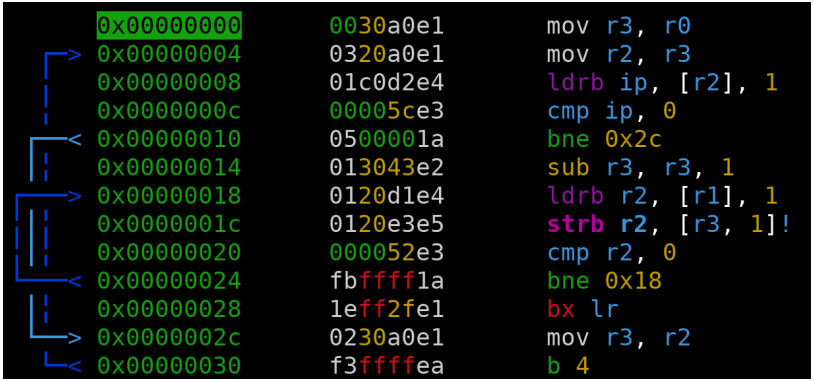
ENGR-E 399/599: Embedded Systems Reverse Engineering

Lecture 9: PCBs and serial protocols

Austin Roach
ahroach@iu.edu

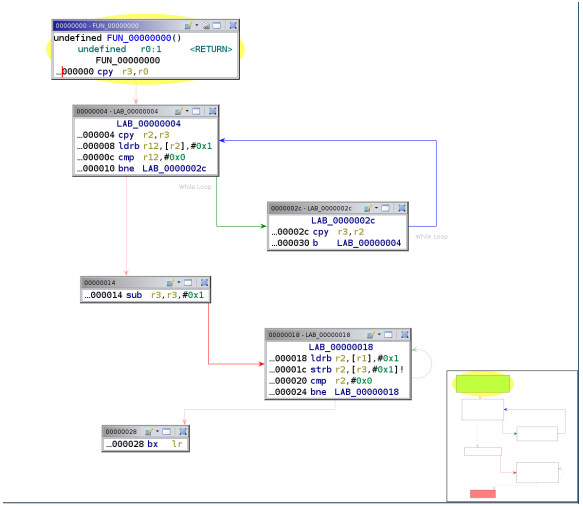
March 10, 2022

Mystery function



- How many arguments does the function take?
- What are the types of the arguments?
- What does the function do?
- What does the function return?

Mystery function graph



Mystery function revealed

```
char *strcat(char *dest, const char *src)
```

Description

The `strcat()` function appends the `src` string to the `dest` string, overwriting the terminating null byte (`'\0'`) at the end of `dest`, and then adds a terminating null byte. The strings may not overlap, and the `dest` string must have enough space for the result. If `dest` is not large enough, program behavior is unpredictable; *buffer overruns are a favorite avenue for attacking secure programs.*

Return value

The `strcat()` function returns a pointer to the resulting string `dest`.

Today's plan

- Printed circuit boards
- Serial protocols overview:
  - UART
  - SPI
  - I2C
- Demo of serial interface identification/interaction on Wifi router
- Setup for Assignment 4

PCBs

Printed Circuit Boards (PCBs) are the base for the most modern electronic systems:

- Provide a means for making electrical connections between components
- Copper layers created using lithographic techniques
- Layers connected by drilling and plating vias
- Advanced PCBs can be very complex
  - Tens of layers
  - Blind and buried vias
  - Trace widths of a few 0.001”
  - Connect BGA components with thousands of pads
  - Buried passives

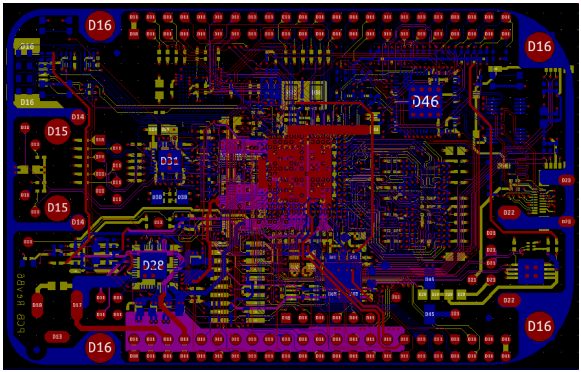
RE hints from PCBs

How do PCBs provide us hints for reverse engineering?

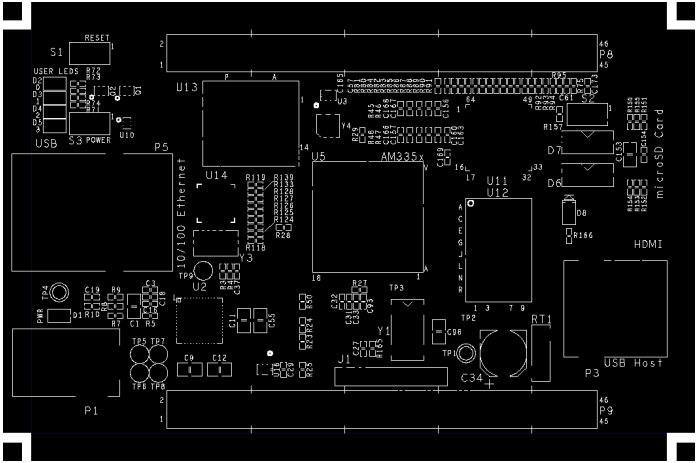
- Follow traces to determine connectivity/schematic
- Silkscreen can identify components, debug ports, probe points
- Ground/power planes identify leads on components

PCB construction

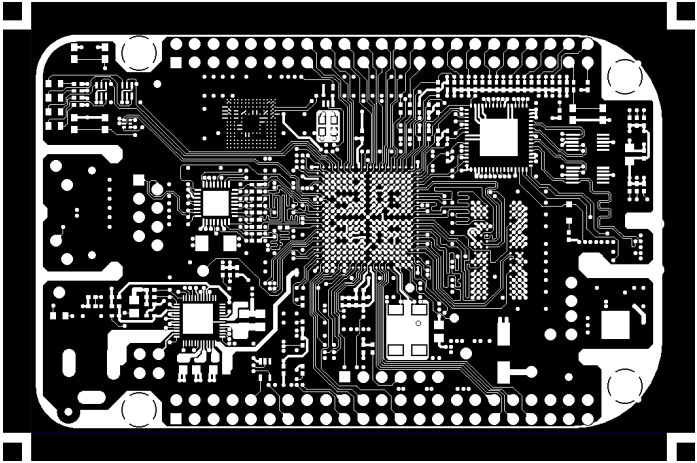
Beaglebone Black signal layers:



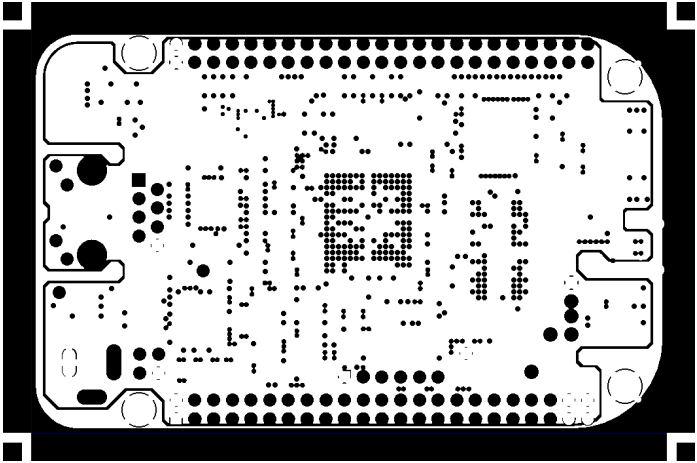
BBB: Top silk


RESET
S1
USER LEDS
S3 POWER
USB
P5
P8
P9
P1
P3
U13
U14
U5
AM335x
U11
U12
10/100 Ethernet
microSD Card
HDMI
USB Host
RT1
D7
D6
D8
U10
U2
Y1
Y3
Y4
C34

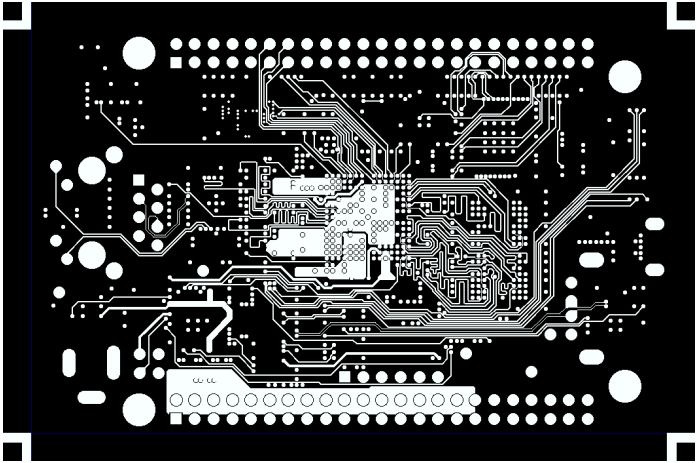
BBB: Layer 1



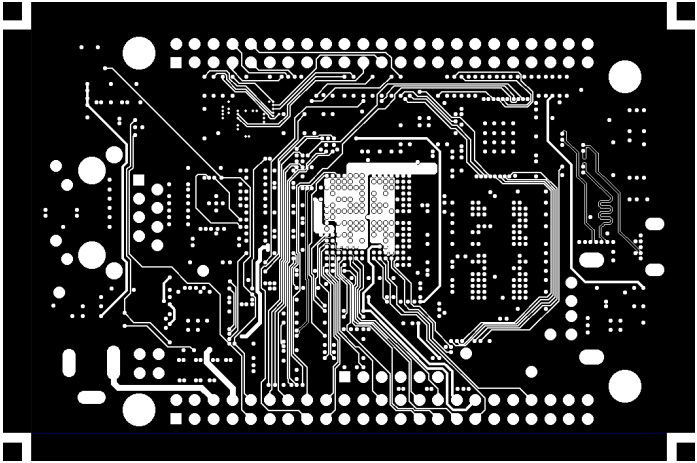
BBB: Layer 2



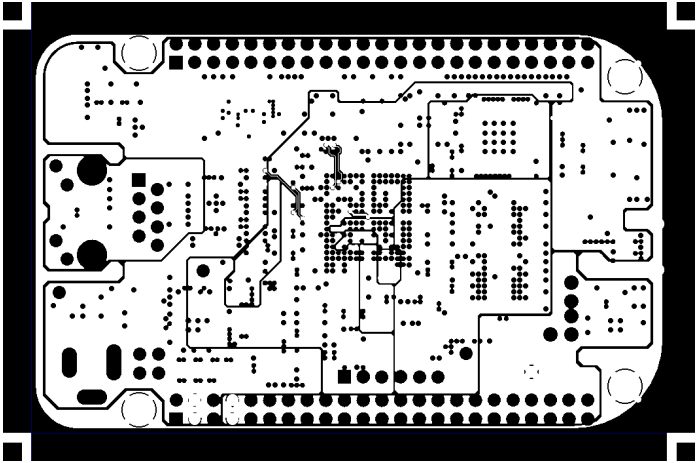
BBB: Layer 3



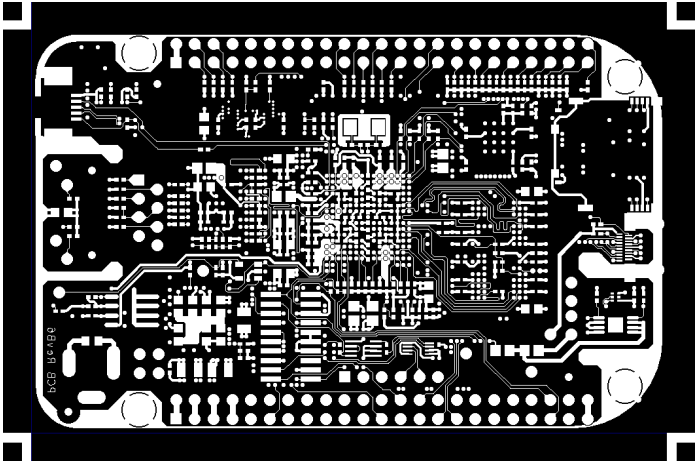
BBB: Layer 4



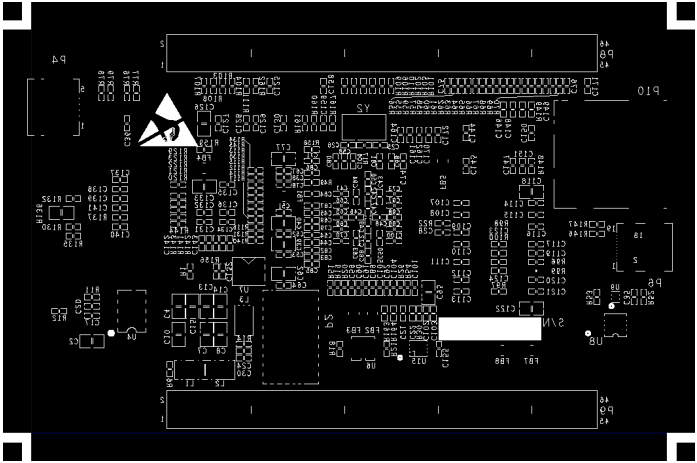
BBB: Layer 5



BBB: Layer 6



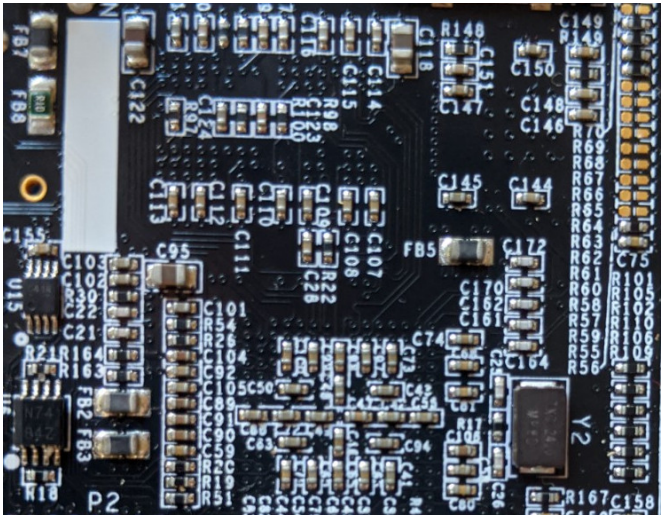
BBB: Bottom silk



Passive components

Frequently surface-mount on modern PCBs

- Resistors
- Capacitors
- Inductors
- Ferrite beads
- LEDs



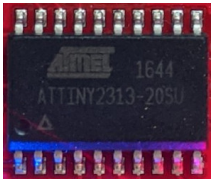
Integrated circuits

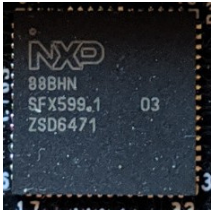
Sparkfun has a really good tutorial:
https://learn.sparkfun.com/tutorials/integrated-circuits/all

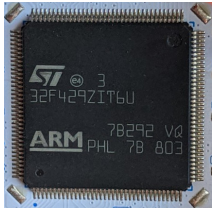
- Dual Inline Packages (DIPs) - Easy to clip to leads
- Surface Mount Devices (SMDs) - Harder to clip to leads
- Quad Flat Packages (QFDs) - Harder to clip to leads
- Quad-Flat No-leads (QFNs) - Very difficult to probe contacts
- Ball Grid Arrays (BGAs) - No way to probe leads

Package markings

- Identify manufacturer, part number, lot code, date code, etc.
- Manufacturer sometimes identified by logo
  - https://how-to.fandom.com/wiki/How_to_identify_integrated_circuit_(chip)_manufacturers_by_their_logos/all_logos
  - https://faq.tweakers.net/cme/iclogos
  - http://www.siliconinvestigations.com/logos/logos.htm
- Find corresponding data sheet to find what the component does

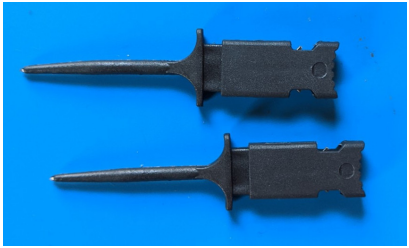


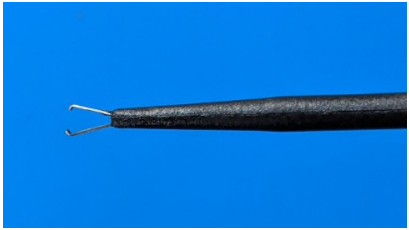




Interfacing to boards: grabbers

- Attach directly to leads of components
- Probe comparison: https://sigrok.org/wiki/Probe_comparison





Other interfacing options

- Populate unpopulated headers
- Add resistors or bridge pads to connect disconnected signals
- Solder fine-gauge wire directly to probe points or pads of unknown connectors
  - Secure wires for mechanical stability
- Remove programmable components to read them out with a programmer

Tracing signals

Following traces gives information about how components are electrically connected

- Can follow surface traces visually
- Probe with continuity mode of a multimeter
  - Difficulties tracing through components (resistors)
  - Difficulties tracing to BGA pads
  - Sometimes easier if you can completely depopulate a circuit board
- CT x-ray reconstruction: https://ieeexplore.ieee.org/document/7820117

Hardware RE

Entire research community dedicated to *hardware* reverse engineering

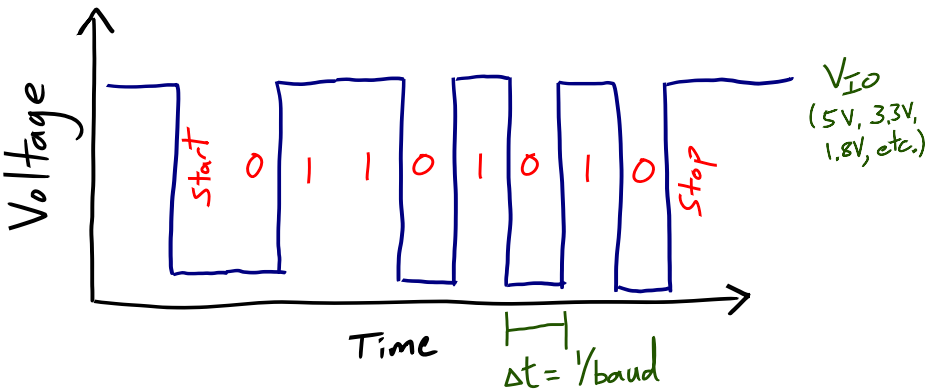
- Integrated circuit RE, hardware security mechanisms, etc.
- Discovering/activating protected or obfuscated debug ports
- Side channel analysis

This is outside the scope of this course–we'll stick to the basics.

Universal Asynchronous Receiver Transmitter (UART)

- Rx and Tx signal lines
- Data typically sent least significant bit (LSB) first
- Asynchronous interface (no shared clock)
  - Endpoints must have an agreed-upon signaling rate (baud rate)
  - Start/stop bits used for synchronization
- May include parity bits or multiple stop bits
- Sometimes includes flow control signals
  - Uncommon for debug applications

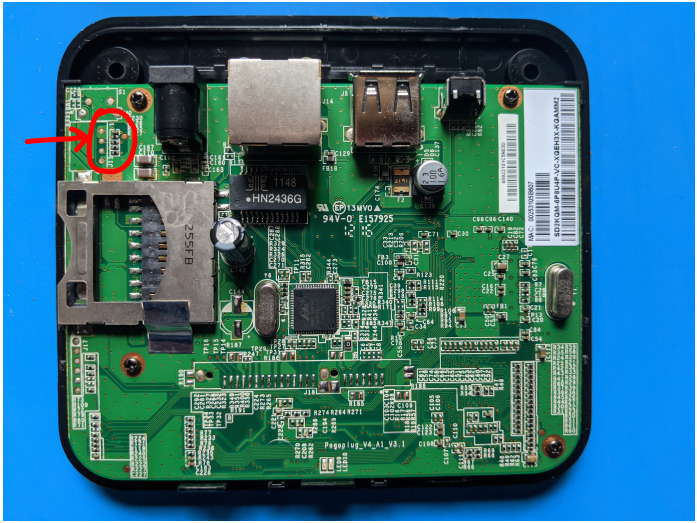
UART TTL signaling



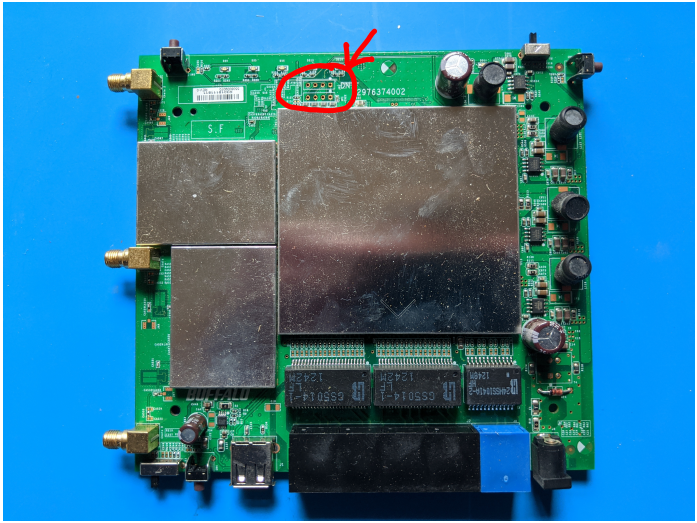
UART uses

- Often used for off-board debug or configuration communication
  - Serial console
  - Debug log
- Sometimes used for inter-system communication
- Sometimes communication between components on board
- Typical header includes ground pin, Rx pin, Tx pin, and often a power pin
  - Header might not be populated
  - UART signal line could also have non-populated series resistors
  - Output might be disabled in software
- Common I/O voltages: 5V, 3.3V, 1.8V, 1.2V
- Related to RS-232:
  - +/-3V to +/-25V signalling
  - Bits are inverted from TTL version

Find the UART: Pogo Plug



Find the UART: Wifi router

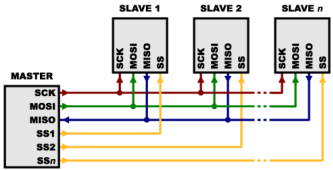


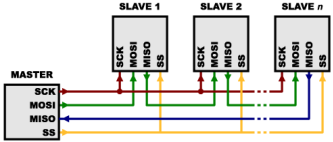
Tools for interacting with UART interfaces

- Multimeter
  - Identify power, ground pins
  - Determine I/O voltage
- Logic analyzer
  - Initial probing of signals
  - Determine baud rate
- USB-to-UART adapter
  - FTDI, Prolific common chipset vendors
  - *Pick one with the correct I/O voltage*
- Assorted hardware hacker tools
  - e.g. Bus Pirate
- Use a microcontroller and make your own!

Serial Peripheral Interface (SPI)

- Signals:
  - Chip Select (CS) or Slave Select (SS)
  - Master out slave in (MOSI or SI)
  - Master in slave out (MISO or SO)
  - Clock (SCK)
- Multiple endpoints on one bus
- TTL voltage levels
- No standard, some variability
  - Clock polarity
  - Phase of data relative to clock
- Proprietary extensions
  - Dual- or quad-SPI modes





Images from sparkfun.com

Sparkfun has a great tutorial on this:
https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all

SPI signaling

Transaction with single byte on MOSI followed by a single byte on MISO, LSB first:

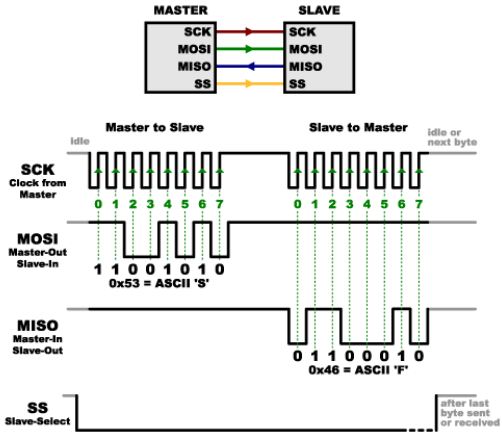
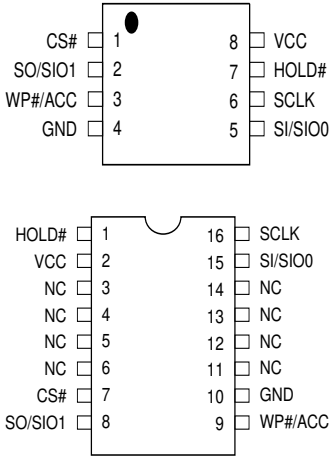


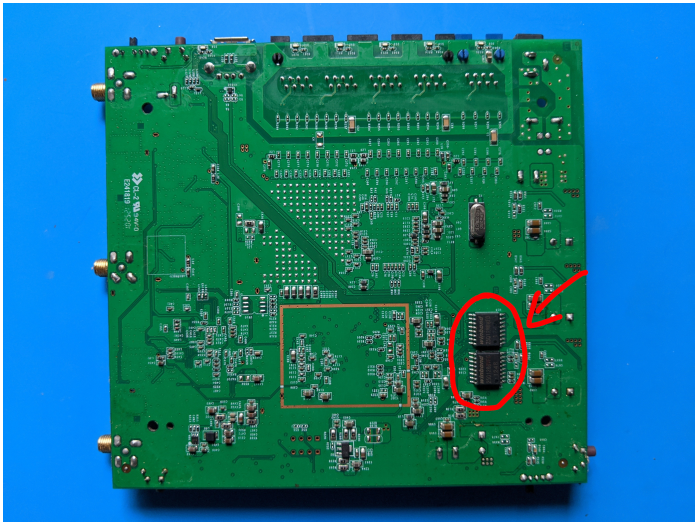
Image from sparkfun.com

SPI uses

- Can be used for many peripherals
- Very frequently used for flash memories
  - Entire firmware image in many embedded systems
  - Low-level boot code in more complex systems
    - ★ e.g. UEFI in a general-purpose computer
  - Very often SO-8 footprints, sometimes SO-16, sometimes others
- Rare to have an SPI header
  - Memories often programmed before being soldered in place



Find the SPI flash memory: Wifi router



Tools for interacting with SPI interfaces

- Logic Analyzer
  - Initial probing of signals
  - Reconstruction using protocol analyzer
- Assorted hardware hacker tools
  - e.g. Bus Pirate
- Protocol analyzers/Host adapters
  - e.g. Total Phase Beagle/Aardvark
- Memory readers
- Use a microcontroller and make your own!

Inter-Integrated Circuit bus (I2C)

- Also known as IIC, TWI (two-wire interface)
  - Related to other buses like SMBus
- Multi-master multi-slave bus
- Only SDA (data) and SCL (clock) pins required
- TTL voltage levels
- Defined messaging protocol to allow shared bus
- Addresses for slave devices pre-programmed, programmable, or set with pin states at start-up

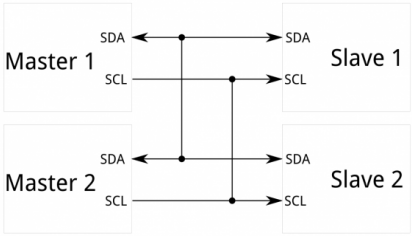


Image from sparkfun.com

Sparkfun has a great tutorial on this: https://learn.sparkfun.com/tutorials/i2c

I2C signaling

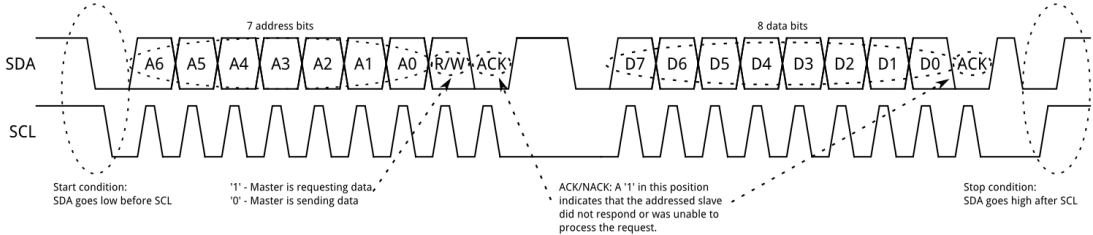
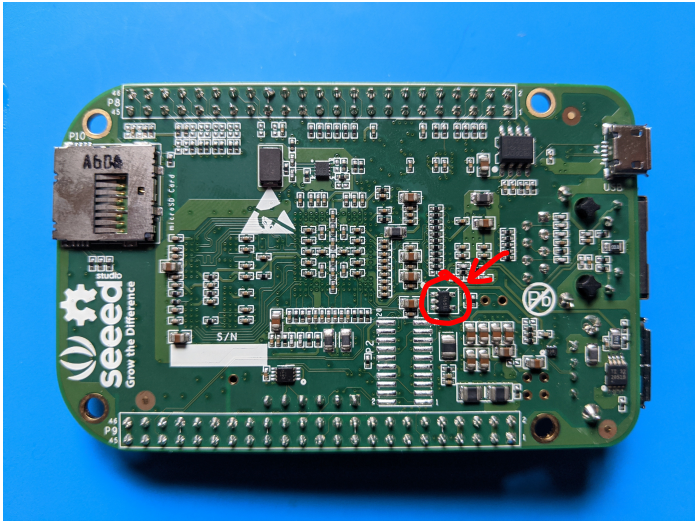


Image from sparkfun.com

I2C uses

- Slow-speed communication
  - Sensors
  - Peripheral devices with low bandwidth needs
- Low-capacity memories
  - Configuration information
  - Read infrequently

Find the I2C flash memory: Beaglebone Green



Beaglebone Green I2C memory contents

```
user@beaglebone:/sys/bus/i2c/devices/0-0050$ hexdump -C eeprom
00000000  aa 55 33 ee 41 33 33 35  42 4e 4c 54 42 42 47 31  |.U3.A335BNLTBBG1|
00000010  42 42 47 32 31 39 30 34  30 38 36 37 ff ff ff ff  |BBG219040867....|
00000020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
00001000  aa 55 33 ee 41 33 33 35  42 4e 4c 54 42 42 47 31  |.U3.A335BNLTBBG1|
00001010  42 42 47 32 31 39 30 34  30 38 36 37 ff ff ff ff  |BBG219040867....|
00001020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
00002000  aa 55 33 ee 41 33 33 35  42 4e 4c 54 42 42 47 31  |.U3.A335BNLTBBG1|
00002010  42 42 47 32 31 39 30 34  30 38 36 37 ff ff ff ff  |BBG219040867....|
00002020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
00003000  aa 55 33 ee 41 33 33 35  42 4e 4c 54 42 42 47 31  |.U3.A335BNLTBBG1|
00003010  42 42 47 32 31 39 30 34  30 38 36 37 ff ff ff ff  |BBG219040867....|
00003020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
00004000  aa 55 33 ee 41 33 33 35  42 4e 4c 54 42 42 47 31  |.U3.A335BNLTBBG1|
00004010  42 42 47 32 31 39 30 34  30 38 36 37 ff ff ff ff  |BBG219040867....|
00004020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
00005000  aa 55 33 ee 41 33 33 35  42 4e 4c 54 42 42 47 31  |.U3.A335BNLTBBG1|
00005010  42 42 47 32 31 39 30 34  30 38 36 37 ff ff ff ff  |BBG219040867....|
00005020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
00006000  aa 55 33 ee 41 33 33 35  42 4e 4c 54 42 42 47 31  |.U3.A335BNLTBBG1|
00006010  42 42 47 32 31 39 30 34  30 38 36 37 ff ff ff ff  |BBG219040867....|
00006020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
00007000  aa 55 33 ee 41 33 33 35  42 4e 4c 54 42 42 47 31  |.U3.A335BNLTBBG1|
00007010  42 42 47 32 31 39 30 34  30 38 36 37 ff ff ff ff  |BBG219040867....|
00007020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
00008000
```

I2C memory only holds device type and serial number

Tools for interacting with I2C interfaces

- Logic Analyzer
  - Initial probing of signals
  - Reconstruction using protocol analyzer
- Assorted hardware hacker tools
  - e.g. Bus Pirate
- Protocol analyzers/Host adapters
  - e.g. Total Phase Beagle/Aardvark
- Memory readers
- Use a microcontroller and make your own!

A note on logic analyzers

$10 logic analyzer

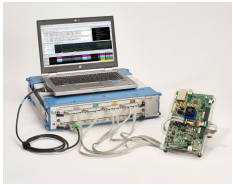
- 8 inputs
- 24 MHz sampling rate
- -0.5V to 5.25V input
- No input protection
- Rudimentary triggering
- Open-source software (firmware/sigrok/pulseview)

$1000 logic analyzer

- 16 inputs
- 500 MHz sampling rate
- -10V to 10V input
- Overvoltage protection to +/- 25V
- Rudimentary triggering
- Free, closed-source application

$100000 logic analyzer

- 136 inputs
- 10 GHz sampling rate
- -5V to 5V threshold
- Input protection?
- Sophisticated triggering
- Closed-source application; features cost $$$

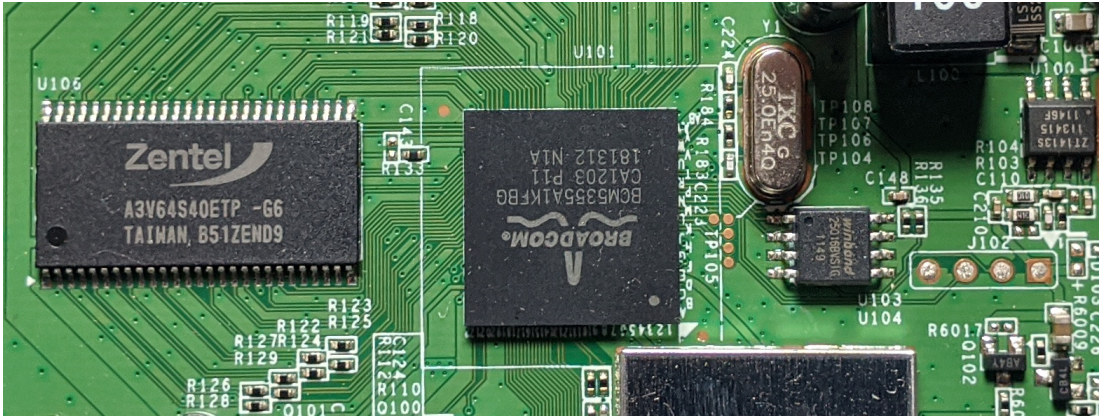


Firmware extraction from a Wifi router

- Target system: Belkin F5D7234-4 V5 wireless router
- Goals: Extract firmware
- Approach:
  - Identify debug/communications interfaces on board
  - Interact to extract firmware contents

System board

System board closeup



UART interaction

- Probe pads with multimeter to determine likely pinout
- Solder header
- Connect logic analyzer
  - Determine Tx pin
  - Determine baud rate – 115200
- Connect TTL UART-to-USB cable
- Serial terminal session

Extract memory through bootloader

Prepare connected host:

- `apt-get install tftpd-hpa`
- `touch /srv/tftp/mem`
- `chmod 666 /srv/tftp/mem`
- Set IP address to 192.168.2.2/24

Upload firmware:

- <space> to interrupt bootloader
- `save 192.168.2.2:mem 0xbfc00000 0x200000`

SPI flash removal and readout

- Identify device
- De-solder from PCB
  - Bismuth solder alloy
- Solder to breakout board
- Connect with USB to MPSSE Serial cable (FTDI C232HM-DDHSL-0)
- Read out with flashrom:

```
flashrom -p ft2232_spi:type=232H -r output.bin
```

SPI traffic capture

- Connect logic analyzer probes to SO-8 SPI flash memory leads
- Record traffic during startup with logic analyzer
- Apply SPI protocol analyzer
- Save output
- Parse and reconstruct the memory contents

Introduction to Assignment 4