DC Circuit Simulator

**Analysis**

Juraj Pavlović

2020./2021.

Content

[Analysis 3](#_Toc67391341)

[Introduction 3](#_Toc67391342)

[The end-user 3](#_Toc67391343)

[Current system investigation 4](#_Toc67391344)

[Flowcharts for existing and proposed system 6](#_Toc67391345)

[Existing programs 7](#_Toc67391346)

[Complex principles 8](#_Toc67391347)

[Components 8](#_Toc67391348)

[Adding resistors 9](#_Toc67391349)

[Measuring voltage and current in the circuit 9](#_Toc67391350)

[Kirchhoff's current law (KCL) 9](#_Toc67391351)

[Kirchhoff’s voltage law (KVL) 10](#_Toc67391352)

[Ohm’s law 10](#_Toc67391353)

[Mesh analysis 11](#_Toc67391354)

[Nodal analysis 12](#_Toc67391355)

[Incidence matrix 14](#_Toc67391356)

[Sparse matrix 15](#_Toc67391357)

[Modified nodal analysis (MNA) 16](#_Toc67391358)

[Netlists 19](#_Toc67391359)

[List of components 19](#_Toc67391360)

[Limitations 20](#_Toc67391361)

[Objectives 20](#_Toc67391362)

[References 23](#_Toc67391363)

# Analysis

## Introduction

Learning about electric circuits for A – level students can be quite demanding and challenging especially to those who are not familiar with the concept. There are plenty of laws and formulas to be considered when solving a circuit question. I found circuits hard to understand without seeing how the formulas are intertwined and work in real life.

My aim with this coursework is to develop a program with a user-friendly interface that will serve as a teaching aid mainly for A – level students, but also engineering and physics students entering university.

The program will give the user the ability to set up a simple DC electric circuit using a couple of essential electronic components. Components will hold certain values, e.g., resistance for resistors, voltage for voltage source. However, these values are not constants. Users will be able to choose the values suitable for their needs.

## The end-user

Even though my specific end-user is an A – level physics student, this program could be used by anyone who wishes to develop an understanding of basic circuitry or further their knowledge in this very field. As everyone learns differently, I believe this tool will help a lot of students struggling to learn the subject using only a textbook and formulas, which sometimes can be quite confusing. Most of us are not used to seeing how resistors interact when put together in an electric circuit, so being able to visualise the problem can be of great help for those who prefer learning things (like me) practically.

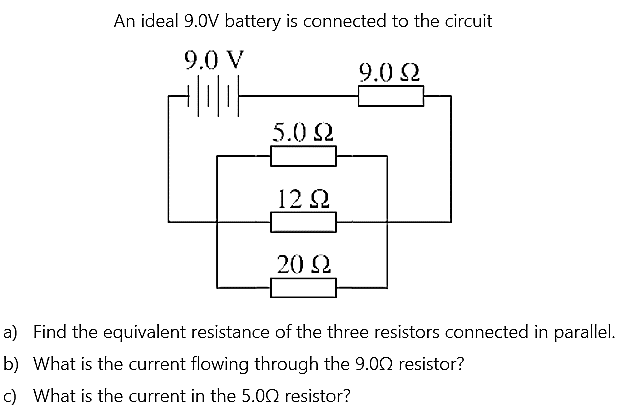
This program could also be used by teachers to give students a better explanation of the topic. When teaching, it can be very time consuming to physically set up the circuits and do the appropriate measurements. This wastes lesson time which can be spent further studying the topic. Furthermore, this program can be of great help during situations like the one in 2020. when the circumstances dictated by the pandemic forced students to study from home. One of the disadvantages of distance learning is that students cannot perform their regular physics practical work. This can have a negative effect on their overall achievement in the subject. Using this program enables students to keep up with their schoolwork even when they are not in school.

I wish users to be able to quickly learn and understand the topic of electricity, which is why the program itself will cover everything an A – level student should know.

## Current system investigation

Electricity is one of the topics in A – level physics covered in year 12. It is most commonly taught by a teacher talking through the concepts using a Power Point presentation and sections in a textbook. Sometimes the teacher would further explain the topic by demonstrating these concepts with a couple of experiments. As a revision and further practice, students are usually given some questions to be answered as their homework.

Below is an example question a student might be assigned as their homework.



To solve such a question student needs to use a couple of different principles and formulas accurately (adding resistors in parallel and in series, Kirchhoff's current law, Kirchhoff's voltage law, etc.). Below is an example of how a student might approach solving this question.

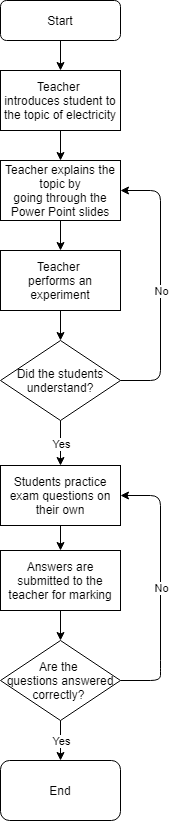
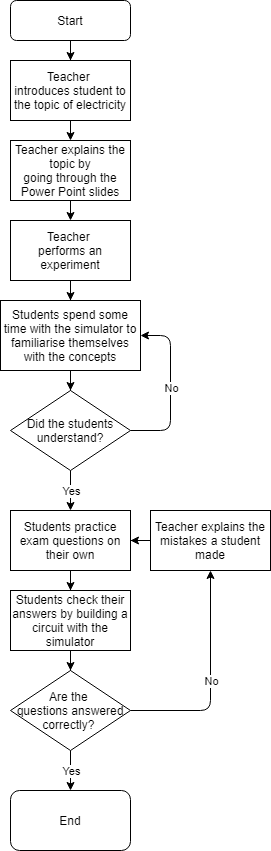
Slika na kojoj se prikazuje tekst

Opis je automatski generiran

For someone with very modest experience in solving circuit questions, this can be quite a challenge. When solving similar questions, I always wanted to be able to quickly make the circuit myself and check whether my results were correct. Unfortunately, due to the lack of electronic components and appropriate equipment for performing measurements, I was unable to do it. My main goal is to enable students to check their results using a simulator that would calculate the needed values.

### Flowcharts for existing and proposed system

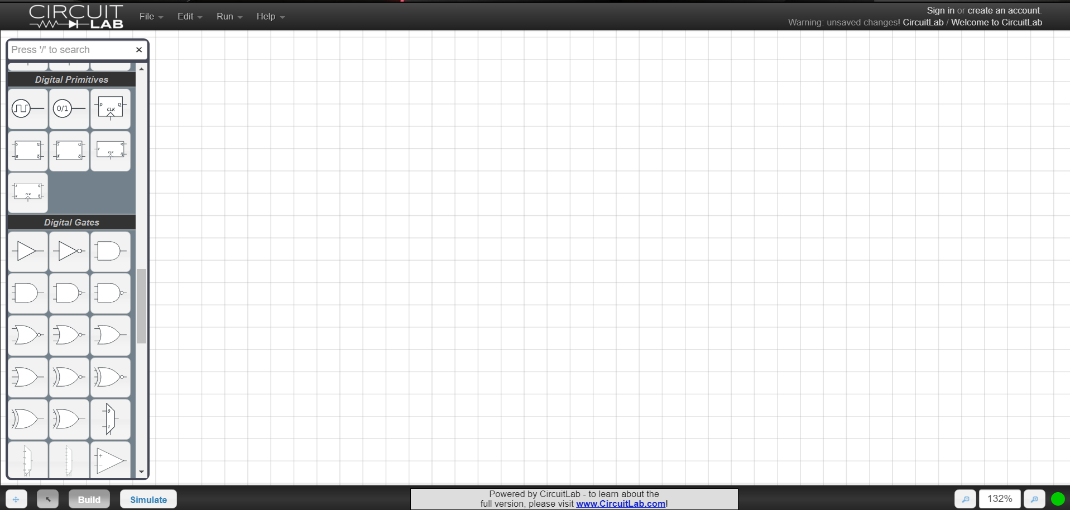
Existing system Proposed system

### Existing programs

There are already several existing programs that can be of help when solving circuit questions. One of them is “DC Circuit Builder” from the *physicsclassroom.com* webpage. With this simulator, a user can perform all the fundamental functions needed for an A – level student. A user can construct a DC circuit with resistors and an independent voltage source and then when the circuit is closed, measure a potential difference and a current flowing through any two points in the circuit. Both voltage of a voltage source and a resistance of resistors can be changed for any individual component. I will be implementing both of these options into my simulator. One component which could be added to this program is the independent current source. That way a student could control a current flowing through a specific part of the circuit.

One example of a more complex simulator is “CircuitLab”. In addition to being able to construct DC circuits, it also offers a wide range of other features. It allows users to control signals of current and voltage sources, add timers, diodes, MOSFETs, relays, and even perform Boolean algebra using digital gates. While these features might be useful for university students and engineers, such a complex program could confuse an average A – level student trying to learn basic circuitry. That is why in my program I will only focus on DC simulation. A benefit of this program over a previous one is that the user interface is much more straightforward. I will build upon this idea of a grid-based system with available components being on the left-hand side of the screen and function buttons on either the top or the bottom of the screen.



Even though all simulators might seem very different because of their user interface design, they all work using the same basic principles (for simulating DC networks at least). For a computer to solve a circuit, some way of describing the topology of the circuit is needed. A common approach would be to use netlists, which is essentially a text describing how different components are connected in a circuit. Then a circuit needs to be analysed using some kind of method. The two most common ones are mesh and nodal analysis. After that, a network of components is turned into a system of linear equations which can be solved numerically using matrixes. These principles were first used in the 1960s and the first such program open to the public was SPICE (Simulation Program with Integrated Circuit Emphasis). SPICE is an open-source analogue circuit simulator that served as a basis for other simulation programs which are today used in academia, industry and commercially. My program will be based upon one aspect of the SPICE engine which is called DC static analysis. There also exists AC analysis and transient analysis, but since these topics are not part of a A – level syllabus, I will not be implementing them.

## Complex principles

### Components

There are two types of electric components which A – level physics students should be familiar with: active and passive. Active components provide a source of energy in the circuit in the form of voltage or current, while the passive components convert or store this energy in other forms (thermal, electric or magnetic energy).

Active components:

* Voltage source
* Current source

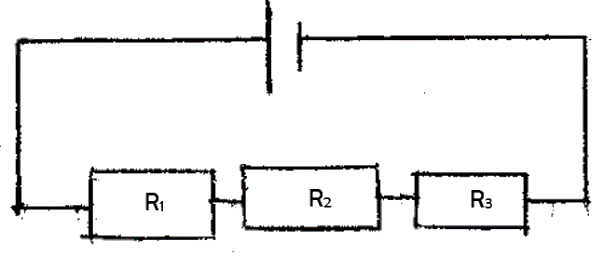
Passive components:

* Resistor
* Capacitor
* Inductor

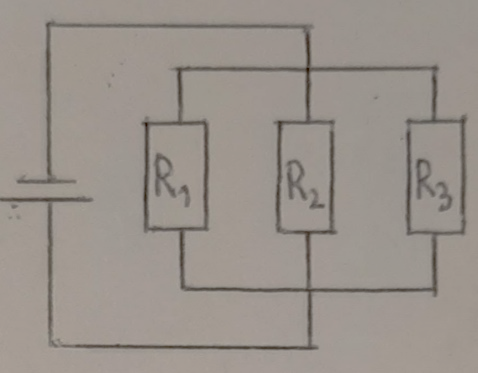
A DC circuit analysis is a static circuit analysis which means that nothing changes over time. Once a circuit is built, currents and voltages across all the components in the system will remain constant. In such a system, capacitor and inductor can be treated as an open and closed circuit, respectively. Therefore, capacitors and inductors do not need to be included in the simulator since they will not have any impact on the final voltage and current values.

### Adding resistors

#### Series

The total resistance for two or more resistors connected in series is equal to the sum of the individual resistances. The resistance of resistors R1, R2, and R3 can be expressed as:

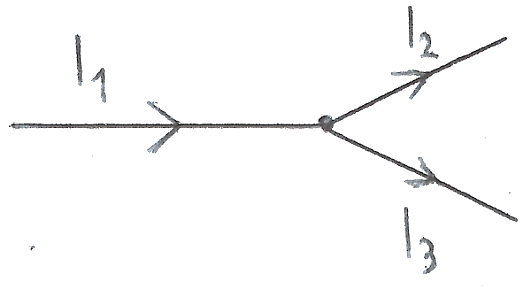
#### Parallel

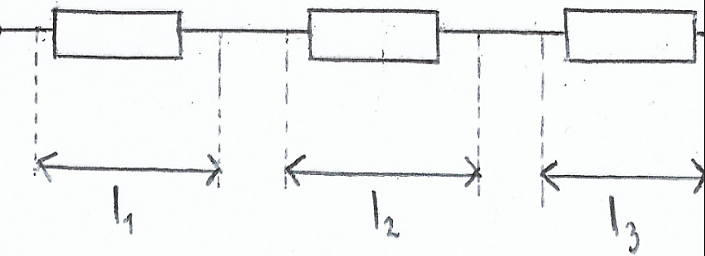
When resistors are connected in parallel different rule applies. The total resistance of resistors connected in parallel is given by:

### Measuring voltage and current in the circuit

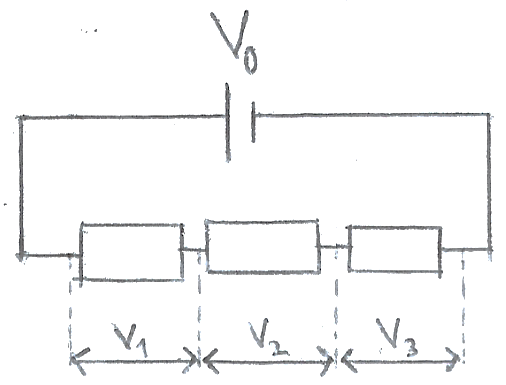
Voltage can be measured across any two points in a circuit using a voltmeter. To do the measurement correctly, the voltmeter needs to be connected in parallel. On the other hand, the current flowing through the circuit is measured by connecting the ammeter in series.

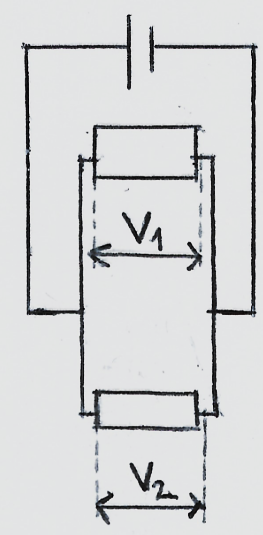
### Kirchhoff's current law (KCL)

At any node in a circuit, the total current leaving the node is equal to the total current entering the node. This is shown in the figure. Current I1 is entering the and currents I2 and I3 are exiting the node. Therefore, the following equation can be applied:

When current is entering a component, the current of the same value is leaving the same component. Also, if the current is passing through two or more components in series, each component receives the same amount of current.

### Kirchhoff’s voltage law (KVL)

The sum of all the voltages in a closed-loop circuit is equal to zero. This means that for two or more components in series, the total voltage across all the components is equal to the sum of the voltage across each component.



When two or more components are connected in parallel their voltages are the same.

### Ohm’s law

Ohm’s law describes a relationship between resistance, voltage and current. According to Ohm’s law, pd (potential difference) across a conductor is proportional to the current through it. In other words, for a component through which passes the current *I* when the pd across it is *V*, its resistance *R* is given by the equation:

Rearranging the above equation gives the following expressions for voltage and current:

Knowing two of those three quantities, a third one can be easily calculated.

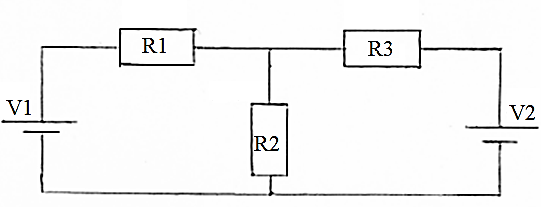
### Mesh analysis

A mesh current analysis uses simultaneous equations, KVL and Ohm’s law to calculate the unknown currents in various loops of the circuit. The steps of the mesh analysis are:

1. Identify meshes in the circuit – Mesh is a loop with no other loops inside it.
2. Assign current variables for each mesh - Inside each mesh, there is a current flowing through it. Its direction needs to be defined as either clockwise or counter clockwise.
3. Apply KVL for each mesh in the circuit – Each mesh in the circuit will have its KVL equation. By writing one KVL equation for each mesh, a system of simultaneous equations will be obtained.
4. Solve the resulting linear system for mesh currents.

An example of these steps being implemented is shown below.

Consider the following circuit:



Where:

V1 = 9 V

V2 = 5 V

R1 = 3 Ω

R2 = 2 Ω

R3 = 1 Ω

Steps:

1. The circuit has two meshes (loops) inside it. The first mesh contains a V1 voltage source, R1 and R2 resistor. The second mesh has a V2 source, R3 and R2 resistor.
2. I will assign mesh currents in a clockwise direction so that the current I1 is flowing through the first mesh and I2 through the second mesh. Since both meshes have a common R2 resistor, a current through it will not be I1 nor I2, but a difference of those two currents, I1 – I2.
3. Applying KVL results in two following simultaneous equations:
4. Plugin in the values and rearranging the equations to solve for I1 and I2 gives:

So I1 = -1 A and I2 = -7 A.

The reason both current values turned out to be negative is that their previously defined directions were wrong. I assumed both currents were flowing clockwise, but their actual direction is counter clockwise. However, the wrong assumption, in the beginning, does not affect the result as only a negative sign needs to be removed.

After applying mesh analysis and calculating the currents, other values (e.g., voltage/power across resistors) can be found using Ohm’s law.

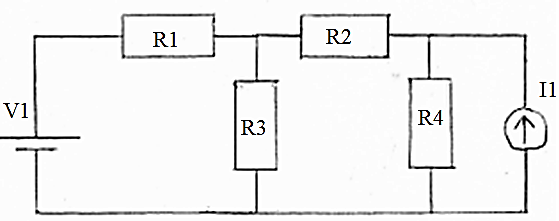
### Nodal analysis

Another basic method used for solving any electrical network is nodal analysis. While mesh analysis is mostly based on the KVL, nodal analysis is based on the KCL and is used to find different nodal voltages around a circuit. The basic principle of nodal analysis is:

1. Identify the nodes in the circuit and assign a reference node – Node is a point of connection between any two or more components. A node with the most connections is taken to be a reference node. Reference node servers as a ground node and therefore has a potential of 0 V. All the other node voltages are referenced with respect to the ground node.
2. Assign labels to all the node voltages.
3. Apply KCL for each node in the circuit, except a reference node – If there are n nodes in the circuit, there will be n – 1 equations.
4. Solve the linear system for node voltages.

An example of these steps being implemented is shown below.

Consider the following circuit:



Where:

V1 = 9 V

I1 = 2 A

R1 = 5 Ω

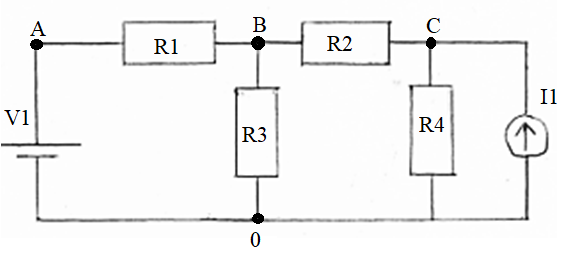
R2 = 8 Ω

R3 = 10 Ω

R4 = 10 Ω

Steps:

1. There are four nodes in the circuit, so there will be three simultaneous equations.
2. Node A connects V1 and R1, node B connects R1, R2 and R3, node C connects R2, R4 and I1, node D connects V1, R3, R4 and I1. Since node D is connected to the most components, it is taken to be a ground node and labelled 0.



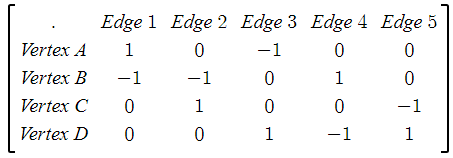
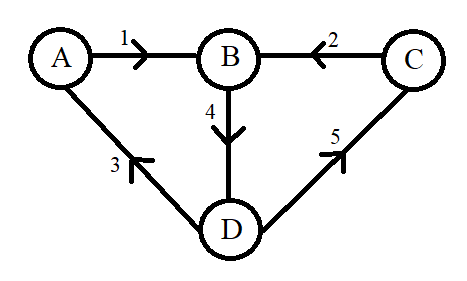
1. Applying KCL results in three following equations:
2. Plugin in the values and solving the equations for unknown nodal voltages gives VA = 9 V, VB = 5.9 V and VC = 12.2 V.

Once all the values of all nodal voltages in the circuit and the resistance of all the resistors is known, electric current across any resistors can be calculated with Ohm’s law.

Both mesh and nodal analysis are very powerful techniques for analysing a circuit. Mesh analysis is more suitable to use when there are voltage sources in the circuit, while nodal analysis is used when there are more current sources in the circuit. Since my simulator will give the user the ability to use both, I will use a combination of two techniques together with incidence and sparse matrices to form a so called “Modified Nodal Analysis”.

### Incidence matrix

“An incidence matrix is a logical matrix that shows the relationship between two classes of objects.” The first class represents vertices and the second represents branches. The matrix has one row for each vertex and one column for each branch. The entry in the matrix is 1 or -1 if a vertex and a branch are related and 0 if they are not. 1 is used when a branch current is leaving a vertex and -1 when it is entering it. Below is an example of a graph and its incidence matrix.



In my simulator, I will use incidence matrices to describe the connection of the voltage sources and so show the effect of voltage sources on the circuit. This method would be fine on its own for straightforward circuits like the ones in the previous examples. As the users will probably want to make more complex circuits, this will result in the matrix being filled with mostly zeros. To optimise it I will be using sparse matrices optimisation techniques.

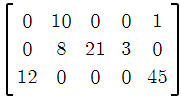
### Sparse matrix

A sparse matrix is a matrix that mostly consists of zero values. When dealing with such a matrix on a computer, it is common to optimise a computer’s processing power and memory by using specialised algorithms and data structures. Using standard matrices can be very inefficient since most of the time during a matrix operation computer would have to deal with a great number of zeros. This can be avoided with various specialised format structures which essentially ignore all the zeros and focus on the non-zero values. However, this does make accessing the individual elements more complex.

There are two types of these formats:

1. Those that use data structures to modify a sparse matrix. These are used to construct the matrices.
2. Those that use data structures which are more suitable for performing efficient operations.

Since my simulator will be using matrices to solve systems of linear equations, a format which allows arithmetic operations is required. Therefore, the second type of sparse matrices format is more suitable for my case. A CSC format is the most common for python implementation using SciPy which is why I will be using it in my program.

Compressed Sparse Row Format (CSC) uses three one dimensional arrays which contain Non\_zero\_value, Row\_index and Column\_pointer. Non\_zero\_value array is made by writing every element in the list which value is different from zero, starting from top to bottom of the first column and then moving right to the next column. Column\_pointer is an array of indices corresponding to the non-zero values. So Non\_zero\_value and Column\_pointer will always have the same length. Row\_index contains indices from the Non\_zero\_value array of every value which is the first non-zero value in each row and a total length of the Non\_zero\_value array at the end. An example of this kind of sparse matrix representation can be seen below.

Non\_zero\_value = [12, 10, 8, 21, 3, 1, 45]

Row\_index = [2, 0, 1, 1, 1, 0, 2]

Column\_pointer = [0, 1, 3, 4, 5, 7]

### Modified nodal analysis (MNA)

Mesh and nodal analysis are fairly easy to use by humans, but they are impossible for a computer to understand and solve without any modifications. MNA is a much more complex way of analysing a circuit since it often results in a larger system of equations which is stored inside matrices, but it is also much easier to implement it algorithmically on a computer. Once an MNA is applied to a circuit it will create a matrix equation in the following form:

Where matrices A and z only hold known quantities, while matrix x holds quantities (node voltages and branch currents) which need to be calculated. To Find quantities stored in x, the equation needs to be rewritten. Matrix A is moved to the right-hand side and so it becomes a transpose matrix:

Each of the three matrices are made up of several sub-matrices.

#### A Matrix

A matrix consists of 4 other matrices G, B, C, D which sizes are N \* N, N \* M, M \* N and M \* M respectively, where N is a number of nodes in the circuit and M is a number of independent voltage sources.

The G or conductance matrix is made according to two following rules:

1. Each element in the diagonal matrix is equal to the sum of the conductance (one over the resistance) of each element connected to the corresponding node.
2. The off diagonal elements are the negative conductance of the element connected to the pair of corresponding nodes.

The B or incidence matrix is a sparse matrix which describes a relation between passive components and each voltage source. If a component is connected to a positive terminal of a voltage source, element 1 is written to the B matrix. If a component is connected to a negative terminal, then the element in the matrix is -1. If there is no connection between the two, the element is 0.

The C matrix is just a transpose of a B matrix. A transpose of a matrix is made by interchanging the rows and the columns. So, if a B matrix is: , then the C matrix is: .

Finally, the D matrix is a matrix which is entirely composed of 0s.

#### x Matrix

The x matrix holds the unknown quantities which need to be calculated. It is formed of two smaller matrices, v and j:

The v matrix contains unknown nodal voltages, and its size is determined by the number of nodes. If there are N, then there are N number of nodal voltages and therefore N number of voltage values in the v matrix.

The j matrix is made up of unknown currents flowing through the independent voltage sources in the circuit. If there are M voltage sources, then the j matrix will have an M number of current elements (one for each voltage source).

#### z Matrix

The z matrix holds the values of independent voltage and current sources in the circuit. It is also made up of two smaller matrices, i and e:

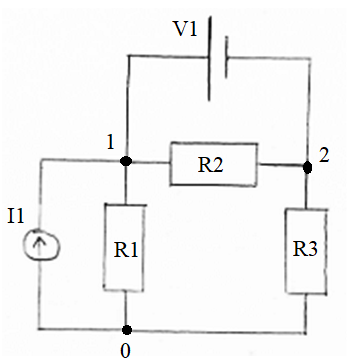
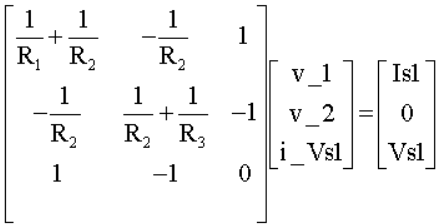
The i matrix consists of the sum of the currents flowing through the passive components. Each element in the matrix corresponds to a particular node. If there are no current sources connected to the node, the current value is 0.

The e matrix holds the values of corresponding independent voltage sources. Therefore, its size is determined by M, a number of voltage sources in the circuit.

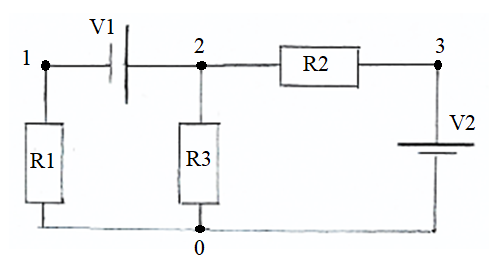
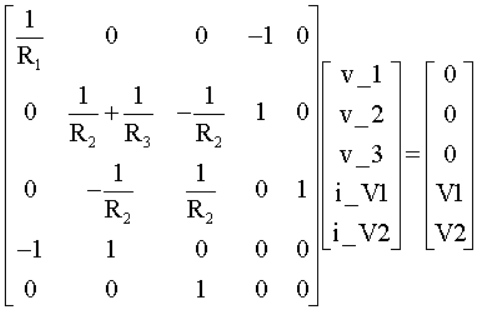
#### Examples

Below are two example circuits and their corresponding matrix equations.

Example 1

Example 2

### Netlists

A netlist is a text representation of a circuit. It consists of a list of electronic components and a list of nodes they are connected to. Its primary function is to describe connectivity in a circuit in a way that a computer can understand it. Netlists are written in lines with each line describing properties of one component. Netlists follow a certain structure. The line begins with the name of the component which tells us if it is a resistor, voltage source, or current source, then follows with two nodes which component is connected to, and in the end, a numerical value with corresponding units of the component. Netlists are also non case sensitive. Multiple spaces in a netlist file are dealt with like a single space.

Below are netlists of the previous example circuits (component values were not defined in the previous examples, so they were added subsequently):

Example 1: Example 2:

V1 1 2 9 V1 2 1 9

I1 1 0 5 V2 0 3 9

R1 1 0 10 R1 1 0 10

R2 1 2 10 R2 2 3 10

R3 0 2 10 R3 2 0 10

## List of components

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Component’s name | Alphabetic Symbol | Unit | Description | Graphical Symbol |
| **Voltage source** | V | Volt (V) | A component with two distinct terminals that provides constant voltage independent of current drawn from it. |  |
| **Current source** | I | Amp (A) | A component with two distinct terminals that supplies the same current to any load connected across its terminals. |  |
| **Resistor** | R | Ohm (Ω) | A component used to reduce current flow and drop voltage potentials by absorbing electrical energy. |  |

## Limitations

A limitation of a current system is that just by solving the questions set by teacher, some students do not develop a real understanding of how components interact in a circuit. A visual or kinetic learner may struggle to see how mathematical equations work in theory. A visual representation of a problem could greatly improve their intuition when approaching a circuit question.

However, while students might develop or deepen their understanding of a topic with a proposed solution, it will not teach them how to use the needed equations to solve a problem. As I stated before, a human and a computer solve the same circuit problem using different methods. The program will not be able to show a full walkthrough of a problem, because the program and the user would have taken different steps to arrive at the same solution. Displaying the program’s mathematical approach to a student who was just introduced to the topic will only further confuse them. Nevertheless, having the ability to construct a circuit and see the final results of it would, on its own, be of a great help to many students.

## Objectives

1. Make a GUI (Graphical User Interface) which will give the user all necessary options to construct a DC circuit.
   1. There needs to be one button for each component which will allow the user to add the component to the main tab. By clicking on the button, a chosen component should appear in the main tab.
   2. A drag and drop system needs to be created for all the components which are currently in the main tab. Once a component is added to the main tab, a user should be able to change its x and y coordinates by clicking on the component and dragging it across the screen.
   3. There needs to be a way for the user to create a network of components in the main tab. User will be able to create a connection between two components by clicking on them with mouse’s right click button. By repeating this process, a user will be able to create a DC circuit network
   4. When a component in the main tab is selected by the user, a suitable feedback should be given to the user showing which component they have selected. This will be done by changing the components appearance so that it becomes unique in respect to other components of the same type in the main tab.
   5. For a selected component, the user should be able to see its name, value, unit prefix, unit and a short description of components use in the circuit. This will be displayed in the specific part of the screen designed for this use.
   6. Each type of component should have their default value and unit prefix, but once a component is selected and its characteristics are displayed, the user should be able to alter the component’s value and unit prefix by typing their own components attributes.
   7. No value a user inputs as a component value and a unit prefix should cause a program to crash. A validation system will be used to check if correct values have been entered. If the values the user entered are incorrect or the unit prefixes do not exist, the user will be shown an appropriate message explaining that only numbers can be used as the component values or the user will be shown a list of possible unit prefixes.
2. In case the user makes a mistake while building a circuit (adds a wrong component to the main tab, makes a wrong connection between two components, etc.) they should be able to correct it by clicking on the specific buttons in the toolbar.
   1. A button which will undo the user’s latest action should be included, so that if the user notices their mistake immediately, they can quickly correct it.
   2. In case a whole circuit is built in an incorrect way, the user should be able to clear the whole main tab. This button will delete all components which are currently there.
3. Once a circuit is completed, it should be analysed and converted into a format which computer can understand.
   1. Once the analysis of the circuit is complete, a netlist in a text format should be created and stored as an external file.
   2. If a computer detects an error, analysis should be stopped, and an appropriate message should be displayed informing the user that circuit was built in an incorrect way.
4. User will have an option to save the circuit once it was analysed. After the circuit’s netlist was created and stored in the external file, user will be asked if they would like to save the circuit. If the user chooses not to save it, the file containing the circuit’s information will be deleted after appropriate calculations have been made (MNA) and final results have been displayed to the user. If the user decides to save it, file will be stored in a specific folder.
5. Once a netlist is stored in a file, MNA will be initiated and once it is complete, final results will be displayed to the user.
   1. The program should begin MNA analysis by processing the information of a netlist. This will be done by first parsing a netlist file and then mapping the nodes into indexes of the MNA matrices.
   2. Make a A matrix by combing G, B, C and D matrices together.
      1. Use the conductance values in the circuit to construct a G matrix
      2. Use sparse matrices to make a B matrix.
      3. Use a transpose of B matrix to get a C matrix.
      4. Fill a D matrix will zero values
   3. Construct an x matrix using the matrix of unknown nodal voltages and matrix of unknown currents flowing through independent voltage sources.
   4. Construct a z matrix using the matrix containing the voltages of independent voltage and matrix containing the currents of the independent current sources.
   5. Once all three main matrices are constructed, using the MNA equation, all unknown values in matrix x need to be calculated.
6. During MNA, once the matrices are constructed, they need to be stored using the sparse matrix optimisation techniques. This needs to reduce the overall time needed to perform all the calculations. To test it, I will be timing the program’s time to calculate the circuit outputs with and without the use of spare matrices.
7. The simulator should be user friendly allowing anyone with limited IT skills to use the program. Five people will be asked to use the program and rate it with a maximum of ten points and a minimum of one point. They will be students in either year 12 or 13 who are taking physics as one of their A – level subjects and preferably not taking computer science.

## References

1. <https://isaacphysics.org/questions/series_parallel_resistors?board=b730b7f6-977c-4f5e-a552-44c6d220ef89> – Picture of a circuit, *Accessed on 5.5.2020.*
2. <https://www.physicsclassroom.com/Physics-Interactives/Electric-Circuits/Circuit-Builder/Circuit-Builder-Interactive> – Example 1 of a current system, *Accessed on 10.5.2020.*
3. <https://www.circuitlab.com/> - Example 2 of a current system, *Accessed on 9.5.2020.*
4. <http://phet.colorado.edu/sims/html/circuit-construction-kit-dc/latest/circuit-construction-kit-dc_en.html> Example 3 of a current system, *Accessed on 9.5.2020.*
5. <https://electronics.stackexchange.com/questions/394068/create-circuit-simulation> – Explanation of how SPICE engine works, *Accessed on 26.6.2020.*
6. <https://en.wikipedia.org/wiki/HSPICE> – SPICE home web page, *Accessed on 27.6.2020.*
7. Breithaupt J. 2015., AQA Physics, 2nd Edition, Oxford University Press, United Kingdom
8. <https://www.tutorialspoint.com/network_theory/network_theory_mesh_analysis.htm>– Explanation of a mesh analysis 1, *Accessed on 11.10.2020.*
9. <https://www.tutorialspoint.com/network_theory/network_theory_nodal_analysis.htm> – Explanation of a nodal analysis 1, *Accessed on 11.10.2020.*
10. <https://www.electronics-tutorials.ws/dccircuits/dcp_6.html> – Explanation of a nodal analysis 2, *Accessed on 11.10.2020.*
11. <https://www.electronics-tutorials.ws/dccircuits/dcp_5.html>– Explanation of a mesh analysis 2, *Accessed on 13.10.2020.*
12. <https://steemit.com/utopian-io/@drifter1/electronic-circuit-simulation-mesh-analysis-python>– Explanation of a mesh analysis 3, *Accessed on 10.10.2020.*
13. <https://steemit.com/utopian-io/@drifter1/electronic-circuit-simulation-nodal-analysis-python>– Explanation of a nodal analysis 3, *Accessed on 11.10.2020.*
14. <http://qucs.sourceforge.net/tech/node14.html#SECTION00415200000000000000> – Explanation of a modified nodal analysis 1, *Accessed on 23.11.2020.*
15. <https://numpy.org/doc/stable/reference/generated/numpy.linalg.solve.html> – NumPy documentation for solving simultaneous equations with matrices, *Accessed on 19.11.2020.*
16. <https://steemit.com/utopian-io/@drifter1/electronic-circuit-simulation-incidence-matrix-and-modified-kirchhoff-laws-python> – Explanation of an incidence matrix 1, *Accessed on 19.11.2020.*
17. <https://en.wikipedia.org/wiki/Incidence_matrix> – Explanation of an incidence matrix 2, *Accessed on 23.11.2020.*
18. <https://en.wikipedia.org/wiki/Sparse_matrix> – Explanation of a sparse matrix 1, *Accessed on 24.11.2020.*
19. <https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.csc_matrix.html> – SciPy documentation for CSC sparse matrix operations, *Accessed on 24.11.2020.*
20. <https://steemit.com/utopian-io/@drifter1/electronic-circuit-simulation-sparse-matrix-optimization-part-1-python> – Explanation of a sparse matrix 2, *Accessed on 24.11.2020.*
21. <https://en.wikipedia.org/wiki/Netlist> – Explanation of a netlists 1, *Accessed on 8.12.2020.*