Software Engineering

Project I

Team 28

Team 28

- Shivam Khandelwal
- Tarun Gupta
- Sumit Kumar Jha
- Anurag Tyagi

Problem Statement

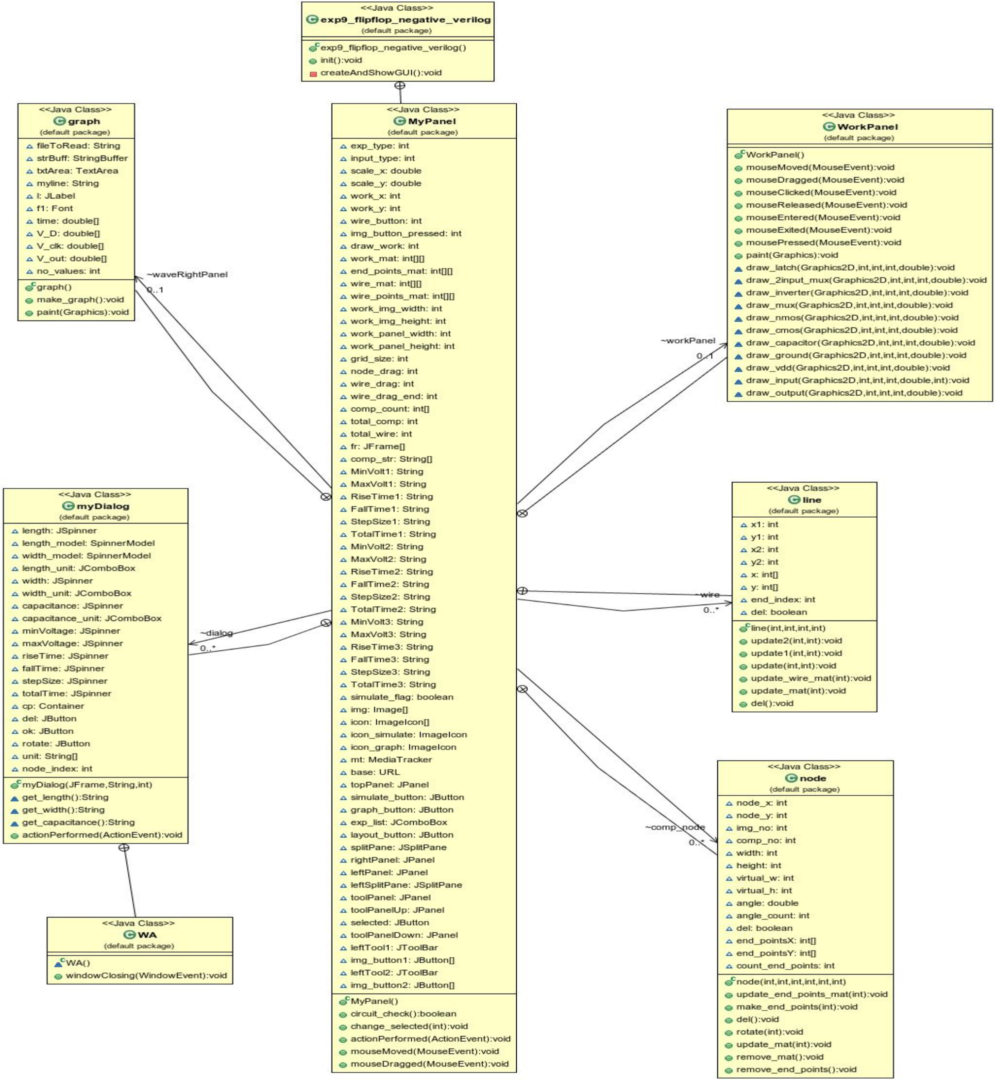
- Current experiments are not compatible on various browsers and do not support responsive design.
- Experiments require a plug-in to be installed on client side and this impacts the user experience.
- No due consideration was given to maintainability, thus making the code difficult to enhance and maintain.
- Existing code has client side dependencies.
- Existing code has very low code quality as indicated by code quality metrics.

VLSI Lab : Design Of D-Flip Flop Using Verilog

VLSI Lab : I erilog

Class Diagrams

Negative Edge Tri



<<Java Class>>
exp9_flipflop_negative_verilog
(default package)

- exp9_flipflop_negative_verilog()
- init():void
- createAndShowGUI():void

<<Java Class>>
graph
(default package)

- fileToRead: String
- strBuff: StringBuffer
- txtArea: TextArea
- myline: String
- l: JLabel
- f1: Font
- time: double[]
- V_D: double[]
- V_clk: double[]
- V_out: double[]
- no_values: int
- graph()
- make_graph():void
- paint(Graphics):void

<<Java Class>>
MyPanel
(default package)

- exp_type: int
- input_type: int
- scale_x: double
- scale_y: double
- work_x: int
- work_y: int
- wire_button: int
- img_button_pressed: int
- draw_work: int
- work_mat: int[][]
- end_points_mat: int[][]
- wire_mat: int[][]
- wire_points_mat: int[][]
- work_img_width: int
- work_img_height: int
- work_panel_width: int
- work_panel_height: int
- grid_size: int
- node_drag: int
- wire_drag: int
- wire_drag_end: int
- comp_count: int[]
- total_comp: int
- total_wire: int
- fr: JFrame[]
- comp_str: String[]
- MinVolt1: String
- MaxVolt1: String
- RiseTime1: String
- FallTime1: String
- StepSize1: String
- TotalTime1: String
- MinVolt2: String
- MaxVolt2: String
- RiseTime2: String
- FallTime2: String
- StepSize2: String
- TotalTime2: String
- MinVolt3: String
- MaxVolt3: String
- RiseTime3: String
- FallTime3: String
- StepSize3: String
- TotalTime3: String
- simulate_flag: boolean
- img: Image[]
- icon: ImageIcon[]
- icon_simulate: ImageIcon
- icon_graph: ImageIcon
- mt: MediaTracker
- base: URL
- topPanel: JPanel
- simulate_button: JButton
- graph_button: JButton
- exp_list: JComboBox
- layout_button: JButton
- splitPane: JSplitPane
- rightPanel: JPanel
- leftPanel: JPanel
- leftSplitPane: JSplitPane
- toolPanel: JPanel
- toolPanelUp: JPanel
- selected: JButton
- toolPanelDown: JPanel
- leftTool1: JToolBar
- img_button1: JButton[]
- leftTool2: JToolBar
- img_button2: JButton[]
- MyPanel()
- circuit_check():boolean
- change_selected(int):void
- actionPerformed(ActionEvent):void
- mouseMoved(MouseEvent):void
- mouseDragged(MouseEvent):void

<<Java Class>>
WorkPanel
(default package)

- WorkPanel()
- mouseMoved(MouseEvent):void
- mouseDragged(MouseEvent):void
- mouseClicked(MouseEvent):void
- mouseReleased(MouseEvent):void
- mouseEntered(MouseEvent):void
- mouseExited(MouseEvent):void
- mousePressed(MouseEvent):void
- paint(Graphics):void
- draw_latch(Graphics2D,int,int,int,double):void
- draw_2input_mux(Graphics2D,int,int,int,double):void
- draw_inverter(Graphics2D,int,int,int,double):void
- draw_mux(Graphics2D,int,int,int,double):void
- draw_nmos(Graphics2D,int,int,int,double):void
- draw_cmos(Graphics2D,int,int,int,double):void
- draw_capacitor(Graphics2D,int,int,int,double):void
- draw_ground(Graphics2D,int,int,int,double):void
- draw_vdd(Graphics2D,int,int,int,double):void
- draw_input(Graphics2D,int,int,int,double,int):void
- draw_output(Graphics2D,int,int,int,double):void

<<Java Class>>
myDialog
(default package)

- length: JSpinner
- length_model: SpinnerModel
- width_model: SpinnerModel
- length_unit: JComboBox
- width: JSpinner
- width_unit: JComboBox
- capacitance: JSpinner
- capacitance_unit: JComboBox
- minVoltage: JSpinner
- maxVoltage: JSpinner
- riseTime: JSpinner
- fallTime: JSpinner
- stepSize: JSpinner
- totalTime: JSpinner
- cp: Container
- del: JButton
- ok: JButton
- rotate: JButton
- unit: String[]
- node_index: int
- myDialog(JFrame,String,int)
- get_length():String
- get_width():String
- get_capacitance():String
- actionPerformed(ActionEvent):void

<<Java Class>>
WA
(default package)

- WA()
- windowClosing(WindowEvent):void

<<Java Class>>
line
(default package)

- x1: int
- y1: int
- x2: int
- y2: int
- x: int[]
- y: int[]
- end_index: int
- del: boolean
- line(int,int,int,int)
- update2(int,int):void
- update1(int,int):void
- update(int,int):void
- update_wire_mat(int):void
- update_mat(int):void
- del():void

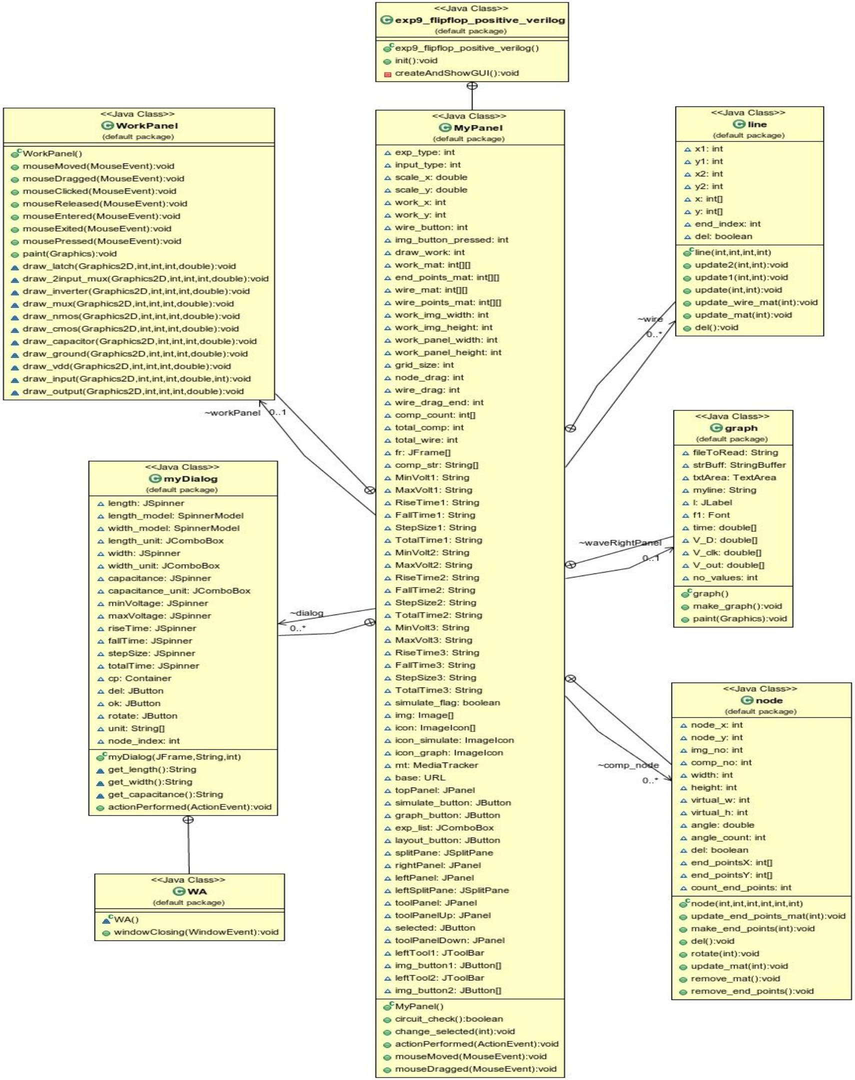
<<Java Class>>
node
(default package)

- node_x: int
- node_y: int
- img_no: int
- comp_no: int
- width: int
- height: int
- virtual_w: int
- virtual_h: int
- angle: double
- angle_count: int
- del: boolean
- end_pointsX: int[]
- end_pointsY: int[]
- count_end_points: int
- node(int,int,int,int,int,int)
- update_end_points_mat(int):void
- make_end_points(int):void
- del():void
- rotate(int):void
- update_mat(int):void
- remove_mat():void
- remove_end_points():void

VLSI Lab : D ng Verilog

Class Diagrams

Positive Edge Trig



<<Java Class>>
exp9_flipflop_positive_verilog
(default package)
exp9_flipflop_positive_verilog()
init():void
createAndShowGUI():void

<<Java Class>>
WorkPanel
(default package)
WorkPanel()
mouseMoved(MouseEvent):void
mouseDragged(MouseEvent):void
mouseClicked(MouseEvent):void
mouseReleased(MouseEvent):void
mouseEntered(MouseEvent):void
mouseExited(MouseEvent):void
mousePressed(MouseEvent):void
paint(Graphics):void
draw_latch(Graphics2D,int,int,int,double):void
draw_2input_mux(Graphics2D,int,int,int,double):void
draw_inverter(Graphics2D,int,int,int,double):void
draw_mux(Graphics2D,int,int,int,double):void
draw_nmos(Graphics2D,int,int,int,double):void
draw_cmos(Graphics2D,int,int,int,double):void
draw_capacitor(Graphics2D,int,int,int,double):void
draw_ground(Graphics2D,int,int,int,double):void
draw_vdd(Graphics2D,int,int,int,double):void
draw_input(Graphics2D,int,int,int,double,int):void
draw_output(Graphics2D,int,int,int,double):void

<<Java Class>>
MyPanel
(default package)
exp_type: int
input_type: int
scale_x: double
scale_y: double
work_x: int
work_y: int
wire_button: int
img_button_pressed: int
draw_work: int
work_mat: int[][]
end_points_mat: int[][]
wire_mat: int[][]
wire_points_mat: int[][]
work_img_width: int
work_img_height: int
work_panel_width: int
work_panel_height: int
grid_size: int
node_drag: int
wire_drag: int
wire_drag_end: int
comp_count: int[]
total_comp: int
total_wire: int
fr: JFrame[]
comp_str: String[]
MinVolt1: String
MaxVolt1: String
RiseTime1: String
FallTime1: String
StepSize1: String
TotalTime1: String
MinVolt2: String
MaxVolt2: String
RiseTime2: String
FallTime2: String
StepSize2: String
TotalTime2: String
MinVolt3: String
MaxVolt3: String
RiseTime3: String
FallTime3: String
StepSize3: String
TotalTime3: String
simulate_flag: boolean
img: Image[]
icon: ImageIcon[]
icon_simulate: ImageIcon
icon_graph: ImageIcon
mt: MediaTracker
base: URL
topPanel: JPanel
simulate_button: JButton
graph_button: JButton
exp_list: JComboBox
layout_button: JButton
splitPane: JSplitPane
rightPanel: JPanel
leftPanel: JPanel
leftSplitPane: JSplitPane
toolPanel: JPanel
toolPanelUp: JPanel
selected: JButton
toolPanelDown: JPanel
leftTool1: JToolBar
img_button1: JButton[]
leftTool2: JToolBar
img_button2: JButton[]
MyPanel()
circuit_check():boolean
change_selected(int):void
actionPerformed(ActionEvent):void
mouseMoved(MouseEvent):void
mouseDragged(MouseEvent):void

<<Java Class>>
line
(default package)
x1: int
y1: int
x2: int
y2: int
x: int[]
y: int[]
end_index: int
del: boolean
line(int,int,int,int)
update2(int,int):void
update1(int,int):void
update(int,int):void
update_wire_mat(int):void
update_mat(int):void
del():void

<<Java Class>>
graph
(default package)
fileToRead: String
strBuff: StringBuffer
txtArea: TextArea
myline: String
l: JLabel
f1: Font
time: double[]
V_D: double[]
V_clk: double[]
V_out: double[]
no_values: int
graph()
make_graph():void
paint(Graphics):void

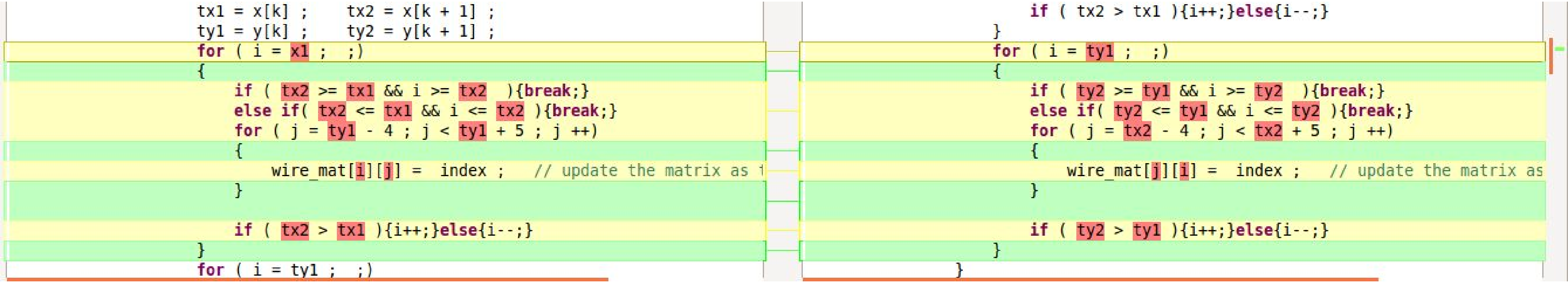
<<Java Class>>
myDialog
(default package)
length: JSpinner
length_model: SpinnerModel
width_model: SpinnerModel
length_unit: JComboBox
width: JSpinner
width_unit: JComboBox
capacitance: JSpinner
capacitance_unit: JComboBox
minVoltage: JSpinner
maxVoltage: JSpinner
riseTime: JSpinner
fallTime: JSpinner
stepSize: JSpinner
totalTime: JSpinner
cp: Container
del: JButton
ok: JButton
rotate: JButton
unit: String[]
node_index: int
myDialog(JFrame,String,int)
get_length():String
get_width():String
get_capacitance():String
actionPerformed(ActionEvent):void

<<Java Class>>
WA
(default package)
WA()
windowClosing(WindowEvent):void

<<Java Class>>
node
(default package)
node_x: int
node_y: int
img_no: int
comp_no: int
width: int
height: int
virtual_w: int
virtual_h: int
angle: double
angle_count: int
del: boolean
end_pointsX: int[]
end_pointsY: int[]
count_end_points: int
node(int,int,int,int,int,int)
update_end_points_mat(int):void
make_end_points(int):void
del():void
rotate(int):void
update_mat(int):void
remove_mat():void
remove_end_points():void

VLSI Lab : Code Smell Identified

- Long Method - Unnecessarily large methods with multiple functionalities which could have been broken down into multiple methods.
- Duplicated code - A lot of similar code performing similar functionalities should have been bundled up into a method.

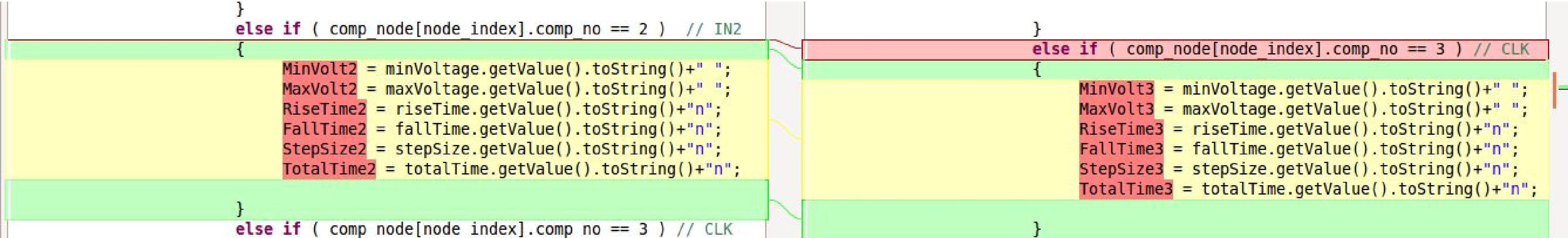


```
        tx1 = x[k] ;    tx2 = x[k + 1] ;
        ty1 = y[k] ;    ty2 = y[k + 1] ;
        for ( i = x1 ;  ;)
        {
            if ( tx2 >= tx1 && i >= tx2  ){break;}
            else if( tx2 <= tx1 && i <= tx2 ){break;}
            for ( j = ty1 - 4 ; j < ty1 + 5 ; j ++)
            {
                wire_mat[i][j] =  index ;   // update the matrix as t
            }

            if ( tx2 > tx1 ){i++;}else{i--;}
        }
        for ( i = ty1 ;  ;)
```

```
            if ( tx2 > tx1 ){i++;}else{i--;}
        }
        for ( i = ty1 ;  ;)
        {
            if ( ty2 >= ty1 && i >= ty2  ){break;}
            else if( ty2 <= ty1 && i <= ty2 ){break;}
            for ( j = tx2 - 4 ; j < tx2 + 5 ; j ++)
            {
                wire_mat[j][i] =  index ;   // update the matrix as
            }

            if ( ty2 > ty1 ){i++;}else{i--;}
        }
    }
```



```
            }
            else if ( comp_node[node_index].comp_no == 2 )  // IN2
            {
                MinVolt2 = minVoltage.getValue().toString()+" ";
                MaxVolt2 = maxVoltage.getValue().toString()+" ";
                RiseTime2 = riseTime.getValue().toString()+"n";
                FallTime2 = fallTime.getValue().toString()+"n";
                StepSize2 = stepSize.getValue().toString()+"n";
                TotalTime2 = totalTime.getValue().toString()+"n";

            }
            else if ( comp_node[node_index].comp_no == 3 ) // CLK
```

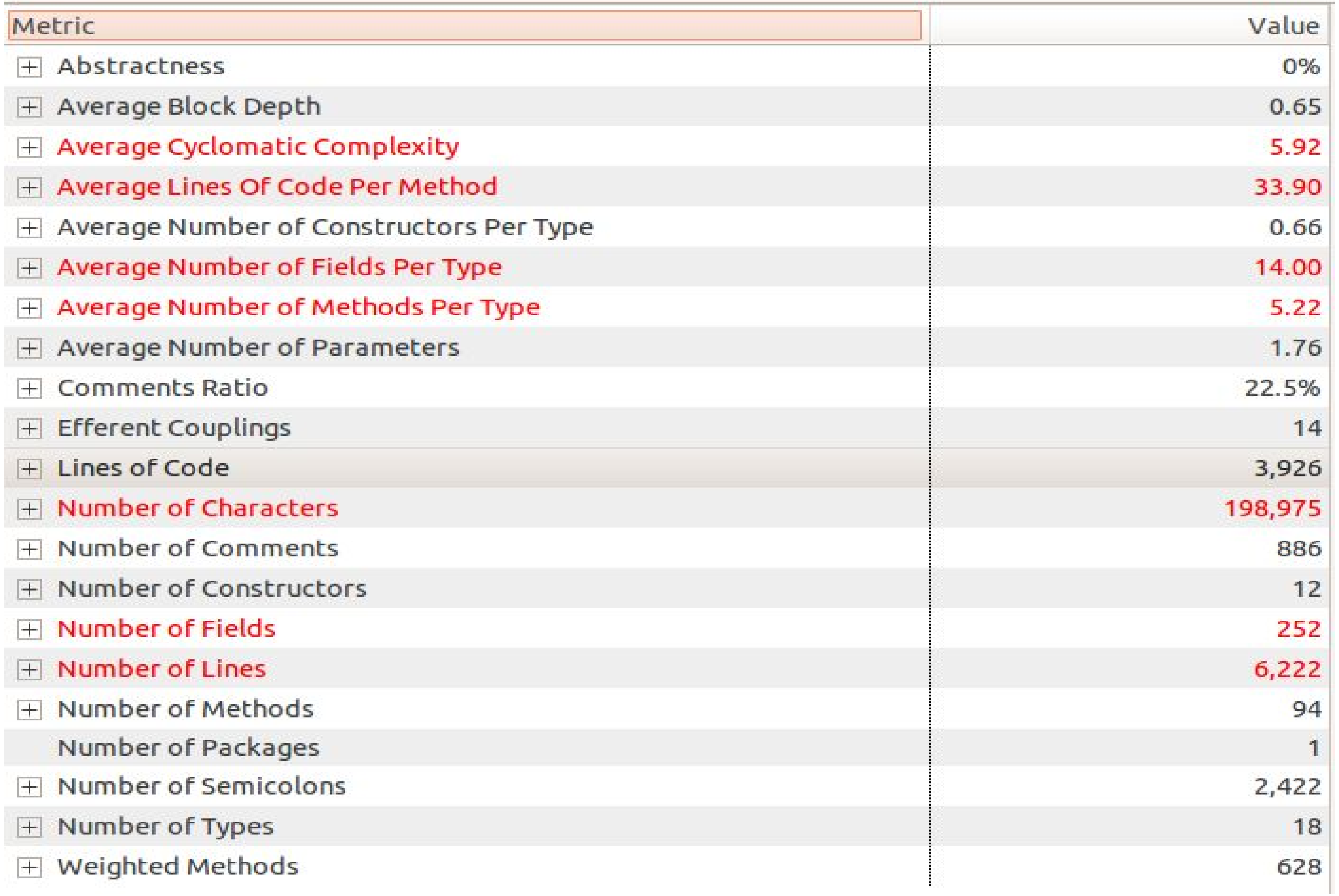
```
            }
            else if ( comp_node[node_index].comp_no == 3 ) // CLK
            {
                MinVolt3 = minVoltage.getValue().toString()+" ";
                MaxVolt3 = maxVoltage.getValue().toString()+" ";
                RiseTime3 = riseTime.getValue().toString()+"n";
                FallTime3 = fallTime.getValue().toString()+"n";
                StepSize3 = stepSize.getValue().toString()+"n";
                TotalTime3 = totalTime.getValue().toString()+"n";

            }
```

VLSI Lab : Code Smell Identified

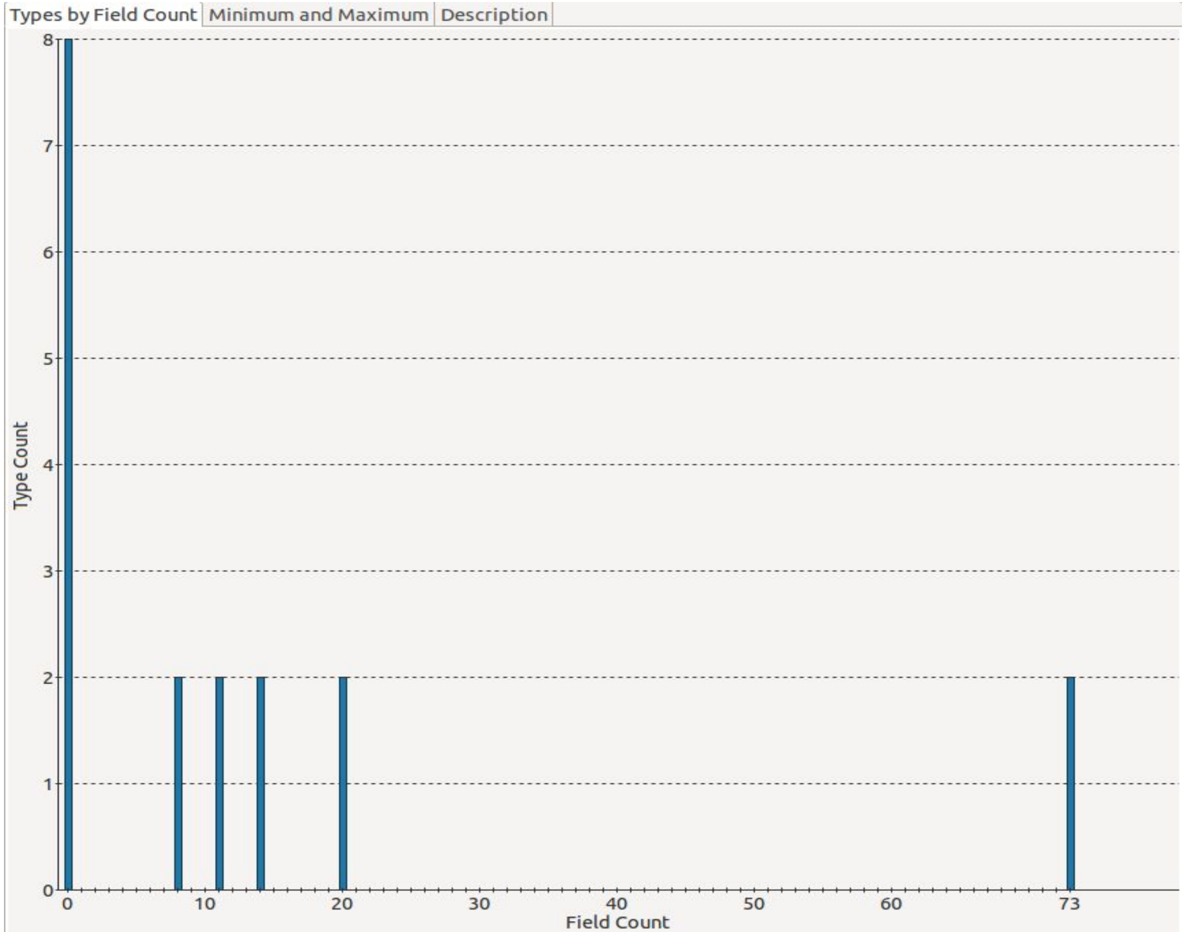
- Uncommunicative Name - Some of the method names were uncommunicative.
- Dead Code - A lot of commented out code doing nothing.
  - 8 unused methods.
  - ~250-300 commented LOC
- Unnecessary import declarations - There should not be imports for types or packages that are not referenced.
  - At Least 11 imports are not being used.
- Log Exceptions - Exceptions that are caught should be logged.
  - 10 exceptions without any logs.
- Equality test with Boolean literals - Boolean literals should never be used in equality tests.
- Empty catch clauses - Catch clauses should not be empty.
  - 6 empty catch clauses doing nothing on catching the exceptions.

VLSI Lab : Code Metrics

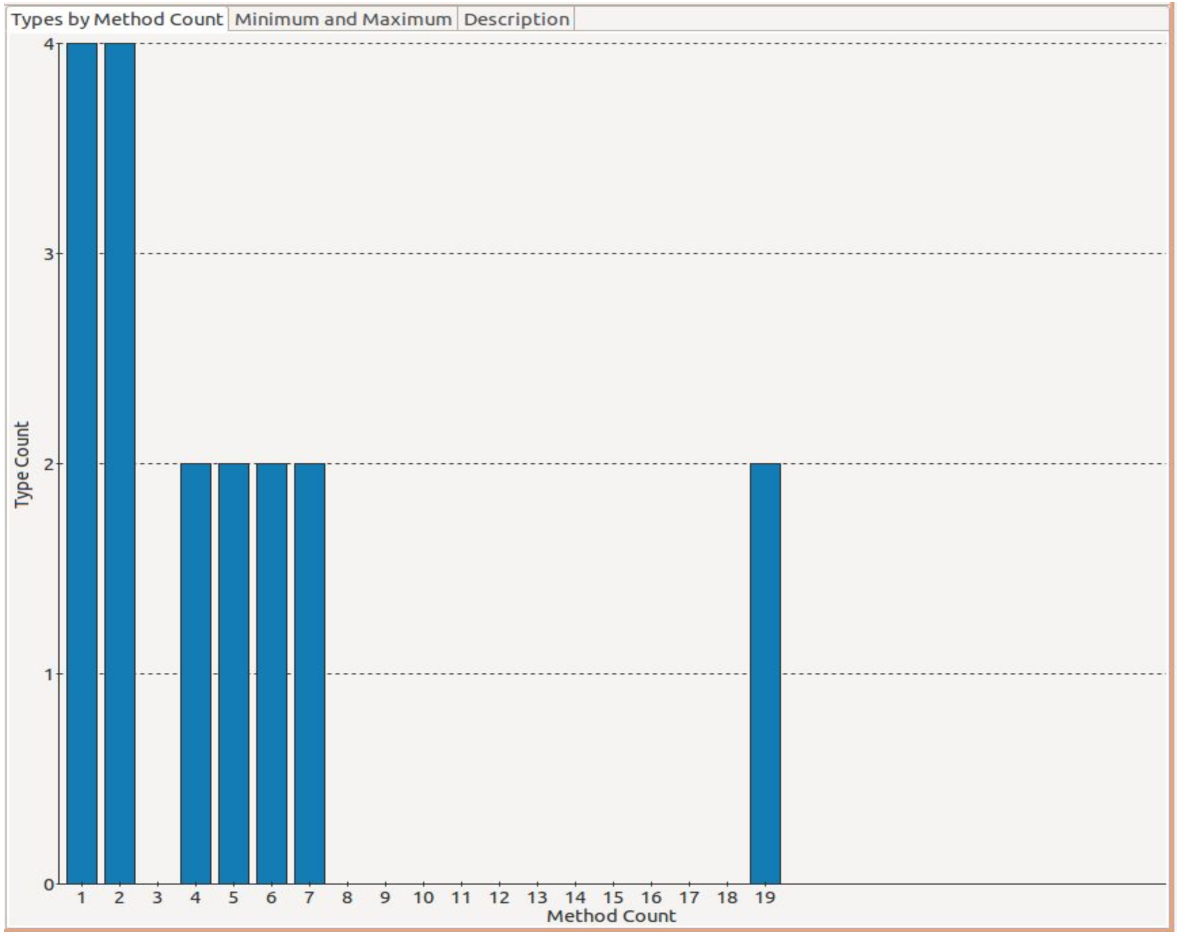


Metric	Value
Abstractness	0%
Average Block Depth	0.65
Average Cyclomatic Complexity	5.92
Average Lines Of Code Per Method	33.90
Average Number of Constructors Per Type	0.66
Average Number of Fields Per Type	14.00
Average Number of Methods Per Type	5.22
Average Number of Parameters	1.76
Comments Ratio	22.5%
Efferent Couplings	14
Lines of Code	3,926
Number of Characters	198,975
Number of Comments	886
Number of Constructors	12
Number of Fields	252
Number of Lines	6,222
Number of Methods	94
Number of Packages	1
Number of Semicolons	2,422
Number of Types	18
Weighted Methods	628

VLSI Lab : Code Metrics



VLSI Lab : Code Metrics



VLSI Lab : JavaScript Code Metrics

- Submitted as a report via Moodle.