ClassTouch™

ClassTouch Reference Guide

Revision 1.2

A2D Fitness Systems, Inc.

ClassTouch™ Pi Enclosure

Contents

ClassTouch™ Pi Enclosure

Revision History

Revision Date	Description	Revision
May 2019	Initial draft	1.0
January 2020	Edits for JX-V2959-UHV Display Driver	1.1
November 2020	Changes to incorporate Raspberry Pi 4	1.2

Welcome

Thank you for purchasing ClassTouch™, the most complete Raspberry Pi Touch Screen Enclosure System available. The ClassTouch™ system was developed for the health and fitness industries after many years of using windows based all-in-one PC's and thermal-formed enclosures. These systems were expensive to build and didn't really look that great. With the introduction of the Raspberry Pi, inexpensive robust computing power became readily available for around $35, and because of the Raspberry Pi's accessory base, integrating them into real-world products in the DIY and consumer markets has become common place.

The challenges that we faced developing touch screen solutions for use in the real-word were significant, plus the fact that cost is always a concern made some requirements impossible to have. Having a good single-board-computer, (SBC) is just the start; you will also need a robust Operating System and preferably one that doesn't include licensing fees. You will also need a touch screen that is large enough to be seen and interacted with by the intended users; built-in speakers and audio capabilities are also required. You will probably need Wi-Fi connectivity to eliminate the need for network cables and lastly, all of the components will need to be housed in a practical yet stylish enclosure.

ClassTouch™ is a brand new product that will allow you to integrate your Raspberry Pi into a system that can be used in industrial, consumer and special projects with the very minimum of setup time, reducing your product's total time-to-market significantly. We have integrated many of the features that you would want and need in a system using the Raspberry Pi and if you need to include your own custom electronics, ClassTouch™ has a large area inside the enclosure that is designed specifically to house custom electronics and keep them protected and secure from the outside world.

Read on to learn more about the ClassTouch™ system.

ClassTouch™ Pi Enclosure

Enclosure Specifications

The ClassTouch enclosure is a beautiful two part injection molded design that's both lightweight and strong. The color is paint-less matt black with a subtle textured finish. We use pigmented colored plastics to reduce the appearance of surface scratches, additionally pigmented plastics hold up better to normal wear. Both the left and right sides of the enclosure sports a Female USB-A port that can be connected to your Raspberry Pi internally if desired with the included USB cables. If external USB ports are not required, then two cover panels are provided that can be used to conceal one or both of the ports.

- Overall size in Millimeters: 264 x 300 x 50 (W:H:D)
- Front panel display opening size in Millimeters: 213 x 160 (W:H)
- 2 External USB ports
- 2 External USB port cover panels
- 1 Mounting system for 100mm and 75mm VESA
- 1 Rear air vent
- 2 Rear speaker grills
- 1 Internal covered compartment used for custom electronics
- Internal compartment size in Millimeters: 200 x 85 x 28 (W:H:D)
- 1 Removable compartment cover panel with 1 x 38mm (1 ½") hole used for cable grommet (not included)
- Mounting for internal Wi-Fi Antenna of up to 125mm long is supported

ClassTouch™ Pi Enclosure

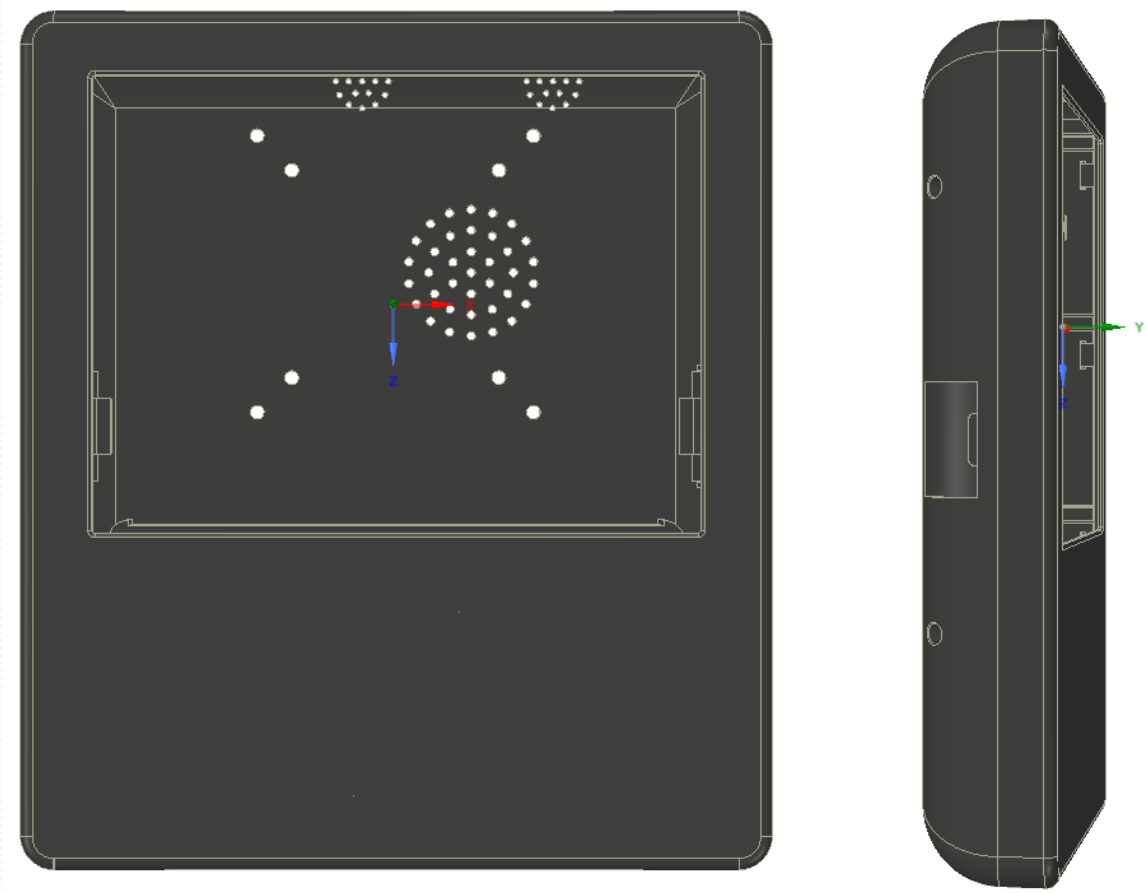


Figure 1 – Enclosure Front and Left Side Elevation

ClassTouch™ Pi Enclosure

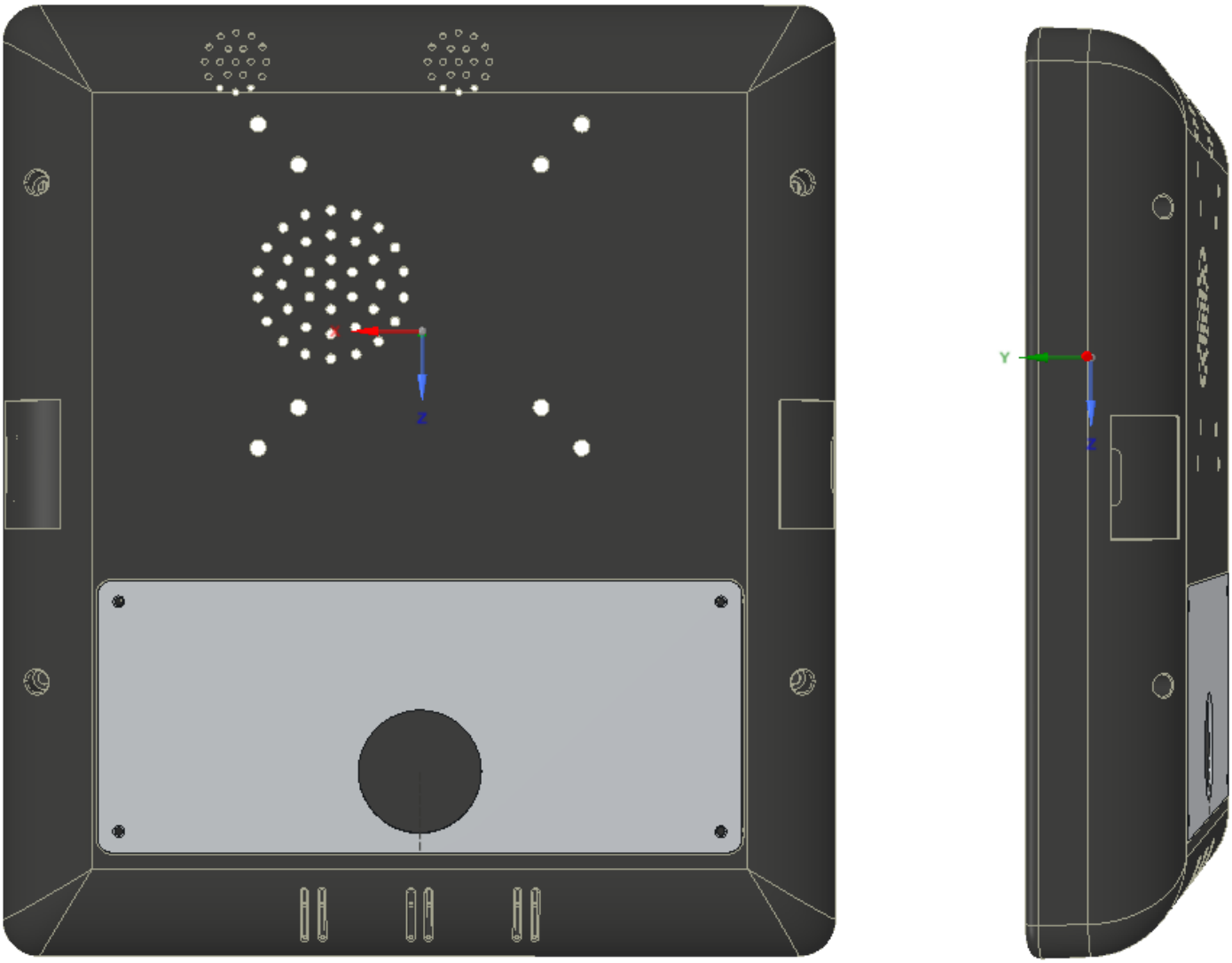


Figure 2 – Enclosure Rear and Right Side Elevation

ClassTouch™ Pi Enclosure

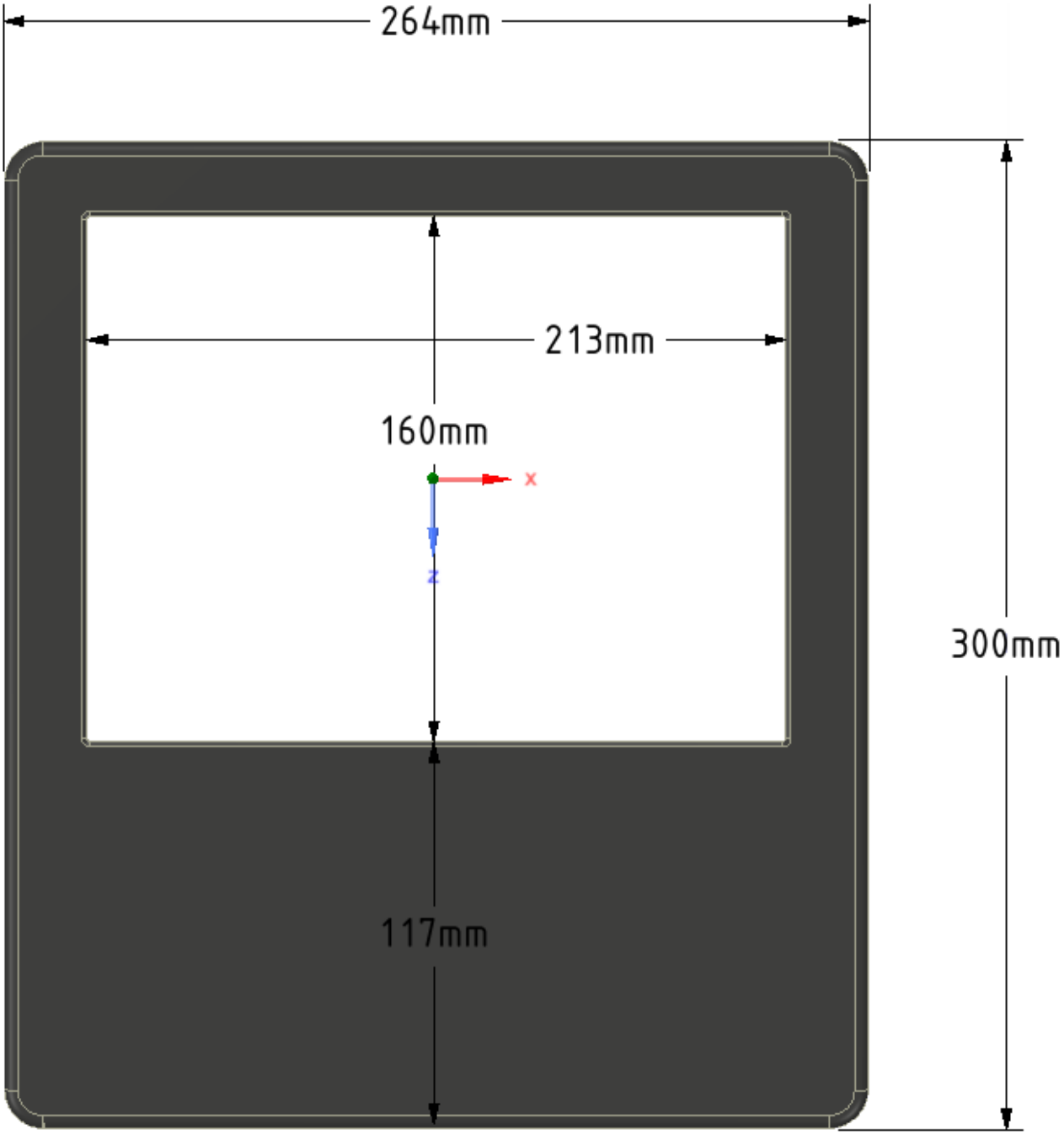


Figure 3 – Enclosure Front Dimensions

ClassTouch™ Pi Enclosure

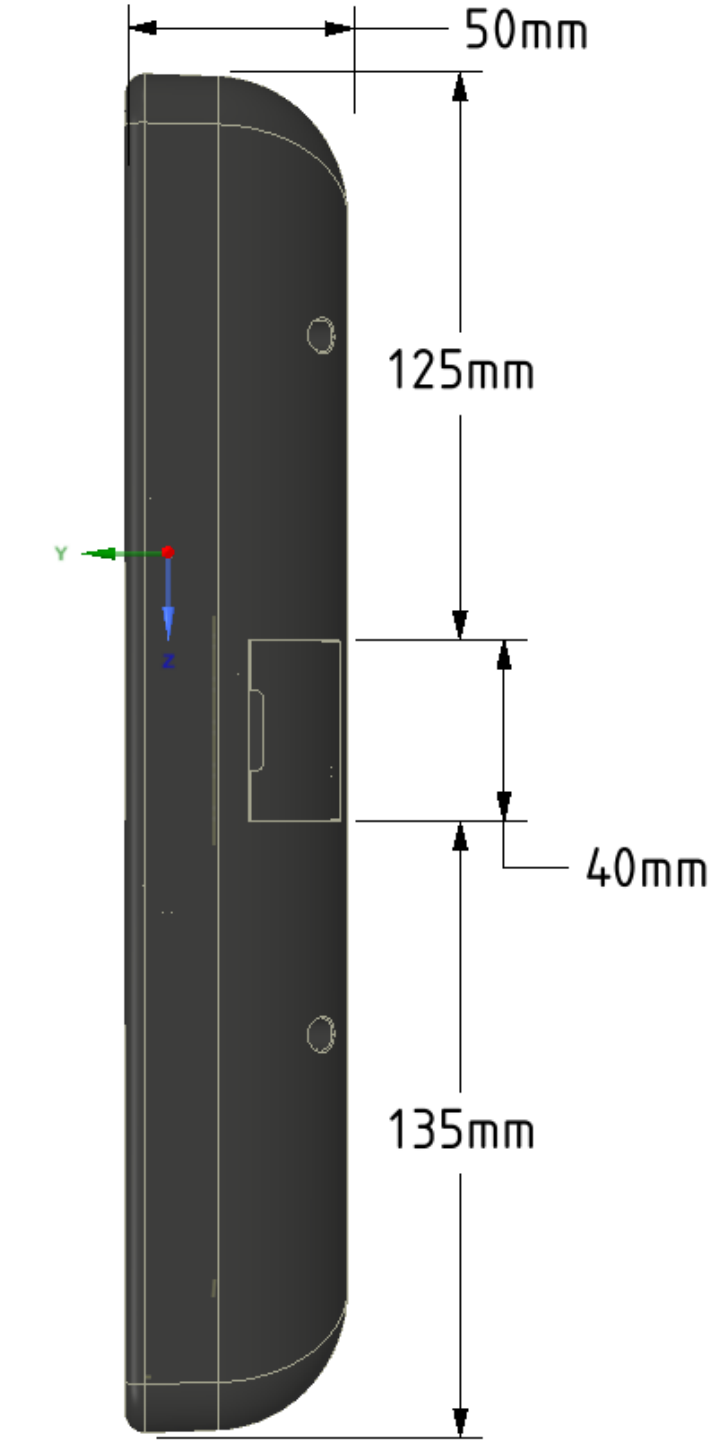


Figure 4 – Enclosure Side Dimensions

ClassTouch™ Pi Enclosure

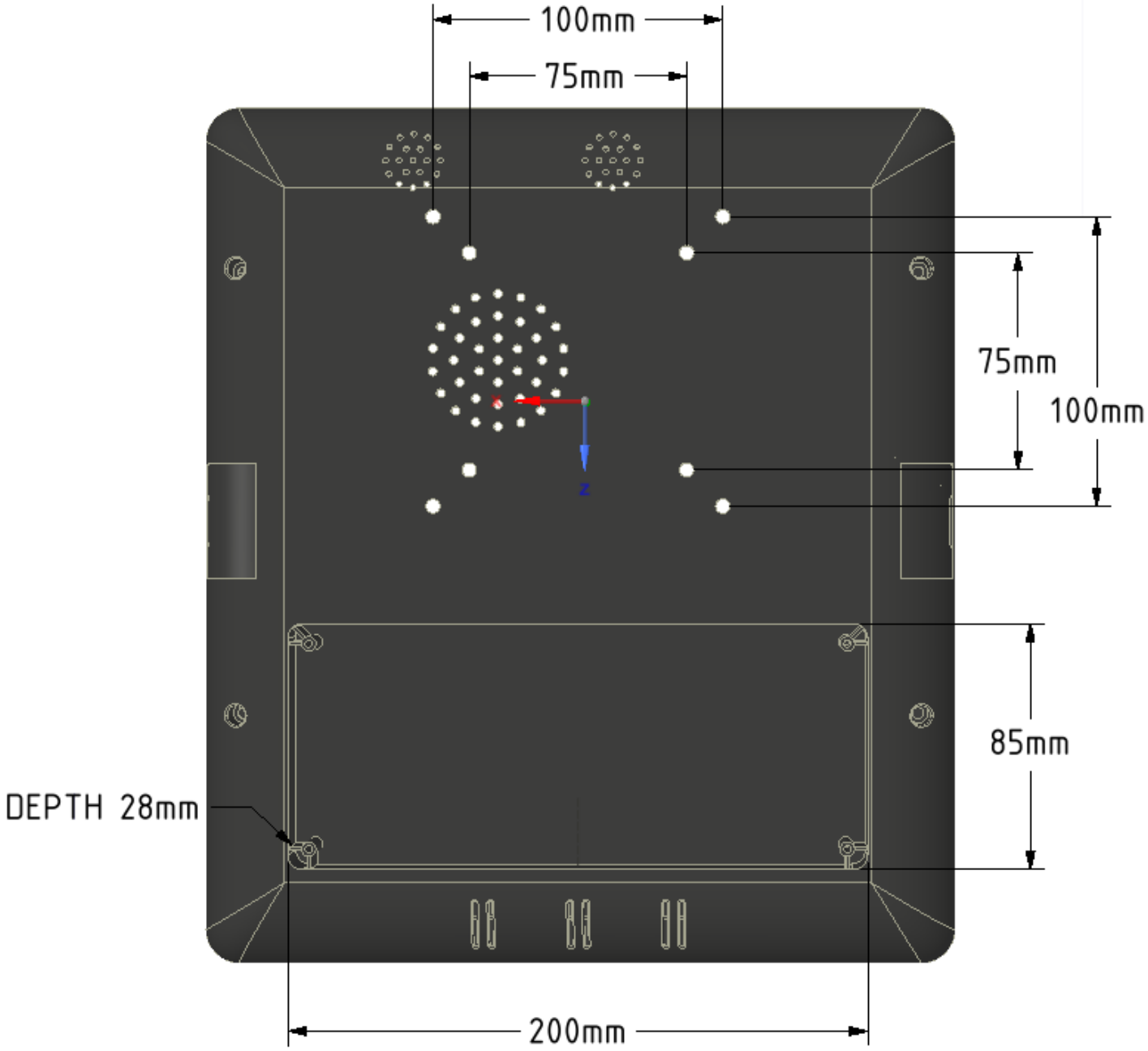


Figure 5 – Enclosure Rear Dimensions

ClassTouch™ Pi Enclosure

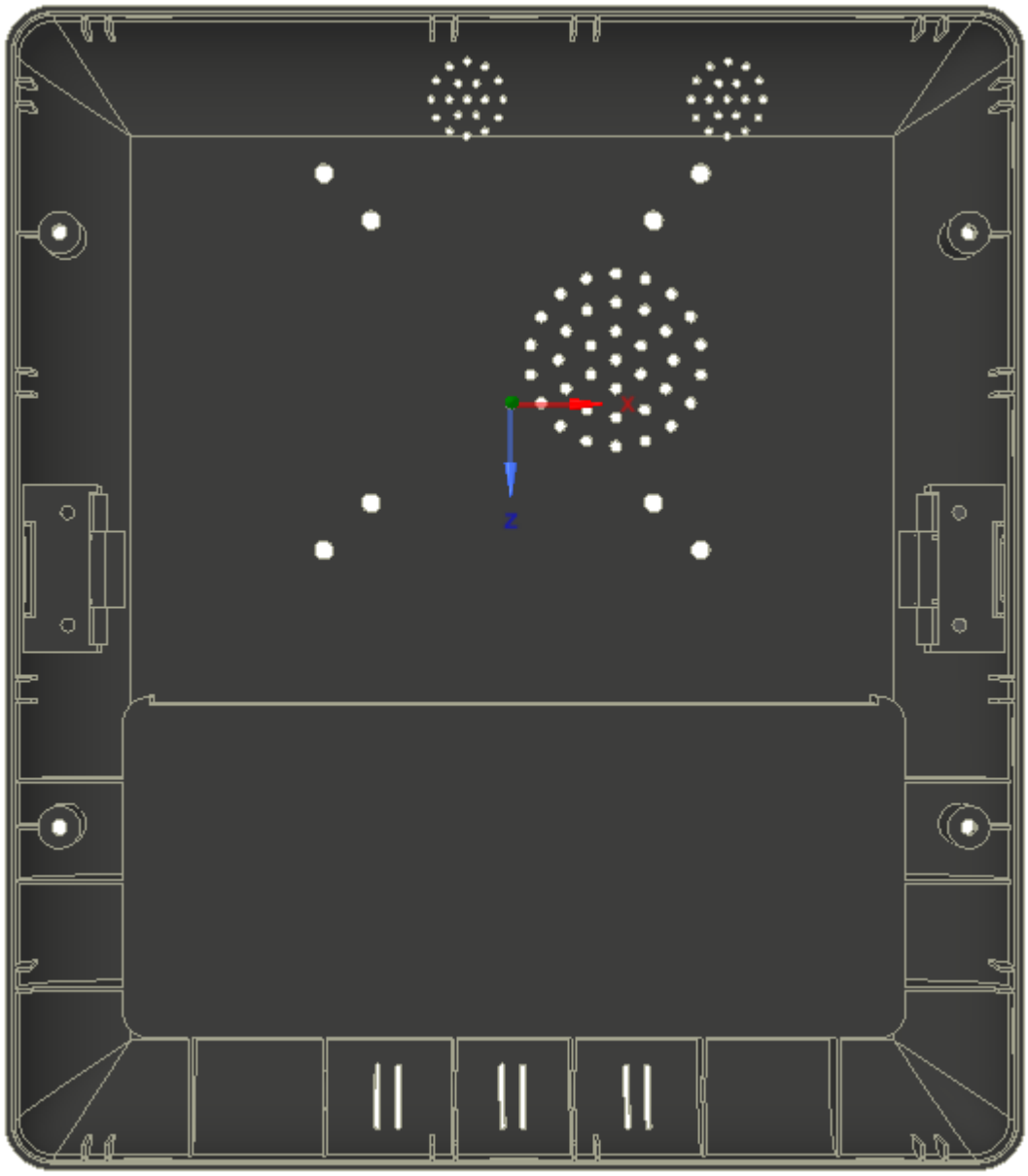


Figure 6 – Enclosure Internal View

ClassTouch™ Pi Enclosure

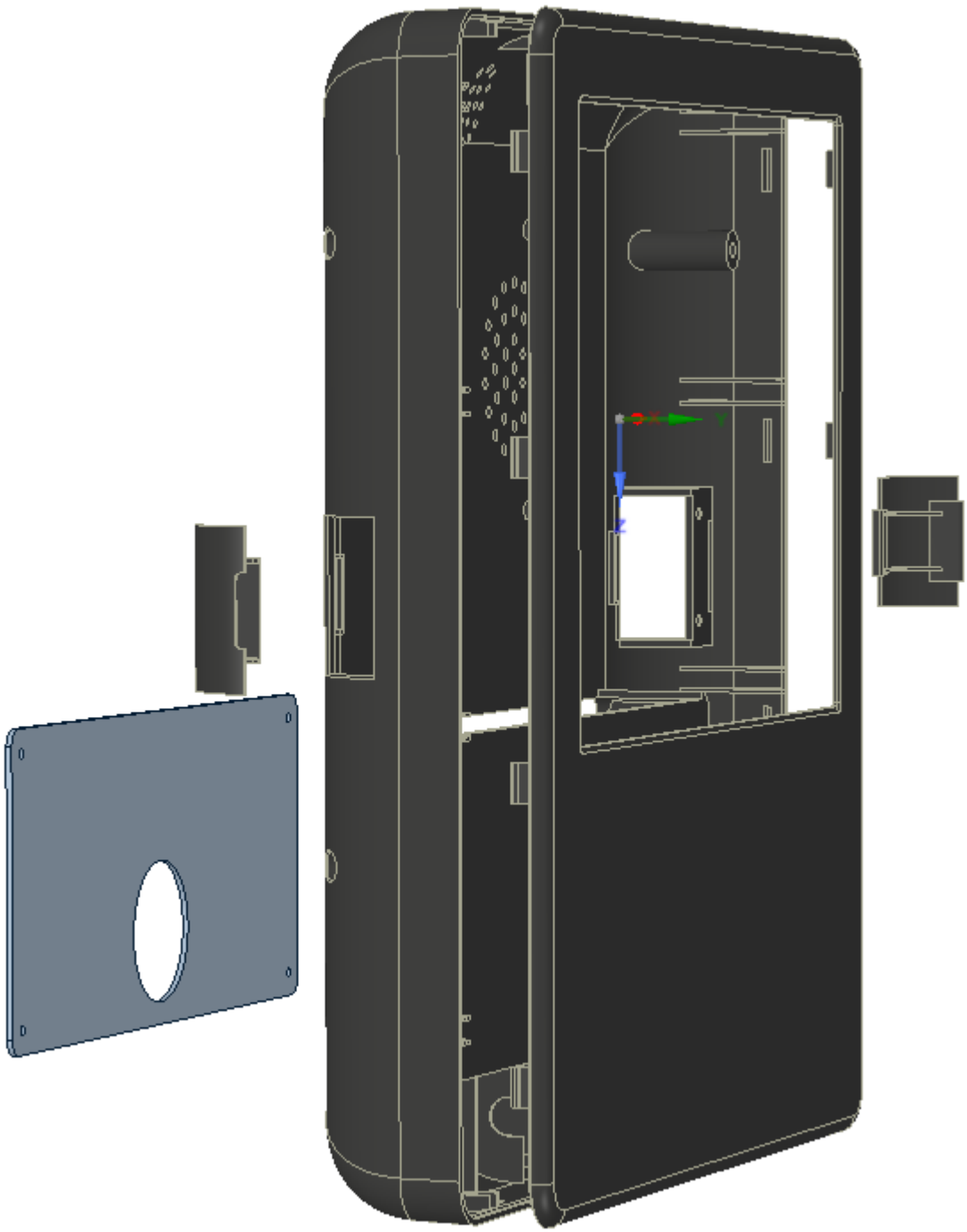


Figure 7 - Enclosure Exploded View

Electrical Specifications

Here we will cover in detail all the components that are used in the ClassTouch™ system:

Touchscreen

- USB Interface
- 4-Wire Resistive
- Active Area 375mm x 300mm
- Operating Temperature -20C to +70C
- No driver software to install
- eGTouch touch driver supported by Raspberry Pi OS (see the Touchscreen Driver section for more details)

Display

- Size 10.4 inch
- Resolution 1024(RGB) x 768
- Interface LVDS 6 bits
- Color Depth 262K
- Technology Type a-Si
- Pixel Pitch (mm) 0.264x0.264
- Pixel Configuration R.G.B. Vertical Stripe
- Display Mode TM with Normally White
- Surface Treatment(Up Polarizer) Anti-Glare(3H)
- Surface Treatment(TSP) Anti-glare type (3H)
- Viewing Direction 12 o'clock
- Gray Scale Inversion Direction 6 o'clock
- Operating Temperature -20C to +70C

Display Driver

- MSTAR Chipset
- Language Support - Chinese, English, French, German, Italian, Spanish, Russian
- Panel Type LED/LCD
- Interface Single/Dual LVDS (8bit)
- Max Resolution 1920 × 1080
- Video Input Signals VGA & HDMI (H-Freq 30~90KHz
- Audio Input - PC Audio, HDMI
- Audio Output - Frequency Response ±3dB
- Max Output power - 2×3W(4Ω)
- Power Requirement 12V DC
- Standby Power Consumption < 1W(Board Only)
- Keyboard Function - POWER,MENU,SOURCE,UP/DOWN,LEFT/RIGHT

Power Requirements

The ClassTouch™ system includes a 12VDC 4A power supply, additionally a 5VDC 3A DC/DC switch mode regulator (also included) is used to power the Raspberry Pi. This makes powering the ClassTouch™ system very convenient as the 12VDC power supply is the only external power source required to power the display, display driver and the Raspberry Pi.

Pi Power Controller

The Pi Power Controller (PPC) is a small computer controlled power board that has many useful features. First is the ability to RESET the Raspberry Pi using the included Reset Button, if the Raspberry Pi is up and running when the reset button is pressed, a small python script (examples of this script are provided later in this manual) running in the background will execute a shutdown command yielding a safe graceful power down process. After about 30 seconds, the PPC then completely disconnects the power to the Raspberry Pi so power consumption is reduced to an absolute minimum. This is also useful as any device powered by the Raspberry Pi's USBs will also be completely powered down.

If the reset button is pressed while the Raspberry Pi is in the powered down state, then the PPC will initiate the power up process while additionally disabling the reset button. The reset button will again be enabled after the Raspberry Pi has completed the boot up process and is consequently in a safe state to then allow a shutdown request.

Lastly the PPC provides the ability to 'Auto Start' the Raspberry Pi allowing you to automatically startup the ClassTouch™ system at a specific time and day of the week, for example, maybe you need ClassTouch™ to start up at 7:AM and shutdown at 5:PM Monday through Friday but not startup at all on Saturday and Sunday. The PPC can control the startup time from a few minutes to approximately 194 days with an accuracy of a few seconds. The 'Auto Start' feature is easily accessed using a Python script on the Raspberry Pi, we have provided code examples later in this manual that you can use to get started and modify as needed.

Installing Your Raspberry Pi

You will first need to disassemble the ClassTouch™ enclosure to gain access to the internals for the installation of the Raspberry Pi, and to make all of the necessary connections to it. As mentioned earlier, the enclosure is a two part design with the front panel being easily removable with the correct technique. You will need a thin, non-sharp knife, a butter knife works well or the "IFIXIT" opening tool from Adafruit and a medium sized Philips screwdriver. Please follow the steps below to install your Raspberry Pi:

1) First, place the ClassTouch™ unit on its back on top of a soft surface, a piece of foam or a towel for example.

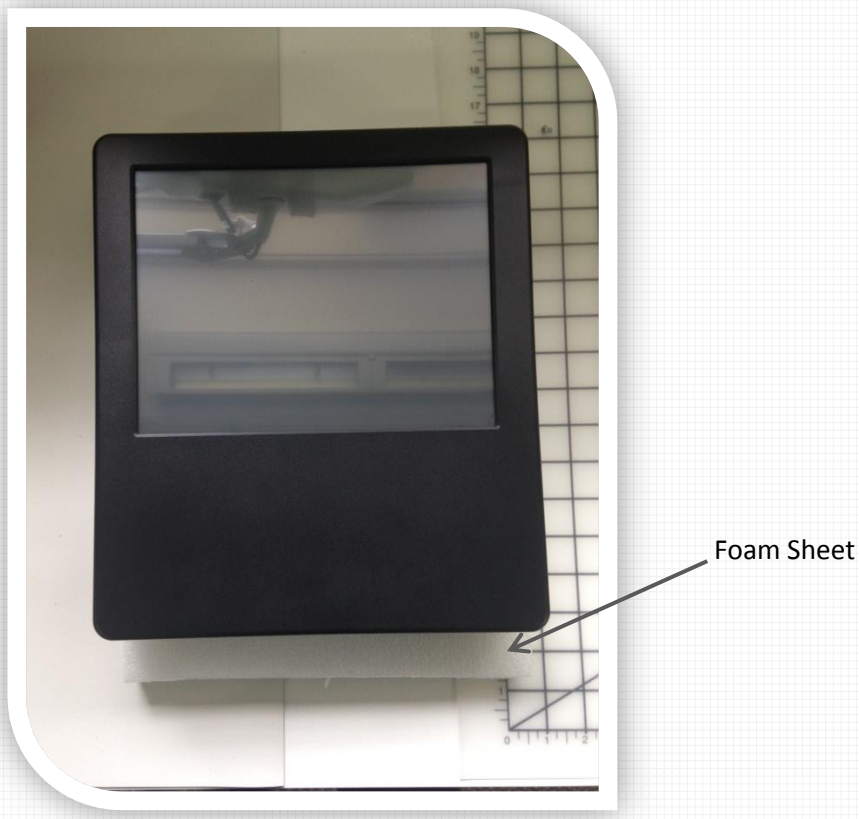


Figure 8 - ClassTouch Ready for Disassembly

2) Next, carefully pull up on the inside edge of the front panel using your thumb and fingers as shown then pull with even force on your thumb and fingers directly upward away from the display, the front panel should separate from the rear section, you may or may not need to use the opening tool for the two halves to separate, use the same technique in different locations around the edge of the front panel until the front panel is completely detached.

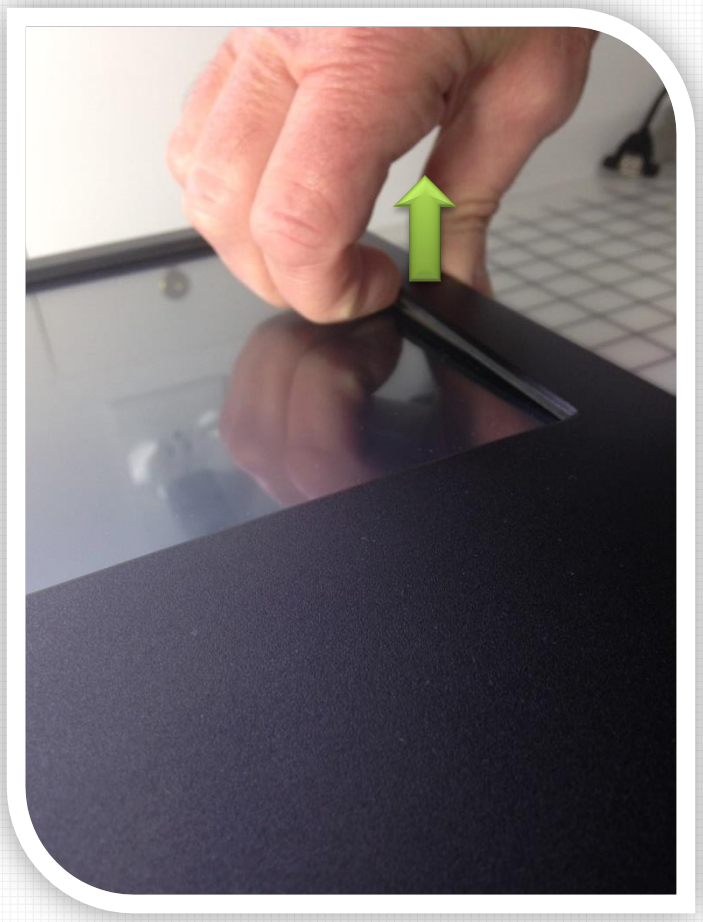


Figure 9 – Opening the Two Part Enclosure

3) If you have to use the knife, insert it between the front panel and the inside of the rear section and apply gentle pressure to lever the side wall outward in direction away from the enclosure.

Figure 10 - Removing the Enclosure Front Panel

4) Please be gentle, using the correct technique will then require a very small amount of force for the front panel to separate from the rear section.

Figure 11 – Enclosure with Front Panel Removed

5) Being careful of the glass display, turn the unit over so the display is face down on the foam sheet with the back of the enclosure facing upward, now remove the 4 screws from the metal cover plate and place the cover plate to the side, then remove the four Philips screws that secure the enclosure to the inner frame.

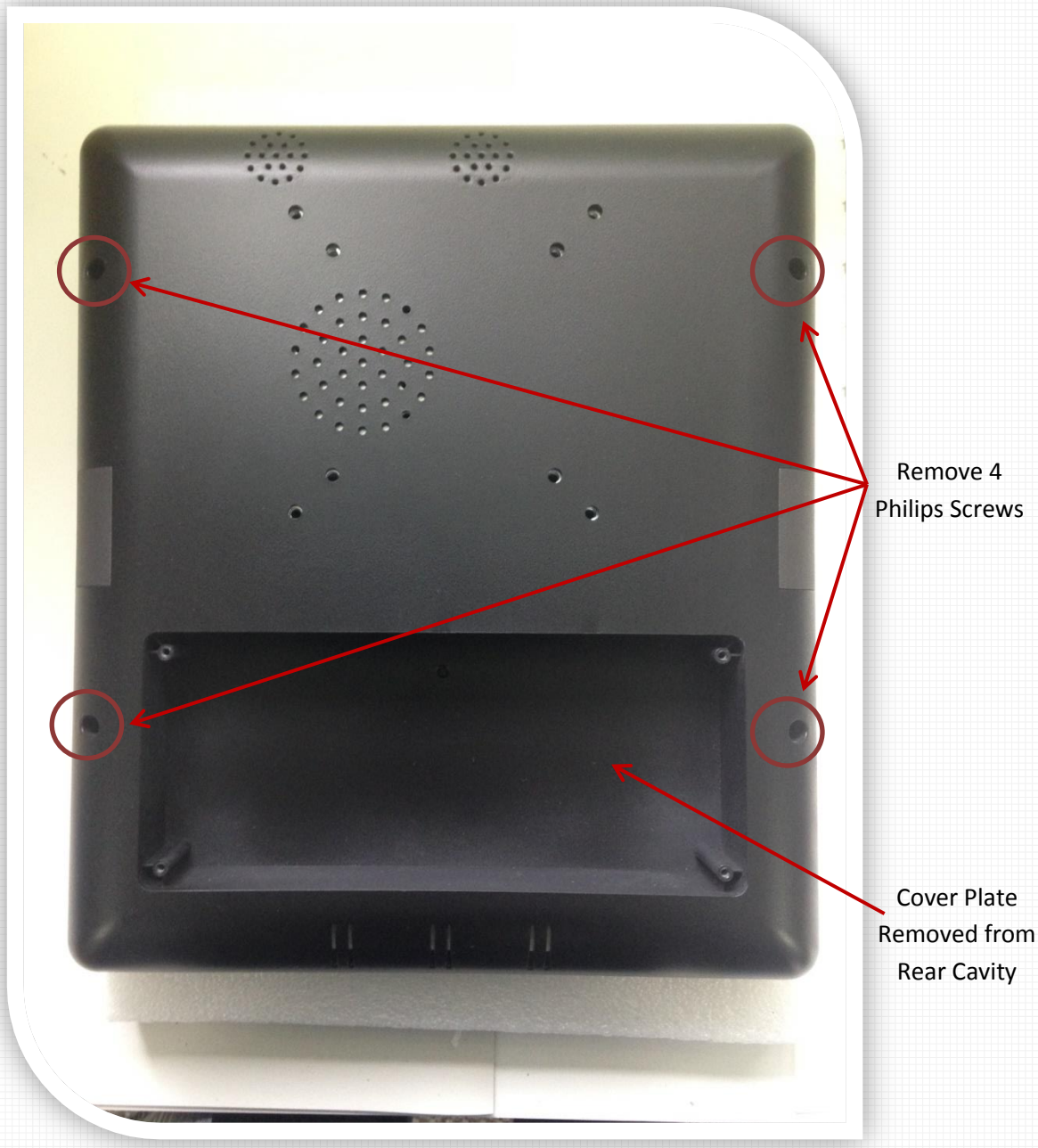


Figure 12 - ClassTouch Rear Enclosure Screws

6) With all four screws removed, gently lift up on the rear of the enclosure and separate it from the inner metal frame. Be careful that the power cable does not get caught in the rear cavity.

7) Now remove the four Philips screws from the cover housing of the metal frame as shown below.

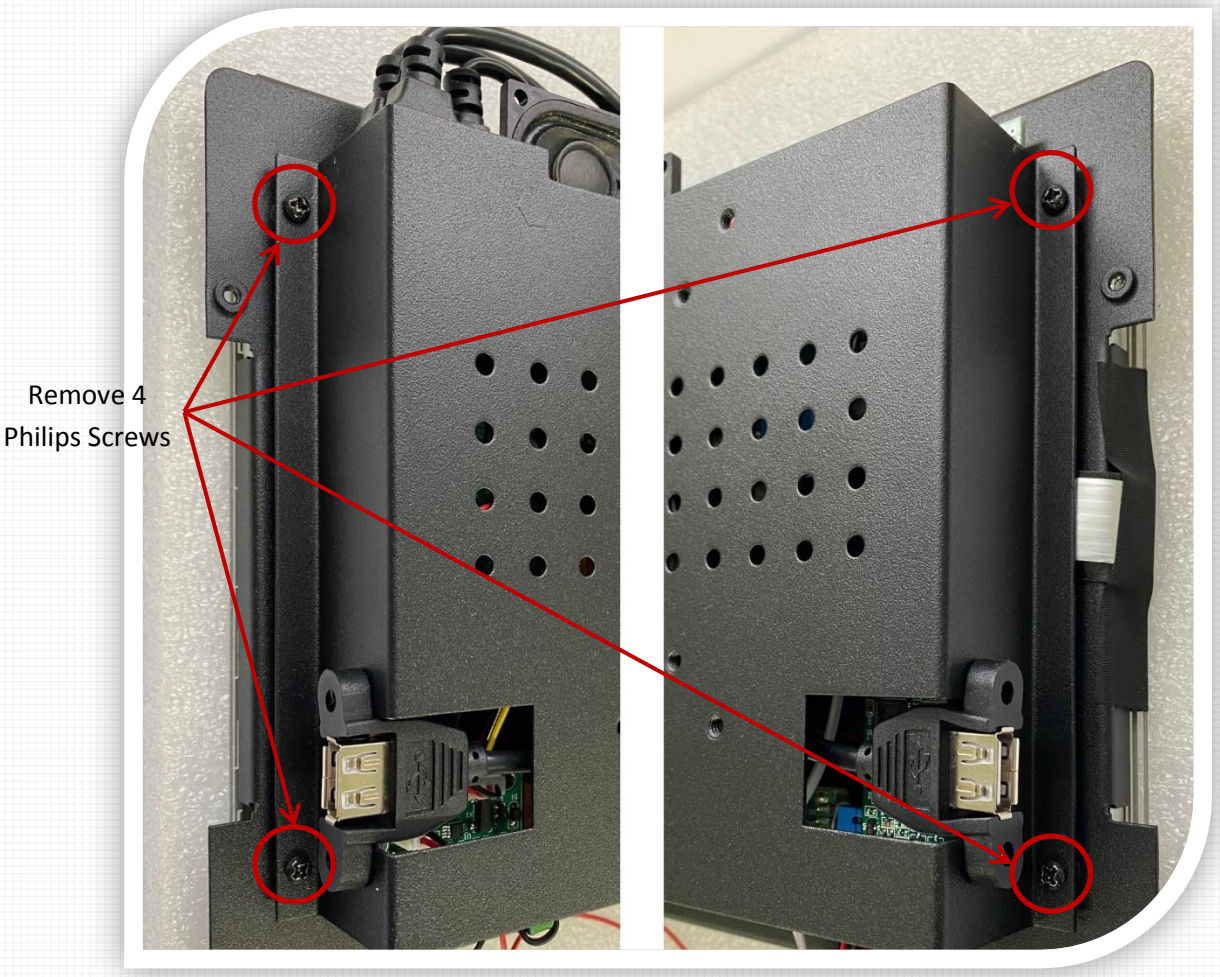


Figure 13 - ClassTouch Cover Housing Screws

8) Carefully remove the cover housing and be aware of the speaker wires connected to the display driver board.

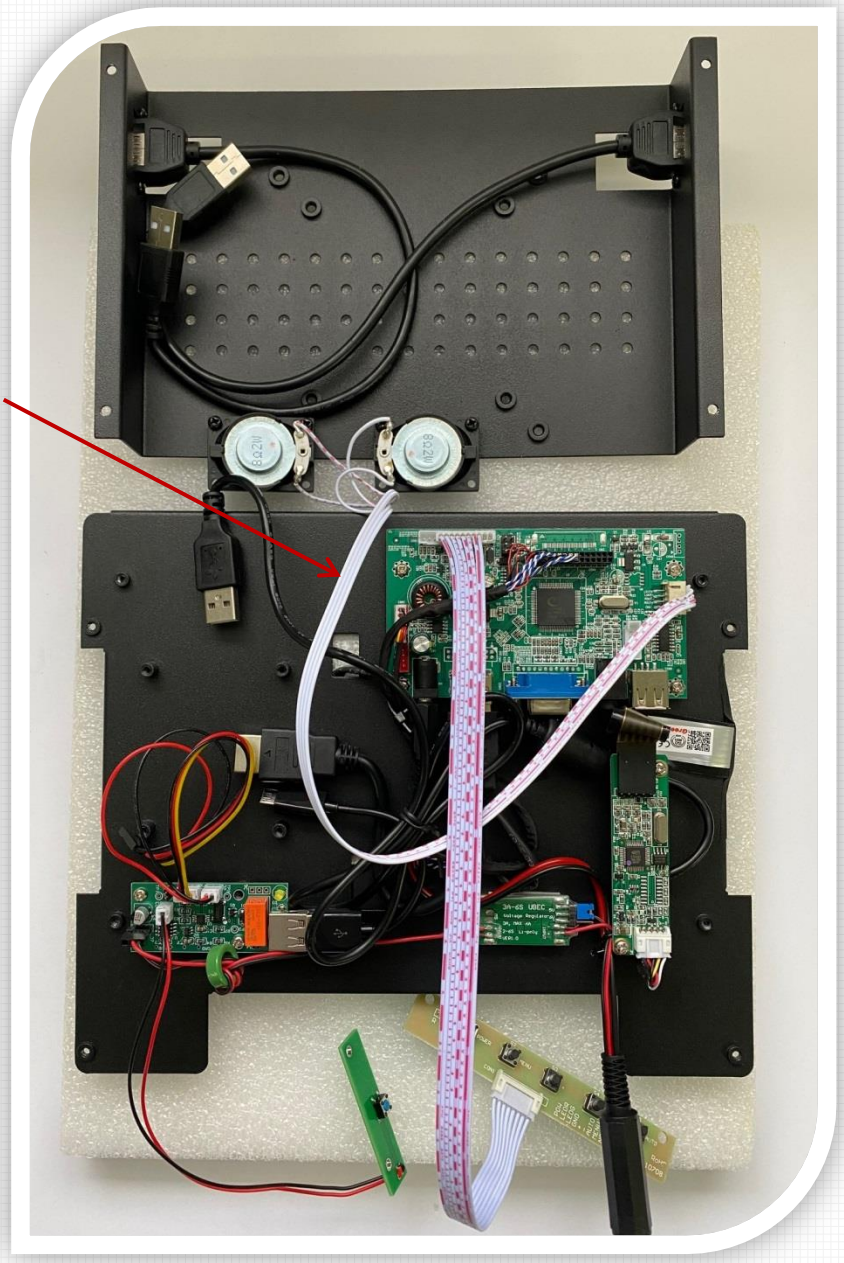


Figure 14 - ClassTouch Cover Housing Removed

9) If the display driver menu keypad is not connected, you can do that now by connecting the ribbon cable as shown below.

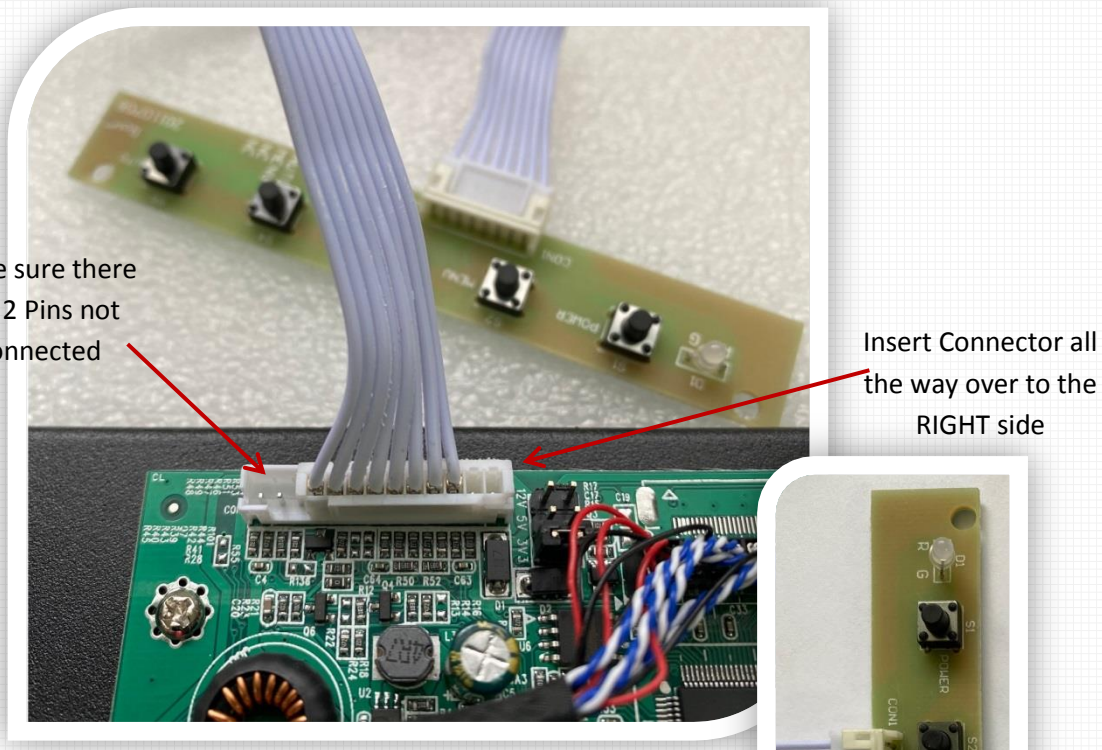


Figure 15 - Display Driver Board Menu Connector

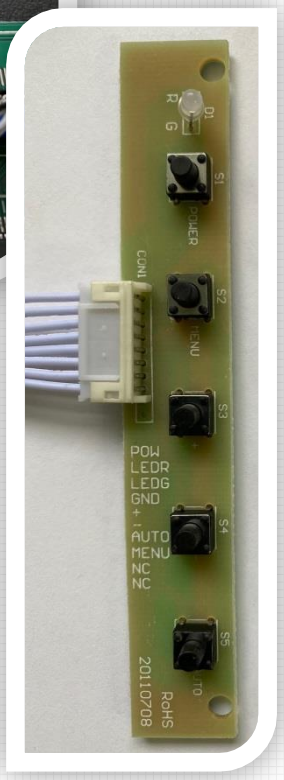


Figure 16 - Display Menu Keypad

10) To install the Raspberry Pi, first remove the screws from the standoffs and then position your Raspberry Pi as shown with the USB Ports toward the top.

11) The holes in the Raspberry Pi's PCB may be a little bit too small for the screws, if so please do not force the screws as this may cause damage to your Raspberry Pi. Instead, simply use a 1/8" drill bit to open up the holes being careful to keep the drill straight and using very little force while drilling.

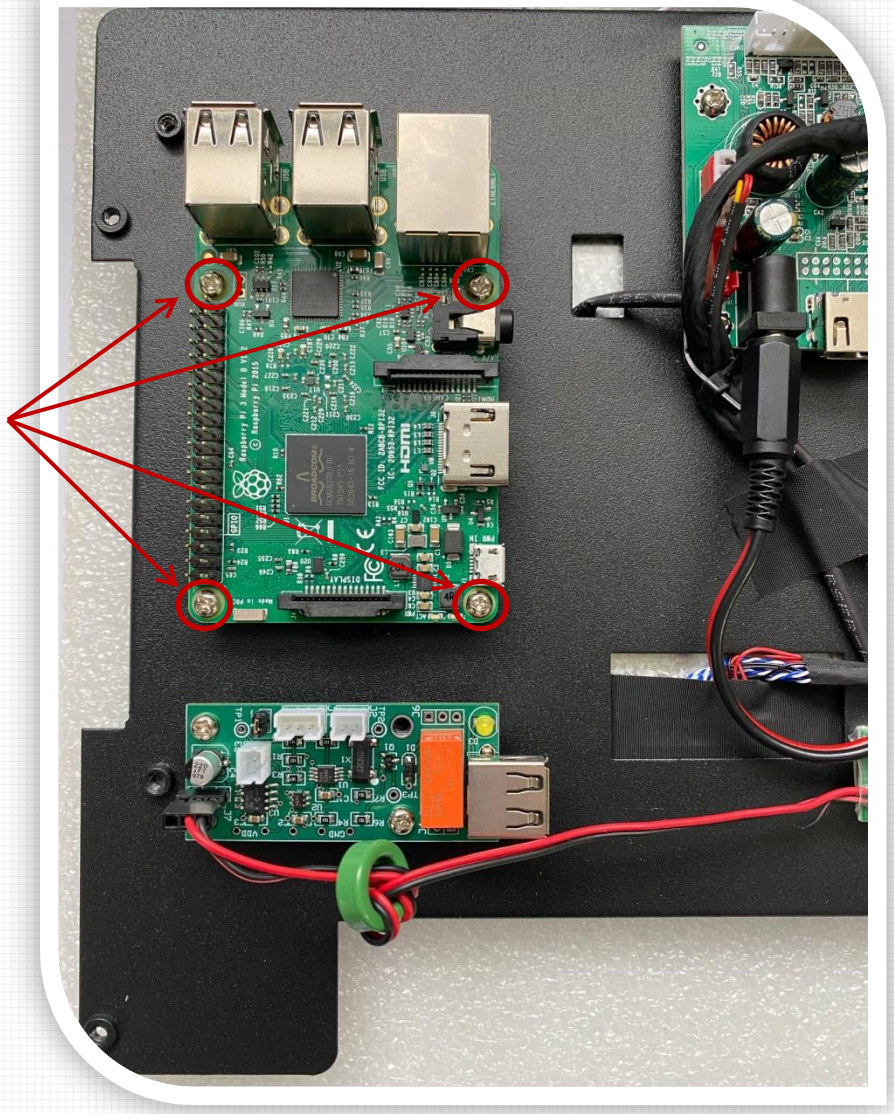


Figure 17 - Installing the Raspberry Pi

12) ClassTouch™ ships with all required cables and connectors, below is a description of each

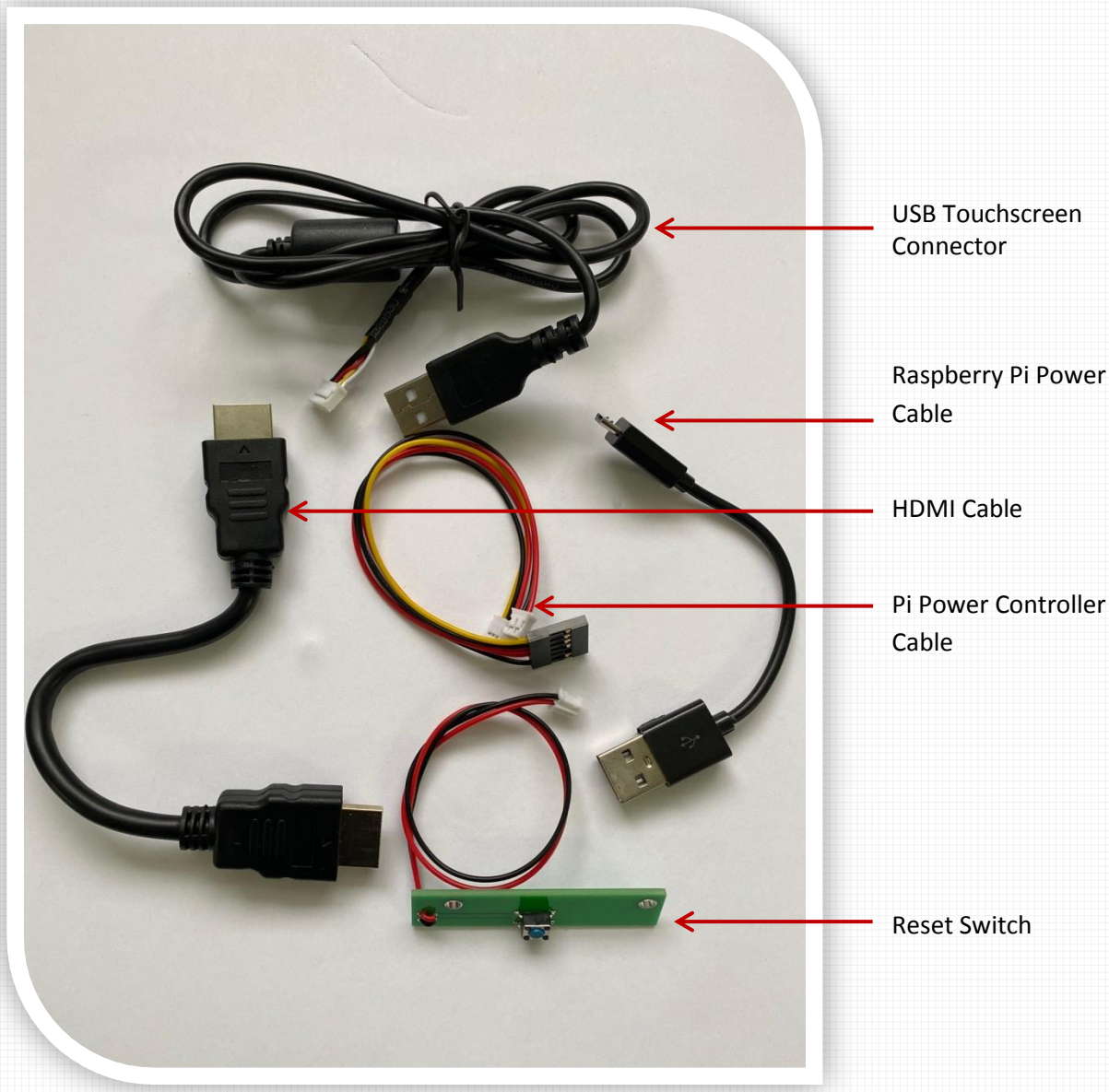


Figure 18 - ClassTouch Supplied Cables

NOTE: *The cables shown in Figure 18 above are for the Raspberry Pi 2 or 3; the HDMI and power cable will be slightly different if your ClassTouch was purchased for the Raspberry Pi 4.*

13) First connect the HDMI cable as shown below, note that it may be necessary to remove the Touchscreen Controller first.

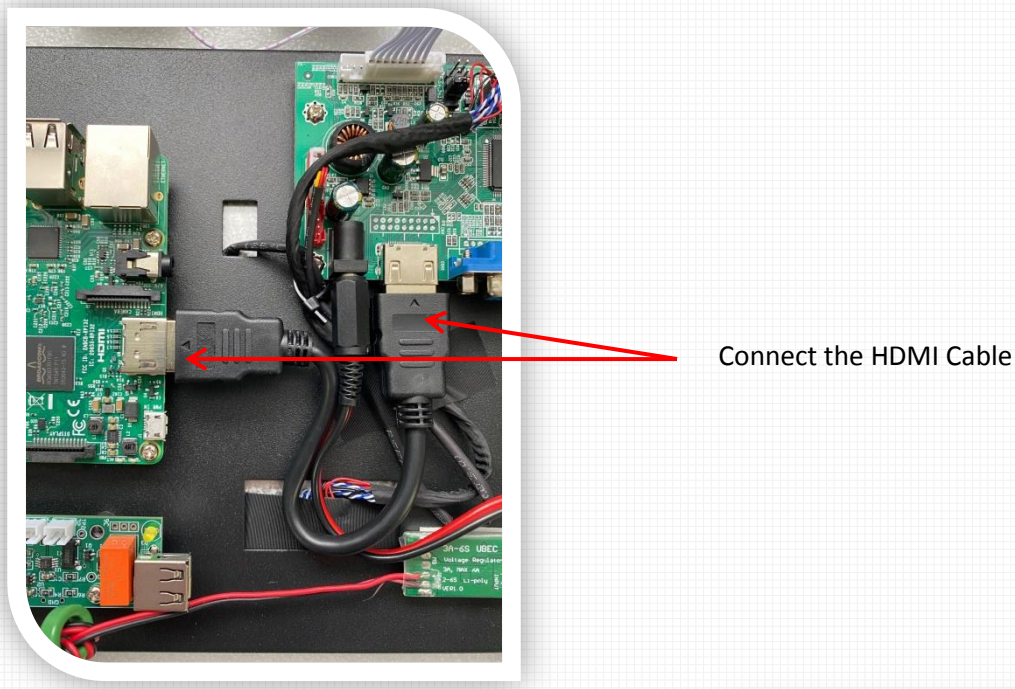


Figure 19 - Attach HDMI Cable

14) Next attach the Pi Power Controller cable.

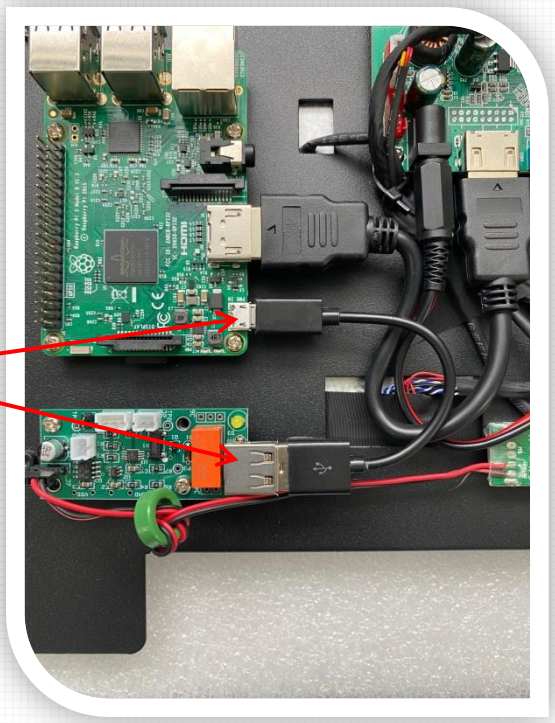
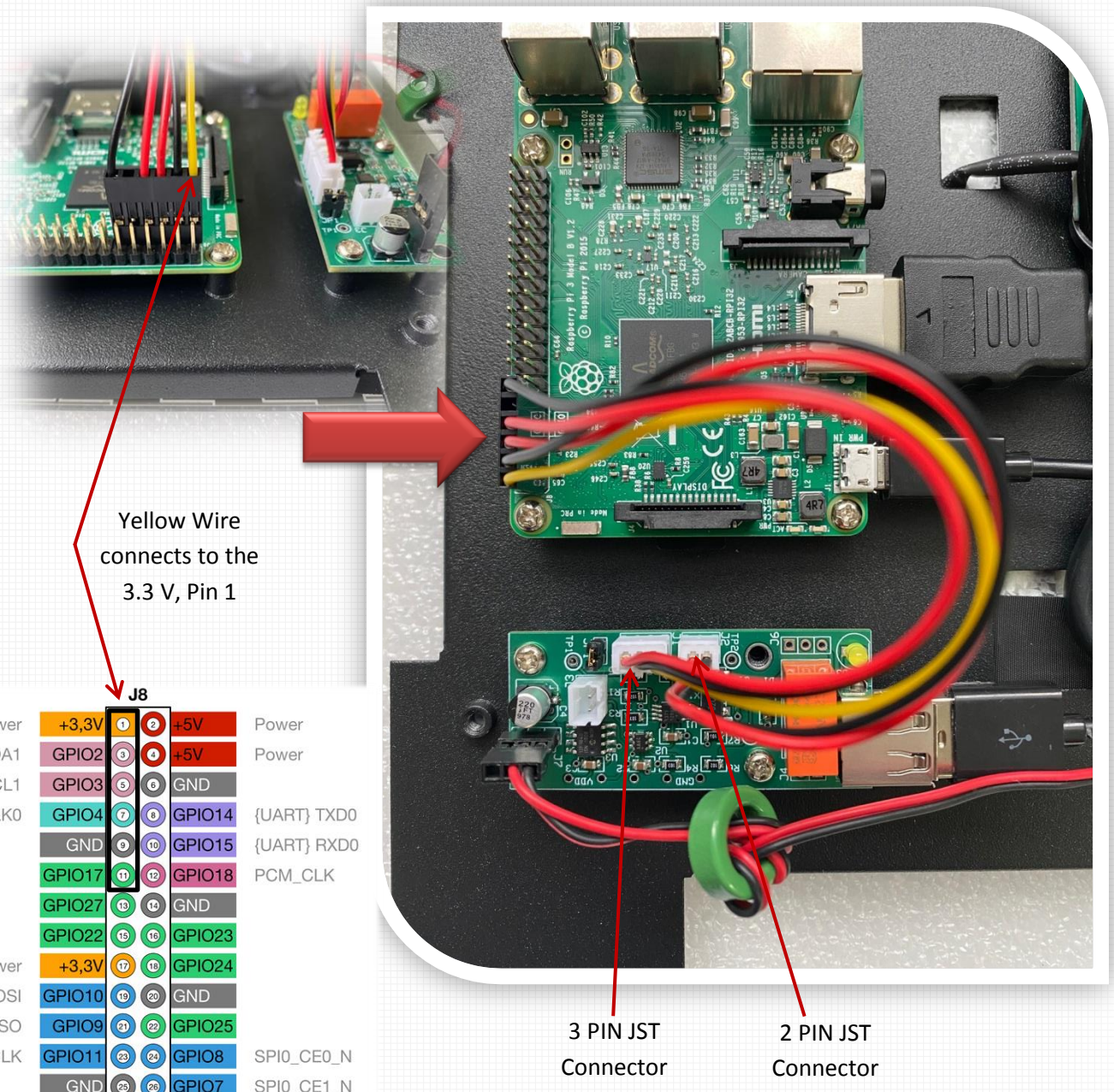


Figure 20 - Attaching the Pi Power Cable

15) Connect the Pi Power Controller to the Raspberry Pi 2, 3 or 4 GPIO pins.



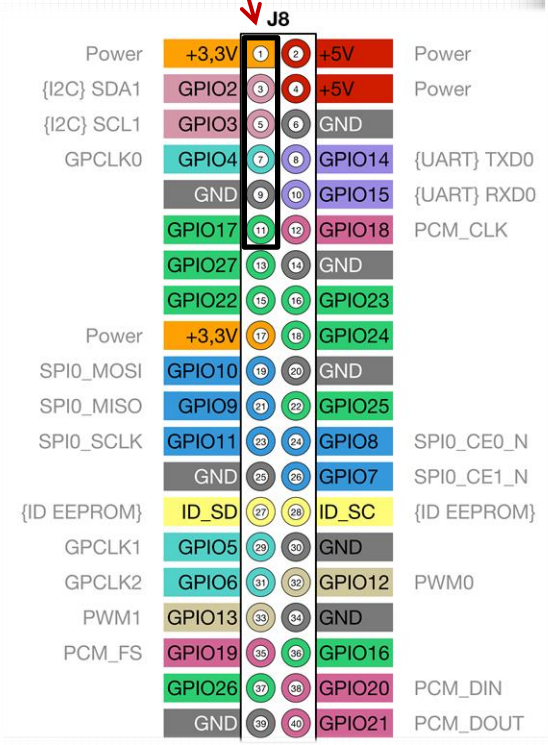


Figure 21 – Raspberry Pi GPIO Cable Connections

16) Now connect the Reset Switch to the Pi Power Controller.

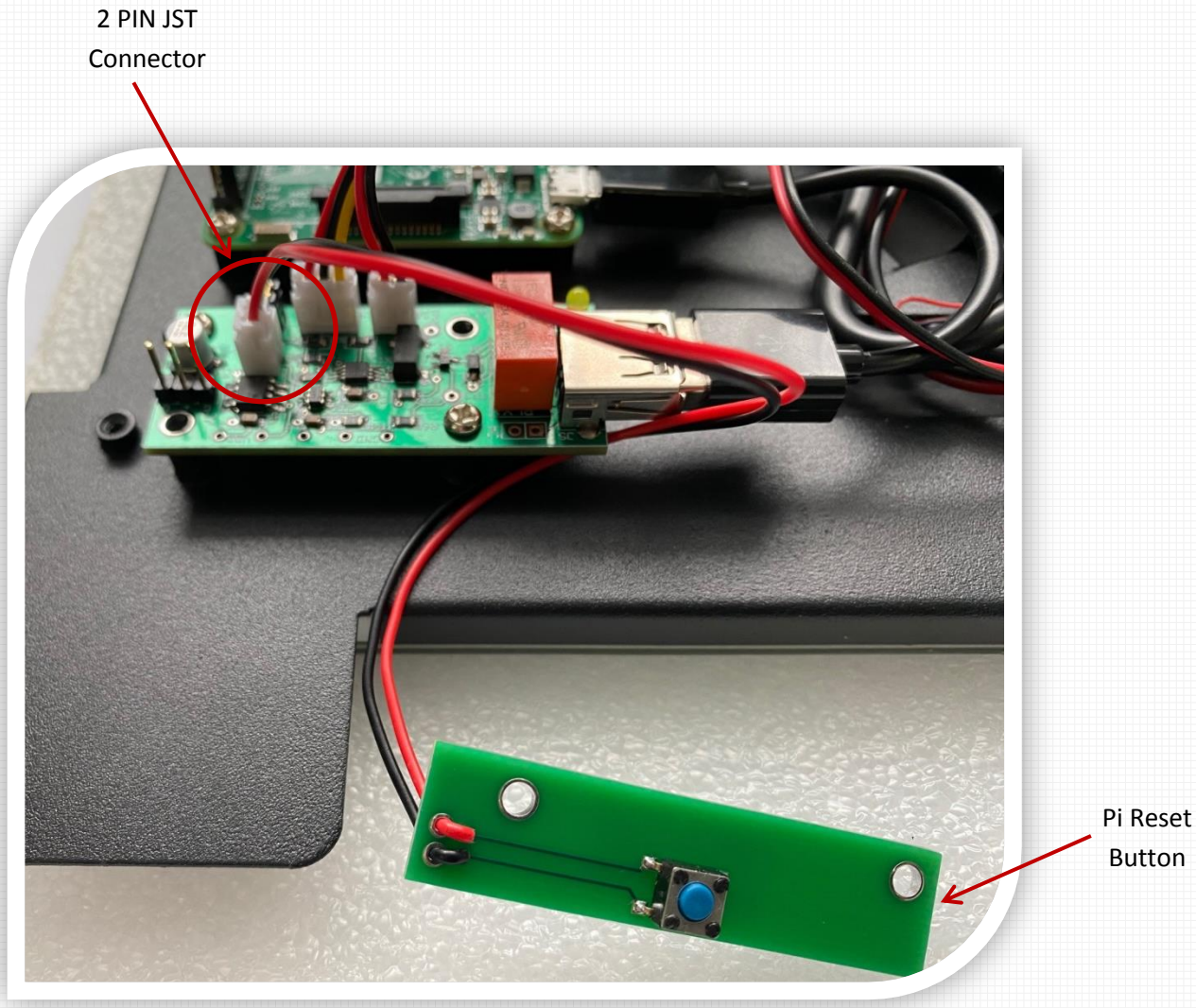


Figure 22 - Pi Reset Button Connection

Power Distribution Cable

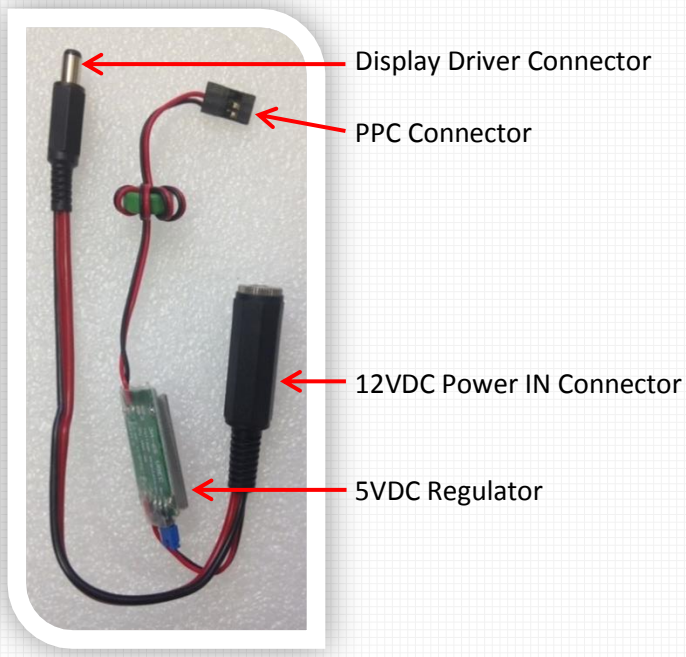
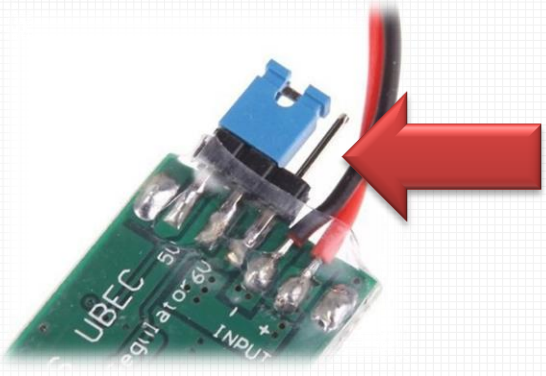


Figure 23 - Power Distribution Cable

Figure 24 - 5 Volt Regulator

The power distribution cable includes a 3A 5.2-5.25V switch-mode DC-DC 5A Max regulator. The Input voltage range is 5.5V-26VDC and with the overall size being just 43mm*17mm*7mm (L*W*H). The regulator is responsible for providing power to the Pi Power Controller (PPC) which then supplies power to the Raspberry Pi. A small red LED indicates that it's receiving power from the external 12VDC 4A power supply. The regulator uses a jumper to select either a 5V or 6V output, please be sure to always have the jumper in the 5 Volt position or serious damage to your Raspberry Pi will result.



WARNING: The Raspberry Pi Requires 5VDC Power

The Jumper **MUST ALWAYS be in the 5 Volt Position** as shown here, this is the default position as set from the factory

Figure 25 - 5 Volt Jumper Position

17) The Power Distribution Cable connects to the Display Driver Board and the Pi Power Controller. **NOTE:** *The female 2 wire 3 position header has no right or wrong way to connect to the 3 pin male connector on the Pi Power Controller board, just make sure that center pins of both the male and female connectors align.*

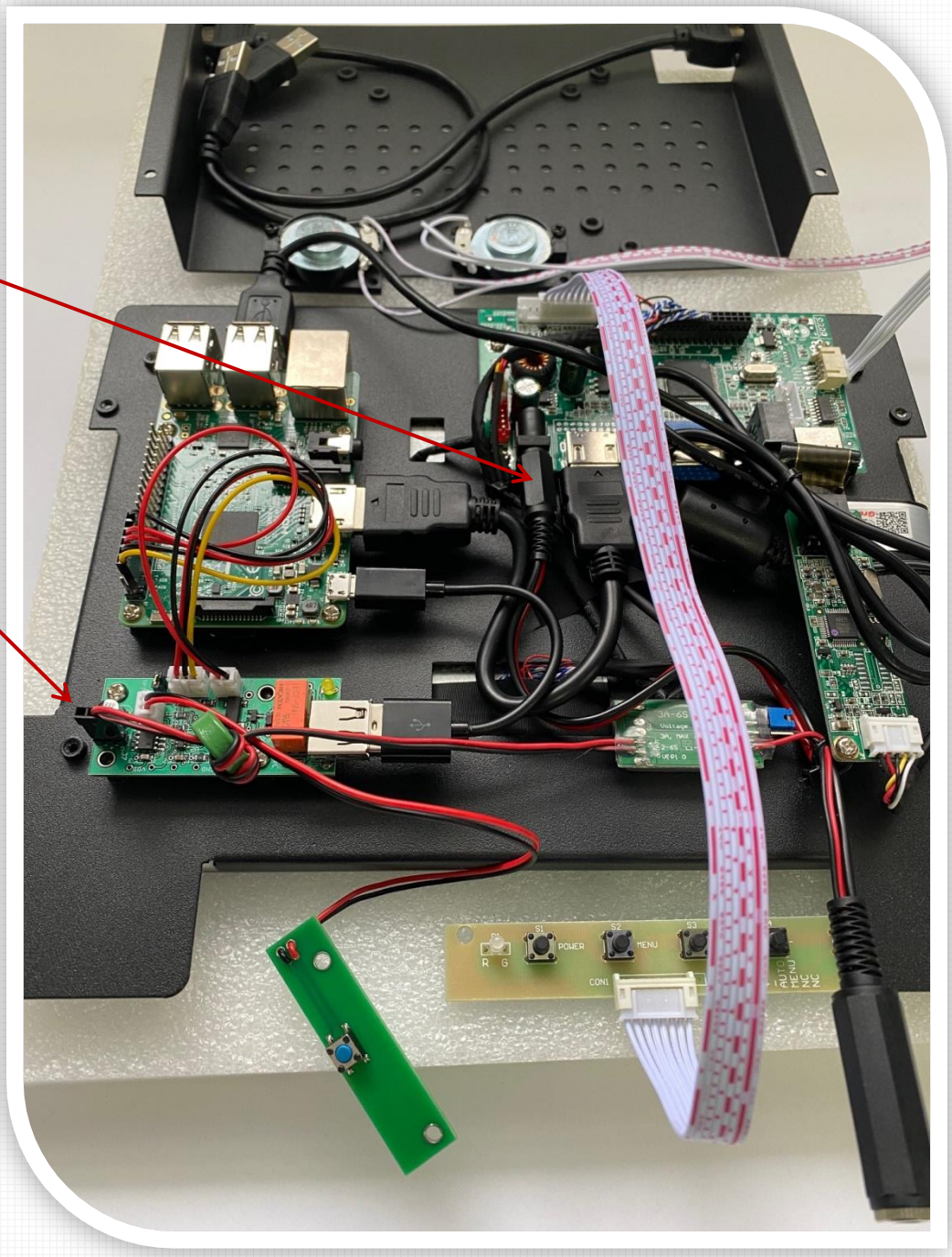


Figure 26 - Power Distribution Cable Connections

18) You can now make all of the USB connections to the Raspberry Pi as shown below. **NOTE:** *The 2 External USB Port connections to the Raspberry Pi are optional and may be used or not depending on your particular requirements. You can now connect any additional USB devices as required for your installation.*

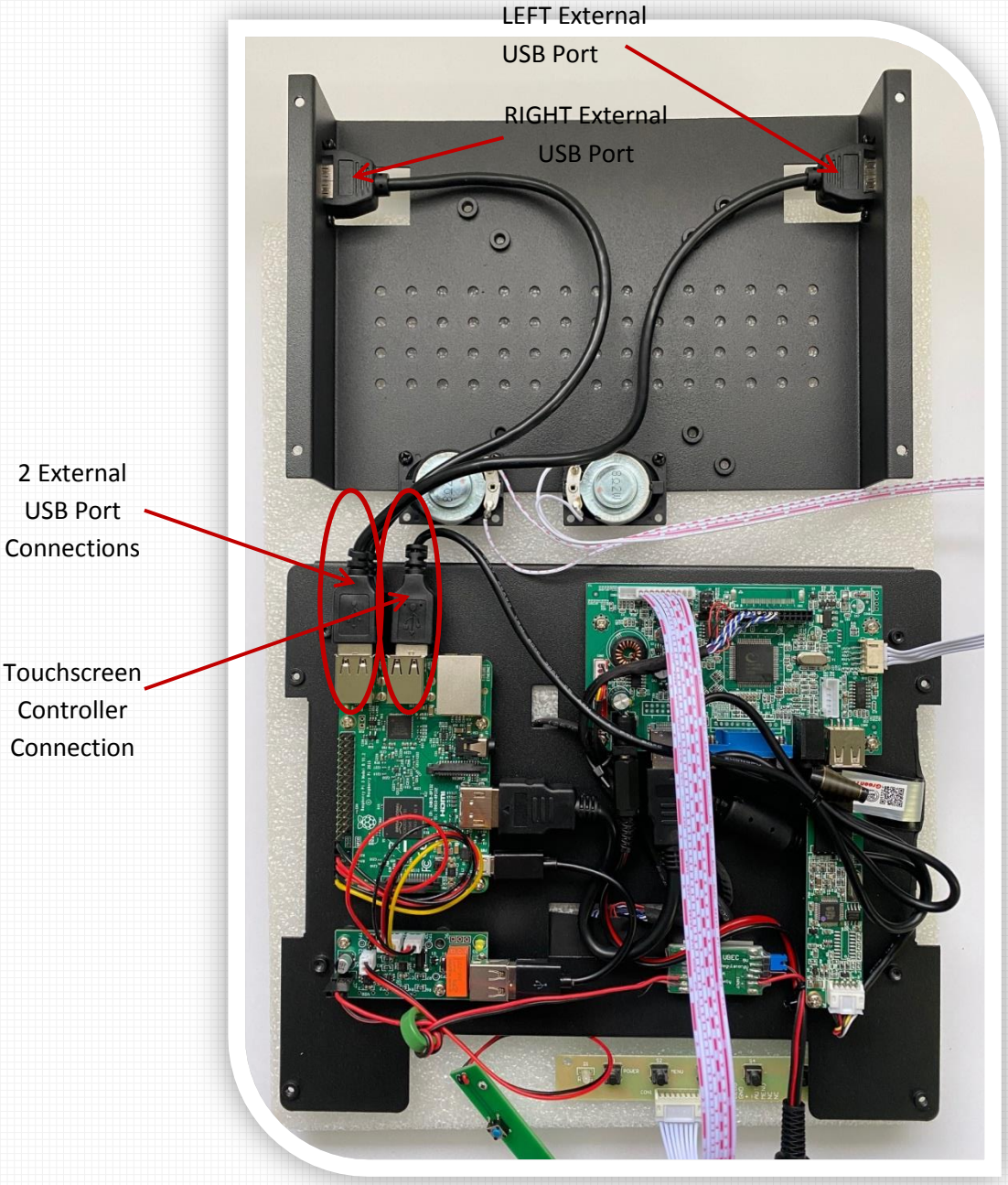


Figure 27 - Connecting USB Devices

19) If all the connections to the Raspberry Pi have been completed, you are now ready to install the cover housing using four Philips screws being careful not to pinch any wires under the housing; you may have to reposition cables and wires in order for the cover housing to sit on the standoffs without using any force. Now attach the Display Menu Keypad and make sure the power cable and the reset button are located at the bottom as shown in preparation for assembling the enclosure.

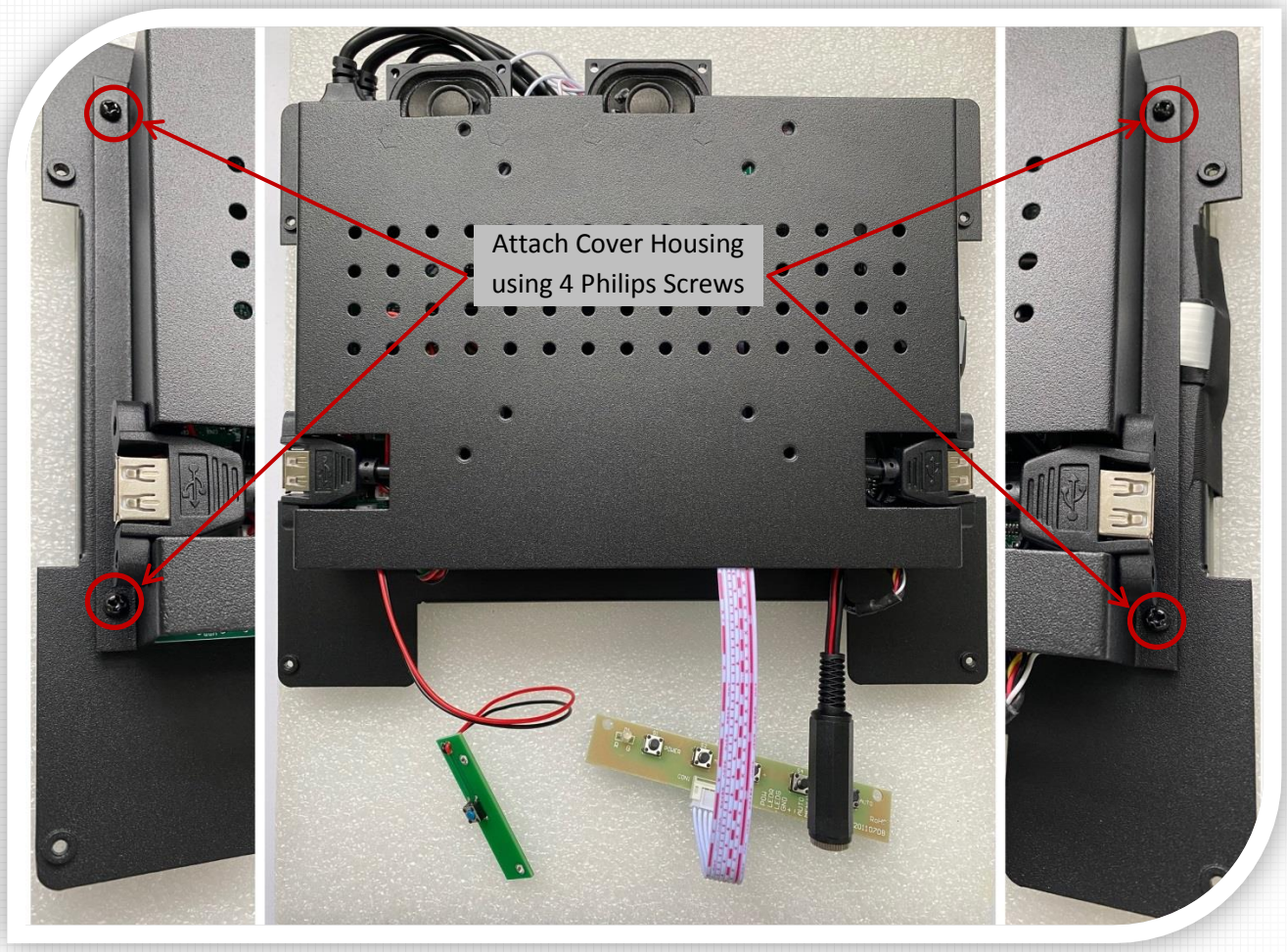


Figure 28 - Install Cover Housing

20) Mounting a Wi-Fi Antenna is easy due to the special compartment in the front of the ClassTouch™ enclosure that can accommodate an Antenna up to 125mm long, mount your Antenna as shown below. **NOTE:** *You can either use a long RP-SMA Extension Cable or use a USB Extension Cable to connect the Wi-Fi Dongle to the Pi's USB Port.*

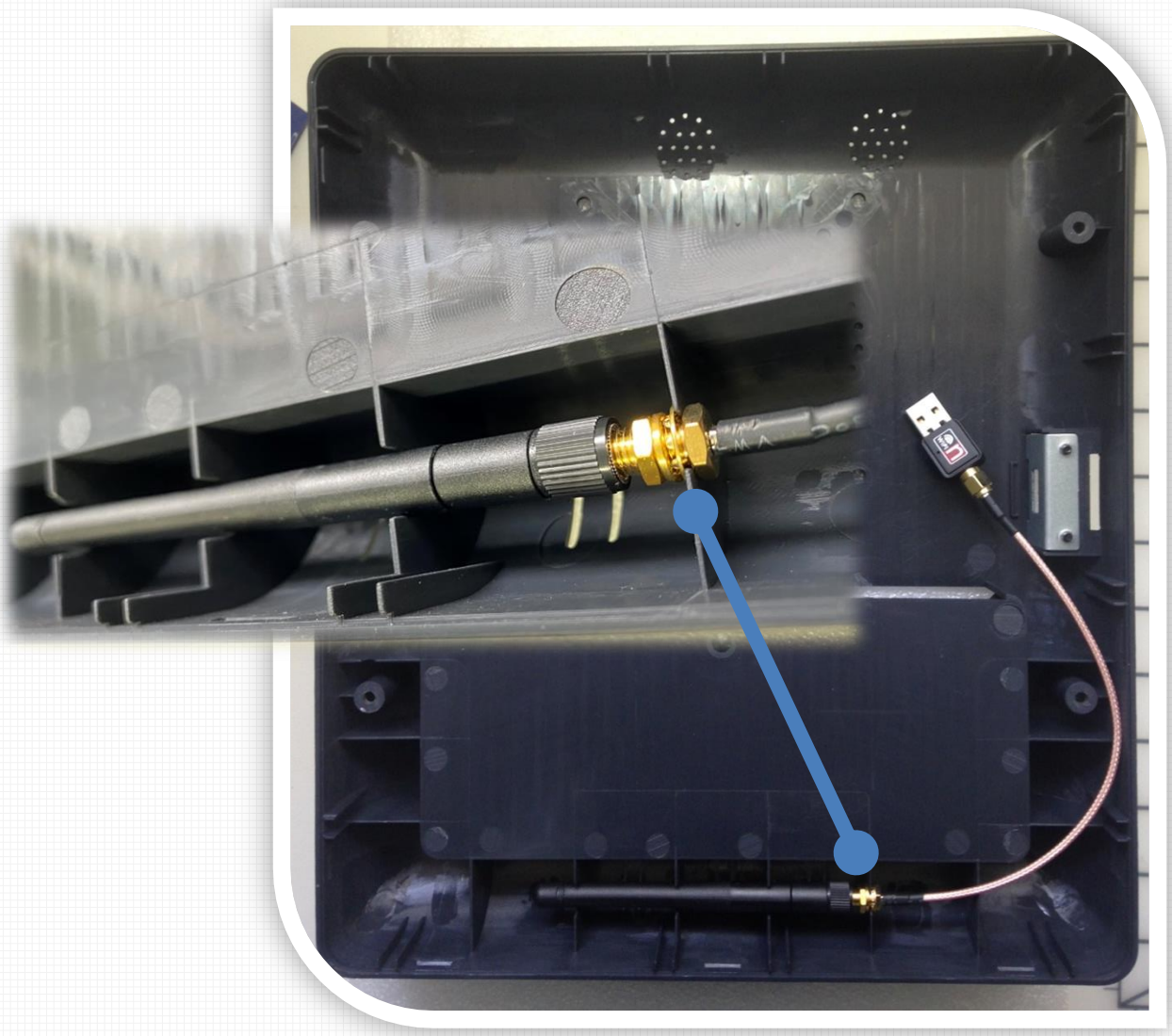


Figure 29 - Wi-Fi Antenna Installation

21) Place the rear of the enclosure over the cover housing while guiding the power and the reset cables through the opening; adjust the enclosure until the cables are pulled through being careful not pinch the wires between the housing and the enclosure. If all is aligned correctly, the VESA holes in the enclosure will align with the threaded VESA holes in the cover housing and the inside of enclosure will be resting on the top of the cover housing with no space between them.

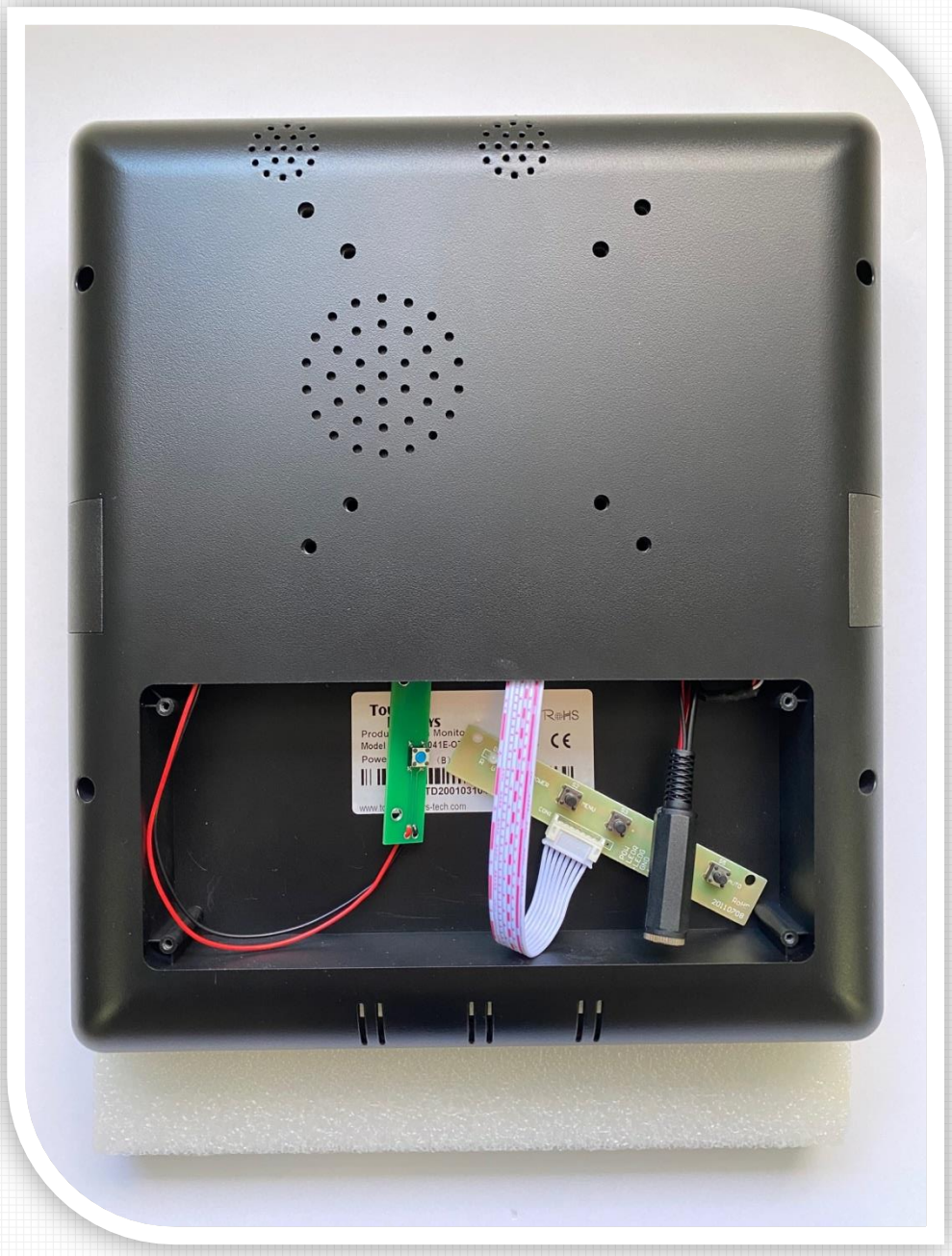


Figure 30 - Rear Enclosure in Position

22) Now replace the four Philips Screws. **NOTE:** *The Display Menu Keypad can be secured at any convenient location using the provided two-sided tape.*

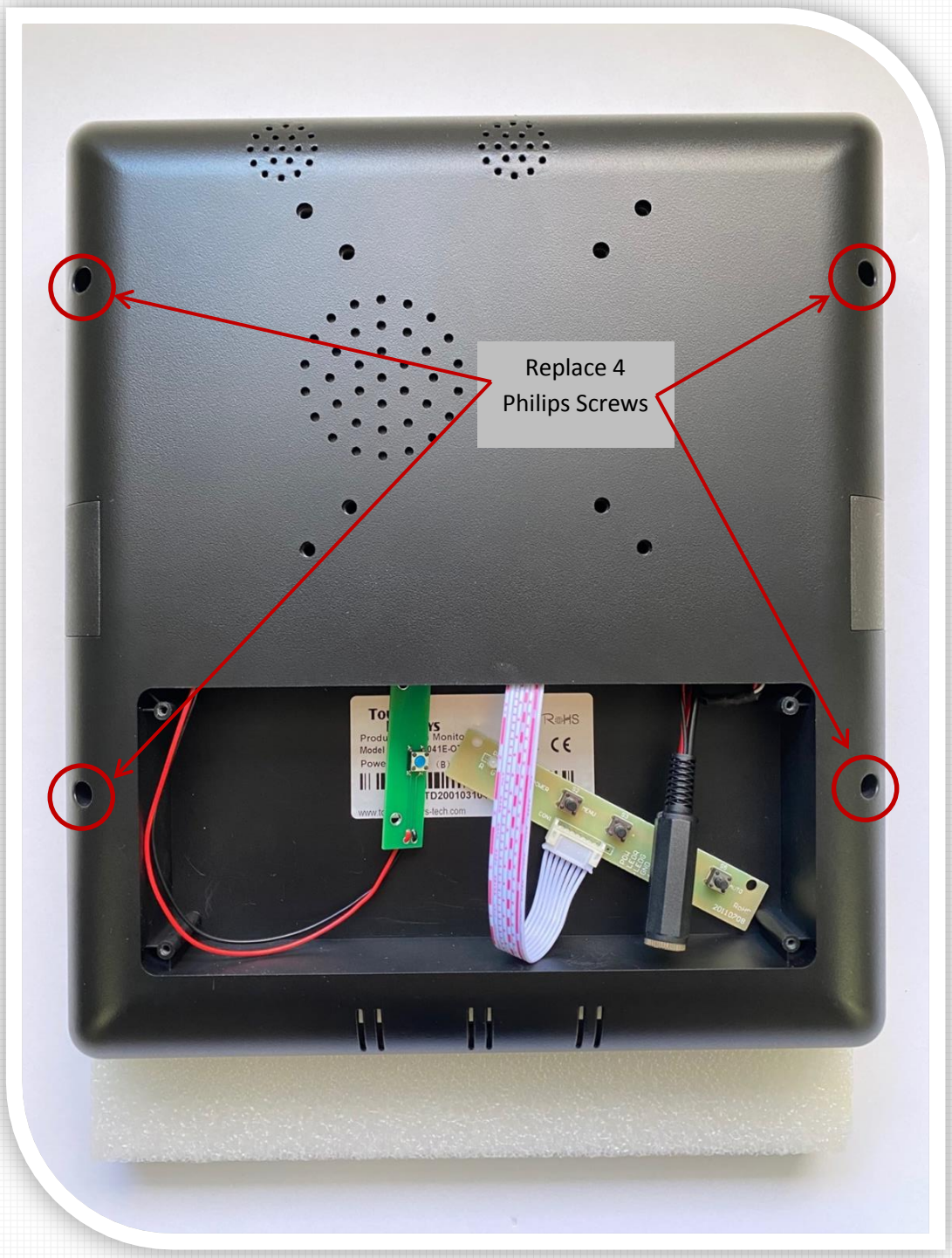


Figure 31 – Replacing Enclosure Rear Screws

23) Attach the reset button board to the inside of the cover panel, notice the small hole in the cover panel that aligns with the reset button, this allows for the insertion of a paperclip to reset the Raspberry Pi when needed.

24) Connect the 12VDC power supply and replace the rear cover panel.

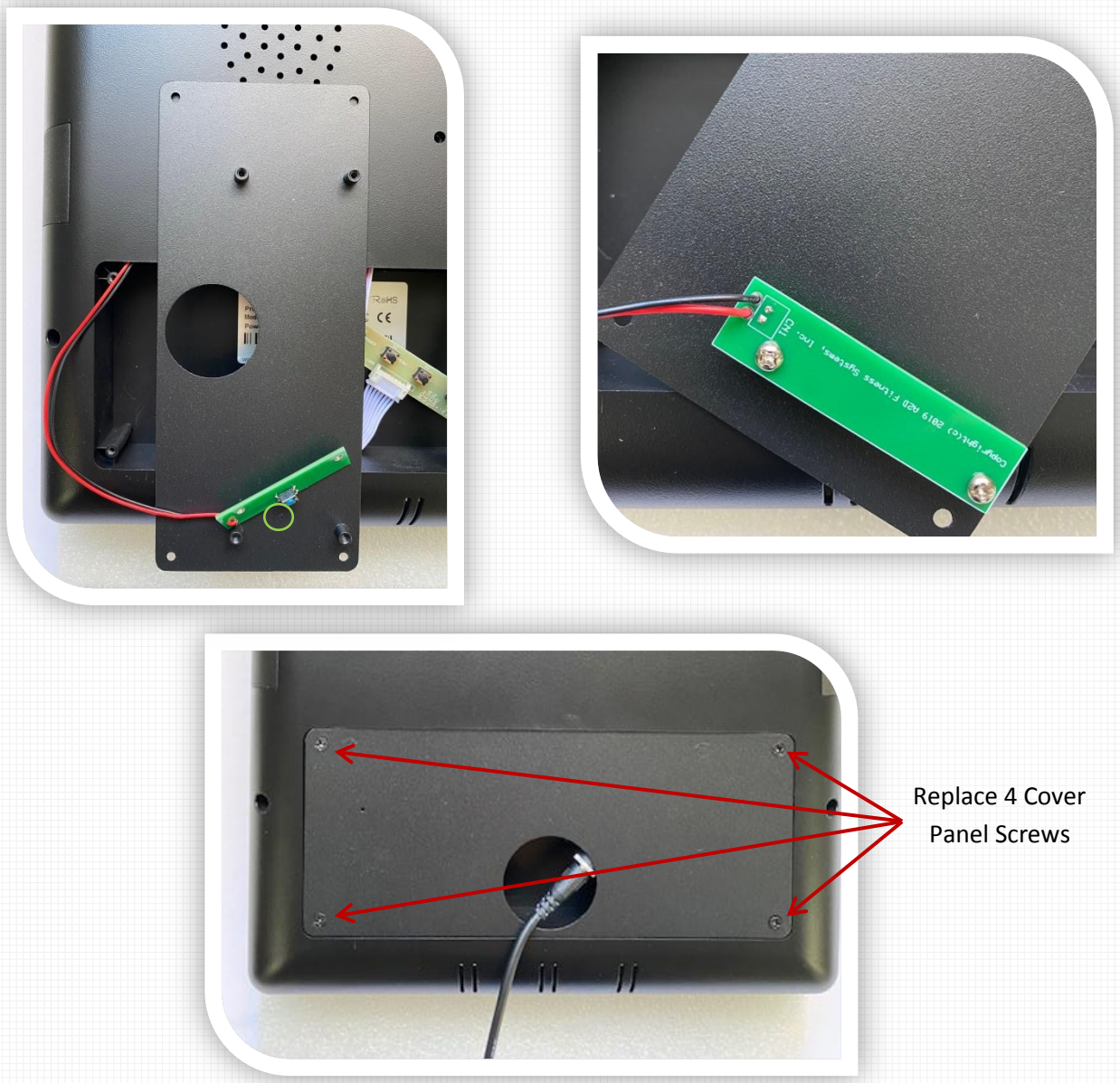


Figure 32 – Mounting the Reset Button

25) Turn the unit over so the display is facing upward, again being careful not touch the glass display.

26) To install the enclosure front panel, align the panel starting at the bottom, gently pushing down while moving upward toward the top, gently press the front panel downward toward the rear of the enclosure until the clips snap into place

Figure 33 - Enclosure Front Panel Installation

Touchscreen Driver

The eGTouch touchscreen driver is now supported by the Raspberry Pi OS; please make sure your Raspberry Pi is running the latest version Raspberry Pi Operating System. To download the latest version go to https://www.raspberrypi.org/downloads/

Installing Raspberry Pi OS

At the time of this writing, the newest Pi OS Version was: Version 5.4 with a Release date of: August 20th 2020. If you need to install the Raspberry Pi OS, you can follow the steps below or skip this section if you already have the Raspberry Pi OS installed.

The steps provided below are for a Windows computer; however the process is the same for MAC. You will need to attach a keyboard and mouse to ClassTouch in order to complete some of the setup steps below.

- From the downloads page at https://www.raspberrypi.org/downloads/
- First download and install the Raspberry Pi Imager if you have not already done so
- Run the Pi Imager
- Select the Operating System that you would like to use, here we use the "Raspberry Pi OS with desktop and recommended software"
- Insert the destination Raspberry Pi SD Card into your computer
- Format the SD Card by right clicking on the drive letter for the SD card and select 'Format', make sure the File System is set to 'FAT32', Select 'Quick Format' and click the Start button
- After the format has completed, return to the Imager application, select the SD card that will be used for the new image then click on the "WRITE" button
- When completed, remove the SD Card from your computer
- Insert the SD Card into the Raspberry Pi
- Make sure all cables including the Touchscreen controller cable are plugged into the Raspberry Pi
- Lastly, plug in the main 12VDC Power source for the ClassTouch™
- The Raspberry Pi should now be starting up
- Once booted up, enter the required information for the first time setup, it's recommended that you let the process check for any updates
- Leave the checkbox un-checked for "If the screen is showing a black boarder around the active display"
- Reboot the Raspberry Pi

In order to have the active display use the entire available screen, we will need to make a change to the display configuration file:

- From the Menu icon in the top left of the screen, select Preferences, then Main Menu Editor, From the left side panel scroll all the way down and click on Preferences, in the right side panel check the Display Settings check box, last click the OK button
- Again, click on the menu icon and scroll down to Preferences then click on Display Settings. From the drop down list select 1024 x 768 and click the Apply button, click OK then click the Save button
- From a terminal window type: “sudo nano /boot/config.txt”
- Use the Up/Down arrow keys to scroll down to the following lines:
    - **#disable_overscan=1** (this line must be un-commented if not already by removing the “#” character
    - The line should now read: **disable_overscan=1**
    - Now the next four lines must be commented out if not already:
        - **#overscan_left=-6**
        - **#overscan_right=-6**
        - **#overscan_top=-20**
        - **#overscan_bottom=-20**
        - **NOTE:** *It makes no difference what the four values are currently set at as adding the # character disables the entire command*
- To save the settings, press and hold the CTRL key and then press the “x” key to Exit
- Next press the “y” key to confirm saving the file
- Lastly press the RETURN key to save the file with the same filename
- To see the changes made to the display settings we must now reboot the Raspberry Pi, so in the terminal window, enter “reboot” and press the RETURN key
- After rebooting, the screen should now use all of the active display space
- You can make further adjustments as needed to the active display using the above steps and changing the four values as needed

ClassTouch™ Raspberry Pi Setup

After Raspbian has been installed, we suggest using a USB Keyboard and Mouse to complete the next few steps. There are a few things you need to do in order to use the Pi Power Controller on the Raspberry Pi, follow the steps below to enable the ClassTouch™ reset button and the Auto Startup feature:

1) In the **/home/pi** folder, create a new folder called "**Projects**"
2) In the **/home/pi/Projects** folder, create a new folder called "**PPC**"

PPC Python Script

3) Copy the following python script and name it "**Pi_Power_Controller.py**"
4) Save it into the **/home/pi/Projects/PPC** folder

```
#!/usr/bin/python
# Copyright (c) 2019 A2D Fitness Systems, Inc.
# Author: Malcolm J. Smith
#
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included in
# all copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
# THE SOFTWARE.

# Code Description:
# This code has 2 primary functions:
# 1 - Initialize the startup timer
# The maximum number of seconds is set to 4194303 (24 Bit) and written
# this also turns off the Square Wave Output
# The PPC uses the MAXIM DS1371 I2C 32-Bit Binary Counter
# See www.maxim-ic.com
```

```
# 2 - Poll the manual reset button
# The GPIO17 pin is set to input and is pulled HIGH by the PPC, if the
# the reset button is pressed then GPIO17 will go LOW, the LOW pin state
# then causes the goShutdown function to be called
# The GPIO4 pin is used to indicate to the PPC the Pi's boot state
# If GPIO17 or GPIO4 are needed for other purposes then any compatible
# GPIO's can be used assuming the electrical connections are changed accordingly

# Required Modules
import RPi.GPIO as gpio
import subprocess
import smbus
import time

# IF Required, Modify GPIO Pin assignment here
reset_control_io_pin = 4
reset_switch_io_pin = 17

def keepAlive():
        while True:
                time.sleep(10)

def goShutdown(pin):
        print("Shuting down...")
        time.sleep(2)
        subprocess.Popen(['sudo','shutdown', '-h', 'now'])

def init_startup_timer():
        address = 0x68
        Reg_Ptr = 4
        channel = 1
        bus = smbus.SMBus(channel)
        data_bytes = [255,255,255,79,0]
        bus.write_i2c_block_data(address,Reg_Ptr,data_bytes)


gpio.setmode(gpio.BCM)
gpio.setup(reset_control_io_pin,gpio.OUT, initial=gpio.HIGH)
gpio.setup(reset_switch_io_pin,gpio.IN)
gpio.add_event_detect(reset_switch_io_pin,gpio.FALLING, callback=goShutdown, bouncetime=2000)

# Comment this IF the Auto Startup Timer is NOT Needed
# AND remove Jumper JP1 from the PPC
init_startup_timer()

keepAlive()
```

Using PPC without the Startup Timer

If you want to have the ability to shut-down and start your Raspberry Pi using the reset button, but will not be using the auto-startup timer because the GPIO I2C pins are already in use by your project, please do the following:

1) Comment out the line **init_startup_timer()** in the Pi_Power_Controller.py script
2) Remove the Jumper (JP1) from the Pi Power Controller (PPC)

NOTE: *If the I2C GPIO Pins are required for other purposes then complete the following steps. If the I2C Pins are not needed for other purposes, then we recommend not removing the connector and wires making the auto-startup timer accessible for later use if required*

3) Remove the 2 Pin JST Connector
4) Remove the SDA and SCL wires from the 6 Pin housing to make available the I2C Pins

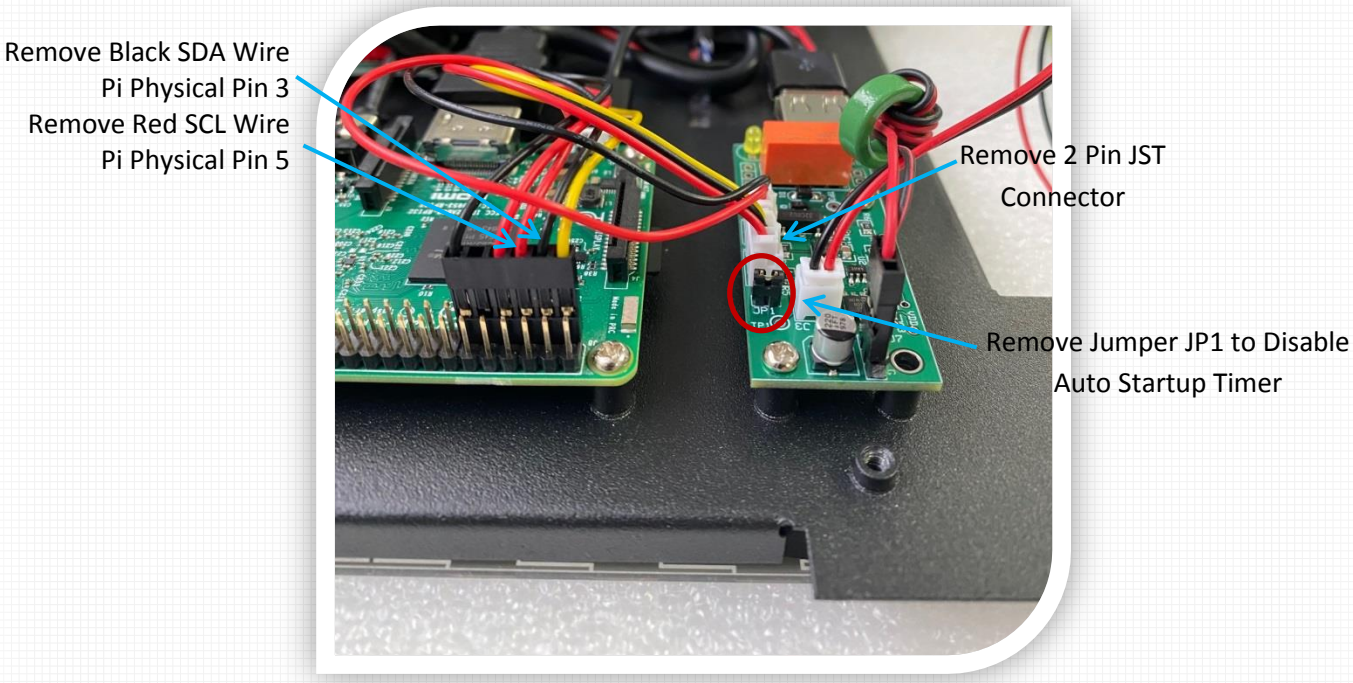


Figure 34 - Disable Auto Startup Timer

Run Python Script at Startup

You now need the ability to start the Pi_Power_Controller.py script every time the Pi is started, to do that we add one line to the rc.local file.

In a terminal, edit the file **/etc/rc.local** using the editor of your choice. You must edit it with root permissions:

Here we used the nano text editor:

```
sudo nano /etc/rc.local
```

Add the following line to the end of the file but before the **exit 0** command

```
python3 /home/pi/Projects/PPC/Pi_Power_Controller.py &
```

Be sure to leave the line **exit 0** at the end, then save the file and exit. In nano, to exit, type **Ctrl-x**, and then **Y**.

Pi_Power_Controller.py Description

The python script Pi_Power_Controller.py when run at startup will run continually in the background, there is no console output associated with it. The script will first set GPIO 17 HIGH, this indicates to the PPC that the script has been executed and is ready. Next GPIO 4 is set as an input and an event established for a HIGH to LOW transition. Next the Auto Startup Timer is initialized using the init_startup_timer() function, setting the timer to its maximum number of seconds; this also turns off the square wave output that is present by default. All communication between the Raspberry Pi and the PPC's Auto Startup Timer is accomplished via I2C. The Raspberry Pi's physical pins 3 and 5 are the SDA and SCL signals respectively and connect via a 2 pin JST connector on the PPC. After the initialization process has completed, the script then starts the keepAlive() function, this runs continually and does nothing more than runs a sleep command forever but without this the script would end.

When the reset button is pressed, the HIGH to LOW transition on GPIO 4 will trigger the event goShutdown(), this function then issues the shutdown command to the Raspberry Pi that will cause the Pi to shut down. After a few seconds the PPC is alerted that the Raspberry Pi has been shut down. The PPC then waits approximately 30 seconds before enabling the reset button again; this allows the Pi to perform a Re-Boot if that was the cause of the shutdown or allows the boot up process to occur if the reset button is pressed, and lastly monitors the Auto Startup Timer, checking if the timer has expired and if so will trigger the auto startup process. If after 30 seconds no events have occurred, the power to the Pi is disconnected completely; this ensures that power consumption by the Raspberry Pi and all attached USB devices is at zero.

GPIO Pins 17 and 4 (physical Pi pins 11 and 7 respectively) can be substituted for different GPIO Pins if needed, first simply change the pin assignment for the two pins at the top of the python script to the desired new GPIO Pin assignment, next you will need to remove the two wires from the 6 pin header and relocate them to the desired new

GPIO Pins. For orientation purposes: on the 6 pin header, Pin #1 (yellow wire) is also Raspberry Pi's Pin #1, GPIO 4 is Pin #4 of the header and GPIO 17 is Pin #6 of the header, Pin 5 of the 6 position header is not used.

Auto Startup Python Script Example

```
#!/usr/bin/python
# Copyright (c) 2019 A2D Fitness Systems, Inc.
# Author: Malcolm J. Smith
#
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included in
# all copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
# THE SOFTWARE.

# Code Description:
# Simply set the number of seconds to wait until startup
# This will shutdown the Raspberry Pi, please save any modifications
# before running this script

# Required Modules
import smbus
import time
import subprocess

channel = 1
address = 0x68

Secs_Until_Startup = 120

bus = smbus.SMBus(channel)

# Initialize the Timer with maximum Seconds
Reg_Ptr = 4
I2C_Bytes = [255,255,255,79,0]
bus.write_i2c_block_data(address,Reg_Ptr,I2C_Bytes)
```

```
# Set control registers
Reg_Ptr = 7
I2C_Bytes = [15,0]
bus.write_i2c_block_data(address,Reg_Ptr,I2C_Bytes)

# Get 3 integer values from Secs_Until_Startup
Reg_Ptr = 4
I2C_Bytes = list((Secs_Until_Startup >> i) & 0xFF for i in range(0,24,8))
print("I2C Bytes:",I2C_Bytes)

# Add Register values
I2C_Bytes += [79,0]
bus.write_i2c_block_data(address,Reg_Ptr,I2C_Bytes)

print("Startup Timer is now counting down, when the timer reaches 0, Auto Startup will occur...")
print("Shutting down in 10 Seconds")
time.sleep(10)

# Shuting down NOW
subprocess.Popen(['sudo','shutdown', '-h', 'now'])
```

Auto_Startup_Test.py Description

The example script demonstrates the overall process of using the Auto Startup Timer and also details the python code needed to accomplish the task. To run the example script, simply modify the value as needed for the Secs_Until_Startup variable to the desired number of seconds to wait until Auto Startup occurs. Please note that when the script is run and after 10 seconds the Raspberry Pi will shut down, so remember to save any changes you have made to the script before running it. To learn more about the DS1371 silicon go to the maxim integrated website:

https://www.maximintegrated.com/en/products/analog/real-time-clocks/DS1371.html

Enable I2C Interfaces

As the PPC uses I2C for communications with the Raspberry Pi, it must be first turned on. If the init_startup_timer() function is called from the python script and I2C is not turned on in the raspi-config interfacing options section, then an error will be generated by the script.

Please enable I2C as shown below if the Auto Startup Timer is to be used.

NOTE: *I2C does not need to be turned on if the Auto Startup Timer is not used.*

In a terminal, enter **`sudo raspi-config`**

From Raspberry Pi Software Configuration Tool Menu

- Select **5** Interfacing Options -> **I2C**-> **Yes** -> **OK** -> **Finish**

Display Driver Keypad Usage

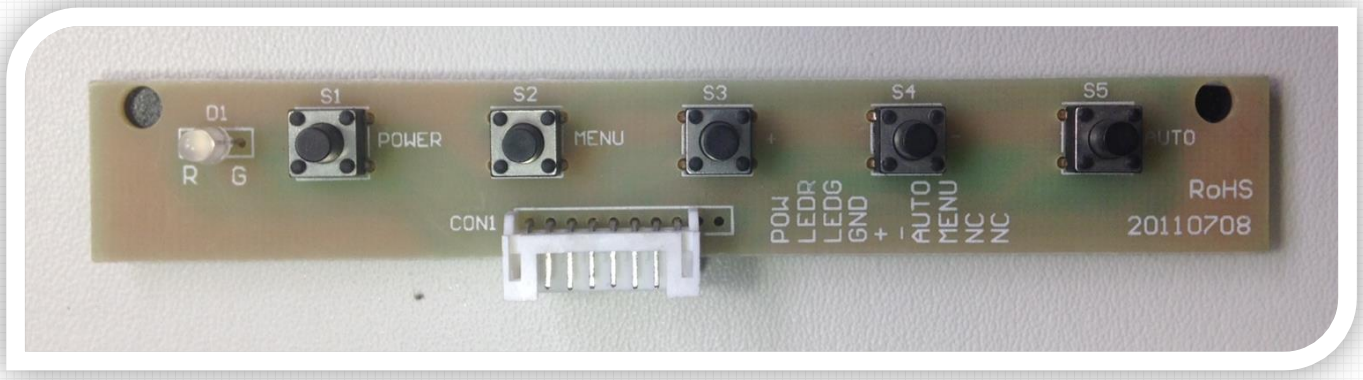


Figure 35 - Display Driver Keypad

Turning Display On and OFF

Pressing S1 (POWER) will Toggle the power to the display driver ON or OFF

Display Driver LED Status

The LED will be Green if the power is turned ON to the display and RED if turned OFF

Setting Audio Volume

Press S3 (+) to Increase Volume and S4 (-) to Decrease Volume

Selecting Language

Press S2 (MENU), Press S3 (Right), Press S3 (Right) Press S5 (Up/Down), Press S5 (Up/Down), Press S4 (Right) to select the language required then Press S2 (MENU) to select it, the language selection will be displayed white, lastly Press S2 (MENU) to exit the menu options

Selecting Display Menu

- Press S2 (MENU) to display the menu
- Press S3 to move Right
- Press S4 to move Left
- Press S5 to move Up and Down
- Press S2 (MENU) button to lock in the changed option
- Press S2 (MENU) to Exit the menu options

Table of Figures