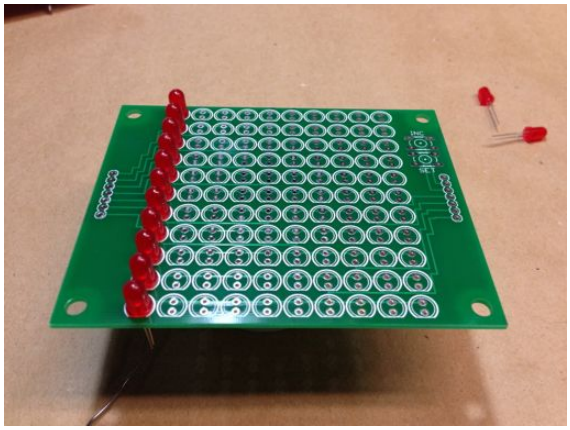
LifeClock Build Instructions

PARTS LIST:

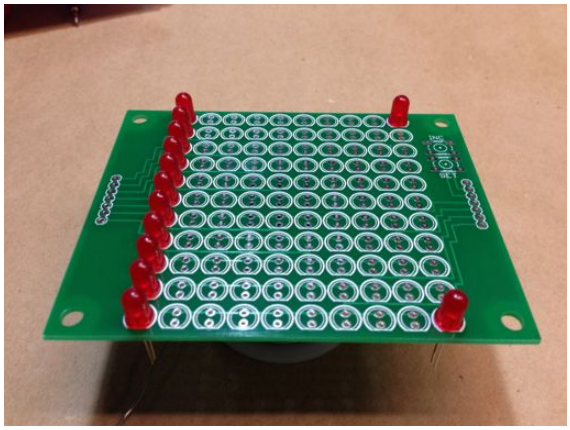
Quantity	Name	Location	Description
1	Display Board	-	
112	LEDs	-	Red
3	Tact Switches	INC, SET, RESET	
2	7 pin male headers	-	
4	Sets of 6-32 hardware	-	
1	Controller board	-	
1	6 pin right angle male header	FTDI	
2	7 pin female headers	-	
2	100 nF capacitors	C1, C5	Labeled with “104”
2	22 pF capacitors	C2, C3	Labeled with “22”
1	10 μF capacitor	C4	10μF electrolytic can
11	68 Ω resistors	R2-R12	blue-gray-black-gold-brown
11	27 Ω resistors	R2-R12	red-purple-black-gold-brown
2	270 Ω resistors	PWR-R, R13	red-purple-black-black-brown
2	2.2k Ω resistors	2.2k	red-red-black-brown-brown
1	4.7k Ω resistor	4.7k	yellow-purple-black-brown-brown
2	10k Ω resistors	10k	brown-black-black-red-brown
1	16 MHz crystal	Q2	16.000 can
1	32.768 kHz crystal	Q1	2x6mm canister
1	PTC	PTC	Crimped legs - reads XF025 one side.
1	USB-B Jack	USB 5v	
1	20mm battery holder	CR2032	
1	CR2032	-	
1	8 pin socket	DS1307	
1	28 pin socket	ATMEGA328	
1	DHT11	DHT11	4 pin SIL, blue body.
1	DS18B20	DS18B20	3 pin TO-92
1	DS1307	DS1307	8 pin DIP chip
1	ATmega328P	ATMEGA328	28 pin DIP chip

BUILD INSTRUCTIONS:

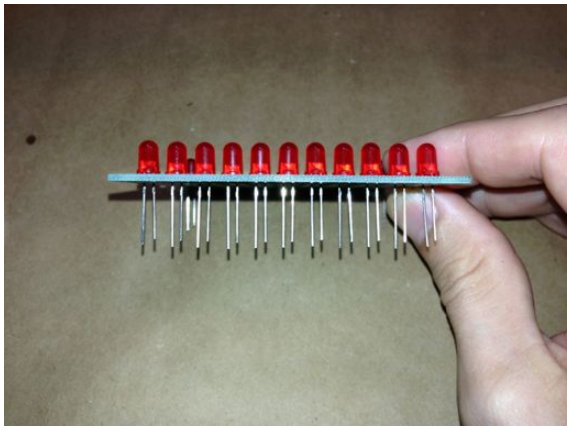
You can build either board first. I usually start with the display board because it's actually the more fiddly board to build, so I like to start it when I'm fresh and relaxed.



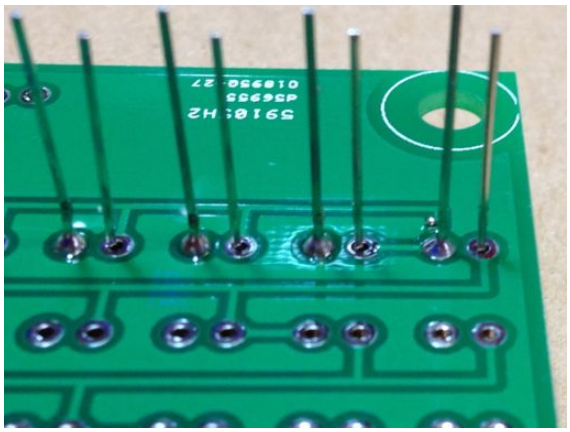
1. Insert a whole column of LEDs at once, either on the left or the right side - whichever is more comfortable. The back will become a bed of nails when you're done regardless of which direction you fill them in. Make sure the flat side of the LED matches the flat side on the silkscreen. You can also just remember that the long lead goes in to the top hole. Whichever way you use, make sure they're all inserted in the proper orientation before proceeding, as LEDs are directional and won't work properly if you wire them in backwards.



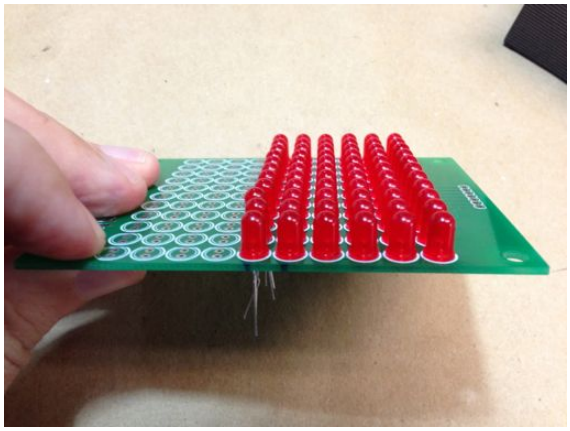
2. Once all 11 LEDs are fit to a column, put 2 more LEDs in the opposite side's corners to assist in keeping the board flat when you're soldering them.



3. Sight down the bottom of the board and make sure all 11 LEDs are fit the same way, then stack the controller board on top of the LEDs and invert the whole sandwich so the leads are pointing up and available for soldering.

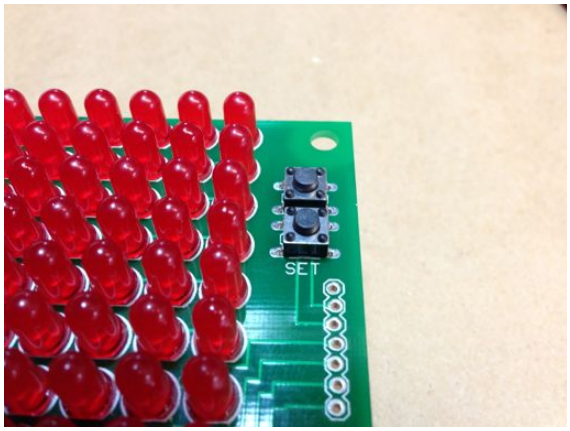


1. Go down the column and solder one leg on each LED. You can but don't have to solder the stabilizer LEDs yet. Do not solder both leads yet, it's easier to align the LEDs when you only have one leg soldered.



2. Flip the board over so you're looking at the face of the board and carefully reheat each solder joint while pressing on the LED head. It should snap into position and end up level and lined up with every other LED in that column. Sight down the board and make sure everything is straight and level before proceeding to the next step. It's much harder to realign a LED after both legs are soldered, so fiddle with it now. The upside is that 5mm LEDs self-level fairly easily, so this is a fairly easy step. But if you don't touch them up the LEDs won't be aligned and the display won't look as nice as it does if you do this step.

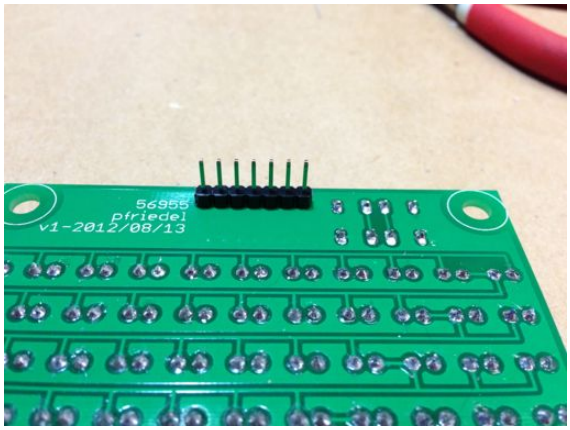
You can see a poorly aligned LED in this picture in the leftmost column. The other 5 columns have been aligned already in both axes, so they're as aligned as you can get without an external jig. I can't stress the importance of making sure every LED is aligned before you solder the second leg.



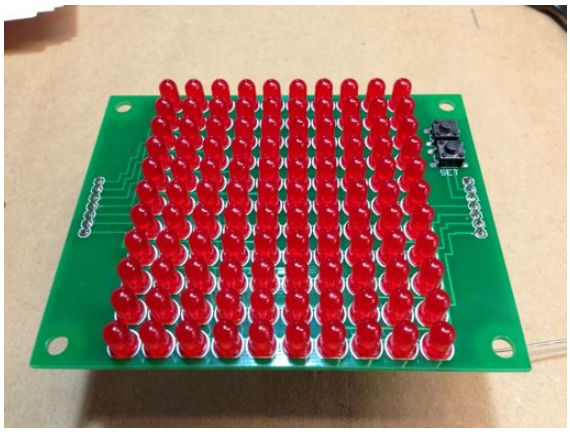
3. Once you're satisfied that every LED is level and lined up, do one last check to make sure you inserted every LED properly, and then solder the other leg. Once all 22 legs are soldered in, clip the leads as close to the board as you can - there isn't much space available between the boards, although I've tried to design it so that there isn't much that conflicts between the two boards..

Repeat the prior steps for the next 9 columns.

4. Snap in the two buttons on the front of the board so the legs stick through the same side as the LEDs and solder all 8 pins. They should snap in easily. You don't need to clip the leads off of the switches.



5. Insert the two 7 pin male headers the opposite way of everything else - you will be soldering them from the top, with the black plastic retainer and longer pins sticking off the bottom of the board. When you're done with this step, you will have all the LEDs and switches on the top of the board and the headers on the bottom of the board. The black plastic retainer should be on the bottom of the board.



6. When you're done, the display board should look like this.

The display board is finished. Take a break before continuing on to the controller board.

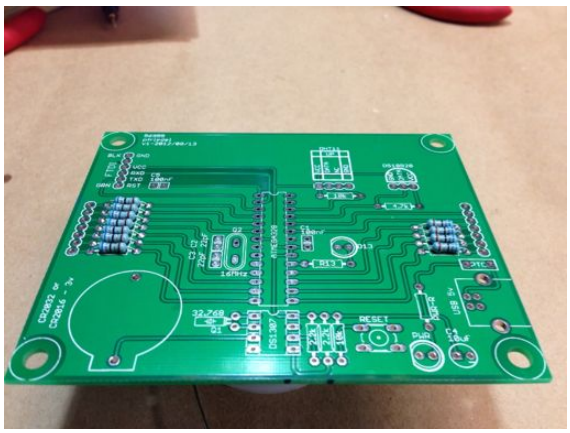
Now you can start on the controller board. Start with the low profile components and work up to the tall components.

1. Populate R2-R12 with your choice of the following resistors:

a. 68 Ω resistors (blue-gray-black-gold-brown) This is the calculated resistance for red and green LEDs at this source voltage, and then halved since each LED is lit through two resistors.

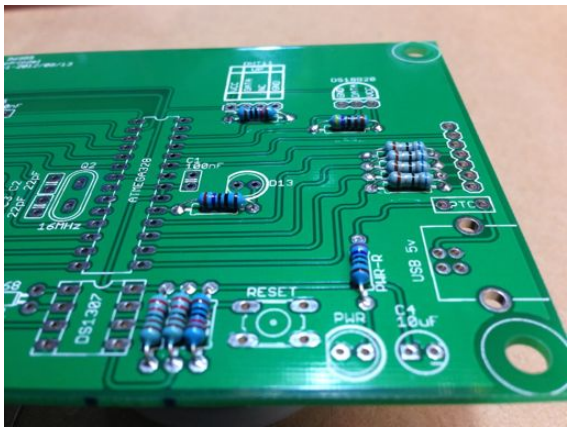
b. 27 Ω resistors (red-purple-black-gold-brown). If you replace the LEDs with blue or white LEDs, this is conveniently also the correct resistance for those. They should work for the red and green LEDs, but I haven't tested them.

c.That said, you can just bridge these with wire jumpers (cut off LED leads, for example) for the brightest display. The charlieplexing code pulses each LED in such quick succession that they never have a chance to overheat and fail. However, if the code crashes or something else gets hung up, it's entirely possible that you will dump between 4 and 5 volts across a LED which will usually cause it to fail. If it fails open, the rest of the board should be fine and you just need to replace the dead LED with a spare. If it fails to a short, you might destroy the ATmega328P. The additional downside to not using a current limiting resistor is that every single LED's particular brightness characteristic is exposed. Some are brighter, some are dimmer, and not using a resistor means you can end up with an uneven display. I don't recommend this as a first resort, but it might be necessary if you're using your own LEDS that are dimmer.



My recommendation is to solder in the 68 Ω resistors as indicated on the board and continue with the rest of the build. If you feel that the display is too dim once you've completed the rest of the kit, solder the 27 Ω resistors in parallel on the back of the board which provides a combined resistance of just under 20 Ω. If the display is *still* too dim, you can replace the 27 Ω resistors with wire leads and bypass the resistors entirely. For the LEDs that are provided with the kit, I feel that the 68 Ω resistors make a display that is bright enough indoors at the maximum brightness level. I can't make any guarantees about the lifespan or performance of the 27, 20 or 0 Ω options, as the brightness with the 68 Ω resistors is acceptable to me. But I've provided options in case you want to replace the LEDs.

Resistors aren't polarized like the LEDs, you can solder them with either lead in either hole. Solder them in place and clip the leads.



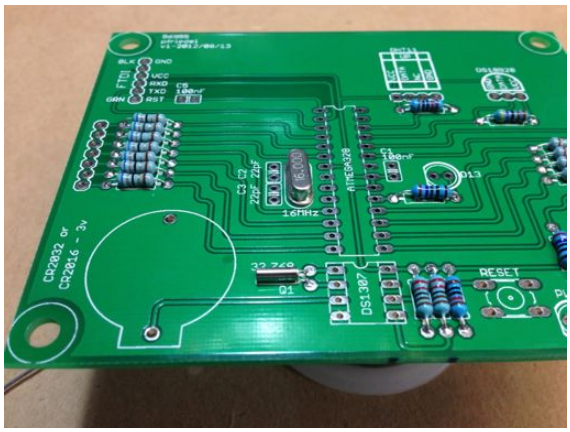
2. Put the two 2.2k Ω resistors (red-red-black-brown-brown) in near the bottom of the board. These are the pull-up resistors for the I2C data and clock lines on the DS1307 real time clock chip. Solder them in place and clip the leads.

3. Put in one 10k Ω (brown-black-black-red-brown) ATmega328P reset pull-up next to the two 2.2k resistors. This stabilizes the reset detection circuitry in the microcontroller. Solder it in to place and clip the leads.

4. Put another 10k Ω pull-up resistor in the spot just under the DHT11 location. Solder it in to place and clip the leads.

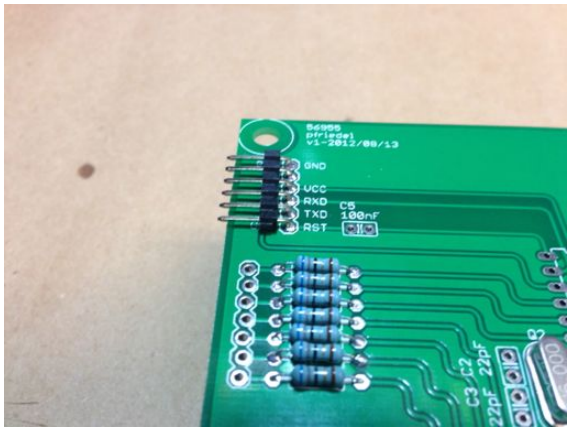
5. Put in the 4.7k Ω (yellow-purple-black-brown-brown) just under the DS18B20. Solder it into place and clip the leads.

6. Put the 270 Ω (red-purple-black-black-brown) resistors in PWR-R and R13. This is about double what those LEDs need, but it helps keep them from being too bright.

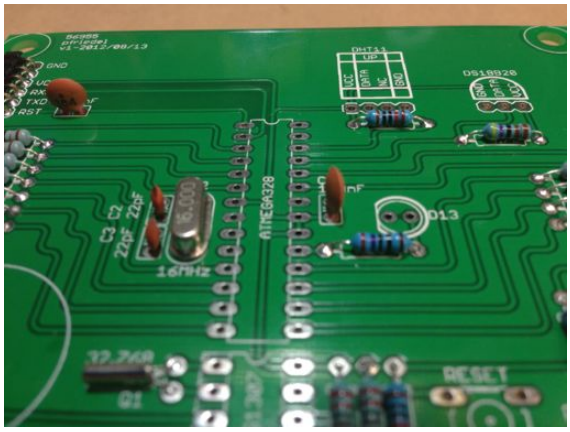


7. Solder in the tiny little 2x6mm 32.768 kHz crystal next to the DS1307 socket. The crystals aren't polarized, you can solder them in either orientation.Clip the leads when it is soldered in.

8. Solder in the bigger 16 MHz crystal next to the ATmega328P socket. This crystal isn't polarized either. Clip the leads again.

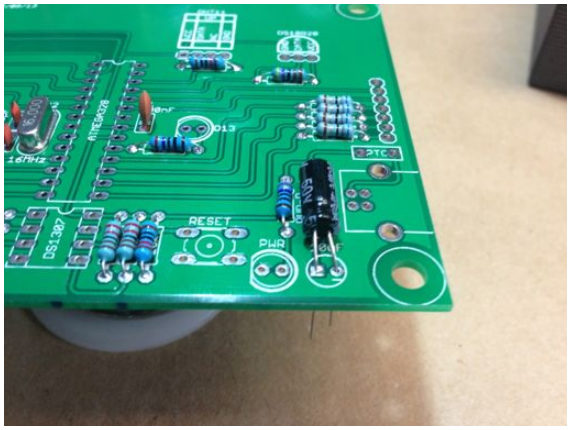


9. Solder in the 6 pin FTDI connector. The long pins should be pointing off the board. Tack one pin in place to ensure that the FTDI connection is flat before soldering the other 5 pins. These leads are too short to be clipped.

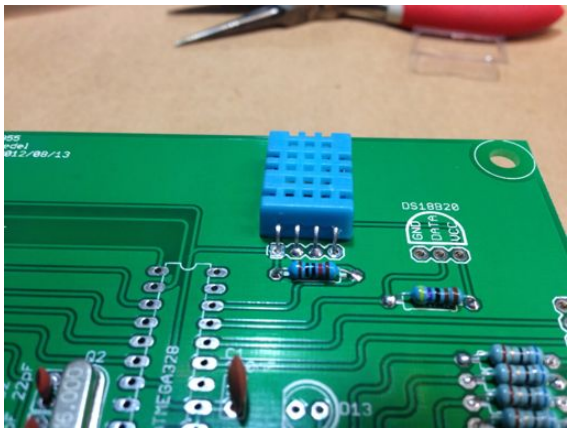


10. Solder in the two tiny little 22pF capacitors next to the 16 MHz crystal in positions C2 and C3. They're redder and smaller than the 100nF capacitors and they have 22 printed on them. These form the rest of the clock signal generation circuit for the microcontroller in addition to the 16 MHz crystal.

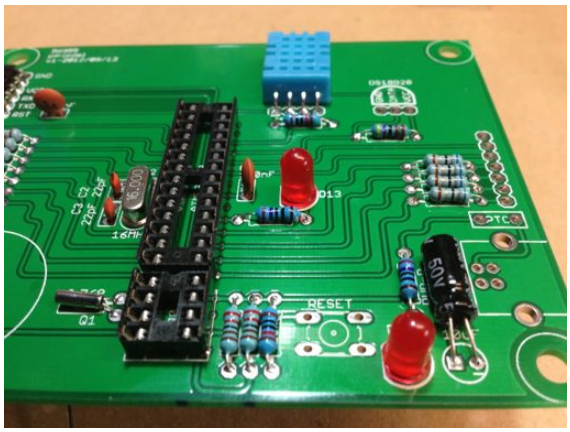
11. Solder in the two 100nF caps in positions C1 and C5. The 100nF caps have 104 printed on them. The capacitor in position C1 is a filter capacitor to help smooth out any noise entering the microcontroller, while the capacitor at C5 is there to allow the FTDI connection to reset the microcontroller to initiate a new upload. Go ahead and clip all the leads that haven't been clipped yet.



12. Solder in the 10μF capacitor in position C4. Note that for clearance reasons, this cap needs to be bent to lie along the board. This capacitor is polarized, unlike the other capacitors that you've soldered so far. Make sure you get the negative lead on the right and the positive lead in the square solder point on the left. This capacitor is also a power filter on the incoming power line. Clip the leads once you're satisfied with its position.

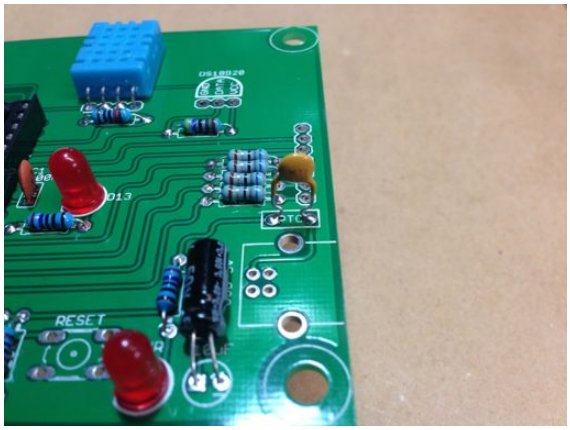


13. Bend the leads for the blue DHT11 sensor so the silver back will rest on the board and the perforated front of it is up, then solder the leads into the board. The DHT is a combined temperature and humidity sensor. While the humidity sensor is fairly consistent for the cost (more accurate sensors get expensive quickly), the temperature sensor is only accurate within 2° C, so it's supplemented with a different temperature sensor on this board.

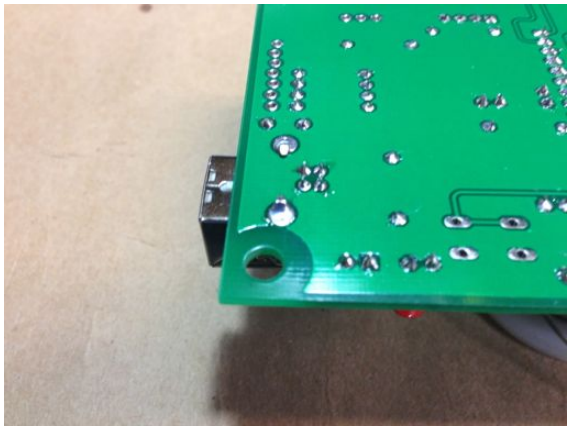


14. Solder the two chip sockets in to the board, making sure you align the notches with the notches on the silkscreen. Tack opposite corners first, then make sure the socket is tight against the board before soldering the rest of the pins. Make sure you're happy with how the sockets are laying against the board before you solder them all the way in as desoldering and adjusting sockets is pretty difficult. If you get the sockets in upside down, don't panic, just remember that the chip needs to be inserted so it is aligned with the silkscreen and not the socket.

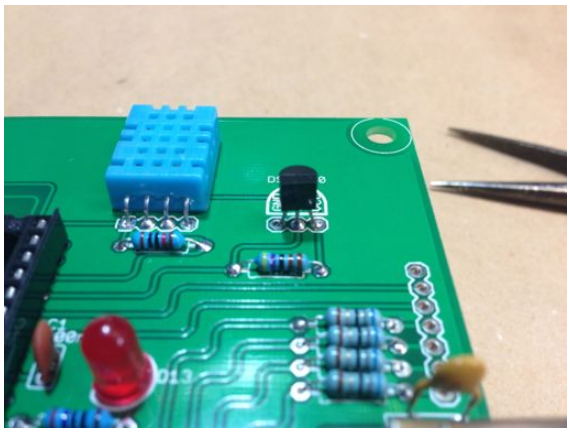
15. **OPTIONAL**: Solder D13 and the PWR LED on to the board, making sure you match the flat side of the LED to the flat side on the silkscreen. Having the power LED populated lets you know if you have power problems. D13 will only blink when you reset the board or upload new code through the FTDI connector. It's pretty hard to see through the display board, but it's still handy to have D13 populated because the default Arduino Blink sketch will toggle D13 on and off. If you experience problems, uploading the Blink sketch will help diagnose what isn't working with the board.



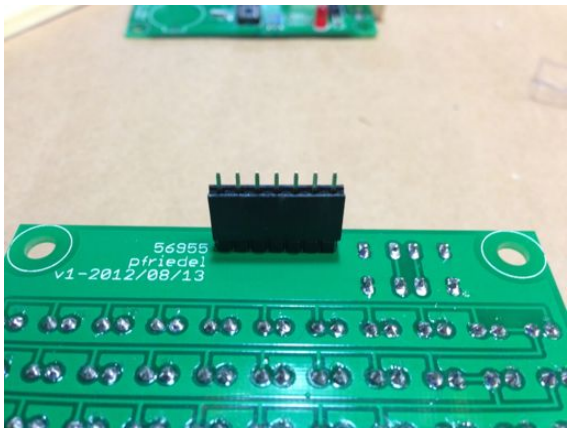
16. Solder the PTC in above the USB connector. The legs are crimped to keep it standing off the board, and it says XF025 on one side. The PTC is there to stand as one last guard against short circuits. If the board draws more than 500mA the PTC will trip and start to block the flow of current, limiting the damage the short circuit can cause to whatever is providing power. That said, the short circuit is still likely to cause damage, so be careful when you're setting down the clock after it's done. Clip the leads once it's soldered in to place.



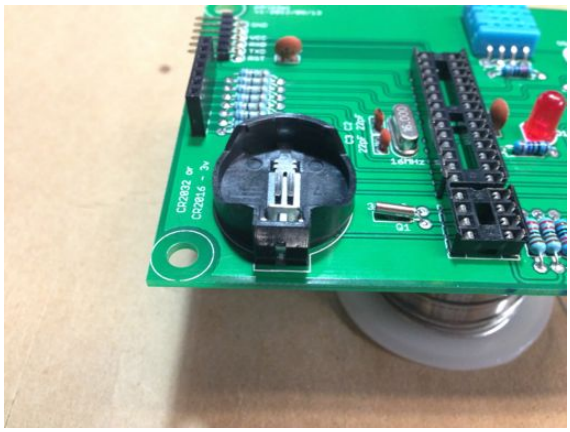
17. Solder the USB connector on to the board. Note that the two legs that go into the large holes should be bent to clasp the board. Use a lot of solder when you attach these - they provide the physical bond between the USB jack and the board, and need to be strong to stand being plugged and unplugged. Solder the 4 pins as normal.



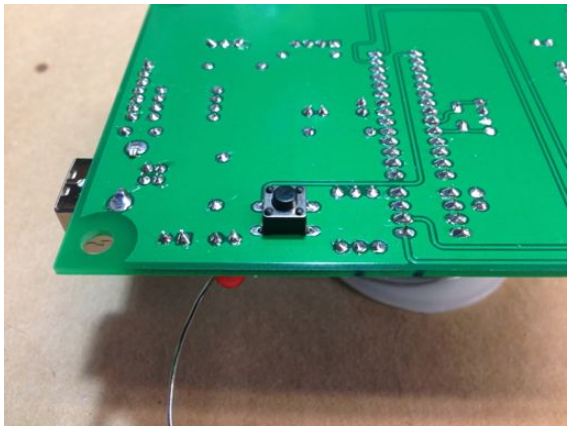
18. Spread the legs of the DS18B20 slightly and put it into the board. The round back should match the silkscreen. The DS18B20 is a thermal sensor with a much better resolution than the DHT11. Clip the excess leads once it's soldered in.



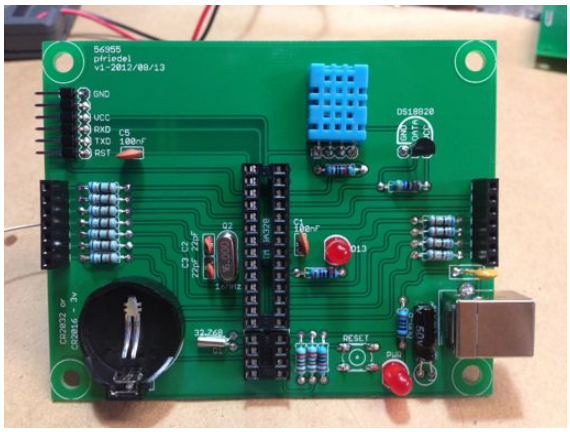
19. Put the 7 pin female headers on the male headers sticking out of the display board, then attach it to the controller board and solder them into place. Although it isn't highly critical at this time that you get the display and controller boards oriented correctly it won't hurt either, so make sure they're aligned properly before you solder them in to place. Detach the display board and proceed.



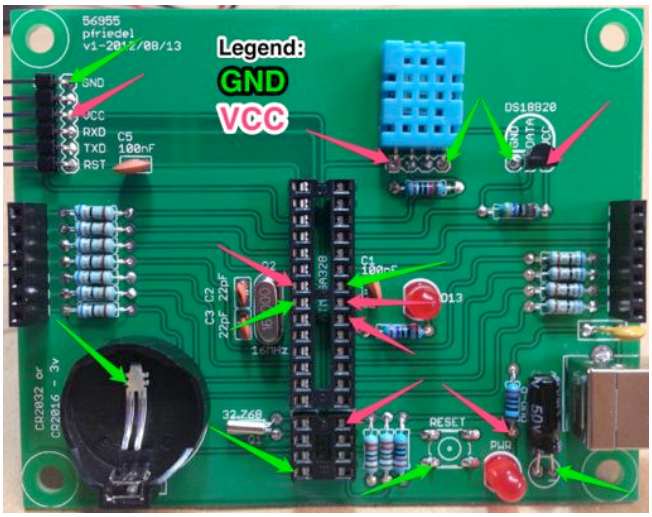
20. Put the battery holder on the board following the silkscreen. The round side of the battery holder should be up so the battery is cradled by the positive battery terminal when the unit is upright. Do not get the battery holder in backwards, as that will destroy the DS1307 realtime clock when you insert the battery. Do not insert the battery at this time. When you do insert the battery before powering the clock up for the first time, the positive face of the battery should be visible in the battery holder.



21. Flip the board over and put the reset switch on the back side of the board and solder it in place. If you put it where the silkscreen indicates, you won't be able to reach it between the board when everything is assembled. On the other hand, having the reset button on the back makes it click a little too easily. Either way works, and is entirely your call. If you eventually plan on putting the clock in a case, having an externally accessible reset switch might be convenient.



The finished controller board should look like this when you're done. Make sure you didn't miss any components. There will be some leftover resistors and LEDs, but there shouldn't be any other electrical components.



Before you apply power, check your soldering job and look for bridges or bad joints. If you want to check power pins, the following should all be connected to each other and not to any of the other side:

Ground:
- FTDI ground pin
- DS1307 pin 4
- ATmega328p pins 8 and 22
- DHT11 pin 4
- DS18B20 pin 1

VCC:
- FTDI VCC pin
- DS1307 pin 8
- ATmega328p pin 7, 20, and 21
- DHT11 pin 1
- DS18B20 pin 3

If those are all good you should be able to plug a USB cable into the socket and see if the power LED lights. If you probe the following points you should see between 4.5v and 5v:

- Ground -> VCC pins on the FTDI connector.
- Pin 4 -> pin 8 on the DS1307.
- Pin 8 -> pin 7, 20 and 21 on the ATmega328p
- Pin 4 -> pin 1 on the DHT11
- Pin 1 -> pin 3 on the DS18B20

If the power LED lis illuminated and you're seeing between 4.5 and 5v on those pins you're safe to put the chips in their sockets. You will need to straighten the pins somewhat to make them fit. Either use your thumb to carefully straighten out the pins or use your desk to straighten all of the pins on a side at once. Make sure you align the notch at the end of the chips with the notches on the sockets and silkscreen. I've dabbed a bit of silver paint on the chips to indicate where pin 1 is, but the ATmega328P and DS1307 both have pretty obvious alignment markers. Go ahead and insert the CR2032 battery now, too. Plug the board back into power and press the reset button when the power light is on. D13 should blink a few times. If D13 blinks you know the microcontroller is alive and functional and you should be safe to unplug it and connect the display board to the controller board. Make sure you get everything aligned properly - the display board should have the buttons in the upper right and the controller board should have the USB connector on the lower right. Once the boards are connected, plug it into power and it should start up. It will automatically play, and the clock will default to 0:00. Which brings us to the setting mode.

SETTING MODE:

Hold down the SET key for approximately 5 seconds any time that the time is being displayed and you will enter setting mode. (Technically: Pressing SET while the time is displayed in the Life mode breaks out of Life and displays the time and temperature. If the SET button is depressed when the humidity finishes scrolling off the screen, it enters setting mode.) There are 3 screens in the setting mode, each accessed with additional presses of the SET key:

1. Hours (displayed as numbers on the top of the display)
2. Minutes (displayed as numbers on the bottom of the display)
3. Brightness (displayed as inverse numbers in the middle of the display)

In each mode, you can press the INC button to increment the number. It will wrap around at the end of the available range - hours from 0-23, minutes from 0-59 and brightness from 1-19. Setting the clock doesn't reset the seconds, so keep that in mind if you try to sync it up with a more well regulated

clock. The DS1307 is only as accurate as the crystal it gets its signal from, and the crystals are usually only accurate within a minute or so per month. The controller will return to normal functions after 5 seconds without a button press.

POWER SOURCES:

1. USB in on the USB-B connection. No data is transmitted, the system just uses the power provided by the USB bus. The system doesn't draw that much power, so even low power USB power supplies will keep the system running.
2. 5V DC in on the FTDI connection. There are 6 pins visible on the top right of the system board. The top one is ground, the second one is not connected and the 3rd one is 5V DC in. The fourth and fifth are serial TX and RX respectively while the 6th is DTR. A standard FTDI cable will work to reprogram the system or to power the system.

There are no protection diodes on the board. This isn't a problem with the USB cable since it's a specified connection, but the FTDI connection has the opportunity for wiring it in backwards.

POWER:

Note that while the LifeClock can accept power over either the FTDI connection or the USB connection, do not plug in both at the same time. It hasn't been tested. The USB connection should work with all but the weakest of USB power adaptors - average draw for the clock is under 40mA, so even 200mA adaptors should work fine. The USB data pins are completely disconnected. I have tested it with several computer USB ports and they all provide sufficient power without going through a USB power handshake, so it should work on most computers. If you get weird numbers (25:16, usually) when you try to enter the clock set mode, your USB power source is providing too little power. I haven't noticed any physical problems as a result of that - the board will happily run at 4.3v, and the display works at 3v, but the clock chip and temperature sensors stop working.

DISPLAY MODES:

1. Conway's Game of Life: Life is a zero-player game that the system will play by itself on an 10x11 grid that wraps around the edges. It updates to a new generation every half-second, like the ticking of a clock. Every 10 generations, it pauses to display the time for a second before resuming the game. Every 60 generations, it displays the time and the temperature. Each generation is about a half second long. Eventually the game will enter a steady state or an oscillation between two states, at which point the system will halt the game and select a new mode.
2. Rain or Matrix mode. Pulses of light descend from the top of the display to the bottom of the display for 5 seconds. The speed and density of the rain are both random.

Between modes, the system will display the current time for 3 seconds before displaying the temperature and humidity.

FTDI PORT:

The FTDI port provides a serial data stream of the activity on the display. If you have a 6-pin FTDI standard connector, connect it with ground (black) up and CTS (green) down. Otherwise if you have another serial solution, connect ground to pin 1, TX to pin 4 and RX to pin 5. The data rate is at 115200 baud, 8-N-1. The FTDI connector will power the board, so don't plug it in to USB power at the same time.

BACK UP BATTERY:

Any 3V 20mm battery will work. I ship with a CR2032 which should last for many years, but if it runs down prematurely, it can be replaced with a CR2016 or a CR2032

SOURCE CODE:

The software repository for this project is at http://github.com/pfriedel/LifeClock

Any questions? Mail me - pfriedel@compulsive.net.