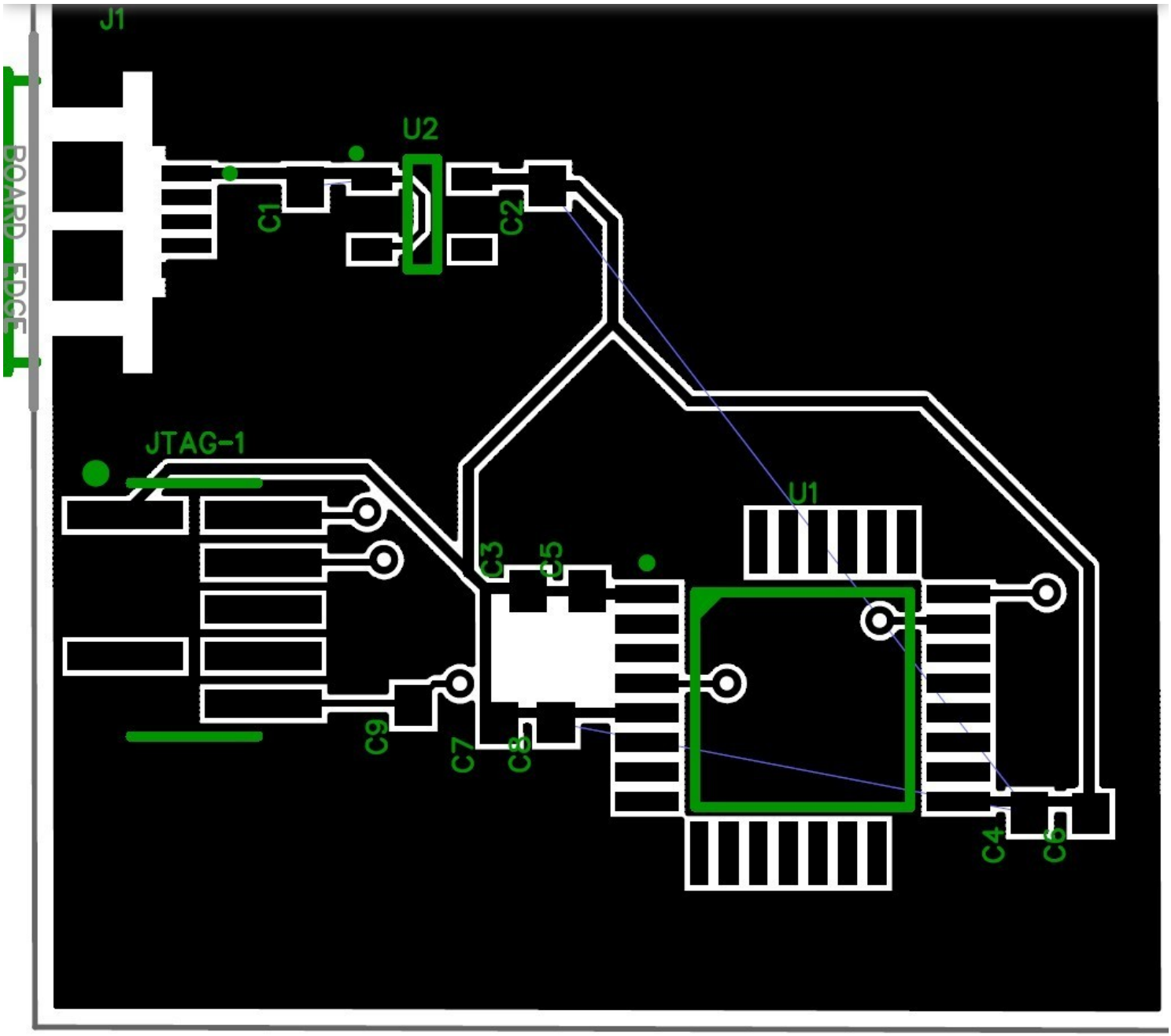
Tutorial: How to Design Your Own C
Microcontroller Board – I

26 Comments



Article Technical Rating: 7 out of 10

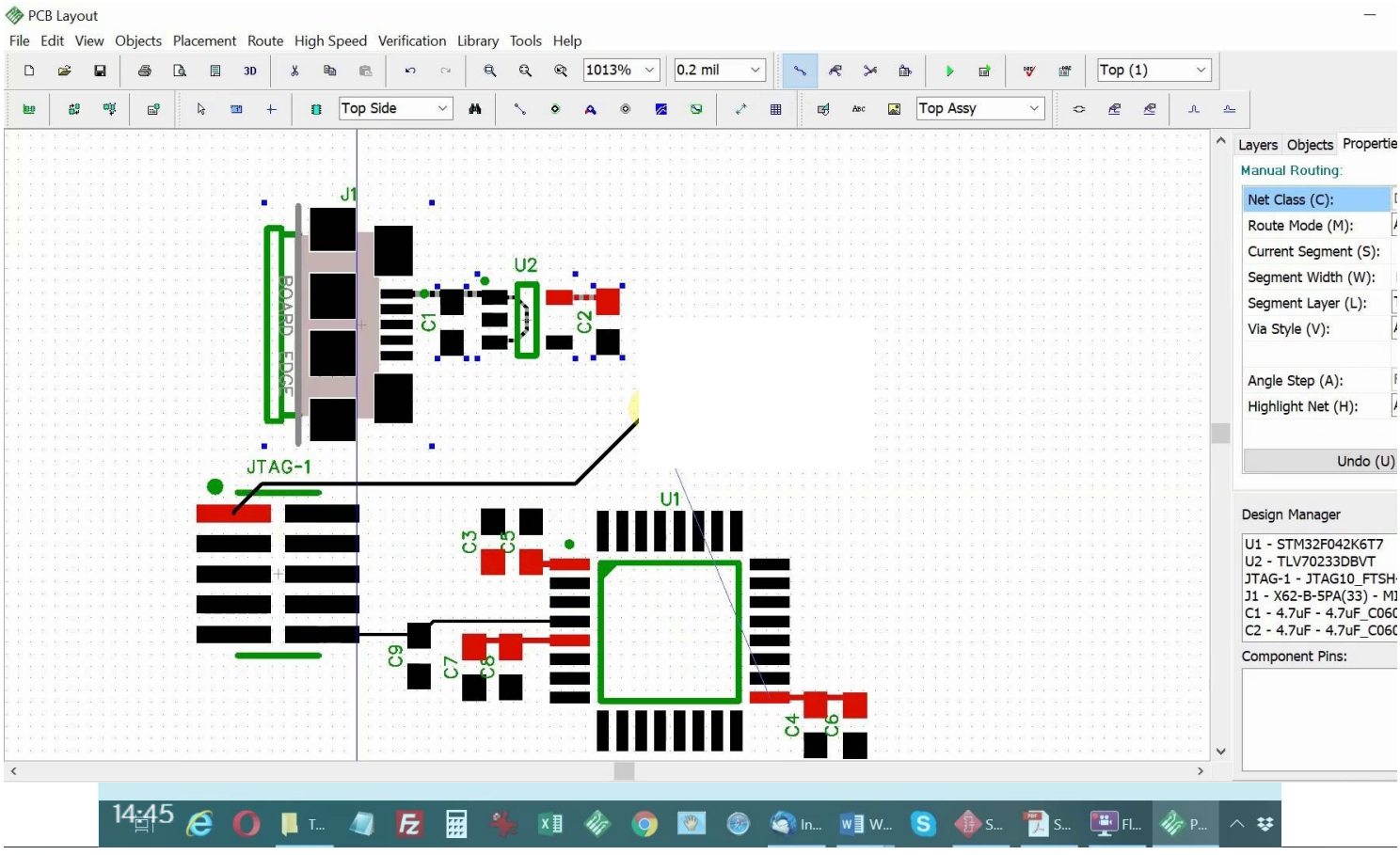
In this tutorial you're going to continue learning how to design your own custom 32-bi microcontroller board based on an Arm Cortex-M0 STM32 from ST Microelectronics.

In part 1 of this tutorial we covered how to design the system level block diagram, sele of the critical components, design the full schematic circuit diagram, and run schemati

In part 2 you're going to learn how to turn the schematic diagram into a real-world PCB layout which can then be sent to a manufacturer for prototyping or mass production.

Once the schematic design is completed, it's time to design the Printed Circuit Board. I by inserting all of the components into the PCB layout. In DipTrace, you can use the "C to PCB" function in the schematic to automatically create the PCB with all of the components inserted.

NOTE: This is a long, very detailed tutorial so here's a **free PDF version** of the full tut (part 1 and 2).



This program will provide you with regular in-depth training sessions (on both technic
entrepreneurial topics) with me and other experts. Plus you get direct access to me fo
questions and advice. To get early access and discounted pricing ***join the waiting list*** *to*

Component Placement

Although all of the components have been inserted, it's your job to determine exactly v
each component is placed on the PCB.

Most PCB design software packages include an auto-placement feature that places components with the goal of minimizing routing lengths. But I never use it, and it's alm
necessary to manually place the components in the best layout.

For our initial tutorial circuit, the component placement is pretty simple. Place the mic
connector next to the linear regulator with its output as close as possible to the input s
pins (VDD) on the microcontroller. Finally, place the programming connector anywher
is convenient.

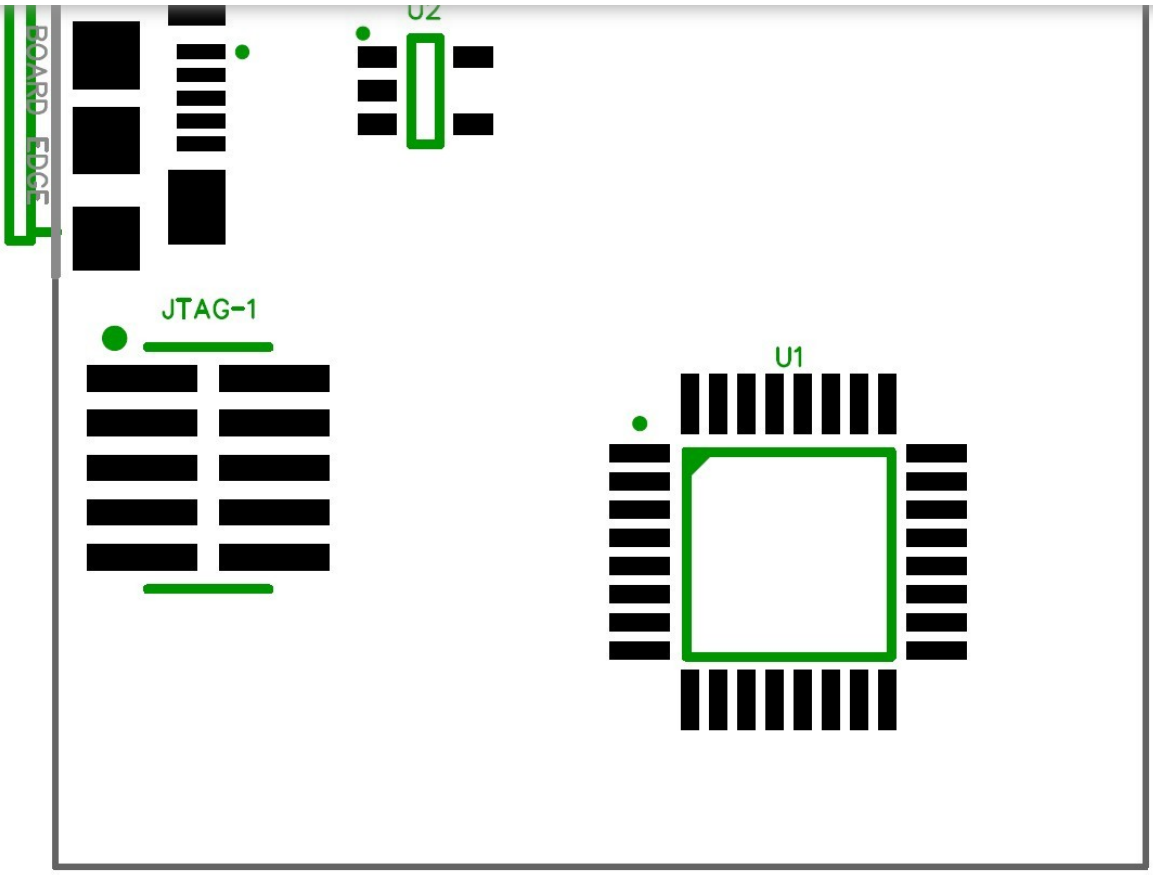


Figure 1 – Placement of the critical components in this initial design: the microcontr (U1), the regulator (U2), the micro USB connector (J1), and the programming conne (JTAG-1).

Once all of the core components are properly placed, your next step is placing all of the passive components (resistors, capacitors, and inductors). For this initial design the on passive components are capacitors.

One key aspect of designing electronics that you need to learn is the concept of parasi Parasitics are passive components (resistors, capacitors, and inductors) that you don't intentionally design into your circuit. But, nonetheless, they are there and impact performance.

trace length increases, and as the number of bends and vias increase.

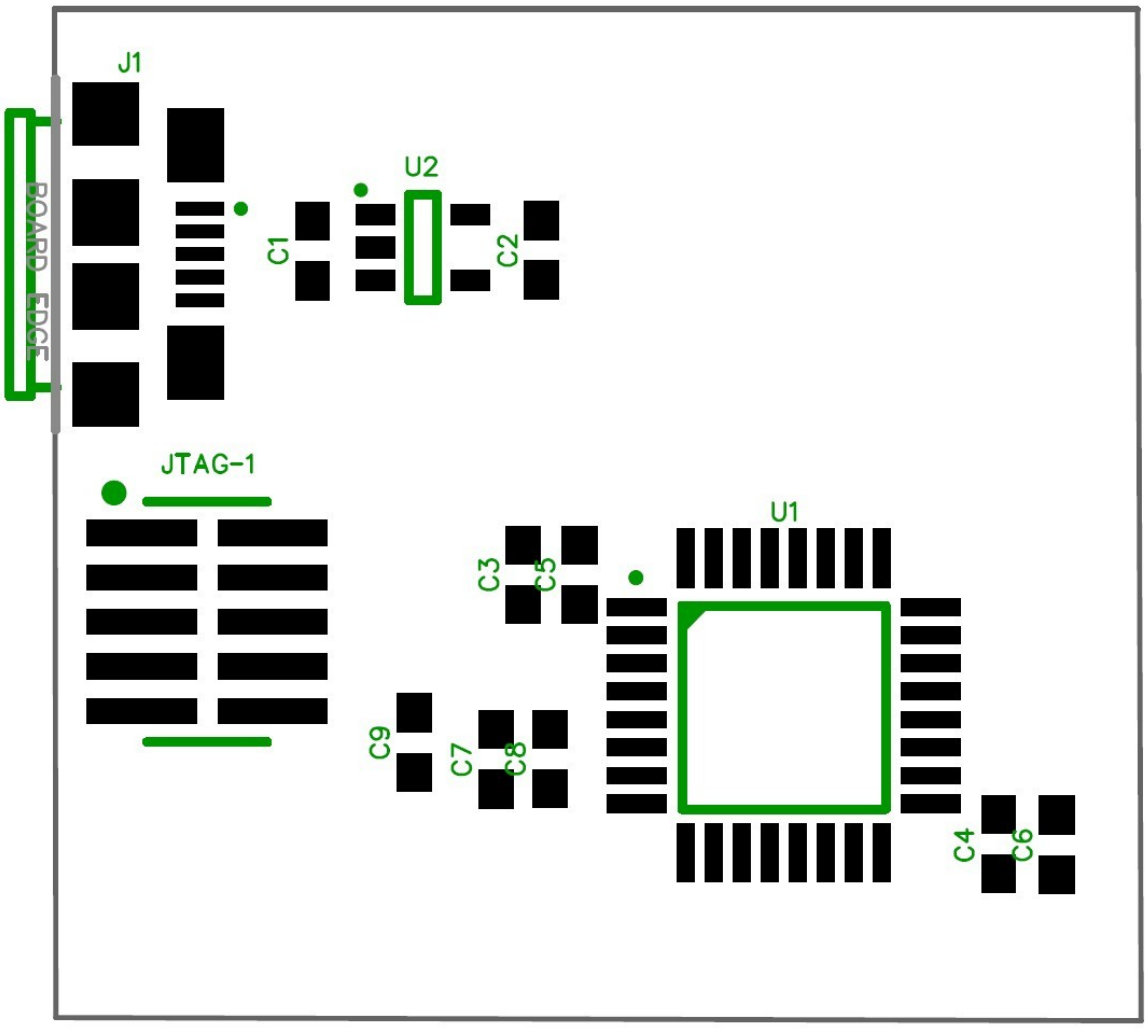


Figure 2 – Placement of all critical components (U1, U2, J1, and JTAG-1) and passi components (capacitors).

So this means that if a voltage source is located far away from the load, which is the ST microcontroller in this case, there is essentially a resistor between the load and the so (neglecting any capacitance and inductance).

So even though the voltage regulator's output may be a perfect 3.30V, the voltage at tl microcontroller pin will be lower during this current surge. Decoupling capacitors are to solve this problem.

Remember, capacitors are like little batteries that store electrical charge. Placing them at the microcontroller's supply pins allows them to supply any fast, transient current n of the microcontroller.

Once the transient load disappears the capacitors are recharged by the power supply they are ready for the next transient increase in load current.

PCB Layer Stack

A printed circuit board is made up of stacked layers. Conducting layers are separated b insulating layers. The minimum number of conducting layers is two. This means the top and the bottom layer can be used for routing signals, and these two layers are separate an internal insulating layer.

For this tutorial we'll start with a 2-layer board to keep things simple. But as the circuit complexity increases you'll find it necessary to add additional layers.

The number of conducting layers is always an even number, so you can have a board w 2,4,6,8,10,12 conducting layers. Most designs will require 4-6 layers, and more advanc designs may require 8 or more layers.

Routing

Once all of the components have been properly placed it's now time to perform the necessary routing. There are two options for routing: manual and automatic.

Unfortunately, auto-routers in general do a horrible job, and in almost all cases you wil
to manually do all of the routing. For this tutorial we will be doing all of the routing ma

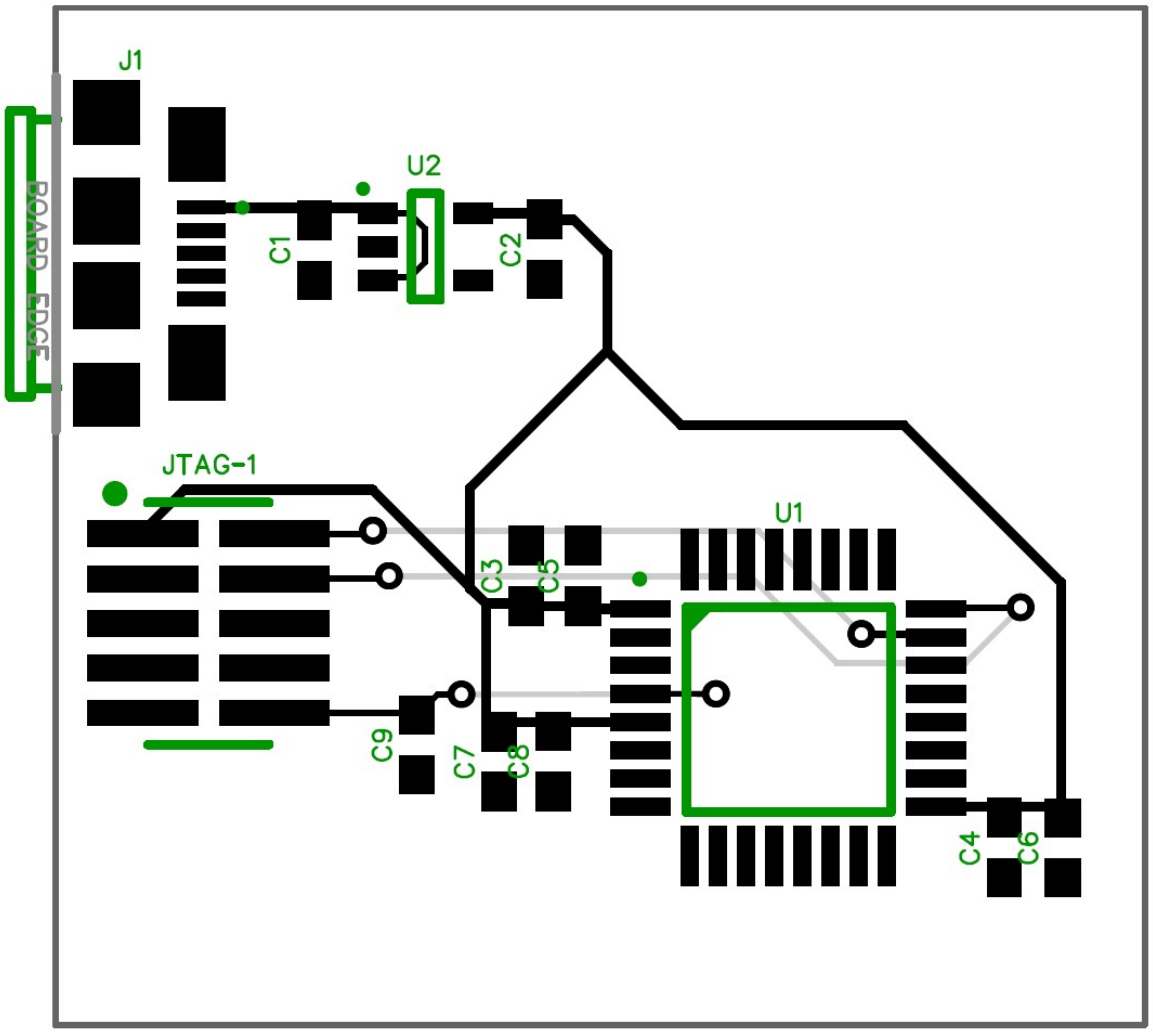


Figure 3 – Routing the PCB (black traces on the top layer, gray traces on the bottom

When routing on a PCB you want to minimize the length of each trace as much as poss
You also want to minimize the number of vias and avoid any 90 degree bends in the tra

two traces on different layers. Most vias are what are known as *through vias* which me
the via tunnels through all layers of the board.

Through vias are the simplest type to manufacture because they can be drilled after th
entire PCB layer stack-up is assembled.

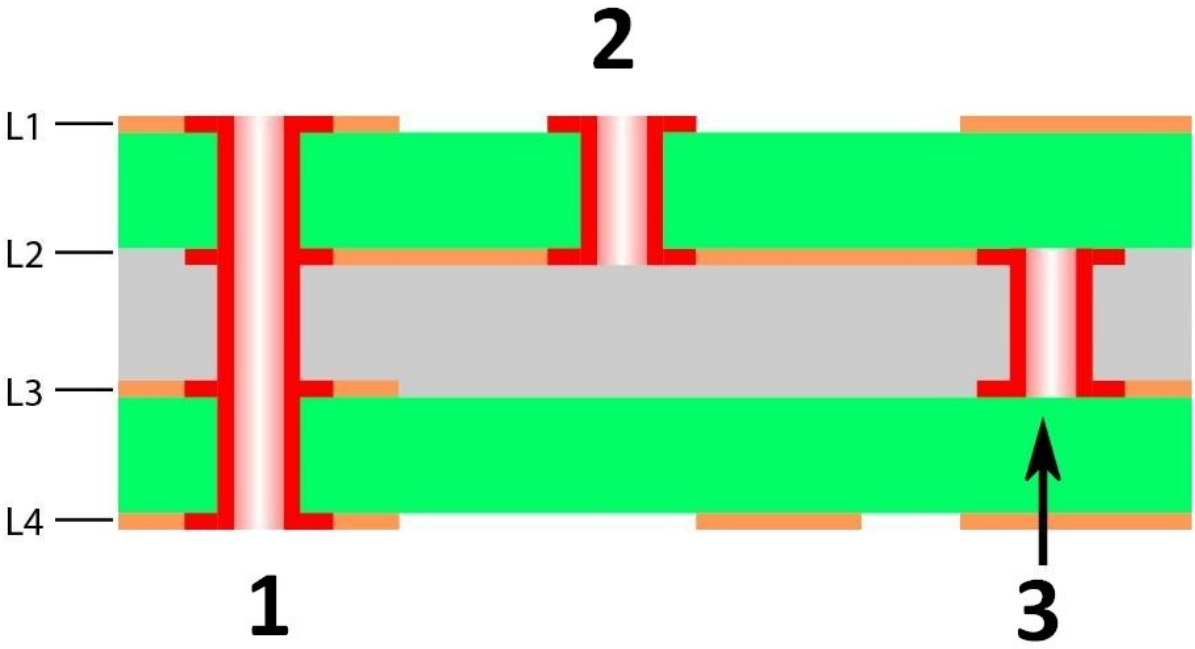


*Figure 4 – Via #1 is a classic **through via**, via #2 is a **blind via**, and via #3 is a **buried***

**Vias that only tunnel through a subset of layers are known as buried and blind vias. Bli
vias connect an external layer to an internal layer (so one end is hidden inside the PCB
up). Buried vias connect two internal layers and are completely hidden on the assembl
PCB.**

layers.

However, be aware that blind and buried vias drastically increase the prototype cost fo your board. In most situations you should restrict yourself to only using through vias. C exceptionally complex designs, that must fit in an exceptionally small space, should like ever require these more advanced via types.

When routing any high current power lines you need to ensure the trace width is capa carrying the necessary current. If you run too much current through a PCB trace it will overheat and melt causing the board to become defective.

To determine the necessary trace width I like to use a PCB trace width calculator. To determine the required trace width you need to first know the trace thickness for your specific PCB process.

PCB manufacturers allow you to select various conducting layer thicknesses, usually measured in ounces per square foot (oz/ft^2) but also measured in mils (a mil is one thousandth of an inch) or millimeters.

A common conducting layer thickness is 1 oz/ft^2. In this tutorial I've made the power s lines 10 mils wide. Using the calculator linked to above shows that a 1 oz/ft^2 trace mea 10 mils wide can actually carry almost 900mA of current.

This is obviously much more than we'll need, and I could have easily made the supply li much more narrow. In the first tutorial I showed that the absolute maximum current required by the STM32F042 is 120mA. Perhaps surprisingly, to handle 120mA we only a trace width of 0.635 mils!

less the resistance and the more stable the supply voltage at each component.

Unless space is extremely tight you should always over design the power supply traces fact, in many cases you'll want the power supply routing on its own layer so you can maximize the routing width.

Finally, in the calculator you'll notice the requirements are different for internal layers versus external layers. For this simple, 2-layer design both layers are external so the "*for External Layers in Air*" is what we need to use.

Internal layers can carry much less current because they don't get the cooling effect of exposed to air so the traces will overheat with much less current.

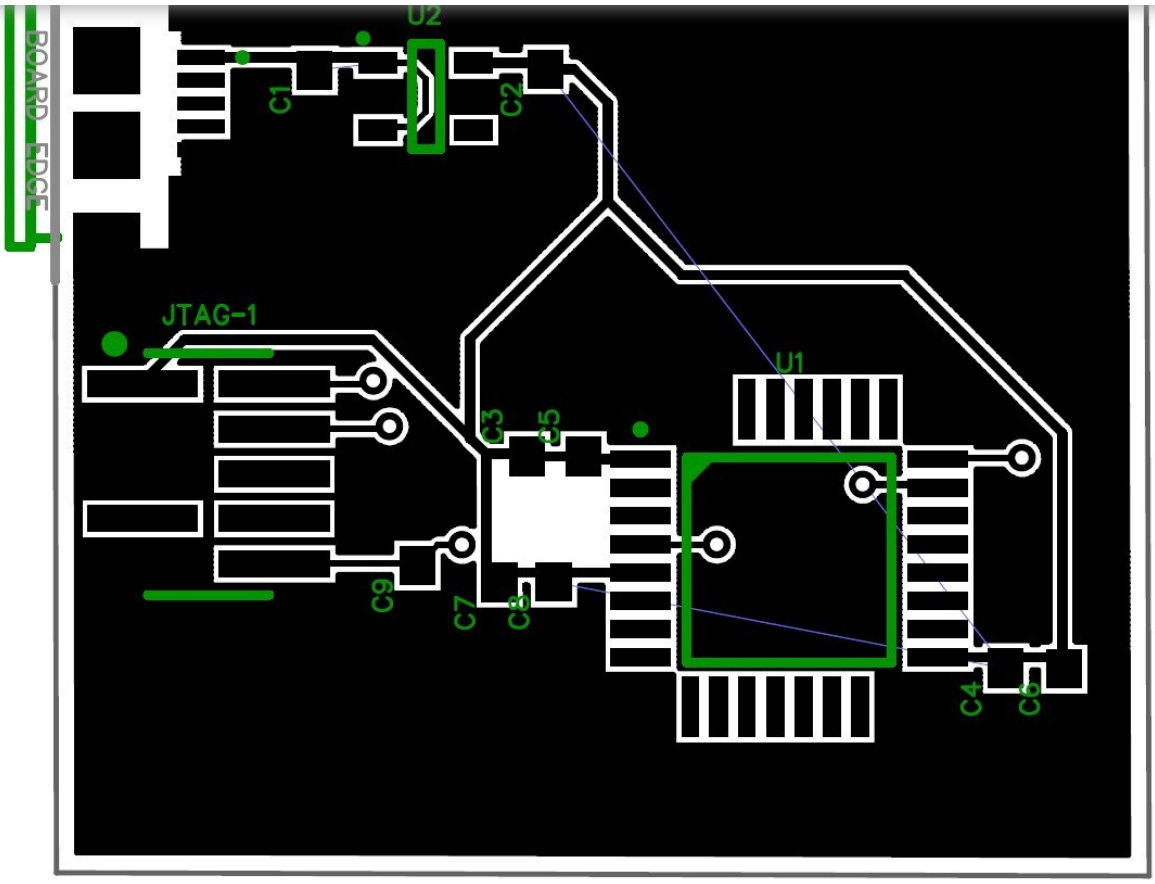


Figure 5 – The completed Printed Circuit Board (PCB) layout for this initial tutori

Verification

Once all of the routing is done it's time to perform verifications to ensure everything is correct. This is where automation really works well and any PCB design tool will offer automatic verification features.

There are broadly two types of verification: design rules check (DRC) and schematic comparison.

minimum spacing between a trace and the board edge, and so on.

In order to run a DRC verification its necessary that you first obtain all of the design ru the specific PCB process you will be using.

Each PCB prototyping process has slightly different rules so you must have the correc before proceeding. You can get the design rules for your specific process from your PC prototype supplier.

In DipTrace, you define the design rules by selecting *Verification->Design Rules*. Once rules have been correctly defined, you can run the DRC by selecting *Verification->Che Design Rules.*

After you have verified that your PCB layout adheres to all of the process design rules, now time to verify that your PCB design matches your schematic diagram. To do this ir DipTrace you simply select *Verification->Compare to Schematic*.

In future tutorials, I'll show you various types of DRC and schematic comparison error how to fix them.

Generating Gerbers

Once you have verified the design adheres to the process design rules and matches the schematic diagram, it's time to order your PCB prototypes.

In order to do this you'll need to convert your PCB layout design (which is currently sto a proprietary file format) into the industry standard file format known as Gerber.

include:

1) Silk layers – Includes the text and component designators.

2) Assembly layers – Similar to silk layers but with specific assembly instructions.

3) Solder mask layers – Indicates the green stuff on a PCB that covers up any conductc that you don't want to solder to. This prevents accidental shorts during soldering.

4) Solder paste layers – Used to precisely place solder paste where soldering will occu

You will also need to generate what is known as a *Pick-and-Place* file which includes th coordinates and orientation for all of the components. This file is used by the manufac automatic component placement machines.

Finally, you need to output a drill file that provides the exact location and size of any h such as vias and mounting holes.

Once you have the Gerbers, the Pick-and-Place file, and the drill file you can send thos to any prototype shop or manufacturer for production of your board.

Summary

In this tutorial you've learned how to design a system-level block diagram, select all of critical components, design the full schematic circuit diagram, design the Printed Circu Board (PCB) layout, and order prototypes of your completed microcontroller PCB des

This tutorial has purposefully kept the circuit itself rather simple so as to not overwhe with circuit complexity. That being said, a microcontroller without any additional

including battery charging, a display, Bluetooth, WiFi, GPS, USB, and various sensors.

Do you need *affordable* coaching, training, and support to bring your new electronic hardware product to market? If so, join the waiting list for the Hardware Academy and early access with discounted pricing.

26 Leave a Comment

b i link b-quote u ul ol li code spoiler

Join the discussion...

16 10 1 16

Subscribe newest oldest mo

Arie de Muijnck

Guest

KiCad is a great free tool (although not so easy for a beginner).
Free review notes: Do not create acid traps (copper corners far below 90 degrees). I see two. Use thermal
for component pads in the ground plane, this makes (de)soldering so much easier.

+ 0 − Reply

John Teel

Guest

Qandeel Sheikh

I am waiting for the next part of the tutorial. Please let me know if you have posted the tutorial already but missing it?

+ 0 − Reply

Author

John Teel

It will be coming soon and you haven't missed it. Be sure to join my email list so you don't miss it. Than the comment.

John

+ 0 − Reply

Guest

Richard Salazar

Great tutorial! I can't wait to see your succeeding tutorials and learn a lot more in terms of using the STM micro controllers. Any idea as to when these might come out?

+ 0 − Reply

Guest

Shine

Thank you for this article. I look forward to your next articles in this series.
Is there a free opensource equivalent to DipTrace?

+ 0 − Reply

Author

John Teel

Thank you Shine. I'm not familiar with an opensource equivalent to Diptrace. There are several free d tools available but they restrict you to only one PCB vendor. Diptrace does have a free version availal smaller designs.

+ 0 − Reply

Guest

Guest

Pachito Marco Calabrese

Really good set of articles, thank you

+ 0 − Reply

Guest

Qandeel Sheikh

Nice article !

+ 0 − Reply

Guest

Stevie

Sir Teel I know you love the stm32 microcontroller but I am wondering if in future you could do a tutorial o microcontroller (more specific ATmega 48) just like the one you are presently doing on the stm32...

+ 0 − Reply

Author

John Teel

Hi Stevie,

Yes, definitely! The STM32 isn't always to best solution for every project. Since the AVR microcontrol so popular with makers I think a tutorial on the microcontroller is a great idea. Thanks for the recommendation.

John

+ 0 − Reply

Guest

Mbavhalelo Neron Maano

Reviewing datasheets is absolutely a critical skill to learn for every Electronic Designer as it may not be po to successfully design for microchips without understanding your datasheets. So, that'll be much appreciat look forward to your series!

Guest

I'd like to say that I really appreciate this series of tutorials. What makes me appreciate them the most is th that you'll be adding advanced features such as a rechargeable battery, a display, Bluetooth, WiFi, USB dat and an accelerometer because it is true that a MCU board without any additional functionality isn't really useful. I'd like to see how the MCU board connects with a rechargeable battery, a display, Bluetooth, WiFi, data, GPS, and an accelerometer.

I'm starting to think that this series will help my upcoming projects become a reality and I'll have you to tha that. So once again, thank you very much for this series! I look forward to your future tutorials of this serie

+ 0 − Reply

Author

John Teel

Thank you Mbavhalelo! That is really awesome to hear and I'm happy you are finding it helpful. I will b creating add-on tutorials soon that will expand the capabilities of the circuit like you mention.

I'm also planning a series of tutorials where I review various component datasheets. When designing electronics you will spend a lot of time reviewing datasheets so I felt this is a critical skill to learn.

Thanks again for the great feedback!

John

+ 0 − Reply

Guest

Mark Jones

Great stuff! Keep up the good work.

+ 0 − Reply

Author

John Teel

Thanks Mark!

+ 0 − Reply

Guest

abba

thanks alot it cleared most of my doubt about mcu prog.
can we use arduino language to program the chip?

Guest

Learn c for microcontrollers. Read app notes from st micro if that will be your mcu.

+ 0 − Reply

Abba

Guest

thanks alot. it was very useful to mcu starters like me. which assembler is suitable for STM and cortex-m ty mcu?

+ 0 − Reply

Jonathan Nesbitt

Guest

Goto http://www.arm.com for all about cortex M series programming and stmicro site for an assembl most use c.

+ 0 − Reply

Santosh Sonar

Guest

Thanks, John. I am deliberately waiting for future mail from predictable design.

+ 0 − Reply

Andreas

Guest

Thank you for this article. I wish I had stumbled upon your website much earlier in my startup process. I lo forward to your next articles in this series!

+ 0 − Reply

John Teel

Author

Thank you Andreas! I wish you had as well, but it's never too late to learn more.

+ 0 − Reply

Guest

+ 0 − Reply

John Teel

Author

Thank you Nicholas! I'm excited that you learned a lot.

+ 0 − Reply

←PREVIOUS POST

NEXT POST

Predictable Designs LLC, 10645 N. Oracle Rd, Suite 121-117, Oro Valley, AZ 85737 USA
Info@PredictableDesigns.com, Phone: (520) 261-1844 (for fastest repsponse please email instead of call)