Pocket Tutor

Build a handheld version Of the Morse Code Tutor

Part 5: PCB Design

Bruce E. Hall, W8BH

As I kid in love with Ham Radio, I always wanted to make my own circuit boards. One day I walked into Radio Shack, bought a bottle of Ferric Chloride solution and resist pen, borrowed my mother's Pyrex baking dish, and diligently worked on my first printed circuit board masterpiece.

Well, my masterpiece didn't turn out very well. It was under-etched. So was my second attempt. Eventually I made ones that were over-etched. Traces disappeared no matter how dark I inked the lines. I couldn't get it right. It was messy and imprecise. I gave up.

Nowadays, we have free PCB creation software, and access to PCB fabrication houses, that were unimaginable in my youth. Making your own PCB has become easy, precise, and affordable. Several years ago, I decided to give it another shot. If you haven't tried it yet, you should.

PCB software.

For the enterprising ham there are several excellent (and free!) applications to choose from. The grand-daddy of them all is Autodesk EAGLE. The full version costs $248/year, but there is a limited version that is free for personal use.

Next is KiCad, my personal favorite. Fully open source. If you are starting from scratch and want to learn about creating PCBs, Chris Gammell from Contextual Electronics created a marvelous KiCad tutorial series on YouTube, "Getting to Blinky". If Chris' fast-paced 2 hour course is not your style, John's Basement 32-part KiCad series may be of interest, especially if you use Linux.

Diptrace and ExpressPCB are also popular with hams. Finally, EasyEDA provides integration with parts distributor LCSC, simplifying component selection and pricing. These are just a few of many choices available to you. A nice EDA comparison table is found here. Whichever software you choose, stick with it and discover its quirks and intricacies. Give yourself time to learn and experiment.

PCB layout.

Whichever EDA software you choose, the basic process is the same: create a schematic, choose parts, arrange the parts on the board, and connect the parts together according to the schematic. By now you have seen many schematics; these are familiar and non-threatening. But for me, laying out my first PCB was altogether different and formidable. The KiCad PCB editor dumps a tightly-clumped wad of the virtual parts on your screen and waits for you to do something with it. But what? How do you start?

Everyone has a different approach. Here are some basic suggestions.

1. Spread out your components. They aren't as intimidating when they are wide apart!
2. Create a temporary outline of your board, free of any components. Start by setting the design grid with reasonable spacing (say, 1 mm), and draw a square or rectangle of the approximate size. Remember my Casio FX calculator example? This outline is just temporary.
3. Narrow the grid for component spacing. Setting your component grid at 25 or 50 mils will help keep standard components and traces in good alignment.
4. With schematic in hand, start to group your components by function. In this project, that means group the power components, then the microcontroller components, then the audio components, etc. When you finish, your screen should have a blank board outline and component groups around it. Take a break from your work.
5. On your break, think about your interface components (jacks, headers, user controls). Where should they be placed? You probably don't want interfaces in the center of the board, and you probably don't want them on all four sides, either. Are there certain switches or jacks that you want in the front, or in the back? How far apart should they be? Leave room for knobs, for example.
6. Work on one component group at a time, remembering what you decided about the interfaces. If there is a central component, start with that. For example, in the audio section, I started with the PAM8302AAD footprint and arranged components around it, one at a time, until I had arranged the entire section.
7. Once a section has been reasonably arranged, grab it and move it into your board outline. If the section has an interface component, like a volume control, move it where you want the interface component to be.
8. Do steps 6-7 with each section, until most of your components are within the board outline.
9. Does your board outline accommodate everything? Either rearrange to fit, or consider a new board outline.

The suggestions above cover parts-placement and skip the tricky detail of routing traces between the parts. I have a few, general suggestions about routing:

1. Route the important signal lines first. For simple projects you will be able to route all traces on the front of the board. More complicated projects will require you to route signals on both sides.
2. After the signal lines are routed, work on power distribution. I do a copper pour for my positive supply on the front face, and a copper pour for GND on the back face.
3. Inspect your copper pours. Power is easier to route if you arrange your outside components such that you have a continuous band of copper along your board outline. Components at the

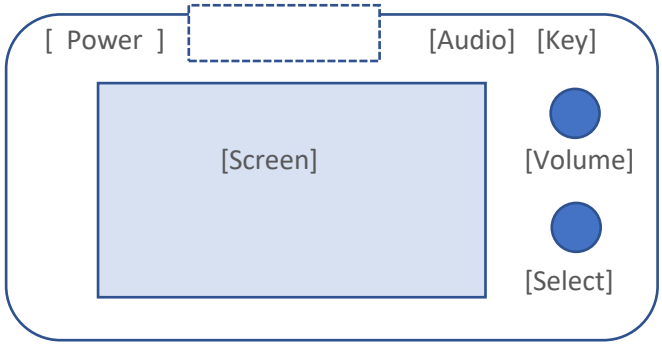
very edge of the board can interrupt the power-carrying pour. The pours for Pocket Tutor have continuous edges.

David Jones, of EEVBlog fame, is a professional PCB designer. I recommend his 25-page PCB Design Tutorial for a more thorough discussion.

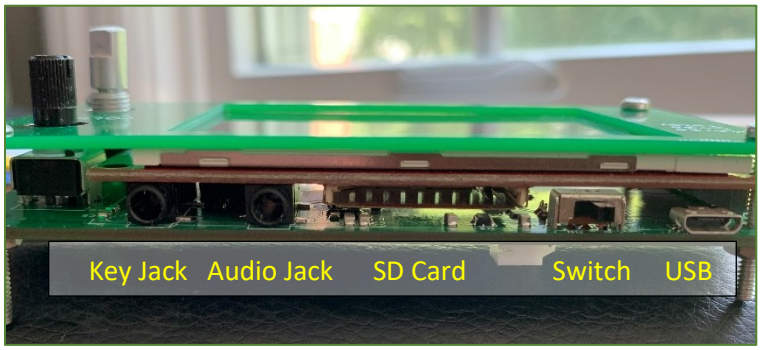
In this project, I chose to put my power connector, switch, and audio jack on the top edge of the board. I avoided using the left or right edges, since the landscape-oriented screen means that you will hold the device on the sides. On the first PCB layout I put the key jack on the bottom edge, thinking that they key jack should face the user. But I realized that bottom edge could not be used if the device was put on a table or desk, bottom-side down. Now all jacks are along the top. The two main controls, volume and select, face towards the user. I put them on the right since I am right-handed.

The audio jack should be near the volume control, since they are part of the same circuit. The key jack is likely to be used more often than the audio jack, however, so it should be easily accessible at a corner rather than along an edge. The Power Jack should also be near on an edge.

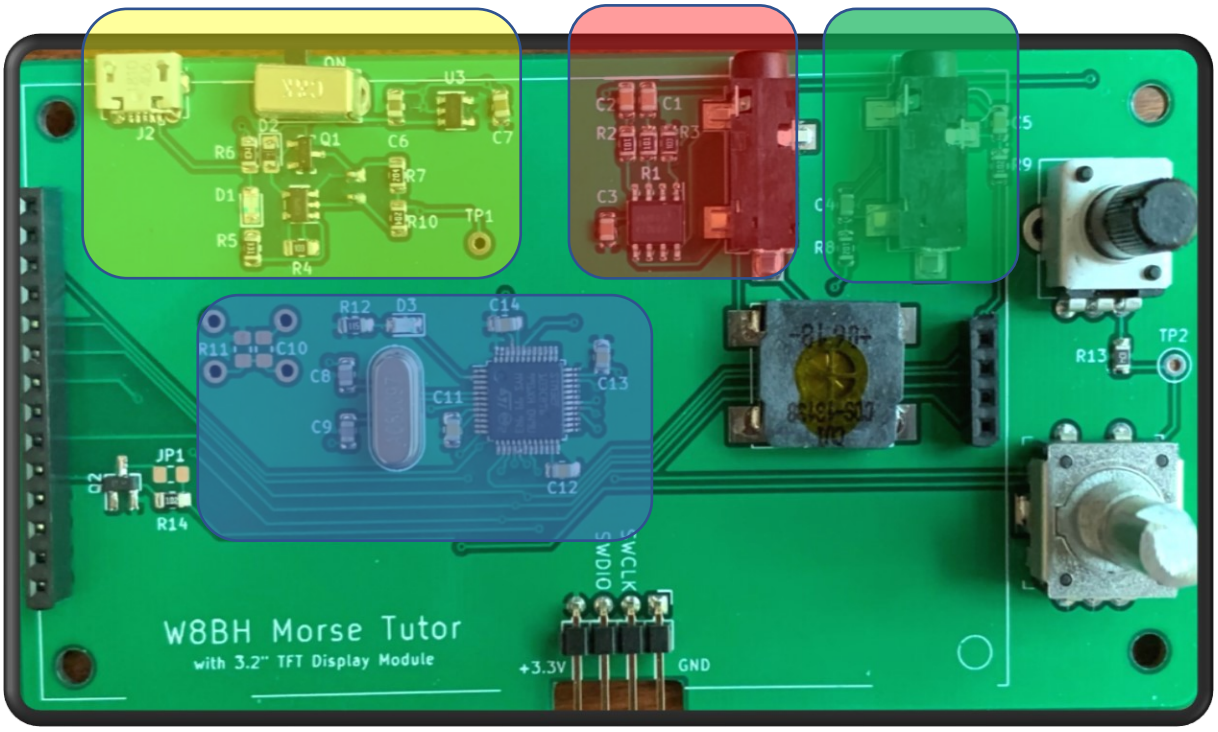
Notice how the connectivity and interface components dictate part placement in general.



Another design consideration was the SD card slot. Although I attempted to remove as much dependency on pre-built modules, the Chinese-manufactured 3.2” display (with built-in SD card holder) is tough to beat. Soldering up and mounting your own TFT display is challenging. So, I stuck with the MSP3218-style board. This board places the SD card on top - see dotted rectangle above. The SD card connector is mounted on the bottom side of the display PCB, and could potentially collide with any components mounted below it on the main PCB. Therefore, this area was kept free of connectors or tall components. The photo at left shows alignment of top edge components with the SD card on the display PCB.



Here is the final PCB layout, showing the power section in yellow, the microcontroller section in blue, the audio section in red, and the Morse key interface in green. The volume control and rotary encoder control are on the right.



The next section contains instructions for building your own Pocket Tutor.

73, Bruce.