Name: Todd and Sofia Krein

Project Number: TI002

Project Name: Scene Controller for A Christmas Tableau (SCACT)

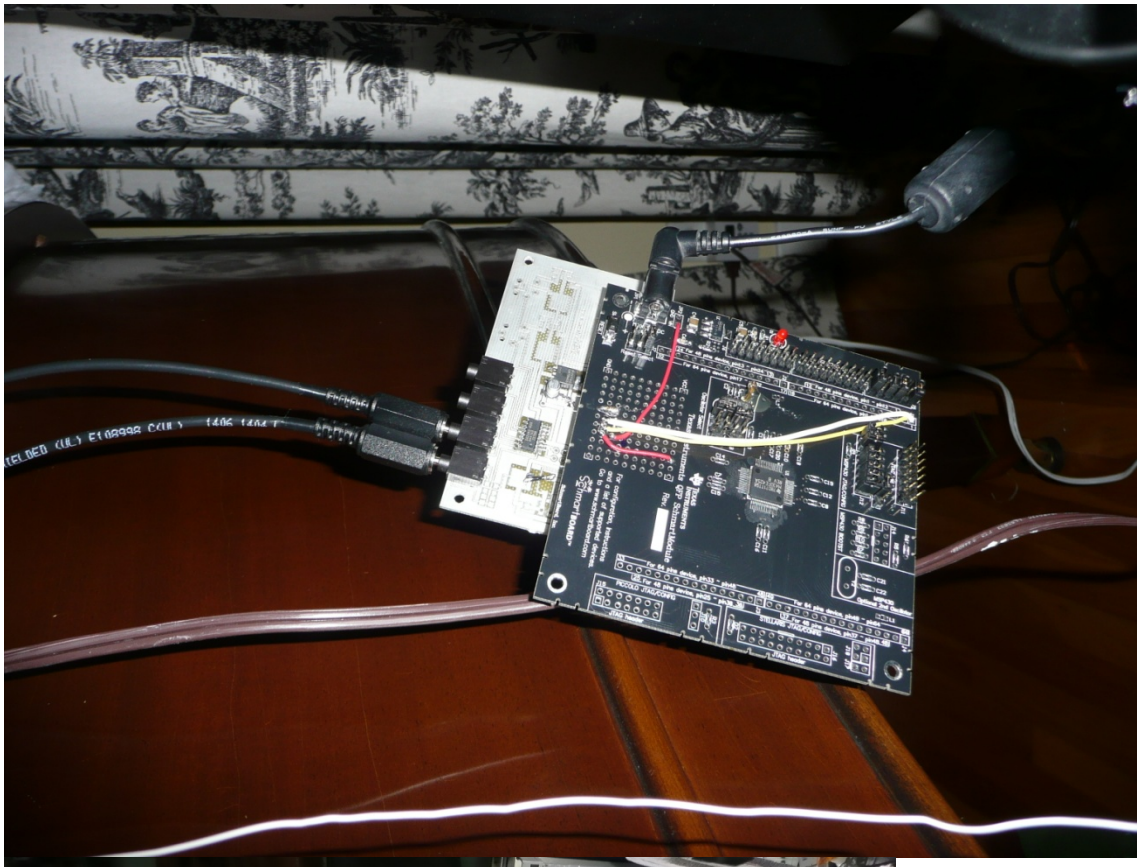
Project Description: SCACT automates the control of a complex Christmas village display, with every node in the display containing its own programming. Distributed power and one-wire networking means that the nodes (houses, characters, street lights, etc.) can be placed anywhere in the display area, or even added or removed, with no changes to the controlling program.

Each node is connected via a three-wire connector to a socket placed below the display board. We use peg board as a base, which allows a three pin pin-header to fit up through any of the holes. This means that power and control are never more than 1" away from where we need it. The three pins supply power (+5V), ground, and a 5V 1-wire network. Although 1-wire devices can power themselves parasitically, most of the nodes include LEDs (or other lights), and I didn't want them to flicker.

Inside each node is one or more 1-wire devices, generally one that contains both EEPROM and programmable I/Os. The EEPROM stores information such as a description of the node, as well as times actions for the node. For example, a simple house node might turn on its one LED at 5PM, and turn it off at 11PM. A more complex example is the village inn, which has its downstairs lights turn on around 5PM, including the flickering fireplace, and then the upstairs light turn on, individually, later in the evening, and then they finally all turn off. This simulates the guests congregating in the main room, and then heading off to bed later in the evening. The church bell is programmed to ring each hour, and the town crier calls at 5:15, 5:30, and 5:45, announcing that Charles Dickens will perform a reading of "A Christmas Carol" at 6PM. (For family sanity, I don't have a module to read the book.) At 10:30, Santa flies over the town.

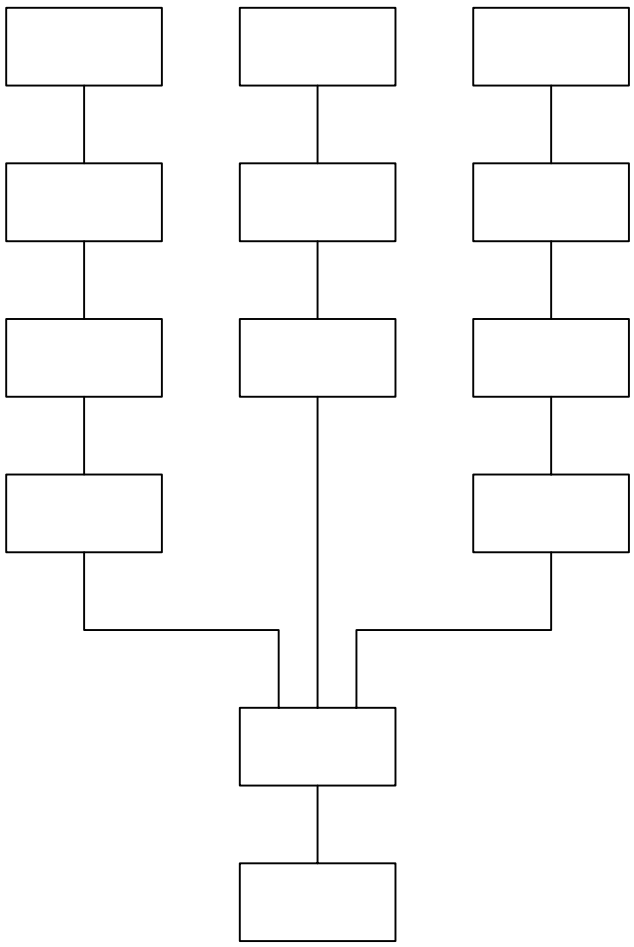
The code within the MSP is fairly straight forward. The network is scanned to find all available nodes. Each node is then scanned to find the first chronological event. Assuming it's in the future, the MSP sleeps until that time. Once awoken, the bus is rescanned, and any events that should take place at that time are executed, and the next event time is calculated. The MSP then sleeps until that next event time.

N.B., there is one piece of explanation required while looking at the design and the system code. The project was originally started using an MSP430G2231 (on a launch pad board), but the programmability desired quickly outstripped the capability of that processor. I moved to the Schmart board with an MSP430FE425, since I had one handy, and continued the work. Unfortunately, it proved unreliable in the soft I2C interface. I then moved to the MSP430F4132, which used for the final project. However, I'm sure that crumbs from the first two trails still remain in the code. Caveat Emptor.




SONY
TC-WR950

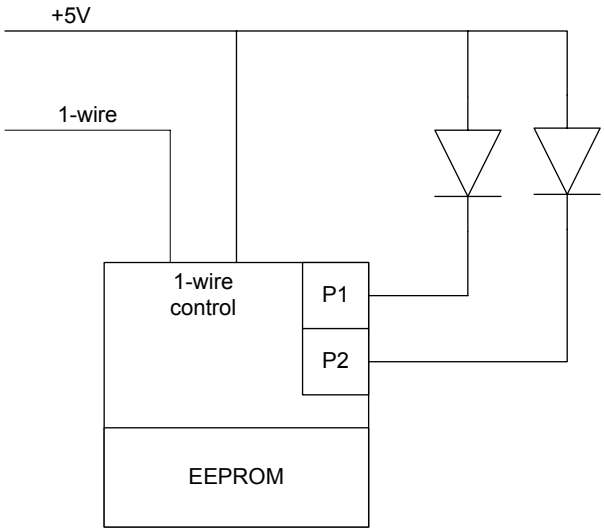
Block Diagram:



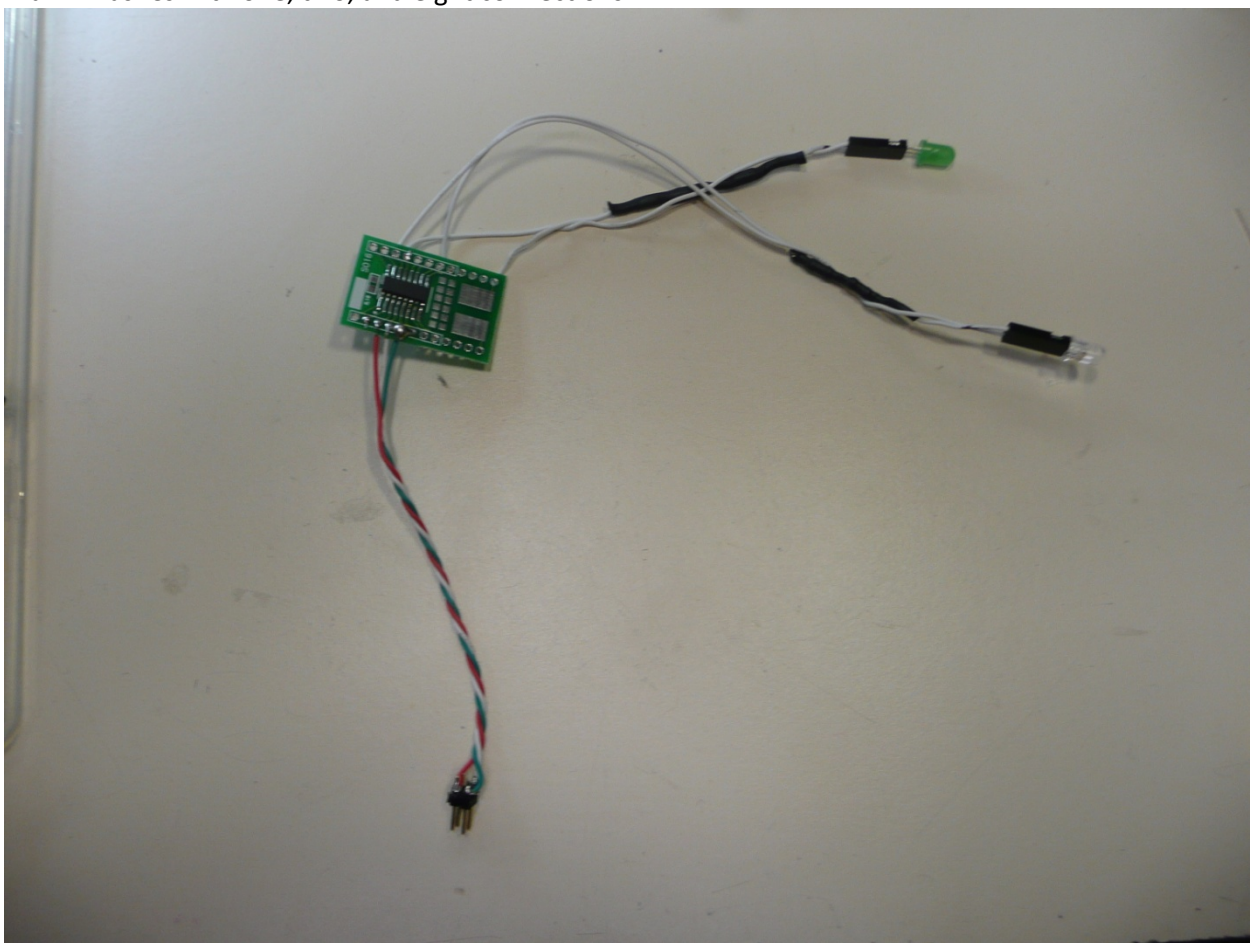
House 1

House 2

Each node contains a variation on:

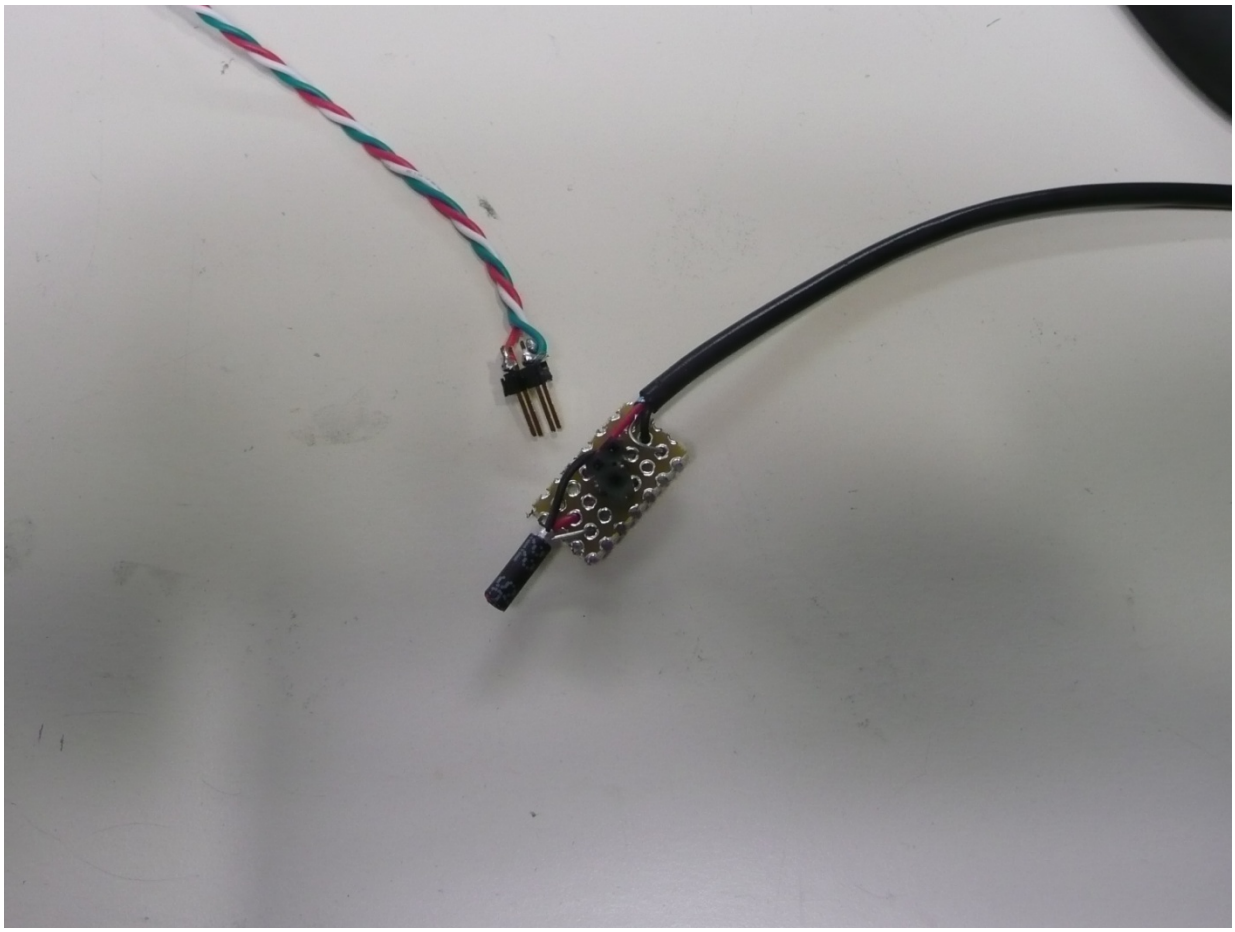


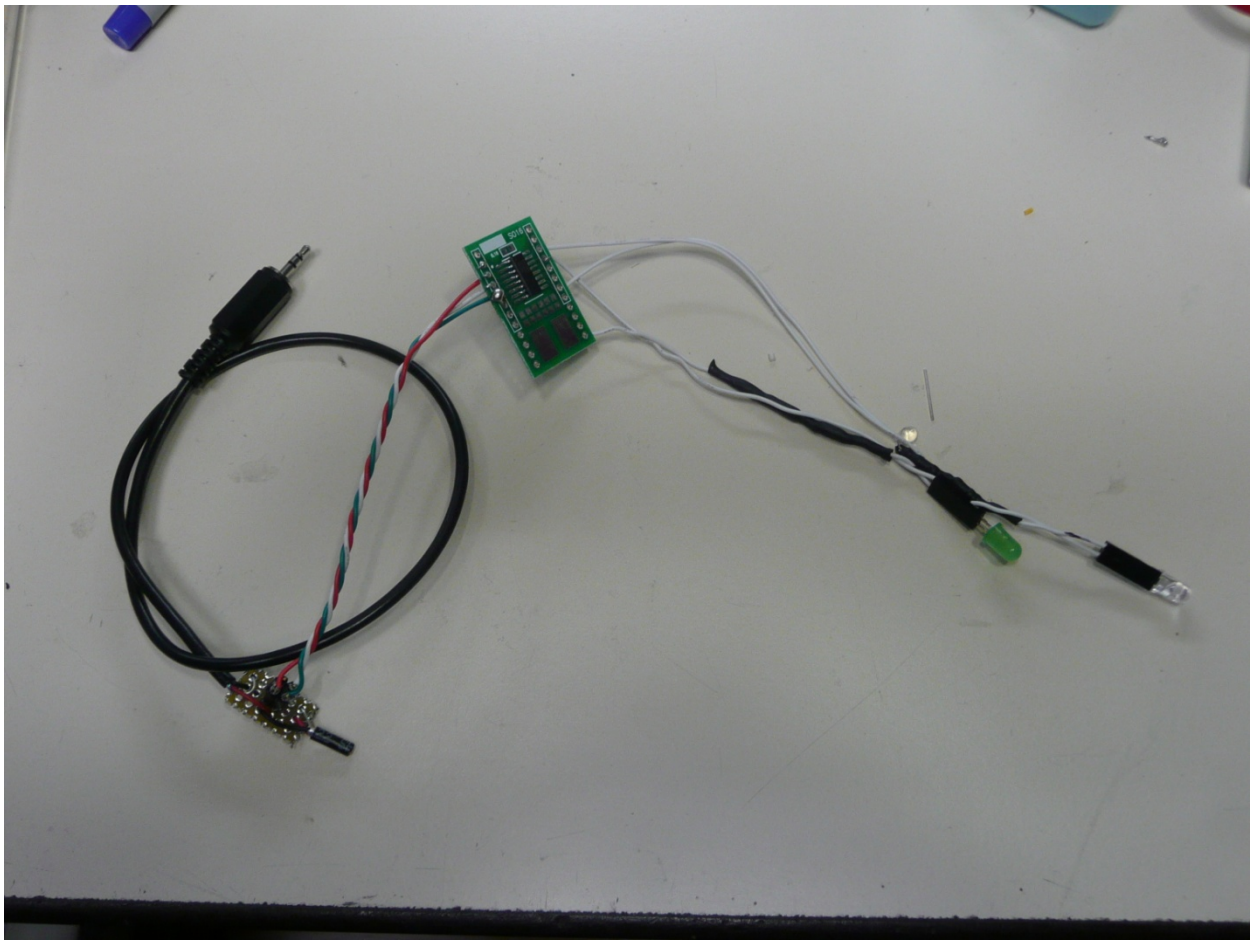
Maxim has ICs with one, two, and eight connections.



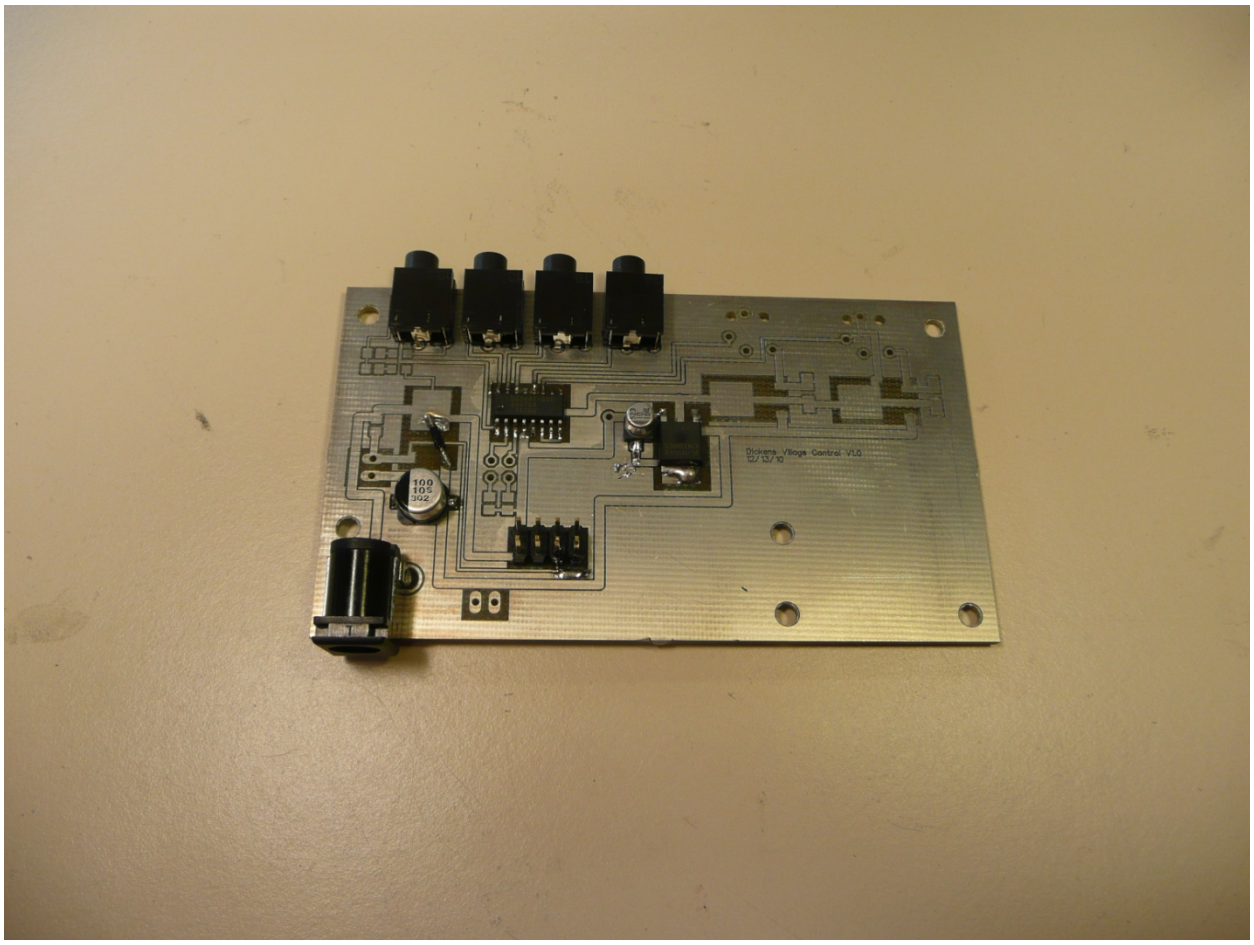
For Santa, a remote control pigtail (available from SparkFun) is controlled by the 1-wire device, as Santa requires full wall power to keep his reindeer moving.

The design is implemented as multiple (17) node boards, included as “power_spigot” in the attachments. (Note that for this year, each was hand soldered on perf board – I hope to have some made for next year, now that it all works.) They are organized into three chains, to keep the network short, ending in a sub-mini plug.





The main power and control board contains the voltage regulators, the main power control FET, the I2C – 1-wire bridge and mux chip, along with a header that connects to the MSP430 dev board. The power board contains two high power connectors that weren't required this year, and remain unstuffed. Likewise, the power control FET wasn't populated to make the debugging easier.



The MSP430 dev board drives the I2C bus, and has a connection to drive the main power FET, as well as an unused connection for an LCD display. We'll see how much further we get next year!

Bibliography and Resources:

- Maxim app note 3684, and its code
- Maxim DS2406 data sheet
- Maxim DS2482-800 data sheet
- TI MSP430F4152 datasheet
- TI MSP430FE425 datasheet
- TI MSP430G2231 datasheet
- TI slac288a sample code