Truth Table

Boolean function

$f \in \mathbb{B}^i \to \mathbb{B}$

Boolean expression

$e ::= 0 \mid 1 \mid x_1 \mid \cdots \mid x_i \mid \neg e \mid e \cup e \mid e \cap e \mid mux(e,e,e)$

Choose an input order

$x_1 \cdots x_i$

$\text{truth}\, f \in \mathbb{D}_{2^i}$

Truth-table

$\text{truth}\, f = \sum f(b_0 \cdots b_{i-1}) 2^n \ : n = \sum b_k 2^k$

$\boxed{\text{truth}\, f = \text{truth}\, g \Leftrightarrow f = g}$

4 bit Codes

$b_3b_2b_1b_0$	$\{i : 1=b_i\}$	$b(z)$	$\sum_{k<4} b_k 2^k$	$\sum_{k<3} b_k 2^k - 8b_3$	$f(x,y) = b_{x+2y}$
0000		0	0	0	0
0001	0	1	1	1	¬x & ¬y
0010	1	z	2	2	x & ¬y
0011	0,1	$1+z$	3	3	¬y
0100	2	z^2	4	4	¬x & y
0101	0,2	$1+z^2$	5	5	¬x
0110	1,2	$z+z^2$	6	6	x ^ y
0111	0,1,2	$1+z+z^2$	7	7	¬x | ¬y
1000	3	z^3	8	-8	x & y
1001	0,3	$1+z^3$	9	-7	¬x ^ y
1010	1,3	$z+z^3$	10	-6	x
1011	0,1,3	$1+z+z^3$	11	-5	x | ¬y
1100	2,3	z^2+z^3	12	-4	y
1101	0,2,3	$1+z^2+z^3$	13	-3	¬x | y
1110	1,2,3	$z+z^2+z^3$	14	-2	x | y
1111	0,1,2,3	$1+z+z^2+z^3$	15	-1	1

Normal Forms

x_1	x_2	x_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

DNF *(OR-AND)*

$$f = x_1'x_2x_3 + x_1x_2'x_3 + x_1x_2x_3$$

CNF *(AND-OR)*

$$f = (x_1+x_2+x_3)(x_1+x_2+x_3')$$
$$(x_1+x_2'+x_3)(x_1'+x_2+x_3)(x_1'+x_2'+x_3)$$

ENF *(XOR-AND)*

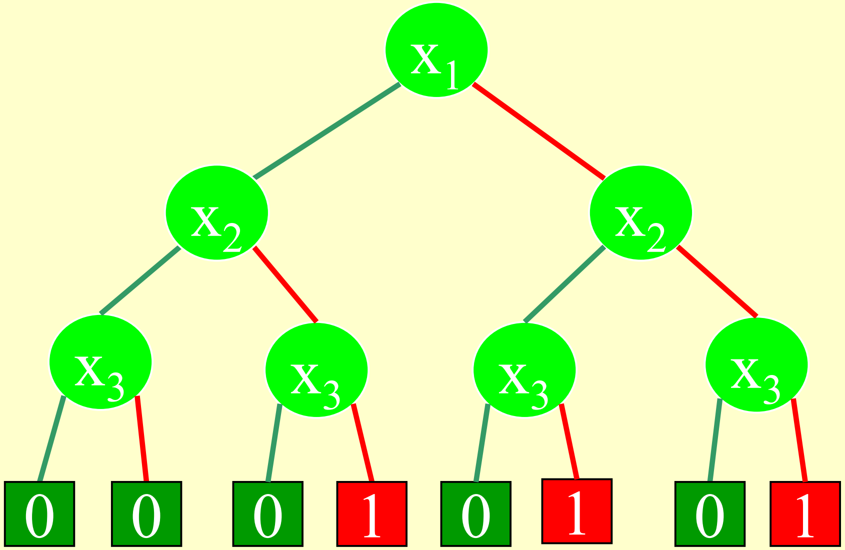
$$f = x_1x_3 + x_2x_3 + x_1x_2x_3$$

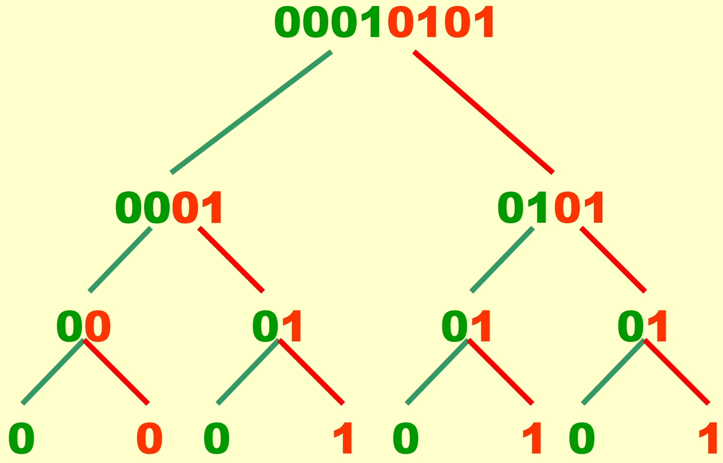
Decision Tree

x_1	0	0	0	0	1	1	1	1
x_2	0	0	1	1	0	0	1	1
x_3	0	1	0	1	0	1	0	1
f	0	0	0	1	0	1	0	1

Shannon Decomposition

$$f(x_1 \cdots x_i) = mux(x_1, f(1x_2 \cdots x_i), f(0x_2 \cdots x_i))$$





00010101

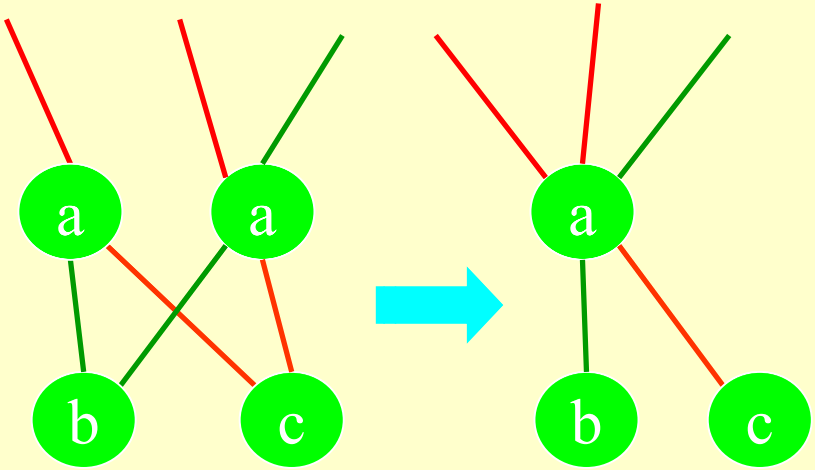
0001 0101

00 01 01 01

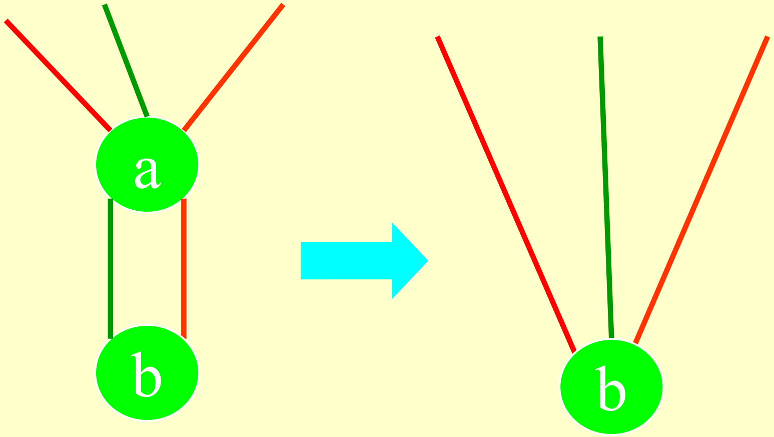
0 0 0 1 0 1 0 1

Reduced Decision Tree

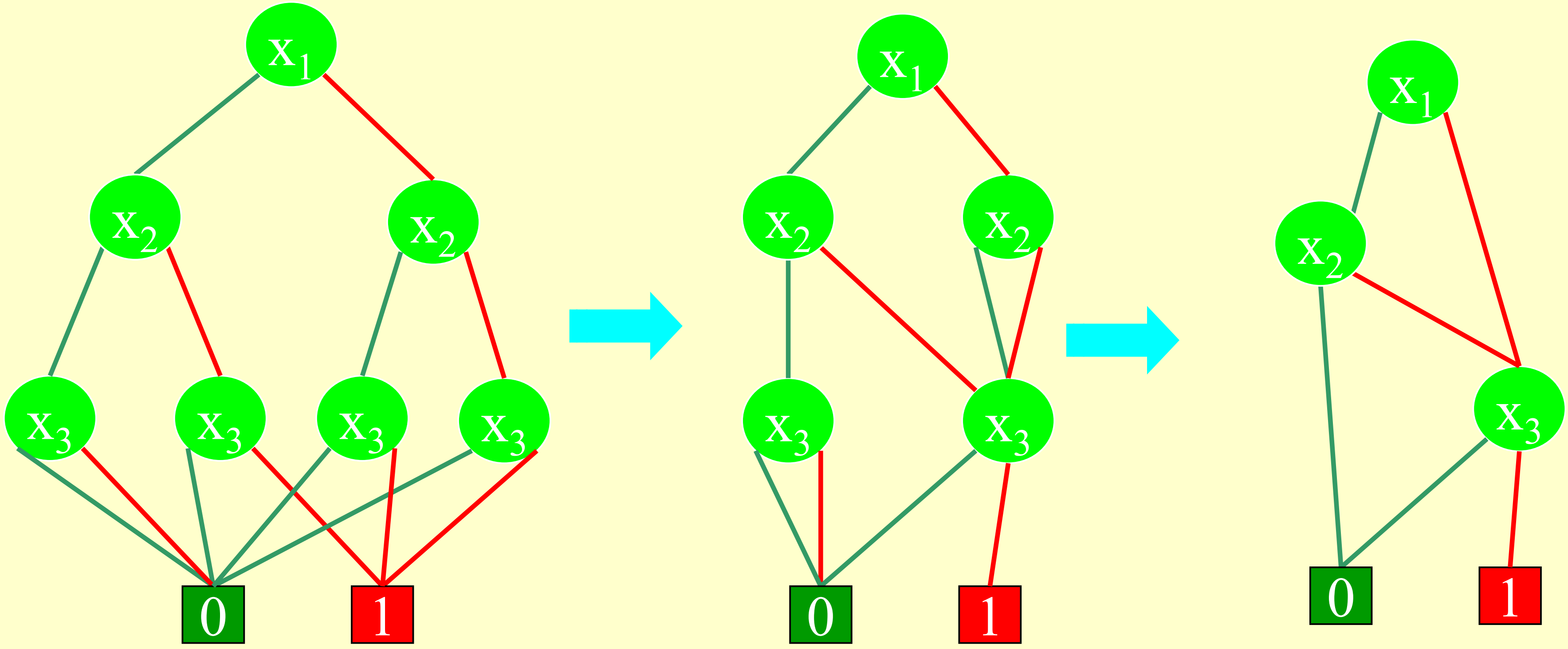
Merge equal nodes



Eliminate redundant tests



Binary Decision Diagram



Decision tree

ROBDD

Full Adder Synthesis

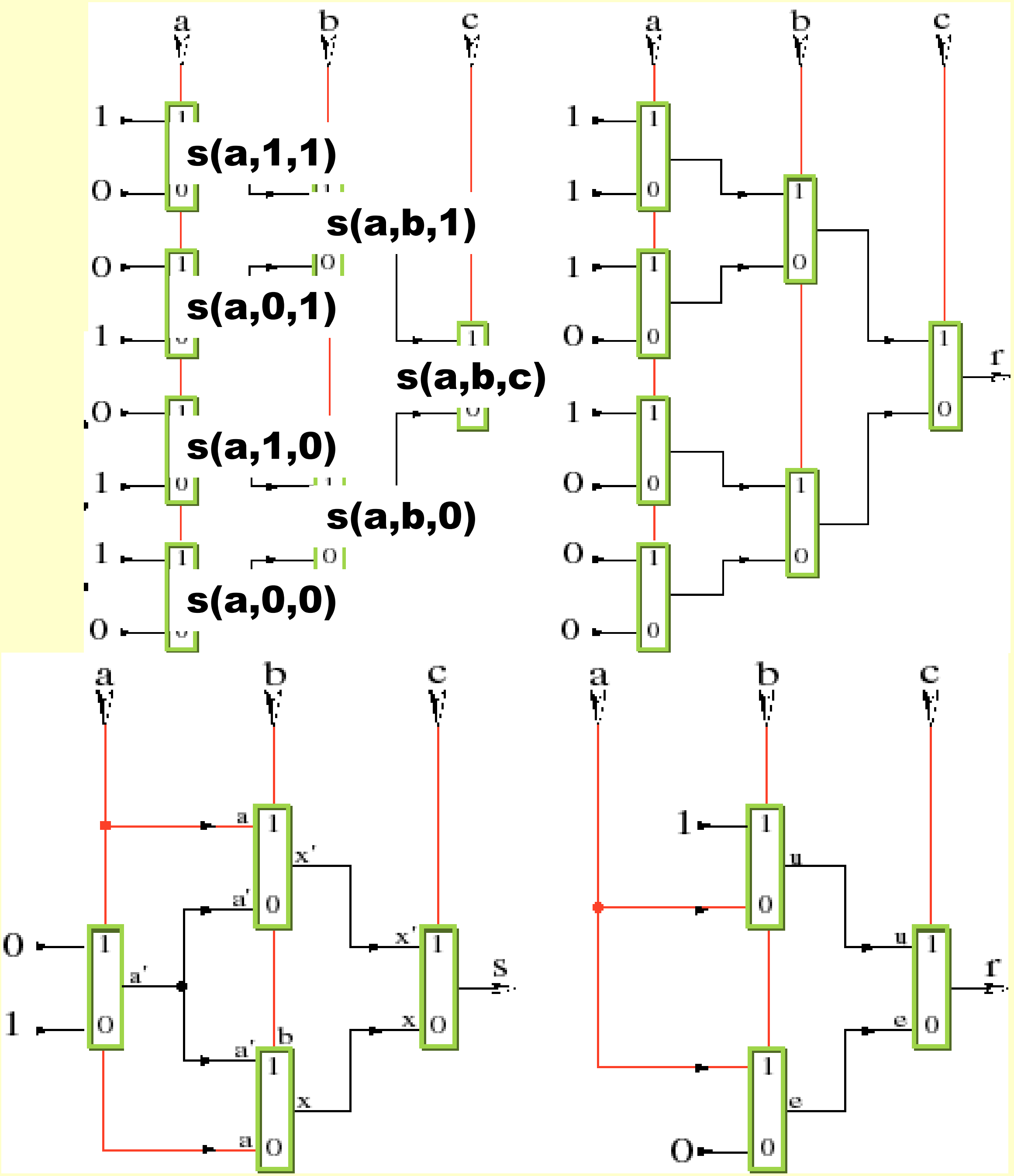
a	b	c	s	r
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

Truth Table

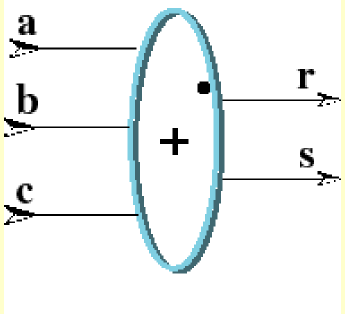
LUT

8-ways MUX

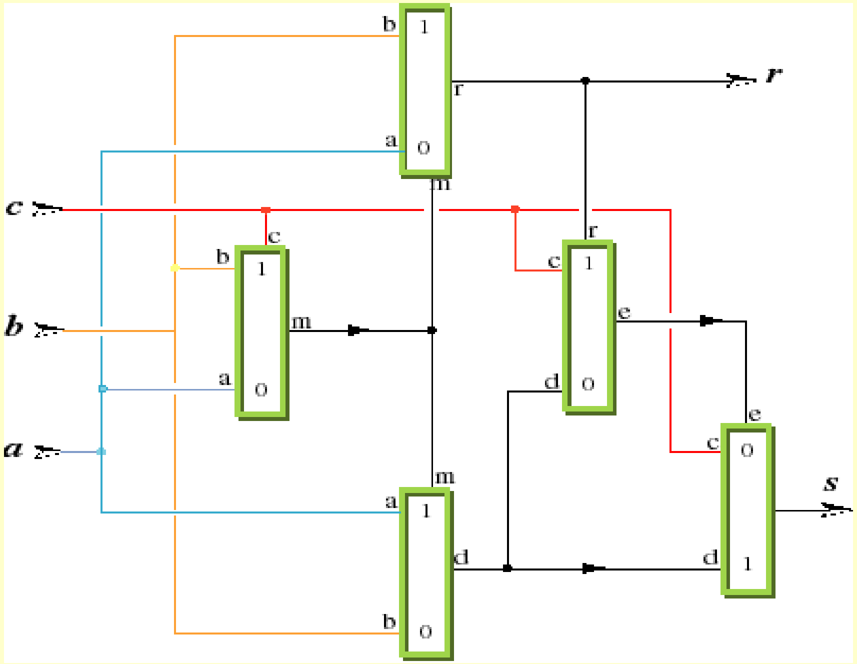
Reduce

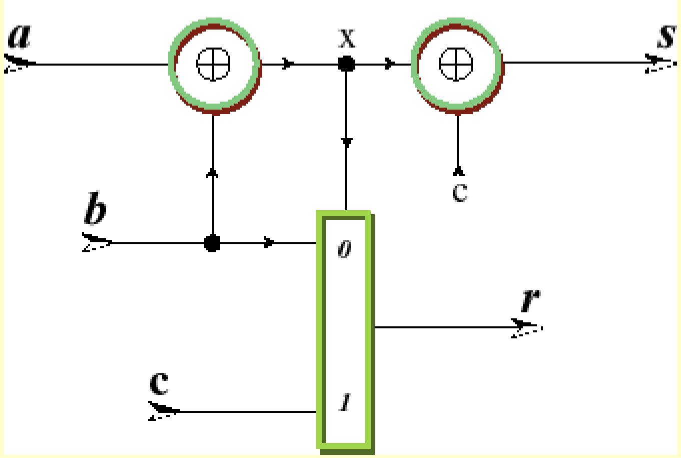


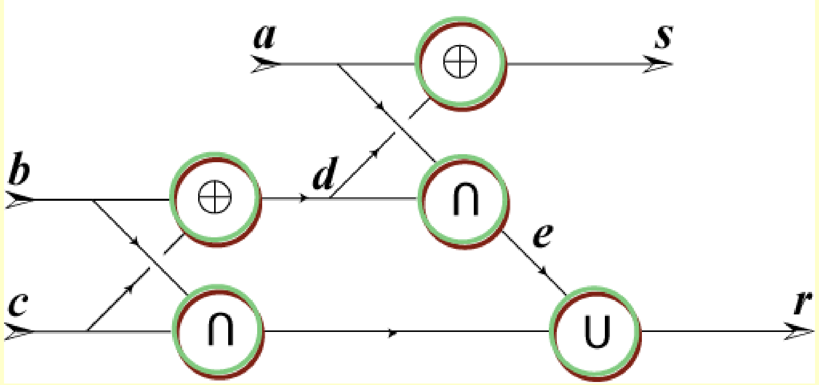
Minimal Full Adder



$$s_N = a_N \oplus b_N \oplus c_N$$
$$r_N = a_N b_N \oplus b_N c_N \oplus c_N a_N$$

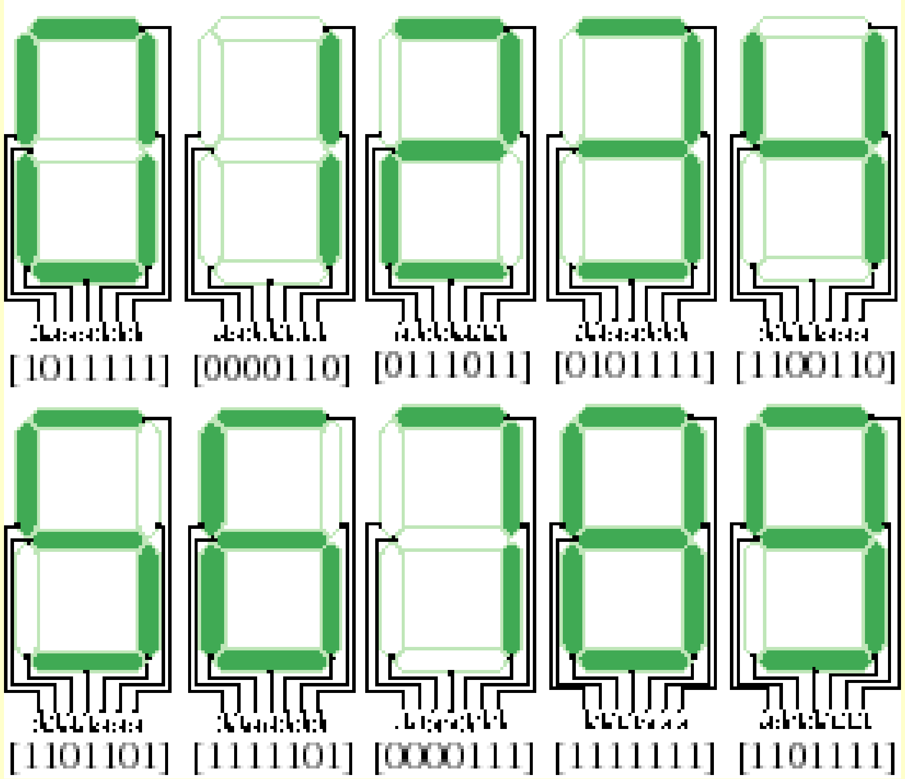




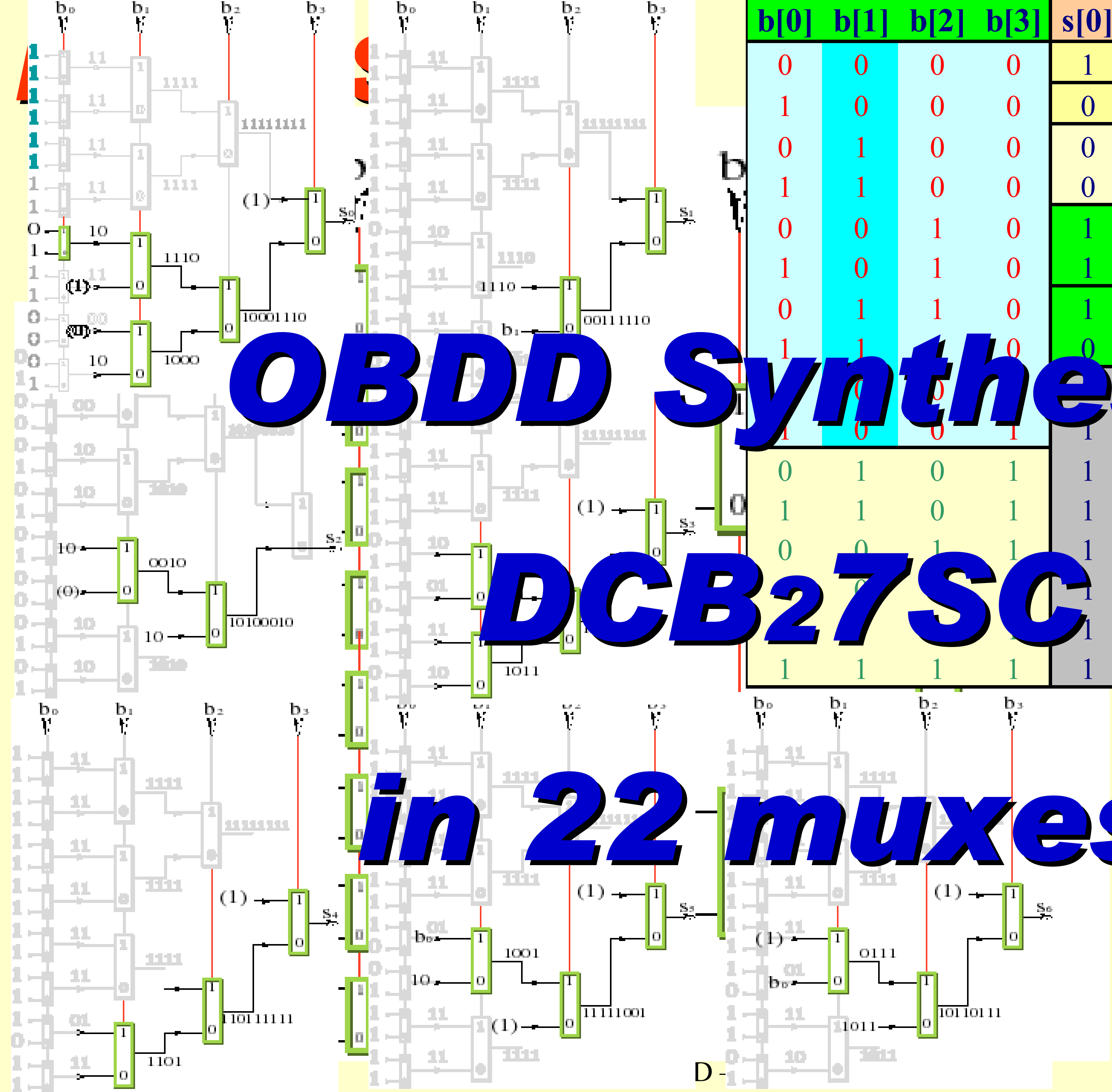


DCB 2 7SC

B	b[0]	b[1]	b[2]	b[3]	s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]
0	0	0	0	0	1	0	1	1	1	1	1
1	1	0	0	0	0	0	0	0	1	1	0
2	0	1	0	0	0	1	1	1	0	1	1
3	1	1	0	0	0	1	0	1	1	1	1
4	0	0	1	0	1	1	0	0	1	1	0
5	1	0	1	0	1	1	0	1	1	0	1
6	0	1	1	0	1	1	1	1	1	0	1
7	1	1	1	0	0	0	0	0	1	1	1
8	0	0	0	1	1	1	1	1	1	1	1
9	1	0	0	1	1	1	0	1	1	1	1
10											
11											
12											
13											
14											
15											



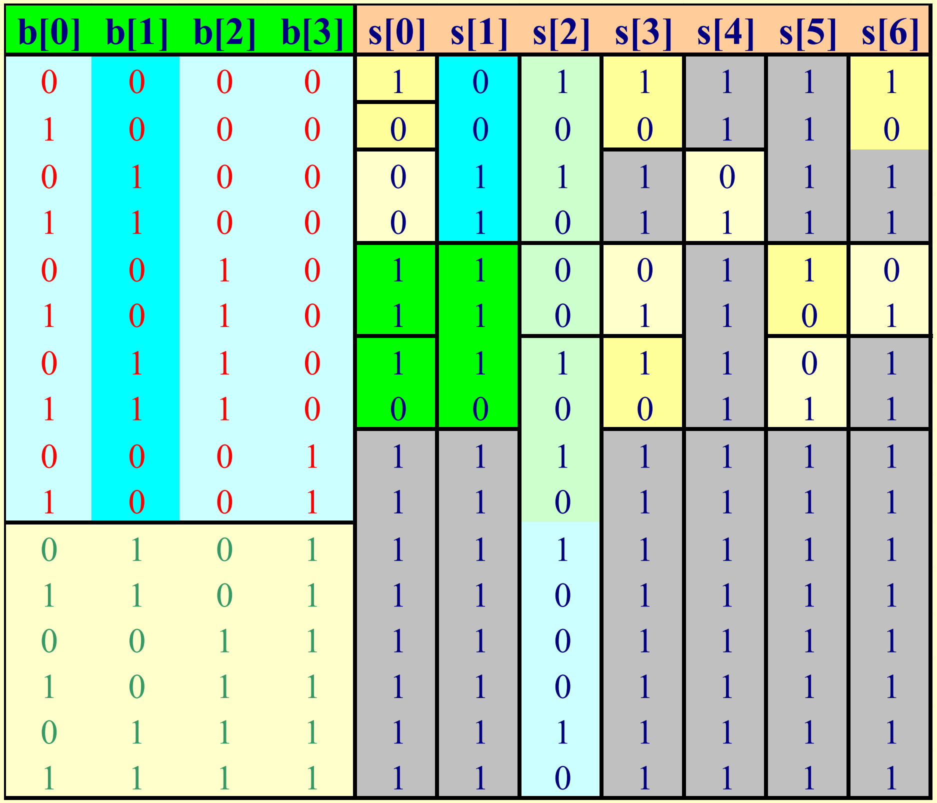
OBDD Synthesis: DCB2 7SC in 22 muxes.



b[0]	b[1]	b[2]	b[3]	s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]
0	0	0	0	1	0	1	1	1	1	1
1	0	0	0	0	0	0	0	1	1	0
0	1	0	0	0	1	1	1	0	1	1
1	1	0	0	0	1	0	1	1	1	1
0	0	1	0	1	1	0	0	1	1	0
1	0	1	0	1	1	0	1	1	0	1
0	1	1	0	1	1	1	1	1	0	1
1	1	1	0	0	0	0	0	1	1	1
[illegible]	0	0	[illegible]	[illegible]	[illegible]	1	1	1	1	1
1	0	0	1	1	1	0	1	1	1	1
0	1	0	1	1	1	1	1	1	1	1
1	1	0	1	1	1	0	1	1	1	1
0	0	1	1	1	1	0	1	1	1	1
[illegible]	0	[illegible]	[illegible]	1	1	0	1	1	1	1
[illegible]	[illegible]	[illegible]	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1

Seven Segment Decoder Synthesis

```
// This circuit was obtained from the Truth Table, by BDD synthesis
decode7segs(b:net[4]) = s:net[7]
{
  // Shared expressions, over 1 bit input b[0]
  z = net(0);               // (0)
  u = net(-1);              // (1)
  n = ~ b[0];               // 10   = ~ b[0]
  // Shared expressions, over 2 bits input b[0..1]
  d1 = mux(b[1],z,n);       // 1000 = ~(b[0] | b[1])
  d2 = mux(b[1],n,u);       // 1110 = ~(b[0] & b[1])
  d3 = mux(b[1],n,z);       // 0010 = ~ b[0] & b[1]
  d4 = mux(b[1],u,n);       // 1011 = ~ b[0] | b[1]
  d5 = mux(b[1],n,b[0]);    // 0110 =   b[0] ^ b[1]
  d6 = mux(b[1],b[0],u);    // 1101 = ~ b[1] | b[0]
  d7 = mux(b[1],b[0],n);    // 1001 = ~(b[0] ^ b[1])
  d8 = mux(b[1],u,b[0]);    // 0111 =   b[0] | b[1]
  // Shared expressions, over 3 bits input b[0..2]
  t1 = mux(b[2],d2,d1);     // 1000 1110
  t2 = mux(b[2],d2,b[1]);   // 0011 1110
  t3 = mux(b[2],d5,d4);     // 1011 0110
  t4 = mux(b[2],u,d6);      // 1101 1111
  t5 = mux(b[2],d7,u);      // 1111 1001
  t6 = mux(b[2],d8,d4);     // 1011 0111
  t7 = mux(b[2],d3,n);      // 1010 0010
  // 7 segments output from decoder, with truth tables
  s[0] = mux(b[3],u,t1);    // 10001110 11111111
  s[1] = mux(b[3],u,t2);    // 00111110 11111111
  s[2] = t7;                // 10100010 10100010
  s[3] = mux(b[3],u,t3);    // 10110110 11111111
  s[4] = mux(b[3],u,t4);    // 11011111 11111111
  s[5] = mux(b[3],u,t5);    // 11111001 11111111
  s[6] = mux(b[3],u,t6);    // 10110111 11111111
}
```



b[0]	b[1]	b[2]	b[3]	s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]
0	0	0	0	1	0	1	1	1	1	1
1	0	0	0	0	0	0	0	1	1	0
0	1	0	0	0	1	1	1	0	1	1
1	1	0	0	0	1	0	1	1	1	1
0	0	1	0	1	1	0	0	1	1	0
1	0	1	0	1	1	0	1	1	0	1
0	1	1	0	1	1	1	1	1	0	1
1	1	1	0	0	0	0	0	1	1	1
0	0	0	1	1	1	1	1	1	1	1
1	0	0	1	1	1	0	1	1	1	1
0	1	0	1	1	1	1	1	1	1	1
1	1	0	1	1	1	0	1	1	1	1
0	0	1	1	1	1	0	1	1	1	1
1	0	1	1	1	1	0	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1

OBDD(BCD27SD)

b[3]	b[1]	b[2]	b[0]	s[0]	s[1]	s[2]	s[3]	s[4]	s[5]	s[6]
0	0	0	0	1	0	1	1	1	1	1
1	0	0	0	*1*	*0*	*1*	*1*	*1*	*1*	*1*
0	1	0	0	0	1	1	1	0	1	1
1	1	0	0	*0*	*1*	*1*	*1*	*0*	*1*	*1*
0	0	1	0	1	1	0	0	1	1	0
1	0	1	0	*1*	*1*	*0*	*0*	*1*	*1*	*0*
0	1	1	0	1	1	1	1	1	0	1
1	1	1	0	*1*	*1*	*1*	*1*	*1*	*0*	*1*
0	0	0	1	0	0	0	0	1	1	0
1	0	0	1	*1*	*1*	*0*	*1*	*1*	*1*	*1*
0	1	0	1	0	1	0	1	1	1	1
1	1	0	1	*1*	*1*	*0*	*1*	*1*	*1*	*1*
0	0	1	1	1	1	0	1	1	0	1
1	0	1	1	*1*	*1*	*0*	*1*	*1*	*0*	*1*
0	1	1	1	0	0	0	0	1	1	1
1	1	1	1	*0*	*0*	*0*	*0*	*1*	*1*	*1*

Choose:

1. **Input Order**
2. **Don’t Cares**

BCD27SC:
14 MUX

n	t1	t2	s[0]	t3	t4	s[1]	t5	s[2]	s[3]	s[4]	t6	s[5]	s[6]
1	1	0	**1**	0	0	**0**	1	**1**	**1**	**1**	1	**1**	**1**
1	1	1	**1**	0	0	**0**	1	**1**	**1**	**1**	1	**1**	**1**
0	0	0	**0**	1	1	**1**	1	**1**	**1**	**0**	1	**1**	**1**
0	0	1	**0**	1	1	**1**	1	**1**	**1**	**0**	1	**1**	**1**
1	1	1	**1**	1	1	**1**	0	**0**	**0**	**1**	1	**1**	**0**
1	1	1	**1**	1	1	**1**	0	**0**	**0**	**1**	1	**1**	**0**
0	1	0	**1**	1	0	**1**	1	**1**	**1**	**1**	0	**0**	**1**
0	1	0	**1**	1	0	**1**	1	**1**	**1**	**1**	0	**0**	**1**
1	1	0	**0**	0	0	**0**	1	**0**	**0**	**1**	1	**1**	**0**
1	1	1	**1**	0	0	**0**	1	**0**	**0**	**1**	1	**1**	**0**
0	0	0	**0**	1	1	**1**	1	**0**	**1**	**1**	1	**1**	**1**
0	0	1	**1**	1	1	**1**	1	**0**	**1**	**1**	1	**1**	**1**
1	1	1	**1**	1	1	**1**	0	**0**	**1**	**1**	1	**0**	**1**
1	1	1	**1**	1	1	**1**	0	**0**	**1**	**1**	1	**0**	**1**
0	1	0	**0**	1	0	**0**	1	**0**	**0**	**1**	0	**1**	**1**
0	1	0	**0**	1	0	**0**	1	**0**	**0**	**1**	0	**1**	**1**

```
// This circuit was obtained from the Truth Table, by OBDD synthesis
// The input order b[3], b[1], b[2], b[0] yields 14 muxes (including an inverter).
decode7segs(b:net[4]) = s:net[7]
{
  // Constants: GND=z and VDD=u
  z = net(0);              // (0)
  u = net(-1);             // (1)
  // Shared expressions, over 2 bits input b[3], b[1]
  n = ~ b[1];              // (1100)   = ~ b[1]
  // Shared expressions, over 3 bits input b[3], b[1], b[2]
  t1 = mux(b[2],u,n);       // 1100 1111 = b[2] | ~b[1]
  t2 = mux(b[2],n,b[3]);    // 0101 1100 = b[2]& ~b[1] | ~b[2]&b[3]
  t3 = mux(b[2],u,b[1]);    // 0011 1111 = b[2] | b[1]
  t4 = mux(b[2],n,b[1]);    // 0011 1100 = b[2] ^ b[1]
  t5 = mux(b[2],b[1],u);    // 1111 0011 = ~b[2] | b[1]
  t6 = mux(b[2],n,u);       // 1111 1100 = ~b[1] | ~b[2]
  // 7 segments output from decoder, with truth tables
  s[0] = mux(b[0],t2,t1);   // 11001111 01011100 = b[0]&t2 | ~b[0]&t1
  s[1] = mux(b[0],t4,t3);   // 01111111 01111100 = b[0]&t4 | ~b[0]&t3
  s[2] = mux(b[0],z,t5);    // 11110011 00000000 = ~b[0]&t5
  s[3] = mux(b[0],t4,t5);   // 11110011 01111100 = b[0]&t4 | ~b[0]&t5
  s[4] = mux(b[0],u,t1);    // 11001111 11111111 = b[0] | t1
  s[5] = mux(b[0],t5,t6);   // 11111100 11110011 = b[0]&t5 | ~b[0]&t6
  s[6] = mux(b[0],t3,t5);   // 11110011 01111111 = b[0]&t4 | ~b[0]&t5
}
```

Gate Complexity

Minimal circuit for $f:B^N \rightarrow B$ ***has size*** $C(f)$.

- ***Depends on basis: multiplicative constants***
- ***Most*** optimal circuits ***were found through exhaustive search => small N.***
- ***Little is known about Optimal Sequential Circuits.***

BDD

Widely used for representing Boolean functions

- For all f: $C(f) < {}_c2^N/N$ (proof: OBDD)
- For almost all f: $C(f) > {}_{c'}2^N/N$ (proof: count!)
- So, OBDD is almost always near optimal!
- Size of OBDD depends on the order of inputs:
  - Seven Segment Decode: ***22*** for (0,1,2,3), ***14*** for (3,1,2,0).
  - $BDD(+) = {}_cN$ vs. $BDD(+) = {}_cN^2$
  - $BDD(*) = c^N$, $(c>1)$ for all orders [Bryant]
- Hence, OBDD can be exponentially bad!
- Used for circuit verification rather than synthesis.
- Symbolic representations for integers, sets, relations.