CryptoCore OTTOcontrol Architecture Reference

CryptoCore OTTOcontrol Software Version: 1.0

April 15, 2022

Part Number: D-RES-0000x

Contents

[1.0 OTTOcontrol Overview 3](#_Toc100849665)

[2.0 Hardware Stack 8](#_Toc100849666)

[2.0.1 Hardware Details 9](#_Toc100849667)

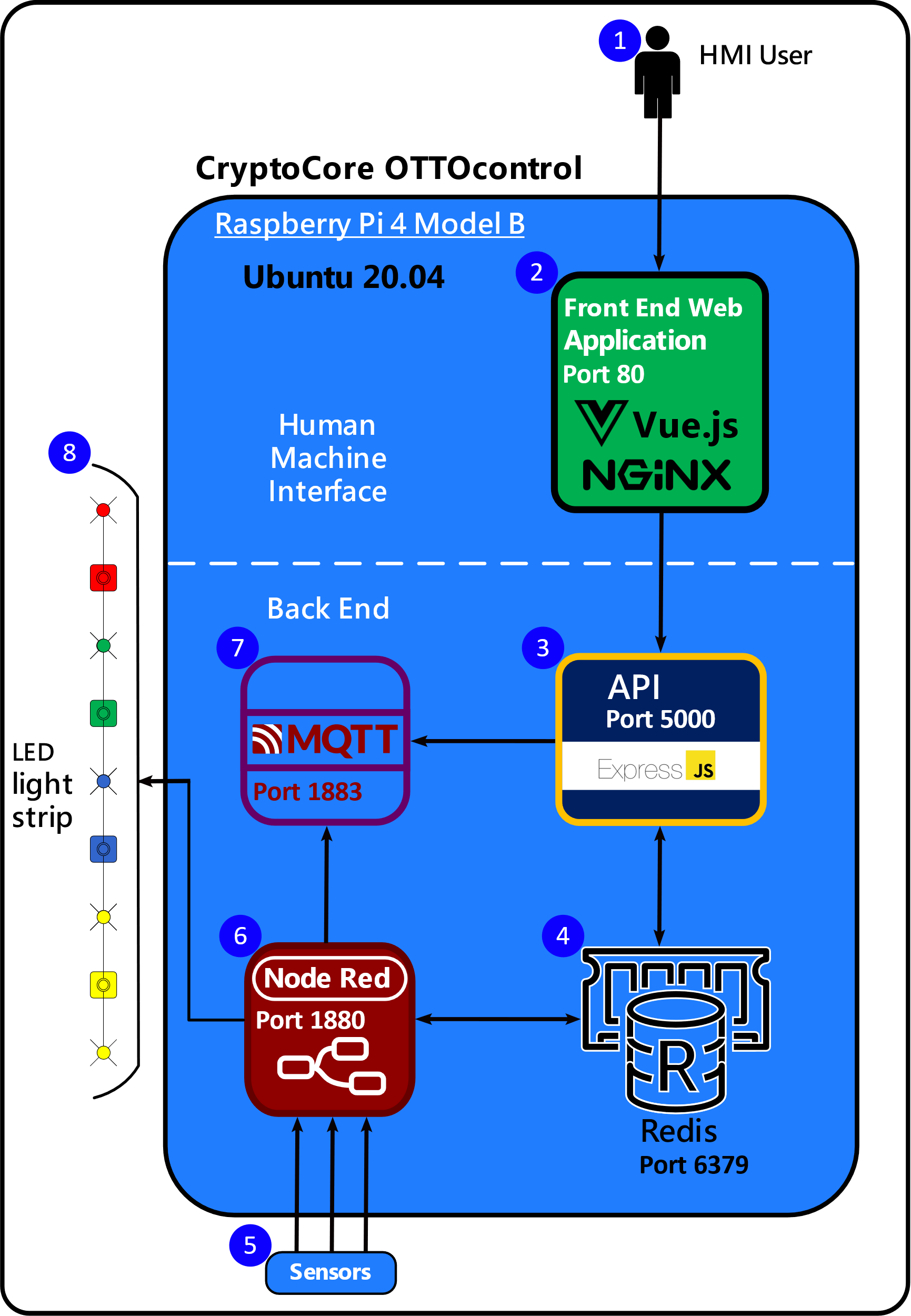
[2.0.2 Raspberry Pi I/O 10](#_Toc100849668)

[Addendum A – Software Installation and Upgrade 12](#_Toc100849669)

[Addendum B – Intellectual Property Rights 12](#_Toc100849670)

## 1.0 OTTOcontrol Overview

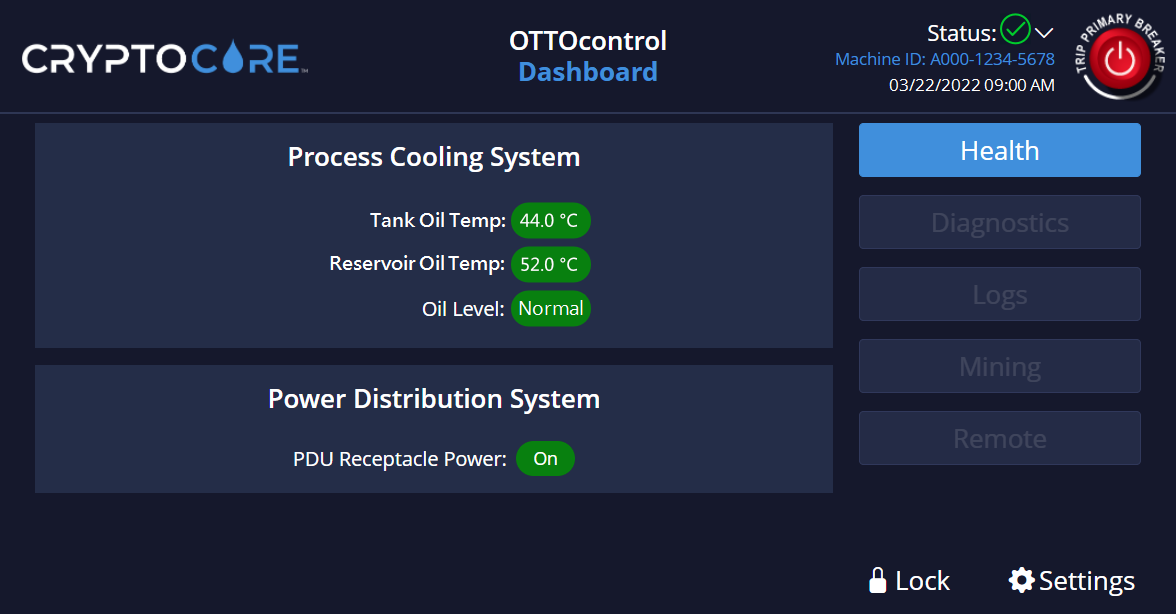
CryptoCore OTTOcontrol provides the HMI user interface (Front End) and necessary back-end software used to manage and monitor a CryptoCore tank and the associated miner hardware running in the CryptoCore tank. This document provides a high-level description of the CryptoCore OTTOcontrol software installed on a Raspberry Pi 4 Model B. The following diagram illustration provides an overview of the major components of CryptoCore OTTOcontrol:



*Figure 1*

The numbered components highlighted in this illustration are described in further detail below:

1. **Engineer at the box** – This icon depicts the 'Human' part of a Human Machine Interface; this represents a person that is physically located at a CryptoCore unit where they can control the unit through a physically attached touch screen panel.
2. **Front End HMI** – The Human Machine Interface is a web application exposed via HTTP on port 80. The HMI is developed using the VUE.js framework and is hosted on a NGINX web server. The HMI is also referred to as a ‘Dashboard’ with a touchscreen interface similar to the following:



*Figure 2*

For more information about VUE.js see [**https://vuejs.org/**](https://vuejs.org/). For more information about NGINX see [**https://nginx.org**](https://nginx.org/).

1. **API** – The API is built on the ExpressJS web application framework, exposed via HTTP on port 5000. All calls to the backend from the Dashboard are made via the API using HTTP verbs such as GET, POST, PUT or DELETE. For more information about ExpressJS see [**https://expressjs.com**](https://expressjs.com).
2. **Redis** – Redis is an open source, in-memory, NoSQL key/value store, used primarily as an application cache but also providing data persistence for the purposes of the CryptoCore OTTOcontrol application. The Redis server listens on port 6379. For more information about Redis see [**https://redis.io**](https://redis.io/).
3. **Sensors** – Includes sensors for detecting tank and reservoir fluid temperature and fluid level. The sensor below is an example of a temperature sensor that may be installed on a CryptoCore unit.



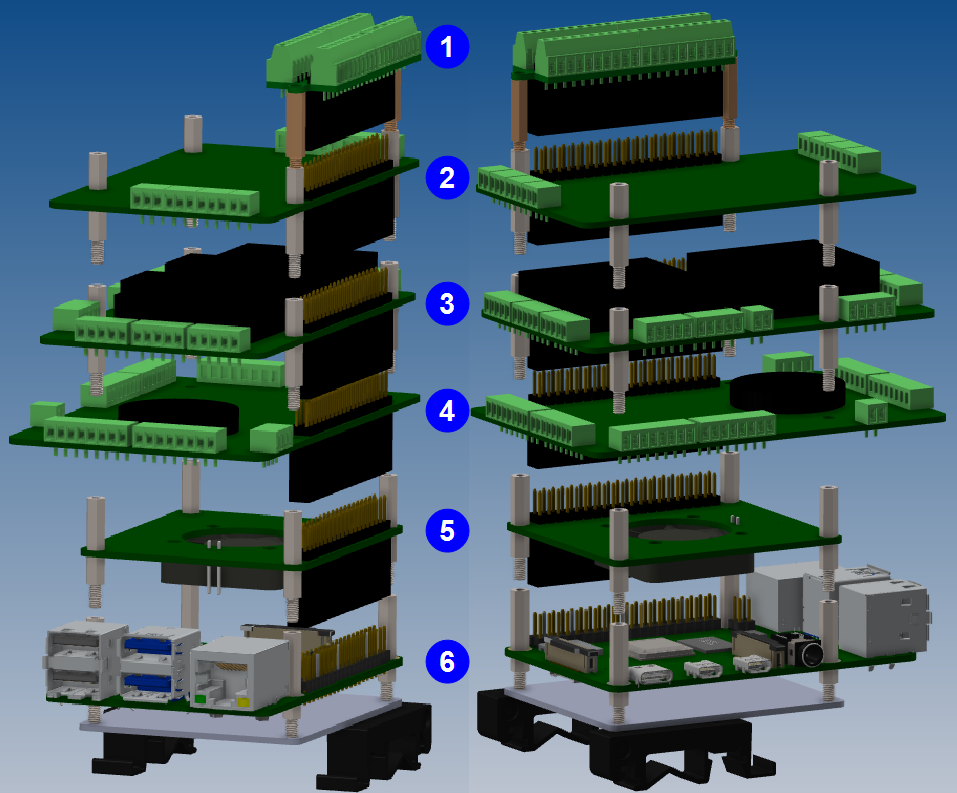
*Figure 3*

1. **Node Red** – Node-RED is a flow-based development tool originally created by IBM for wiring together hardware devices, APIs, and online services. Node-RED enables Programmable Logic Control (PLC) functionality for monitoring various hardware installed on each CryptoCore unit. PLC functionality enables the software to periodically monitor various aspects of the CryptoCore unit such as voltages, fan speeds, temperatures, and other relevant metrics. Node Red listens on port 1880. For more information about Node RED see [**https://nodered.org**](https://nodered.org).
2. **MQTT Broker** – The Mosquitto Message Queuing Telemetry and Transport (MQTT) Broker is a store and forward message routing service listening on port 1883. The MQTT broker will be used to publish messages received from the API and Node-RED to remote subscribers for purposes of remotely managing CryptoCore units via OTTOmanager software. For more information about the Mosquitto MQTT broker see [**https://mosquitto.org**](https://mosquitto.org).
3. **External LED lighting** – External LED lights are installed on CryptoCore units to quickly notify end users of the current operational status of each unit. The following image is an example of string lighting that may be installed on a CryptoCore unit. External LED Lights change color in response to alarm conditions or other operating circumstances.



*Figure 4*

## 2.0 Hardware Stack

*Figure 5*

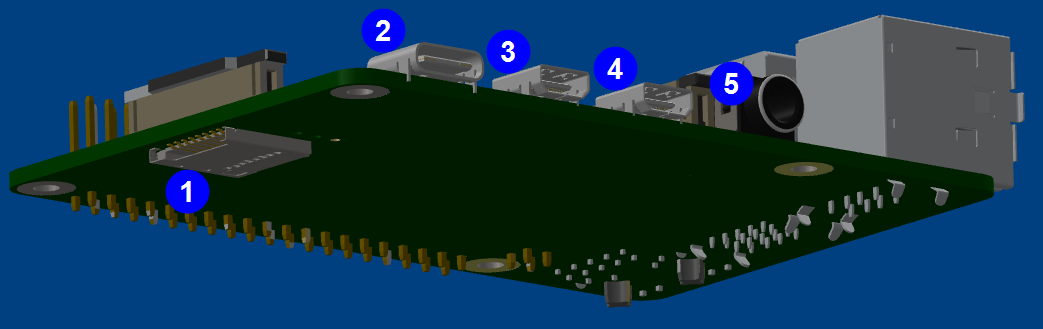
### 2.0.1 Hardware Details

Each of the numbered items in Figure 18 is described in further detail below:

1. **CZH-Labs Ultra-small RPi GPIO Terminal Block Breakout Board Module** – This unit is used to connect to and facilitate communication with the CryptoCore LED strip. For more information see [**RPi Terminal Block Breakout Board Module**](https://czh-labs.com/products/ultra-small-rpi-gpio-terminal-block-breakout-board-module-for-raspberry-pi).
2. **Sequent Microsystems RTD Data Acquisition HAT** – This unit receives data from various temperature sensor inputs distributed throughout the CryptoCore unit. For more information see [**RTD Data Acquisition HAT**](https://sequentmicrosystems.com/collections/all-io-cards/products/rtd-data-acquisition-card-for-rpi).
3. **Sequent Microsystems Eight Relay HAT –** This unit handles dry contact mechanisms on the CryptoCore system and sends the appropriate commands to the control unit. For more information see [**Eight Relay HAT**](https://sequentmicrosystems.com/collections/all-io-cards/products/raspberry-pi-relays-stackable-card?variant=37993168732355).
4. **Sequent Microsystems Industrial Automation HAT** – This unit provides various electrical interfaces to communicate with sensors and digital I/O. For more information see [**Industrial Automation HAT**](https://sequentmicrosystems.com/collections/all-io-cards/products/industrial-raspberry-pi)
5. **Sequent Microsystems Smart Fan HAT** – This unit is the cooling solution used for the Raspberry Pi CPU, for more information see [**Smart Fan HAT**](https://sequentmicrosystems.com/collections/all-io-cards/products/raspberry-pi-fan?variant=37993169256643).
6. **Raspberry Pi 4 Model B with 4 GB Ram** – This unit supplies the computing and storage requirements for the OTTOcontrol software stack. The operating system is Ubuntu Linux 20.04 which hosts several software components including services for NODE Red and Redis, programming logic for processing various I/O and the front-end software for the Human Machine Interface (HMI). For more information about the Raspberry PI 4 model B see [**Raspberry Pi 4**](https://www.raspberrypi.com/products/raspberry-pi-4-model-b/).

### 2.0.2 Raspberry Pi I/O

The Raspberry Pi 4 provides multiple interfaces for connecting to external devices. The following diagram depicts a Raspberry Pi 4 as viewed from underneath:

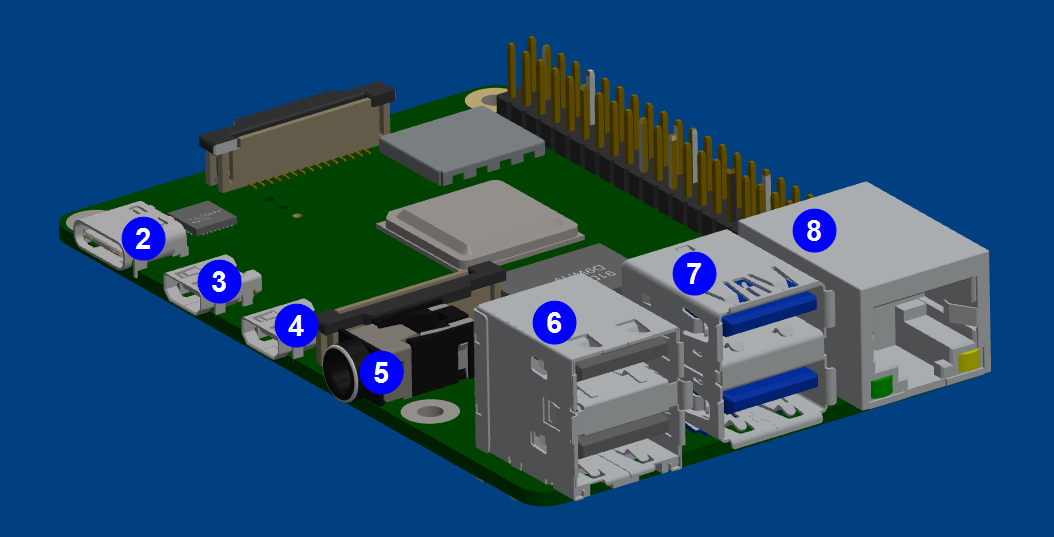


*Figure 6*

Each of the numbered items in Figure 19 is described in further detail below:

1. **MicroSD slot** – Accepts a MicroSD card to house the Raspberry Pi file system.
2. **USB-C Power supply input** – Input for a 15W USB-C power supply.
3. **Micro HDMI port** – Provides connectivity to a 4K display.
4. **Micro HDMI port** – Provides connectivity to a 4K display.
5. **Audio-video port** – 3.5 mm analog audio-video jack.

The following diagram depicts a Raspberry Pi 4 as viewed from the top:



*Figure 7*

Each of the numbered items in Figure 20 is described in further detail below:

1. **USB-C Power supply input** – Used to connect a 15W USB-C power supply.
2. **Micro HDMI port** – Provides connectivity to a 4K display.
3. **Micro HDMI port** – Provides connectivity to a 4K display.
4. **Audio-video port** – 3.5 mm analog audio-video jack.
5. **USB 2.0 ports** – 2 USB 2.0 ports
6. **USB 3**.**0 ports** – 2 USB 3.0 ports
7. **Ethernet port** – Gigabit ethernet port

## 3.0 CryptoCore OTTOcontrol Architecture Lab

This lab should help illustrate communication between the various software components described in section 1.0. To get started, you will need access to the hardware stack (including a physical keyboard, mouse, and display) with the latest build of CryptoCore OTTOcontrol installed. The following photograph is of a CryptoCore OTTOcontrol Raspberry Pi kit created by the manufacturing team with an ethernet connection, mouse, and keyboard:



*Figure 8*

### 3.0.1 Configure your Hardware Kit for RDP

For purposes of this lab, follow these steps to configure your Raspberry Pi for remote desktop access.

1. Open terminal on your Pi with the Ctrl+Alt+t key combination.
2. Run the following commands to install and enable Remote Desktop server on your Pi:
   * su – tmgadmin (switch user to the tmgadmin account)
   * Enter the password tmgcore when prompted and press Enter.
   * sudo apt install xrdp -y
   * sudo ufw allow from any to any port 3389 (allow RDP through firewall)
   * sudo ufw reload (reload firewall)
3. Run the following command to determine the IP address of your Pi:
   * ip a

This command will return a list of your network connections and associated IP addresses. You want to find the inet value associated with the eth0 connection, this IP address is what you will connect to from your laptop computer via the Remote Desktop protocol.

### 3.0.2 Connect to your Pi remotely

1. From your laptop computer, right-click **Start**, select **Run**, type mstsc and press **Enter** to launch the Remote Desktop Connection client.
2. Enter the Pi’s IP address as determined in step 3 above for **Computer:** and click **Connect**.
3. When prompted enter **tmgadmin** for username and **tmgcore** for password.
4. Click **OK**.
5. If prompted, enter the password **tmgcore** again

### 3.0.3 Open the Redis CLI and view key/value pairs

1. From your Remote Desktop connection, click **Menu**, select **System Tools,** and click **MATE Terminal** to open a new terminal session.
2. Enter the command redis-cli and press **Enter** to open the Redis command line interface, note that the command prompt reflects the port that Redis is listening on (6379).
3. From the Redis command line interface, type keys \* and press **Enter**. This should return a list of the keys currently stored in Redis running on the Pi.
4. To retrieve the value associated with a particular key, type get <key> and press **Enter**, where <key> is one of the keys returned by the keys \* command. For example, if you type get iaHatVersion and press **Enter**, you should see the value “1.1” returned.
5. To set the value associated with a key, type set <key> and press **Enter**. Note that the value you set a key to will almost immediately be updated by Node Red so if you set a key to a particular value and then get the value of that same key it will very likely be a different value than the value that you set the key to.
6. Enter the command set test true to create a new key named test with a value of true. Since this new key is not being updated by Node Red, you should be able to retrieve the value of the key test as true with the command get test.
7. Enter the command del test to remove the key/value pair test:true from the Redis server.
8. Enter the command get test to verify that a result of (nil) is returned and confirm that test:true is removed from Redis.
9. Type the command keys \* and press **Enter** to view the list of keys. You can review this list to verify that the key test has been removed.

## Addendum A – Software Installation and Upgrade

For information about installing software on your Raspberry Pi see [**Install CryptoCore OTTOcontrol on a Raspberry Pi**](https://tmgcore0.sharepoint.com/:w:/s/TMGcoreInternalSharePoint/Ecx6Q1nO05tAj5oHcBBtUnYB1LdE4e685LBRffTZlRlrqg?e=84ETah).

## Addendum B – Intellectual Property Rights

Unless otherwise indicated, this documentation is TMGcore, Inc. proprietary information and all functionality, software, text, and graphics in this document and the trademarks, service marks, and logos contained therein are owned or controlled by us or licensed to us and are protected by copyright and trademark laws and various laws and various other intellectual property rights and unfair competition laws of the United States, foreign jurisdictions, and international conventions. The Content is provided in this document “AS IS” for your information and personal use only. Except as expressly provided in these Terms of Use, no part of this document may be copied, reproduced, aggregated, republished, uploaded, posted, publicly displayed, encoded, translated, transmitted, distributed, sold, licensed, or otherwise exploited for any commercial purpose whatsoever, without express prior written permission.