UNLV

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
UNIVERSITY OF NEVADA LAS VEGAS

EE498 Senior Design

Spring 2024

USB Defense

Final Project Report

Group Members:

Sean Yocum	CPE
Name (Print)	CPE/EE/ME
Vince Bengco	CPE
Name (Print)	CPE/EE/ME

Instructor: Dr. Grzegorz Chmaj

Table of contents

Abstract

USB Defense is a project designed to help alleviate the growing security concerns of cyber security through hardware design. Current standards in the industry, at least in areas with sensitive information, is to disable USB usage across the board. Although this solves most security concerns with regards to USB, it does not solve all. This is where the USB Defense comes in. Any USB device, whether it's a flash drive or a cable, will be able to be used with the product. Rather than utilizing a computer to help analyze, a stand-alone device using a microcontroller is used. This is done to prevent any real damage to computers, and even potentially the network it's connected to. So in the worst case scenario only the device would be affected and not any important computers/equipment. Once connected, the device will analyze USB protocols, device descriptions and even vendor IDs to determine whether any malicious intent is discovered. A combination of microcontrollers, fuses, and secure coding standards will be used to help minimize the cost and protect the device from being manipulated itself. The project itself is to help realize how big security concerns are hardware wise, and not just in the cyber realm. Both work in tandem, so security must be the utmost priority when designing circuits. In conclusion, the USB defense is an analysis tool in determining whether the USB drive/cable in question is normal, or whether it has hidden capabilities unbeknownst to the user.

Introduction & background

Welcome to USB Defense, your computer's defender against potential USB threats. An unknown USB device poses several risks, as it could potentially introduce security threats and compromise the safety of your computer or data. When you connect an unknown USB device to your computer, it may automatically execute or install malware, putting your system at risk. Malicious USB devices can be designed to steal sensitive information from your computer, such as personal data, login credentials, or financial information. Our project is designed to reveal the unknown aspects of such USB devices.

Imagine your USB Defense system as a bouncer at a club for your computer. When a new USB device tries to get access to your computer's door, USB Defense acts like a bouncer with a list of guests' information. It carefully checks the USB device for any hidden tricks or bad intentions. If everything looks good, it gives the USB device a green light, just like a traffic light saying it's safe to proceed. But if USB Defense senses something fishy or potentially harmful, it immediately warns you with a message, helping you keep your computer safe and sound. Remember, USB Defense is like a guardian angel for your computer, making sure only the good USB devices get in, while keeping the potential troublemakers out!

Once our device is turned on it requests confirmation of possible selected devices to scan. When a USB device is plugged into the USB Defense, the system initiates the scanning process. The system examines the USB device for patterns or behaviors commonly associated with malicious intent. The Malicious Intent Detector analyzes the collected data and makes a decision on the safety of the connected USB device. If the system identifies a potential threat, it immediately notifies the user, providing guidance on disconnecting the USB device.

One major device that performs similar functions is the Malicious Cable Detector by O.MG [1]. As the provider of many malicious devices O.MG has also included an item called Malicious Cable Detector that requires additional hardware in the form of an external usb port plug. The USB Defense is a standalone device that accepts basic USB-A devices and determines the use cases of unknown USB devices.

Current Market Solutions

In the market there are homebrew devices that require a Raspberry Pi to function [2]. Most Raspberry Pi's range from $25 to $80. There aren't that many devices in the market that are stand alone without the need of a Raspberry Pi. However there still are such devices as the Malicious Cable Detector by O.MG that does not use the Raspberry Pi. As the most readily available device for purchase as a plug and play device the Malicious Cable Detector is the best known.

	Vendor	Function 1	Function 2	Price	Comment
CIRClean	Homebrew	Copies untrusted files from malicious USBs and stores on a good USB		$35+Installation	Needs to be ordered and manufactured by user
Malicious Cable Detector	O.MG	Detects all known malicious USB cables	Functions as a Data Blocker	$39.99	Easy to buy
USB Guardian	Homebrew	Detect and remove malware from USB sticks		$85+Installation	Needs to be ordered and manufactured by user
USB Defense	(Our Device)	Detects whether USB cable and accessories are malicious on a stand alone device. No computer necessary		$32	Could be cheaper with mass production

Table 1. Comparison of available devices

	Resolution	Battery life	Price	Strengths	Weaknesses
CIRClean	Single LED	Plugged In	Affordable	• Multifunctional Device	• Homebrew, needs some technical knowledge • Only copies files
Malicious Cable Detector	Single LED	Plugged In	Affordable	• Ready made • Small	• Sold by same seller as Malicious Device • Requires Power
USB Guardian	480 x 800	Plugged In	High	• Multifunctional Device • User Interface	• Homebrew, needs some technical knowledge
USB Defense	340 x 240	8 hrs	Affordable	• Standalone • User Interface	• Not available • Still has bugs • Requires User Input

Table 2. Strengths and weaknesses of available devices

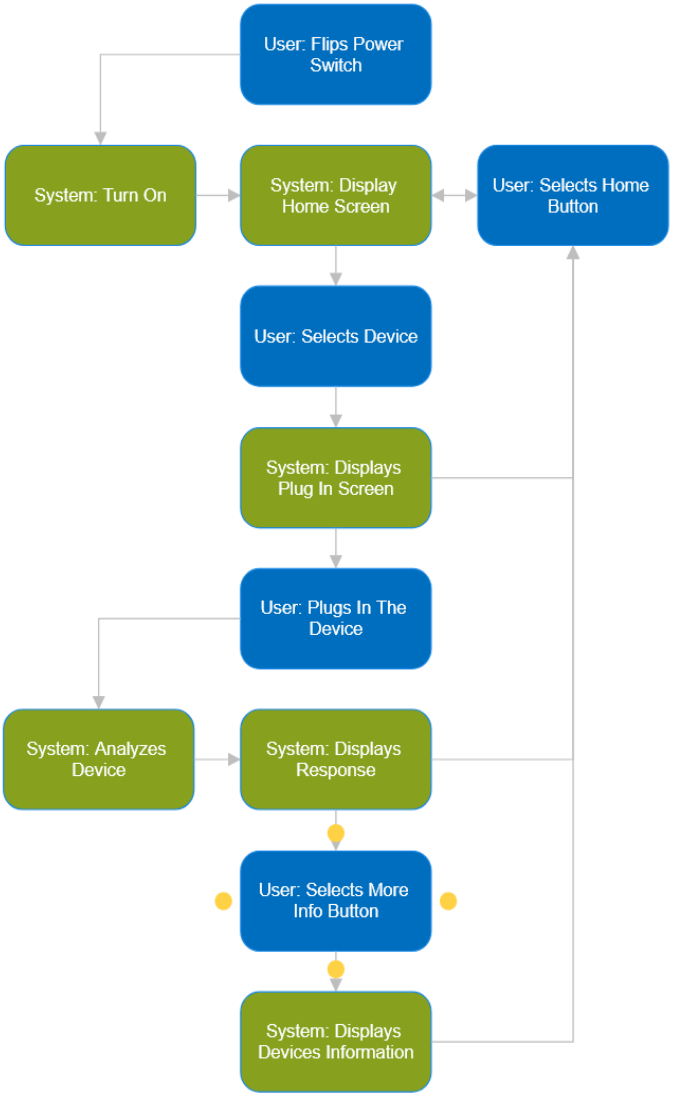
Research results

In order to detect maliciousness of a device/cable, research must be done on the device itself. USB protocols are the first thing to analyze when trying to determine whether a device has any bad intentions. In our case, the USB Defense will act as a host, and anything connecting to it will be a device so it will be in host-device configuration. Once a USB device has been plugged into the host, the USB enumeration process begins. Essentially what that means is that the host will detect the presence of a device, and determine what the device is based on the descriptors the device yields.

However, cables such as the O.MG cable can yield improper enumeration, and pretend to essentially be a broken cable. Upon further research on the O.MG cable, the active end can only be connected to 5V(max) in order for pass-through charging to work. Using this knowledge, manipulating certain voltage, current and power requirements can be used on the Raspberry Pi or PCB to help determine whether cable is good or not.

Specification of the project

Functionality & conceptual design:



Step 1: Actor flips the power switch, and goes to the next step.

Step 2: System starts up, and goes to the next step..

Step 3: System displays home screen, and goes to the next step.

Step 4: Actor selects the USB device they will analyze, and goes to the next step
or Select Home Button, and goes to the 3rd step.

Step 5: Actor plugs the USB device in.
or Select Home Button, and goes to the 3rd step.

Step 6: System scans selected USB device, go to next step

Step 7: System makes decisions, and goes to the next step.

Step 8: System displays decisions with possible options, and goes to the next step.

Step 9: Actor selects More Info Button, go to next step
or Select Home Button, and goes to the 3rd step.

Step 8: System displays devices Information.

Step 9: Actor Select Home Button, and goes to the 3rd step.

At any point if the Actor flips the power switch Device will turn off.

Architecture:

Control Unit

-Microcontroller(RP2040)[3]

-Memory

-Basic Component

USB-A Port(Device)

-Reads USB capable device

-Protects against USB capable device

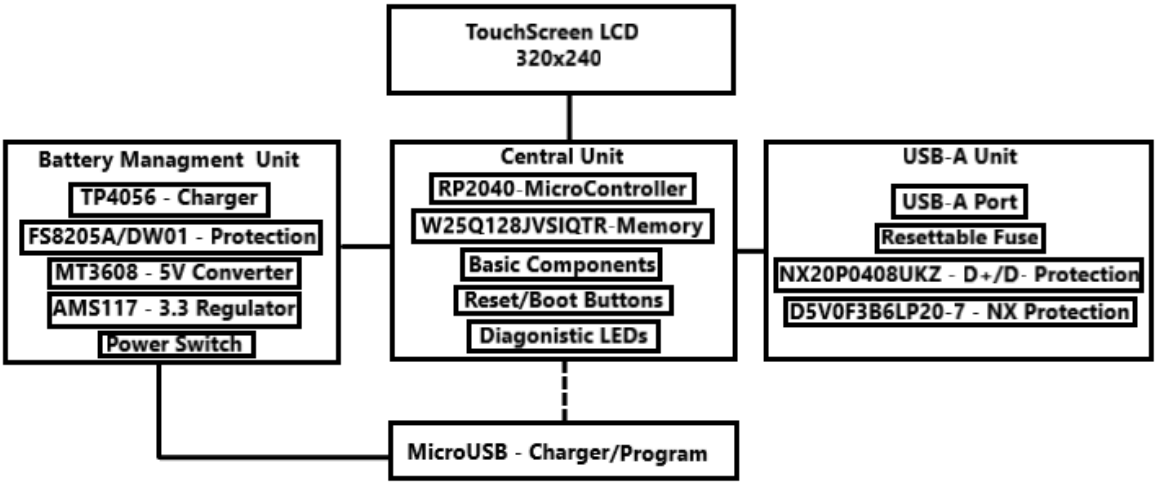
LCD Touch Screen

-Outputs results to LCD Screen

-Enables inputs to send to the Raspberry Pi

Charging Port/Lipo Battery

-Charges and powers on the Raspberry Pi



Design:

RP2040 Board

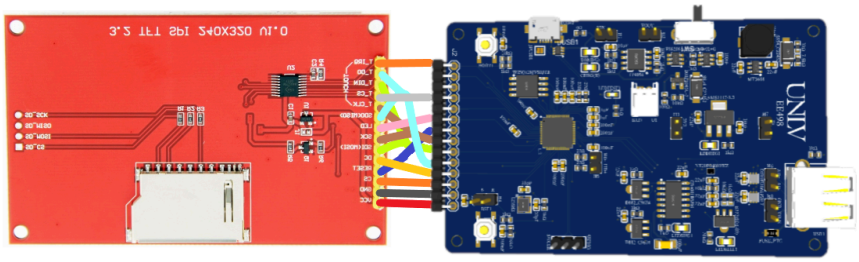
- Connected to USB-A Port using 5V, GND, D+, and D-
- Connected to LCD Touch Screen using header pins
- Programmed using libraries TFT_eSPI.h, FS.h, pio_usb.h and Adafruit_TinyUSB.h.

USB-A Port(Device)

- Connected to RP2040 Board using 5V, GND, D+, and D-

LCD Touch Screen

- Connected to RP2040 Board using header pins



Test plan

	Inputs / actions to the device	Expected output	Comment
Test case #1	* Reading Rubber Ducky Turn on device Select USB Device Plug in Rubber Ducky Receive result Turn off device	Detection of Malicious device	Plan to still function after the test.
Test case #2	* Reading Regular USB Turn on device Select USB Device Plug in Regular USB Receive result Turn off device	No Detection of Malicious device	
Test case #3	* Reading O.MG Cable Turn on device Select USB Device Plug in Rubber Ducky Receive result Turn off device	Detection of Malicious device	Plan to still function after the test.
Test case #4	* Reading Regular Cable Turn on device Select USB Device Plug in Regular Cable Receive result Turn off device	No Detection of Malicious device	

Table 3. Test plan

Current form of a project

The current form of the project consists of the designed board and an LCD Module. The board has a working LiPo Battery Circuit that converts to both 5V for the USB-A port and 3.3V for the rest of the board. The board is separated and connected to the LCD module by multiple M2.5 Hex Brass Spacer Standoffs and screws using the holes on each corner. Although the device is in one package it does not have an outer case. When it comes to the code there is a basic framework for the USB Defense and Touchscreen. In the case of USB Defense to obtain the descriptors of any plugged-in device to the USB-A Ports, the library Adafruit TinyUSB with a change to the setup header files to match our devices pin layout[4]. When coding for the Touchscreen the TFT_eSPI library was used however the button setup was not in favor of a custom button code to use images[5]. There are quite a few problems in both hardware and software.

On the hardware side the connection to the LCD module does not work properly as it requires different pins. To get it to work properly there are jumper wires connecting the lcd pins to the correct header pins which can be found in “PCB Traces”. Another major issue is when ordering from JLCPCB the components used for the USB-A protection circuit were not available. There were attempts to solder by hand however it turned out futile, so there are two pairs of jumper wires for the 5V-GND and D+D- to connect them to the correct pins. As they were not placed there is no protection against the USB Killer. As well as a reverse pins for the battery connections. This was skipped over using a dongle that swapped the pins leading to a longer cable from the board to the battery.

On the software side there is only one device that was programmed to analyze the different devices however it can only analyze a regular USB Cable after powering on as it will act as the previous device. Sadly after multiple revisions the partially working code was lost. While the other codes cannot analyze the O.MG Cable given its difference in which it operates. At this moment in time there is only code for Mass Storage Devices, Hubs, and partial cables. To work around this limitation there was an added button to print the descriptors of any other device that was plugged in.

Roles & skills in the project

	Required skills	
Role 1	Hardware	**Role 1**
Role 2	Software	**Role 2**
Microcontroller programmer	▪ Knowledge of microcontroller ▪ C++ programming ▪ AVR Studio experience	▪ **Microcontroller programmer**
USB programmer	▪ Knowledge of USB Protocol ▪ Knowledge to read USB	▪ **USB programmer**
PCB designer	▪ KiCad skills sufficient to design the PCB	▪ **PCB designer**

Table 4. Roles & skills

List all the roles mentioned in Table 3 and assign names of team member to each role:

	Assignment
Role 1	Sean Yocum
Role 2	Vince Bengco
Microcontroller programmer	Main: Vince Bengco, Support: Sean Yocum
USB programmer	Main: Vince Bengco
User Interface programmer	Main: Sean Yocum
PCB designer	Main: Sean Yocum

Table 5. Roles assignment

ITEM	Name
EE497 Modules	Vince Bengco, Sean Yocum
EE497 Topic Proposal (Full)	Vince Bengco, Sean Yocum
EE497 Topic Proposal (Signed)	Sean Yocum
EE497 Part List	Sean Yocum
EE497 Progress Report	Vince Bengco
EE497 Modules Declaration	Vince Bengco, Sean Yocum
EE497 slides presentation	Sean Yocum
EE497 final report	Vince Bengco, Sean Yocum
EE498 Timeline	Vince Bengco
EE498 Parts List	Vince Bengco
EE498 Progress Report	Vince Bengco
EE498 Final Report	Vince Bengco, Sean Yocum
EE498 Abstract	Vince Bengco, Sean Yocum
Research regarding O.MG Cable	Vince Bengco
Research regarding Rubber Ducky	Vince Bengco
Research regarding Arduino USB Host	Vince Bengco, Sena Yocum
Research regarding Arduino Touchscreen/Display	Sean Yocum
Schematic design	Sean Yocum
PCB design	Sean Yocum
PCB soldering	Sean Yocum
Program Malicious Analysis	Vince Bengco
Program User Interface	Sean Yocum
Combine Programs	Vince Bengco, Sean Yocum

Table 6. Items done

Parts list

	Quantity	Manufacturer	Price	Picture	Att. id
3.2" TFT LCD with Resistive Touch	**1**	**Hosyond**	**15.99**		Attachment 1
O.MG Cable	1	Hak5	119.99		
USB Rubber Ducky	1	Hak5	79.99		
USB Killer	1	USBKill	88.00		
3.7 Lipo Battery	1	YDL	8.99		
R0603	1	UNI-ROYAL(厚声)	0.001		Attachment 2
SOT-363_L2.0-W1.3-P0.65-LS2.1-BR	1	onsemi(安森美)	0.162		Attachment 3
SW-TH_SK12D07VG3	1	SHOU HAN(首韩)	0.02		Attachment 4
HDR-TH_2P-P2.54-V-M	8	kinghelm(金航标)	0.017		Attachment 5
HDR-TH_3P-P2.54-V-M	1	kinghelm(金航标)	0.016		Attachment 6
C0603	2	SAMSUNG(三星)	0.006		Attachment 7
R0603	6	UNI-ROYAL(厚声)	0.001		Attachment 8
CONN-TH_S2B-PH-K-S-LF-SN	1	JST	0.027		Attachment 9
C0603	13	YAGEO(国巨)	0.002		Attachment 10
R0603	14	UNI-ROYAL(厚声)	0.001		Attachment 11
R0603	4	UNI-ROYAL(厚声)	0.001		Attachment 12
SOIC-8_L5.3-W5.3-P1.27-LS8.0-BL	1	WINBOND(华邦)	0.574		Attachment 13
SMA_L4.4-W2.8-LS5.4-R-RD	2	MDD	0.041		Attachment 14
SOT-23-6_L2.9-W1.6-P0.95-LS2.8-BR	1	FUXINSEMI(富芯森美)	0.06		Attachment 15

SOT-89-3_L4.5-W2.5-P1.50-LS4.2-BR	2	CJ(江苏长电/长晶)	0.059		Attachment 16
SOT-89-3_L4.5-W2.5-P1.50-LS4.0-BR-1	1	CJ(江苏长电/长晶)	0.062		Attachment 17
C0603	2	SAMSUNG(三星)	0.004		Attachment 18
C0603	12	SAMSUNG(三星)	0.008		Attachment 19
MICRO-USB-SMD_U253-051N-4BH89-S2S	1	XKB Connectivity(中国星坤)	0.059		Attachment 20
SW-SMD_4P-L5.1-W5.1-P3.70-LS6.5-TL-2	2	XKB Connectivity(中国星坤)	0.016		Attachment 21
C0603	3	SAMSUNG(三星)	0.017		Attachment 22
C1206	1	SAMSUNG(三星)	0.062		Attachment 23
U-DFN2020-6_L2.0-W2.0-P0.65-LS2.0-BL-EP	2	DIODES(美台)	0.257		Attachment 24
PTC0805	1				Attachment 25
7*7*4 - 22 uH	1	LCSC	0.165		Attachment 26
LED0805-R-RD	1	国星光电	0.012		Attachment 27
LED0805-R-RD	1	KENTO	0.013		Attachment 28
LED0805-R-RD	1	KENTO	0.012		Attachment 29
LED0805-R-RD	1	国星光电	0.012		Attachment 30
LED0805-R-RD	1	KENTO	0.013		Attachment 31
LED0603-R-RD_BLUE	1	EVERLIGHT	0.02		Attachment 32
SOT-23-3_L2.9-W1.3-P1.90-LS2.4-BR	1	Diodes Incorporated	0.087		Attachment 33
R0603	1	UNI-ROYAL(厚声)	0.001		Attachment 34
R0603	1	UNI-ROYAL(厚声)	0.001		Attachment 35
R0603	1	UNI-ROYAL(厚声)	0.001		Attachment 36
LQFN-56_L7.0-W7.0-P0.4-EP	1	Raspberry Pi(树莓派)	0.991		Attachment 37
ESOP-8_L4.9-W3.9-P1.27-LS6.0-BL-EP	1	TOPPOWER(南京拓微)	0.168		Attachment 38
SOT-23-6_L2.9-W1.6-P0.95-LS2.8-BR	1	FORTUNE(富晶)	0.1		Attachment 39
SOT-23-6_L2.9-W1.6-P0.95-LS2.8-BL	1	西安航天民芯	0.071		Attachment 40
SOIC-14_L8.7-W3.9-P1.27-LS6.0-BL	1	TI(德州仪器)	0.259		Attachment 41
WLCSP-12_L1.7-W1.3-R4-C3-P0.40-TL	1	NXP(恩智浦)	1.62		Attachment 42
SOT-223-3_L6.5-W3.4-P2.30-LS7.0-BR	1	AMS	0.132		Attachment 43
OSC-SMD_4P-L3.2-W2.5-BL	1	YXC	0.058		Attachment 44
USB-A-TH_USB-231-ARY	1	XUNPU(讯普)	0.125		Attachment 45

1X14 Header Pin	1				Attachment 46

Table 7. List of required parts

Engineering standards used in the project

The following engineering standards have been used for the project:

Number (reference)	Symbol	Description	Justification why this standard has been used
1	IEC 63181	LCD Display	Specifies the functional components for the LCD and describes their structure and signal flow
2	IEC 62680-2-1:2015	USB compliant standards for compatibility	The industry standard for USB, so that OEMs and other developers can utilize similar interfaces and not lose any compatibility.
3	IEEE 1625-2008	Lipo Battery	Establishes the criteria for mobile computing devices

Table 8. List of engineering standards

Engineering constraints for the project

The following engineering constraints have been used for the project:

#	Name	Description	Specification of the constraint
1	Power Consumption	Maximum Power Drawn by the Device	5W ref. 3, 5
2	Size	Maximum Device Size	Case Size 3.8x2.8x1 inches
3	Resolution	Minimum Screen Resolution	240 x 320 resolution ref. 1
4	Input Voltage	Maximum Voltage from USB Port	5V ref. 4, 5

Table 9. List of engineering constraints

User Manual

USB DEFENSE

Instructions: Use of Device

Members: Sean Yocum

Vince Bengco

User Manual

Step 1:

IMG

Turn on Power Switch

This I/O switch turns the USB Defender on and off "I" to turn on and "O" to turn off.

Step 2:

IMG

Select Desired Device to Analyze

On the screen there is six touch buttons indicated by their respective devices.

Step 3:

IMG

Plug Device Into Input Port

Plug device into designated port 1 to initialize analysis.

USB DEFENSE

Location: Address or Room Number

Members: Sean Yocum

Vince Bengco

User Manual

Step 4:

Wait

Please do not turn off the device.

Step 5:

Check Device Information

Select the "More Info" Button

Step 6:

Repeat or Turn Off

If you wish to analyze another device, press "Home" Button and go back to Step 2.

Else, Turn off Power Switch

Development Manual

RP2040 DEVELOPMENT BOARD

Instruction: Programming of Device

Members: Sean Yocum

Arduino IDE Manual

Step 1:	Open Arduino IDE
Step 2:	Go To Tools/Board:/Board Manager Or Ctrl + Shift + B
Step 3:	Install “Raspberry Pi Pico/RP2040” by Earlephilhower
Step 4:	Go To File/Preferences Or Ctrl + Comma
Step 5:	In Additional boards manager URLs: Paste Below https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json

RP2040 DEVELOPMENT BOARD

Instruction: Programming of Device

Members: Sean Yocum

Arduino IDE Manual

Step 6: Go To Sketch/Include Library:/Manage Libraries

Or Ctrl + Shift + L

Step 7: Install “Adafruit TinyUSB Library” by Adafruit with any and all additional Libraries

Step 8: Install “Pico PIO USB” by skeigongonnoc with any and all additional Libraries

Step 9: Install “TFT_eSPI” by Bodmer with any and all additional Libraries

Step 10: Go To Tools/Board:/Raspberry Pi Pico/RP2040

RP2040 DEVELOPMENT BOARD

Instruction: Programming of Device

Members: Sean Yocum

Arduino IDE Manual

Step 11: Select “Raspberry Pi Pico”

Step 12: Go To Tools/CPU Speed:

Step 13: Select “120 MHz”

Step 14: Go To Tools/USB Stack:

Step 15: Select “Adafruit TinyUSB”

RP2040 DEVELOPMENT BOARD

Instruction: Programming of Device

Members: Sean Yocum

Arduino IDE Manual

Step 16: On Your Computer Go To Documents/Arduino/libraries/TFT_eSPI/User_Setup.h

Step 17: Commet out the TFT pins

Replace with

```
//USB DEFENSE SETUP - PCB Setup
#define TFT_MISO 12 //7
#define TFT_MOSI 11
#define TFT_SCLK 10
#define TFT_CS   13  // Chip select control pin
#define TFT_DC    14  // Data Command control pin
#define TFT_RST   7  // Reset pin (could connect to RST pin)
#define TOUCH_CS 5     // Chip select pin (T_CS) of touch screen
```

Uncomment

```
#define TFT_SPI_PORT 1
```

Step 18: To Use LCD Screen

In your code set PIN 8 to High

Step 19: To Use USB-A Port

In your code write

```
#define PIN_USB_HOST_DP  16
#define PIN_5V_EN        18
```

Final remarks

Future implementations/improvements

-Wifi Module/Network Interface Card (NIC)
    -Detect new wifi signals upon inserting a USB
    -Secure IoT device
        -Originally considered for our design, was scrapped due to security concerns
        -More than 77% have an increase in attacks on IoT devices in 2022
        -More than 25% of all cyber attacks against businesses involve IoT
-More subsequent algorithms
    -Abnormal D+/D- Bit Detection
    -Search for possible MAC address
        -Essentially a serial number of the NIC
        -Can be spoofed so it can be logged/analyzed for later use
    -User defined time limit/scan
        -Specify how long to scan the device for changes in descriptors such as PID/VID
    -Allow logging of data to compare between scans
-Expand on USB to other peripherals
    -E.g. SD cards, NVRAM
    -Essentially a UTM(Unified Threat Management) device

PCB Traces

PCB Label	LCD Screen
T_CS	T_CS
LCD_MISO	RST
LCD_SCK	SCK + T_CLK
LCD_MOSI	MOSI + T_DIN
LCD_DC	MISO + T_DO
LCD_RST	CS
LCS_CS	DC
3V3	VCC
GND	GND
LCD_LED	LCD_LED

Table 10 LCD Jumper Wire Placement

The RP2040 has two SPI modules on it, unfortunately the traces for the LCD touchscreen are not 1 to 1. So in order to work around the issue we crimped Y-splitters according to where the traces were coming from. Table 10 shows the connections that need to be made in order for the LCD and the touchscreen itself to work properly with the PCB.

References

[1] Hak5. “Hacking Gear & Media | Hak5 Official Site.” Hak5, hak5.org/.

[2] Ltd, Raspberry Pi. “Buy a Raspberry Pi Pico.” *Raspberry Pi*, www.raspberrypi.com/products/raspberry-pi-pico/.

[3] “Raspberry Pi Documentation - RP2040.” *www.raspberrypi.com*, www.raspberrypi.com/documentation/microcontrollers/rp2040.html.

[4] “Adafruit TinyUSB Library - Arduino Reference.” *www.arduino.cc*, www.arduino.cc/reference/en/libraries/adafruit-tinyusb-library/. Accessed 7 May 2024.

[5] “TFT_eSPI - Arduino Reference.” www.arduino.cc, www.arduino.cc/reference/en/libraries/tft_espi/.