



Task 2: Combinatorial logic digital circuit design in Verilog

Design a combinatorial circuit that implements given Boolean functions

Individual variant: two Boolean functions composed on letters of the student name and surname; a restricted base.

Subtasks:

- 1) Create a truth table and compose a Perfect Disjunctive Normal Form (PDNF) for each of two given functions.
- 2) Use a Karnaugh maps to minimize the Boolean functions, compose minimal DNFs.
- 3) Compose a Verilog module to compute functions on arguments.
- 4) Use online Verilog to obtain graphical layout of the circuit.
- 5) Implement manual testing of the digital circuit.
- 6) Synthesize circuit for the same functions using given restricted base with limitations on the number of inputs. Specify the given base functions as Verilog modules and use them in the upper layer module.

Optional study: automatic testing of circuits in Verilog.

Directions:

- At first, work in conventional basis: and, or, not.
- Treat undefined values ("*") of Boolean function either as 1 or 0 depending on usefulness for minimization purposes
- When minimizing, define undefined values to facilitate minimization.
- Save Verilog programs with your comments.
- Try to draw a circuit layout by hand before obtaining automatic layout from Verilog.
- Do manual testing to all combinations of independent variables.

Questions to muse: how to minimize functions in other bases, say for Zhegalkin polynomial

References:

- Lecture 3 – Digital circuits design with Verilog. Combinatorial logic
- Overview of propositional logic

Supplemental materials: Overview of propositional logic

Task variants:

We consider Boolean functions depending on 4 independent Boolean variables.

Function 1: take 8 character of your name, compute the function's 1s, based on 5 first characters and undefined values ("*") based on the last 3 characters.

Function 2: take 8 character of your surname, compute the function's 1s, based on 5 first characters and undefined values ("*") based on the last 3 characters.

To compute the argument set number on a character: take the character number in alphabet and compute the number modulo 16.

Restricted bases (repeated for 6-10):

Stud No	Base
1	Peirce arrow
2	Sheffer Stroke
3	And, not
4	Or, not
5	Implication, false

Restricted inputs:

Stud No	Number of inputs
1-5	3
6-10	2

Note: "*" overlaps "1".

Example: "Dmitry", "Zaitsev"

Extended by father name and mother surname: N="DmitryAnatoly", S="ZaitsevBeliaieva"

Function 0:

i	1	2	3	4	5	6	7	8
N[i]	d	m	i	t	r	y	a	n
A(N[i])	4	13	9	20	18	25	1	14
A(N[i]) mod 16	4	13	9	4	2	9	1	14
Variable set	0100	1101	1001	0100	0010	1001	0001	1110

Function 1:

i	1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---	---

S[i]	z	a	i	t	s	e	v	b
A(S[i])	26	1	9	20	19	6	22	2
A(S[i]) mod 16	10	1	9	4	3	6	6	2
Variable set	1010	0001	1001	0100	0011	0110	0110	0010

Truth table:

Vector \mathbf{x}				f_1	f_0
x_3	x_2	x_1	x_0		
0	0	0	0	0	0
0	0	0	1	*	1
0	0	1	0	1	*
0	0	1	1	0	1
0	1	0	0	1	1
0	1	0	1	0	0
0	1	1	0	0	*
0	1	1	1	0	0
1	0	0	0	0	0
1	0	0	1	*	1
1	0	1	0	0	1
1	0	1	1	0	0
1	1	0	0	0	0
1	1	0	1	1	0
1	1	1	0	*	0
1	1	1	1	0	0

