DESIGN DOCUMENT

COP290

AC Circuit Solver

Prepared by

$HimanshuKejriwal - 2016CSJ0011$

$RahulByasSherwan - 2016CSJ0028$

$Hrushikesh - 2016CSJ0030$

April 8, 2018

Contents

1 Introduction

1.1 Purpose

Simulation of electrical circuits has always been an interesting and useful topic, as we have been solving circuits right from 7th Grade.To solve a small cicuit with few commponents is easy, but it gets really difficult to solve circuits with a huge number of nodes. The key analysis in Circuit design like DC analysis, Frequency response, n-port analysis is much simpler to simulate in Circuit Solver than analyzing all these responses theoretically which is a time.

1.2 Target Audience

The document is intended for developers, testers, students and professors. Electronics/Electrical Engineering students interested in developing deeper insight into Analog Circuit Design, by applying all they learn in Analog Circuit Course in building an entire application. This course should also be helpful for teachers who wish to use this material for their courses.

1.3 Project Scope

It will be useful for every engineering students as everyone one of them comes face to face with various complex circuits which is very difficult to visualize directly. It will save the precious time for various people who remain involved in these type of complex circuit problems. Not only for visualizing but also for solving (if demanded), the complex in seconds. When the software reaches its final state(with all special graphics features, it can also be introduced into the market for global use.

1.4 References

1. MIT AC Circuit analysis notes. http://web.mit.edu/8.02t/www/802TEAL3D/visualizations/coursenotes/modules/guide12.pdf
2. https://www.youtube.com/playlist?list=PL2EPLswhM_KYKCq9eu3EVUmRDoGXUV04V
3. Solving multi-frequency circuits. http://www.allaboutcircuits.com/textbook/alternating-current/chpt-7/circuit-effects
4. https://www.swarthmore.edu/NatSci/echeeve1/Ref/mna/MNA3.html
5. Fundamentals of electrical engineering - Book by Leonard S. Bobrow ISBN 9780195111248

2 Overall Description

2.1 Functioning

The AC circuit Solver takes input in the form of a SPICE Netlist.This netlist will then be decoded for using in our program to design the corresponding AC circuit. The same netlist is first converted in a matrix form, which contains it's endpoints which is represented by "1", magnitude of Impedence of each component, values of component and Voltage and Current source with multiple specifications.

The same matrix but in modified form (sorted) is used to make SVG images.

The same matrix will also be used to solve the circuit given the values of the various circuit components and position of the components in the circuit, by using the various formulas governing the flow of the current in the circuit depending on the potential difference across various components.

Inbuit libraries for solving matrix equations is used to solve matrix.

2.2 Operating Environment

The software will work work on both windows, linux and mac. The system should also contain a browser that is able of running the svg files. One may use the most common ones like Chrome and Mozilla firefox.

3 Circuit Drawer

3.1 SVG

SVG is an XML language, similar to XHTML, which can be used to draw vector graphics, such as the ones shown to the right. It can be used to create an image either by specifying all the lines and shapes necessary, by modifying already existing raster images, or by a combination of both. The image and its components can also be transformed, composited together, or filtered to change their appearance completely. It will be used to create circuits for Part 1 of the assignment.

3.2 Parsing

In this program, parsing is done using if else and regex.
Then the data is decoded and broken into smaller elements. A (n X m) matrix is generated where

n = no of NETS detected + 1 for earthing (as "Netlist 0" and "0" are different).
m = no of components in netlist which is equal to(no of new lines detected in .cir file)

The Matrix(n X m) is zero matrix. The nodes where a component is connected will be converted to ones. So, in a column only 2 "1's" will be possible. This is used to check error.
The rows are sorted in decreasing order, i.e. the net3 will be on top then net2 then net1 . . . .
As net0 as well as 0(earthing) is present in .cir file, we are considering net0 as 0 and earthing as -1 for easy calculations.

3.3 SVG Area Co-ordinates

For SVG files, we have taken each component of length 70 unit.
The size of SVG area is decided by following

width = (no of INSTRUCTIONS * 100) + 200 (100 on each left and right ends)
height = (no of ELEMENTS * 70) + 200 (100 on each top and bottom ends)

3.4 draw.cpp

This is the main function of the draw which calls the above functions whenever required. it makes a matrix of n X m containing 0 and 1 where n denotes no. Of distinct nets and m denotes to no. Of instruction in the input file.

We are basically creating a space of (100*m+220)x(70*n+200) in the SVG file since 70 is the width of your element. And 100 because after each iteration we are moving the x coordinate by 100. The matrix described above contains 2 1s in each column since each element has two terminals and this matrix denotes its connection with different nets.

In order to remove cutting wires, we have sorted the matrix by considering the difference of its nets. However, there is still a possibility of cuttings. So, we have made a dot function above to determine the exact nets to which it is connected.

4 Circuit Solver

4.1 Data Handling

We have created two vector array, one is string type and other is float type.

In first string vector array, we have all details of one component in one column. No of rows for this vector array is 9 as Voltage/Current Source has 9 values.

In second vector array, we have converted strings into values by encoding everthing related to a component. Following things are mentioned in a column

Row 1 - (0 for Resistor, 1 for Capacitor, 2 for Inductor, 3 for Volt Source and 4 for Current Source)
Row 2 - Index of component i.e. R2 implies 2 in this location.
Row 3 - Endpoint 1
Row 4 - Endpoint 2
Row 5 - Converted values in SI Units, i.e. all mili,kilo etc are removed in this step
Row 6-10 - Rest of the values for Voltage and Current Source

4.2 solve.cpp

We have applied Modified Nodal Analysis for calculating the voltage and current across components as if we apply only Nodal Analysis (KCL), we can't equate if we have current source in the circuit.

A matrix equation is made to find the Voltage at nets.

The equation used is [A][X] = [S]

$$A = \begin{bmatrix} G & B \\ C & D \end{bmatrix}$$

A is a square matrix of size (m+n)

where m is no of nets excluding earthing,
where n is no of voltage sources in circuit

where G matrix is calculated by applying KCL size on circuit. It is of size m X m

B matrix contains value 1,-1 and 0. If Voltage terminals is connected to net 3 and net 2. Then the first column will have values

$$ex = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$$

1, -1 and 0. The sixe of B matrix is of size m X n

C matrix is transpose of B

D matrix is generally 0, if there are no dependent sources. it is square matrix of size n

X matrix is (m + n) X 1 matrix. First m rows have unknown Potential values at that net. Next n rows have the unknown currents through the voltage sources.

S matrix is (m + n) X 1 matrix. First m rows contains the sum of the currents through the passive elements into the corresponding node (either zero, or the sum of independent current sources). Next n rows contains the values of the independent voltage sources.

If the source has multiple frequencies, then we will make 3-d array where each source with different frequencies are stored in 3rd dimension of array.

Used Eigen-function to solve this equation and send it back as string.

Now the normal calculations are done to find out Voltage and Current across components.