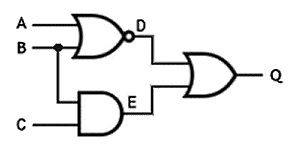
OOD2

DIGITAL CIRCUIT PROJECT



Created by:

Ali Hamzehei

Govinda Poudel

Juarni Sutedjo

Contents

[Introduction 3](#_Toc358192375)

[Class Diagram 4](#_Toc358192376)

[Description of Class Diagram and the Method 5](#_Toc358192377)

[class Circuit 5](#_Toc358192378)

[class Connection 8](#_Toc358192379)

[abstract class Gate 9](#_Toc358192380)

[class Source: Gate 10](#_Toc358192381)

[class Not: Gate 11](#_Toc358192382)

[class OR: Gate 12](#_Toc358192383)

[class And: Gate 13](#_Toc358192384)

[class Sink: Gate 14](#_Toc358192385)

[Sequence Diagram 16](#_Toc358192386)

[Open Circuit 16](#_Toc358192387)

[Add Gate 17](#_Toc358192388)

[Remove Gate 18](#_Toc358192389)

[Change Value 19](#_Toc358192390)

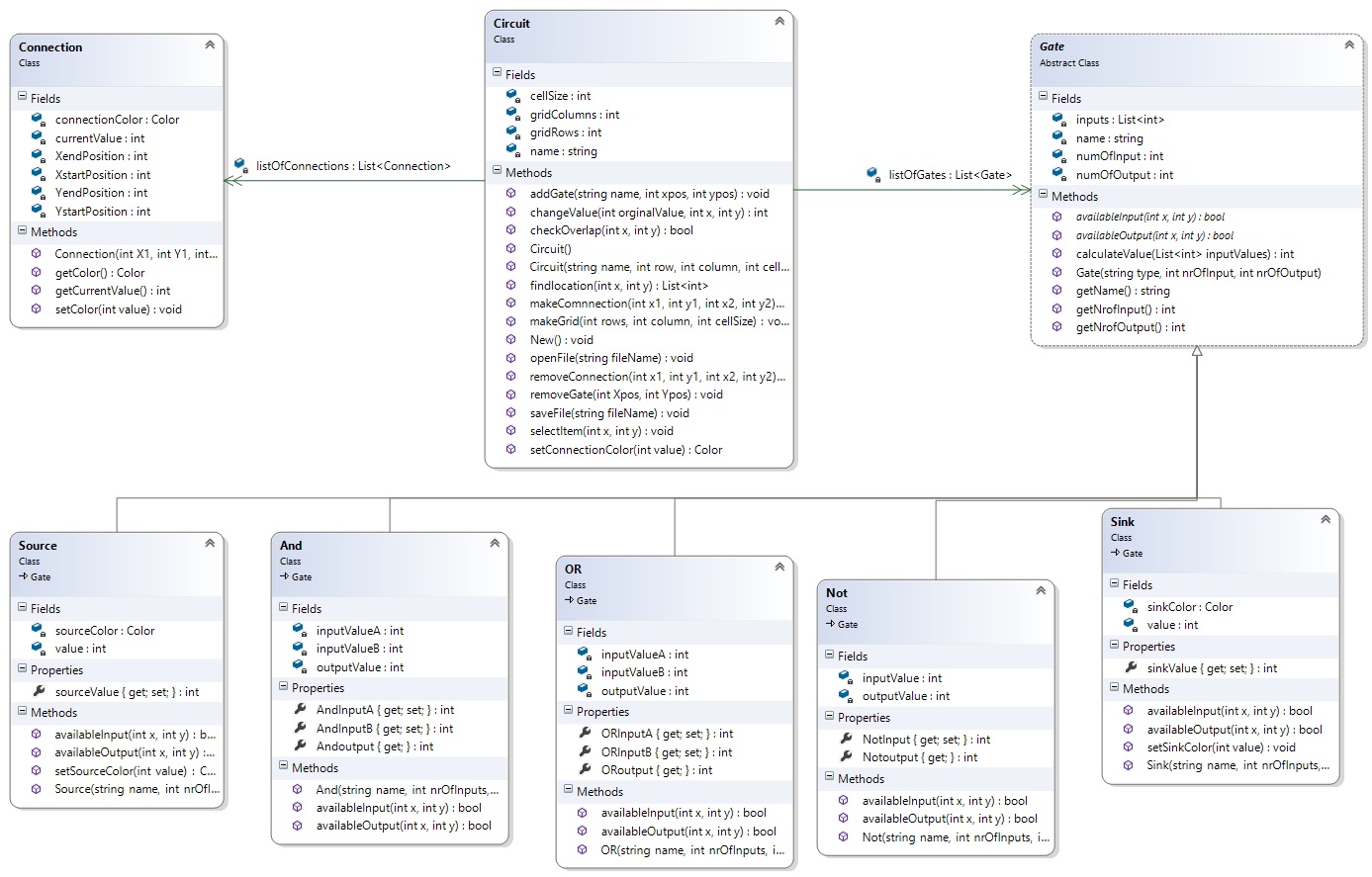
[Clear All 20](#_Toc358192391)

# Introduction

Digital Circuit Project is a small part of a big project which deals with the creating and testing of the simple electrical circuits. The electrical circuit will try to make connection in source, AND gates, OR gates, NOT gates and sink. This Design document is the part of this project.

This Design document is focused on how software looks like and how it works. This document is all about Class Diagram, description about the method and some Sequence Diagrams.

# Class Diagram



# Description of Class Diagram and the Method

namespace digitalCircuit

{

## class Circuit

{

private string name;

private List<Gate> listOfGates= new List<Gate>();

private List<Connection> listOfConnections =new List<Connection>();

private int gridColumns;

private int gridRows;

private int cellSize;

/// <summary>

/// empty constructor for circuit

/// </summary>

public Circuit()

{

}

/// <summary>

/// the constructor is set to set the properties of circuit

/// </summary>

/// <param name="name">just given name can be any string</param>

/// <param name="row">number of row required for grid</param>

/// <param name="column">number of column required for grid</param>

/// <param name="cellSize">length of individual square cell</param>

public Circuit(string name, int row, int column, int cellSize)

{

this.name = name;

this.gridColumns = column;

this.gridRows = row;

this.cellSize = cellSize;

}

/// <summary>

/// find the locationof cell , to put gate or connection

/// </summary>

/// <param name="x">x-position of mouseclick on grid </param>

/// <param name="y">y-position of mouseclick on grid</param>

/// <returns>the x,y as a list of coordinate</returns>

<int> coordinate = new List<int>();// we using this for point coordinate

return coordinate;

}

public // point List<int> findlocation(int x, int y)

{

List

/// <summary>

/// paints gates in the grid

/// </summary>

/// <param name="name">type of gate i.e. sorce, sink, and , not , or</param>

/// <param name="xpos">x-position of mouse click</param>

/// <param name="ypos">y-position of mouseclick</param>

public void addGate(string name, int xpos, int ypos)

{

List<int> position = new List<int>();

position = findlocation(xpos, ypos);

}//

How do we paint

/// <summary>

/// removes the gate from the cell at given position

/// </summary>

/// <param name="xpos">x-position of mouse click</param>

/// <param name="ypos">y-position of mouseclick</param>

public void removeGate(int Xpos, int Ypos)

{

List<int> position = new List<int>();

position =findlocation(Xpos, Ypos);

}

/// <summary>

/// connect between two points

/// </summary>

/// <param name="x1">it is the x-position of mouse first click</param>

/// <param name="y1">it is the y-position of mouse first click</param>

/// <param name="x2">it is the x-position of mouse second click</param>

/// <param name="y2">it is the y-position of mouse second click</param>

public void makeComnnection(int x1, int y1, int x2, int y2)

{

}

/// <summary>

/// remove connection between two points

/// </summary>

/// <param name="x1">it is the x-position of mouse first click</param>

/// <param name="y1">it is the y-position of mouse first click</param>

/// <param name="x2">it is the x-position of mouse second click</param>

/// <param name="y2">it is the y-position of mouse second click</param>

public void removeConnection(int x1, int y1, int x2, int y2)

{

List<int> position1,position2 = new List<int>();

position1 =findlocation(x1, y1); // finds the start position

position2 = findlocation(x2,y2); // find end position

}

/// <summary>

/// this function is called when we do the calcultion of circuit

/// </summary>

/// <param name="value">value currently in the connection</param>

/// <returns>the color according to the value in connection</returns>

public Color setConnectionColor(int value)

{

Color c = ColorTranslator.FromHtml("Green"); //

return c ;

}

// set connection

/// <summary>

/// this function is used to select items on the circuit

/// </summary>

/// <param name="x">x-coordinate from mouse click</param>

/// <param name="y">y-coordinate from mouse click</param>

public void selectItem(int x, int y)

{

}

/// <summary>

/// it creats the grid when application is loaded

/// </summary>

/// <param name="rows">numer of rows of cell in circuit</param>

/// <param name="column">number of column of cell in circuit</param>

/// <param name="cellSize">size of individual square cell</param>

public void makeGrid(int rows, int column, int cellSize)

{

}

/// <summary>

/// checks weather provoded coordinates are already taken or not

/// </summary>

/// <param name="x">x-position from mouse click</param>

/// <param name="y">y-position from mouse click</param>

/// <returns>true if provided points are empty, returns false if points are already taken</returns>

public bool checkOverlap(int x, int y)

{

return false;

}

/// <summary>

/// to change the value of source from 1 to 0 and vice-verse

/// </summary>

/// <param name="orginalValue">current value on source</param>

/// <param name="x">x-postion of source </param>

/// <param name="y">y-position 0f source</param>

/// <returns> changed value</returns>

public int changeValue(int orginalValue, int x, int y)

{ // just need to call and no parameter

return 343;

}

/// <summary>

/// storing of circuit file

/// </summary>

/// <param name="fileName">any string value as name of file</param>

public void saveFile(string fileName)

{

}

/// <summary>

/// to open saved circuit

/// </summary>

/// <param name="fileName">name of file to open</param>

public void openFile(string fileName)

{

}

public void New()

{

}

## class Connection

{ // one color for static, if we want to paint it all then when 0 is should be green, or etc

May be btter static color array and one static and do it in the beginning

New clor = color.blue

private static Color connectionColor;

private int XstartPosition;

private int YstartPosition;

private int XendPosition;

private int YendPosition;

private int currentValue=0; // default value in connection is 0

/// <summary>

/// connection is done between two gates always output with input,output output or input input cannot be connected

/// </summary>

/// <param name="X1">x-position from the first mouseclick</param>// not mouse click but “ end point of the line”

/// <param name="Y1">y-position from the first mouseclick</param>

/// <param name="X2">x-position from the second mouseclick</param>

/// <param name="Y2">y-position from the second mouseclick</param>

public Connection(int X1, int Y1, int X2, int Y2)

{

this.XstartPosition = X1;

this.YstartPosition = Y1;

this.XendPosition = X2;

this.YendPosition = Y2;

}

public int getCurrentValue() // gives the current value in individual connection

{

return currentValue;

}

public Color getColor() // gives the color of connection

{

return connectionColor;

}

public void setColor(int value) // sets the color of connection based on its value

{ // only three type of color is used in circuit

if (value == 1) { connectionColor = ColorTranslator.FromHtml("Green"); }

if (value == 0) { connectionColor = ColorTranslator.FromHtml("Yellow"); }

else connectionColor = ColorTranslator.FromHtml("Red");

}

.// we can do it set color in class connection. If u want to draw something and we have to depending on the color.

## abstract class Gate

{

private string name;

private int numOfInput;

private int numOfOutput;

private List<int> inputs = new List<int>();

/// <summary>

/// a gate is constructed

/// </summary>

/// <param name="type">gate type i.e source , or , and , not or sink</param>

/// <param name="nrOfInput">number of input for the gate</param>

/// <param name="nrOfOutput">number of output of the gate</param>

public Gate(string type, int nrOfInput, int nrOfOutput)

{

this.name = type;

this.numOfInput = nrOfInput;

this.numOfOutput = nrOfOutput;

}

/// <summary>

/// get the name of circuit

/// </summary>

/// <returns></returns>

public string getName()

{

return this.name;

}

/// <summary>

/// get number of input point

/// </summary>

/// <returns>number of input point in a gate</returns>

public int getNrofInput()

{

return numOfInput;

}

/// <summary>

/// to calcualte how many output socket does a gate has

/// </summary>

/// <returns>number of output point</returns>

public int getNrofOutput()

{

return numOfOutput;

}

public int calculateValue(List<int> inputValues)

{

int value= 0;

return value;

}

/// <summary>

/// checks weather the gate has available input for connection or not

/// </summary>

/// <param name="x">x-position of the gate in circuit </param>

/// <param name="y">y-position of the gate in circuit</param>

/// <returns>true if there is available input</returns>

public abstract bool availableInput(int x, int y);

/// <summary>

/// checks weather the gate has available output for connection or not

/// </summary>

/// <param name="x">x-position of the gate in circuit </param>

/// <param name="y">y-position of the gate in circuit</param>

/// <returns>true if there is available output</returns>

public abstract bool availableOutput(int x , int y); // this function is overriden in its inheritance class

## class Source: Gate

{ // source has only 1 and output is 0. So don’t have to call it.

Don’t forget there is just only 0 and 1.

private Color sourceColor;

int value;

// name?????

public Source(string name, int nrOfInputs, int nrOfOutputs)

: base(name,nrOfInputs,nrOfOutputs)

{

// default color of source is green i.e it will have value 1 by default

this.sourceColor = ColorTranslator.FromHtml("Green");

this.value = 1; // default value of source is 1

}

public int sourceValue

{

get {return value ;}

set { }

}

public Color setSourceColor(int value)

{

return ColorTranslator.FromHtml("Green");

}

public override bool availableInput(int x, int y)

{

return false; //always false for sink

}

public override bool availableOutput(int x, int y)

{

return true;

}

}

## class Not: Gate

{ /\* NPUT OUTPUT

A B

0 1

1 0 \*/

private int inputValue ;

private int outputValue;

public Not(string name, int nrOfInputs, int nrOfOutputs)

: base(name, nrOfInputs, nrOfOutputs)

{

this.inputValue = 0; // initially Not gate has 0 as input

this.outputValue = 1; //initially Not gate has 1 as output

}

/// <summary>

/// property of input value , to get and set the value of gate at that socket

/// </summary>

public int NotInput

{

get { return inputValue; }

set { }

}

/// <summary>

/// property of output value , to get and set the value of gate at that socket

/// </summary>

public int Notoutput

{

get { return outputValue; }

}

public override bool availableInput(int x, int y)

{

return false;

}

public override bool availableOutput(int x, int y)

{

return true;

}

## class OR: Gate

{ /\* NPUT OUTPUT

A + B

0 0 0

0 1 1

1 0 1

1 1 1 \*/

private int inputValueA ;

private int inputValueB ;

private int outputValue;

public OR(string name, int nrOfInputs, int nrOfOutputs)

: base(name, nrOfInputs, nrOfOutputs)

{

this.inputValueA = 0;

this.inputValueB = 0;

this.outputValue = 0; //initially and gate has 0 value as inputs and output

}

/// <summary>

/// property of input value A , to get and set the value of gate at that socket

/// </summary>

public int ORInputA

{

get { return inputValueA; }

set { }

}

/// <summary>

/// property of input value B , to get and set the value of gate at that socket

/// </summary>

public int ORInputB

{

get { return inputValueB; }

set { }

}

/// <summary>

/// property of output value , to get and set the value of gate at that socket

/// </summary>

public int ORoutput

{

get { return outputValue; }

}

public override bool availableInput(int x, int y)

{

return false; //checks both input point

}

public override bool availableOutput(int x, int y)

{

return true;

}

## class And: Gate

## 

{ /\* NPUT OUTPUT

A + B

0 0 0

0 1 0

1 0 0

1 1 1 \*/

private int inputValueA ;

private int inputValueB ;

private int outputValue;

public And(string name, int nrOfInputs, int nrOfOutputs)

: base(name, nrOfInputs, nrOfOutputs)

{

this.inputValueA = 0;

this.inputValueB = 0;

this.outputValue = 0; //initially and gate has 0 value as inputs and output

}

/// <summary>

/// property of input value A , to get and set the value of gate at that socket

/// </summary>

public int AndInputA

{

get { return inputValueA; }

set { }

}

/// <summary>

/// property of input value B , to get and set the value of gate at that socket

/// </summary>

public int AndInputB

{

get { return inputValueB; }

set { }

}

/// <summary>

/// property of output value , to get and set the value of gate at that socket

/// </summary>

public int Andoutput

{

get { return outputValue; }

}

public override bool availableInput(int x, int y)

{

return false; //always false for sink

}

public override bool availableOutput(int x, int y)

{

return true;

}

## class Sink: Gate

{

private Color sinkColor;

private int value;

public Sink(string name, int nrOfInputs, int nrOfOutputs)

: base(name,nrOfInputs,nrOfOutputs)

{

// default color of sink is white i.e it will have value 0 by default

this.sinkColor = ColorTranslator.FromHtml("white");

this.value = 0; // default value of source is 0

}

/// <summary>

/// this property Sink value is used to get and set the value on sink

/// </summary>

public int sinkValue

{

get {return value ;}

set { }

}

/// <summary>

/// this set function is used to set the color of sink according to value eithr 0 (white) or 1 ( blue)

/// </summary>

/// <param name="value">it is the final value on sink after calculating all circuit </param>

public void setSinkColor(int value)

{

if (value == 1 ) {this.sinkColor = ColorTranslator.FromHtml("blue");}

else ColorTranslator.FromHtml("white");

}

//this function is ovver ride here to

public override bool availableInput(int x, int y)

{

return true;

}

public override bool availableOutput(int x, int y)

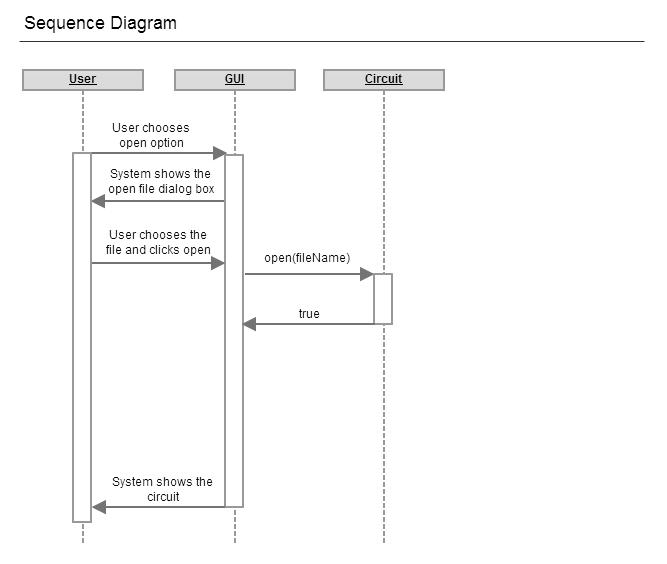
{

return false; // always false for sink

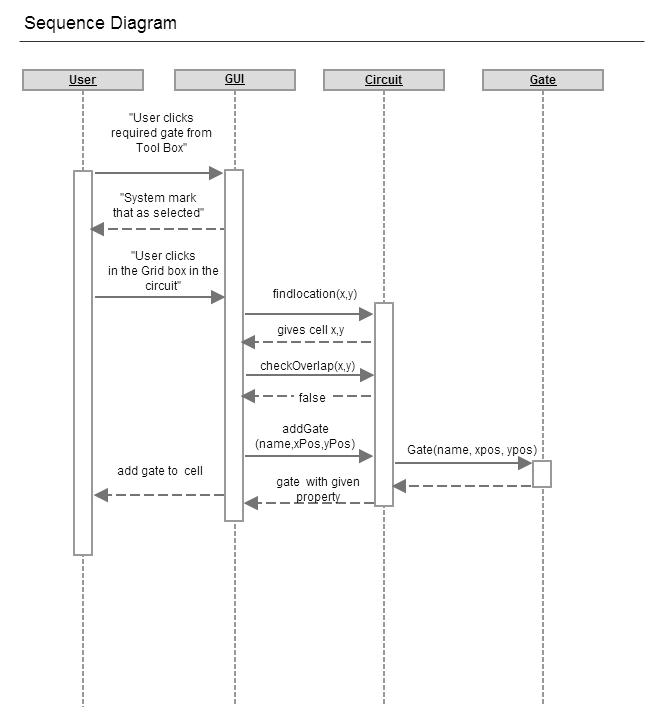
}

# Sequence Diagram

## Open Circuit

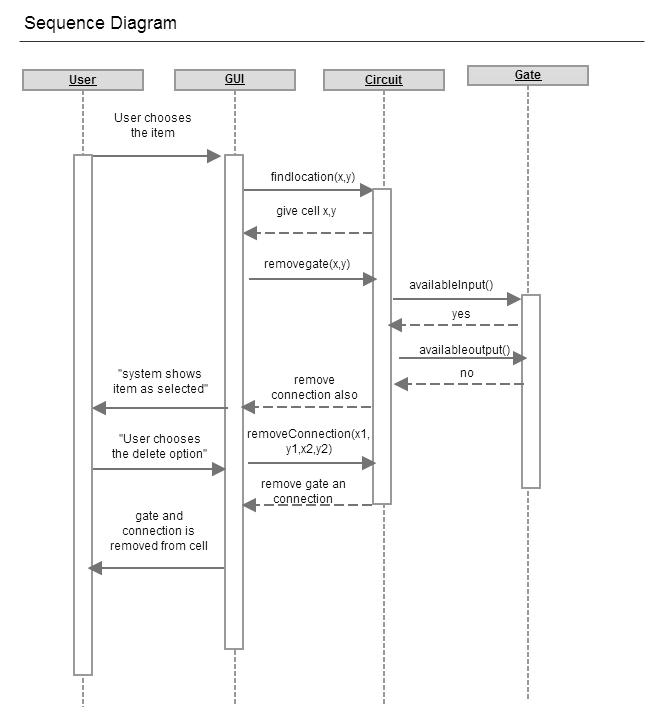


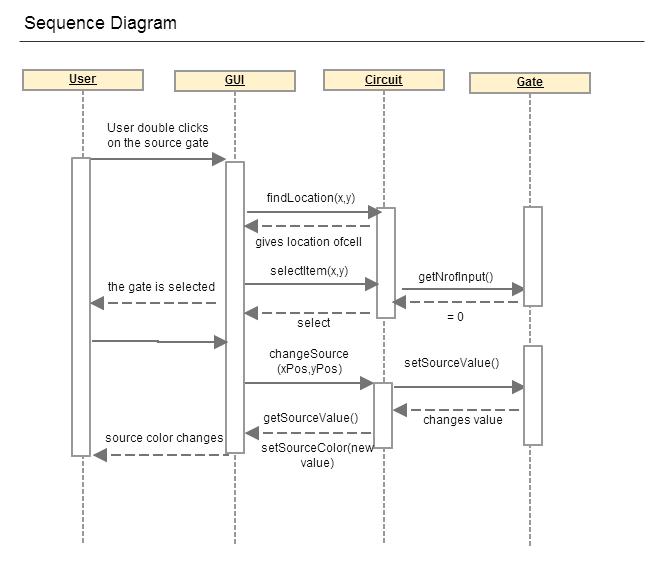
## Add Gate



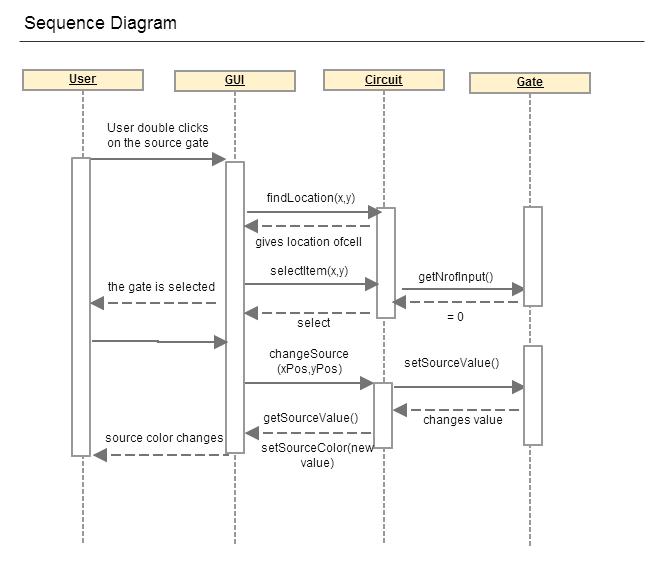
// add

## Remove Gate





## Change Value



## Clear All

