How To Use the Schematic and Board Importer in CadSoft EAGLE for modularized Projects

Doc. Version 2013-06-20-1

Author: Mario Blunk

Abstract: *Guideline to use the schematic and board (layout) importing functions provided since CadSoft EAGLE V6.x via example schematics and board layouts. Focus on modularizing projects in a multi-user environment and precautions to care for before importing. Net classes, net labels and net naming are addressed.*

Keywords: *schematics, board, layout, PCB, module, merge, import, label, off-page connector, net, class, supply, grid, library, prototype, instantiate, re-usability*

Contents

Preface

In a multi user environment i.e. in a corporation with lots of engineers working on the same EAGLE project, modularizing schematics and layouts is a must. The big advantages of modularized schematics and layouts:

- re-usability of modules
- distributed working

Since EAGLE version 6.x, improved schematic and board import functions have been provided. This document assist engineers in a tutorial like manner how to use these functions. The example project here is an RGB driver consisting of six sub modules:

- two individual but equal PIO connectors
- three individual but equal LED drivers (for the red, green and blue channel respectively)
- an oscillator module

These modules have been designed independently of each other and will be merged/imported into the final product called “RGB driver”.

1 Initial Situation

There are two different initial cases to consider. This document addresses both scenarios in detail:

CASE #1:

There are individual schematics - without any board layout - which will merge into a final schematic.

This scenario arises where the board does not matter yet, where the focus is on schematics only. Here some examples of things which are usually not specified at this early development stage:

- board outline
- device packages
- board layer stack
- via types
- EMC (electromagnetic compliance)
- SI (signal integrity)
- DFT (design for test)
- various manufacturing details

Furthermore there may be schematic modules which need to be multiplied once they have been designed completely.

CASE #2:

Ready made board layouts - together with their schematics - are to be merged with each other.

This situation arises in projects where:

- a main module layout is approved and not to be changed any more, and further sub module layouts are to be added, depending on the final product version.

- there may be board layout sections which need to be multiplied once they have been designed completely.

Section 3 from page 27 on addresses this scenario in detail.

2 Case #1 - Merging Schematics only

2.1 General Workflow

1. Draw sub module schematics with development frames (Figure 1).
2. Care for net classes and net names according to project wide conventions.

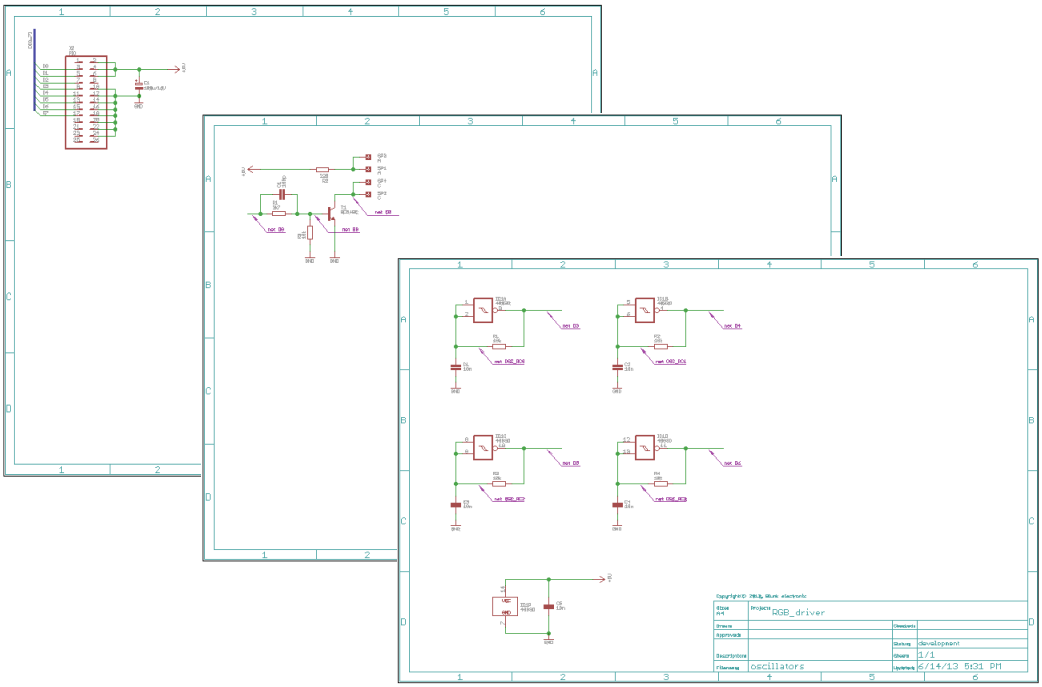


Figure 1: individual sub module schematics

3. Create a new schematic file for the main module.
4. Optional: Draw core circuitry of the main module.

5. Import sub modules one after another (Figure 2).
6. Do fine tuning, documentation, ...
7. Replace development frames by “final product” frames.
8. Renumber devices in main module (using ULP renumber-sheet.ulp).

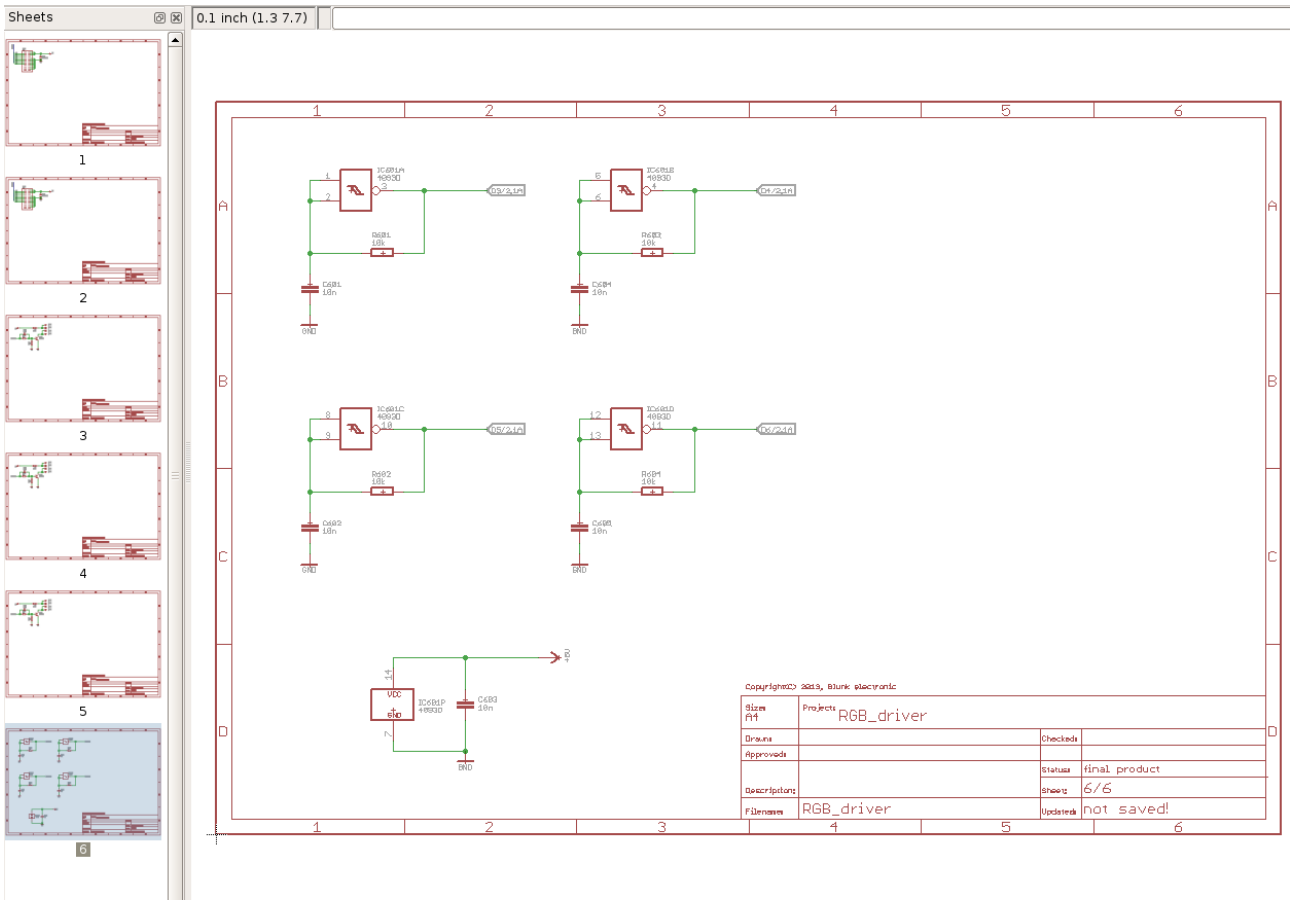


Figure 2: final main module schematic

2.2 Main Module Overview

To get an impression of where this tutorial is heading to, Figure 3 outlines the main module as it consists of sub modules and their interconnections.

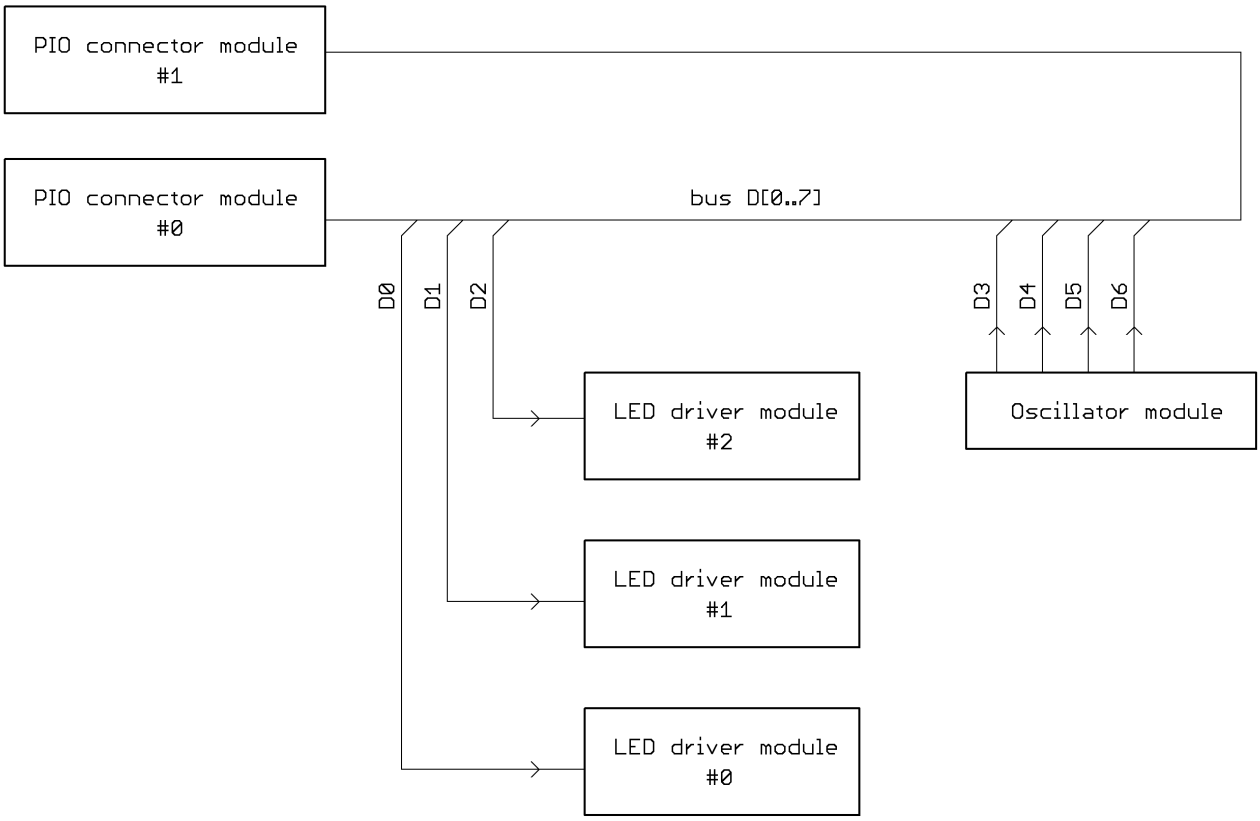


Figure 3: main module overview

2.3 Drawing Sub Modules

As said in the preface, the sub modules are drawn independently of each other. They may reside:

a) inside a global EAGLE project directory (recommended), see Figure 4

b) elsewhere in various EAGLE project folders (without the engineers/subcontractors knowing the higher level project they are contributing to)

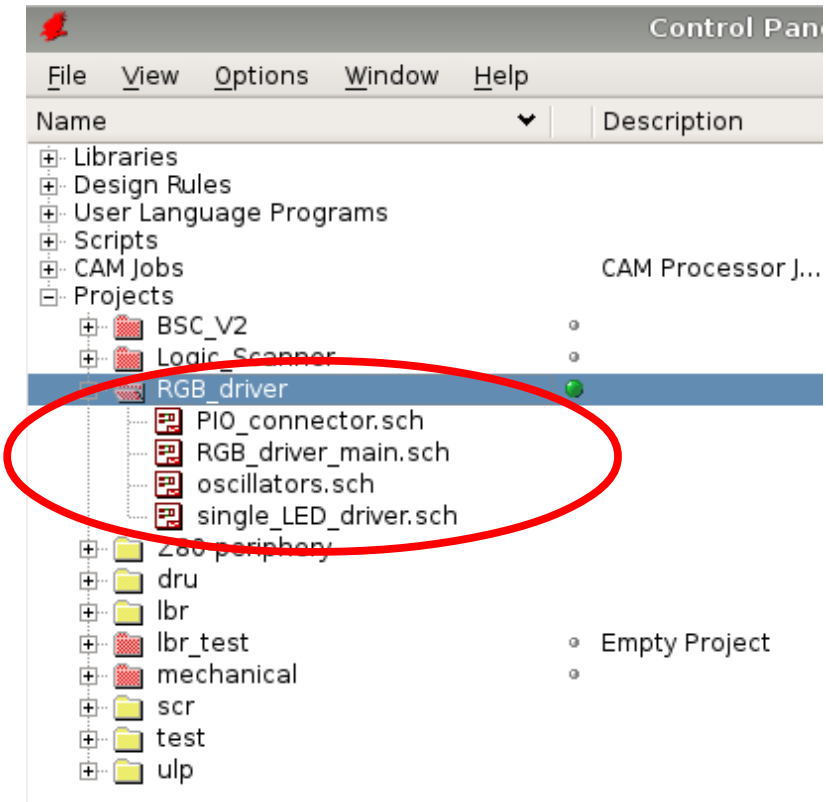


Figure 4: Project Overview

PRECAUTIONS:

1) Keep on a common net class naming. For example supply nets must be in class “pwr”, differential signals must be in class “diff”, mains voltage signals must be in class “mains” and so on.

2) All drawings must meet a common net naming convention for nets crossing the module boundaries. For example a data bus connecting modules must be named D[0..31] on all sub modules, the +3V3 rail must have this very name on all sub modules, ...

3) All project contributors should work on the same component library.

4) Keep on a common drawing grid in all modules (recommended 0.1 inch).

The sub modules we are working with in this tutorial consist of only one sheet each in order to keep things simple.

2.3.1 Sub Module “PIO_connector”

Figure 5 shows the drawing of the sub module “PIO_connector”. The power voltage is fed in here.

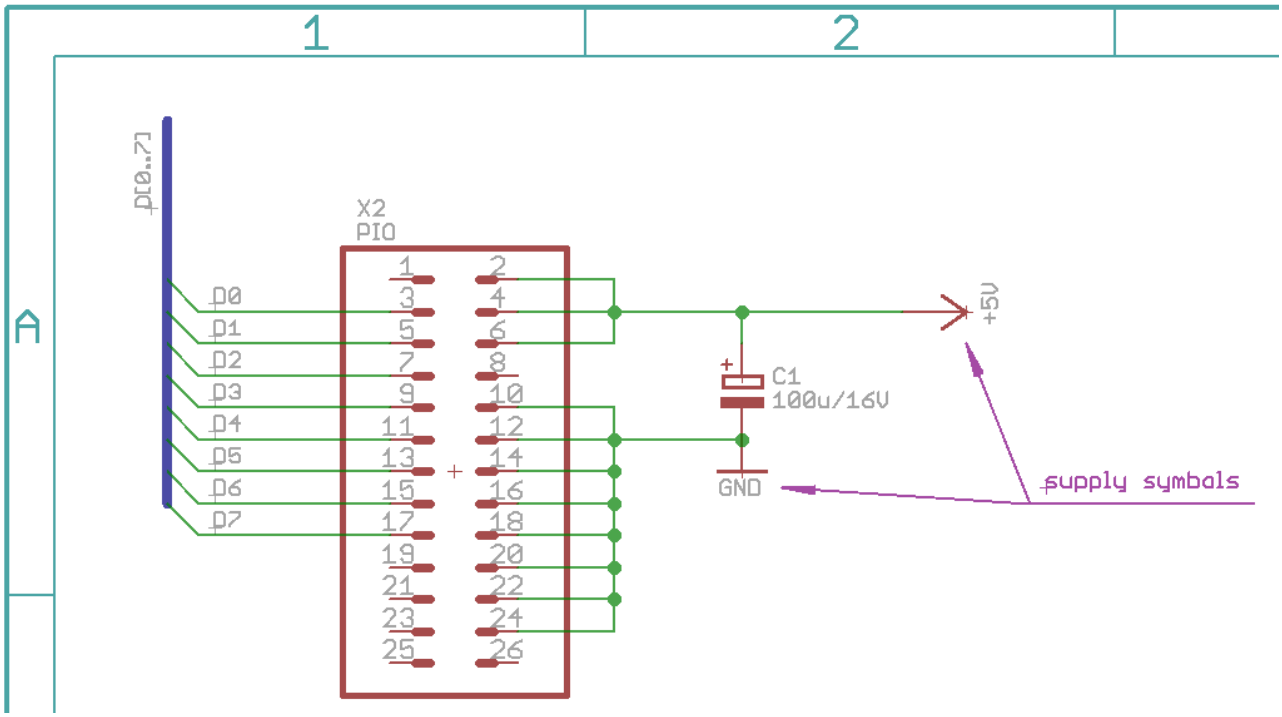


Figure 5: PIO connector

Important to point out the:

1) supply nets +5V and GND. A net becomes a supply net if it gets connected to a supply symbol like GND or +5V. Supply net names are by default not subject of indexing while merging with other modules. So the names of supply nets are static.

2) intentionally placed net labels for the bus D[0..7]. A net label disables the automatic indexing of these nets, instructing EAGLE to leave the respective net name untouched later during import. So the net names of bus D[0..7] are static too.

   The label type may be a regular one or an off-page connector (Figure 6).

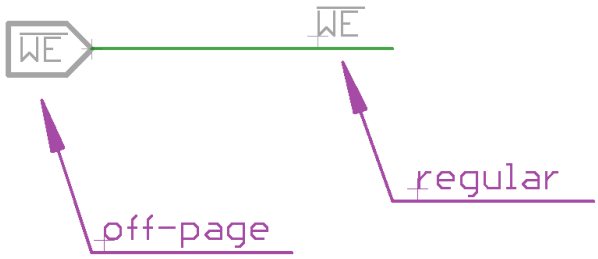


Figure 6: label types

3) net class “pwr” for all supply nets (use EAGLE commands *class* and *name*).

4) net class “default” for all non-supply nets (use EAGLE commands *class* and *name*).

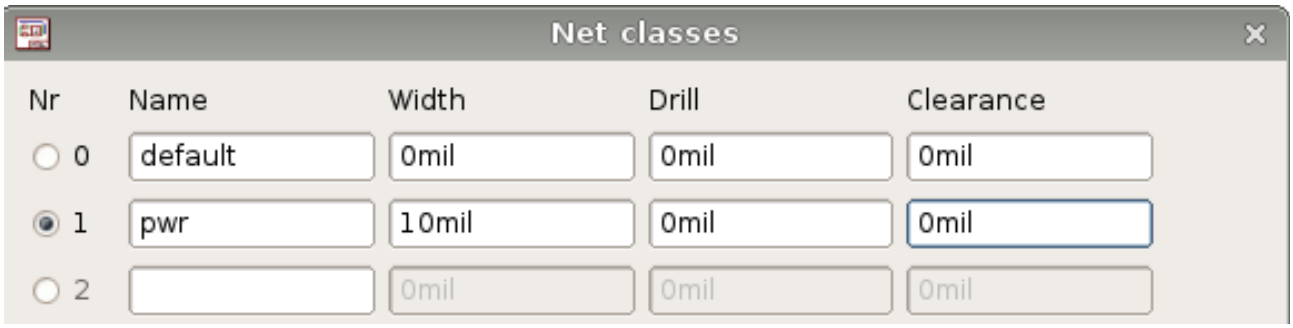


Figure 7: net classes

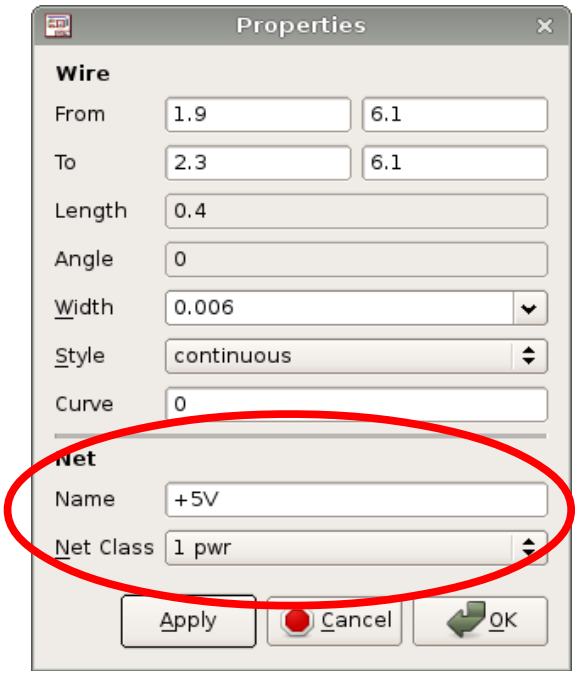


Figure 8: net +5V properties

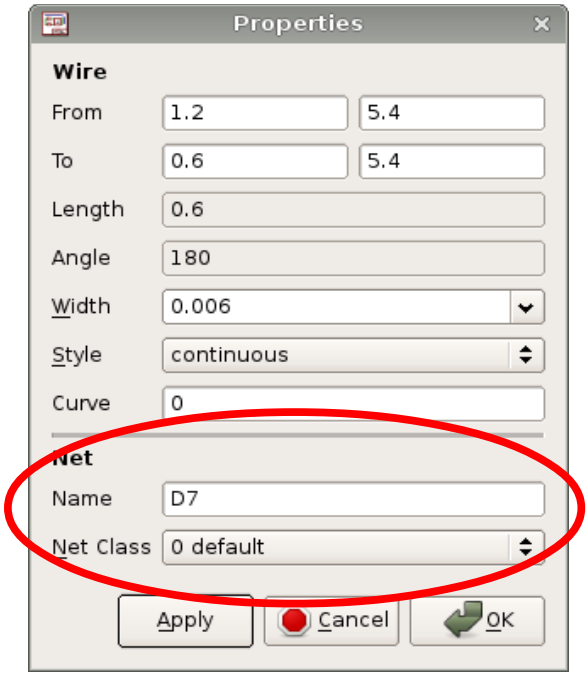


Figure 9: net D7 properties

2.3.2 Sub Module “single_LED_driver”

Figure 10 shows the schematic of the LED driver as it is drawn only once in file “single_LED_driver.sch”. Even if this sub module will be “instantiated” three times, we need only one “prototype”. Later this whole schematic file will be imported three times in three subsequent steps into the main module schematic.

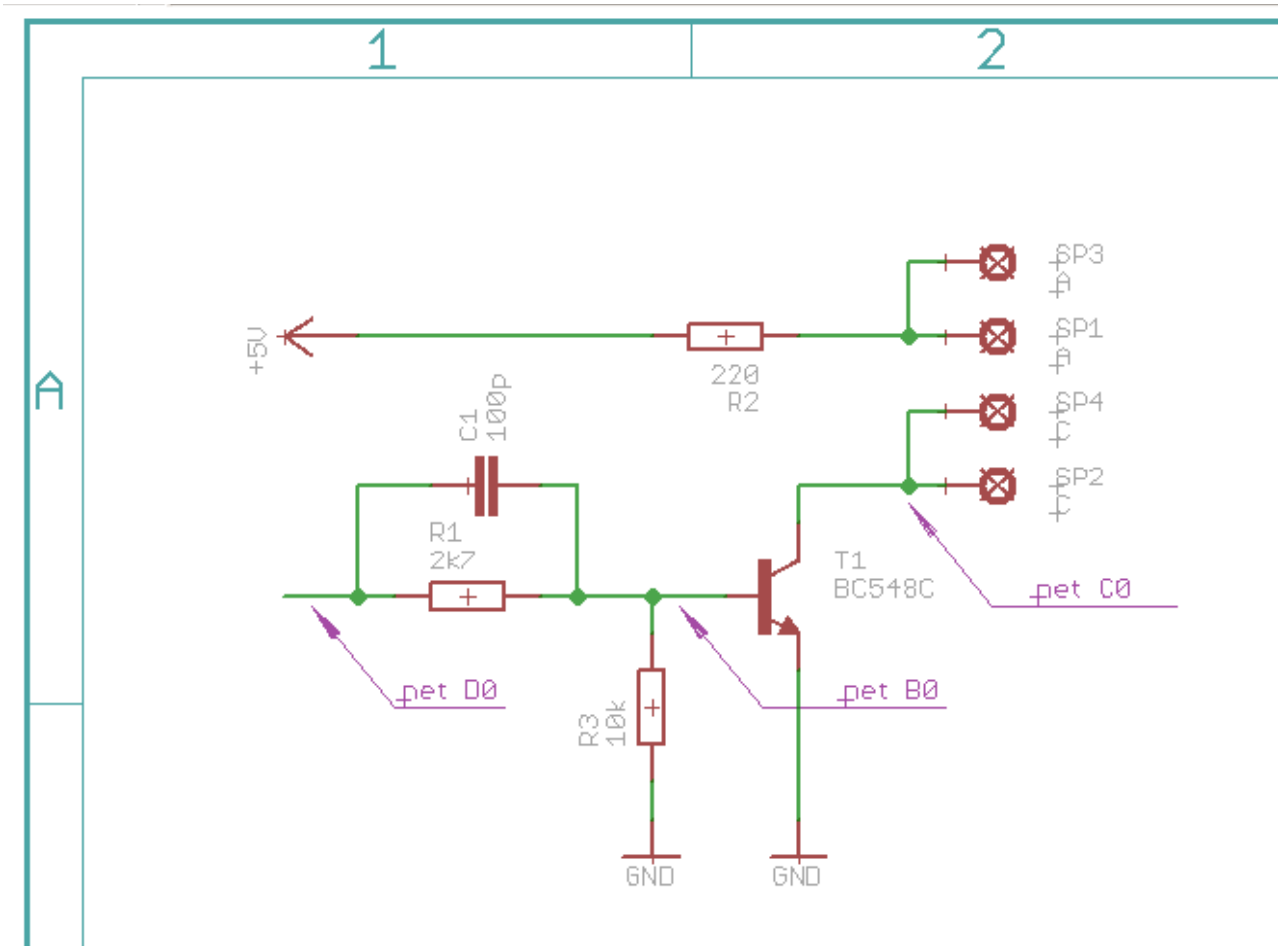


Figure 10: single LED driver

Important to point out the:

1) supply nets +5V and GND. A net becomes a supply net if it gets connected to a supply symbol like GND or +5V. Supply net names are not subject of indexing while merging with other modules. So the names of supply nets are static.
2) intentionally **not** placed net labels. The non-supply nets do have a name but do **not** have a label. The net name will be indexed later, starting with the lowest index number available while import into the main module. Later while import, the net name D0 will change to D1, D2, D3 and so on. The same applies for B0, C0 and all other non-labeled nets.

3) net class “pwr” for all supply nets (use EAGLE commands *class* and *name*).

4) net class “default” for all non-supply nets (use EAGLE commands *class* and *name*).

2.3.3 Sub Module “oscillators”

Figure 11 shows the schematic of the oscillator array as it is drawn only once in file “oscillators.sch”. Later this whole schematic file will be imported only once into the main module schematic.

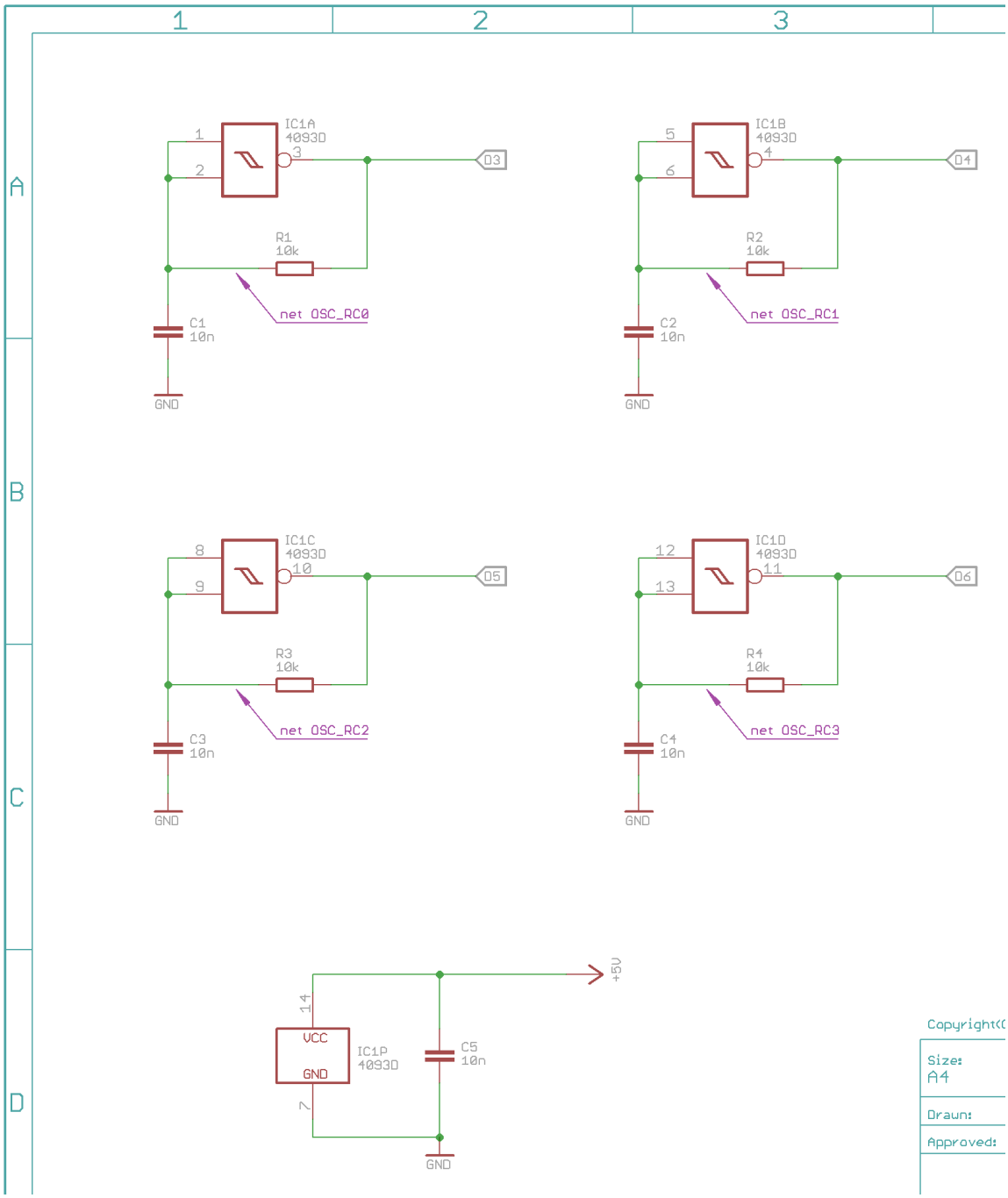


Figure 11: oscillator module

Important to point out the:

1) supply nets +5V and GND. A net becomes a supply net if it gets connected to a supply symbol like GND or +5V. Supply net names are not subject of indexing while merging with other modules. So the names of supply nets are static.
2) intentionally **placed** net labels on nets D3 to D6. At this development stage it is already clear, that the oscillators will drive onto these nets – no indexing allowed. The net names are static.
3) intentionally **not** placed net labels on nets OSC_RCx as they appear on this very sheet of this schematic only. Later while importing into the main module, this measure also prevents these nets to be connected inadvertently with other nets having the same names.
4) net class “pwr” for all supply nets.
5) net class “default” for all non-supply nets.

2.4 Set Up Main Module

Now that the sub modules are ready to use, the main module needs some attention:

1) If not there already, in the project directory “RGB_driver” a new and empty schematic named “RGB_driver_main” is to create (see red pointer in Figure 12).

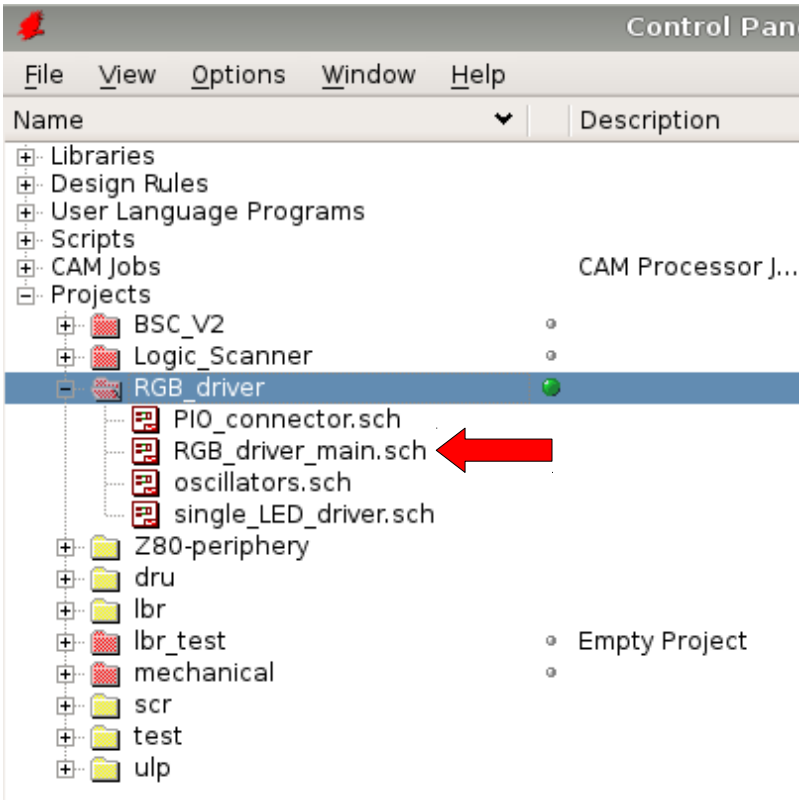


Figure 12: main module schematic

2) Optionally, certain essential core sections can be drawn now in it. Net class and net name conventions must be met in order not to contend with net classes and names of the sub modules drawn already. In this tutorial we skip this step.

3) Now inside the main module schematic, go to `Edit|Global Attributes` and define the project name placeholder and its value.

2.5 Import Sub Module “PIO_connector”

1) Go to `File|Import|EAGLE drawing` and select the “PIO_connector” as the first sub module to be imported (Figure 13 and 14).

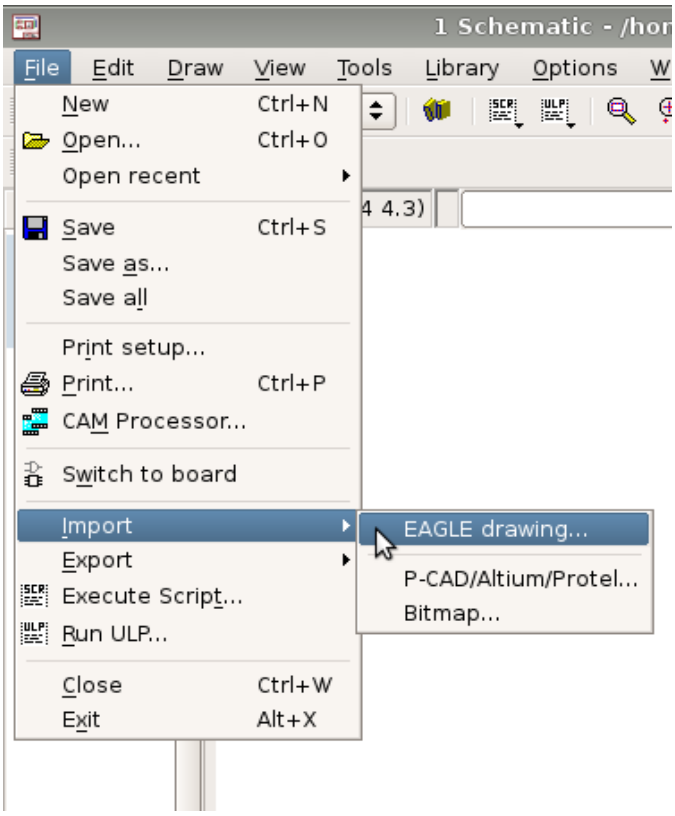


Figure 13: import EAGLE drawing

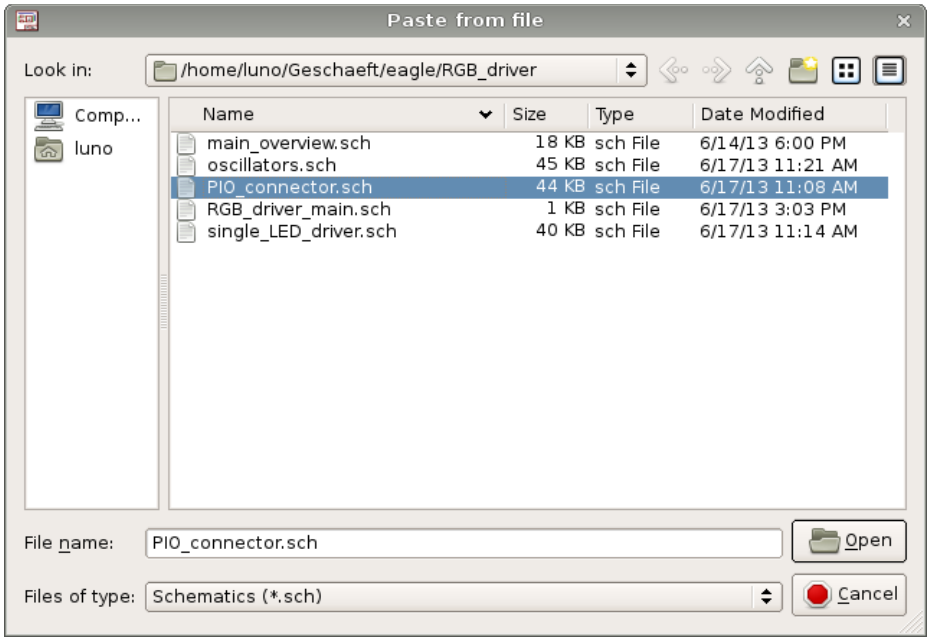


Figure 14: importing the PIO_connector sub module

2) Now a window pops up notifying you about possible net name indexing or static net names while import. The column “Old name” lists nets of the sub module **before** importing, the column “New name” lists nets of the sub module **after** importing. If an unknown net is introduced by the sub module, the respective field in the column “Old name” is empty. In our example in Figure 15 we are importing into a blank schematic. Hence the entire column “Old names” is empty.

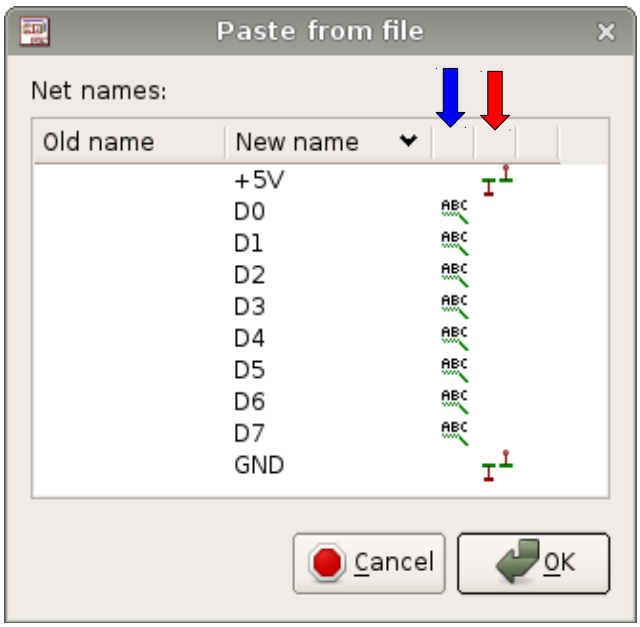


Figure 15: net names while import

There are two more columns to dwell on: The blue arrow in Figure 15 points to the column that shows whether a net name has been set static by a label. The column where the red arrow is pointing to indicates supply nets. Supply net names are static by default. However, the name may still be changed by left click on the name field in the “New name” column.

3) Upon click on OK the sub module gets imported into the main module schematic as shown in Figure 16.

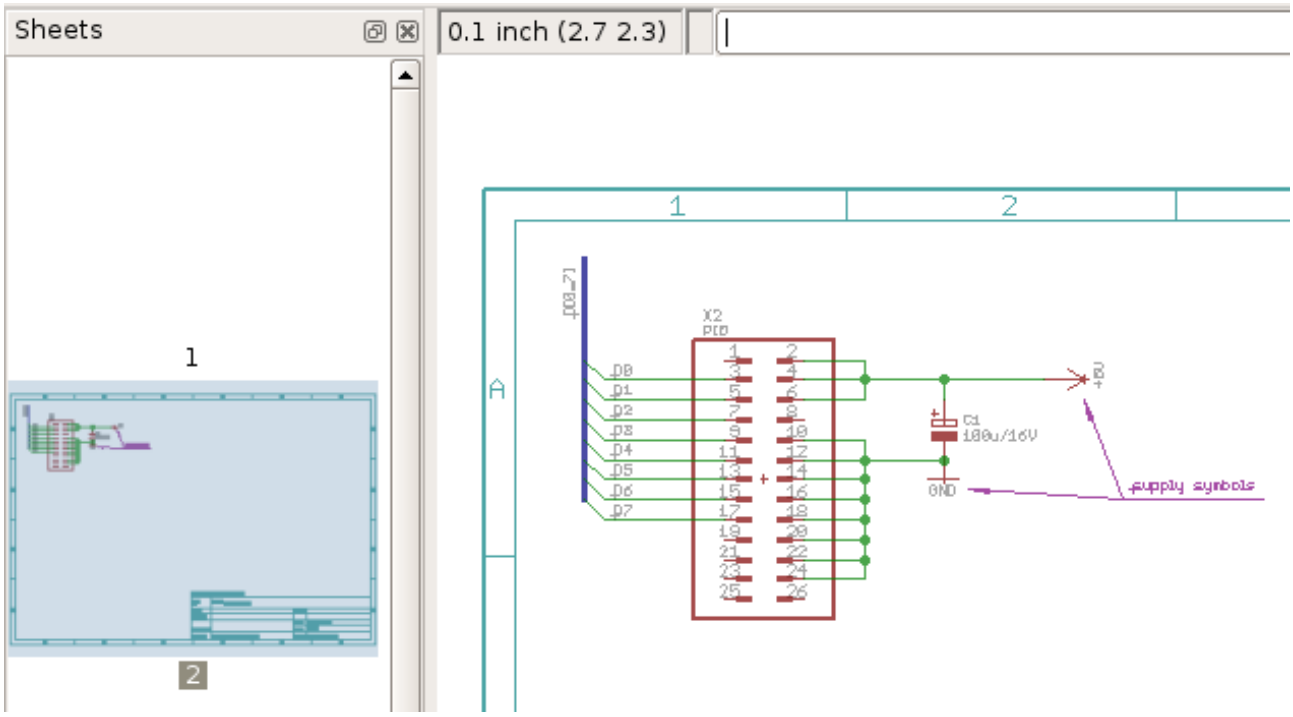


Figure 16: the first sub module imported

4) Cleaning up: Remove the empty default sheet #1 by right click on it (Figure 17).[1]

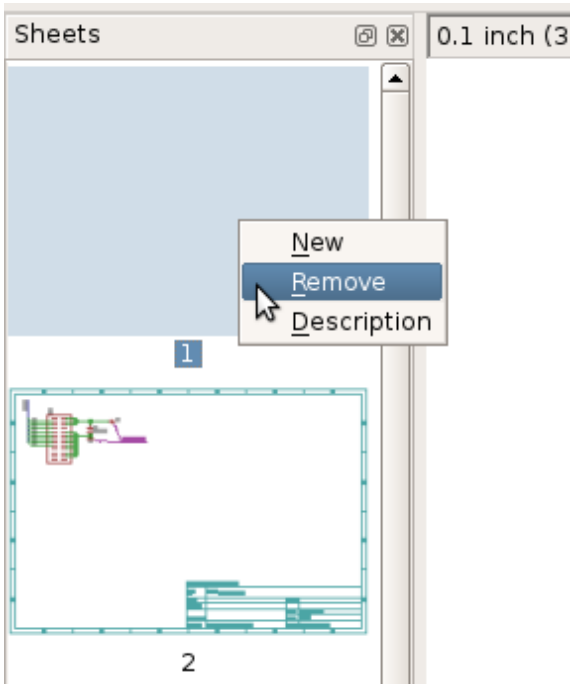


Figure 17: remove first sheet

1 The first sheet may contain the contents of the schematic and should not be removed in this case.

5) As outlined in the project overview (Figure 3 page 8) two “PIO_connector” modules are required. So please do step 1 on page 16 a second time.

6) Now the net names window shows a third column (**red** pointer in Figure 18) notifying you about static nets which will **not** change their name, thus will be connected with the main module. The column “Old name” lists nets of the sub module **before** importing, the column “New name” lists nets of the sub module **after** importing. Since they are equal, the sub module being imported is left as it is, without any net name changing.

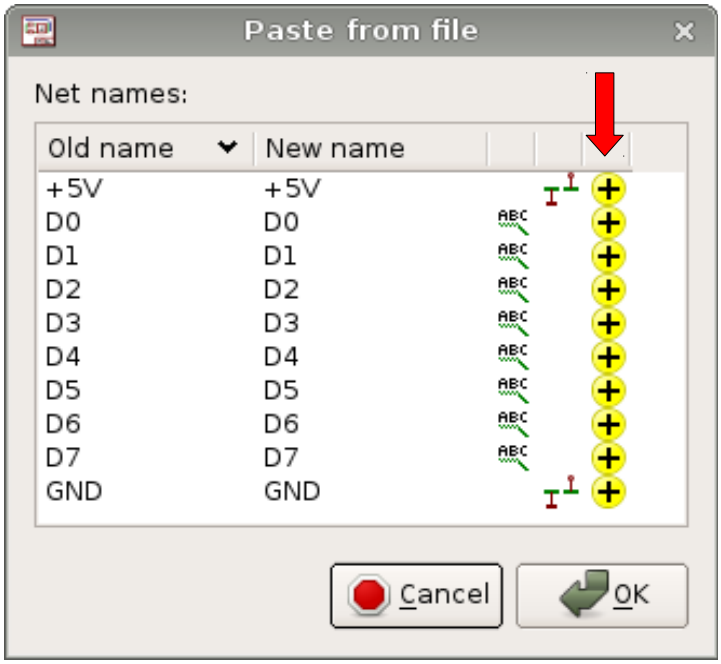


Figure 18: net names while import

7) Upon click on OK you get a main module schematic with two “PIO_connectors” connected to each other in parallel in a 1:1 manner.

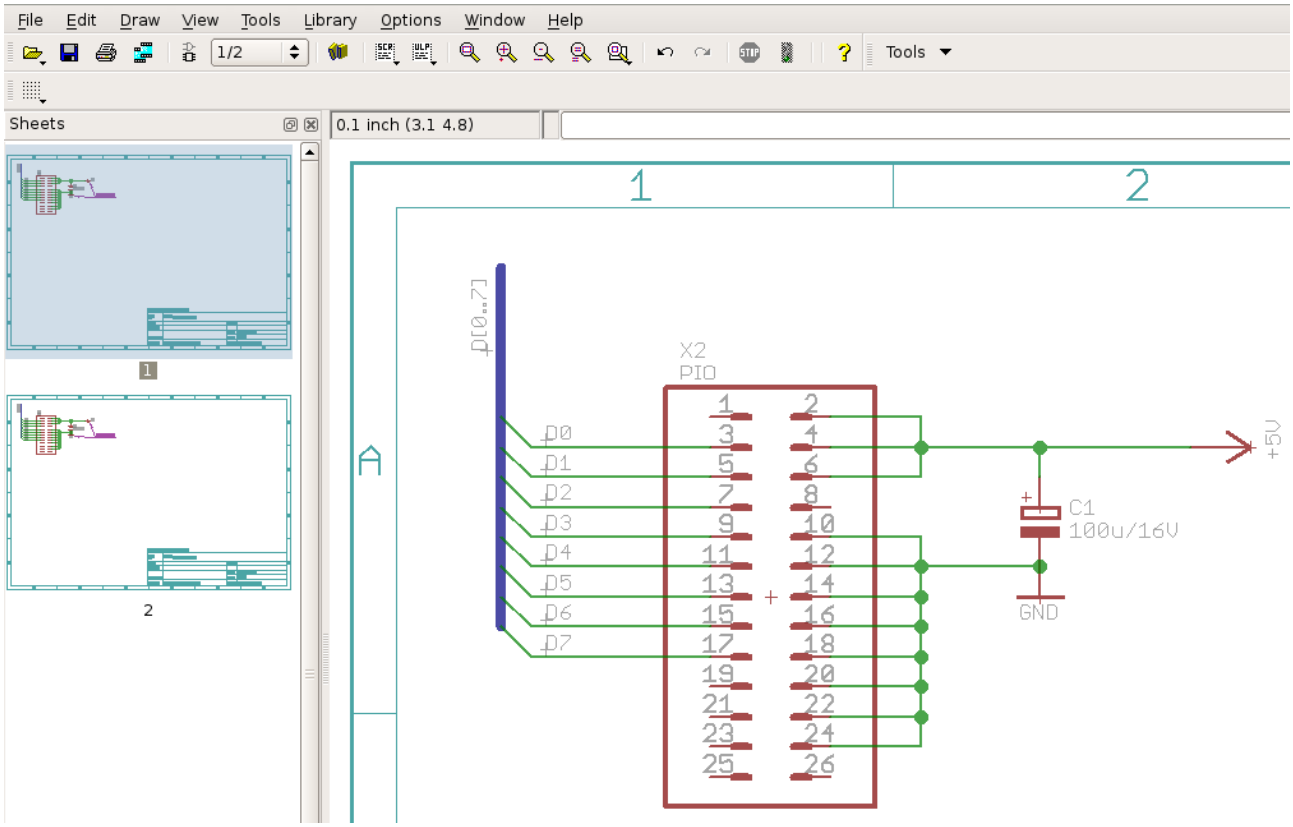


Figure 19: two PIO_connectors imported

2.6 Import Sub Module “single_LED_driver”

1) Go to `File|Import|EAGLE drawing` and select the “single_LED_driver” as the next sub module to be imported.
2) The net names window shows what happens to a sub module net if indexing is enabled: D0 is to be renamed as D8. This is a safety measure since the next available net of the D-group is D8. But this is not what we want. We want the input of the first LED driver to be connected with D0 of the main module.

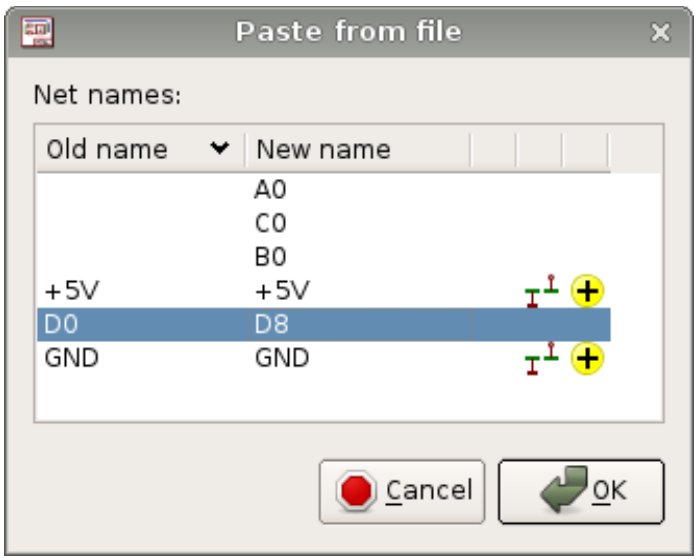


Figure 20: net names while import

3) So do a left click on D8 and change it to D0 as shown in Figure 21.

Figure 21: rename net

4) Now the net names window shows the intended action.

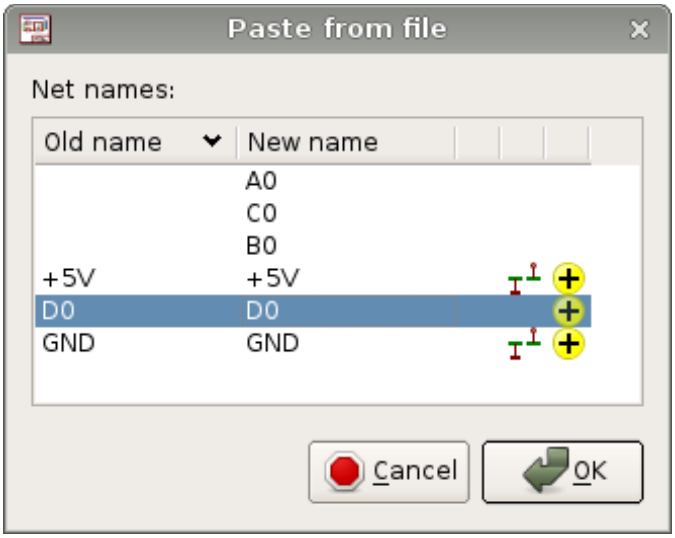


Figure 22: net D8 renamed

5) Upon click on OK the schematic of the main module gets extended by a third sheet holding the first “single_LED_driver” (Figure 23).

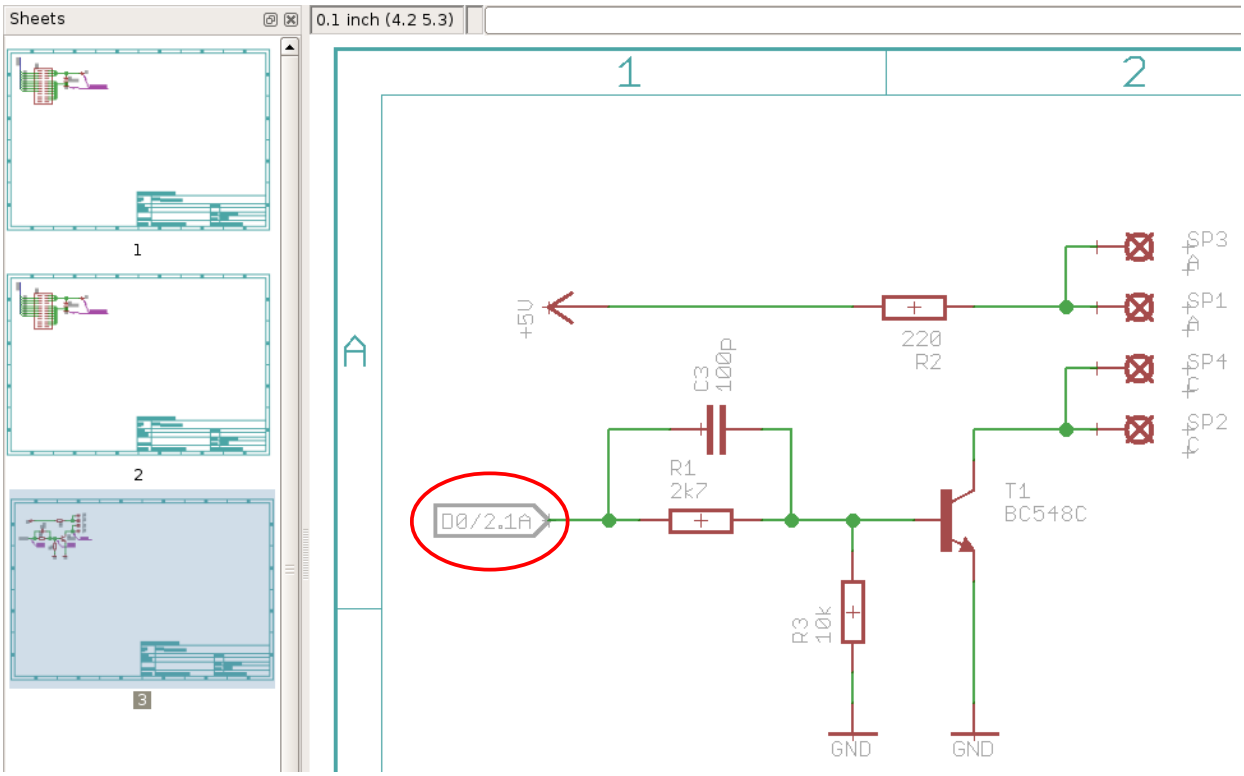


Figure 23: first LED driver imported

6) Place an off-page label on the input of the LED driver (**red circle** in Figure 23).

7) Repeat steps #1 through #6 for the remaining two driver modules with the input nets D1 and D2 respectively.

2.7 Import Sub Module "oscillators"

1) Go to `File|Import|EAGLE drawing` and select the "oscillators" as the next sub module to be imported.
2) The net names window shows exactly what we want: The static output nets D3 through D6 of the oscillator sub module keep their name and thus will be connected to the same nets of the main module. No renaming required.

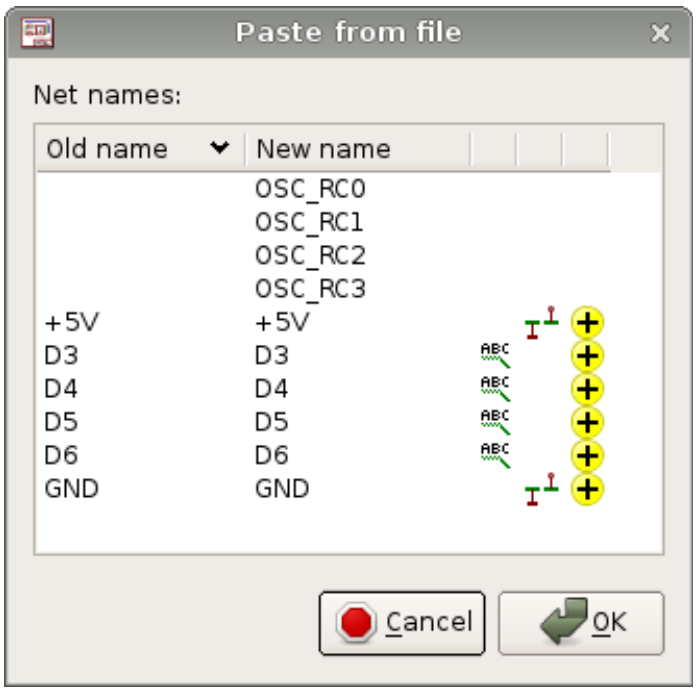


Figure 24: net names while import

3) Upon click on OK the schematic of the main module gets extended by a sixth sheet holding the "oscillators" sub module (Figure 25).

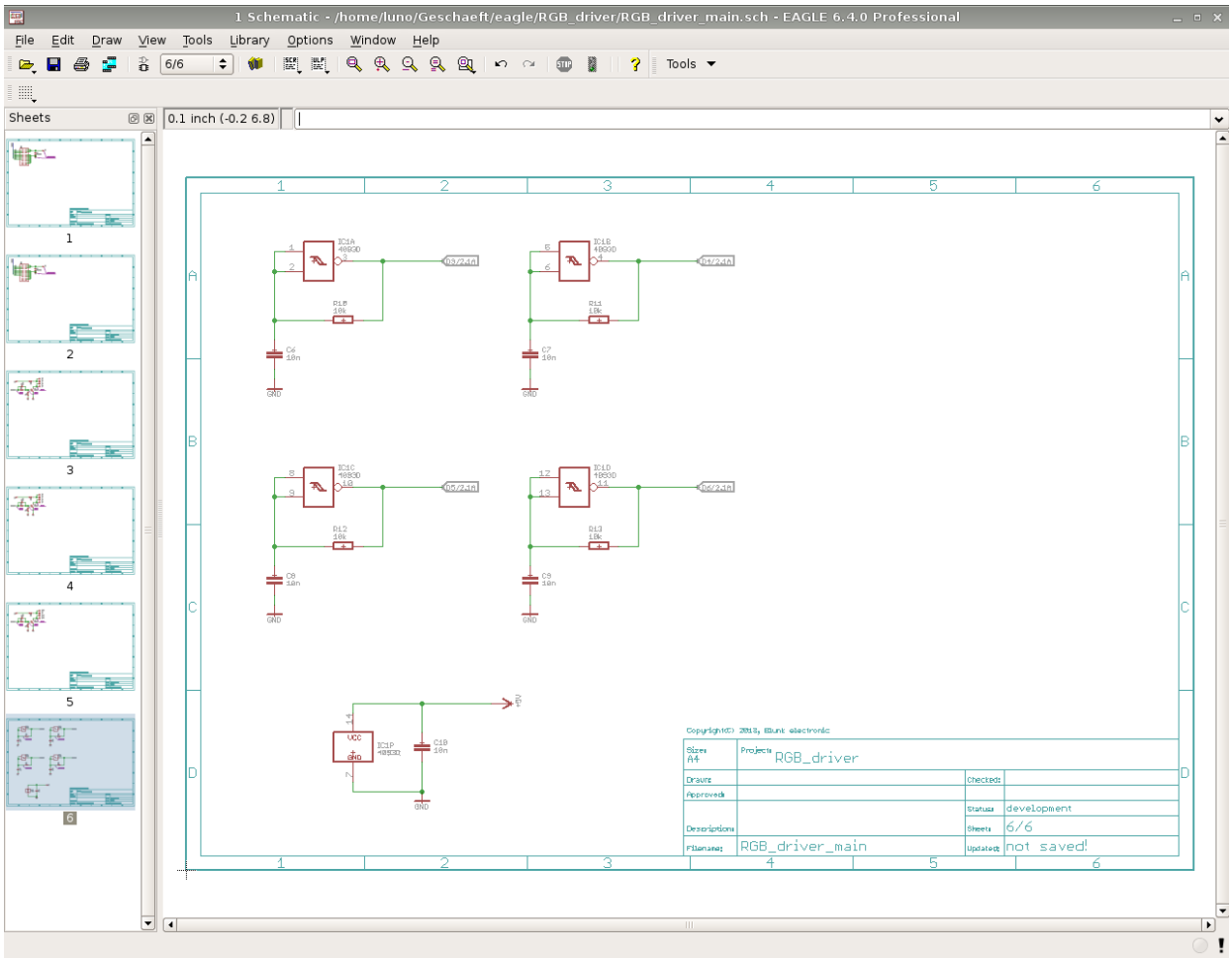


Figure 25: oscillator sub module imported

2.8 Finalizing

There may be lots of things to care for in the main module schematic now. They are mostly project specific and not of generic nature so that going into various details is not helpful here. Two general actions to mention here are:

1) Replacing “development” drawing frames by “final product” frames. Use the EAGLE command `replace` , select the new frame from the library and left click on the drawing origin (**red circle** in Figure 26). The color of the frame changes from turquoise to red in the schematic drawing. See final schematic in Figure 27.

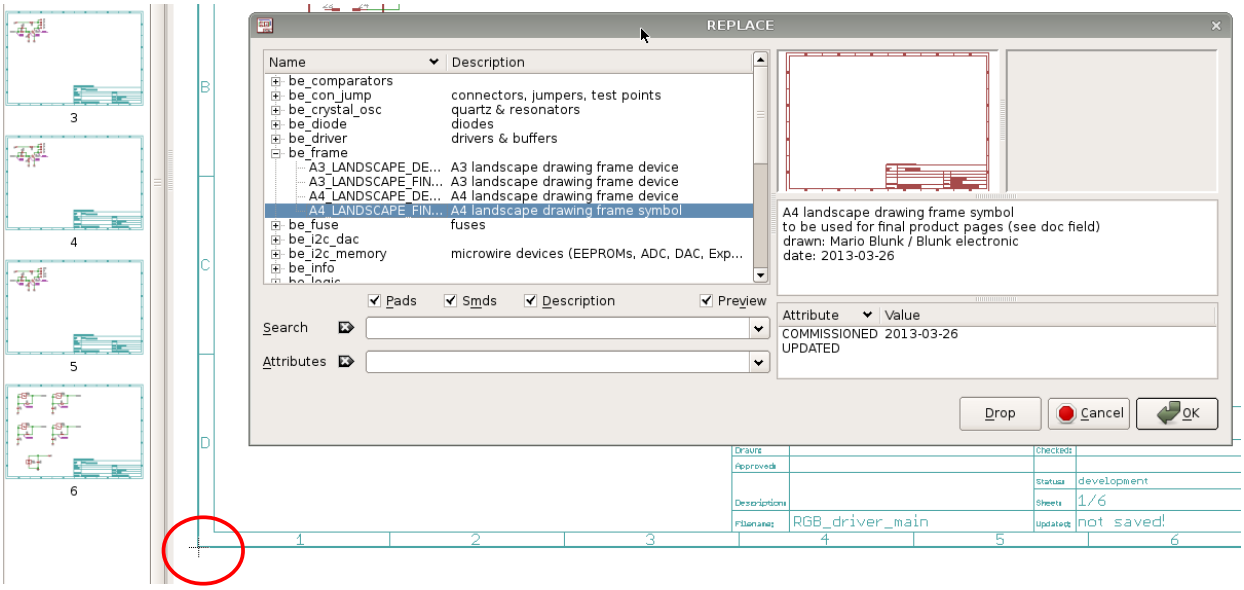


Figure 26: replace drawing frames

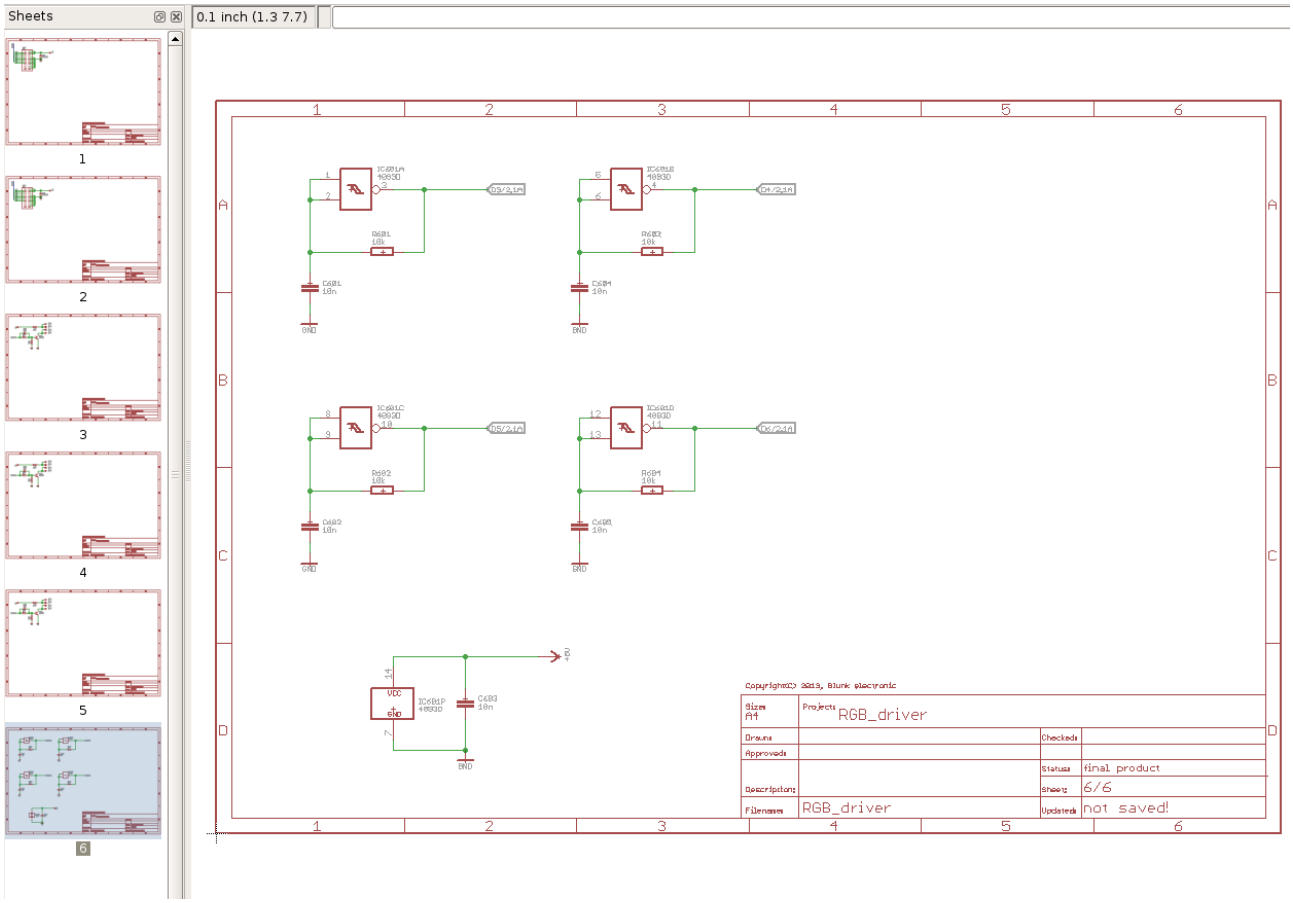


Figure 27: final main module schematic

2) Renumber devices using the ULP renumber-sheet.ulp.

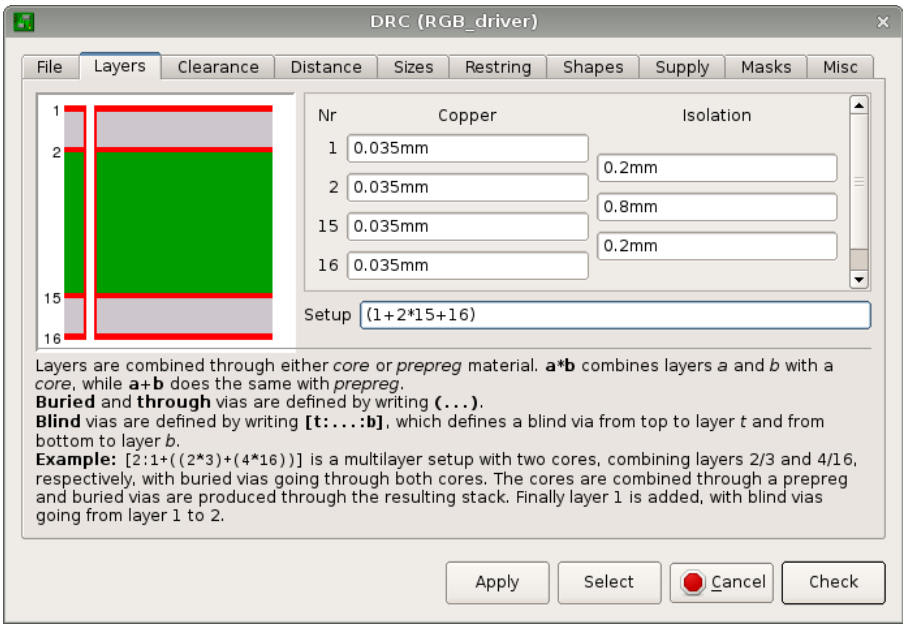
3 CASE #2 – Merging Boards

3.1 General Workflow

In contrast to case #1 where no board files existed, case #2 takes them into account too. There might be a partially routed main module, where further sub modules will be attached to. It is a question of viewpoint which module is the core module.

When merging board files (or layout designs) **all** layout imports are to carry out from inside the main module layout editor. This is the most convenient approach. Importing from inside the main module schematic is possible but less flexible.

1. Draw individual sub module schematics with development frames (Figure 29).
2. Care for net classes and net names according to project wide conventions.
3. Create individual sub module board files.
4. Care for layer setup, via types, manufacturing parameters, ... → all users must apply the same *.dru file (Figure 28).



5. Care for a common agreement regarding layer usage. The important question is: Which layers are reserved for supply polygons ?
6. Do the layout design (place & route) for each sub module. Leave off mechanical “devices” like mounting holes, marks and texts in copper layers as they will be placed in the main module layout later. If possible also delete board outlines. For documentation use a drawing frame. Later, after import into the main module, the sub module outlines and the drawing frame can be deleted as well.
7. Run DRC for each sub module.
8. Create a new and empty schematic and board file for main module.

9. Care for mechanical components of the main module board (outlines, mounting holes, texts in copper layers, restricted areas, …)
10. Optionally: Draw the schematic of the main module and do the layout for the main module. Then run DRC on this state of the main module.
11. Import from **inside the main module board layout** the individual sub module layouts and place them where required. Use a coarse grid if possible.
12. Route module interconnections.
13. Run DRC on the main module.
14. Do final adjustments, documentation, silk screen, ...
15. Replace schematic development frames by “final product” frames.
16. Renumber devices in main module (using ULP renumber-sheet.ulp).

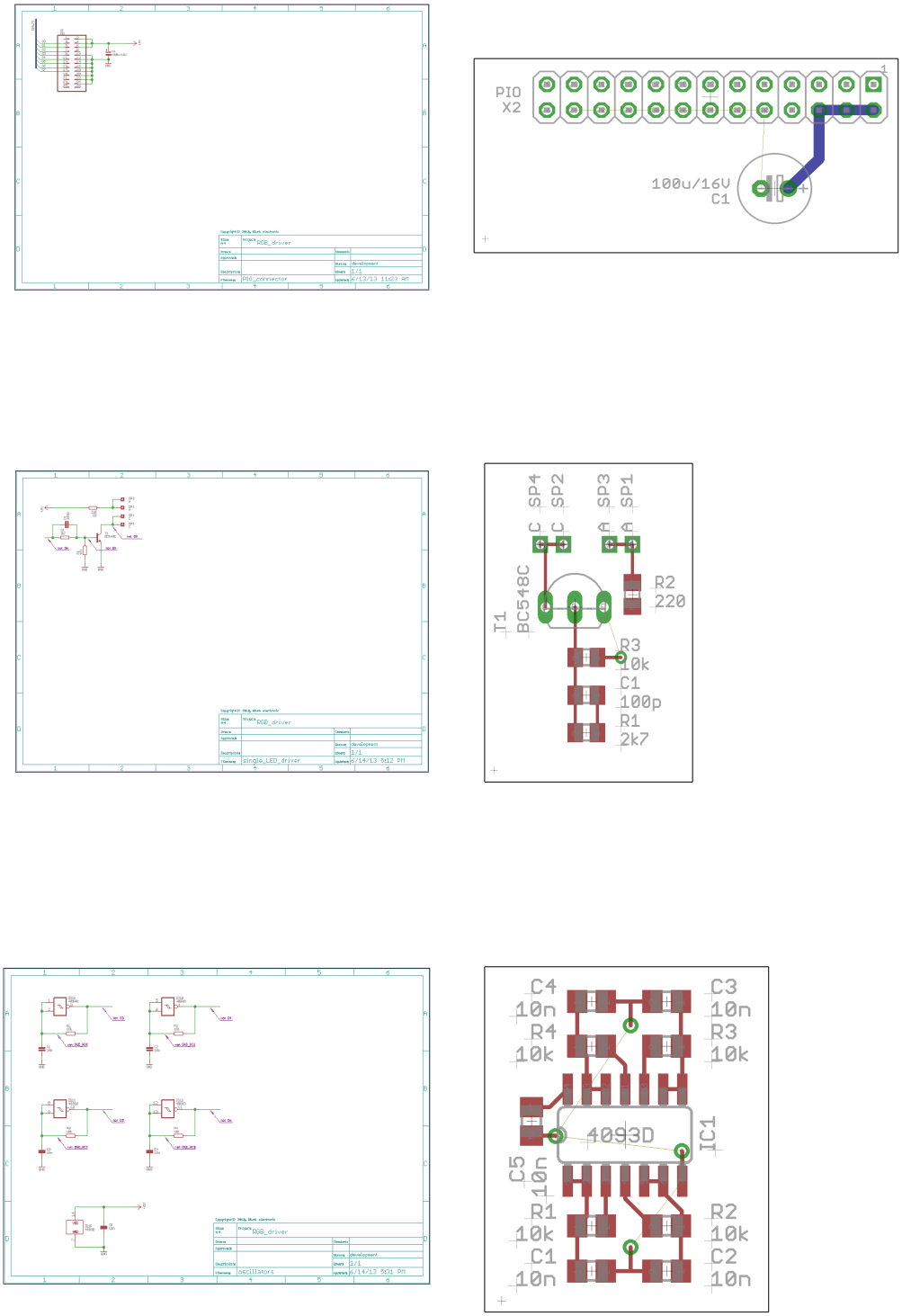


Figure 29: individual sub module schematics and their layouts

3.2 Main Module Overview

Figure 30 outlines the main module as it consists of sub modules and their interconnections.

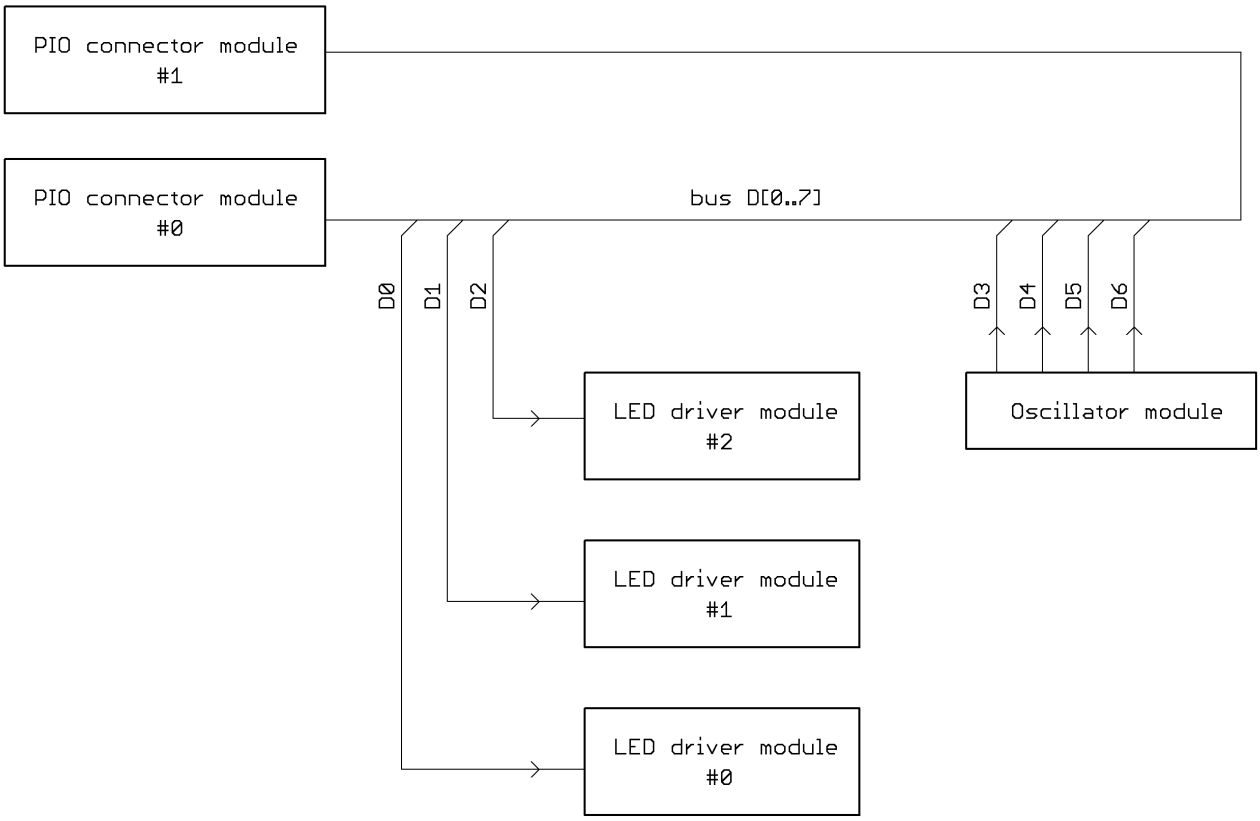


Figure 30: main module overview

3.3 Drawing Sub Modules

All the sub modules have to be designed independently of each other. They may reside:

a) inside a global EAGLE project directory (recommended)

b) elsewhere in various EAGLE project folders (without the engineers/subcontractors knowing the higher level project they are contributing to)

Regarding their schematics please refer to sections 2.3 page 9. The steps to carry out here are basically the same as in case #1 where only schematics are dealt with.

In addition to a sub module schematic, a board layout is to do for each sub module. In the following sections the board layout is addressed only.

3.3.1 Sub Module Layouts

Figure 31, 33 and 32 show the very simple sub modules “PIO_connector”, “single_LED_driver” and “oscillators” where place & route and silk screen are done.

Note:

– the drawing origin (**red circle**). The layout should be as close to the origin as possible.

– Even if there are no airwires (unrouted signals) visible for signals crossing the sub module boundaries, they are there in the background.[2]

– From inside the main module schematic and layout, go to `Edit|Global Attributes` and define placeholders for the project name and its value.

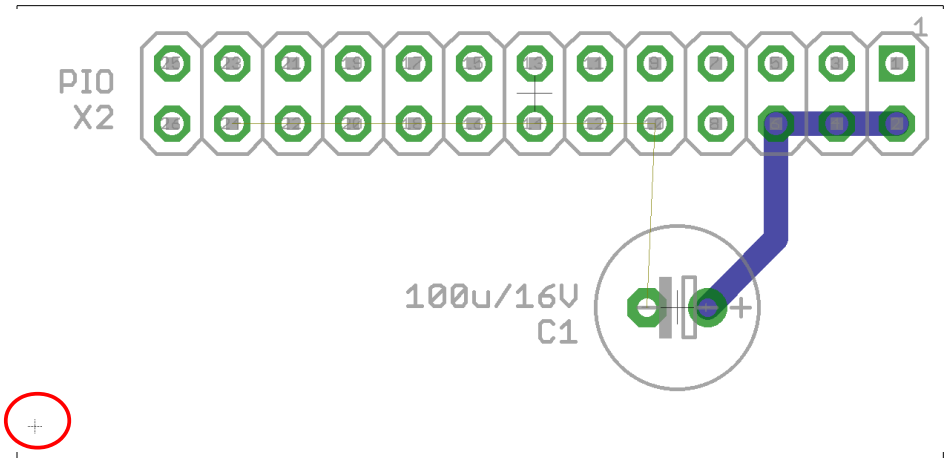
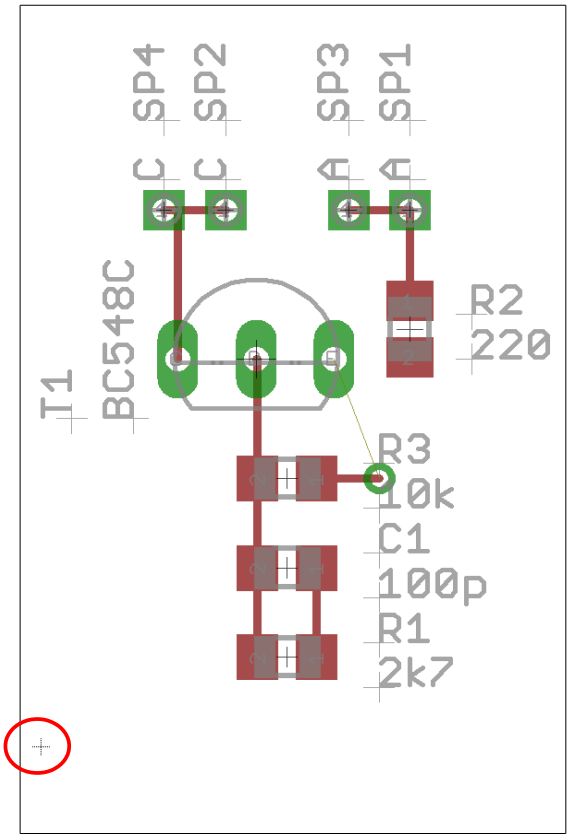
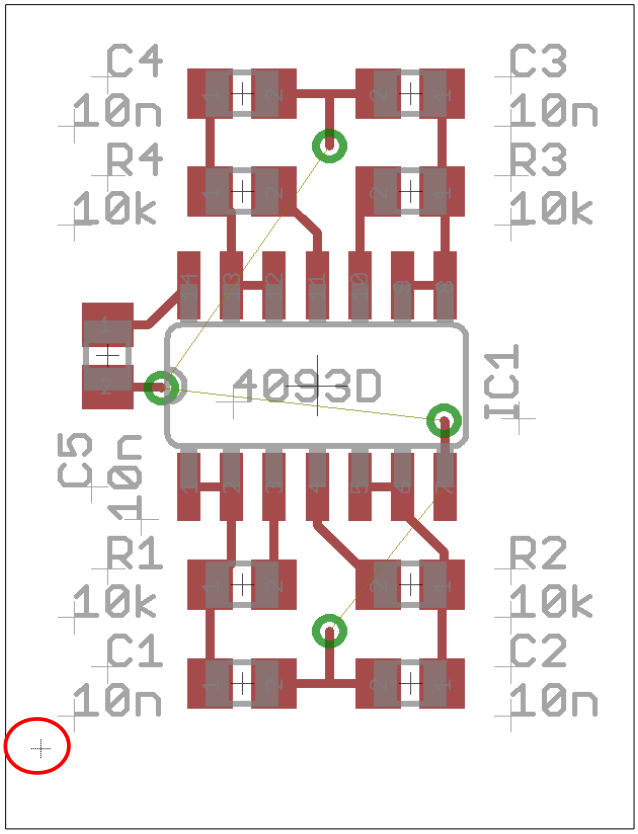


Figure 31: PIO_connector layout

2 In general, so called one-pin nets are never displayed in layer #19 (unrouted nets).


SP4
SP2
SP3
SP1
C
C
A
A
T1
BC548C
R2
220
R3
10k
C1
100p
R1
2k7


C4
10n
R4
10k
C3
10n
R3
10k
4093D
IC1
C5
10n
R1
10k
C1
10n
R2
10k
C2
10n

3.4 Set Up Main Module

Now that the sub modules are ready to use, the main module needs some attention:

1) If not there already, in the project directory “RGB_driver” a new and empty schematic named “RGB_driver_main” is to create (see **red** pointer in Figure 34).

2) Even if the schematic is empty at this stage, switch over to the layout editor so that a new and empty layout is created too. As shown in Figure 34 both the main module schematic and layout are now listed in the project tree of the EAGLE control panel.

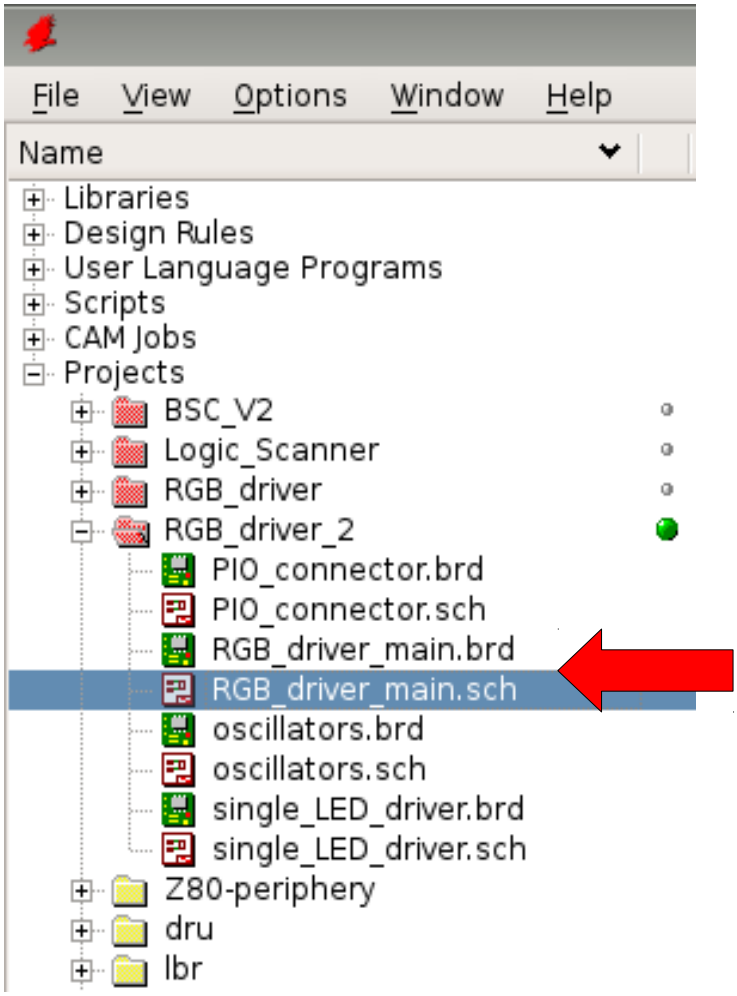
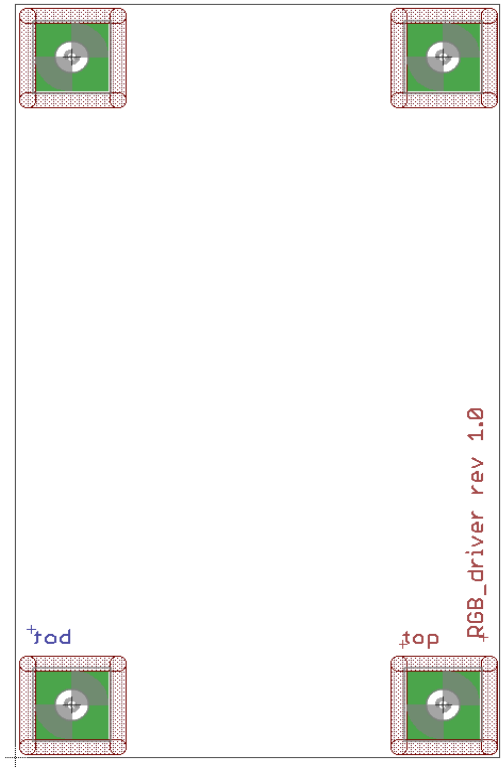


Figure 34: main module schematic

3) Optionally, certain essential core sections can be drawn now in the main module schematic. Net class and net name conventions must be met in order not to contend with net classes and names of the sub modules drawn already. In this tutorial we skip this step.

4) Care for mechanical components of the main module board (outlines, mounting holes, texts in copper layers, restricted areas, drawing frame, …) (Figure 35)



5) From inside the main module schematic and layout, go to `Edit|Global Attributes` and define the project name placeholder and its value.

3.5 Importing Sub Module Layouts

1) Change into the main module layout file, go to `File|Import|EAGLE drawing` and select the “PIO_connector” as the first sub module to be imported (Figure 36 and 37).

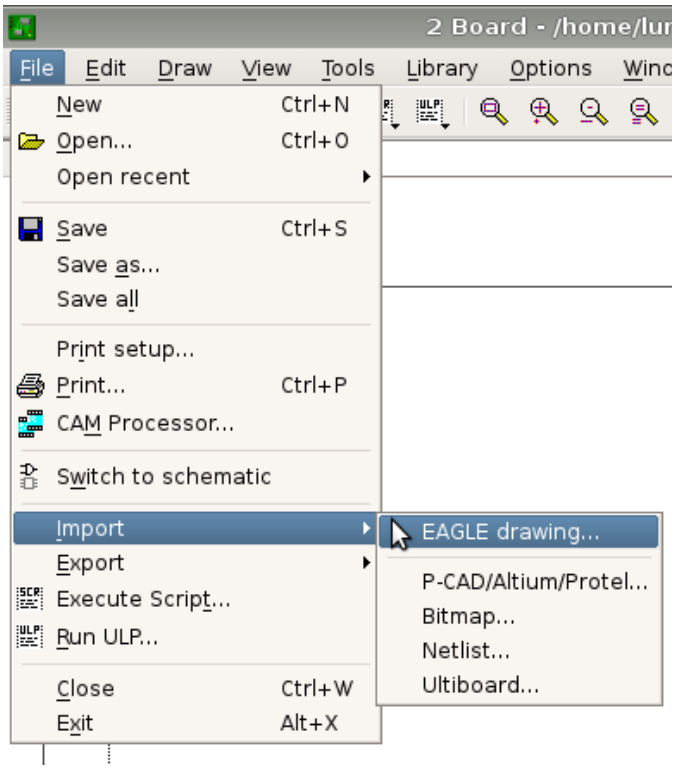


Figure 36: import EAGLE drawing from inside the layout editor

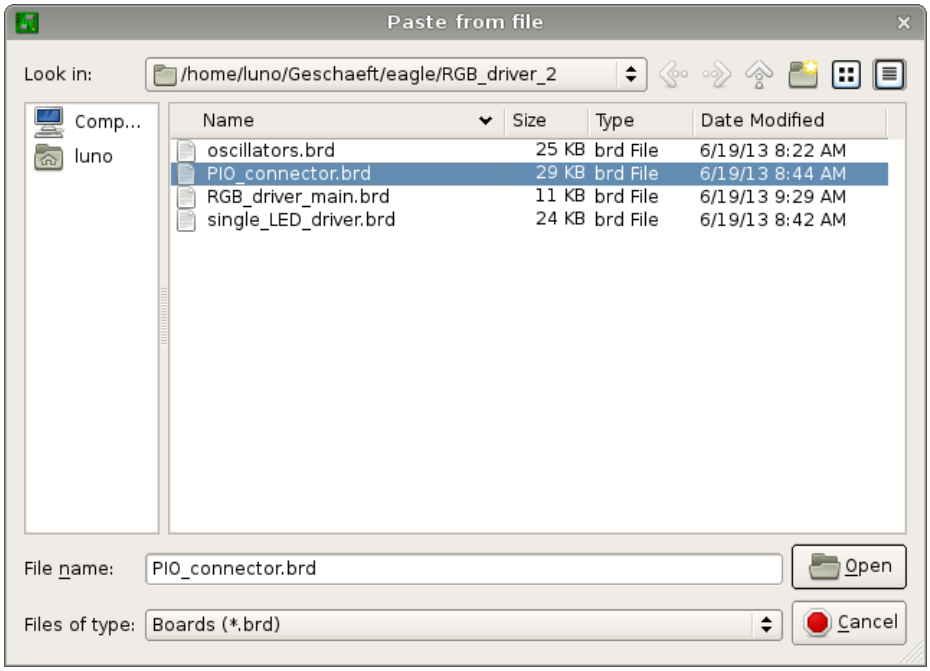
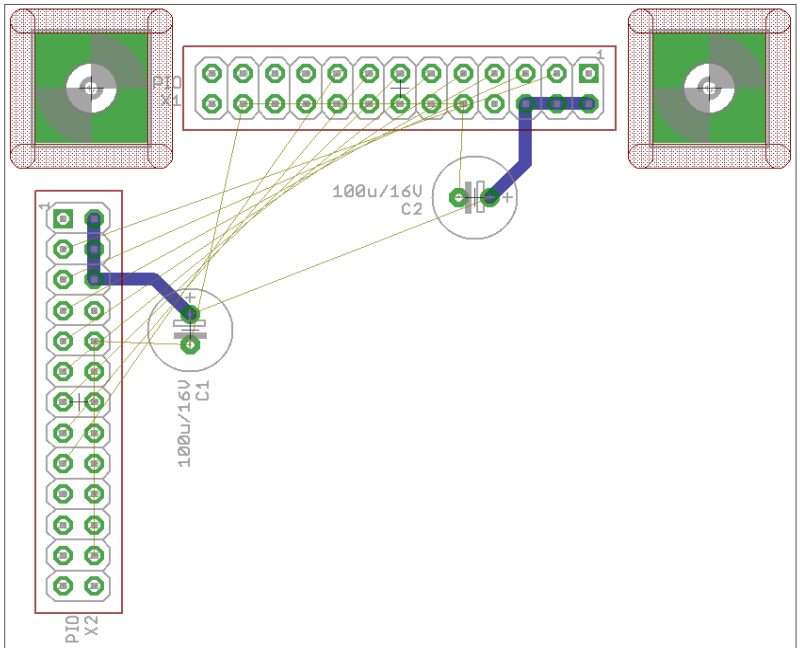


Figure 37: importing the PIO_connector sub module layout

2) The “PIO_connector” layout is now stuck to your cursor. So you are free to place it wherever you need it in your main module layout.

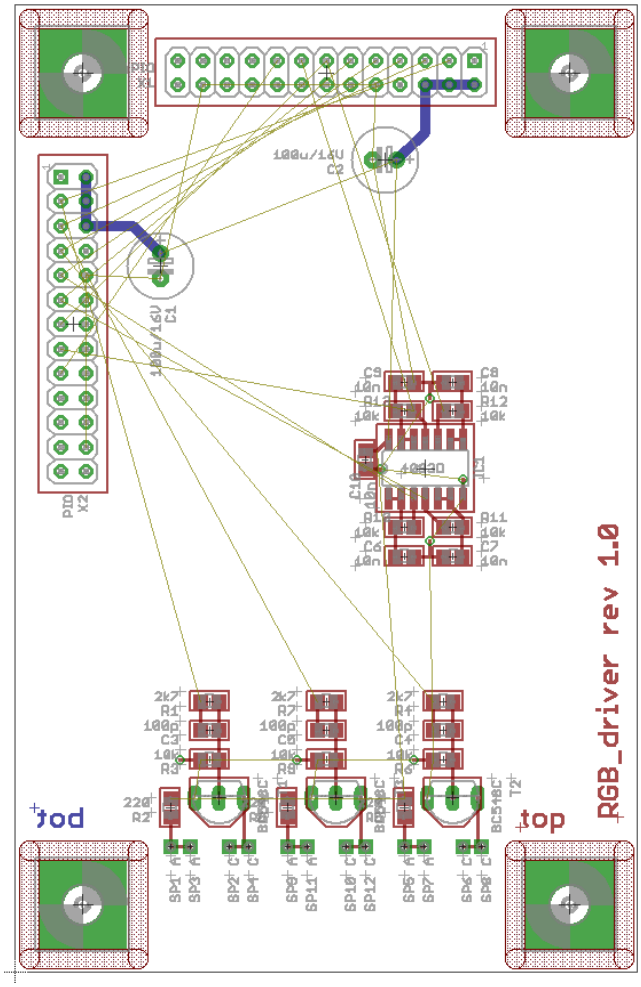
3) Now the net names window pops up notifying you about possible net name indexes or static net names while import. How to handle this window has been described in case #1 sections 2.5 through 2.7 (pages 16 and 23).[3]

4) Import the “PIO_connector” a second time. The result is shown in Figure 38. Note the airwires connecting the pinheaders.



5) Repeat the steps above for three “single_LED_drivers” and the “oscillators” sub module layout. See Figure 39.

3 Each time a layout is imported, a new schematic sheet is appended automatically to the main module schematic.


RGB_driver rev 1.0
top
top
PID
X2
X1
100u/16V
C1
100u/16V
C2
IC1
T1
T2
BC548C
BC548C
BC548C
2k7
R1
100p
C3
10k
R3
220
R2
2k7
R7
100p
C5
10k
R8
2k7
R4
100p
C4
10k
R6
R11
10k
R12
10k
C7
10n
C8
10n
C9
10n
C10
10n
SP1
SP3
SP2
SP4
SP9
SP11
SP10
SP12
SP5
SP7
SP6
SP8

3.6 Route Module Interconnections and Finalizing

Now that all sub module layouts have been imported an well placed, the nets interconnecting them need to be routed. Supply polygons may also be drawn according to the project wide layer usage conventions (see section 3.1 point 5 on page 27). After all, this is basic stuff, the skills required to carry out the actions here should be well known to the reader [3].

Additional copper layers may be introduced at this point if needed. In this case, the corresponding DRC settings should be saved in a different *.dru file than the global one used for the sub modules.

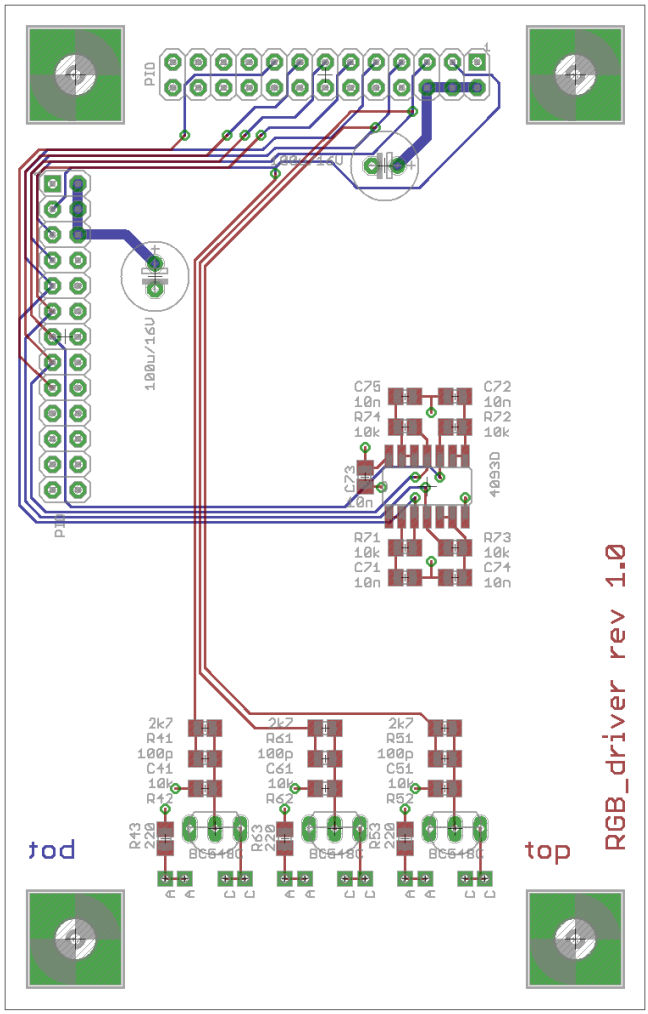
Run the ULP renumber-sheet.ulp on the schematic.

Replace schematic development frames by “final product” frames.

If necessary do a fine positioning of the device names in the layout (silk screen).

Run the DRC on the board.

To give an idea of the outcome see Figure 40.



4 References

1. Mario Blunk, "*Design Checklist"*
   at http://www.train-z.de/pdf/Design_Checklist_en.pdf
2. Mario Blunk, "*EAGLE Library Tutorial"* at http://www.train-z.de/pdf/library_tutorial.pdf
3. *CadSoft EAGLE* Version 6.x English users manual

5 Disclaimer

This paper is believed to be accurate and reliable. I do not assume responsibility for any errors which may appear in this document.

Mario Blunk, June 20th 2013

Blunk electronic / owner : Dipl. Ing. Mario Blunk / Buchfinkenweg 5 / 99097 Erfurt / Germany +49 176 2904 5855 / http://www.train-z.de