I'm sorry I rooted your Smart House, I wish it was mine

A modular anthology guide to go from zero to hardware hacking hero (or at least learn how to pop open a shell on your Smart Fridge)

Nika Korchok

Part 0

whoami
Nika Korchok

- Tufts University/SMFA dual-degree undergraduate program, graduated in 2019 with Bachelor of Science and Bachelor of Fine Arts
  - Computer Science, French, Fine Arts
- Security Researcher/Reverse Engineer
- Artist
- ~* h4x0r *~
- ig: @nikaroxanne
- Twitter: @v1kt0r_frnknstn
- website: https://nikaroxanne.github.io/

Agenda

Part 1

Part 2

Part 3

Agenda

Part 1: Essential Info for Hardware Hacking

Part 1:

1. An introduction, including definitions of important terms
2. Methodology and motivations for reverse engineering embedded devices
3. An overview of electrical engineering fundamentals necessary to understand serial communication on an embedded device.
4. An overview of serial communication protocols

Agenda

Part 2: The Starving Artist’s Guide to Building a Hardware Hacking Lab

Part 2:

Creating your hardware hacking lab and reverse engineering toolkit

Agenda
Part 3: Let’s break some stuff!

- Part 3: Proof of Concept: Foscam surveillance camera
- An overview of hardware hacking techniques
  a. Definitions and illustrations of hardware
  b. Identifying test points
  c. Soldering
  d. Dumping contents of serial flash chips
  e. Patching contents of serial flash chips
- Conclusion
- References

Motivations

Why learn about hardware hacking and embedded device security?

- Having the security of your organization compromised because no one patched the firmware on the Smart Fridge in the break room is not a cute look!!

Motivations

Why learn about hardware hacking and embedded device security?

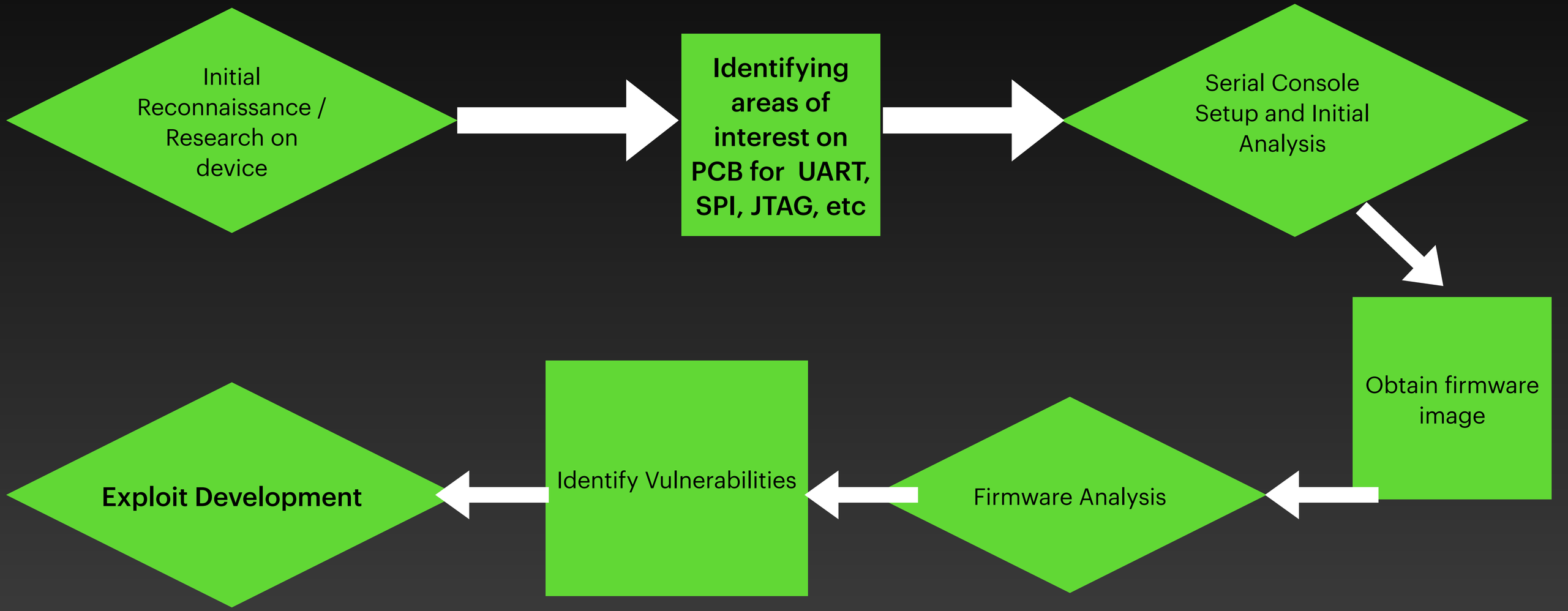
- Having the security of your organization compromised because no one patched the firmware on the Smart Fridge in the break room is not a cute look!!
- Learn how to create Mr. Robot-esque art installations using an IoT art botnet

Reverse Engineering Methodology

General Overview

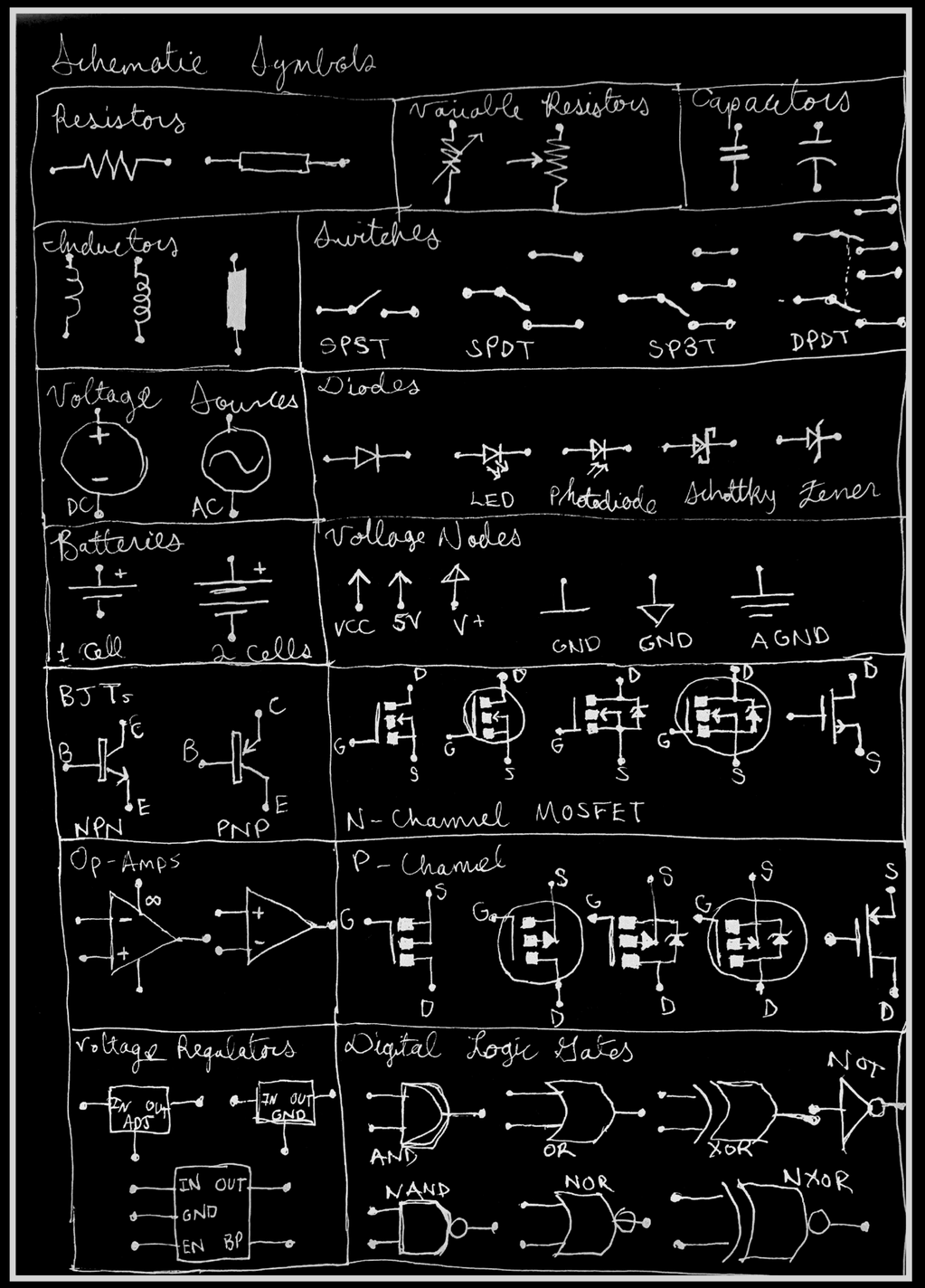
1. Reconnaissance / research — reading data sheets for the device, as well as data sheets for specific hardware components on the device PCB
    1. Compile relevant related research on the device, including current CVEs, research papers or talks
2. Identification of access points for the device, including JTAG, SPI, and UART
3. Serial console setup and analysis
4. Obtain firmware image
5. Static and dynamic analysis of firmware image
6. Identification of vulnerabilities on the device
7. Exploit development

Reverse Engineering Methodology
(but make it Euclidean)

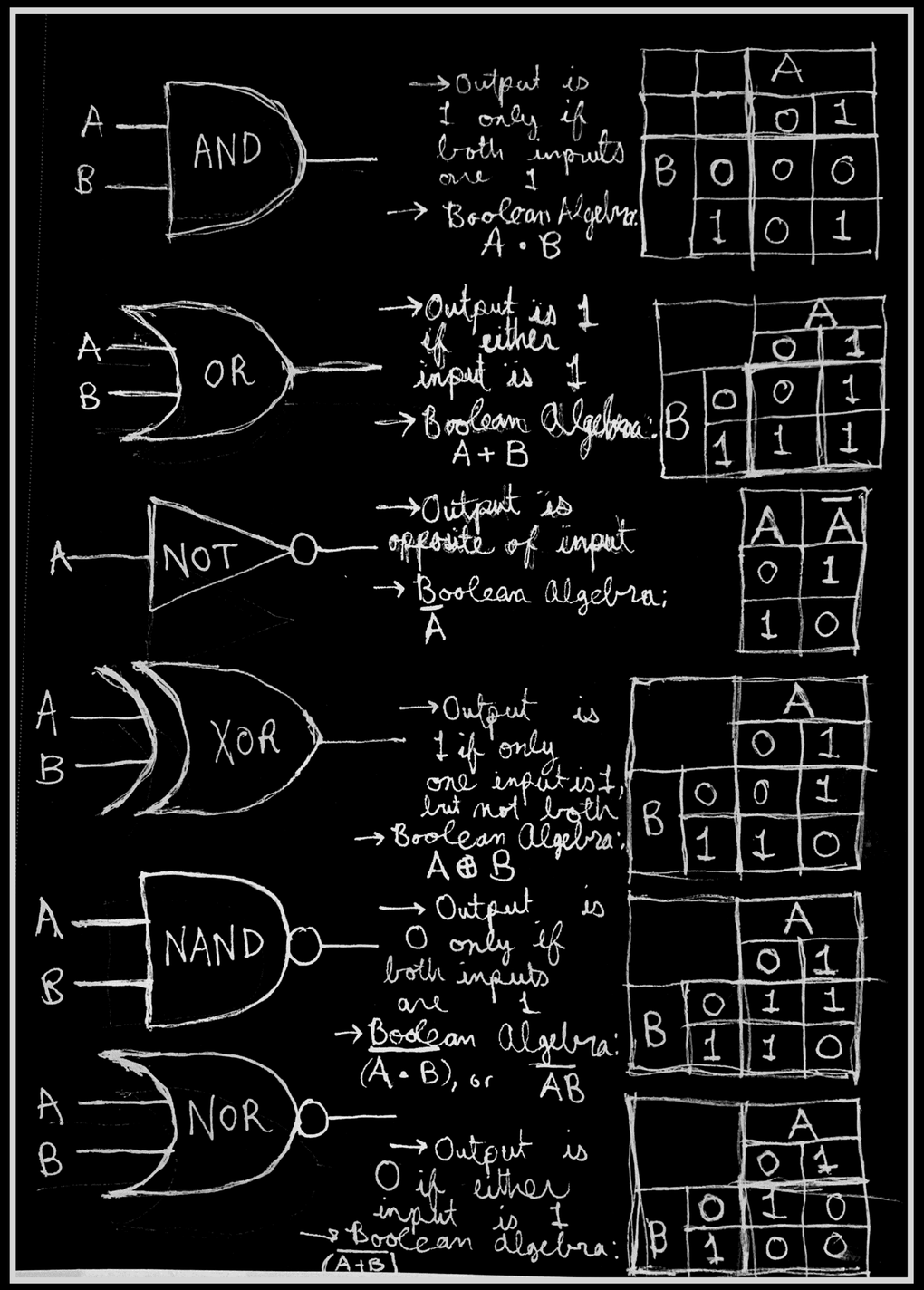


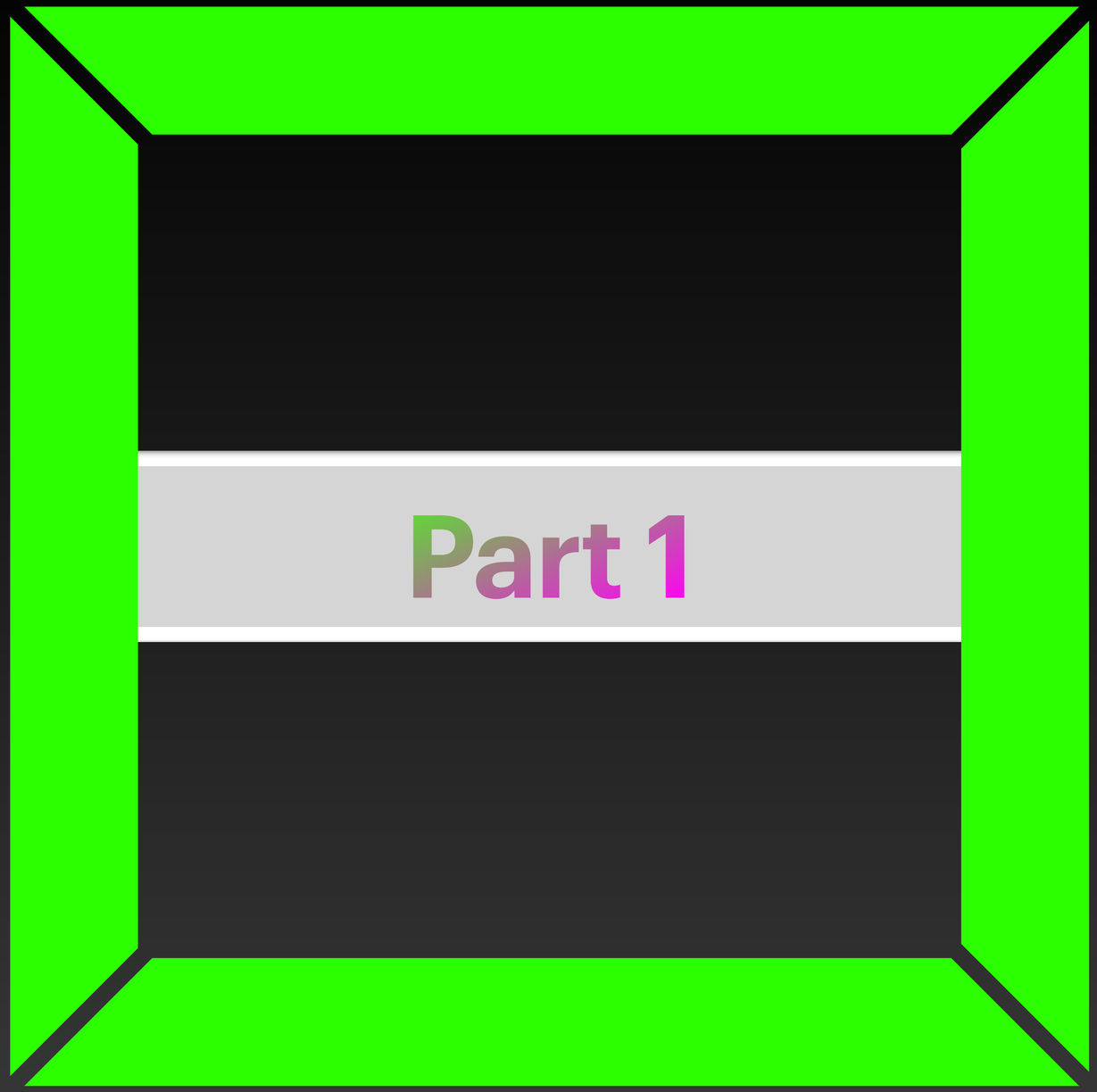

Presentation Format

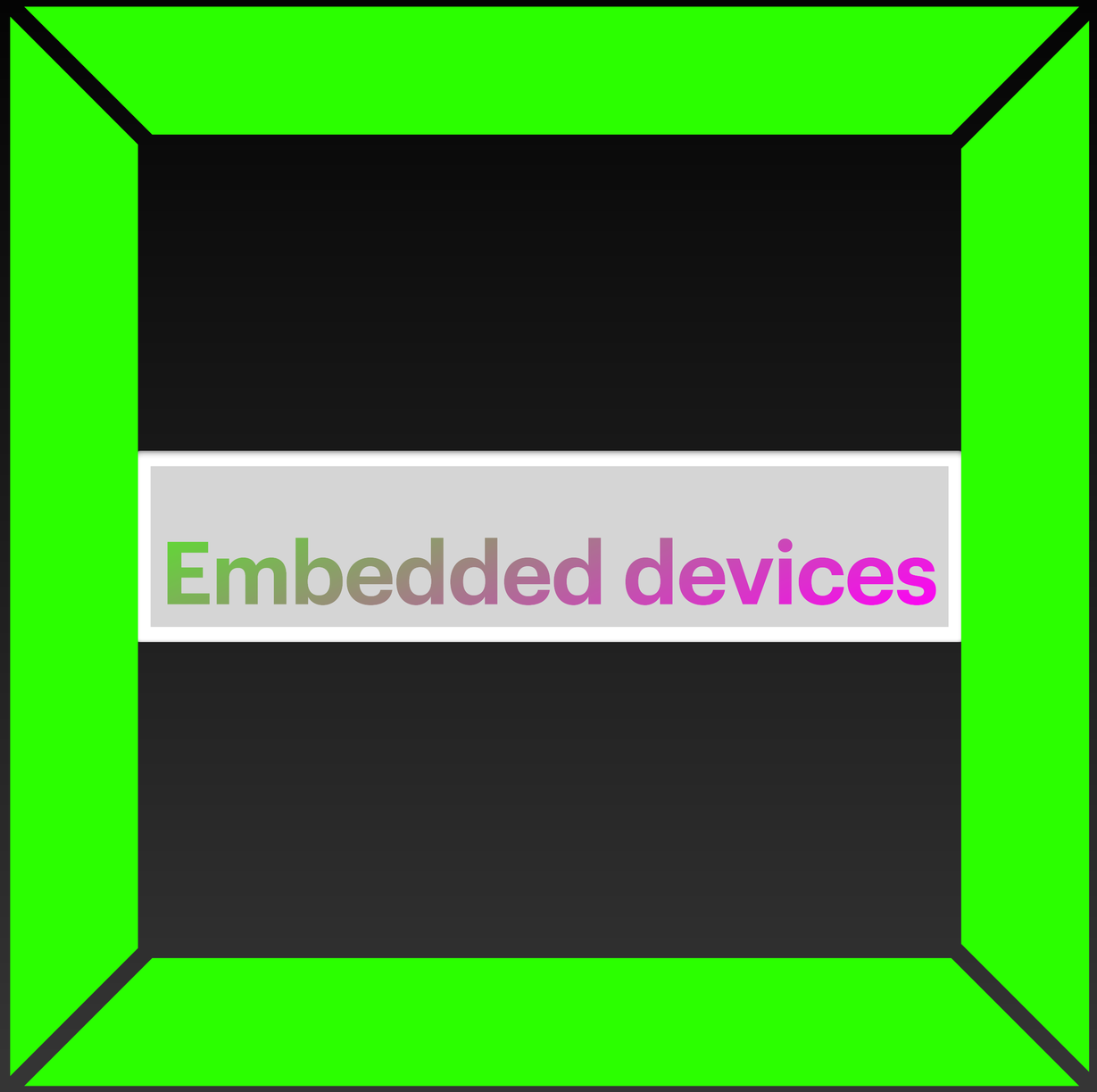
Circuit Schematic Symbols: A Rosetta Stone



Circuit Schematic Continued: Logic Gates

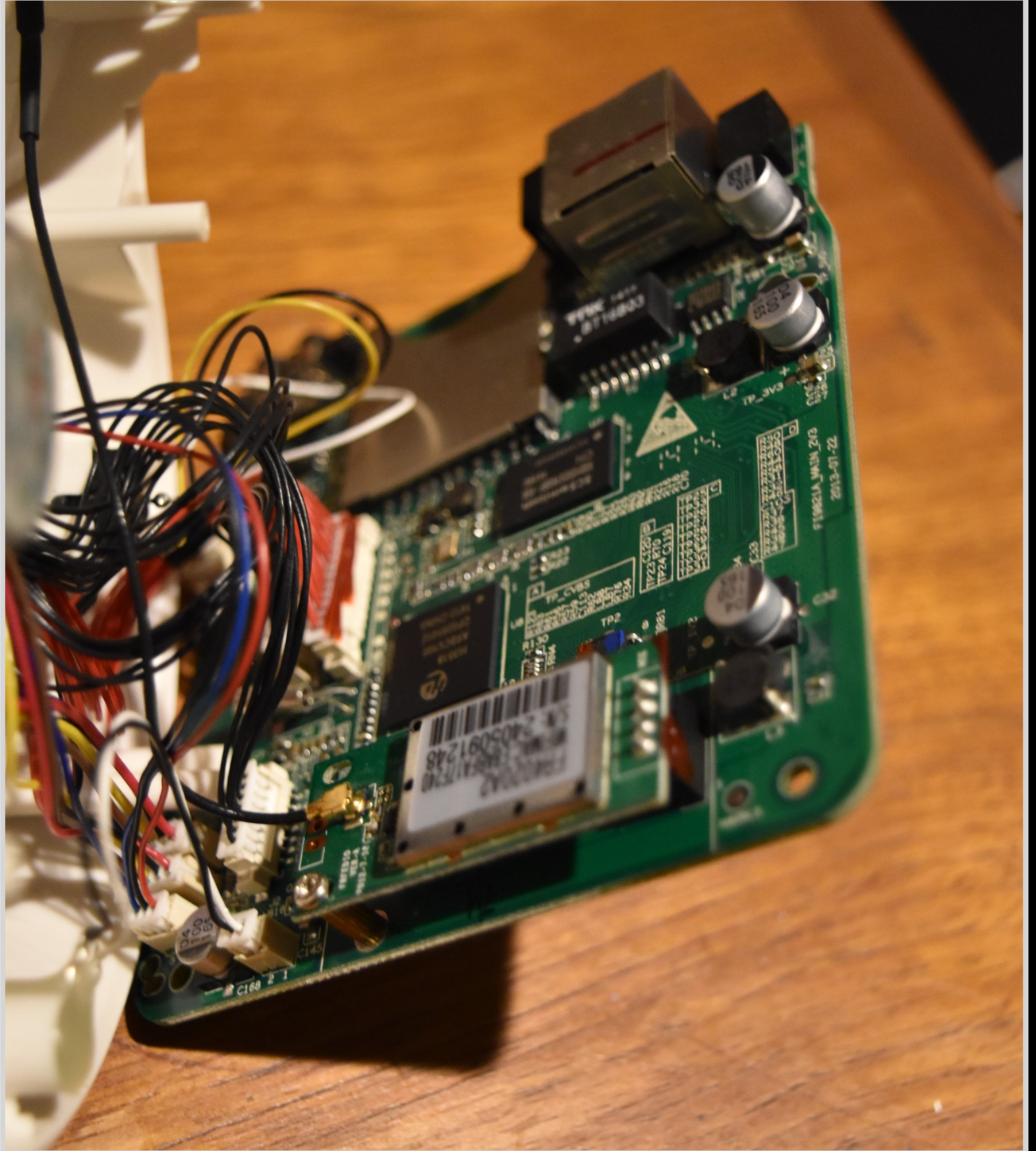



Part 1


Embedded devices

Embedded device

A computer, in miniature



Foscam F19821W

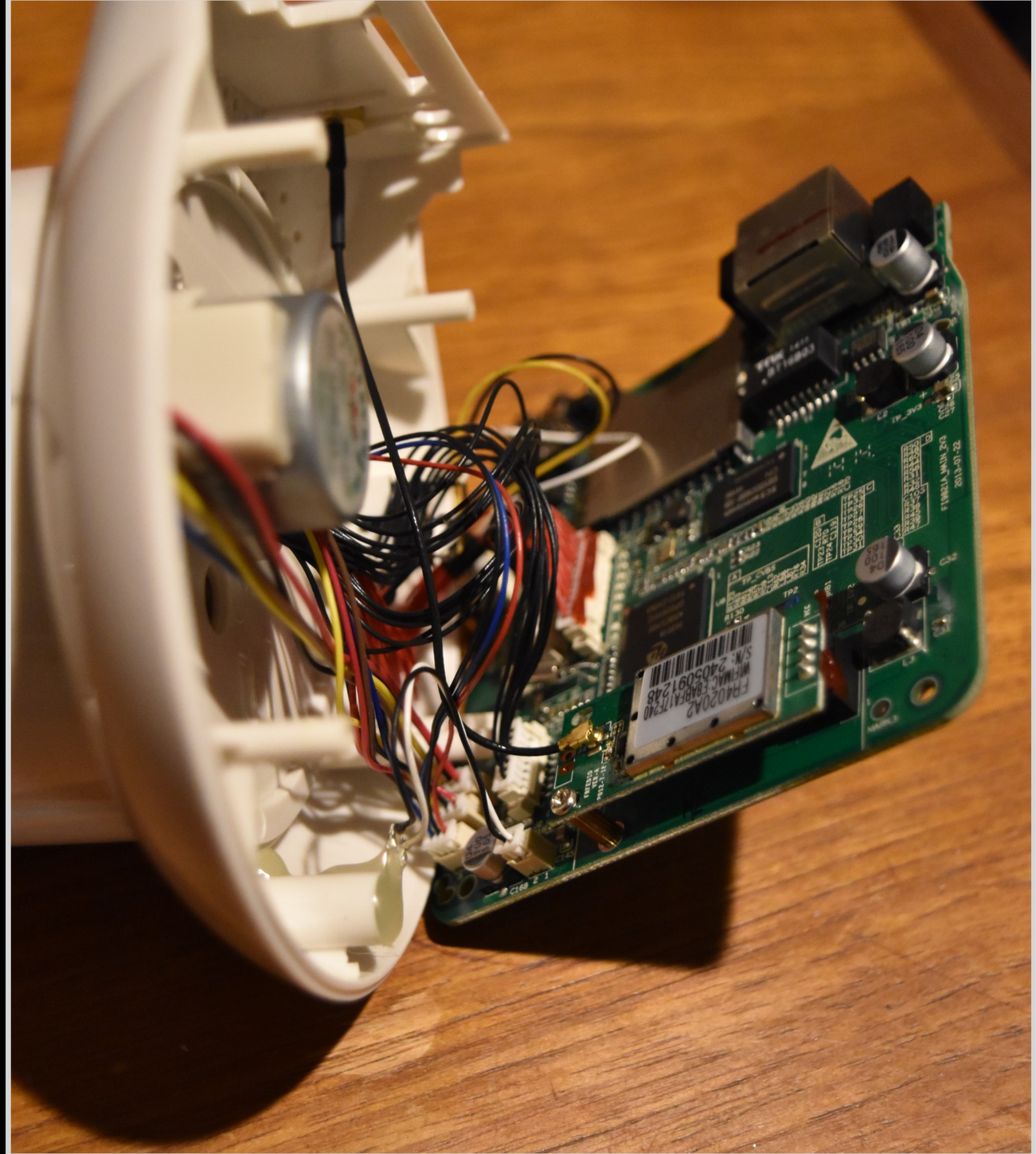
Firmware version:
1.11.1.16

Foscam F19821W

A handy sticker with our default admin credentials

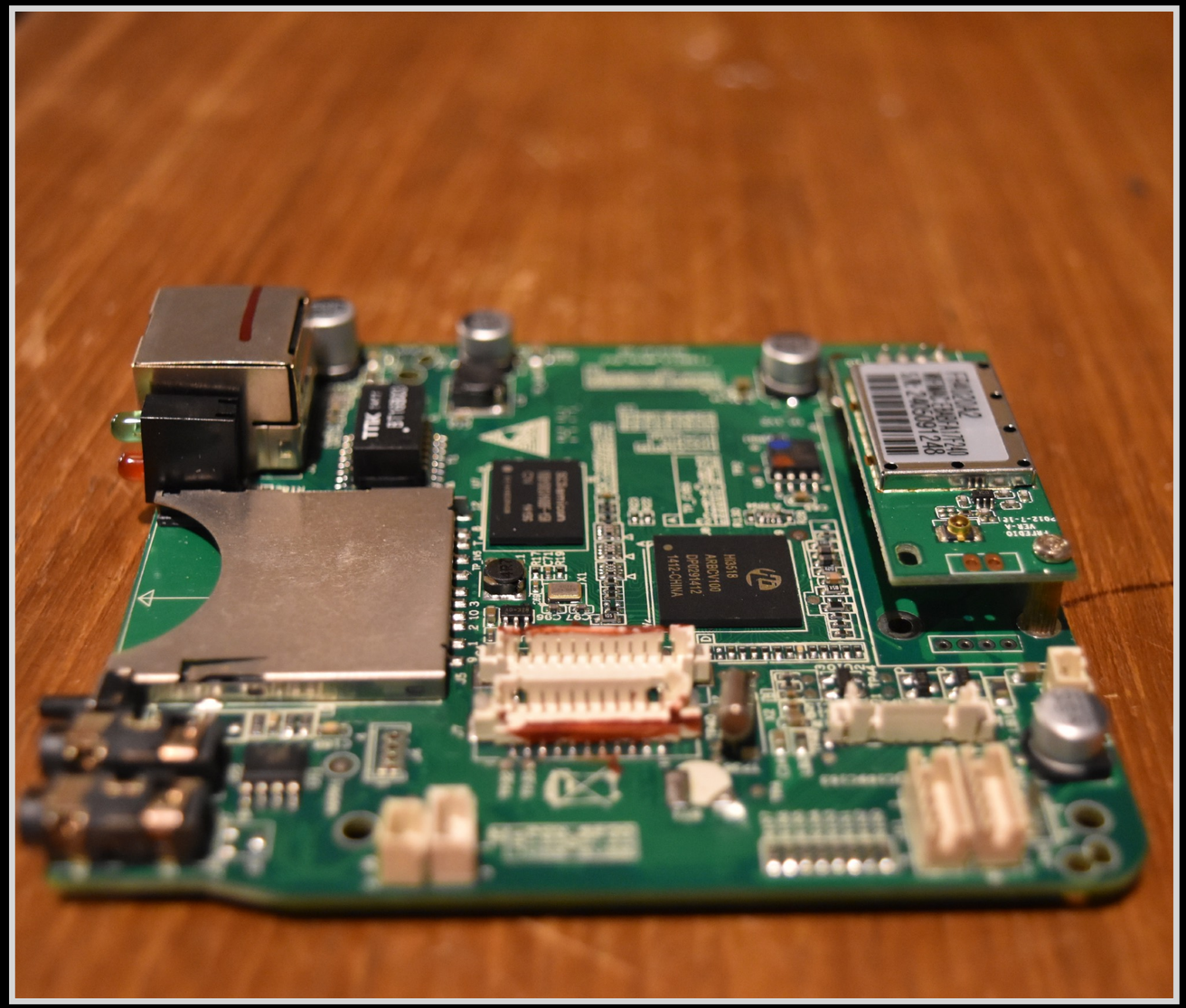
Foscam F19821W

Opening up the device



Foscam F19821W

PCB of Foscam



Unique features/challenges of hacking embedded devices

1. It’s headless - need to set up a serial console to see terminal during important processes
2. It runs an architecture that will not be x86. I learned ARM assembly and have been working primarily with devices that run on that architecture throughout my work on this project. There is also MIPS, PowerPC, others


Firmware

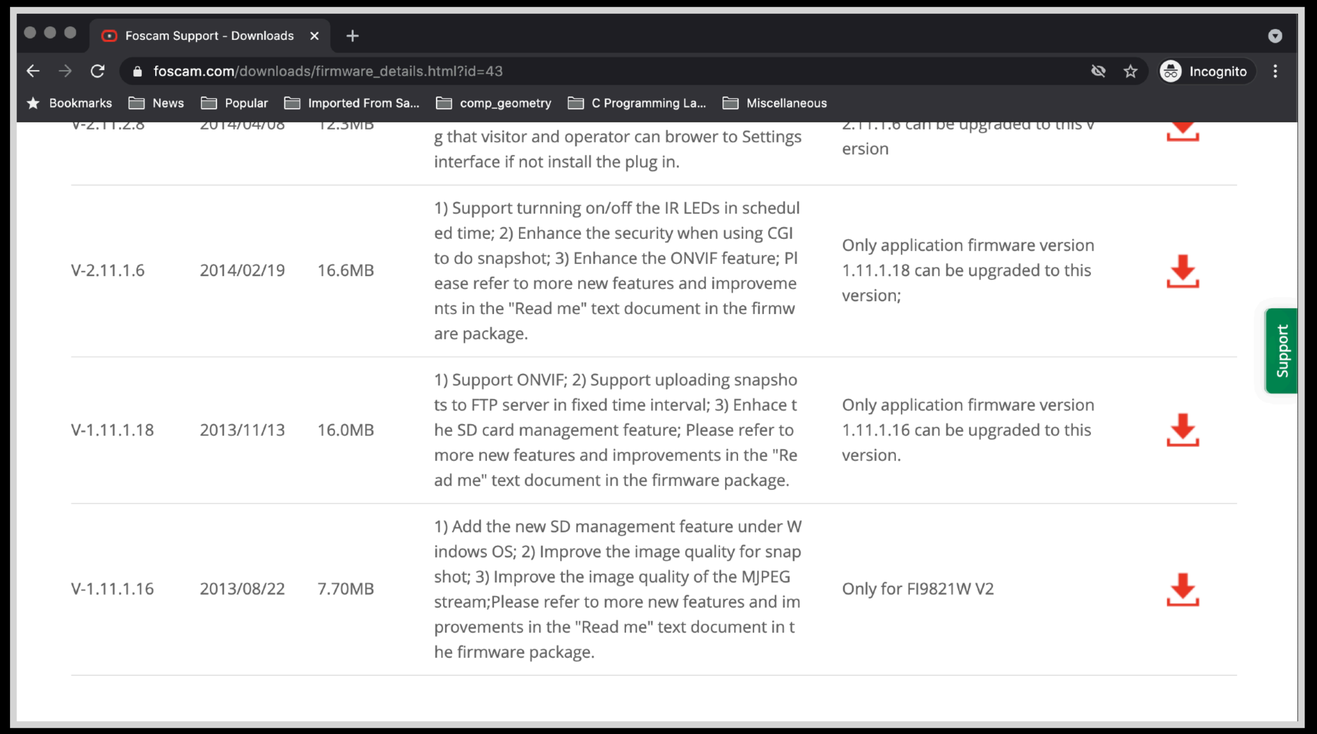
Firmware

Intermediary piece of software that serves as the interface between software/ hardware

Similar to an OS kernel

Foscam F19821W

obtaining the firmware: directly from vendor website



Foscam F19821W

file FI9821A_app_ver1.11.1.16.bin

output from file command indicates that the firmware image is encrypted with openssl

```
root@kali:~/Desktop/dc207_hardware_talk_demo/_FI9821W_V2-1.11.1.16-20130822.zip.extracted/firmware_binaries# file FI9821A_app_ver1.11.1.16
.bin
FI9821A_app_ver1.11.1.16.bin: openssl enc'd data with salted password
```

Foscam F19821W

binwalk -v FI9821A_app_ver1.11.1.16.bin

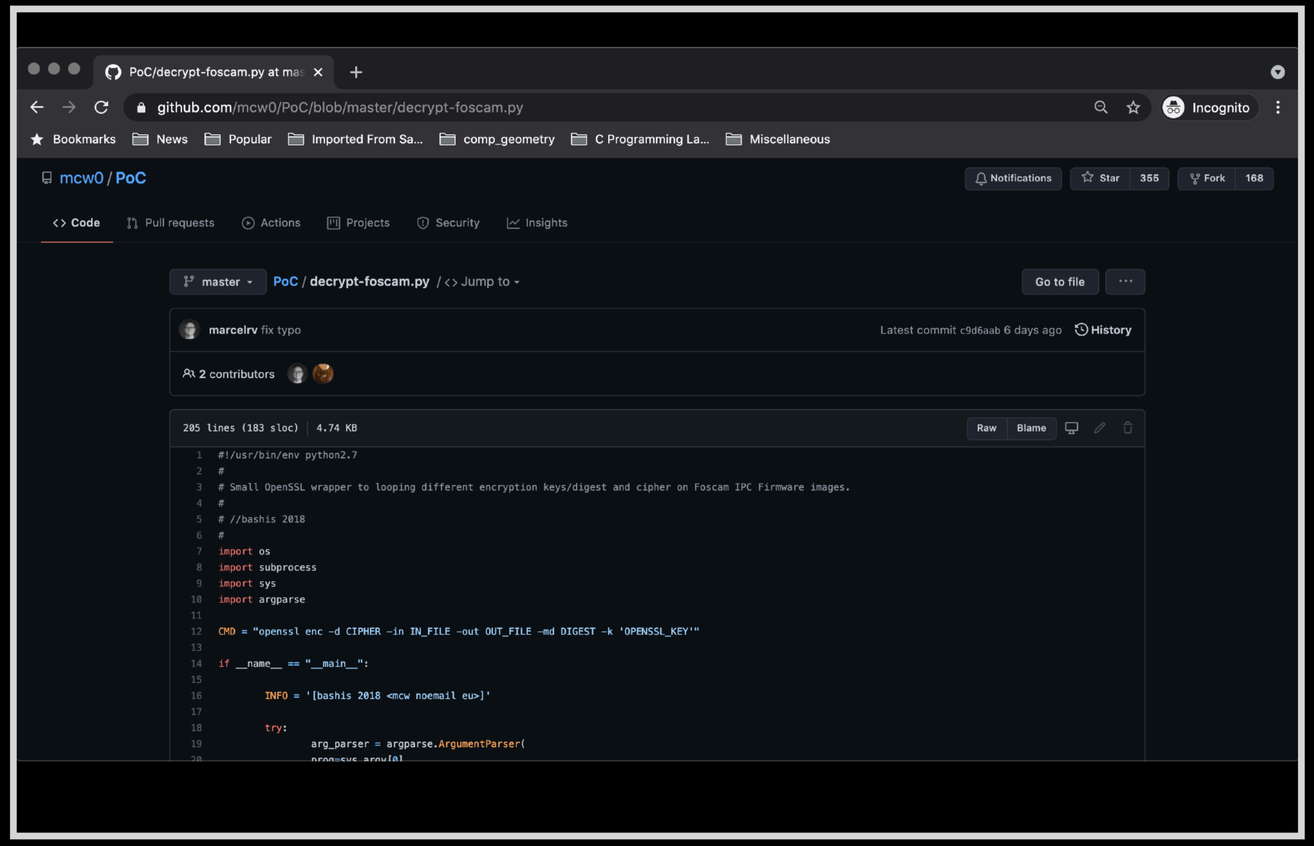
binwalk also tells us that the file is encrypted with openssl

```
root@kali:~/Desktop/dc207_hardware_talk_demo/_FI9821W_V2-1.11.1.16-20130822.zip.extracted/firmware_binaries# binwalk -v
FI9821A_app_ver1.11.1.16.bin

Scan Time:     2021-05-10 01:18:22
Target File:   /root/Desktop/dc207_hardware_talk_demo/_FI9821W_V2-1.11.1.16-20130822.zip.extracted/firmware_binaries/FI9
821A_app_ver1.11.1.16.bin
MD5 Checksum:  23c03a5d0d2a1d7e66b373a82159d6ec
Signatures:    391

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             OpenSSL encryption, salted, salt: 0x26C939CA4C17754E
```

Foscam F19821W

Firmware decryption script for Foscam firmware images

mcw0 Github



Foscam F19821W

Firmware decryption script for Foscam firmware images

list of potential decryption keys

mcw0 Github

```
#
#
ENC_KEYS = {
        0:'Wxift*',             # Decrypt: HI3518A_ddr256M_sys_ver 1.4.1.7 / 1.4.1.8 + FI9x Ver 1, recovertool
        1:'Wxift*v2',   # FW Decrypt 'B' AKA 'FosBaby'
        2:'WWxift*',
        3:'WWxift*v2',  # FW Decrypt 'B' AKA 'FosBaby'

        4:'Wyift*',
        5:'Wyift*v2',   # FW Decrypt 'C' AKA FI9903P
        6:'WWyift*',
        7:'WWyift*v2',  # FW Decrypt 'C' AKA FI9903P

        8:'Wzift*',
        9:'Wzift*v2',
        10:'WWzift*',
        11:'WWzift*v2', # FW Decrypt 'E' AKA 'C1'

        12:'Weift*',
        13:'Weift*v2',
        14:'WWeift*',
        15:'WWeift*v2', # FW Decrypt 'G' AKA 'C1-Lite'

        16:'Pxift*',    # exist (config)
        17:'Pxift*v2',
        18:'PPxift*',
        19:'PPxift*v2',

        20:'Xti1f*',    # recovertool
        21:'Xti1f*v2',
        22:'XXti1f*',
        23:'XXti1f*v2',

        24:'Ktf1g*',    # exist (config)
        25:'Ktf1g*v2',
        26:'KKtf1g*',
        27:'KKtf1g*v2',
```

Foscam F19821W

decrypted firmware image contents

```
root@kali:~/Desktop/dc207_hardware_talk_demo/_FI9821W_V2-1.11.1.16-20130822.zip.extracted/decrypted_firmware_dir# ls
app  app.tar  boot.sh  FWUpgradeConfig.xml  fwupgrade.md5  resolv.conf
root@kali:~/Desktop/dc207_hardware_talk_demo/_FI9821W_V2-1.11.1.16-20130822.zip.extracted/decrypted_firmware_dir# cat resolv.conf
nameserver 192.168.1.1
nameserver
root@kali:~/Desktop/dc207_hardware_talk_demo/_FI9821W_V2-1.11.1.16-20130822.zip.extracted/decrypted_firmware_dir#
```

Foscam F19821W

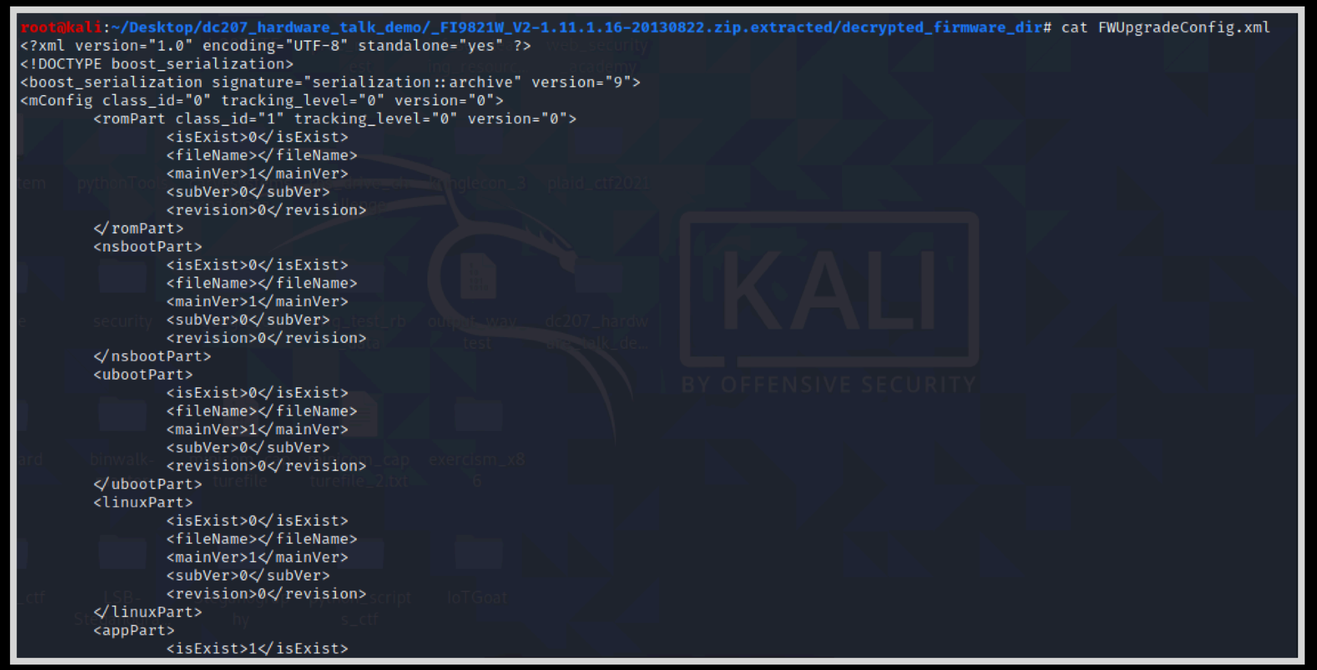
strings in the FirmwareUpgrade binary reveal that the decryption command is hardcoded, along with key

```
root@kali:~/Desktop/dc207_hardware_talk_demo/_FI9821W_V2-1.11.1.16-20130822.zip.extracted/decrypted_firmware_dir/app/bin#
 strings FirmwareUpgrade | grep -n --color -i "openssl"
184:openssl enc -d -aes-128-cbc -k Wxift* -in %s > %s
185:openssl exist abnormally
186:FWUpgrade openssl decryption fail, result=%d
```

Foscam F19821W

FirmwareConfig.xml

Tells us that we only have one partition of the device, have to extract firmware to directly from chip to explore further



Foscam F19821W

boot.sh

```
#init network
#if [ -e /mnt/mtd/debug ]; then
#ifconfig eth0 down
#ifconfig eth0 hw ether 004657166815
#ifconfig eth0 up
#ifconfig eth0 192.168.1.15
#mount -t nfs -o nolock 192.168.1.223:/home/foscam/nfs /mnt/nfs
#fi
ifconfig lo up

#telnetd
rm -rf /mnt/mtd/app/bin/lighttpd-1.4.30-hi
mkdir /usr/local
cp -R /mnt/mtd/app/bin/lighttpd-1.4.31-hi /usr/local/

cd /tmp/
cp /mnt/mtd/app/www.tar.gz .
tar -zxf /tmp/www.tar.gz -C /tmp/www
rm -f /tmp/www.tar.gz
cp /mnt/mtd/app/bin/cgi-bin/* /tmp/www/cgi-bin/
ln -s /mnt/mtd/app/plugins/FSIPCam.cab /tmp/www/FSIPCam.cab
ln -s /mnt/mtd/app/plugins/plugins.crx /tmp/www/plugins.crx
ln -s /mnt/mtd/app/plugins/plugins.pkg /tmp/www/plugins.pkg
ln -s /mnt/mtd/app/plugins/plugins.xpi /tmp/www/plugins.xpi

#cp /mnt/mtd/app/bin/* /bin/
#cp -rf /mnt/mtd/app/bin/ppp /bin/

#echo 8192 > /proc/sys/vm/min_free_kbytes
cp /mnt/mtd/app/bin/ftpd/FtpPortConfig.xml /mnt/mtd/app/config/
mkdir /usr/local/pureftpd
mkdir /usr/local/pureftpd/etc
cp /mnt/mtd/app/bin/ftpd/pureftpd.passwd /usr/local/pureftpd/etc/
cp /mnt/mtd/app/bin/ftpd/pureftpd.pdb /usr/local/pureftpd/etc/
```

Foscam F19821W

boot.sh continued

```
cd /mnt/mtd/app/modules
./load3518 -i
 eval $(cat /mnt/mtd/app/config/PTZConfig.xml | grep 'SelfTestMode' | awk -F ">" '{print $2}' | awk -F "<" '{printf("ptzstate_xmlcfg=%d",$1
);}')
     eval $(cat /mnt/mtd/app/config/PTZConfig.xml | grep 'PreHorPos_Appointed' | awk -F ">" '{print $2}' | awk -F "<" '{printf("prehorpos_x
mlcfg=%d",$1);}')
        eval $(cat /mnt/mtd/app/config/PTZConfig.xml | grep 'PreVerPos_Appointed' | awk -F ">" '{print $2}' | awk -F "<" '{printf("preverp
os_xmlcfg=%d",$1);}')
          /sbin/insmod /mnt/mtd/app/modules/extdrv/fos_ptz.ko ptz_state=$ptzstate_xmlcfg horizen_desire_pos=$prehorpos_xmlcfg vertical_d
esire_pos=$preverpos_xmlcfg
if [ $? -ne 0 ]
then
   echo "insmod fos_ptz.ko fail, now try default param"
   /sbin/insmod /mnt/mtd/app/modules/extdrv/fos_ptz.ko ptz_state=1    #Normal self test mode
fi
/sbin/insmod /mnt/mtd/app/modules/hirtc.ko
/sbin/insmod /mnt/mtd/app/modules/wdt.ko default_margin=5

hwclock -s

mkdir -p /etc/Wireless/RT2870STA
cp -f /mnt/mtd/app/etc/RT2870STA.dat /etc/Wireless/RT2870STA/

mkdir /tmp/www
#cp -rf /mnt/mtd/app/lib/* /lib/
mkdir /etc/ppp

mkdir /mnt/windows

#init network
                                                                                                                    21,1          Top
```

Foscam F19821W

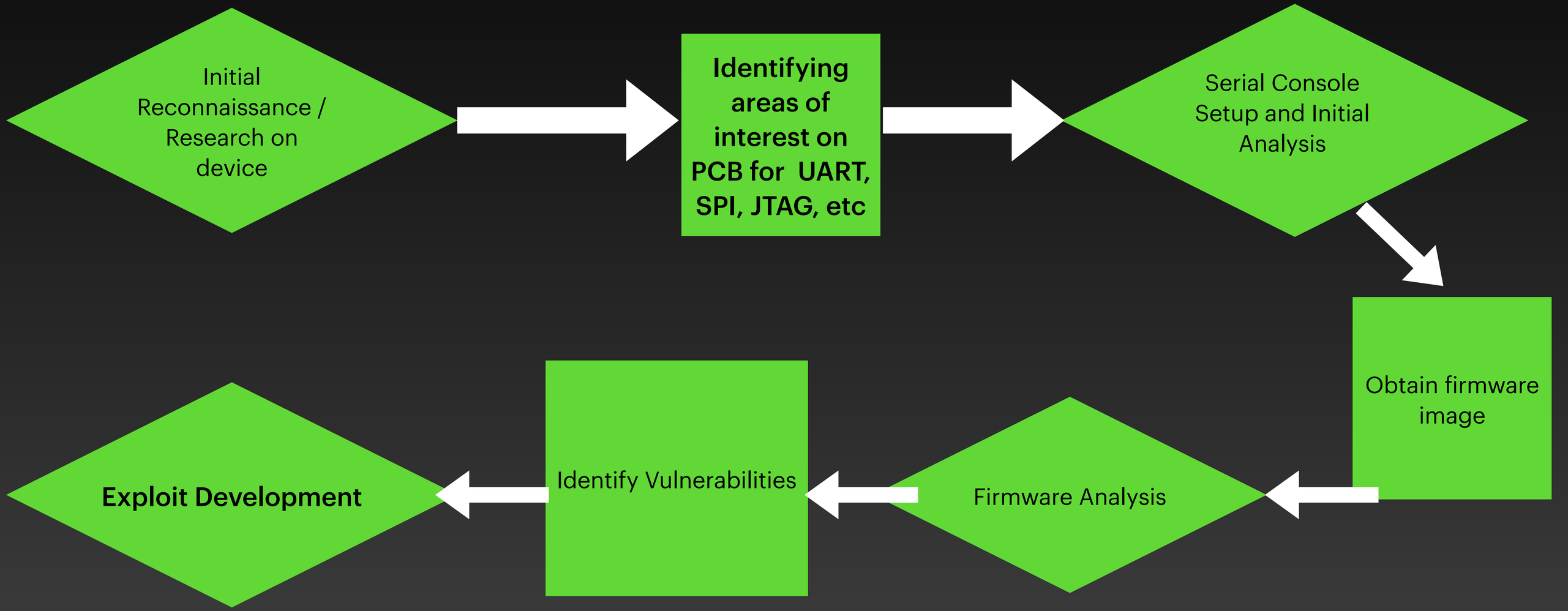
decompressed app.tar archive

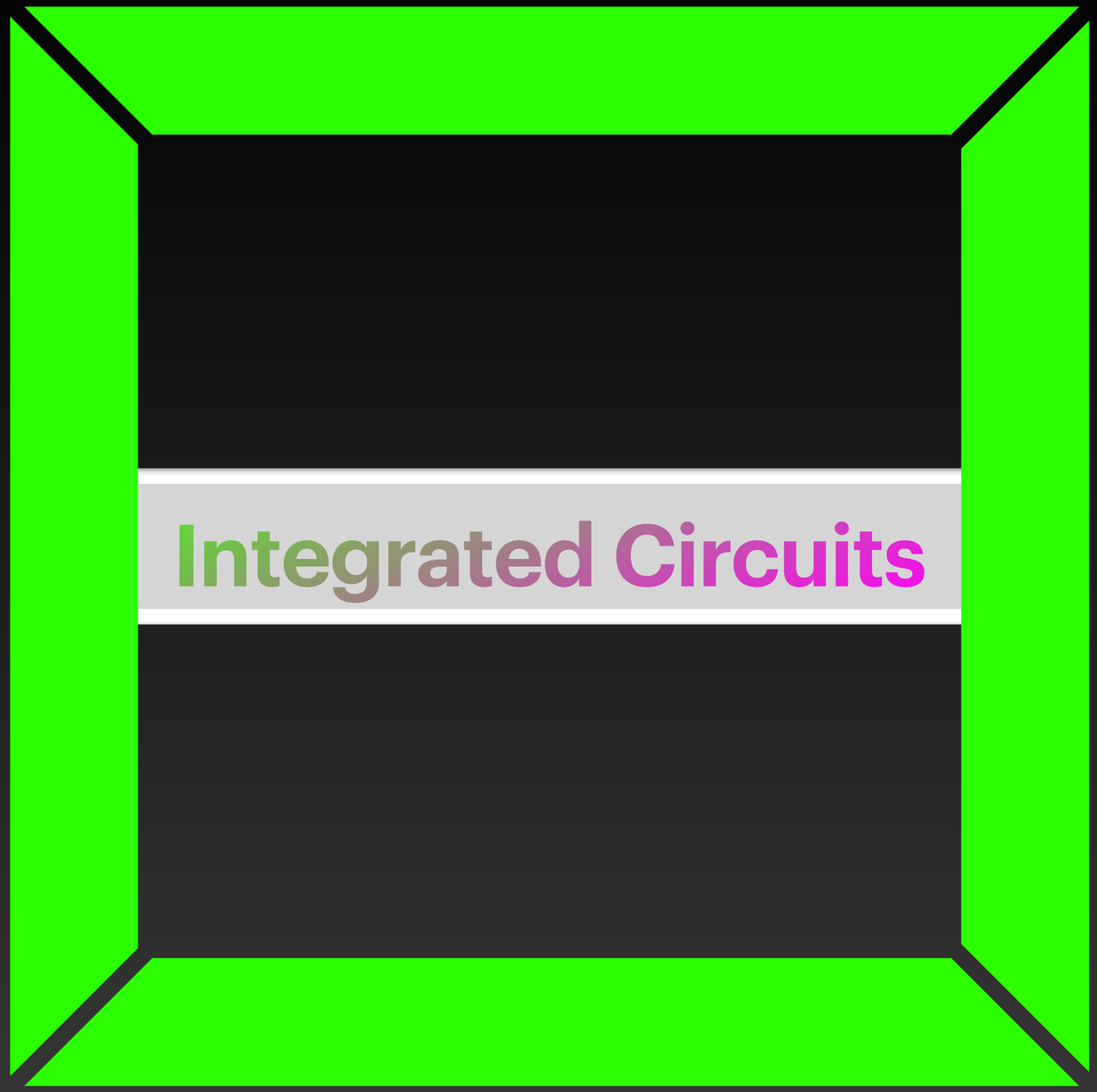
Foscam F19821W

contents of /bin/ dir

```
root@kali:~/Desktop/dc207_hardware_talk_demo/_F19821W_V2-1.11.1.16-20130822.zip.extracted/decrypted_firmware_dir/app/bin# ls
cgi-bin         FirmwareUpgrade  iwlist           MsgServer  ppp          rtctool        watchdog
codec           flashcp          iwpriv           msmtp      pure-ftpd    RtspServer     webService
ddnsclient      ftpd             lighttpd         ntpclient  pure-pw      storage        wpa_cli
devMng          iwconfig         lighttpd-1.4.31-hi openssl  releaseShm.sh UDTMediaServer wpa_supplicant
```

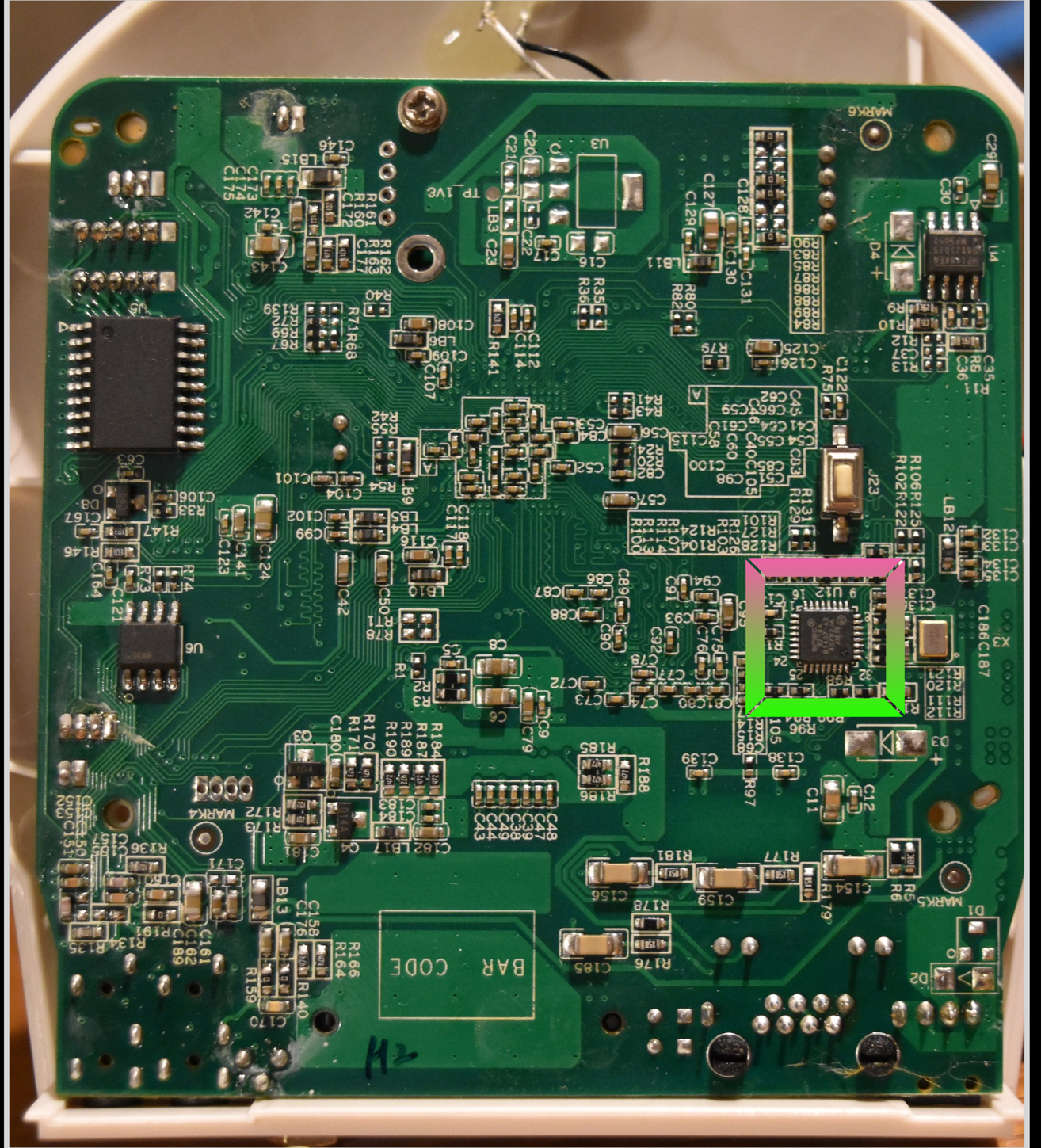
Reverse Engineering Methodology
(but make it Euclidean)




Integrated Circuits

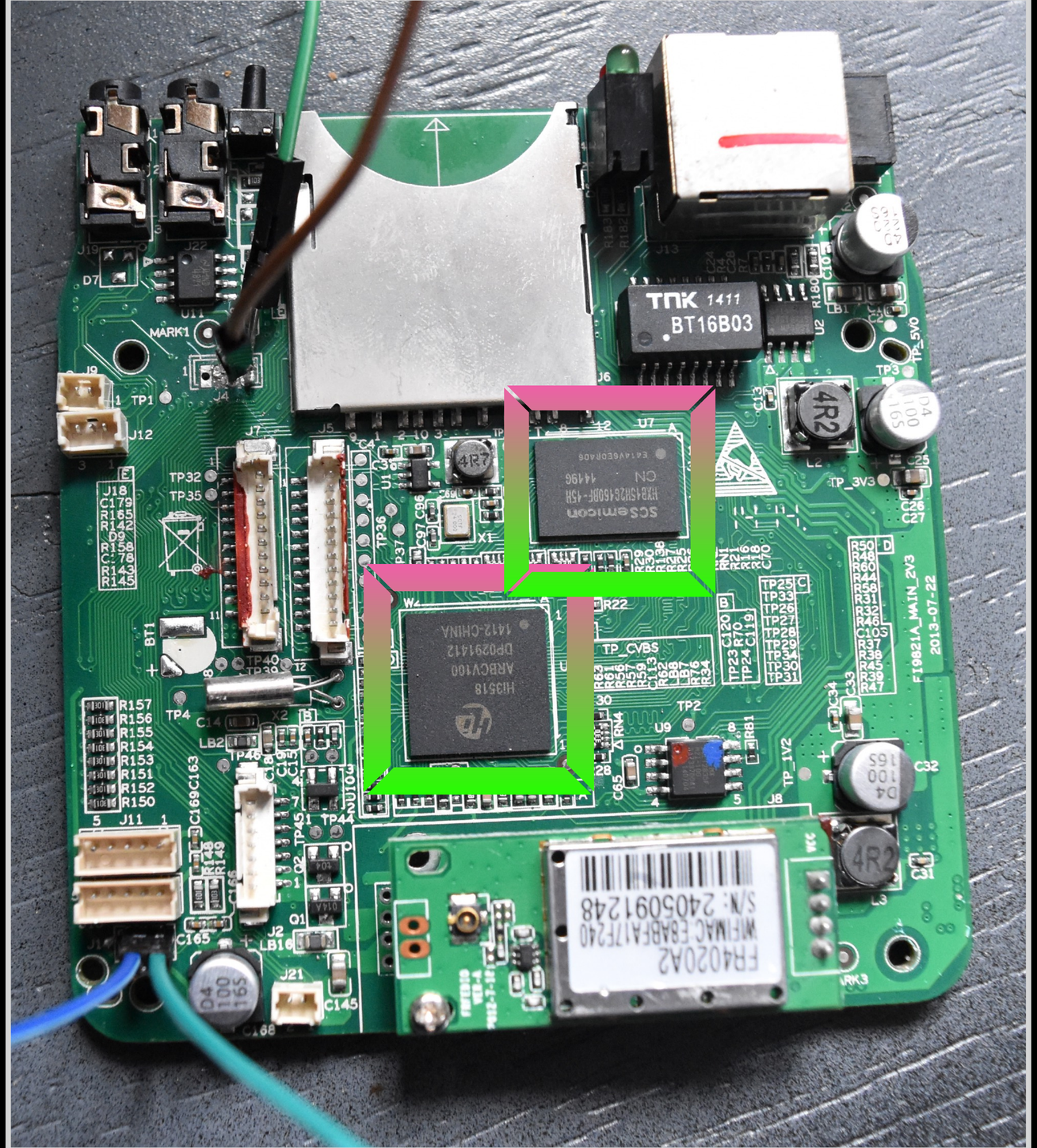
Processor Candidate

Find data sheet for specified chip, identify architecture etc.



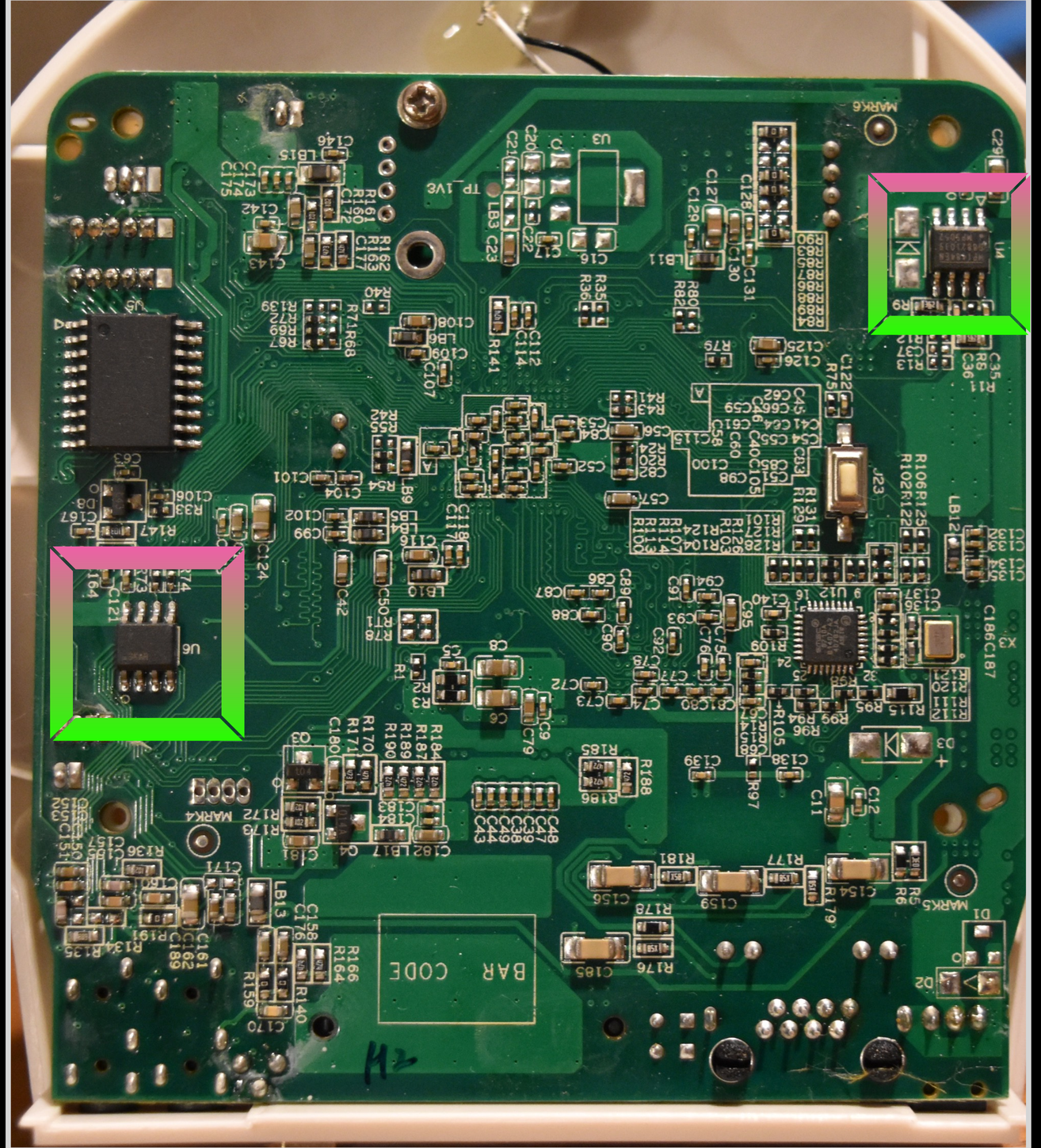
Processor candidates, opposite side of PCB

Identified Processor chip:
H1358



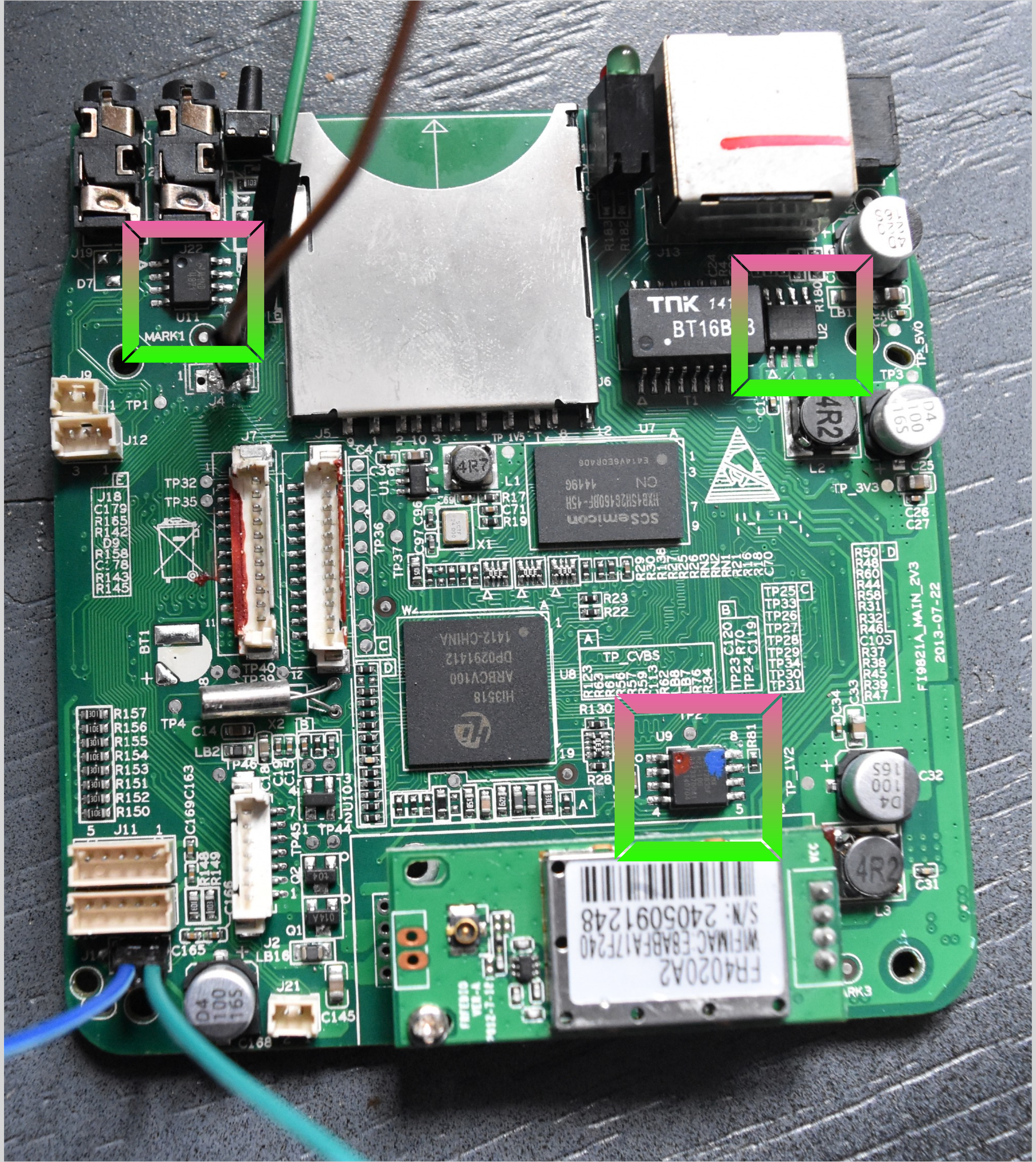
Flash chip candidates

Find associated data sheets for part number, cross reference with other device specs



Flash chip candidates, opposite side of PCB

Identified Flash Chip:
MX25L12835F



Datasheets

MP1484
Voltage Regulator

The Future of Analog IC Technology

MP1484

3A, 18V, 340KHz Synchronous Rectified Step-Down Converter

DESCRIPTION

The MP1484 is a monolithic synchronous buck regulator. The device integrates top and bottom 85mΩ MOSFETS that provide 3A of continuous load current over a wide operating input voltage of 4.75V to 18V. Current mode control provides fast transient response and cycle-by-cycle current limit.

An adjustable soft-start prevents inrush current at turn-on and in shutdown mode, the supply current drops below 1μA.

The MP1484 is PIN compatible to the MP1482 2A/18V/Synchronous Step-Down Converter.

FEATURES

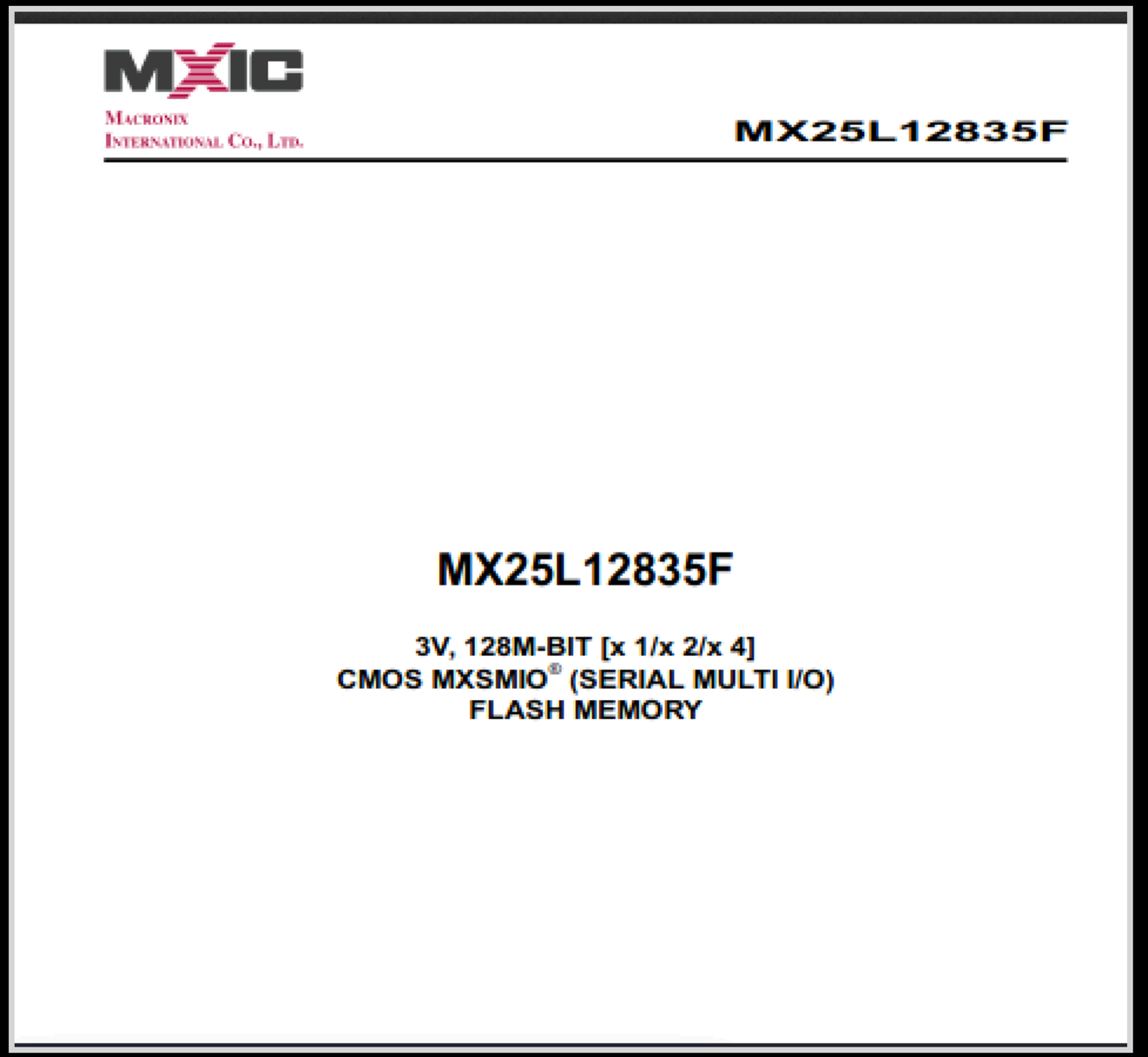
- 3A Continuous Output Current
- Wide 4.75V to 18V Operating Input Range
- Integrated 85mΩ Power MOSFET Switches
- Output Adjustable from 0.925V to 20V
- Up to 95% Efficiency
- Programmable Soft-Start
- Stable with Low ESR Ceramic Output Capacitors
- Fixed 340KHz Frequency
- Cycle-by-Cycle Over Current Protection
- Input Under Voltage Lockout
- Thermally Enhanced 8-Pin SOIC Package

APPLICATIONS

- FPGA, ASIC, DSP Power Supplies
- LCD TV
- Green Electronics/Appliances
- Notebook Computers

Datasheets

MX25L12835F



Datasheets

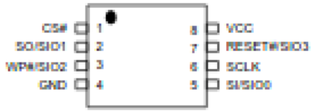
MX25L12835F
Serial Flash

MACRONIX
INTERNATIONAL CO., LTD.

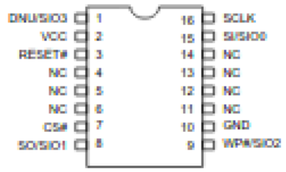
MX25L12835F

3. PIN CONFIGURATIONS

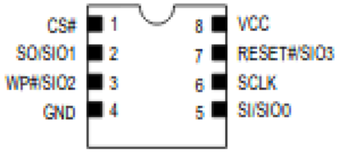
8-PIN SOP (200mil)



16-PIN SOP (300mil)



8-WSON (6x5mm, 8x6mm)

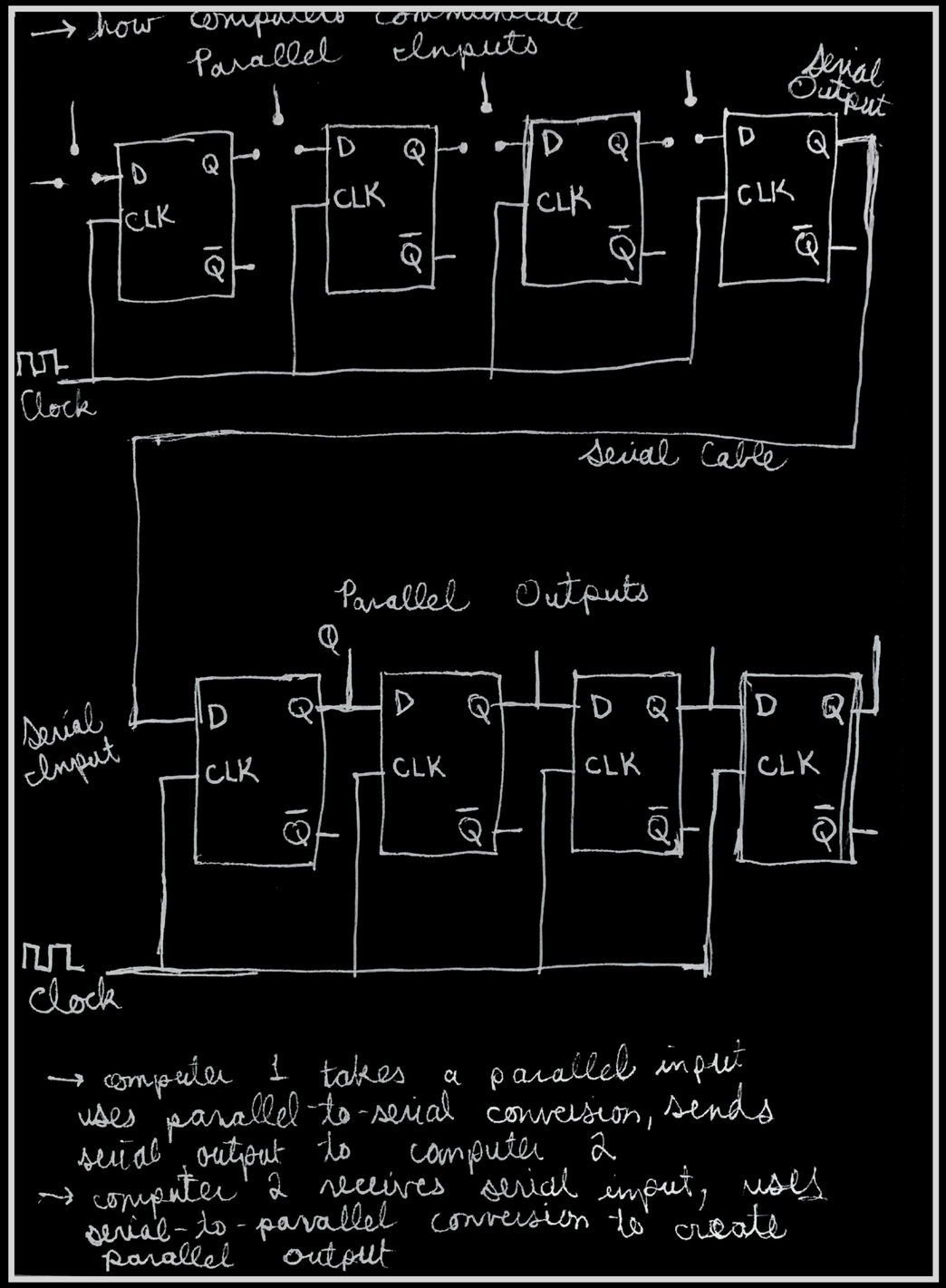


4. PIN DESCRIPTION

SYMBOL	DESCRIPTION
CS#	Chip Select
SI/SIO0	Serial Data Input (for 1 x I/O)/ Serial Data Input & Output (for 2xI/O or 4xI/O read mode)
SO/SIO1	Serial Data Output (for 1 x I/O)/ Serial Data Input & Output (for 2xI/O or 4xI/O read mode)
SCLK	Clock Input
WP#/SIO2	Write protection Active low or Serial Data Input & Output (for 4xI/O read mode)
RESET#/SIO3	Hardware Reset Pin Active low or Serial Data Input & Output (for 4xI/O read mode)
VCC	+ 3V Power Supply
GND	Ground
NC	No Connection
DNU	Do not use

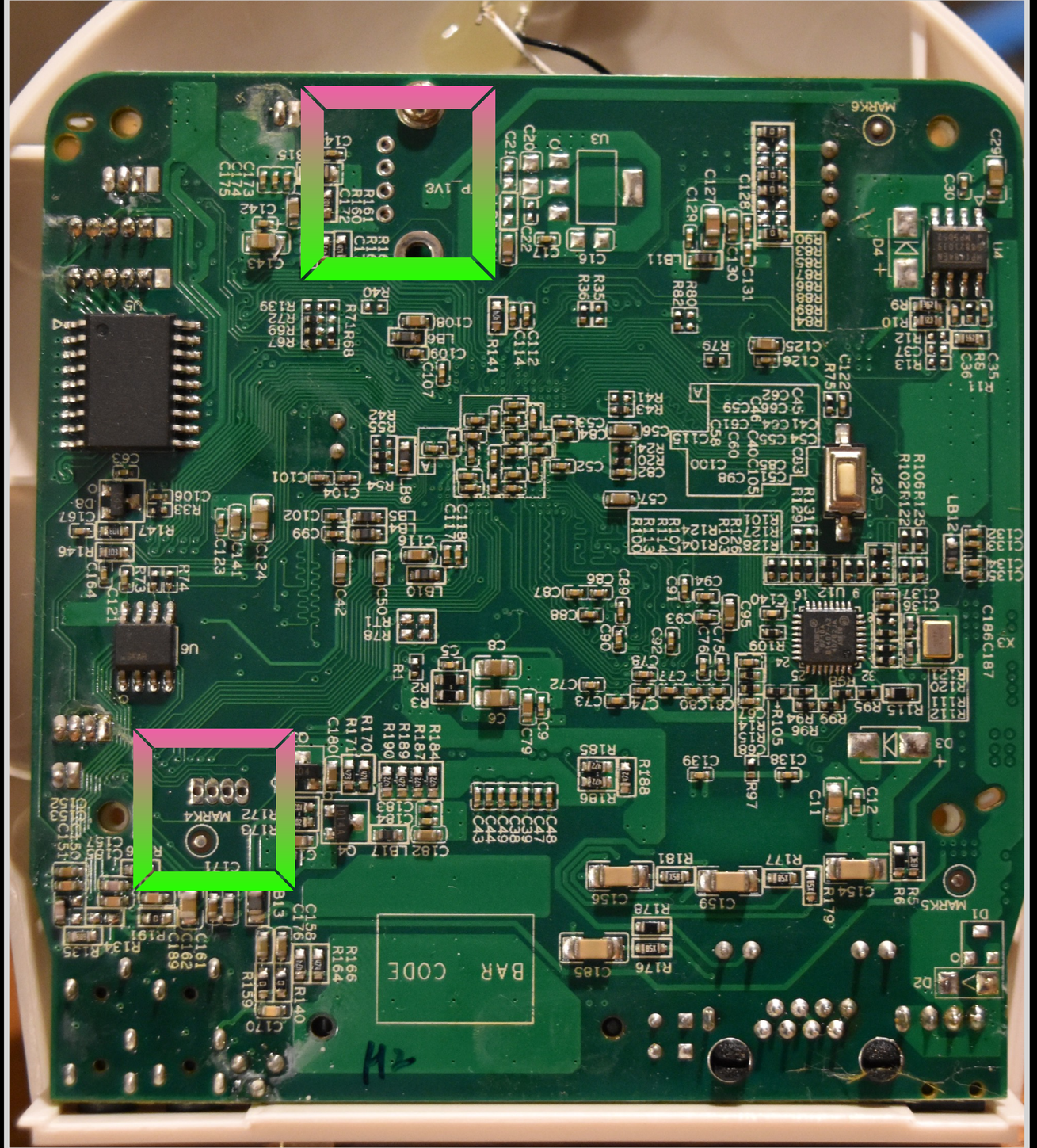
Notes:
1. RESET# pin has internal pull up.
2. When using 1I/O or 2I/O (QE bit not enable), the DNU/SIO3 pin of 16SOP can not connect to GND. Recommend to connect this pin to VCC or floating.

Simplified View of Communication between two computers

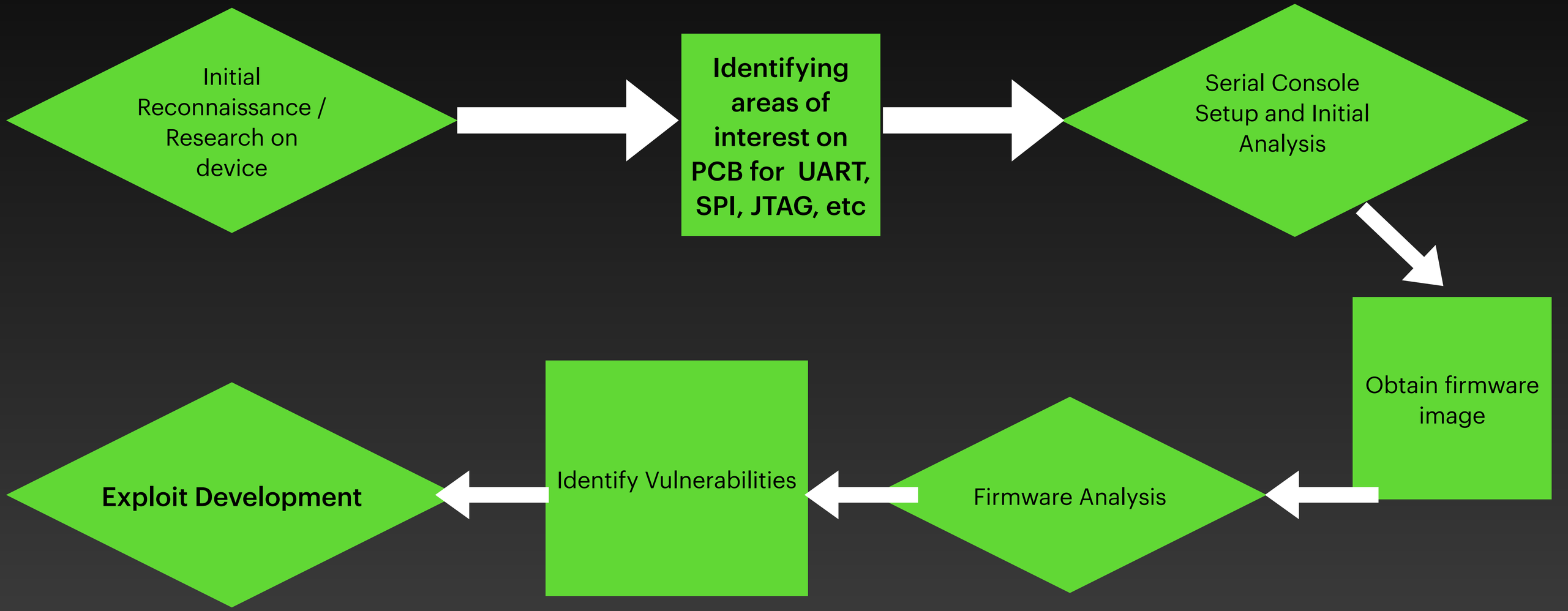


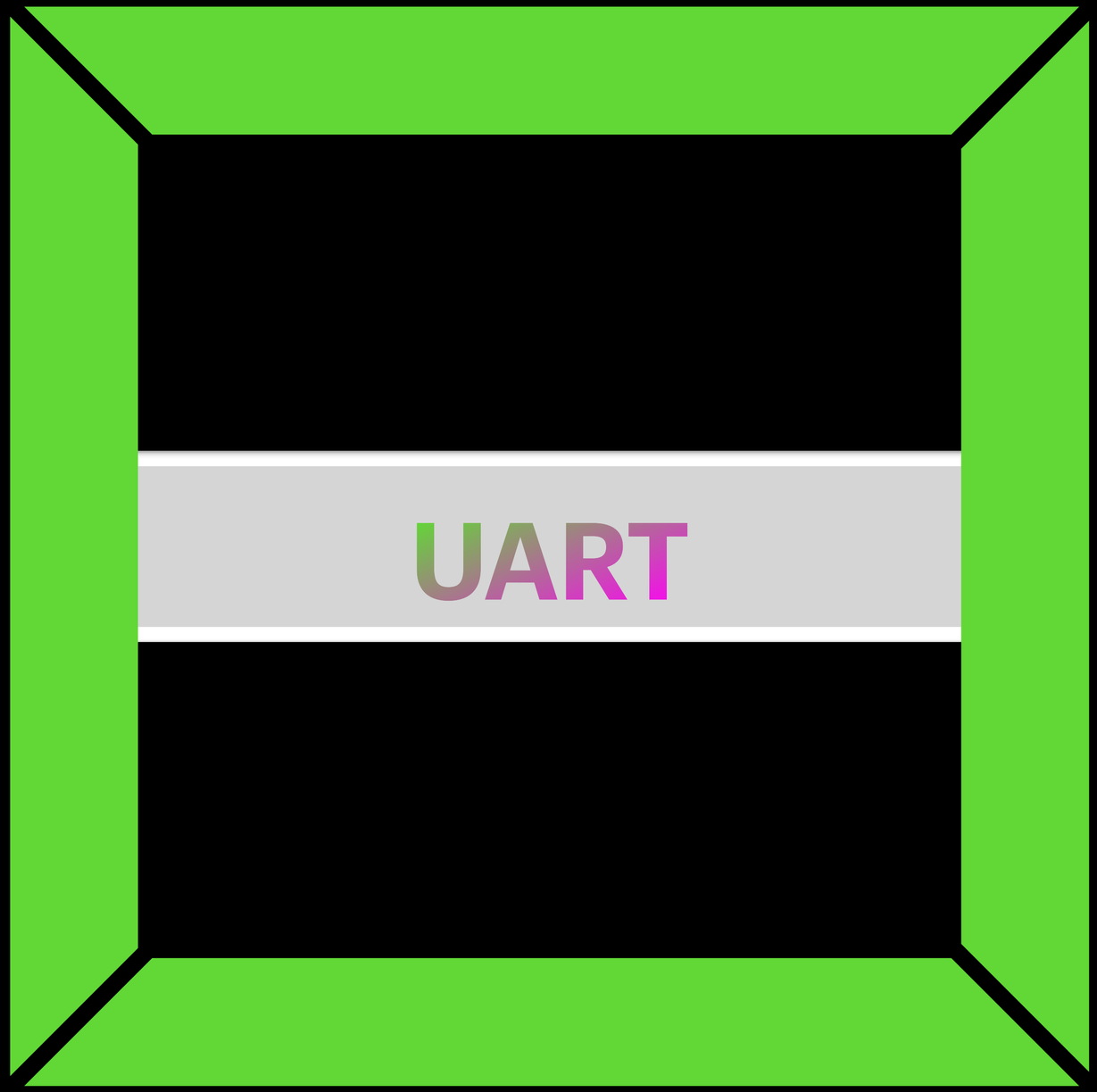
UART headers

Typically 4 pins in a row



Reverse Engineering Methodology
(but make it Euclidean)




UART

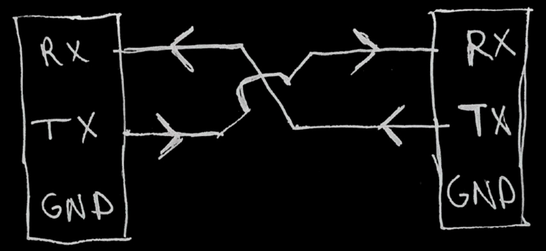
UART

Universal Asynchronous Receiver/Transmitter

3 pins:
TX
RX
GND

Frame:	Start	Data	Parity	Stop
Size (bits)	1	5-9	0-1	1-2

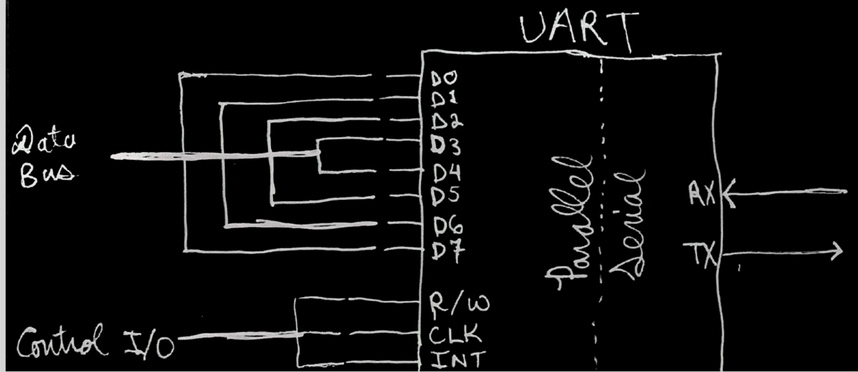
→ a serial bus consists of just two wires: the receiver (RX) and the transmitter (TX)



→ a full-duplex serial interface means that both devices can send and receive data simultaneously

→ half duplex serial interface means that devices must take turns sending and receiving data

→ a universal asynchronous receiver/transmitter (UART) is a block of circuitry responsible for implementing serial communication

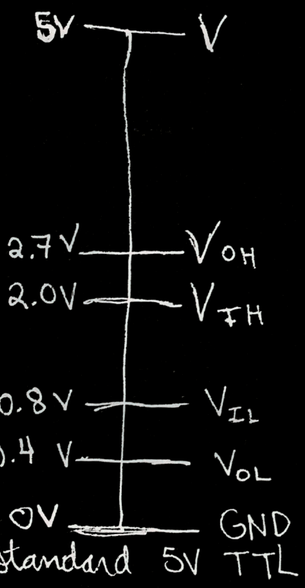


TTL (Transistor-Transistor Logic)

→ TTL or Transistor-Transistor Logic is a standard for defining logic states based on threshold voltage levels

→ TTL relies on circuits built from bipolar transistors to achieve switching and maintain logic states

→ a majority of systems in everyday life use 3.3V or 5V TTL levels



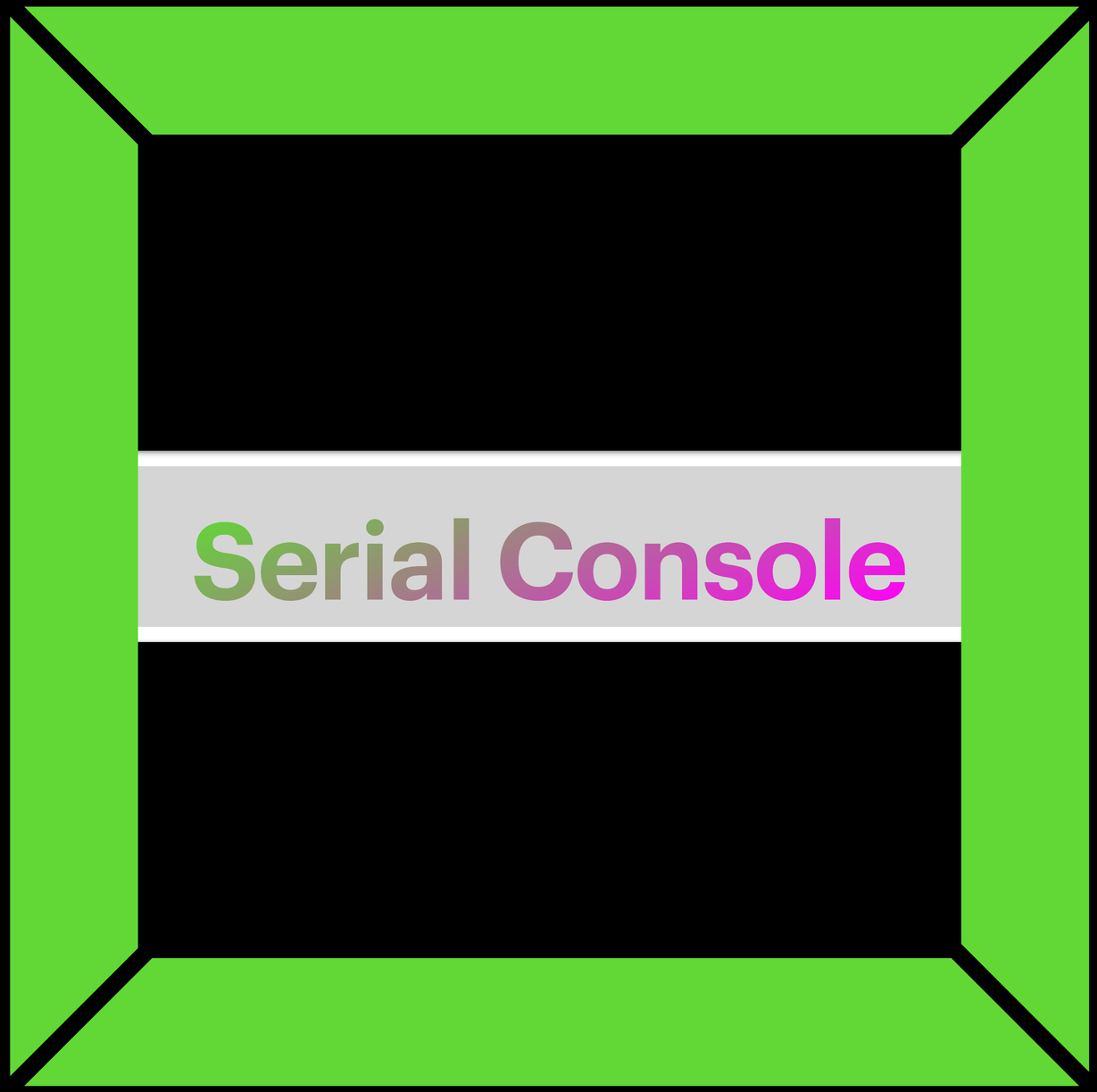
Standard 5V TTL

→ V_{OH} is the minimum output high voltage
→ the output voltage of the device driving HIGH will always be at least 2.7V

→ V_{IH} is the minimum input high voltage
→ basically any voltage that is at least 2V will be read in as a logic 1 (HIGH) to a TTL device

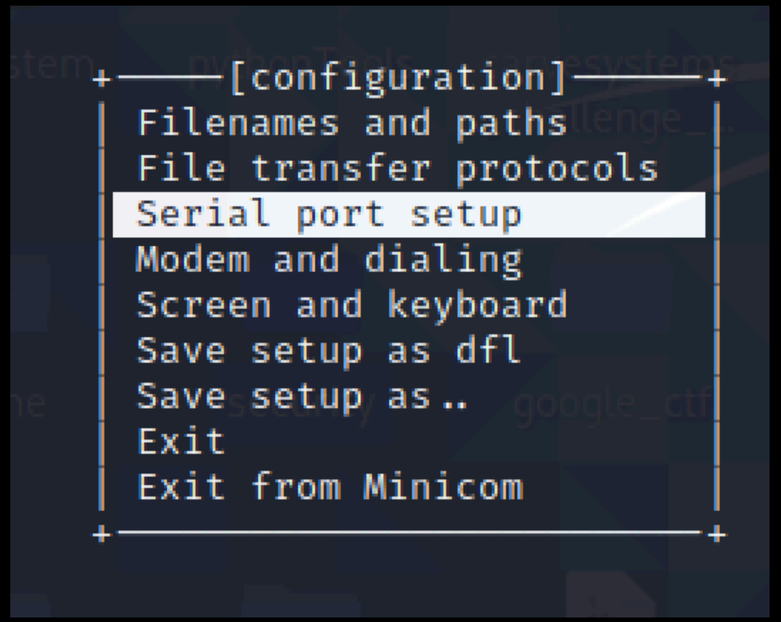
→ V_{IL} is the maximum input LOW voltage
→ any input signal that is below 0.8V will be considered a logic 0 (LOW) when read into a device

→ V_{OL} is the maximum output low voltage
→ a device trying to send a logic 0 will be below 0.4 V


Serial Console

Serial Console

The serial console uses a serial port connection to access a computer, allowing a user to interact with a device and issue commands through a terminal emulator



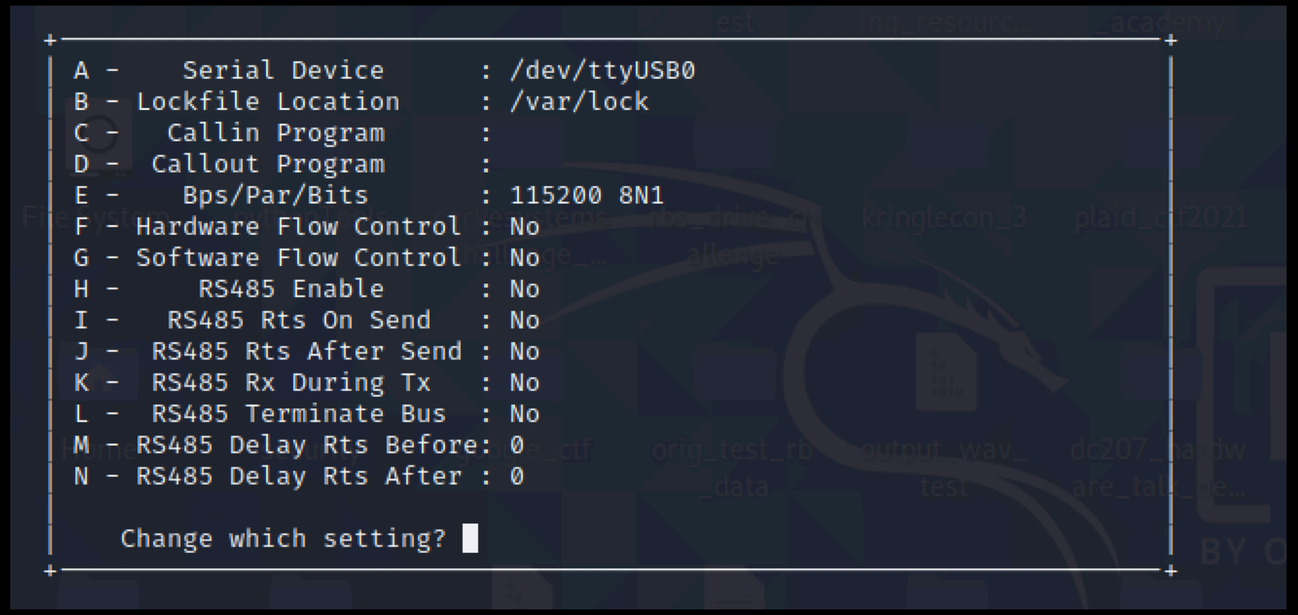
Serial Console

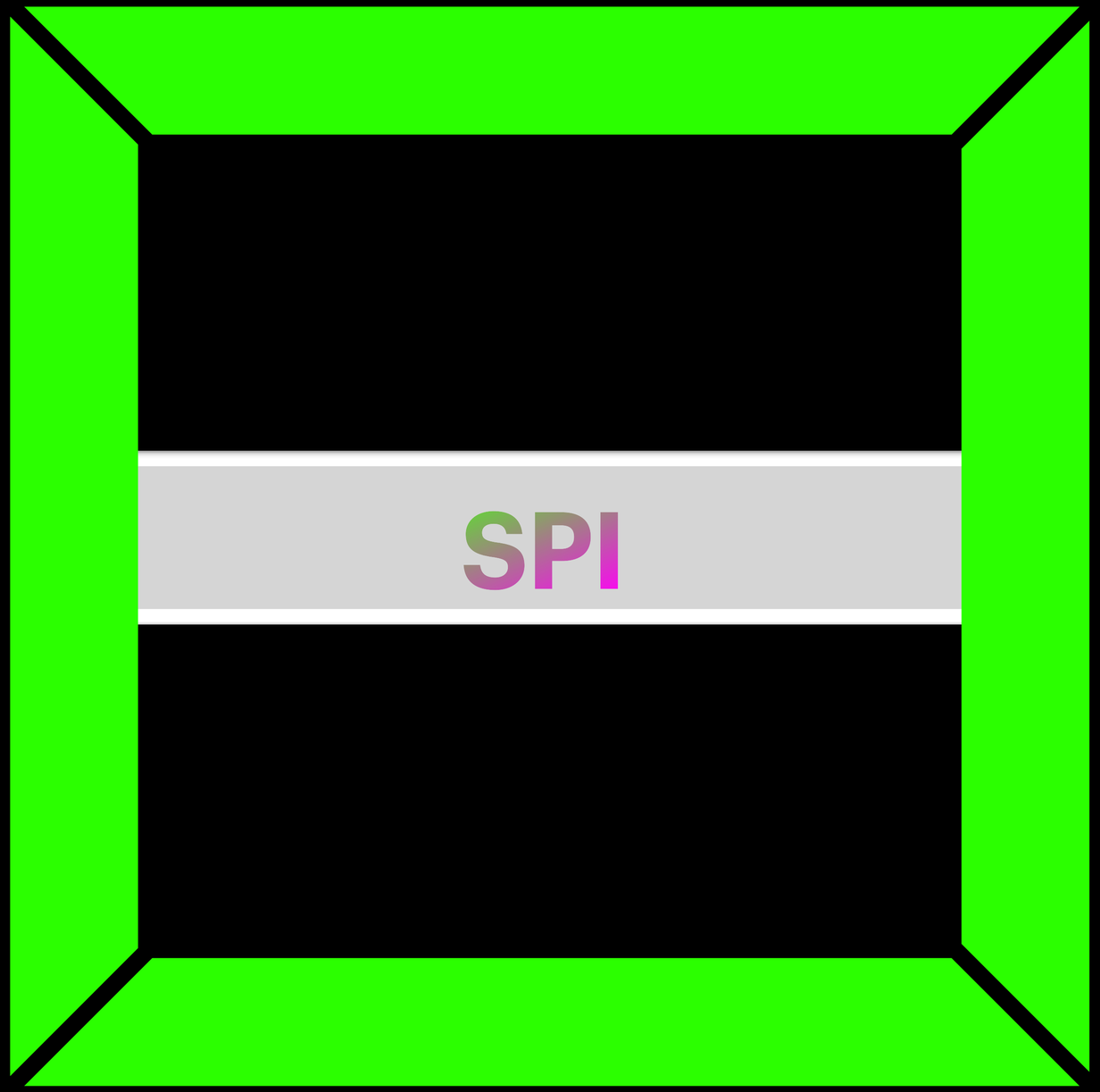
Serial terminal programs include (but are not limited to): minicom, cutecom, screen, Putty, etc.

setup command for minicom:
sudo minicom -s
(pictured to the right)

Enter configuration settings for the following:
baud rate, frame size, parity, start/stop bits

8N1 is a fairly common configuration for frame size, parity, and start/stop bits



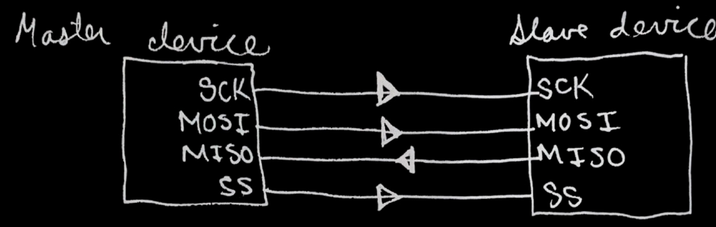

SPI

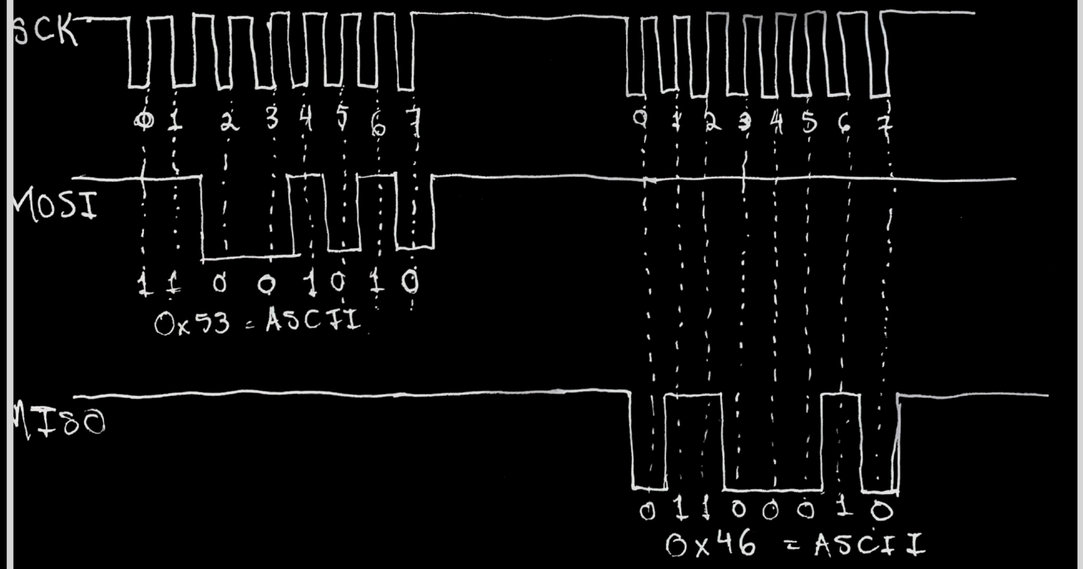
SPI

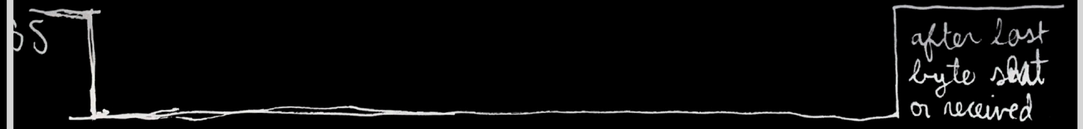
Serial Peripheral Interface

4 pins:
CS (Chip Select)
CLK (Clock)
MISO/DO (Master In/Slave Out, Data Output)
MOSI/DI (Master Out/Slave In, Data Input)

→ SS ("Slave Select") is the line that communicates to a specific slave device to cause it to wake up and be ready to send/receive data; it is used when multiple slave devices are present







→ the SS line is normally held high, which disconnects it from the SPI bus (meaning it's active-low)

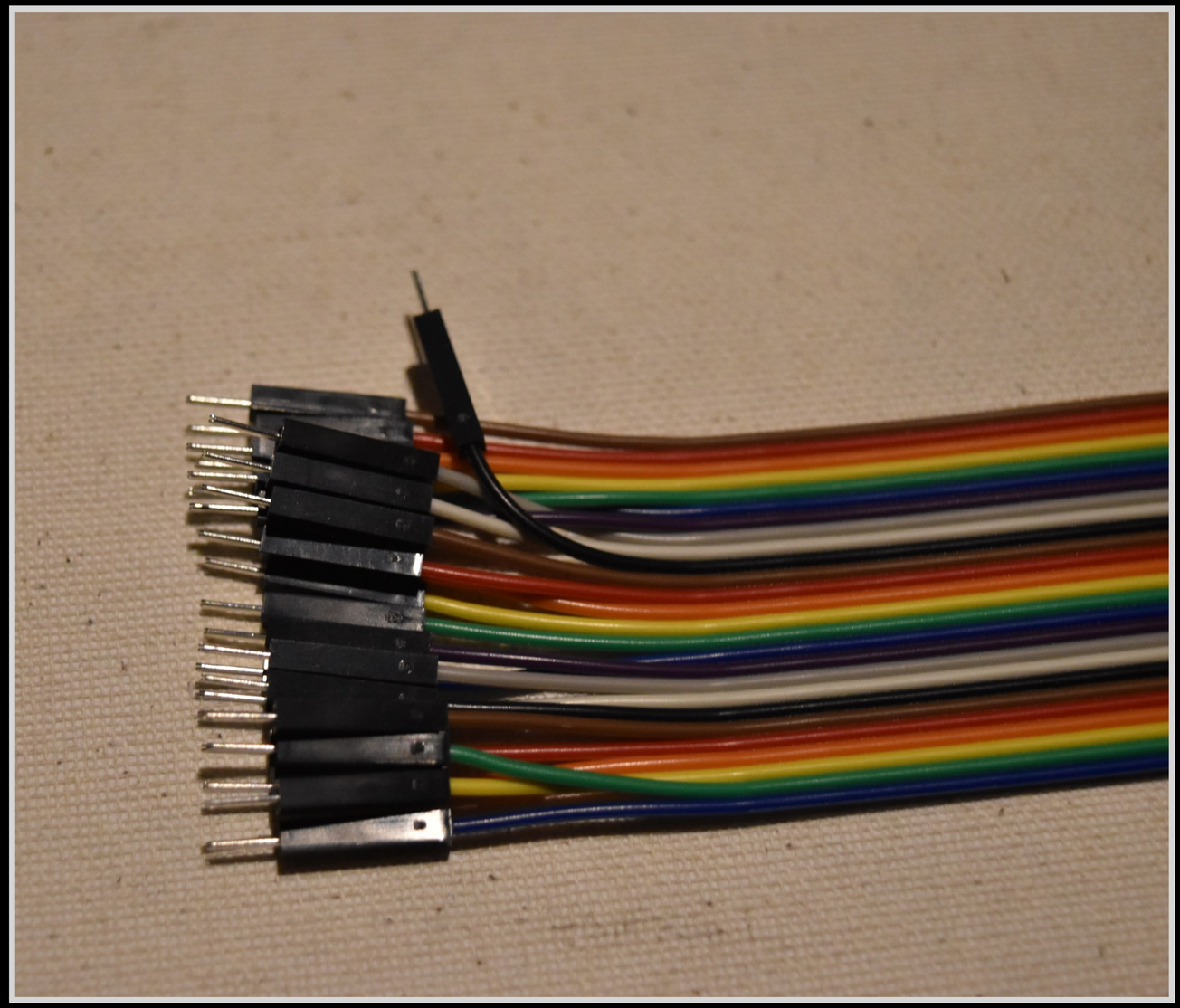

Part 2


Lab Materials


Essentials

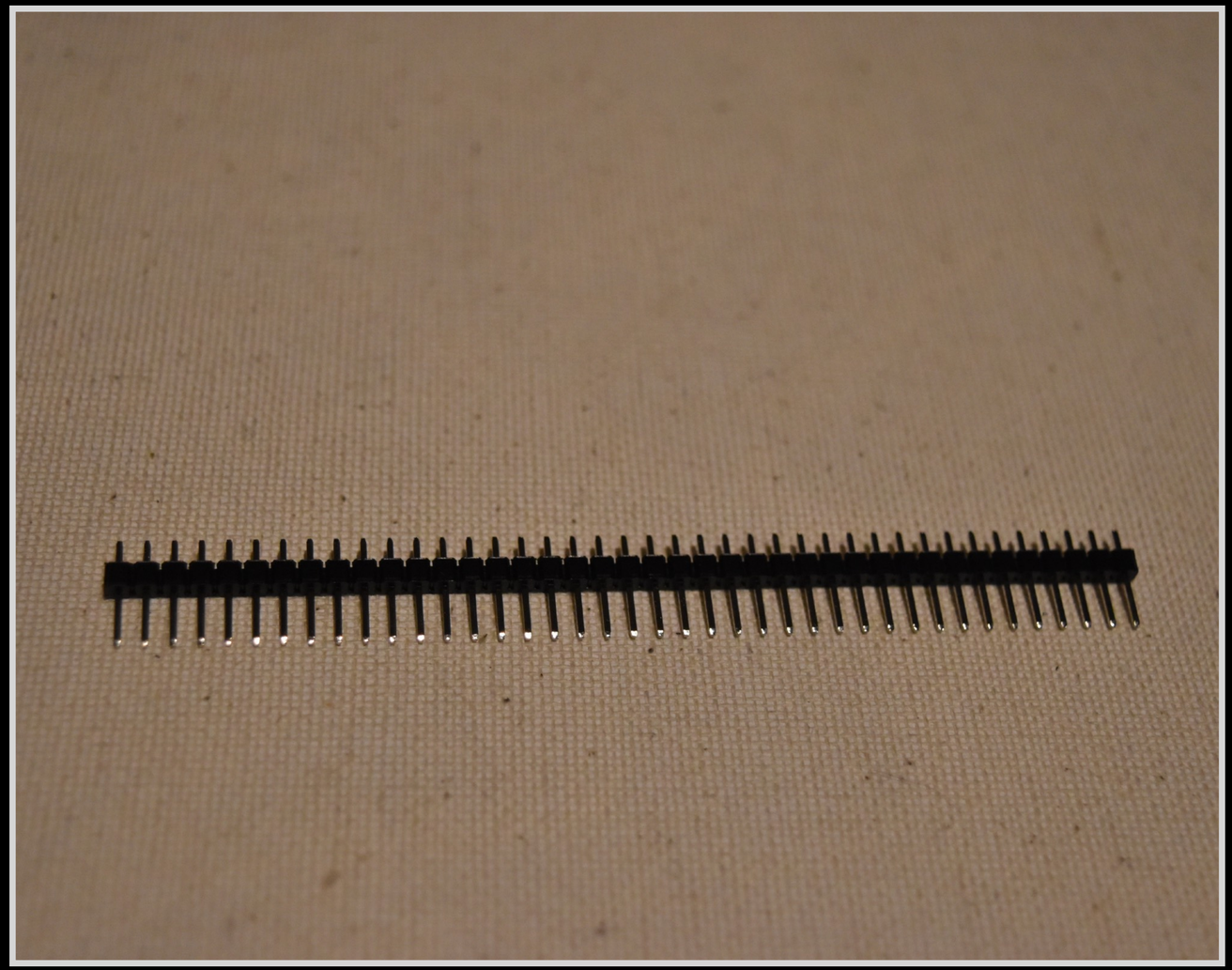
Jumper Wires

M/M, M/F, F/F
Variety for myriad uses



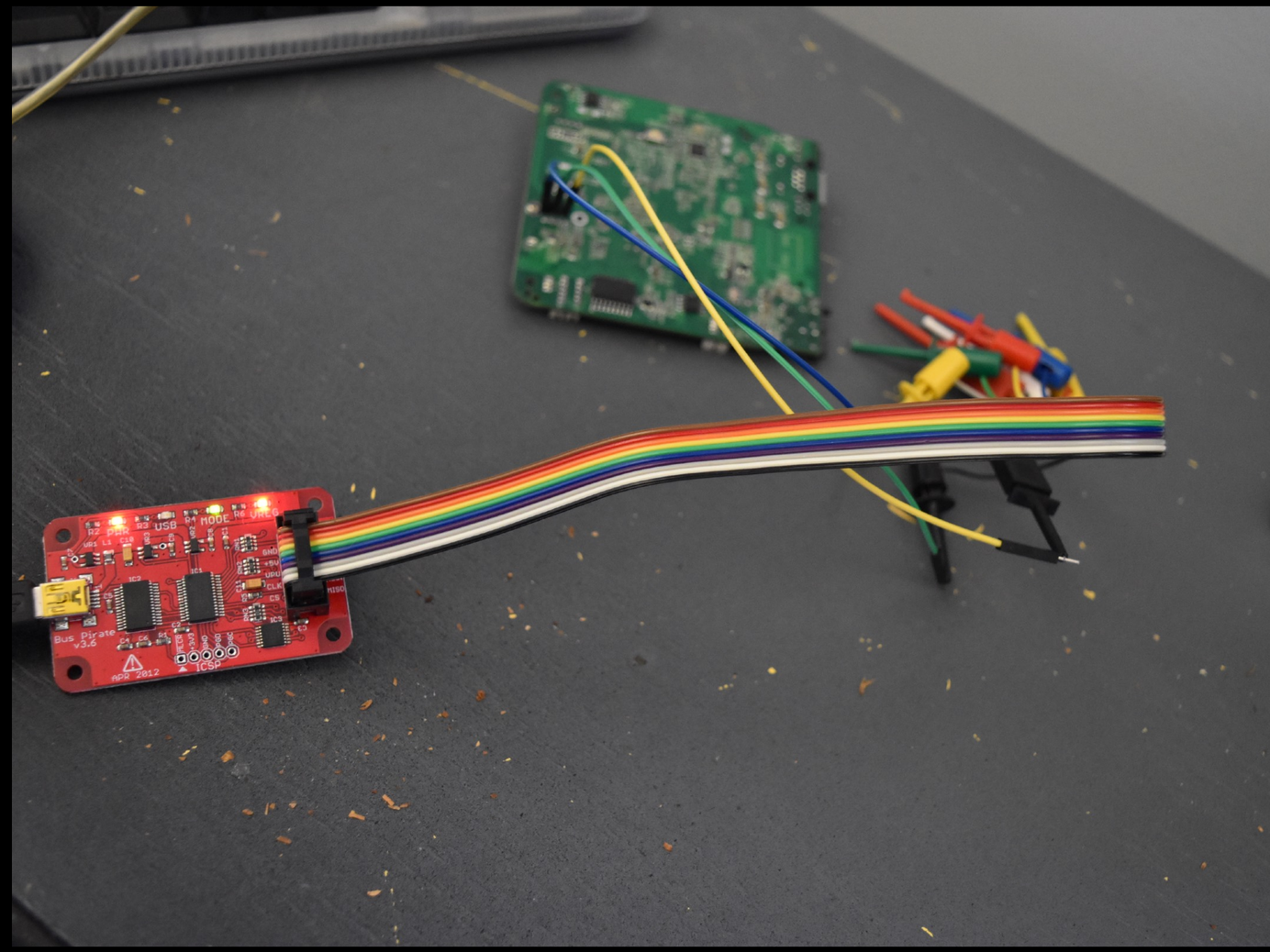
Header pins

Pitch on pins that you use should match the pitch of pins on the board



Serial-to-USB Converter

Serial-to-TTL adapter with 3.3V and 5V settings



Multimeter

For testing continuity between points, verifying pins

Soldering Materials

Soldering Iron

Temperature-controlled, to adjust for different types of solder

Soldering Iron Stand

For supporting the soldering iron

Flux

Heraclitus <3

Solder

Leaded or lead-free

Soldering wick

a copper vacuum for excess solder

Miscellaneous items for soldering workbench

- Kitchen sponge or brass sponge
- Razor or straight edge
- Isopropyl alcohol
- A workbench
- Silicon baking sheets
- A vice grip or a Helping Hands™

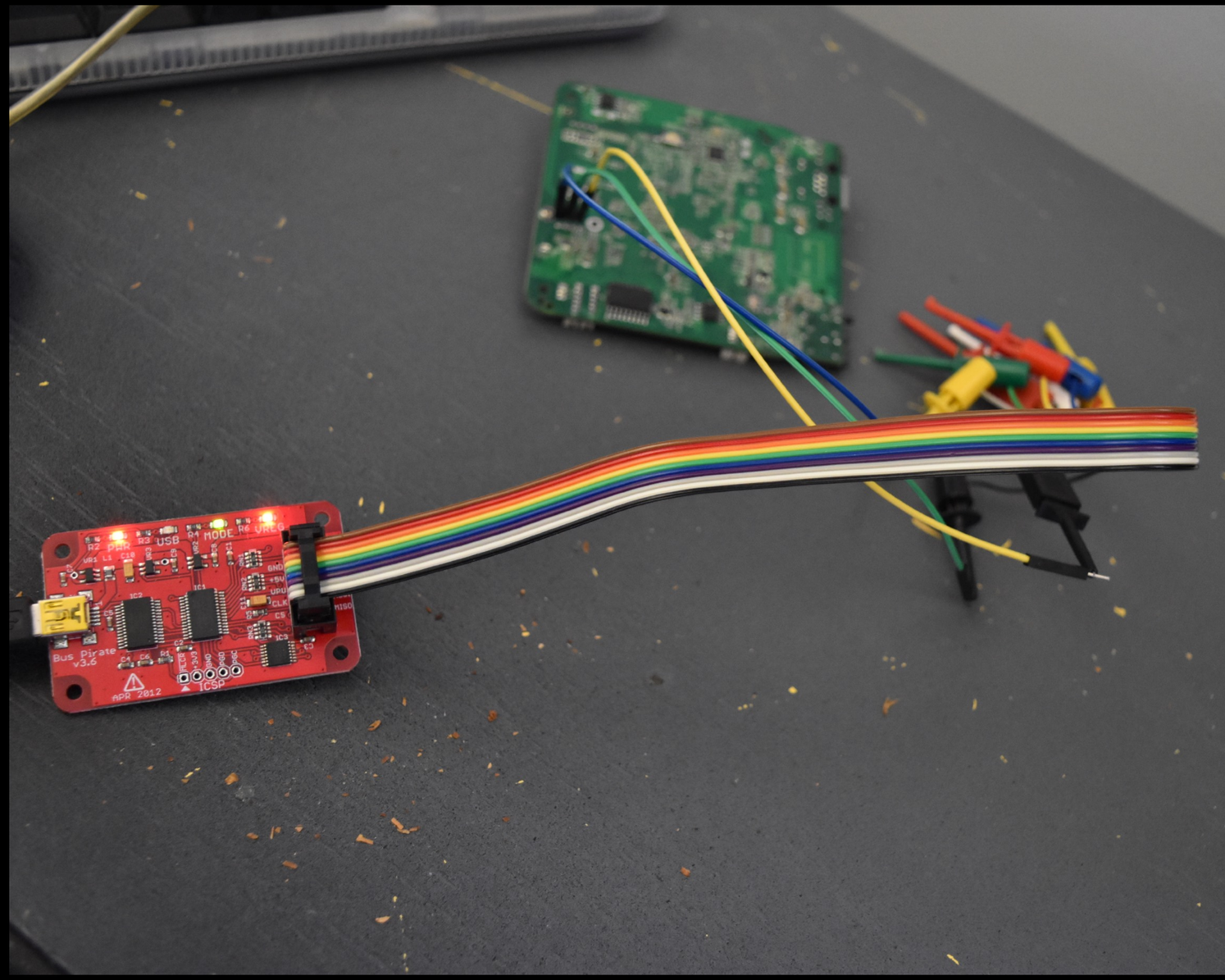
Safety Items

- Safety glasses
- A mask
- A small fan
- An open window, or a suitable outdoor/semi-outdoor space, i.e. a garage


Extra Credit!!

Bus pirate

Interfaces for many of your favorite serial communication protocols: JTAG, SPI, UART, I2C, etc.



Raspberry Pi

Provides a stable environment for running utilities like flashrom

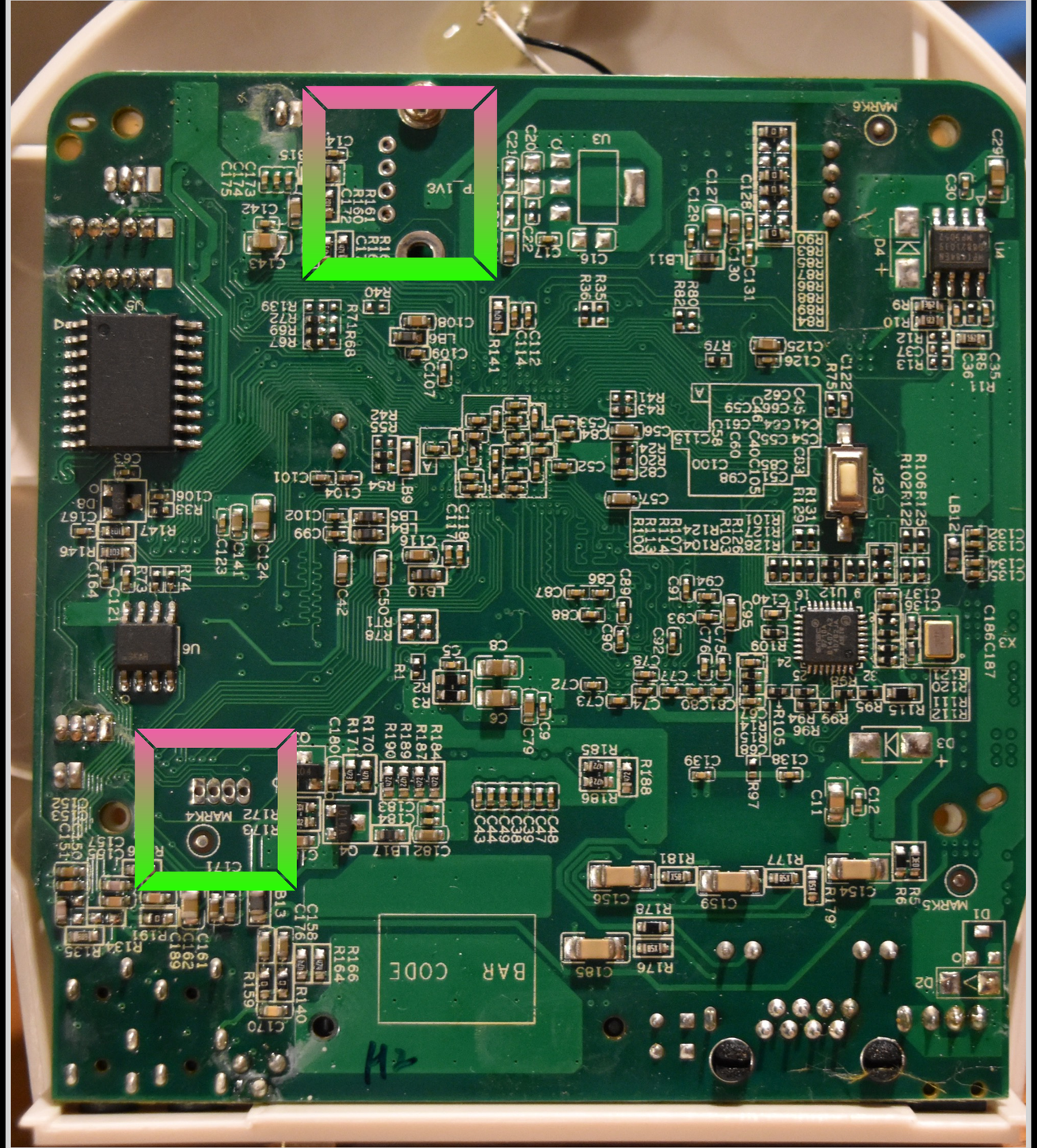
Also a good environment for reverse engineering ARM binaries without the need for an emulator like QEMU

Extra Extra credit if you run your Raspberry Pi with the Kali re4son kernel!

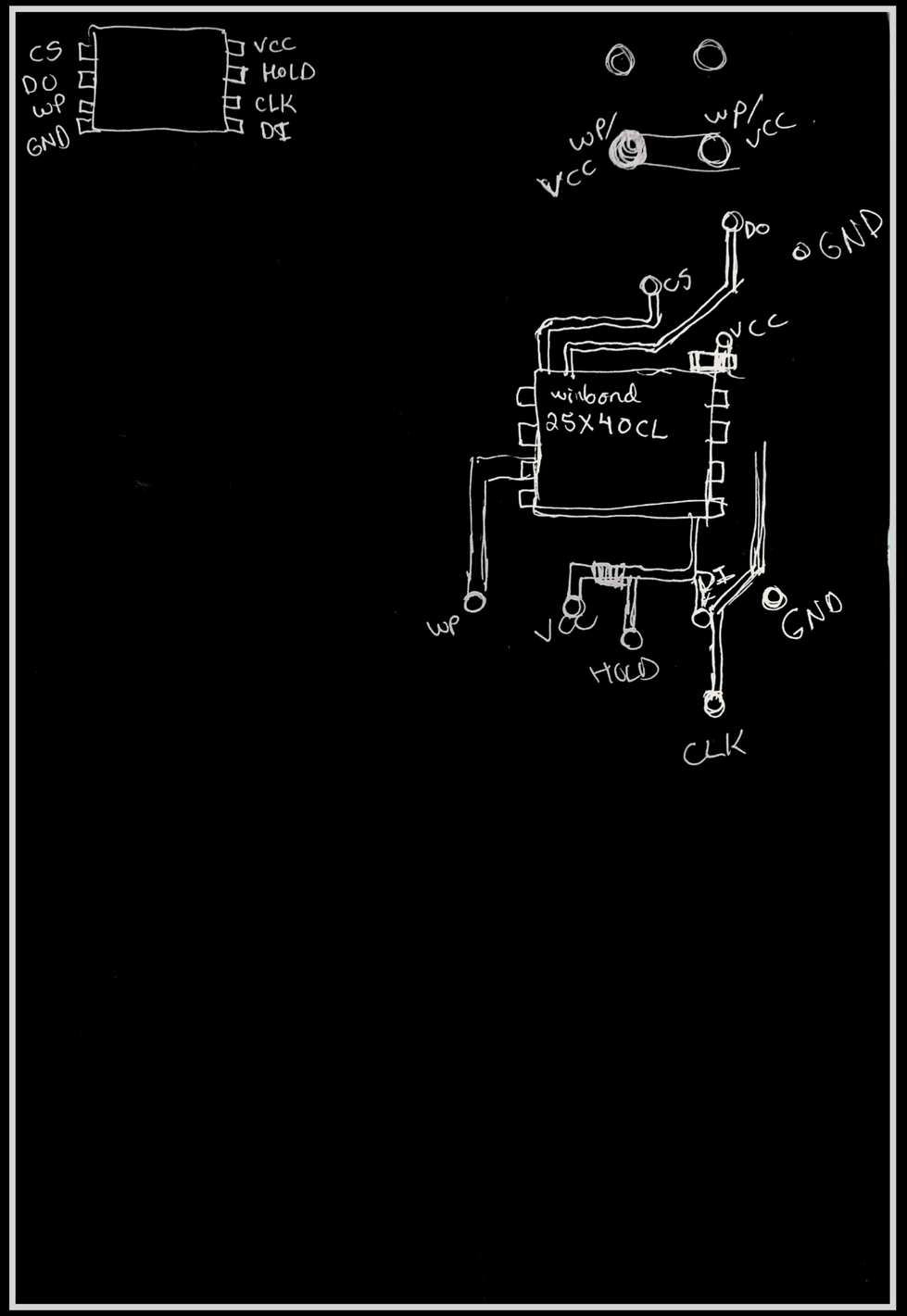

Part 3

Confirm UART headers

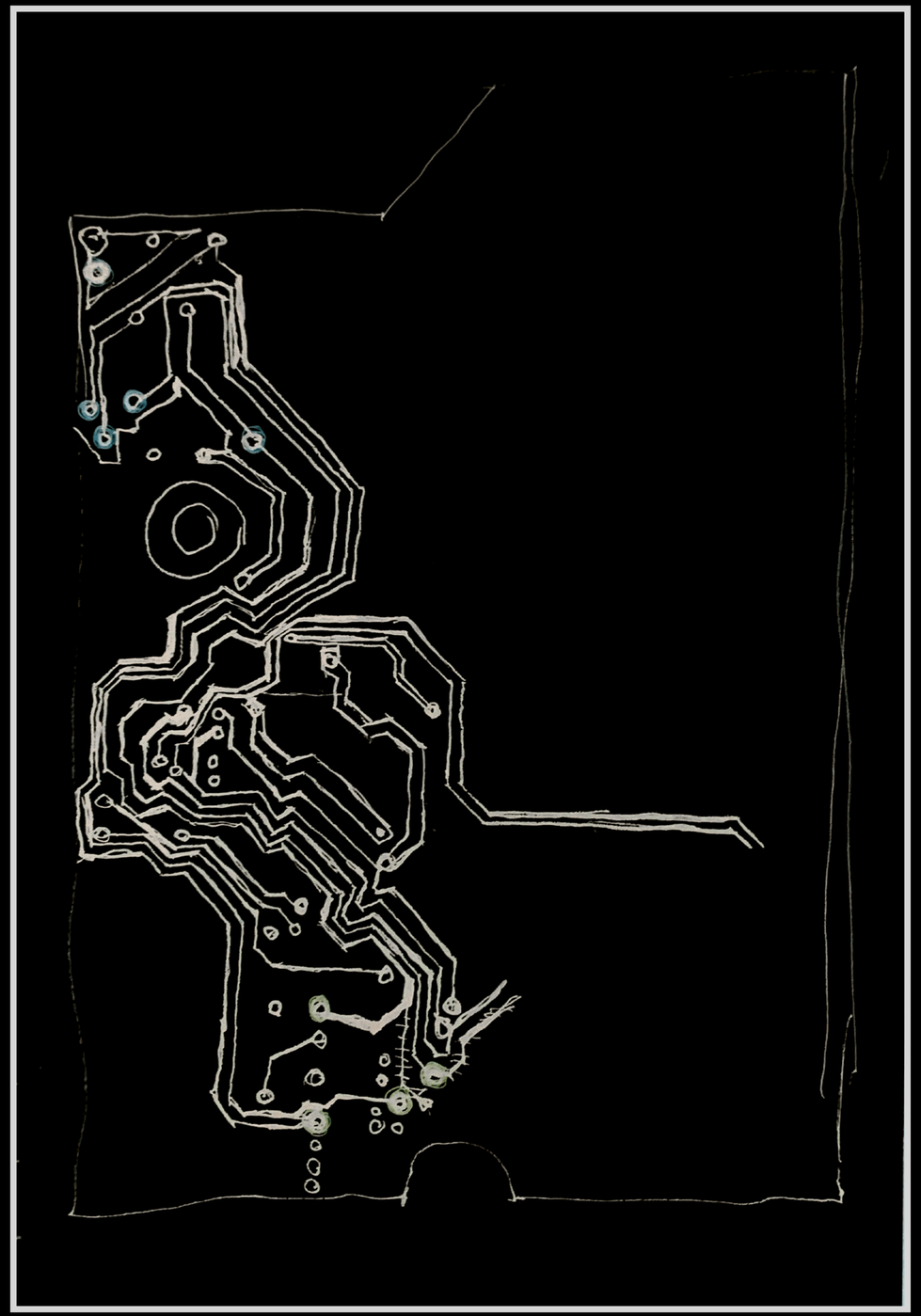
- Confirm TX, RX, GND and VCC pins
- Confirm voltage reading on pins matches specs
- Test with serial console



Drawing identified components and traces on the PCB



Drawing identified components and traces on the PCB



Connect to pins on Flash Chip

- Confirm pinout of chip (CS, SO, WP, GND, VCC, RESET, CLK and SI pins)
- Solder wires to SO, SI, CLK, CS pins
- Attach wires to test clips on Bus Pirate

dump firmware from flash chip with flashrom

```
root@kali:~# flashrom -p buspirate-spi:dev=/dev/ttyUSB0,spispeed=115200 -r flash.bin
```

References

References for this talk

- “Major Vulnerabilities and Exploit Found in Foscam Cameras”
  Or Peles
  VDOO
  https://www.vdoo.com/blog/vdoo-has-found-major-vulnerabilities-in-foscam-cameras

- “decrypt-foscam.py”
  bashis
  Github
  https://github.com/mcw0/PoC/blob/master/decrypt-foscam.py

References

References for this talk continued — Foscam Articles

- “Is This Cam Inviting Hackers into Your Home?”
  Jason Sattler
  F-Secure
  https://blog.f-secure.com/foscam-ip-cameras-insecure-iot/
- “Foscam Issues Patches for Vulnerabilities in IP Cameras”
  Lindsey O’Donnell
  Threatpost
  https://threatpost.com/foscam-issues-patches-for-vulnerabilities-in-ip-cameras/132738/
- “If Your Company Uses Foscam Security Cameras, Patch Them Now to Avoid Critical Vulnerabilities”
  Brandon Vigliarolo
  Tech Republic
  https://www.techrepublic.com/article/if-your-company-uses-foscam-security-cameras-patch-them-now-to-avoid-critical-vulnerabilities/
- “WIP: Fun with the Foscam FI9853EP”
  cliff
  https://oisec.net/articles/2019-03-27/wip-fun-foscam-fi9853ep
- “Multiple Flaws in Foscam IP Cameras Open Devices, Networks to Attackers”
  F-Secure
  https://www.f-secure.com/gb-en/press/p/multiple-flaws-in-foscam-ip-cameras-open-devices-networks-to-attackers
- “Of Cameras and Compromise: How IoT Could Dull Your Competitive Edge”
  Melissa Michael
  F-Secure
  https://blog.f-secure.com/foscam_cameras_and_compromise/
- “This is Why People Fear the Internet of Things”
  Brian Krebs
  KrebsOnSecurity
  https://krebsonsecurity.com/2016/02/this-is-why-people-fear-the-internet-of-things/

References

References for learning about Hardware Hacking/Firmware Analysis/Reverse Engineering

JCJC-dev
“Practical Reverse Engineering Part 4: Dumping the Flash”
https://jcjc-dev.com/2016/06/08/reversing-huawei-4-dumping-flash/

J Michel
“From NAND chip to files”
https://www.j-michel.org/blog/2014/05/27/from-nand-chip-to-files

PenTest Partners
“How to do Firmware Analysis — Tools, Tips and Tricks”
https://www.pentestpartners.com/security-blog/how-to-do-firmware-analysis-tools-tips-and-tricks/

“Extracting Firmware from Microcontrollers’ Onboard Flash Memory, Part 4: Texas Instruments RF Microcontrollers“
Rapid7
https://blog.rapid7.com/2019/05/07/extracting-firmware-from-microcontrollers-onboard-flash-memory-part-4-texas-instrument-rf-microcontrollers/

“Extracting Firmware from Microcontrollers’ Onboard Flash Memory, Part 3: Microchip PIC Microcontrollers“
Rapid7
https://blog.rapid7.com/2019/04/30/extracting-firmware-from-microcontrollers-onboard-flash-memory-part-3-microchip-pic-microcontrollers/

“Emulating and Exploiting Firmware Binaries”
InfoSec Institute
https://resources.infosecinstitute.com/emulating-and-exploiting-firmware-binaries-offensive-iot-exploitation-series/#gref

References

References for learning about Hardware Hacking/Firmware Analysis/Reverse Engineering

FireEye

Embedded Hardware Hacking 101

Gordon Johnson

https://www.fireeye.com/blog/threat-research/2016/08/embedded_hardwareha.html

“Accessing the inaccessible: In a World of Embedded Devices”

CSIAC

https://www.csiac.org/journal-article/accessing-the-inaccessible-incident-investigation-in-a-world-of-embedded-devices/

OWASP

Firmware Security Testing Methodology

https://scriptingxss.gitbook.io/firmware-security-testing-methodology/#stage-1-information-gathering-and-reconnaissance

“What I wish I’d known when I was an Embedded Linux Newbie”

Embedded ARM

https://www.embeddedarm.com/blog/what-i-wish-id-known-when-i-was-an-embedded-linux-newbie/

References

References for learning about Hardware Hacking/Firmware Analysis/Reverse Engineering

“BasicFUN Series Part 1: Hardware Analysis / SPI Flash Extraction”
wrongbaud
https://wrongbaud.github.io/posts/BasicFUN-flashing/

“BasicFUN Series Part 2: Revere Engineering Firmware / Reflashing SPI Flash”
wrongbaud
https://wrongbaud.github.io/posts/BasicFUN-rom-analysis/

“BasicFUN Series Part 4: I2C Sniffing, EEPROM Extraction and Parallel Flash Extraction”
wrongbaud
https://wrongbaud.github.io/posts/Holiday-Teardown/

“Hardware Debugging for Reverse Engineers Part 2: JTAG, SSDs and Firmware Extraction”
wrongbaud
https://wrongbaud.github.io/posts/jtag-hdd/

“Router Analysis Part 1: UART Discovery and SPI Flash Extraction”
wrongbaud
https://wrongbaud.github.io/posts/router-teardown/

References

References for learning about Hardware Hacking/Firmware Analysis/Reverse Engineering

“Hard Disk Hacking”

Sprites Mods

2016

https://spritesmods.com/?art=hddhack

“Writing ARM Assembly”

Azeria Labs

https://azeria-labs.com/writing-arm-assembly-part-1/

“Practical Reverse Engineering: x86, x64, ARM, Windows Kernel, Reversing Tools and Obfuscation”

Bruce Dang, Alexandre Gazet and Elias Bachaalany

Wiley

2014

“Designing BSD Rootkits”

Joseph Kong

O’Reilly

No Starch Press

2007

References

References for learning about Electrical Engineering Fundamentals

“Serial Communication - UART”

SparkFun

https://learn.sparkfun.com/tutorials/serial-communication/uarts

“Serial Communication”

learn - SparkFun

https://learn.sparkfun.com/tutorials/serial-communication

“Integrated Circuits”

learn - SparkFun

https://learn.sparkfun.com/tutorials/integrated-circuits

“PCB (Printed Circuit Board) Basics”

learn - SparkFun

https://learn.sparkfun.com/tutorials/pcb-basics

References

References for learning about Electrical Engineering Fundamentals

“SPI (Serial Peripheral Interface)”

learn - SparkFun

https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi

“Connector Basics”

learn - SparkFun

https://learn.sparkfun.com/tutorials/connector-basics

“Shift Registers”

learn - SparkFun

https://learn.sparkfun.com/tutorials/shift-registers

“How to Use a Breadboard”

learn - SparkFun

https://learn.sparkfun.com/tutorials/how-to-use-a-breadboard

References

References for learning about Electrical Engineering Fundamentals

“What is a Circuit”

learn - SparkFun

https://learn.sparkfun.com/tutorials/what-is-a-circuit

“Voltage, Current, Resistance and Ohms Law”

learn - SparkFun

https://learn.sparkfun.com/tutorials/voltage-current-resistance-and-ohms-law

“How to Use a Multimeter”

learn - SparkFun

https://learn.sparkfun.com/tutorials/how-to-use-a-multimeter

“Series and Parallel Circuits”

learn - SparkFun

https://learn.sparkfun.com/tutorials/series-and-parallel-circuits

“Capacitors”

learn - SparkFun

https://learn.sparkfun.com/tutorials/capacitors