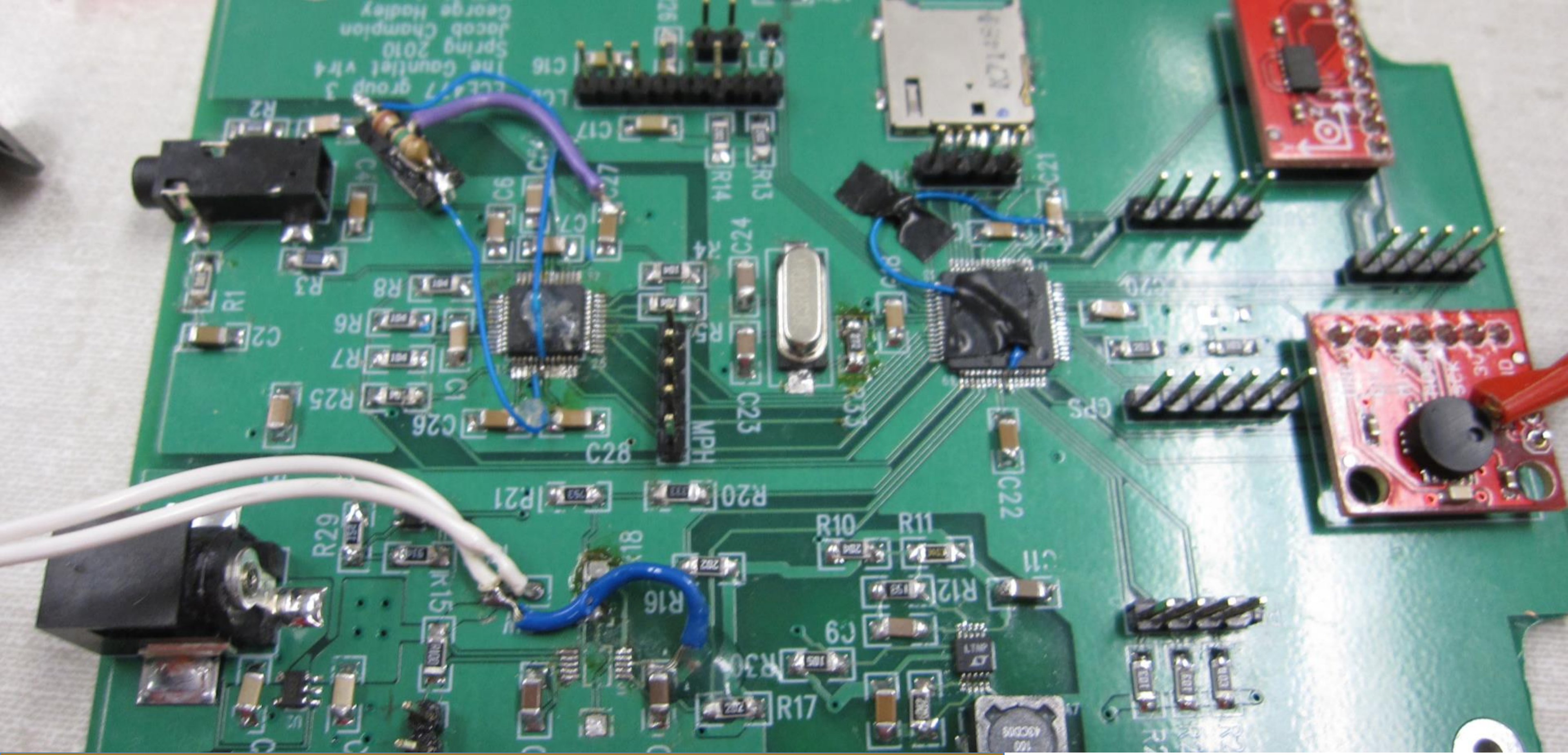


VERIFICATION AND DEBUGGING

OUTLINE

- Purpose and Motivation
- Do’s and Don’ts of Debugging
- The Debugging Mindset (“What could go wrong?”)
- Hardware Verification Process
- Software Verification Process
- Debugging Case Study

PURPOSE AND MOTIVATION

- Once hardware has been designed and software has been written, it comes time to verify whether or not the proposed design works
- The process of verifying hardware and software can be tedious
- Employing a methodical approach to debugging can deliver superior results and shorten overall verification times

PURDUE
UNIVERSITY

DO'S AND DON'TS OF DEBUGGING

- DON'T solder parts, attach wires, connect probes, etc. while a circuit board is energized – Temporary shorts can destroy your circuit board and its components. The tip of the iron is GROUNDED, and tying random points of the board to ground can be very bad!
- DON'T attempt to probe between the leads of your microcontroller – A short can be disastrous and possibly destroy your microcontroller. Include test points and pin headers instead.

PURDUE UNIVERSITY

DO'S AND DON'TS OF DEBUGGING

- DON'T attempt to power different parts of your circuit with different (external) power supplies – Even small differences in voltage between two isolated supplies can result in large amounts of unwanted current flowing across a set of traces. (ground loops)
- DON'T connect port pins directly to power supply rails to obtain a logic "1" or "0" when debugging – This is an easy way to destroy pin drivers and buffers, killing individual pins, ports, or the entire device

PURDUE
UNIVERSITY

DO'S AND DON'TS OF DEBUGGING

- DO systematically check for the root causes of an error wherever possible, changing only a single variable at a time – This will help eliminate potential causes and reduce your problem space.
- DO keep updated notes about errors that occur in your hardware, including your hardware setup, conditions under which the error occurs, and steps necessary to reproduce the problem – This will facilitate easier sharing of debugging information with others.
- DO keep rovers, quadcopters, etc. tethered and immobile until full system testing is necessary – this will reduce the risk of breaking equipment and injuring people

PURDUE
UNIVERSITY

THE DEBUGGING MINDSET

Root Causes and Problem Spaces

- Problem: “<x, y, z> doesn’t work.”
- Problem Space: The set of all possible issues in the hardware and software which could lead to the issue at hand
- Root Causes: The underlying issues which ultimately cause the erroneous behavior
- Efficient debugging employs a series of tests to reduce the problem space as much as possible, identify root causes, and remediate the issues to produce functional hardware

PURDUE
UNIVERSITY

THE DEBUGGING MINDSET

Class Exercise: Reducing the Problem Space

- Class Exercise:
  - The instructor is thinking of an integer between 1 and 100
  - A student must guess a number, and the instructor will then tell the student whether their guess is correct, too low, or too high
  - Objective: Using as few guesses as possible, the student must correctly guess the instructor’s number. What strategy will reliably return the answer in the fewest number of guesses?

PURDUE
UNIVERSITY

THE DEBUGGING MINDSET

Reducing the Problem Space

- In debugging hardware, there are many ways to reduce the problem space. For a given issue, consider:
  - Is the issue hardware or software (or both)?
  - Which device is the source of the problem?
  - (In the case of debugging communication links) Is the issue a transmit issue? Receive issue?

PURDUE
UNIVERSITY

THE DEBUGGING MINDSET

Identifying Root Causes

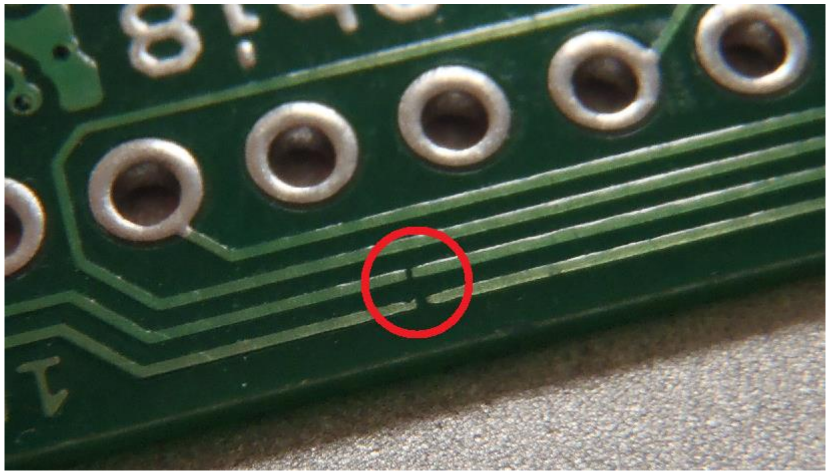
- In debugging, errors are often propagated throughout a system. Therefore, it is useful to attempt to isolate and identify the root cause of an issue.

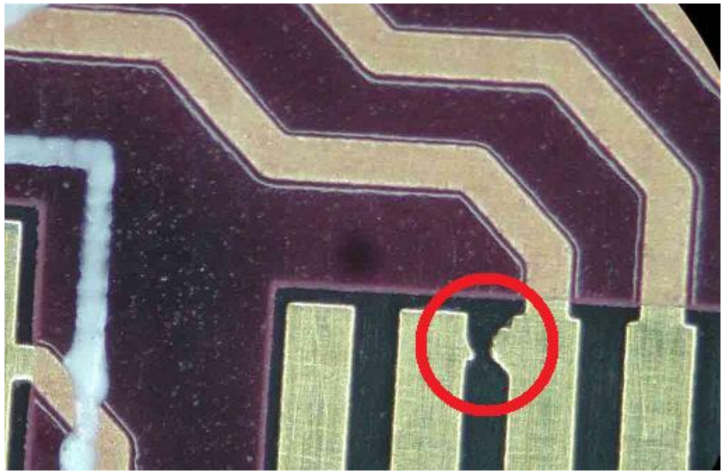
- When the problem space has been reduced as much as possible, determine the conditions under which an error occurs and the steps needed to reproduce a problem.

HARDWARE VERIFICATION PROCESS

When Circuit Boards are First Received

- Visually inspect your received circuit boards for any manufacturer defects, such as signal plane flooding. Use a multimeter to check the resistance between power and ground and ensure there are no shorts.



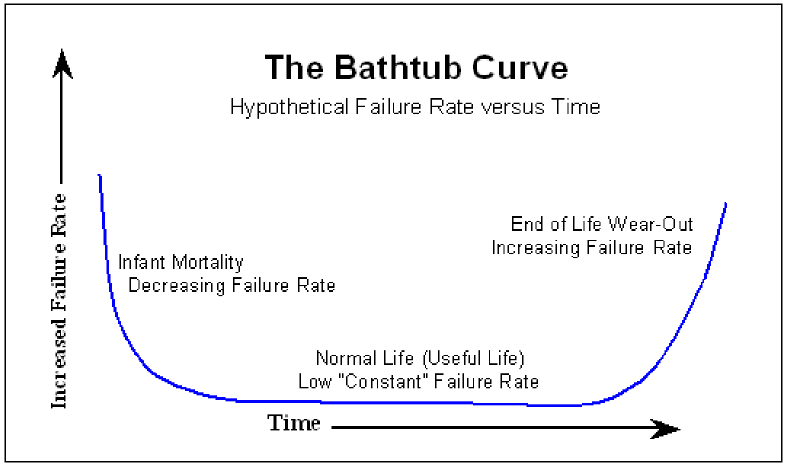


PURDUE UNIVERSITY

HARDWARE VERIFICATION PROCESS

Power Supply Testing

- Once boards have been determined to be sound, the power supply should be assembled and tested on board.
- Power supplies should generally undergo an “endurance” test, being left on for a full day (helps mitigate the possibility of infant mortality rate failures)
- Power supplies should be tested to ensure they can deliver necessary current and voltage under load. To simulate a load for the power supply, use of a “load resistor” is recommended



PURDUE UNIVERSITY

HARDWARE VERIFICATION PROCESS

Microcontroller Testing

- Once the power supply works correctly, add the microcontroller (or other relevant device) and all needed support components (decoupling caps, etc.) to the circuit board.
- Troubleshoot any issues related to programming your microcontroller, then run a simple heartbeat program to verify that the microcontroller executes code
- Bring interfaces to other devices online, one by one, troubleshooting any issues that occur interface by interface

HARDWARE VERIFICATION PROCESS

Identifying Short Circuits

- Short circuits are a common occurrence in early prototypes of a piece of hardware
- Causes: board design errors, PCB manufacturing errors, improper component soldering, damaged components
- Applying power to a board containing a short circuit can be very damaging, requiring considerable money and time to address
- Recommendation: use a multimeter to check resistance between power rails (3V, 5V, etc.) and ground *every time a new subsystem is added to a board*.
- Typical populated board power/ground resistance ~> 1kΩ

PURDUE
UNIVERSITY

HARDWARE VERIFICATION PROCESS

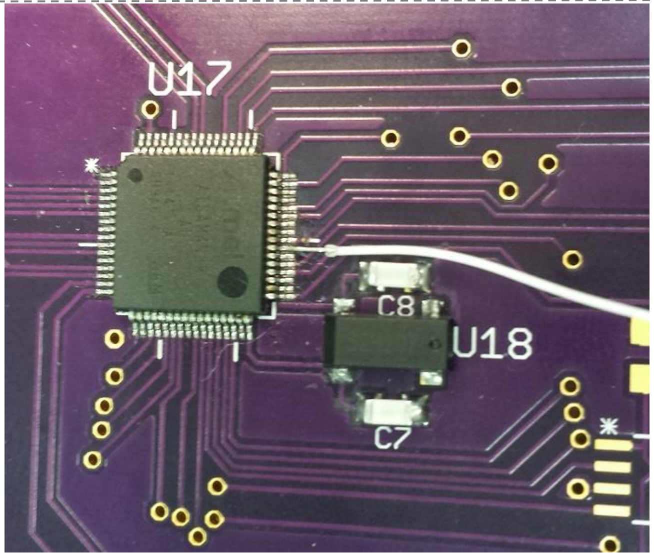
Identifying Short Circuits

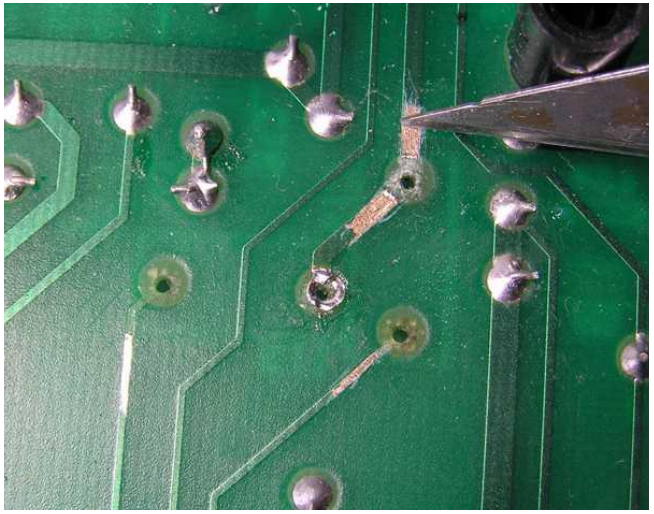
- If a short circuit is found on a board, it must be located and addressed in order for the board to function properly
- Methods of locating a short circuit between two nets:
  - <u>Visual/X-Ray Inspection:</u> Inspect the nets visually or with an X-Ray machine to look for traces which have been shorted together (nondestructive)
  - <u>Thermal Analysis:</u> Connect current-limited power supply to board, then use thermal camera to identify parts of board which heat up most (minimally destructive)
  - <u>Voltage Drop Testing:</u> Connect current-limited power supply to board (.5A or more) then measure voltage drop of traces as current travels from power to ground (destructive)
  - <u>Systematic Component Removal:</u> Remove any components connecting the two nets one-at-a-time, and check for changes in resistance between nets (destructive)

HARDWARE VERIFICATION PROCESS

Flywiring, Trace Cutting and Scraping

- Flywiring: Adding an electrical connection with fine-gauge wire (and a steady hand)
- Trace cutting: Removing an electrical connection (generally done using an exacto blade or sharpened screwdriver)
- Scraping: Carefully scraping away soldermask (but not the copper) to create new pads and electrical connection points



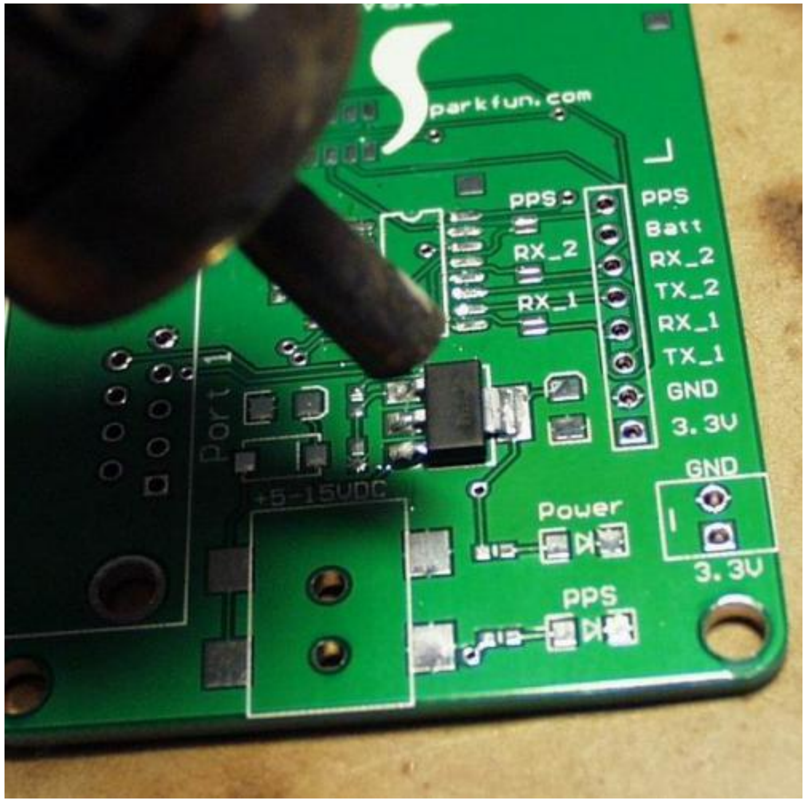


PURDUE UNIVERSITY

HARDWARE VERIFICATION PROCESS

IC Removal

- 2 Techniques for IC Removal:
  - Hot-Air Rework: Reflow all solder joints to an IC simultaneously, then carefully remove IC using tweezers or pick (while hot air is still being applied)
  - Iron and "chisel": Using a chisel, cut each IC lead individually, remove die, and then remove leads one by one with soldering iron

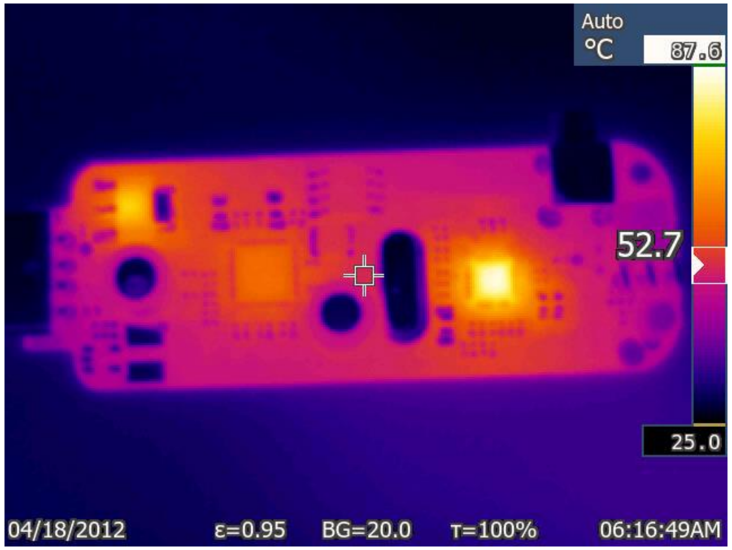


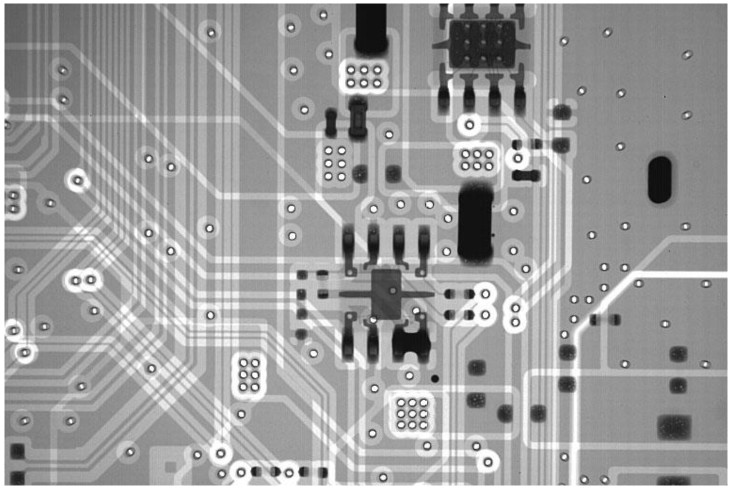
PURDUE UNIVERSITY

HARDWARE VERIFICATION PROCESS

X-Rays and Thermal Cameras

- Lab staff have access to high end debugging equipment:
  - Thermal Camera: Quickly identify hot-spots when design features a short
  - X-Ray: Verify proper soldering of leadless parts; verify boards with inner layers





PURDUE UNIVERSITY

SOFTWARE VERIFICATION PROCESS

Modular Coding

- Modularize software into a hierarchy of software “blocks”
- Verify the functionality of all low level blocks before testing higher level functional blocks
- Repeat process until software functions as desired
- Depending on code, programmers may have to test a variety of corner cases to ensure software reasonably functions under all conditions
- Microcontroller background debuggers available on most major platforms to assist with embedded-level debugging efforts

PURDUE
UNIVERSITY

SOFTWARE VERIFICATION PROCESS

RTFM

- "My Code Isn't Working."
- Follow these steps:
  1. Acquire the necessary datasheets (device datasheet, library documentation, device family reference manuals)
  2. Read said documentation to get a thorough understanding of how the system operates
  3. Verify system operation through the microcontroller debugger, step-by-step
- Google searches and Stack Overflow answers are useful for specific topics, but are not good for developing the base level of understanding needed to program a device

PURDUE
UNIVERSITY

DEBUGGING CASE STUDY

PIC24 Programming Issue

- Year: 2014 Semester: Fall
- Description of problem:
  A team had a microcontroller (PIC32MX360F512L) on their board which they were unable to program. Multiple replacement microcontrollers were tried, and it was determined that 3 of the microcontrollers placed on the board were "bad". An entire week had been dedicated to this problem and had not yet returned results.
- Exercise:
  Suggest tests that could be done to narrow the problem space and identify the root cause of the problem.

PURDUE
UNIVERSITY

DEBUGGING CASE STUDY

PIC24 Programming Issue

- Cause of Issue:
  The team had configured the ENVREG input of their microcontroller incorrectly. By grounding the ENVREG pin of their microcontroller, they disabled the voltage regulator for the core voltage of their microcontroller device. This then fed the voltage present on their VCORE pin (3.3V) into the microcontroller core, destroying the core logic of the microcontroller. By tying this pin to Vdd and using an appropriate capacitor, they were able to program their microcontroller and stopped burning up new micros.

Questions?