

Luento 3

Tietokoneen rakenne

Digital logic

Stallings: Appendix B

- n Boolean Algebra
- n Combinational Circuits
- n Simplification
- n Sequential Circuits

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 1

Tietokoneen rakenne

Boolean Algebra

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 2

Boolean Algebra

- n **George Boole**
 - u ideas 1854
- n **Claude Shannon (kuva) (gradu)**
 - u apply to circuit design, 1938
 - u "father of information theory"

Topics:

- n **Describe digital circuitry function** (piirisuunnittelu)
 - u programming language?
- n **Optimise given circuitry**
 - u use algebra (Boolean algebra) to manipulate (Boolean) expressions into simpler expressions

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 3

Boolean Algebra

- n **Variables: A, B, C**
- n **Values: TRUE (1), FALSE (0)**
- n **Basic logical operations:**
 - u binary: AND (·) $A \cdot B = AB$ ja / tai / ei
 - u binary: OR (+) $B + C$ product / sum
 - u unary: NOT (̄) \bar{A} negation
- n **Composite operations, equations**
 - u precedence: NOT, AND, OR
 - u parenthesis
$$D = A + \bar{B} \cdot C = A + ((\bar{B})C)$$

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 4

Boolean Algebra

- n **Other operations**
 - u XOR (exclusive-or)
 - u NAND $A \text{ NAND } B = \text{NOT}(A \text{ AND } B) = \overline{AB}$
 - u NOR $A \text{ NOR } B = \text{NOT}(A \text{ OR } B) = \overline{A + B}$
- n **Truth tables**

Boolean Operators

P	Q	NOT P	P AND Q	P OR Q	P XOR Q	P NAND Q	P NOR Q
0	0	1	0	0	0	1	1
0	1	1	0	1	1	1	0
1	0	0	0	1	1	1	0
1	1	0	1	1	0	0	0

(Sta06 Table B.1)

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 5

Postulates and Identities

n How can I manipulate expressions?

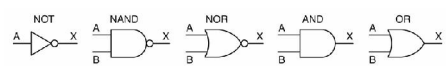
u Simple set of rules?

Basic Postulates		
$A \cdot B = B \cdot A$	$A + B = B + A$	Commutative Laws
$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$	$A + (B \cdot C) = (A + B) \cdot (A + C)$	Distributive Laws
$1 \cdot A = A$	$0 + A = A$	Identity Elements
$A \cdot \bar{A} = 0$	$A + \bar{A} = 1$	Inverse Elements
Other Identities		
$0 \cdot A = 0$	$1 + A = 1$	alkion ja komplementin tulo ja summa
$A \cdot A = A$	$A + A = A$	tulo 0'n kanssa, summa 1'n kanssa
$A \cdot (B \cdot C) = (A \cdot B) \cdot C$	$A + (B + C) = (A + B) + C$	liitännäislaajat
$\bar{\bar{A}} = A$	$\overline{A + B} = \bar{A} \cdot \bar{B}$	DeMorgan's Theorem
$\bar{A} \cdot B = \overline{A + \bar{B}}$	$\bar{A} + B = \overline{A \cdot \bar{B}}$	

(Sta06 Table B.2)

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 6

Gates (veräjät / portit)



- Implement basic Boolean algebra operations
- Fundamental building blocks
 - 1 or 2 inputs, 1 output
- Combine to build more complex circuits
 - memory, adder, multiplier, ...
- Gate delay
 - change inputs, after gate delay new output available
 - 1 ns? 10 ns? 0.1 ns?

yhteenlaskupiiri, kertolaskupiiri

<http://tech-www.informatik.uni-hamburg.de/applets/cmos/cmosdemo.html> (extra material)

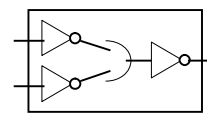
Sta06 Fig B.1

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 7

Functionally Complete Set

Funktionaalisesti täydellinen joukko => joukosta voidaan muodostaa kaikki portit

- Can build all basic gates (AND, OR, NOT) from a smaller set of gates
 - With AND, NOT (Nämä seuraavat suoraan DeMorganin kaavoista)
 - With OR, NOT
 - With NAND alone
 - With NOR alone

$$A + B = \overline{\overline{A} \cdot \overline{B}}$$


OR with AND and NOT gates

Sta06 Fig B.2, B.3

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 8

Combinational Circuits yhdistelmäpiirit

- Interconnected set of gates
 - m inputs, n outputs
 - change inputs, wait for gate delays, new outputs
- Each output
 - depends on combination of input signals
 - can be expressed as Boolean function of inputs
- Function can be described in three ways
 - with Boolean equations (one equation for each output)
 - with truth table
 - with graphical symbols for gates and wires

Sta06 Fig B.4

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 9

Describing the Circuit

- Boolean equations

$$F = \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\overline{C}$$
- Truth table

inputs			output
A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0
- Graphical symbols

Sta06 Fig B.4

(Sta06 Table B.3)

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 10

Tietokoneen rakenne

Simplification Piirin yksinkertaistaminen

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 11

Simplify Presentation (and Implementation)

- Boolean equations
 - Sum of products form (SOP) tulojen summa

$$F = \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\overline{C}$$
 - Product of sums form (POS) summien tulo

$$F = (A + B + C) \cdot (A + B + \overline{C}) \cdot (\overline{A} + B + C) \cdot (\overline{A} + B + \overline{C}) \cdot (\overline{A} + \overline{B} + C)$$

Boolean algebra

Sta06 Table B.3

Sta06 Fig B.4

Sta06 Fig B.5

- Which presentation is better?
 - Fewer gates? Smaller area on chip?
 - Smaller circuit delay? Faster?

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 12

Algebraic Simplification

- Circuits become too large to handle?
- Use basic identities to simplify Boolean expressions

$$F = \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\overline{C}$$

$$= \overline{A}B(\overline{C} + C) + A\overline{B}\overline{C}$$

$$= \overline{A}B + A\overline{B}\overline{C}$$

Sta06 Fig B.4

Sta06 Fig B.6

- May be difficult to do!
- How to do it automatically?
- Build a program to do it "best"?

$$f = \overline{a}bcd + \overline{a}bc\overline{d} + \overline{a}b\overline{c}d + \overline{a}b\overline{c}\overline{d}$$

$$+ abcd + abc\overline{d} + ab\overline{c}d + ab\overline{c}\overline{d}$$

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 13

How so?

$$F = \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\overline{C}$$

$$= \overline{A}B\overline{C} + \overline{A}BC + \overline{A}B\overline{C} + A\overline{B}\overline{C}$$

$$= (\overline{A}B\overline{C} + \overline{A}BC) + (\overline{A}B\overline{C} + A\overline{B}\overline{C})$$

$$= \overline{A}B(\overline{C} + C) + (\overline{A} + A)\overline{B}\overline{C}$$

$$= \overline{A}B(1) + (1)\overline{B}\overline{C}$$

$$= \overline{A}B + \overline{B}\overline{C}$$

$$= B(\overline{A} + \overline{C})$$

Boolean algebra:
A+A=A

And this? $f = \overline{a}bcd + \overline{a}bc\overline{d} + \overline{a}b\overline{c}d + \overline{a}b\overline{c}\overline{d}$

Entäs tämä? $+ abcd + abc\overline{d} + ab\overline{c}d + ab\overline{c}\overline{d}$

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 14

Karnaugh Map

Karnaugh kartta

- Represent Boolean function (i.e., circuit) truth table in another way
 - Use canonical form: each term has each variable once
 - Use SOP presentation
- Karnaugh map squares
 - Each square is one product (input value combination)
 - Value is one (1) iff the product is present o/w value is "empty"

(a) $F = A\overline{B} + \overline{A}B$

(b) $F = \overline{A}\overline{B}C + \overline{A}BC + A\overline{B}\overline{C}$

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 15

Karnaugh Map

- Adjacent squares differ only in one input value (wrap around)
- Square for input combination $\overline{A}\overline{B}\overline{C}D = 1001$

(c) $F = \overline{A}\overline{B}C\overline{D} + \overline{A}\overline{B}CD + A\overline{B}\overline{C}\overline{D}$

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 16

Karnaugh Map Simplification

- If adjacent squares have value 1, input values differ only in one variable
- Value of that variable is irrelevant (when all other input variables are fixed for those squares)
- Can ignore that variable for those expressions
 - ... + $\overline{A}B\overline{C}D + \overline{A}BCD + \dots$ ignore C ... + $\overline{A}BD + \dots$

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 17

Using Karnaugh Maps to Minimize Boolean Functions (8)

Original function

$$f = \overline{a}bcd + \overline{a}bc\overline{d} + \overline{a}b\overline{c}d + \overline{a}b\overline{c}\overline{d}$$

$$+ abcd + abc\overline{d} + ab\overline{c}d + ab\overline{c}\overline{d}$$

Canonical form (already OK)

Karnaugh Map

Find smallest number of circles, each with largest number (2¹) of 1's

- can wrap-around

Select parameter combinations corresponding to the circles

Get reduced function $f = bd + ac + ab$

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 18

Impossible Input Variable Combinations (3)

What if some input combinations can never occur?

- Mark them "don't care", "d"
- Treat them as 0 or 1, whichever is best for you
- More room to optimize

$f = bd + a$

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 19

Example: Circuit to add 1 (mod 10) to 4-bit BCD decimal number (3)

5 = 0101 → 0110 = 6
9 = 1001 → 0000 = 0

Truth table?
Karnaugh maps for W, X, Y and Z?

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 20

Example cont.: Truth Table

Input				Output					
Number	A	B	C	D	Number	W	X	Y	Z
0	0	0	0	0	1	0	0	0	1
1	0	0	0	1	2	0	0	1	0
2	0	0	1	0	3	0	0	1	1
3	0	0	1	1	4	0	1	0	0
4	0	1	0	0	5	0	1	0	1
5	0	1	0	1	6	0	1	1	0
6	0	1	1	0	7	0	1	1	1
7	0	1	1	1	8	1	0	0	0
8	1	0	0	0	9	1	0	0	1
9	1	0	0	1	0	0	0	0	0
Don't care condition	1	0	1	0		d	d	d	d
	1	0	1	1		d	d	d	d
	1	1	0	0		d	d	d	d
	1	1	0	1		d	d	d	d
	1	1	1	0		d	d	d	d
	1	1	1	1		d	d	d	d

(Sta06 Table B.4) No carry!

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 21

Example cont: Karnaugh Map

(Sta06 Table B.4) (Sta06 Fig B.10)

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 22

Other Methods to simplify Boolean expressions

Why?

- Karnaugh maps become complex with 6 input variables

Quine-McKluskey method

- Tabular method
- Automatically suitable for programming

Luque Method click

- Based on dividing circle in different ways
- Can be fractally expanded to infinitely many variables

Interesting, but not part of this course
Details skipped

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 23

Tietokoneen rakenne

Basic Combinatorial Circuits

Building blocks for more complex circuits

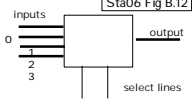
- Multiplexer
- Encoders/decoder
- Read-Only-Memory
- Adder

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 24

Multiplexers

limitin

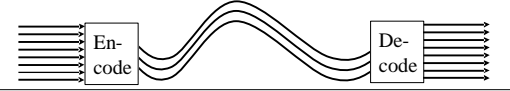
- Select one of many possible inputs to output
 - black box
 - truth table [Sta06 Table B.7]
 - implementation [Sta06 Fig B.12]
- Each input/output "line" can be many parallel lines
 - select one of three 16 bit values
 - $C_{0,15}, IR_{0,15}, ALU_{0,15}$
 - simple extension to one line selection [Sta06 Fig B.14]
 - lots of wires, plenty of gates ...
- Used to control signal and data routing
 - Example: loading the value of PC



Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 25

Encoders/Decoders

- Exactly one of many Encoder input or Decoder output lines is 1
- Encode that line number as output
 - hopefully less pins (wires) needed this way
 - optimise for space, not for time [space-time tradeoff]
- Example: [Sta06 Fig B.15]
 - encode 8 input wires with 3 output pins
 - route 3 wires around the board
 - decode 3 wires back to 8 wires at target
 - Ex. Choosing the right memory chip from the address bits.



Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 26

Read-Only-Memory (ROM) (5)

- Given input values, get output value
 - Like multiplexer, but with fixed data
- Consider input as address, output as contents of memory location
- Example
 - Truth tables for a ROM [Sta06 Table B.8]

Mem (7) =	4
Mem (11) =	14

 - 64 bit ROM
 - 16 words, each 4 bits wide
 - Implementation with decoder & or gates [Sta06 Fig B.20]

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 27

Adders

- 1-bit adder

A=1	→	?	→	Carry=0
B=0	→		→	Sum=1
- 1-bit adder with carry

Carry=1	→	?	→	Carry=1
A=1	→		→	Sum=0
B=0	→		→	
- Implementation [Sta06 Table B.9, Fig B.22] Compare to ROM?
- Build a 4-bit adder from four 1-bit adders [Sta06 Fig B.21]

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 28

Tietokoneen rakenne

Sequential Circuits

sarjalliset piirit

- Flip-Flop
- S-R Latch
- Registers
- Counters

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 29

Sequential Circuit (sarjallinen piiri)

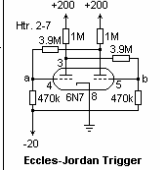
- Circuit has (modifiable) internal state
 - remembers its previous state
- Output of circuit depends (also) on internal state
 - not only from current inputs
 - output = $f_o(\text{input}, \text{state})$
 - new state = $f_s(\text{input}, \text{state})$
- Circuits needed for
 - processor control
 - registers
 - memory

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 30

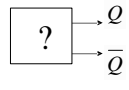
Flip-Flop (kiikku)

http://www.du.edu/~etuttle/electron/elect36.htm

- William Eccles & F.W. Jordan
 - with vacuum tubes, 1919
- 2 states for Q (0 or 1, true or false)
- 2 outputs
 - complement values
 - both always available on different pins
- Need to be able to change the state (Q)



Eccles-Jordan Trigger



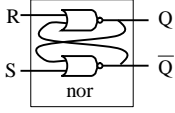
Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 31

S-R Flip-Flop or S-R Latch (salpa)

Usually both 0 $\begin{cases} R=0 \\ S=0 \end{cases} \rightarrow \begin{cases} Q \\ \bar{Q} \end{cases}$

S = "SET" = "Write 1" = "set S=1 for a short time"
 R = "RESET" = "Write 0" = "set R=1 for a short time"

nor (0, 0) = 1
nor (0, 1) = 0
nor (1, 0) = 0
nor (1, 1) = 0

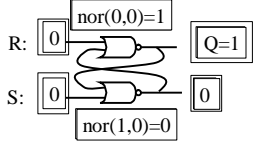


Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 32

S-R Latch Stable States (4)

- 1 bit memory (value = value of Q)
- bistable, when R=S=0
 - Q=0?
 - Q=1?

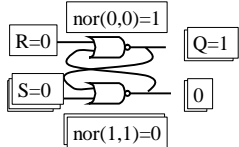
nor (0, 0) = 1
nor (0, 1) = 0
nor (1, 0) = 0
nor (1, 1) = 0



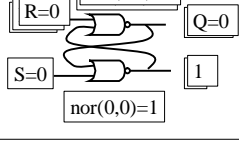
Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 33

S-R Latch Set (=1) and Reset (=0) (17)

Write 1: S= 0 → 1 → 0



Write 0: R= 0 → 1 → 0

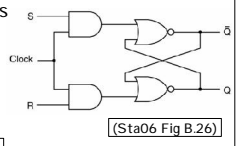


nor (0, 0) = 1
nor (0, 1) = 0
nor (1, 0) = 0
nor (1, 1) = 0

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 34

Clocked Flip-Flops

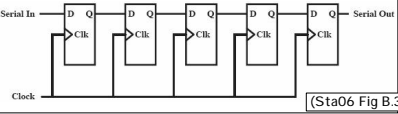
- State change can happen only when clock is 1
 - more control on state changes
- Clocked S-R Flip-Flop
- D Flip-Flop
 - only one input D (Sta06 Fig B.27)
 - D = 1 and CLOCK → write 1
 - D = 0 and CLOCK → write 0
- J-K Flip-Flop (Sta06 Fig B.28)
 - Toggle Q when J=K=1 (Sta06 Fig B.29)



Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 35

Registers

- Parallel registers (Sta06 Fig B.30)
 - read/write
 - CPU user registers
 - additional internal registers
- Shift Registers
 - shifts data 1 bit to the right
 - serial to parallel?
 - ALU ops?
 - rotate?



Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 36

Counters

- n Add 1 to stored counter value
- n Counter
 - u parallel register plus increment circuits
- n Ripple counter (aalto, viive)
 - u asynchronous
 - u increment least significant bit, and handle "carry" bit as far as needed
- n Synchronous counter
 - u modify all counter flip-flops simultaneously
 - u faster, more complex, more expensive
 - u space-time tradeoff

Sta06 Fig B.32

A 4-bit synchronous "up" counter

(http://www.faboulcircuits.com)

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 37

Summary

- n Boolean Algebra & Gates & Circuits
 - u can implement all with NANDs or NORs
 - u simplify circuits:
 - § Karnaugh, (Quine-McKluskey, Luque, ...)
- n Components for CPU design
 - u ROM, adder
 - u multiplexer, encoder/decoder
 - u flip-flop, register, shift register, counter

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 38

-- End of Appendix B: Digital Logic --

Simple processor

http://www.gamezero.com/team-0/articles/math_magic/micro/stage4.html

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 39

Kertauskysymyksiä/Review questions

- n DeMorganin laki?
 - n Miten boolean funktio minimoidaan Karnaugh-kartan avulla?
 - n Mitä eroa sarjallisessa piirissä on verrattuna "normaaliin" kombinatoriseen piiriin?
 - n Miten S-R kiikku toimii?
- n DeMorgan's theorem?
 - n How to minimize a Boolean function using Karnaugh's map?
 - n How do sequential circuits differ from 'normal' combinational circuits?
 - n How does the S-R flip-flop function?

Computer Organization II / 2007 / Liisa Marttinen 5.11.2007 Lecture 3 - 40