PCB Auto-routing

Computer Science Coursework

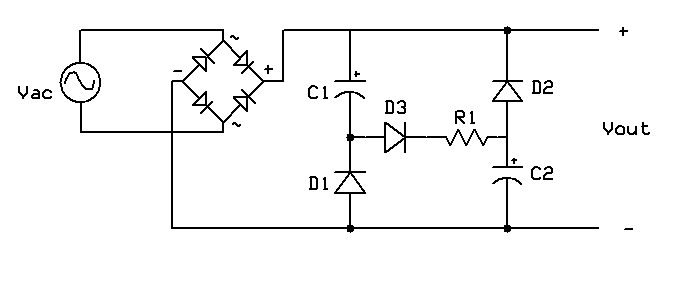
Edwin Shepherd

# Analysis

## The Problem

Electronics engineers develop circuits which are often printed on to a circuit board (PCB), keeping wiring tidy. However routing the wires is a very tedious task. Additionally the wiring must conform to rules or else malfunctions in the circuit may occur. For example, high current tracks must be of a set width, certain components cannot have tracks under them. This is takes more time and is difficult to ensure 100% accuracy.

To solve this problem PCB Routing Software is common place in PCB design tools. Its purpose is to automatically route all of the components and to perform “Design Rule Checking” DRC. PCB software usually has two modes one where the circuit is designed:

[](https://www.google.co.uk/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwjSqtfQ-NPdAhVIWBoKHcJ_DAsQjRx6BAgBEAU&url=https://en.wikipedia.org/wiki/File:Valley-fill_circuit_schematic_1.png&psig=AOvVaw1u6-1MKZyxtGEkY9-Jxwfn&ust=1537888868345021) Figure 1 a schematic for a basic diode rectifier coupled with a valley-fill circuit - Image credit: WikiMedia available: <https://goo.gl/NJdaqG> , accessed 16:28 24/09/2018

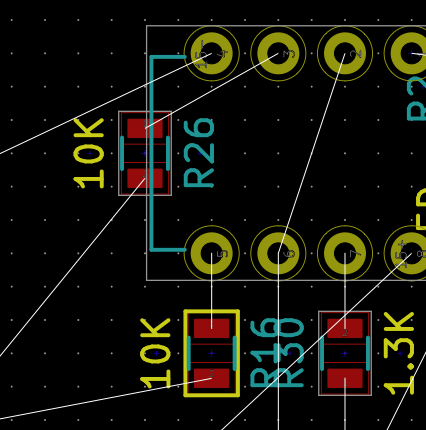
[](https://www.google.co.uk/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwj-0Oz0rqveAhUOThoKHfDXAwgQjRx6BAgBEAU&url=https://electronics.stackexchange.com/questions/200093/what-does-yellow-border-around-component-in-kicad-pcbnew-mean&psig=AOvVaw12652bQlXOE6Df1Zygvx7C&ust=1540892724447123)Then another mode where the “nets” highlighted, these are the connections between components (in figure 2 they are blue) and then can be routed by the user (or by software). Finally it’s converted into a file that can be printed.

Figure 2 - Nets are shown in white straight lines. This screen shot is taken from KiCAD's PCBnew software. Image credit; Scanny avalible <https://goo.gl/muyghy> date accessed 29/10/2018

In this project I will solely focus on the challenges that are presented by the routing.

**End user**: Stuart Jessup

# Research

## Interview

After speaking with my client there are a range of areas that my project could support and they were ranked according to value. The list below is ranked.

1. Being able to route a board
2. Multiple layer support
3. Different track widths
4. Keep out areas
5. Thermal sinks
6. Isometric routing
7. Integration with EDA (Electronic Design Application) software.

## Web research

Auto routing is a hard problem which is not easily solved by

* a brief introduction that describes the **problem area**and specific problem that you are solving or investigating. ​Make it clear who your **user/consultant**is (name them). This can be an updated version of your original proposal. ***TICK!***
* an outline of **how you researched and investigated**the problem. Try to use a range of sources: interviews/questionnaires, research on the web, documents from any existing system. Include a ​summary of your **prototyping** work.
* provide **background** in enough detail for the reader to understand the problem. Include**evidence** such as intervew questions and answers, URLs, screenshots. Illustrations are a very powerful and concise way of communicating this.
* produce a **numbered list of measurable, specific objectives**, covering all required functionality of the solution or areas of investigation. This is the most important part of your analysis. Make sure your objectives are unambiguous, detailed and comprehensive, describing everything that you are aiming to achieve.The success (or otherwise) of your solution will be judged against this. Over the Autumn Term you will develop and refine this list, helped by your teacher and your own experiences during the prototyping period.

Be detailed and specific, not vague and waffly:

|  |  |
| --- | --- |
| **Don't say** | **Do say** |
| "it must be able to store all the data" | "it must be able to store details of orders, customers, materials...." |
| "it must produce all the necessary printouts" | " it must generate reports on: overdue subscriptions, analyses of sales, mailmerged letters….." |
| "my system must be user-friendly" | "my user must be able to use it after 10 minutes training" or "my user must be able to train themselves using my user guide" or "the interface must be entirely menu-driven" |
| " my system must be fast and efficient" | "my system must be able to retrieve the …. data within 2 seconds" |
| "my system must be secure" | "the …data must be encrypted", "there must be a backup routine included" |

 ​ ​

​There are lots of different ways of structuring this list. You could divide it into sections covering hardware constraints, user-related issues, security, input, storage, processing, output…. It really depends on the nature of the project and your preferences, but try to provide some sort of logical sequence, rather than a random list.

It's strongly recommended that you **grade or prioritise**your objectives: divide your list into two or three sections: start with the "must do" ones, then move on to the "could do" or extension ones. This will give you flexibility and ensure you have enough to keep you busy. You will not be penalised if you do not complete meet all your objectives, as long as you explain why in your evaluation.

The examiners want a **numbered** list and objectives should be "**SMART**" (Specific, Measureable, Achievable, Realistic, Time-related).

Even presented concisely as a numbered list, this is likely to take up a page of your report.

* appropriate **diagrams and models**to illustrate your problem; for example: an ER model (essential for any project that will include designing a database); if you intend to take an Object-Oriented approach then identify the classes you think you will need and draw an object diagram showing the relationships between them (agg​regation/composition/inheritance).
* a **brief** outline of the techniques and methods you intend to use. For example, the core data structures and algorithms, whether you intend to use a DBMS, whether you wil​l be using an Object-Oriented approach.