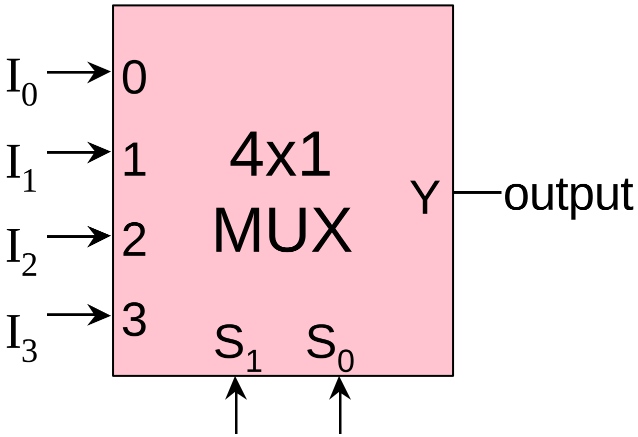
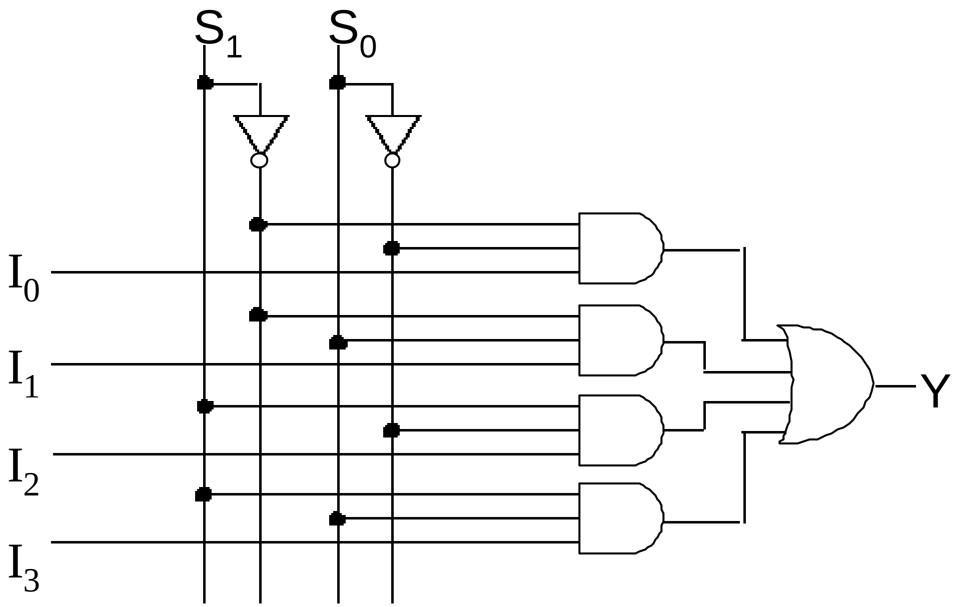
Multiplexer(MUX) or Data Selector

- ❑ 여러개의 입력중 하나를 선택하여 출력으로 전송
- ❑ 4x1 multiplexer

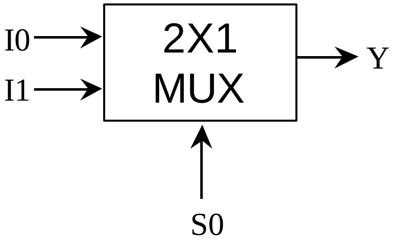


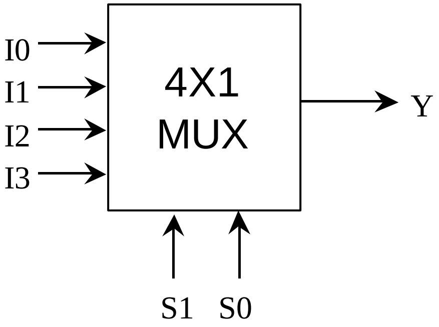
selector		output
S_1	S_0	**Y**
0	0	**I_0**
0	1	**I_1**
1	0	**I_2**
1	1	**I_3**

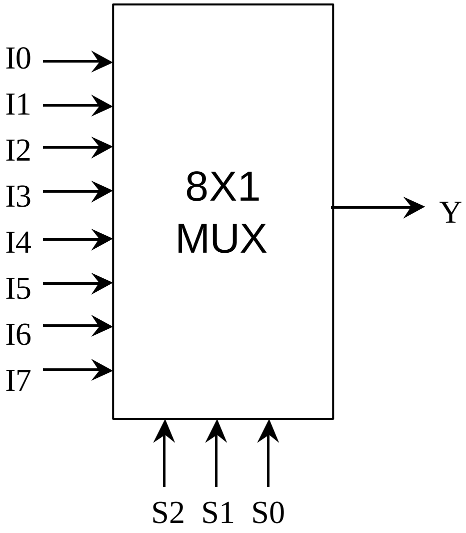


Multiplexer

- ❑ 2x1 MUX : Y= $S_0' I_0+S_0I_1$
- ❑ 4x1 MUX : Y= $S_1' S_0' I_0+S_1' S_0I_1+S_1S_0' I_2 +S_1S_0I_3$
- ❑ 8x1 MUX : Y= $S_2' S_1' S_0' I_0+ S_2' S_1' S_0I_1+S_2' S_1S_0' I_2+S_2' S_1S_0I_3$ +........

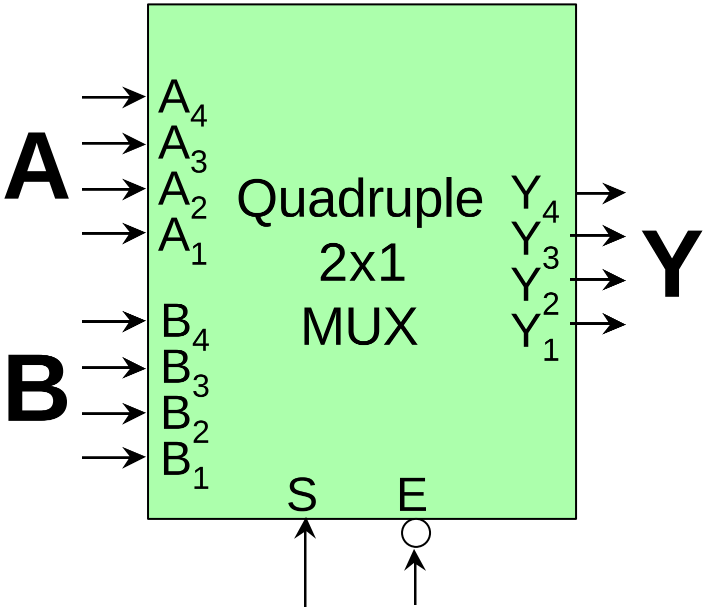




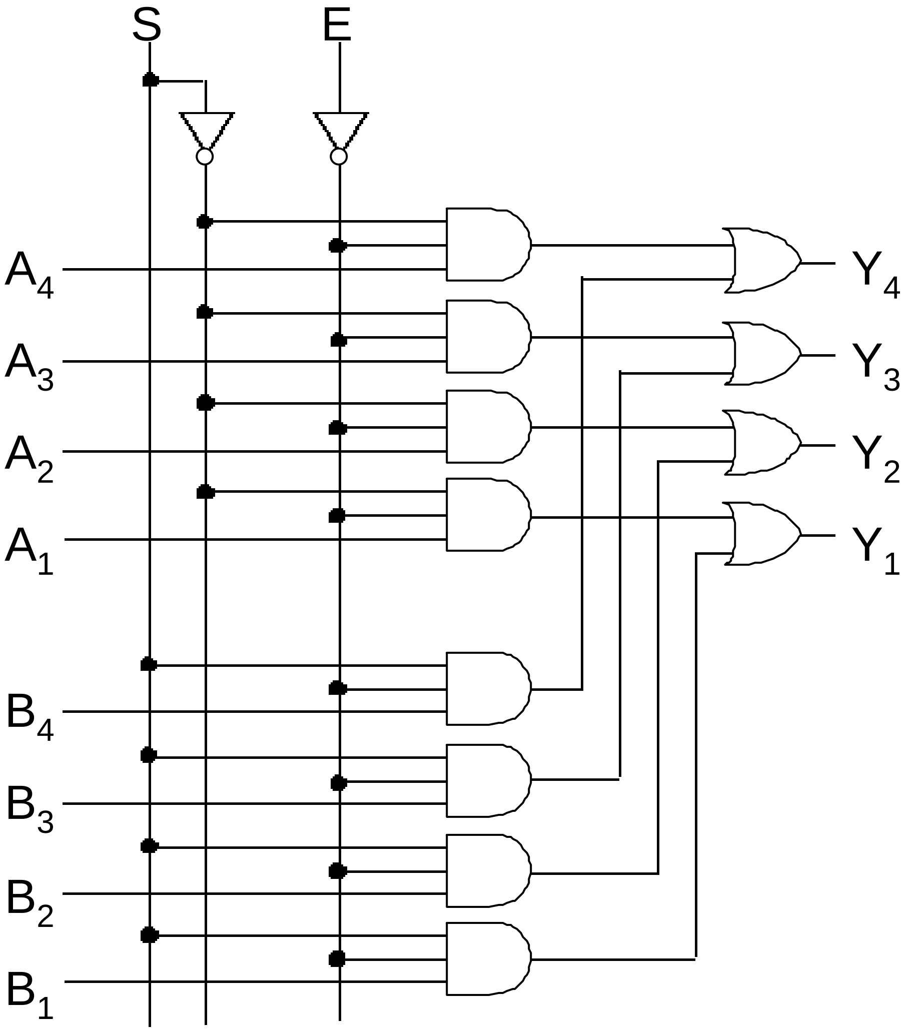


Quadruple 2-to-1 Line Multiplexer

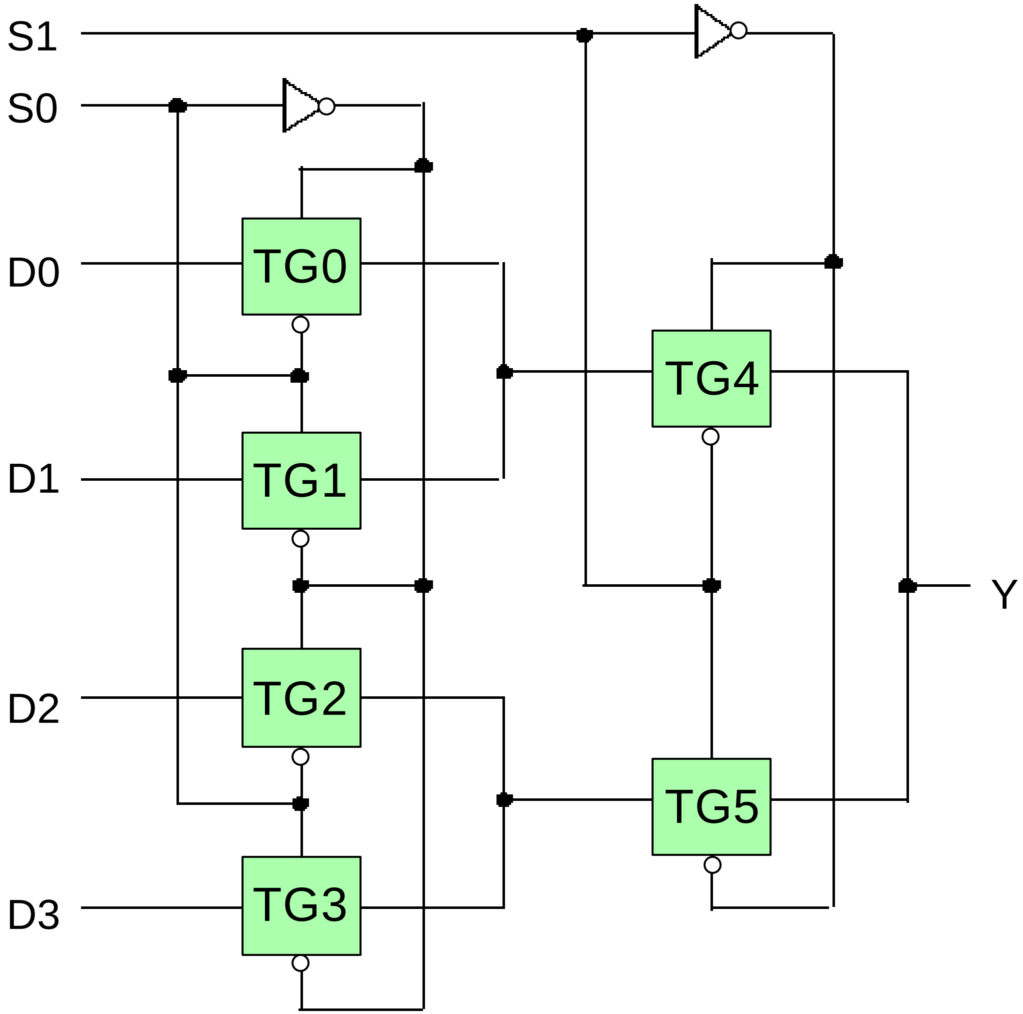
❑ 각각 4bit인 A, B중 하나를 선택하여 4bit를 출력으로 전송



E	S	Y
1	X	all 0
0	0	A
0	1	B



4x1 Multiplexer with Transmission Gate



		ON
S0	0	0,2
	1	1,3
S1	0	4
	1	5

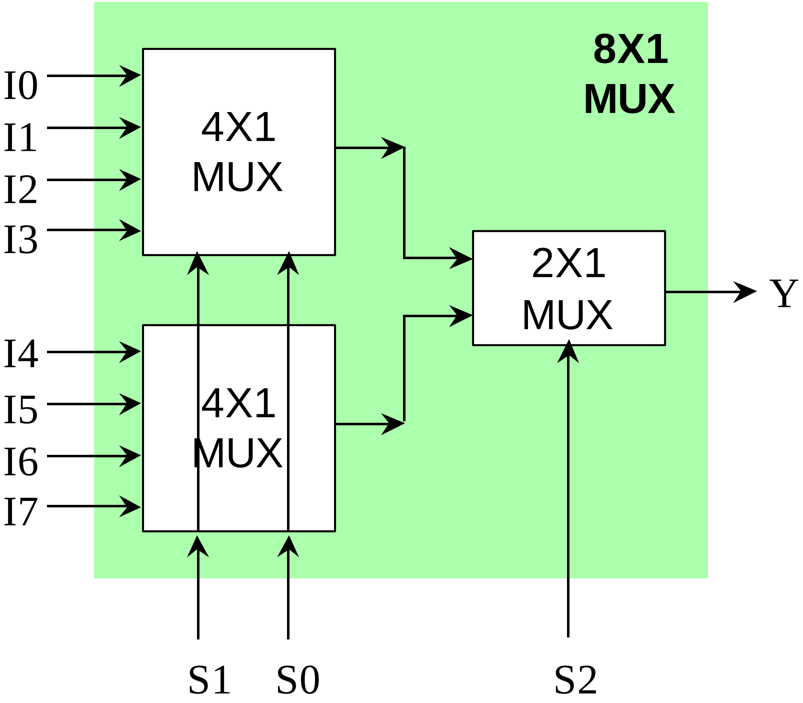
S1 S0	0 0	0 1	1 0	1 1
1st level	0,2	1,3	0,2	1,3
2nd level	4	4	5	5
Y	D0	D1	D2	D3

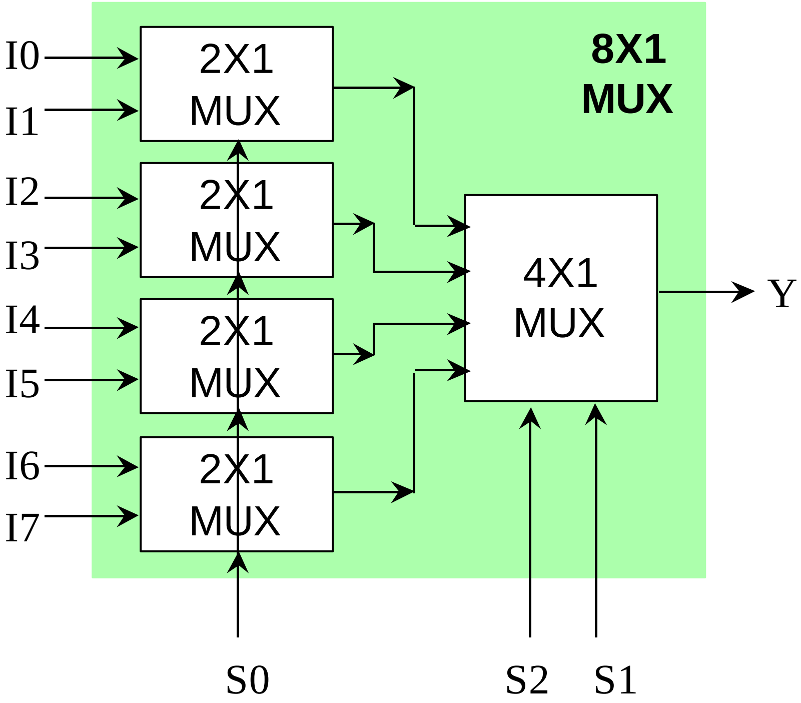
8x1 Multiplexer with Transmission Gate

8x1 MUX

Cascading Multiplexer

- ❑ Large multiplexers can be formed from smaller ones





Multiplexer를 이용한 Bool함수의 구현

(예) $F(A,B,C) = \sum(1,3,5,6)$

minterm	A	B	C	F
m_0	0	0	0	0
m_1	0	0	1	1
m_2	0	1	0	0
m_3	0	1	1	1
m_4	1	0	0	0
m_5	1	0	1	1
m_6	1	1	0	1
m_7	1	1	1	0

	BC	00	01	10	11
	입력	I_0	I_1	I_2	I_3
A=0	A’	0	①	2	③
A=1	A	4	⑤	⑥	7
		0	1	A	A’

즉 I_0=0, I_1=1, I_2=A, I_3=A’

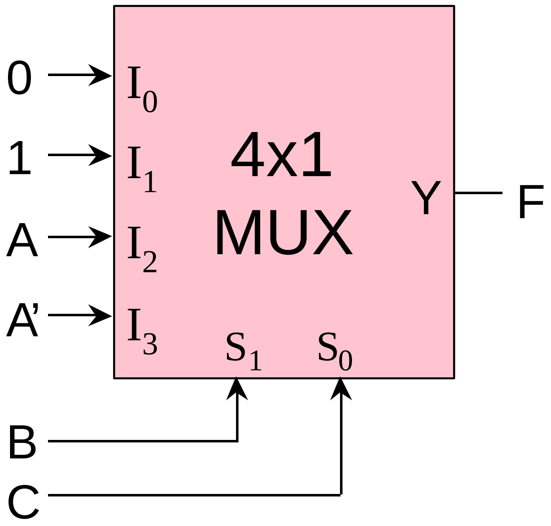
- B,C를 선택입력으로 사용하는 경우

  if BC=00 then F=0

  if BC=01 then F=1

  if BC=10 then F=A

  if BC=11 then F=A’



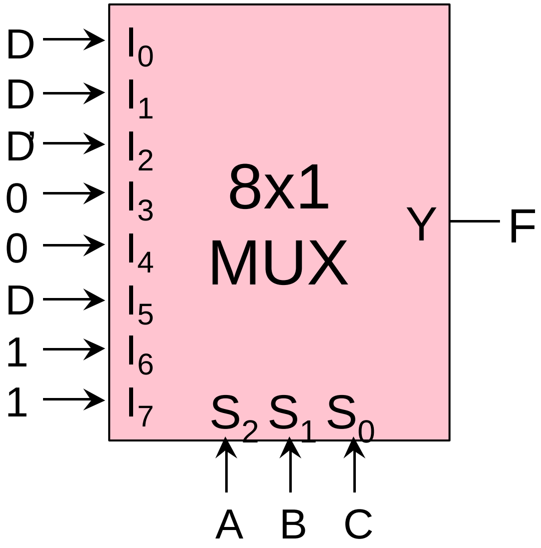
Multiplexer를 이용한 Bool함수의 구현

(예) F(A,B,C,D) = ∑(1,3,4,11,12,13,14,15)

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

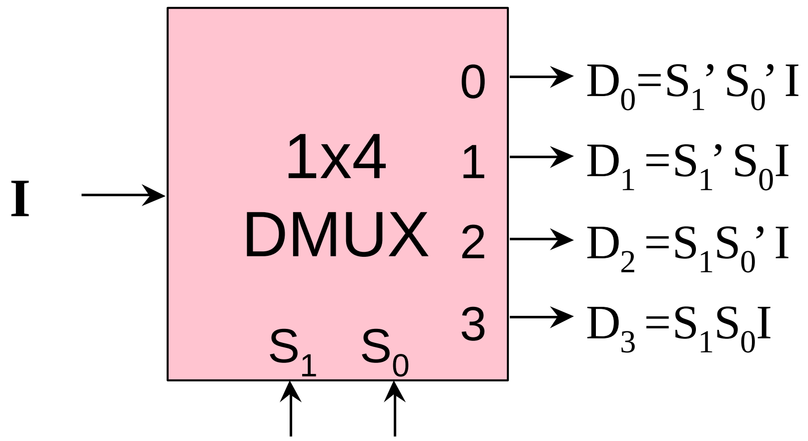
A,B,C를 선택입력으로 사용하는 경우

	ABC	000	001	010	011	100	101	110	111
		I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
D=0	D’	0	2	4	6	8	10	12	14
D=1	D	1	3	5	7	9	11	13	15
		D	D	D’	0	0	D	1	1



Demultiplexer(DMUX)

- ❑ 하나의 입력정보를 받아들여 2^n개의 출력선중 하나를 택하여 전송
- ❑ 출력선은 선택선에 의해 결정



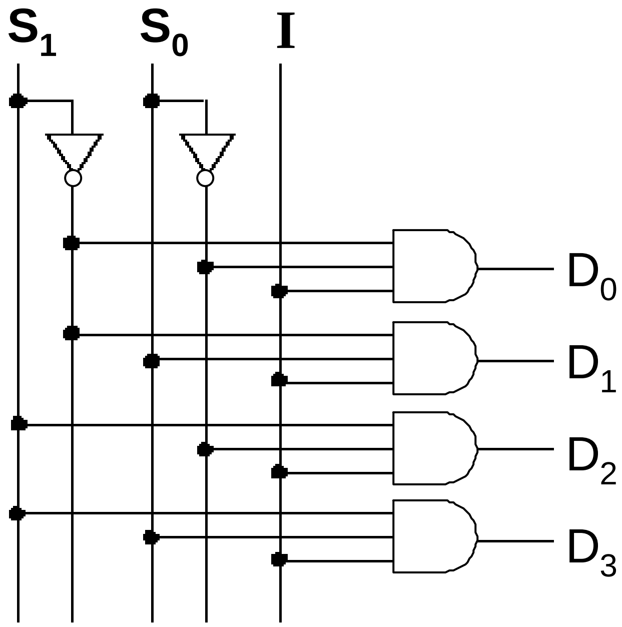
$D_0=S_1'S_0'I$

$D_1=S_1'S_0I$

$D_2=S_1S_0'I$

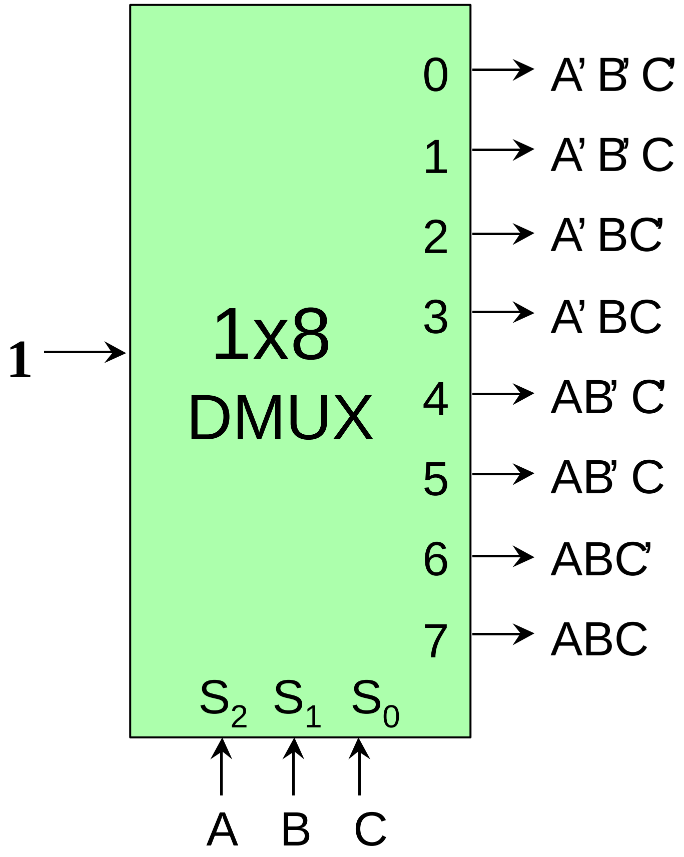
$D_3=S_1S_0I$

S_1	S_0	D_0	D_1	D_2	D_3
0	0	I	0	0	0
0	1	0	I	0	0
1	0	0	0	I	0
1	1	0	0	0	I



Demultiplexers as General-Purpose Logic

- A 1x2^n DMUX can implement any function of n variables
  - Apply the variables as select inputs
  - Tie the enable input to a logic "1"
  - Sum the appropriate minterms
- DMUX decodes appropriate minterms from the control signals
- Decoder with enable is identical to DMUX



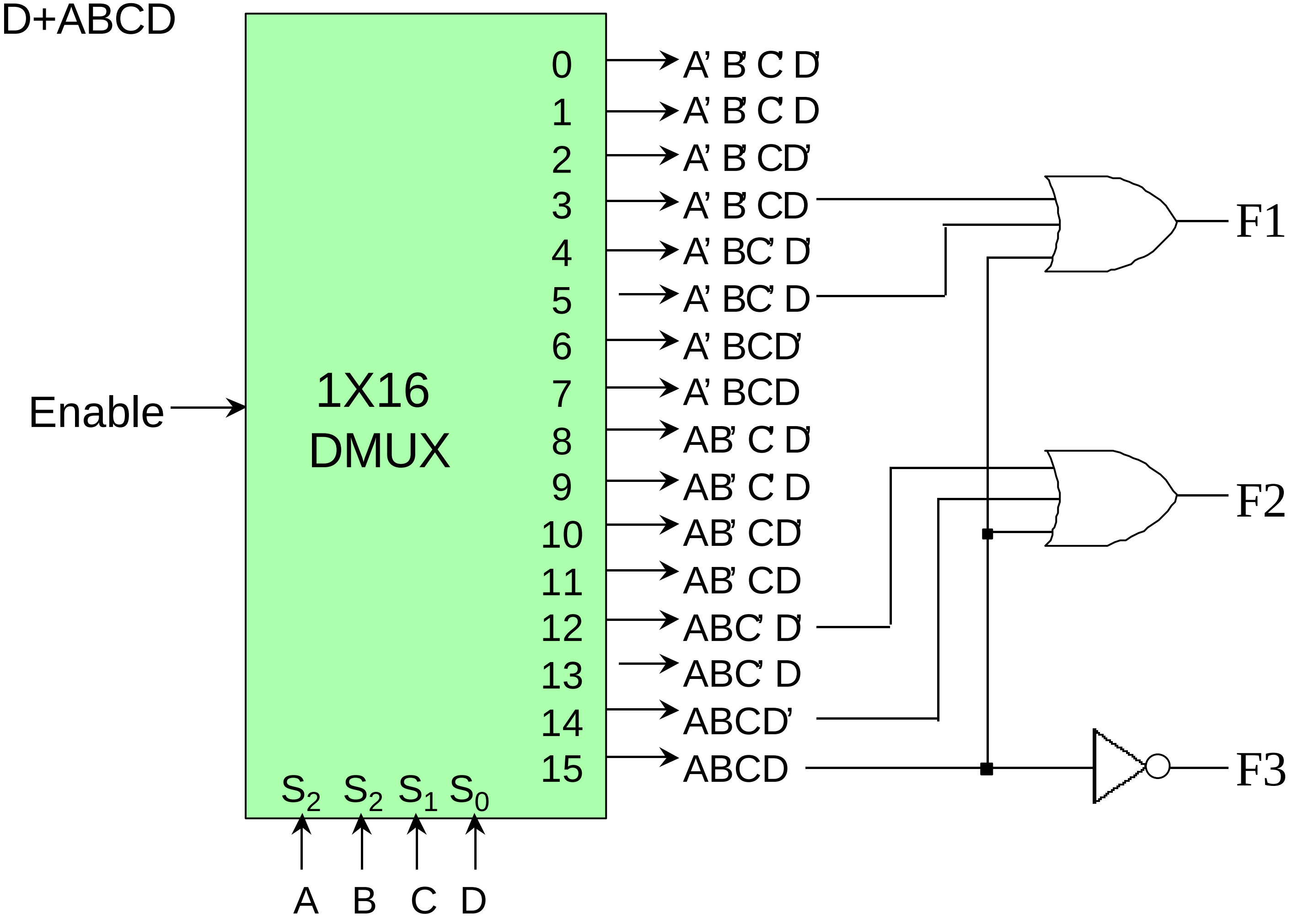
Demultiplexers as General-Purpose Logic

❑ Examples

F1= A' BC' D+A' B' CD+ABCD

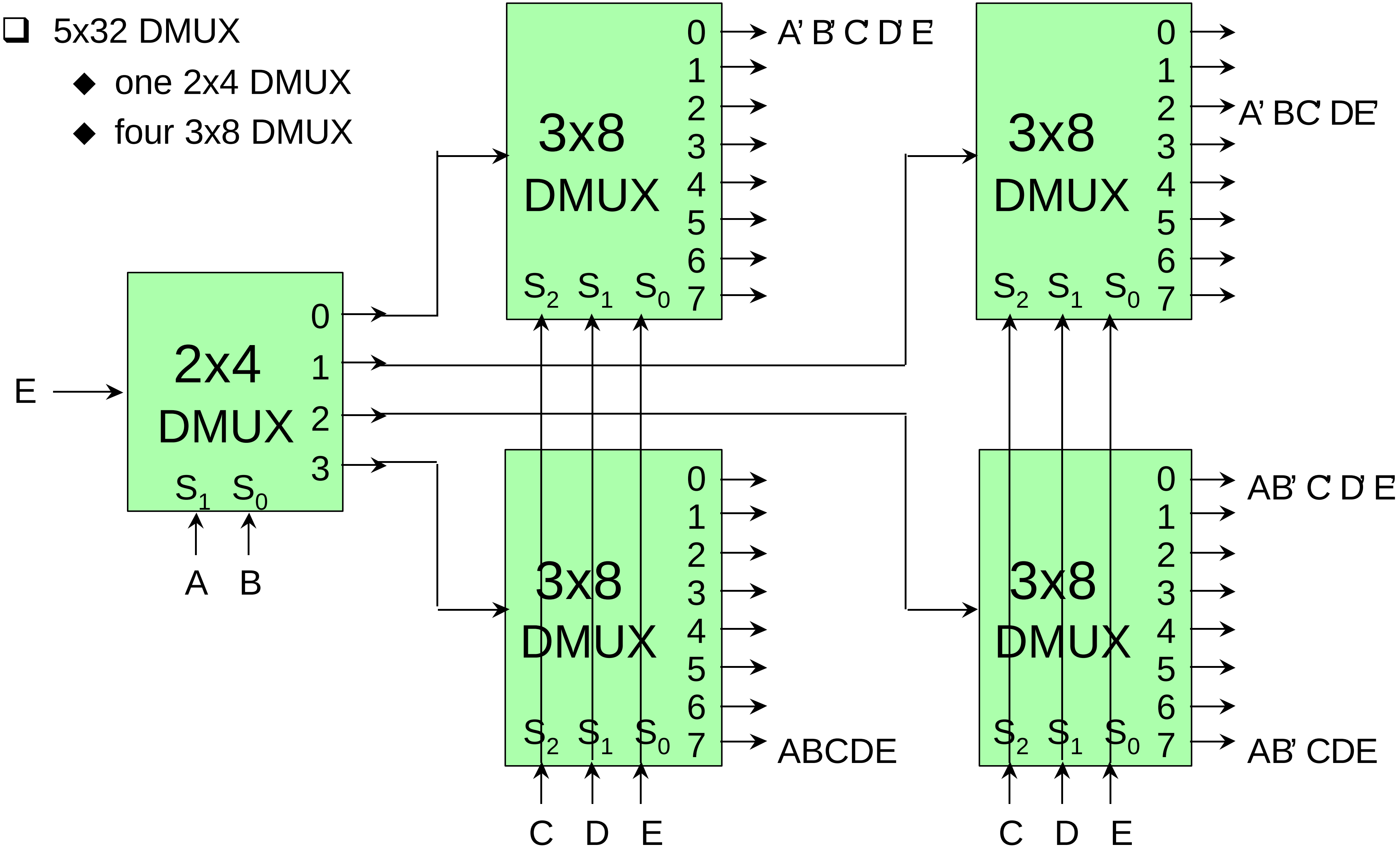
F2= ABC' D' +ABC

F3= A' +B' +C' +D'



Cascading DMUX

- ❑ 5x32 DMUX
  - ◆ one 2x4 DMUX
  - ◆ four 3x8 DMUX


E
2x4
DMUX
0
1
2
3
S1 S0
A B
3x8
DMUX
0 1 2 3 4 5 6 7
S2 S1 S0
A' B' C' D' E'
ABCDE
A' BC' DE'
AB' C' D' E'
AB' CDE
C D E
C D E