Combinational Logic Building Blocks and Bus Structure

ECE 152A – Summer 2009

Reading Assignment

- Brown and Vranesic
  - 3 Implementation Technology
    - 3.8 Practical Aspects
      - 3.8.7 Passing 1s and 0s Through Transistor Switches
      - 3.8.8 Fan-In and Fan-Out in Logic Gates
        - Tri-State Buffers (only this section of 3.8.8)
    - 3.9 Transmission Gates
      - 3.9.2 Multiplexer Circuit

Reading Assignment

- Brown and Vranesic (cont)
  - 6 Combinational-Circuit Building Blocks
    - 6.1 Multiplexers
      - 6.1.1 Synthesis of Logic Functions Using Multiplexers
      - 6.1.2 Multiplexer Synthesis Using Shannon's Expansion
    - 6.2 Decoders
      - 6.2.1 Demultiplexers
    - 6.3 Encoders
      - 6.3.1 Binary Encoders
      - 6.3.2 Priority Encoders
    - 6.4 Code Converters

Reading Assignment

- Roth
  - 9 Multiplexers, Decoders and Programmable Logic
    - 9.1 Introduction
    - 9.2 Multiplexers
    - 9.3 Three State Buffers
    - 9.4 Decoders and Encoders

Multiplexer

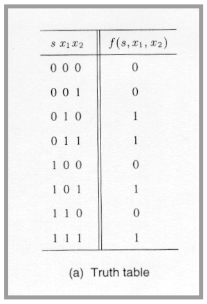
- Passes one of several data inputs to output
  - Generally 2^n data inputs and always a single data output
  - *n* control lines determine which input is "steered" to the output
- Allows logical (not "tri-state" or electrical) implementation of buses
  - Buses and register transfer operations fundamental to digital system design

Multiplexer

- Also possible to implement arbitrary combinational logic with multiplexers
  - Universal, combinational logic element
- Also known as "Data Selector" and "Mux"
- In sequential operation, provides parallel to serial conversion

Two-to-One Multiplexer

- F = *Select’* · x_0 + *Select* · x_1

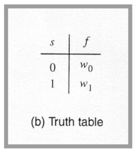


$s\ x_1 x_2$	$f(s, x_1, x_2)$
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

(a) Truth table

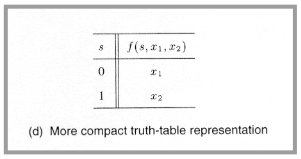
← x1 = w0 →

X2 = w1



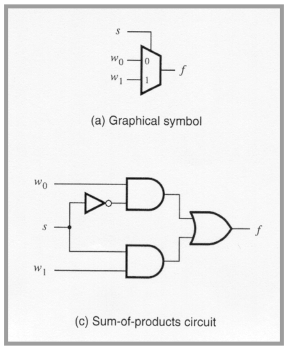
s	f
0	w_0
1	w_1

(b) Truth table



s	$f(s, x_1, x_2)$
0	x_1
1	x_2

(d) More compact truth-table representation

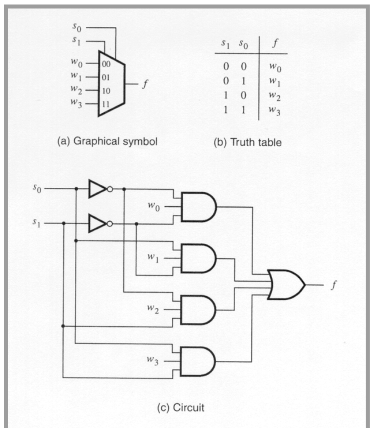


(c) Sum-of-products circuit

Four-to-One Multiplexer

- i^{th} data input ANDed with minterm m_i
  - Embedded circuit generating minterms will become known as a decoder

$$f = \underbrace{\bar{s}_1\bar{s}_0 w_0}_{m_0 w_0} + \underbrace{\bar{s}_1 s_0 w_1}_{m_1 w_1} + \underbrace{s_1\bar{s}_0 w_2}_{m_2 w_2} + \underbrace{s_1 s_0 w_3}_{m_3 w_3}$$

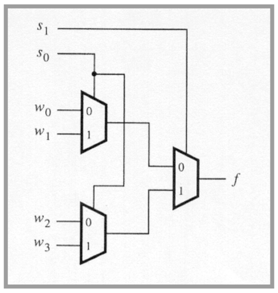


s_1	s_0	f
0	0	w_0
0	1	w_1
1	0	w_2
1	1	w_3

Figure 6.2 A 4-to-1 multiplexer.

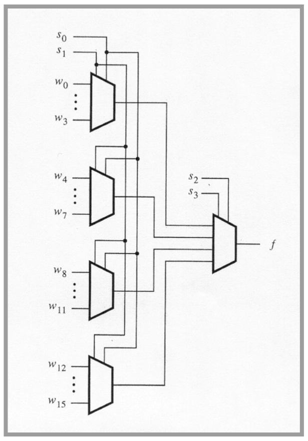
Building Larger Multiplexers

- 4-to-1 (4:1) Mux using 2-to-1 (2:1) Muxes
  - Simple and modular
  - Adds 2 levels of gate (propagation) delay



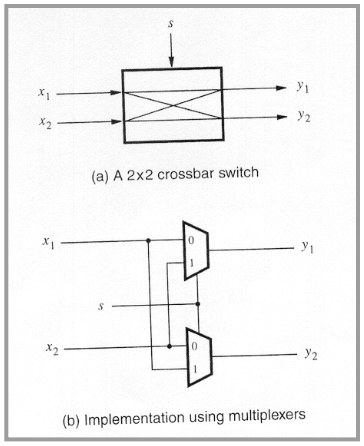
Building Larger Multiplexers

- 16:1 Mux constructed from 4:1 Muxes
  - Expandable to 32:1 and 64:1 with additional 2:1 and/or 4:1 Muxes
    - With additional levels of propagation delay



Multiplexer Application

- Crossbar Switch
  - In general, n-inputs by n-outputs
    - Connectivity is any input to any output
  - Important component of networking hardware
    - The bigger, the faster, the better…



(a) A 2x2 crossbar switch

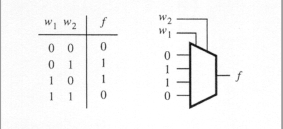
(b) Implementation using multiplexers

Combinational Design Using Multiplexers

- Input variables applied to Mux select lines
  - "Steer" (constant) value of function to output
    - Allows implementation of n-variable function with 2^n-to-1 multiplexer
  - "Steer" derived function (a variable, its complement, the constant 1 or the constant 0) to the output
    - Allows implementation of n-variable function with 2^{n-1}-to-1 multiplexer

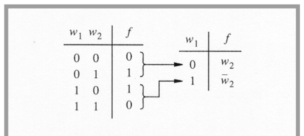
Combinational Design Using Multiplexers

- Example 1: XOR Function
  - Using a 4:1 Mux
  - The modified Truth Table
    - Possibilities are x, x’, 0, 1
  - The 2-input XOR using a 2:1 Mux



w_1	w_2	f
0	0	0
0	1	1
1	0	1
1	1	0

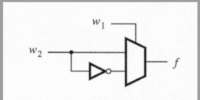
(a) Implementation using a 4-to-1 multiplexer



w_1	w_2	f
0	0	0
0	1	1
1	0	1
1	1	0

w_1	f
0	w_2
1	$\bar{w}_2$

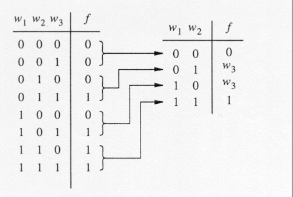
(b) Modified truth table



(c) Circuit

Combinational Design Using Multiplexers

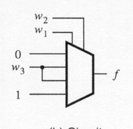
- Example 2 : Three input majority function
  - Three input function with (2^{n-1}-to-1) 4:1 Mux



w_1	w_2	w_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	1
1	1	1	1

w_1	w_2	f
0	0	0
0	1	w_3
1	0	w_3
1	1	1

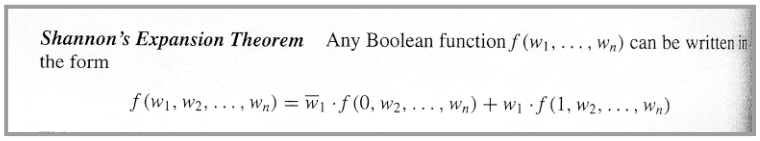
(a) Modified truth table



(b) Circuit

Combinational Design Using Multiplexers

- Multiplexer Synthesis Using Shannon's Expansion
  - By adding gate level circuitry to Mux inputs, an arbitrary combinational function can be realized with a 2-to-1 Mux
    - Externally generating a function of one of the variables



Shannon's Expansion Theorem Any Boolean function $f(w_1, \ldots, w_n)$ can be written in the form

$$f(w_1, w_2, \ldots, w_n) = \overline{w}_1 \cdot f(0, w_2, \ldots, w_n) + w_1 \cdot f(1, w_2, \ldots, w_n)$$

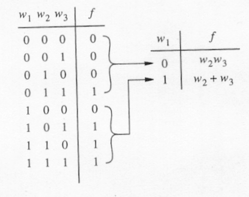
Combinational Design Using Multiplexers

- Example 3 : Three input majority function with 2:1 Mux
  - Algebraic expansion

$$f(w_1, w_2, w_3) = (w_1 w_2 + w_1 w_3 + w_2 w_3)$$
$$f(w_1, w_2, w_3) = (w_1 w_2 + w_1 w_3 + w_2 w_3)(w_1' + w_1)$$
$$f = w_1'(w_1 w_2 + w_1 w_3 + w_2 w_3) + w_1(w_1 w_2 + w_1 w_3 + w_2 w_3)$$
$$\ldots and\ \ from\ Shannon$$
$$f = w_1'(0w_2 + 0w_3 + w_2 w_3) + w_1(1w_2 + 1w_3 + w_2 w_3)$$
$$f = w_1'(w_2 w_3) + w_1(w_2 + w_3)$$

Combinational Design Using Multiplexers

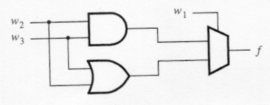
- Example 3: Three input majority function with 2:1 Mux
  - Truth Table and circuit implementation



w_1	w_2	w_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	1
1	1	1	1

w_1	f
0	$w_2 w_3$
1	$w_2 + w_3$

(a) Truth table



(b) Circuit

Combinational Design Using Multiplexers

- Shannon's Expansion with 4:1 Mux
  - Three input majority function
    - Expansion in terms of w_1 and w_2
      - Verifies earlier (heuristic) solution

$$f(w_1, w_2, w_3) = (w_1 w_2 + w_1 w_3 + w_2 w_3)$$

$$f = w_1' w_2' (00 + 0w_3 + 0w_3) + w_1' w_2 (01 + 0w_3 + 1w_3)$$

$$+ w_1 w_2' (10 + 1w_3 + 0w_3) + w_1 w_2 (11 + 1w_3 + 1w_3)$$

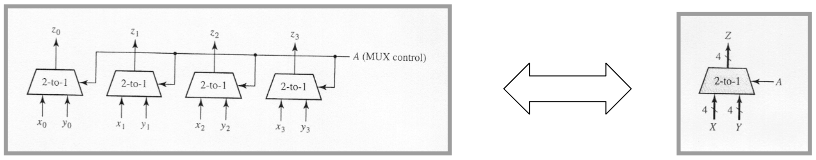
$$f = w_1' w_2' (0) + w_1' w_2 (w_3) + w_1 w_2' (w_3) + w_1 w_2 (1)$$

Multiplexers and Buses

- Bus allows data transfers between multiple sources and single or multiple destinations over a shared path (wires)
  - Bus includes multiple bits
    - Parallel data bus
  - Only one source on the bus at any time
    - Bus contention

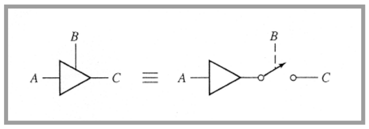
Multiplexers and Buses

- Example below illustrates two, four-bit words (X and Y) multiplexed onto the Z bus
  - Register transfer operations
    - A' : Z ← X , A : Z ← Y



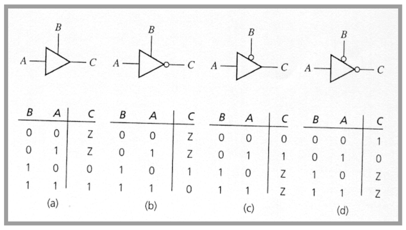
Tri-State Outputs

- Utilizes third, high impedance output state
  - In Hi-Z state, output appears as an open circuit to bus connection
  - Mux disconnects from bus logically, tri-state output device disconnects electrically



Tri-State Outputs (cont)

- Flavors of tri-state outputs and control



B	A	C
0	0	Z
0	1	Z
1	0	0
1	1	1

(a)

B	A	C
0	0	Z
0	1	Z
1	0	1
1	1	0

(b)

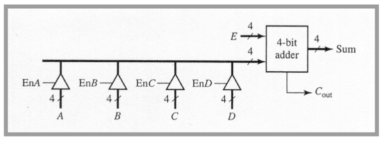
B	A	C
0	0	0
0	1	1
1	0	Z
1	1	Z

(c)

B	A	C
0	0	1
0	1	0
1	0	Z
1	1	Z

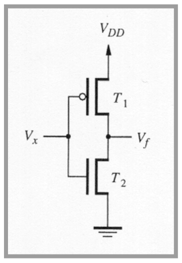
(d)

- Bus implementation



NMOS and PMOS Transistors

- Recall static CMOS circuits
    - Logic high output passed to output through PMOS transistor(s)
        - PMOS transistor passes “good” 1 and “bad” 0
    - Logic low output passed to output through NMOS transistor(s)
        - NMOS transistor passes “good” 0 and “bad” 1
    - “Good” 0s and 1s are GND and V_{DD}
    - “Bad” 0s and 1s have degraded DC voltage (logic) levels

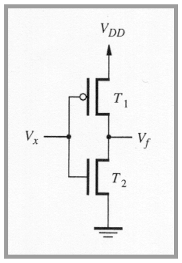


NMOS and PMOS Transistors

- Degradation of DC signal levels is a result of the “threshold voltage” (V_T) of transistor and the “body effect”
    - To “turn on” the transistor, the gate to source voltage (V_{GS}) must exceed the transistor’s threshold voltage (V_T)
        - An NMOS transistor has a positive V_T
        - A PMOS transistor has a negative V_T
    - The threshold voltage itself is increased by the body effect by a factor of ~1.5

NMOS and PMOS Transistors

- For the inverter below, assume the NMOS device has a V_T of 1V ($V_{GS} > 1V$) and the PMOS device has a V_T of -1V ($V_{GS} < -1V$) and V_{DD} = 5V

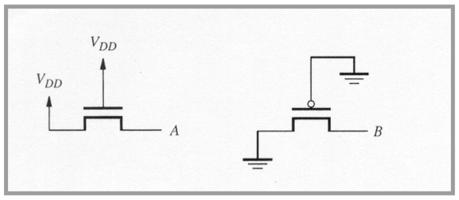


Input = 5V, V_{GS} (T1) = 5V (off), V_{GS} (T2) = 5V (on)
Output = 0V (GND)

Input = 0V, V_{GS} (T1) = -5V (on), V_{GS} (T2) = 0V (off),
Output = 5V (V_{DD})

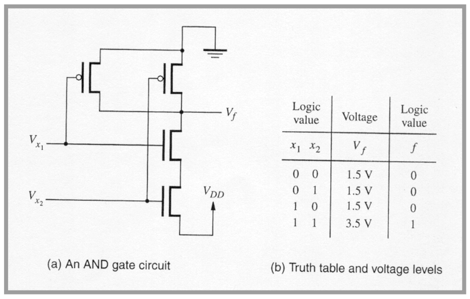
NMOS and PMOS Transistors

- Bad 1s (NMOS) and Bad 0s (PMOS)
  - $V_A = V_{DD} – V_T$ (NMOS)
    - Input going high; turns off at $V_{GS} = V_T$
  - $V_B = -V_T$ (PMOS)
    - Input going low; turns off at $V_{GS} = V_T$



CMOS AND Gate

- Note degradation in DC signal (logic) levels
  - AND Gates are <u>never</u> built this way in CMOS



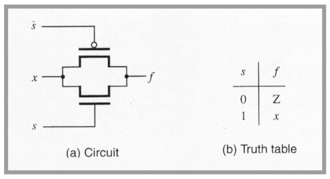
(a) An AND gate circuit

Logic value		Voltage	Logic value
x_1	x_2	V_f	f
0	0	1.5 V	0
0	1	1.5 V	0
1	0	1.5 V	0
1	1	3.5 V	1

(b) Truth table and voltage levels

The CMOS Transmission Gate

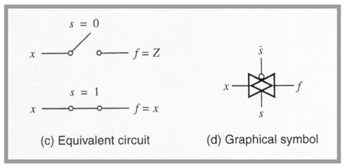
- When enabled, the CMOS Transmission Gate:
  - Passes "good" 1s (through the PMOS transistor)
  - Passes "good" 0s (through the NMOS transistors)
- When disabled, the CMOS Transmission gate acts like a Tri-State Buffer



(a) Circuit

s	f
0	Z
1	x

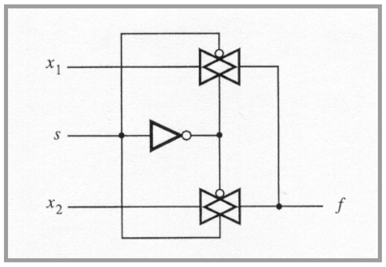
(b) Truth table



(c) Equivalent circuit (d) Graphical symbol

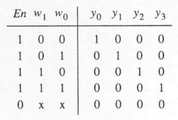
CMOS Transmission Gate MUX

- 2:1 Multiplexer implementation with transmission gates



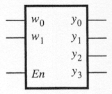
Decoders

- 2-to-4 Decoder shown
  - 2-to-2^n in general
  - Enable input allows construction of decoder tree and demultiplexer
  - Generates all minterms when enabled
    - Multiple output circuits
  - One hot decoding

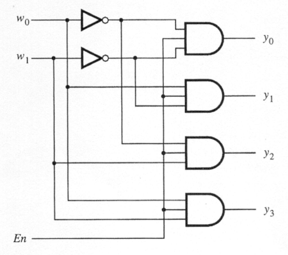


En	w_1	w_0	y_0	y_1	y_2	y_3
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1
0	x	x	0	0	0	0

(a) Truth table



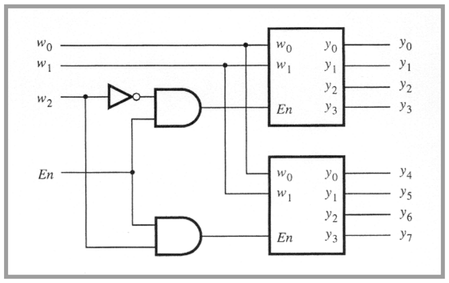
(b) Graphical symbol



(c) Logic circuit

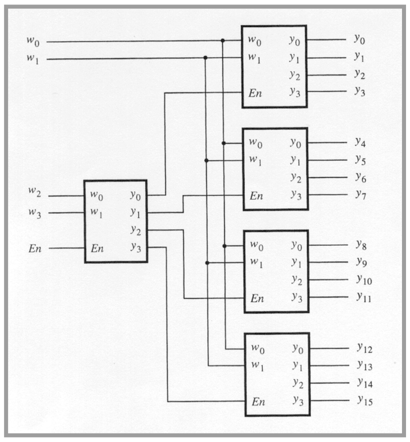
Decoder Tree

- One-bit expansion (3-to-8) by adding external decoding circuitry



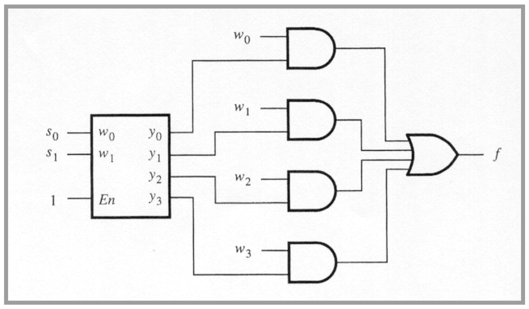
Decoder Tree

- Two-bit expansion (4-to-16) by adding another 2-to-4 decoder



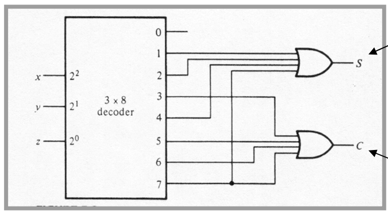
Decoder Applications

- Multiplexer from decoder
  - Recall “embedded decoder”



Decoder Applications

- Multiple Output Circuits
  - Full Adder using 3 X 8 Decoder

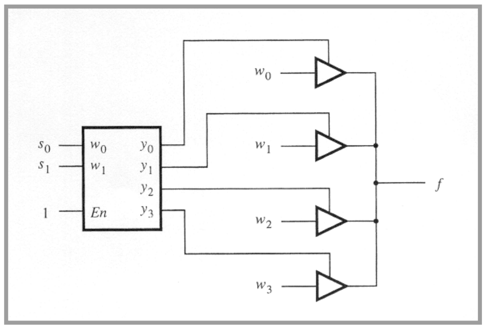


Sum = m1 + m2 + m4 + m7
= x’y’z + x’yz’ + xy’z’ + xyz

Carry = m3 + m5 + m6 + m7
= x’yz + xy’z + xyz’ + xyz

Decoder Applications

- Decoder Bus Control/Multiplexer

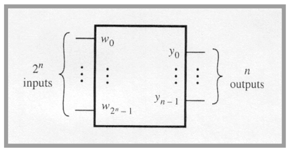


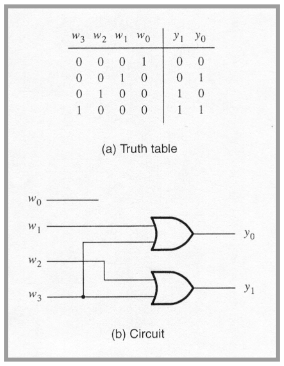
Demultiplexers

- Serial to parallel conversion
  - Send a single data bit to a specific address
- A 1-to-2^n demultiplexer is implemented using an n-to-2^n decoder
  - The (value of) the data is applied via the enable input
- Valuable circuit in sequential circuits
  - Not so much in combinational circuits
- Also referred to as Dmux's

Encoders

- Binary Encoders
  - “One hot” input, binary (or other code) representation output
    - Reverse of decoder





w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

(a) Truth table

(b) Circuit

Encoders

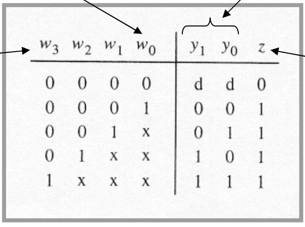
- Priority Encoders
  - Used in prioritizing interrupts (or other events)

Least significant

Binary encoding

Most significant

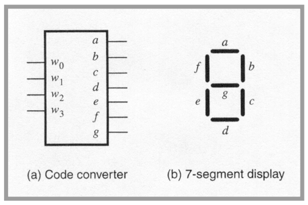
Output valid



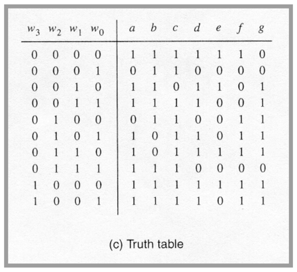
w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

Code Converters

- BCD to 7-Segment Display Code Converter



(a) Code converter (b) 7-segment display



w_3	w_2	w_1	w_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

(c) Truth table