ganeshprabhu2005 / **BOOLEAN_FUNCTION_MINIMIZATION**   Public

forked from naavaneetha/BOOLEAN_FUNCTION_MINIMIZATION

⚖ GPL-3.0 license

☆ **0** stars   **374** forks   Branches   Tags   Activity

☆ Star   ▽

🔔 Notifications

<> **Code**   Pull requests   ▷ Actions   Projects   Security   Insights

ᛘ main ▾ | ᛘ **1** Branch | ◌ **0** Tags | ᛘ | ◌ | 🔍 Go to file | Go to file | Code | •••

This branch is [3 commits ahead of](#) `naavaneetha/BOOLEAN_FUNCTION_MINIMIZATION:main` .

🟣 **ganeshprabhu2005** Update README.md   2 weeks ago ••• 🕘

📁 db	EXP2	3 months ago
📁 hc_output	EXP2	3 months ago
📁 incremental_db	EXP2	3 months ago
📁 output_files	EXP2	3 months ago
📁 simulation	EXP2	3 months ago
📄 Boolean_min.qpf	EXP2	3 months ago
📄 Boolean_min.qsf	EXP2	3 months ago
📄 Boolean_min.qws	EXP2	3 months ago
📄 Boolean_min.v	EXP2	3 months ago
📄 Boolean_min.v.bak	EXP2	3 months ago
📄 LICENSE	Initial commit	3 months ago
📄 README.md	Update README.md	2 weeks ago
📄 Waveform2.vwf	EXP2	3 months ago
📄 c5_pin_model_dump.txt	EXP2	3 months ago
📄 cio_dump_disallowed_list...	EXP2	3 months ago

📖 **README**   ⚖️ License

BOOLEAN_FUNCTION_MINIMIZATION

AIM:

To implement the given logic function verify its operation in Quartus using Verilog programming.

F1= A′B′C′D′+AC′D′+B′CD′+A′BCD+BC′D

F2=xy′z+x′y′z+w′xy+wx′y+wxy

Equipment Required:

Hardware – PCs, Cyclone II , USB flasher

Software – Quartus prime

Theory

Logic Diagram

Procedure

1. Type the program in Quartus software.

2. Compile and run the program.

3. Generate the RTL schematic and save the logic diagram.

4. Create nodes for inputs and outputs to generate the timing diagram.

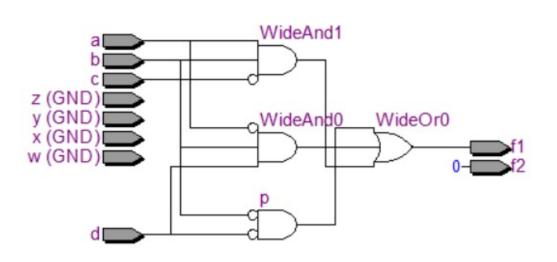5. For different input combinations generate the timing diagram.

Program:

/* Program to implement the given logic function and to verify its operations in quartus using Verilog programming.

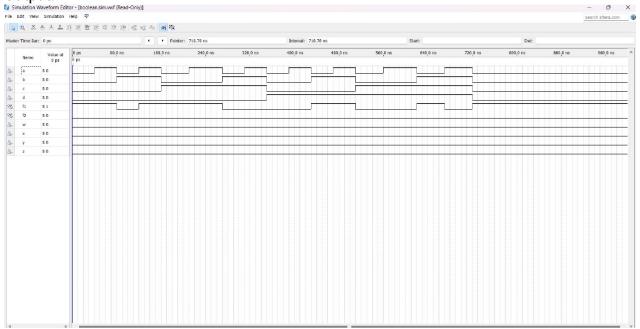Developed by: GANESH PRABHU J RegisterNumber: 212223220023

```
module booleanfunction_top(a,b,c,d,w,x,y,z,f1,f2);
input a,b,c,d,w,x,y,z;
output f1,f2;
wire adash,bdash,cdash,ddash,ydash,p,q,r,s,t,u;
not(adash,a);
not(bdash,b);
not(cdash,c);
not(ddash,d);
not(ydash,y);
and(p,bdash,ddash);
and(q,adash,b,d);
and(r,a,b,cdash);
or(f1,p,q,r);
```

//type code for f2 as like f1 endmodule

RTL realization

Output:

RTL

A	B	C	D	E	F	G	H	I	J
A	B	C	D	A'B'C'D'	AC'D'	B'CD'	A'BCD	BC'D	F1
0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	1
0	0	1	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	1	1
0	1	1	0	0	0	0	0	0	0
0	1	1	1	0	0	0	1	0	1
1	0	0	0	0	1	0	0	0	1
1	0	0	1	0	0	0	0	0	0

1	0	1	0	0	0	1	0	0	1
1	0	1	1	0	0	0	0	0	0
1	1	0	0	0	1	0	0	0	1
1	1	0	1	0	0	0	0	1	1
1	1	1	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0

Timing Diagram

Result:

Thus the given logic functions are implemented using and their operations are verified using Verilog programming.

Releases

No releases published

Packages

No packages published

Languages

- **VHDL** 51.8%
- **Verilog** 19.1%
- **Stata** 13.6%
- **HTML** 13.1%
- **Standard ML** 2.4%