arithmetic operation:
Addition
Subtraction
Shift operand *A* by one position
Magnitude comparison
….

Logic operation:
NOT, AND, NAND, OR, NOR
Ex-OR, Ex-NOR
….

Function outputs: (F3…F0)

Other outputs:
Group carry Generation and Propagation,
Ripple carry, Equality

Function Table

SELECTION				ACTIVE-LOW DATA		
				M = H	M = L; ARITHMETIC OPERATIONS	
S3	S2	S1	S0	LOGIC FUNCTION	Cn = L (no carry)	Cn = H (with carry)
L	L	L	L	$F = \overline{A}$	F = A MINUS 1	F = A
L	L	L	H	$F = \overline{AB}$	F = AB MINUS 1	F = AB
L	L	H	L	$F = \overline{A} + B$	F = $A\overline{B}$ MINUS 1	F = $A\overline{B}$
L	L	H	H	F = 1	F = MINUS 1(2's COMP)	F = 0
L	H	L	L	$F = \overline{A + B}$	F = A PLUS (A + $\overline{B}$)	F = A PLUS (A + $\overline{B}$) PLUS 1
L	H	L	H	$F = \overline{B}$	F = AB PLUS (A + $\overline{B}$)	F = AB PLUS (A + $\overline{B}$) PLUS 1
L	H	H	L	$F = \overline{A \oplus B}$	F = A MINUS B MINUS 1	F = A MINUS B
L	H	H	H	$F = A + \overline{B}$	F = A + $\overline{B}$	F = (A + $\overline{B}$) PLUS 1
H	L	L	L	$F = \overline{A}B$	F = A PLUS (A + B)	F = A PLUS (A + B) PLUS 1
H	L	L	H	$F = A \oplus B$	F = A PLUS B	F = A PLUS B PLUS 1
H	L	H	L	F = B	F = $A\overline{B}$ PLUS (A + B)	F = $A\overline{B}$ PLUS (A + B) PLUS 1
H	L	H	H	F = A + B	F = (A + B)	F = (A + B) PLUS 1
H	H	L	L	F = 0	F = A PLUS A	F = A PLUS A PLUS 1
H	H	L	H	$F = A\overline{B}$	F = AB PLUS A	F = AB PLUS A PLUS 1
H	H	H	L	F = AB	F = $A\overline{B}$ PLUS A	F = $A\overline{B}$ PLUS A PLUS 1
H	H	H	H	F = A	F = A	F = A PLUS 1

- ($A = B$) output is open collector providing wired-AND option (to compare > 4 bits)
- A PLUS A = 2 x A which makes one left shift of A.

Note:
74181 uses 1's C subtraction.

C_n	C_{n+4}	Result
L	L	$A \leq B$
L	H	$A > B$
H	L	$A < B$
H	H	$A \geq B$

Magnitude Comparison

Data: Active Low
Mode, M: L

$S_3S_2S_1S_0$: LHHL
(Subtraction)

Function Table

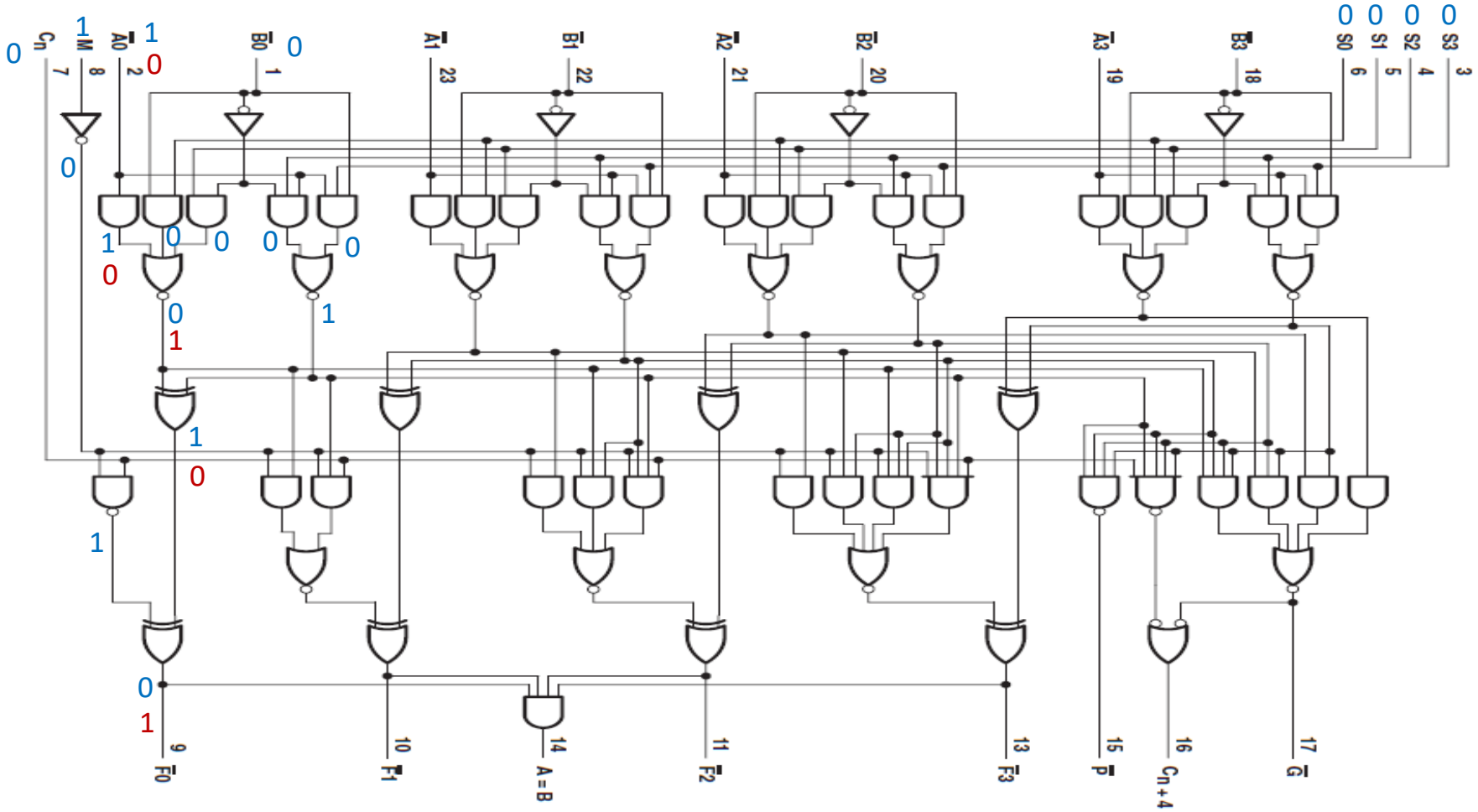
SELECTION S3 S2 S1 S0	M = H LOGIC FUNCTION
L L L L	$F = \overline{A}$
L L L H	$F = \overline{AB}$
L L H L	$F = \overline{A} + B$
L L H H	$F = 1$
L H L L	$F = \overline{A + B}$
L H L H	$F = \overline{B}$
L H H L	$F = \overline{A \oplus B}$
L H H H	$F = A + \overline{B}$
H L L L	$F = \overline{AB}$
H L L H	$F = A \oplus B$
H L H L	$F = B$
H L H H	$F = A + B$
H H L L	$F = 0$
H H L H	$F = A\overline{B}$
H H H L	$F = AB$
H H H H	$F = A$

A	B	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
		13	5	9	1	14	6	10	2	15	7	11	3	16	8	12	4

S3	S2	S1	S0:	Function No.
L	L	L	L	1
L	L	L	H	2
L	L	H	L	3
		.....		
H	H	H	H	16

All possible logic functions of 2 variables are generated. Note that *A* and *B* are of 4-bits and bit-wise logic operation is done.

Logic Circuit



SELECTION S3 S2 S1 S0	M = H LOGIC FUNCTION
L L L L	$F = \overline{A}$
L L L H	$F = \overline{AB}$
L L H L	$F = \overline{A} + B$

S3 S2 S1 S0: 0000

M: 1, B: 0000, C_n: 0

A0: 1 ➡ **F0:** 0

A0: 0 ➡ **F0:** 1

F0 = A0'

References:

- Donald P. Leach, Albert P. Malvino, and Goutam Saha, Digital Principles & Applications 8e, McGraw Hill