Hardware Reverse Engineering

What is it?

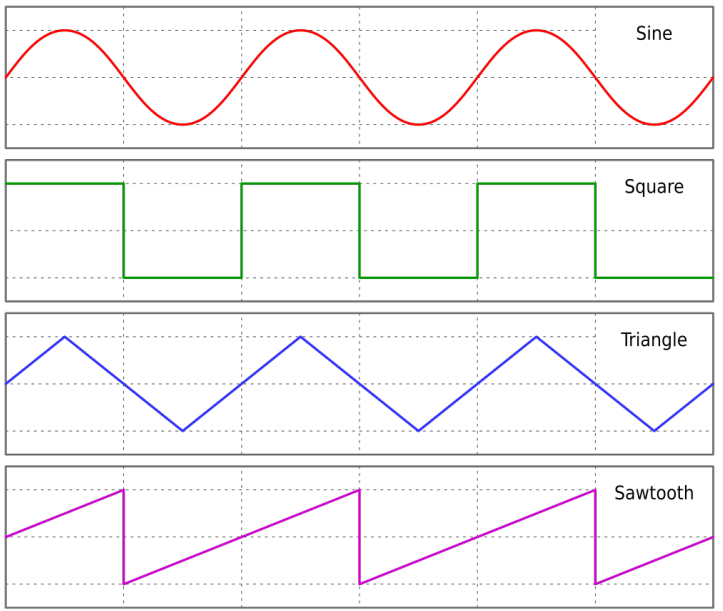
- This style of Reverse Engineering is focused on the Hardware and Firmware of a device(typically embedded) through which more traditional means of investigation are impractical
- This can be one of the most fun or the most frustrating disciplines as it relies on a knowledge of
  - Physics
  - Electrical Engineering
  - Computer Engineering
  - Chemistry
  - And a more…
- Overall, you either love it or hate it but it’s a crucial skill nonetheless

First Steps

- If we bring it back to the first session of the series you'll remember part of the theory of computing, ni which it describes the basis of most modern computers, this is where we put it into practice
- Practically, this helps us Identify sections of PCBs that are of interest to us!
- Not in order of importance we need to identify certain sections
  - CPU
  - Non/Volatile Memory
  - Power Management Circuits
  - Flash Memory
  - Factory Testing Areas
  - Physical Control Circuts e.g, Serial, JTAG, SWD ports and interfaces
- Give it a shot!!

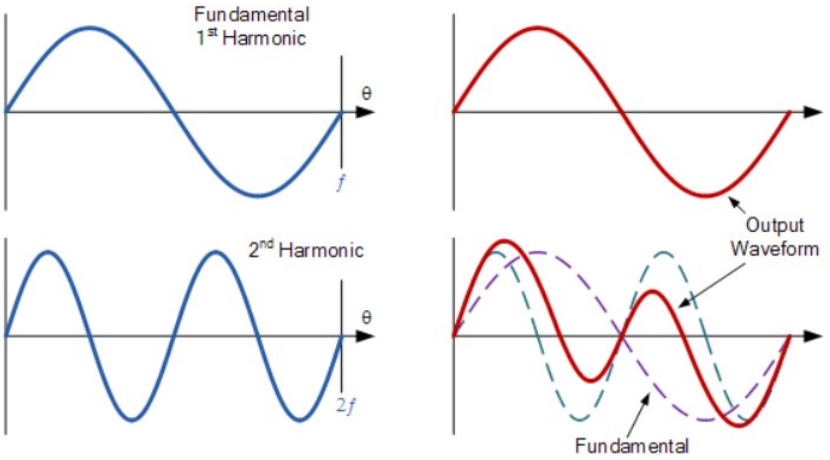
Signal Primer

- It’s common to hear people talk about “Digital vs. Analog” for our purposes we’re going to be looking a digal signals but it’s important to understand that although they may not be mutually exclusive they are inexorably linked
- Barring the Physics related questions there are some typical waveform type’s you’ll see
  - Sine
  - Sawtooth
  - Square
  - Triangle
- Concept – Complex Waveform
- Understanding Voltage Levels



Complex Waveform

- Complex Waveforms are effectively the outcome of natural harmonics of a frequency(yes including electrons)
- In which a given waveform's(Fundamental's) waveform is superimposed by a "harmonic" waveform which can be defined as having either a periodic/frequency related aspect which contributes to this superpostion

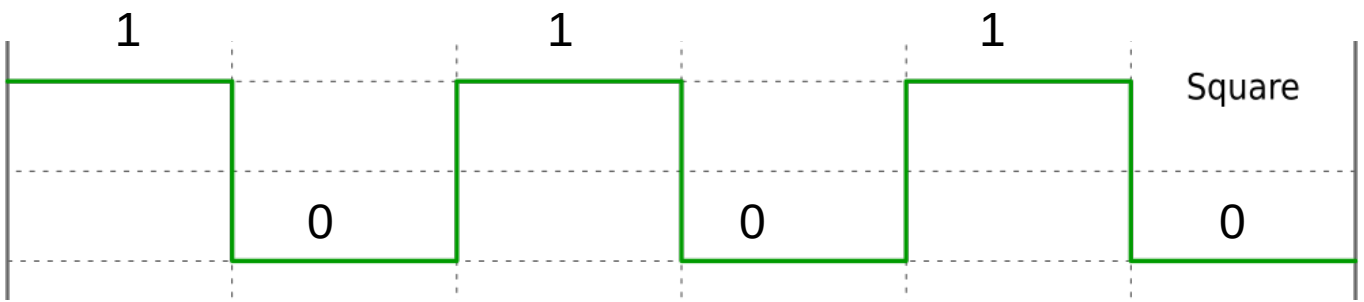


If *f* = frequency of the fundamental(ex. 60hz)

Then the 2nd harmonic is *"2f"* or 60 * 2 = 120Hz

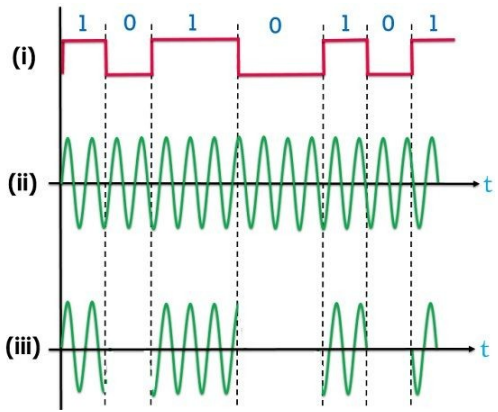
Digital Waves

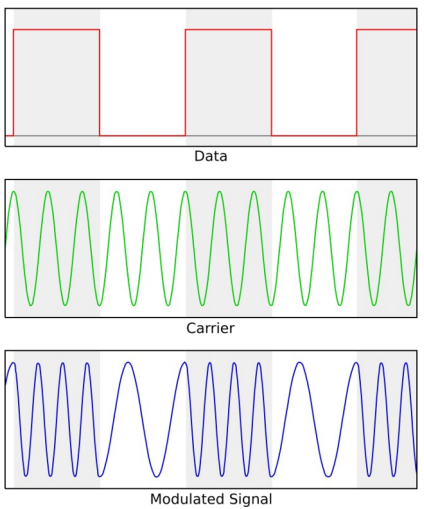
- Effectively, most digital electronics use Square Waves to transmit Binary information over a wire as these waves are excellent at portraying binary values
- As the Trough of the Wave can be understood as Zero the Peak can be it’s 1
- In reality, since Voltage cannot be absolute and can Fluctuate we assign ranges of voltage to be interpreted as either 1 or 0



Some Digital Terminology

- Digital Low /Pulled Low
- Digital High / Pulled High
- FSK/ASK





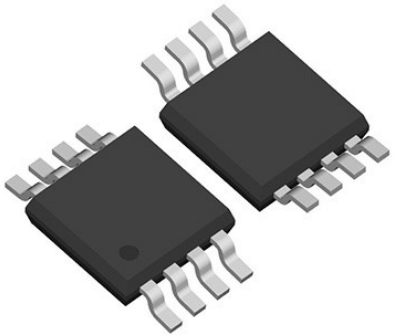
Points of Interest

- Points of Interest on the PCB typically will become the areas where Crucial information or function can be manipulated to your benefit
- A Useful point is locating the Flash memory where the bootloader and other early resources are stored, as these are not typically provided from manufacturer firmware files(typically just a filesystem and sometimes a kernel)
- These are typically stored on a separate chip rather than being stored in more complex NV memory such as NOR/NAND Flash memory as with many forms of embedded linux the bootloader will inhabit a different partion(even if on the same disk)
- From experience the majority of these chips are SPI(**S**erial **P**eripheral **I**nterface) flash chips with a SOIC footprint

Small **O**utline **I**ntegrated **C**ircut

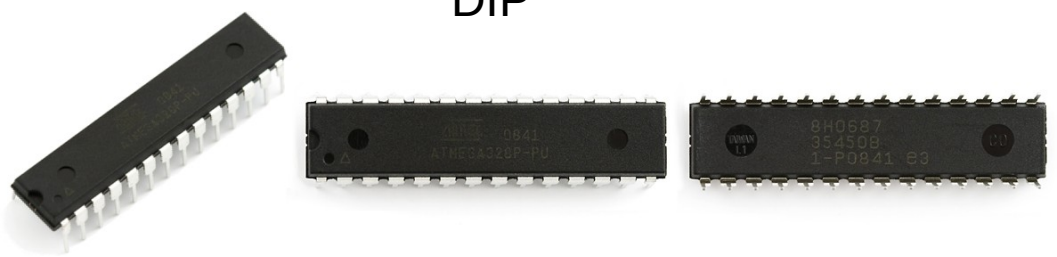
Points of Interest

- Two other common packages you'll come across is the
  - DIP(**D**ual **I**nline **P**ackage)
  - MSOP(**M**ini **S**mall-**O**utline **P**ackage)



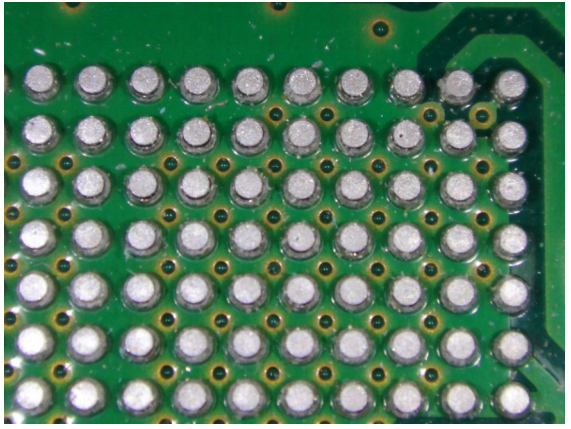
MSOP

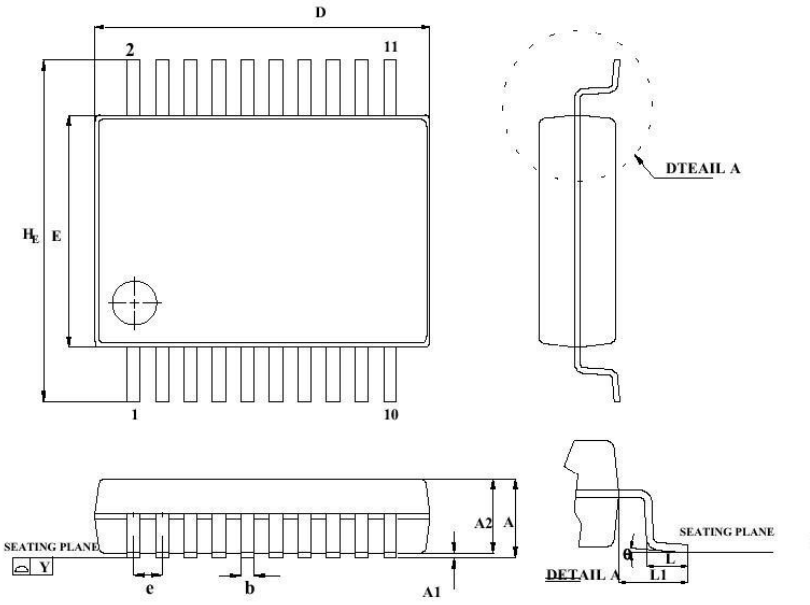
DIP



Points of interest

- Other Areas are the actual “disk” where information is stored as many of these chips are NOR/NAND flash most of time with either TSOP or BGA packages
- They are INCREDIBLY difficult to remove with other the proper equipment so realistically if they cannot be read or sniffed **in-situ**(Also very difficult), try a different easier approach



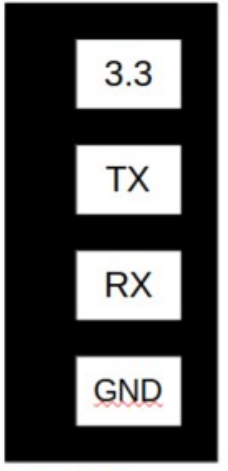


Points of interest

- One of the easiest to use of all the components is the actual physical control circuts as
  a) Often are Through hole solderable
  b) If not then the Open Pads are fluxed and read to solder
  c) If you’re very lucky the manufactures may have just never taken the pins off from the KiCAD file
- In order we’ll go through Three common control types used in embedded devices
  - UART(**U**niversal **A**synchronus **R**eciver **T**ransmitter)
  - JTAG(**J**oint **T**est **A**ction **G**roup)
  - SWD(**S**erial **W**ire **D**ebug)

UART

- A UART terminal is a fundamental interface for not just embedded device but for most computers as it allows a person with physical access to interact with a console on the device with having to use GUI’s
- Typically used for debugging interfaces, UART(In This Context) Contains a
  - 3.3V Pin(Labeled VCC typically)
  - TX(Transmit)
  - RX(Receive)
  - GND/DGND
- A UART Term. Can function with up to only 2 of these GND and TX/RX
- When configuring these ports use the <baud>/<bit><n><error bit>(ex 115200/8n1) for example
  - Baudrate
  - Data Bit
  - Error Bit



JTAG

- JTAG is significantly more complex than UART, and typically has a wide variety of header styles to inyferace with and is often based on the Manufactures of the Chip and Board
- JTAG while incredibly varied can be viewed as a type of OCD in which you can use the console to interact with registers, memory, and objects provided to the debugger
- At the VERY least a JTAG port will have 4 pins
  - TDI
  - TDO
  - TCLK
  - TMS
- Almost all modern devices have a Type of JTAG interface but for lack of better term it has become a basal part of it’s ‘biology’

SWD

- An alternative interface to JTAG optimized for use in ARM processor environment is built to be low footprint as opposed to massive JTAG ports(see Ammotech Port list)
- At MAX SWD only has 2 pins in the standard
  - SWDIO
  - SWCLK
- While very similar in functionality to the JTAG port a SWD(DAP) port accesses the processor and memory function in a similar manner in which a typically debugger might just at a hardware level

Chip Protocols

- Although there are many different protocol used to control Chips, when it comes to dumping chips a familiarity with some of the more common protocols is useful
- Many Earlier Memory chips were simply Address and Data I/O, Feed the chip an Address and a number and it Reads out the data of the passed length at the address
- But during the 1980s, when semiconductors were reaching a more commercial level and memory density was steady increasing
- Many new Protocols were developed
  - SPI(Serial Peripheral Interface)
  - i2c(Inter-Integrated Circut)

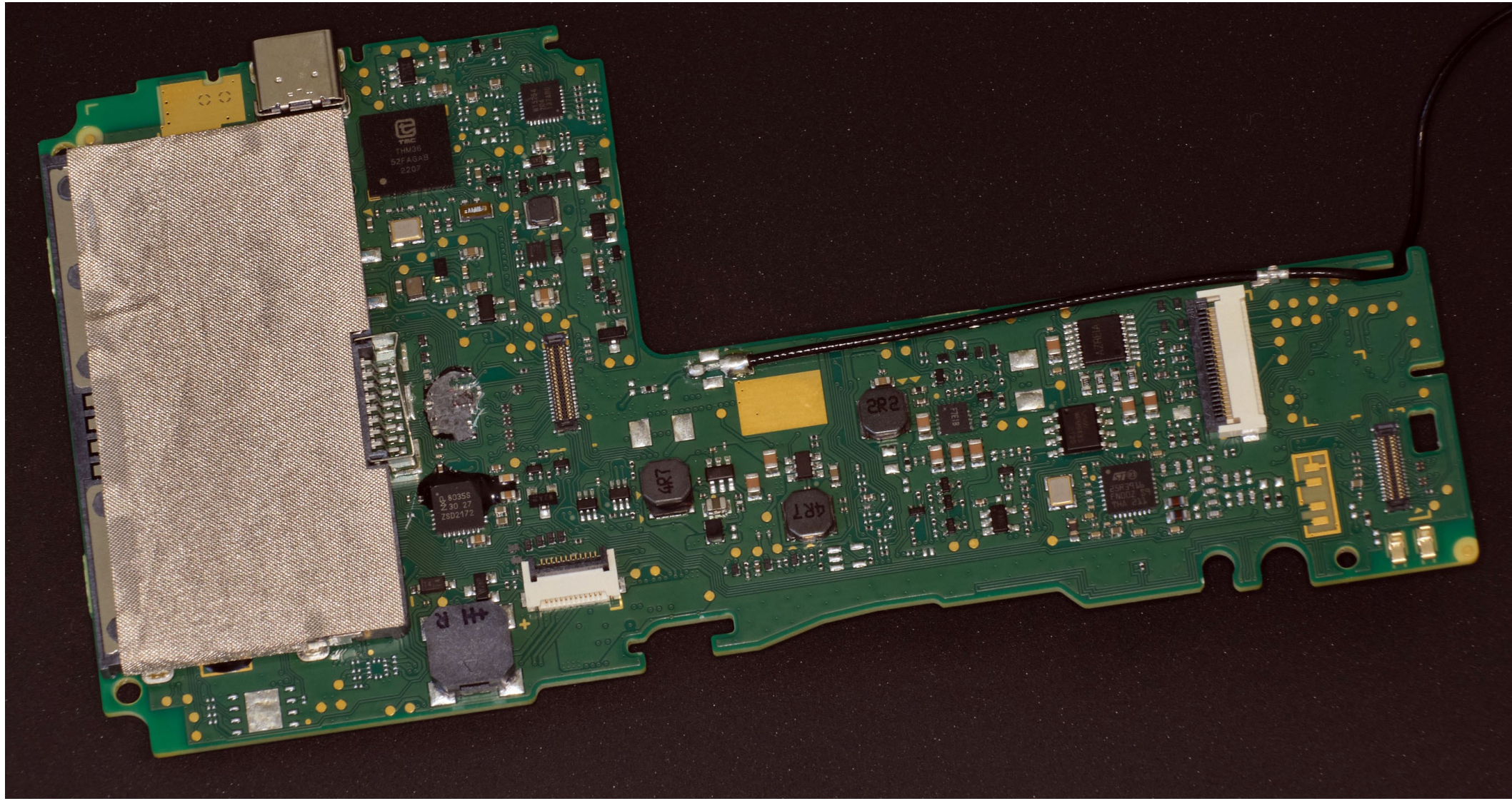
SPI

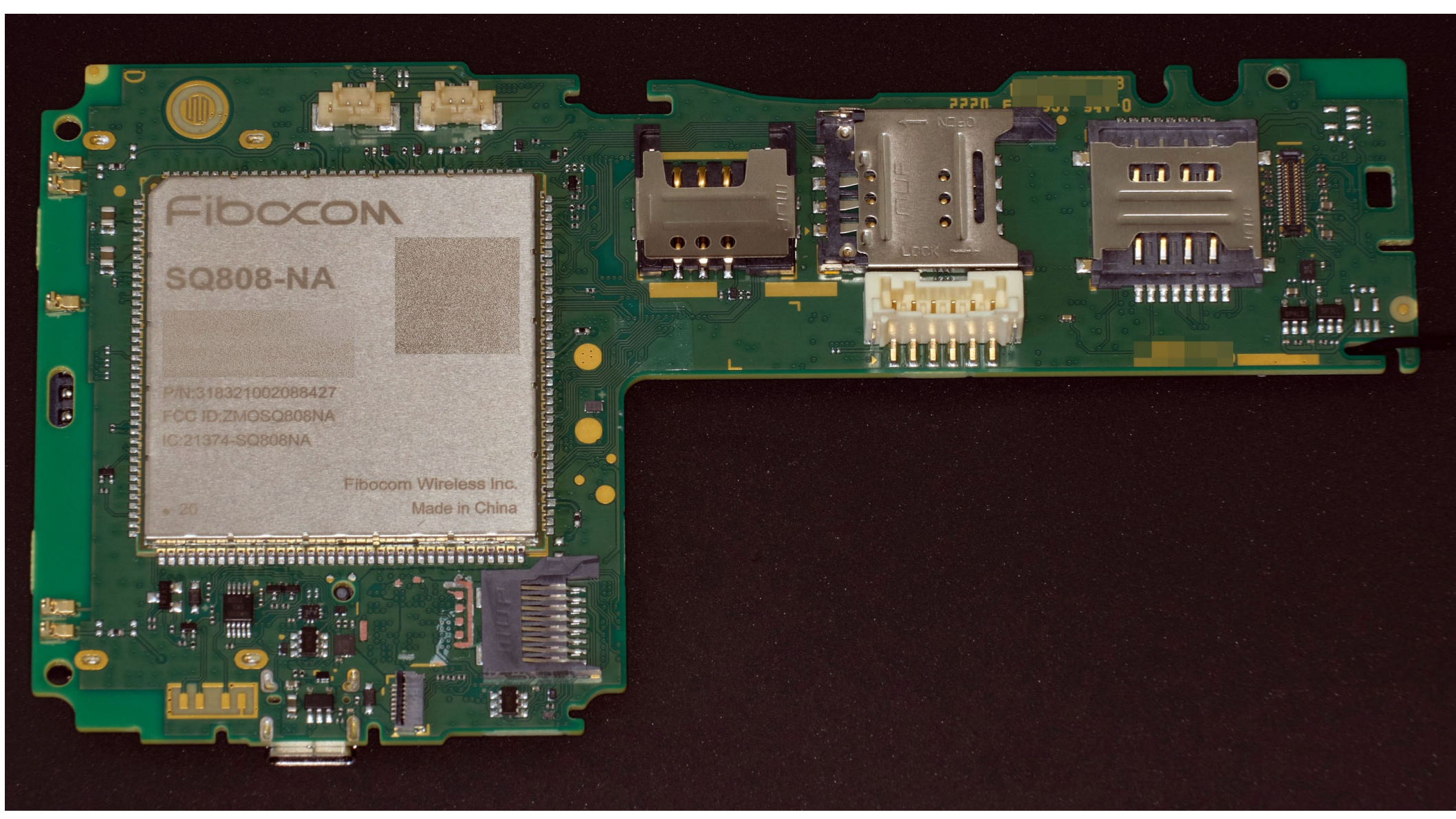
- SPI is used for more than just for memory often SPI is used to control peripheral devices and on many devices is often used by the kernel for low level Non-RAM/NVM processes(bootloaders, ROMS, Environment Settings) in which a ROM is treated as a type of RO file system
- SPI is Standardized Widely and is used as a basis for many memory chips but almost always uses at least 4 pins on()
  - IOX
  - IOX
  - CS
  - SCLK
- SPI-Modes 0-4; CPOL/CPHA
- Quad-SPI

SPI mode	Clock polarity (CPOL)	Clock phase (CPHA)	Data is shifted out on	Data is sampled on
0	0	0	falling SCLK, and when $\overline{SS}$ activates	rising SCLK
1	0	1	rising SCLK	falling SCLK
2	1	0	rising SCLK, and when $\overline{SS}$ activates	falling SCLK
3	1	1	falling SCLK	rising SCLK

I2C

- I2C is mainly used for peripheral at lower speeds(>5Mb)
- It uses a 7-bit address space(Up to 10) for addressing in which a
  - Device is designated a Address at which it can be accessed
  - There many different ”commands” can be used to interact with the device given the address
  - As well as an array of signals used for transmission
- Understand that i2C mostly is used as a configuration interface, Unless it’s explicitly being used for memory interfaces such as EEPROM
- Just understand that for memory most chip will use faster protocols


THM36
52FAGAB
2207
2R2
4R7
4R7


Fibocom
SQ808-NA
P/N:318321002088427
FCC ID:ZMOSQ808NA
IC:21374-SQ808NA
Fibocom Wireless Inc.
Made in China
OPEN
LOCK