QUANSER
INNOVATE·EDUCATE

USER MANUAL

QBot 2e for QUARC

Set Up and Configuration

CAPTIVATE. MOTIVATE. GRADUATE.

1 Introduction

The Quanser QBot 2e (Figure 1.1) is an innovative autonomous ground robot system incorporating a robust educational ground vehicle with the Microsoft Kinect and a QUANSER® embedded target. The QBot 2e is comprised of a Yujin Robot Kobuki platform, a Microsoft Kinect RGB camera and depth sensor, and a wireless embedded computer (also refered to as the target computer). The embedded computer system mounted on the vehicle uses the Raspberry Pi 3 Model B+ computer [1] to run QUARC®, Quanser's real-time control software, and interface with the QBot 2e data acquisition card (DAQ).

Figure 1.1: The Quanser QBot 2e

The interface to the target computer is MATLAB® SIMULINK® with QUARC. The QBot 2e is accessible through three different block sets: the Quanser Hardware-In-the-Loop (HIL) block set to read from sensors and/or write to outputs, the Quanser Stream API blockset to perform communications over wired and wireless communication channels, and the Quanser Multimedia blockset to read RGB and depth image data from the Kinect sensor. Controllers are developed in Simulink with QUARC on the host computer, and these models are cross-compiled and downloaded to the target (Raspberry Pi 3 Model B+[1]) seamlessly. A diagram of this configuration is shown in Figure 1.2.

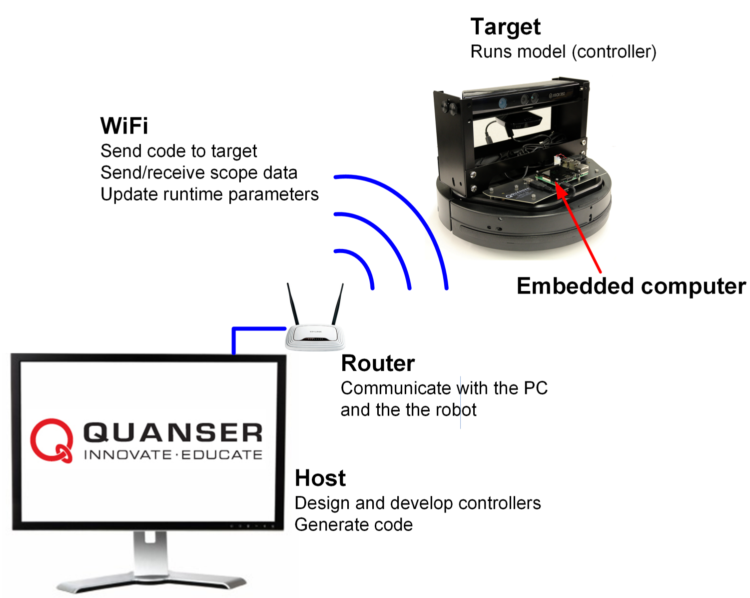


Figure 1.2: Communication Hierarchy

The general system description, component nomenclature, specifications, and model parameters are all given in Section 2. Section 3 goes into detail on how to setup the QBot 2e. Lastly, Section 4 contains a troubleshooting guide.

1.1 Prerequisites

To successfully operate the QBot 2e, the prerequisites are:

1. To be familiar with the components of the QBot 2e.
2. To have QUARC 2018 SP1 (version 2.7.2582) or later installed and properly licensed.
3. To be familiar with using QUARC control and monitor the vehicle in real-time, and in designing a controller through Simulink. See Reference [2] for more details.

1.2 References

[1] Raspberry Pi 3 Model B+: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/
[2] QUARC User Manual (type `doc quarc` in MATLAB to access)

2 Components

The QBot 2e is made up of four main components: the Kobuki robot platform, the QBot 2e, the Raspberry Pi 3 Model B+ embedded computer, and the Kinect sensor. This section outlines these components in more detail.

2.1 The Kobuki Robot Platform

The QBot 2e uses an Kobuki mobile robot platform (Figure 2.1). The QBot 2e follows the Quanser standard for body frame axes, where the x-axis is in the forward direction, the y-axis is to the left, and the z-axis is up. The diameter of the vehicle is 34 cm, and its height (without attachments) is 10 cm. The Kobuki is driven by two differential drive wheels with built-in encoders. The Kobuki comes with a bumper sensor as well as a built-in gyroscope and cliff sensors. The embedded computer target can access data from these sensors.

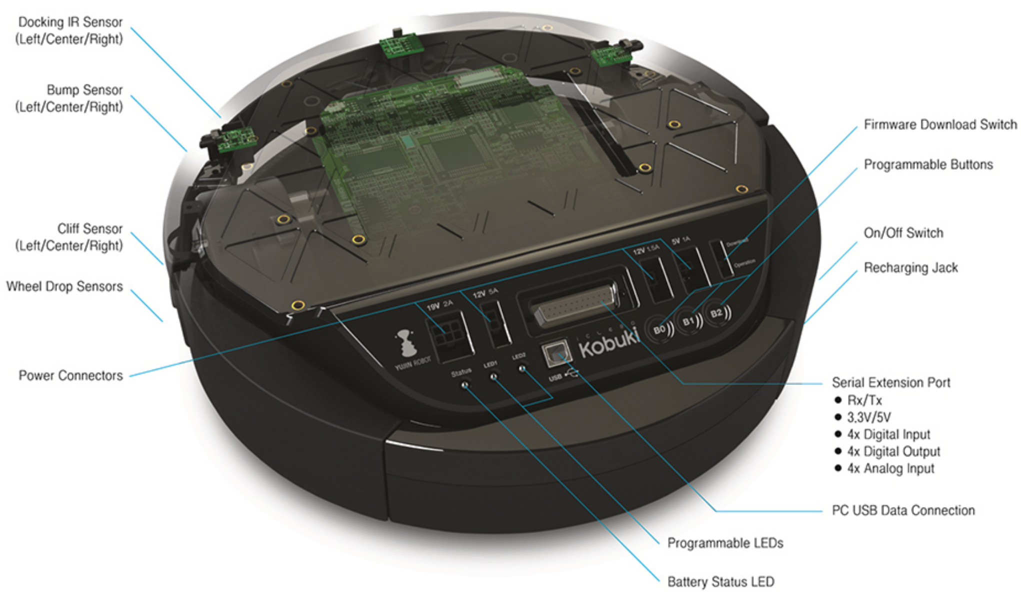


Figure 2.1: The Kobuki mobile robot platform

The QBot 2e is powered by a Lithium ion battery pack (Figure 2.2a) provided by Yujin Robot. The battery fits underneath the QBot 2e, and can last continuously for about 3 hours after a full charge. The battery takes less than 3 hours to charge. While charging, the power light pulses slowly with an orange colour. A battery charger is provided (Figure 2.2b). To recharge the QBot 2e, plug in the battery charger and connect it to the charger input port on the QBot 2e next to the *ON/OFF* switch (Figure 2.2c).

(a) The QBot 2e battery

(b) The QBot 2e charger

(c) The QBot 2e charger input

Figure 2.2: The QBot 2e battery and charger input

2.2 System Specifications and Model Parameters

Table 2.1 lists the main parameters associated with the QBot 2e.

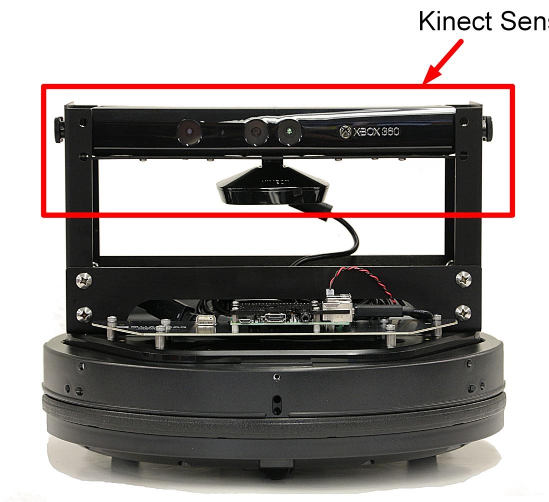
Symbol	Description	Value	Unit
D	Diameter of the QBot 2e	0.35	m
d	Distance between the left and right wheels	0.235	m
h	Height of the QBot 2e (with Kinect mounted)	0.27	m
ν_{max}	Maximum speed of the QBot 2e	0.7	m/s
m	Total mass of the QBot 2e	3.82	kg

Table 2.1: QBot 2e specifications

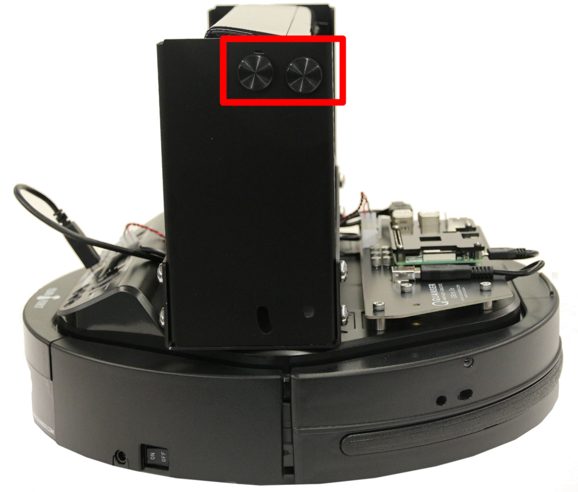
2.3 The Kinect Sensor

The Microsoft Kinect sensor is an integrated RGB camera and depth sensor used in a variety of experiments (Figure 2.3a). The Kinect camera provides RGB image capture and 11-bit depth sensing at a resolution of 640×480. The

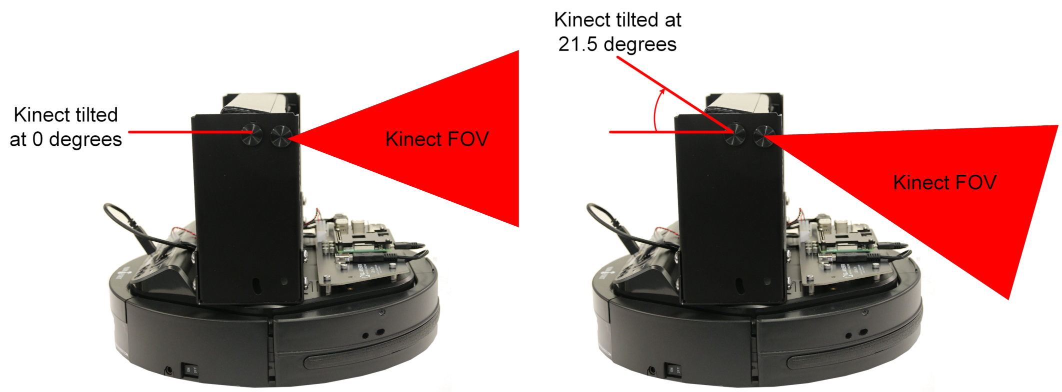
Kinect's depth sensor utilizes infrared light and has a range of 0.5 m to 6 m. Due to the type of infrared sensor, the QBot 2e Kinect should only be used indoors in locations without direct sunlight for best results.



(a) The QBot 2e Kinect sensor



(b) The QBot 2e Kinect thumb screws



(c) The QBot 2e Kinect tilted at 0 degrees and 21.5 degrees

Figure 2.3: The QBot 2e Kinext sensor

The QBot 2e Kinect is mounted on a structure that allows the Kinect to be oriented horizontally as well as tilted downward for various viewing positions. The QBot 2e Kinect structure has four thumb screws (two on each side) as shown in Figure 2.3b. To adjust the Kinect's viewing angle, loosen all of the thumb screws slightly but do not remove them all the way. Tilt the Kinect to the desired angle and tighten all of the thumb screws. The maximum and minimum angles are designed to allow the kinect to be oriented facing stright ahead (tilted at 0 degrees) as well as at an angle of 21.5 degrees such that the ground in front of the QBot 2e in the viewing angle as shown in Figure 2.3c.

2.4 Embedded Computer and PCB

The QBot 2e uses a Raspberry Pi 3 Model B+ as a small-scale embedded computer that runs the QUARC runtime. The Raspberry Pi 3 Model B+ interfaces with the Kobuki as well as the Kinect sensor. With QUARC installed, code generated from MATLAB Simulink is cross-compiled, downloaded, and executed directly on the Raspberry Pi 3. As shown in Figure 2.4, the Raspberry Pi 3 is mounted on a PCB which itself is mounted on the Kobuki chassis base. The Raspberry Pi 3 also comes with integrated 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN to allow wireless connection between the target Raspberry Pi 3 and the host computer and/or other vehicles.

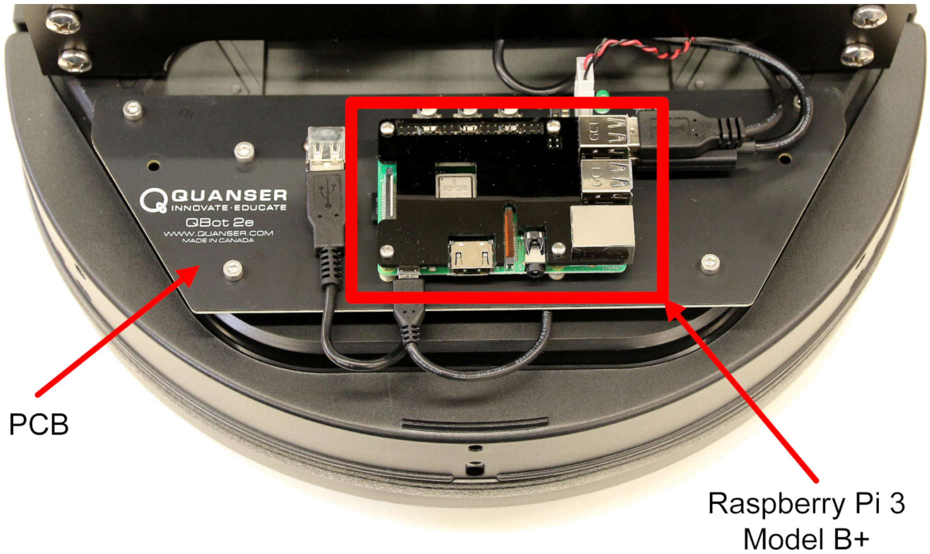


Figure 2.4: The Raspberry Pi 3 mounted on the front PCB

Below are the full specifications of the Raspberry Pi 3 Model B+:

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- 1GB LPDDR2 SDRAM
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- Extended 40-pin GPIO header
- Full-size HDMI
- 4 USB 2.0 ports
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- 4-pole stereo output and composite video port
- Micro SD port for loading your operating system and storing data
- 5V/2.5A DC power input
- Power-over-Ethernet (PoE) support (requires separate PoE HAT)

For further information on the Raspberry Pi 3 Model B+ refer to [1].

As noted earlier, both the Kinect sensor and the Kobuki base interface with the Raspberry Pi 3 and PCB. As shown in Figure 2.5, the Kobuki 12 V, 5 A power source is connected to the PCB using a twisted pair of cables that are terminated using 2-pin connectors. This connection provides power form the Kobuki chassis to the PCB board.

The USB-A socket on the PCB is used to power the Raspberry Pi 3. As shown in Figure 2.6, the USB-A socket on the PCB is connected to the micro USB-B connector on the Raspberry Pi 3 Model B+ using a 0.3 m USB-A to micro USB-B cable.

Figure 2.5: Supplying power to the PCB board

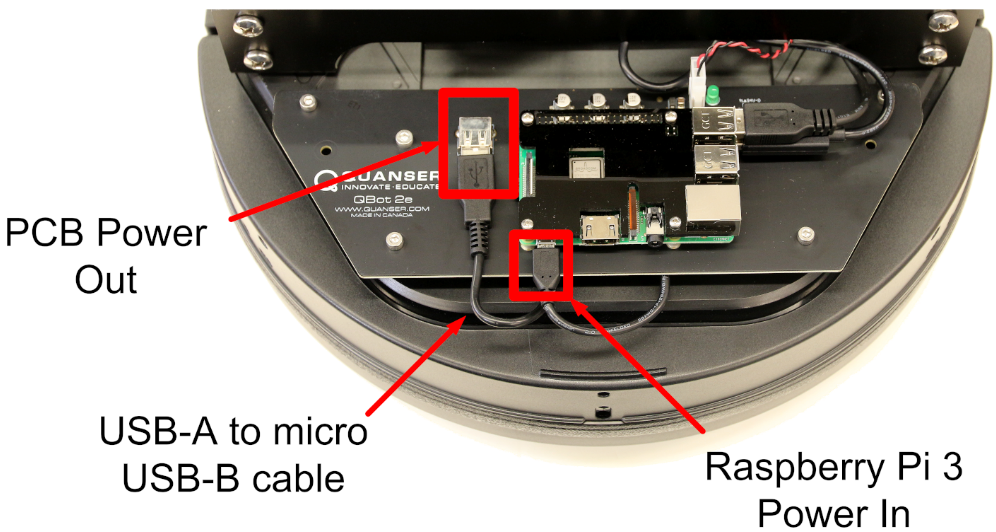


Figure 2.6: Supplying power to the Raspberry Pi from the the PCB

As shown in Figure 2.7, the USB-B socket on the Kobuki chassis is directly connected to one of the USB ports on the Raspberry Pi 3 using a USB-A to USB-B cable. In a similar fashion, and as shown in Figure 2.8, the Kinect sensor plugs directly into one of the USB ports of the Raspberry Pi 3 .

Figure 2.7: Connecting the Kobuki chassis to the Raspberry Pi 3

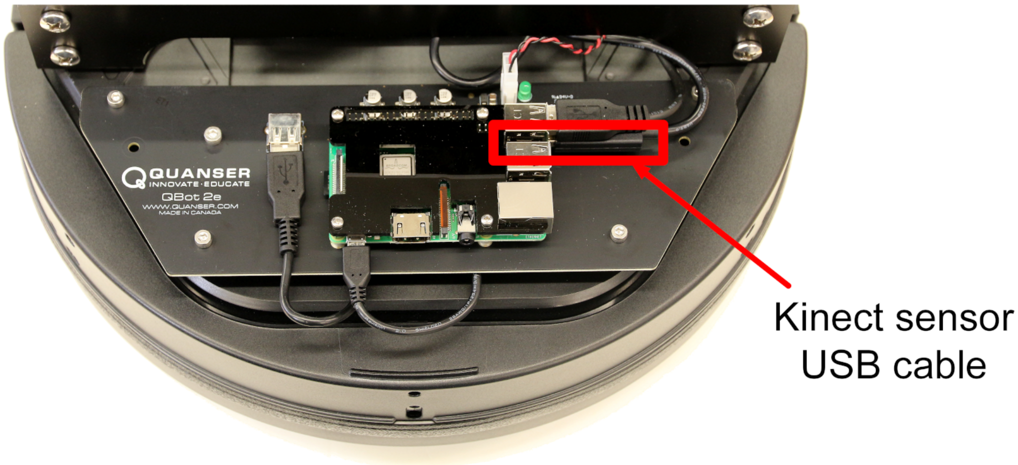


Figure 2.8: Connecting the Kinect sensor to the Raspberry Pi 3

2.5 QBot 2e Sensors Interface

This section provides a summary of the blocks that are used to read the QBot 2e sensors in Simulink, write outputs to the QBot 2e motors, as well as Raspberry Pi 3 Model B+ general purpose I/O accessible to the user via the Raspberry Pi GPIO connector.

The QUARC Hardware-In-the-Loop (HIL) blockset it used to communicate with the QBot 2e sensors and motors. For detailed information on the HIL blockset see the QUARC HIL user guide in the MATLAB help under *QUARC Targets* | *User's Guide* | *Accessing Hardware*.

2.5.1 QBot 2e I/O

The QBot 2e provides an interface to the I/O of the Kobuki robot platform including its wheel encoders, sensors, and motor outputs, as well as several general purpose I/O channels that can be used for interfacing with a variety of third-party sensors and actuators.

The general purpose I/O accessible to the user include:

- 2 PWM output channels, 3.3V
- 28 reconfigurable digital I/O, 3.3V
- 1 general purpose +3.3V serial port (see note below)
- 1 general purpose SPI port (see note below)
- 1 general purpose I2C port (see note below)
- Integrated with MATLAB/Simulink/Simulink Coder via QUARC

Note: Communications channels are accessed through the QUARC Stream API blocks. For detailed information refer to the QUARC help.

The Kobuki platform also provides the following I/O:

- 2 wheel encoder inputs
- 2 wheel speed outputs

- 2 digital LED outputs
- 4 digital power enable outputs
- 3 analog and digital cliff sensors
- 2 analog motor current inputs
- 3 digital bumper sensors
- 3 digital wheel drop sensors
- 3 digital buttons
- 18 IR dock sensor outputs
- 2 wheel over-current sensors
- 2 wheel PWM measurements
- 3-axis gyroscope
- 1 Z-axis angle measurement (heading)
- 1 battery voltage sensor

The analog, digital, robot, and general purpose I/O listed above are accessed using the QUARC HIL blockset. The serial, SPI, and I2C communication channels are accessed through the QUARC Stream blockset. For more information on accessing communication stream data see the QUARC help under *QUARC Target* | *User's Guide* | *Communications*. Table 2.2 lists the HIL blocks used to communicate with the QBot 2e hardware.

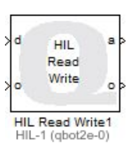
Block	Description
	The HIL Initialize block selects the hardware and configures the its parameters. The HIL Initialize block is named via the *Board name* parameter, and all other HIL blocks reference the corresponding HIL Initialize through its name. The HIL blocks will interface to the hardware specified in the HIL Initialize Board type parameter `qbot2`.
	The HIL Read Write block is used to read sensor measurements from the robot, the general purpose I/O, and write motor commands to the QBot 2e motors. The various inputs and outputs are specified with numeric channel numbers given in Table 2.3, Table 2.4, Table 2.5, and Table 2.6. For full details refer to the MATLAB help under *QUARC Targets* | *User's Guide* | *QUARC Data Acquisition Card Support* | *Quanser Cards* | *Quanser QBot 2e*.

Table 2.2: HIL blocks

Channel type	Write channel numbers	Description	Units
PWM	0 - 1	User PWM outputs	%
Digital	0 - 27	Reconfigurable digital I/O	
	28	LED1 red colour	
	29	LED1 green colour	
	30	LED2 red colour	
	31	LED2 green colour	
	32	Enable 3.3V power	
	33	Enable 5V power	
	34	Enable 12V/5A power	
	35	Enable 12V/1.5A power	
Other	2000	Right wheel speed	m/s
	2001	Left wheel speed	m/s
	14000	Custom sound	Hz
	16000	Predefined sound (0=on sound, 1=off sound, 2=recharge, 3=button, 4=error, 5=task start, 6=task end)	-

Table 2.3: QBot 2e output channels

Channel type	Read channel numbers	Description	Units
Analog	0	Supply voltage (battery)	V
	1	Right cliff sensor	V
	2	Central cliff sensor	V
	3	Left cliff sensor	V
	4	Right motor current	A
	5	Left motor current	A
Encoder	0	Right wheel (2578 counts per revolution)	counts
	1	Left wheel (2578 counts per revolution)	counts

Table 2.4: QBot 2e analog and encoder input channels

Channel type	Read channel numbers	Description	Units
Digital	0 - 27	User reconfigurable digital I/O	
	28	Right bumper	
	29	Central bumper	
	30	Left bumper	
	31	Right wheel drop	
	32	Left wheel drop	
	33	Right cliff detected	
	34	Central cliff detected	
	35	Left cliff detected	
	36	Button B0	
	37	Button B1	
	38	Button B2	
	39	Right overcurrent	
	40	Left overcurrent	
	41	Right dock IR near right sensor	
	42	Right dock IR near center sensor	
	43	Right dock IR near left sensor	
	44	Right dock IR far right sensor	
	45	Right dock IR far center sensor	
	46	Right dock IR far left sensor	
	47	Central dock IR near right sensor	
	48	Central dock IR near center sensor	
	49	Central dock IR near left sensor	
	50	Central dock IR far right sensor	
	51	Central dock IR far center sensor	
	52	Central dock IR far left sensor	
	53	Left dock IR near right sensor	
	54	Left dock IR near center sensor	
	55	Left dock IR near left sensor	
	56	Left dock IR far right sensor	
	57	Left dock IR far center sensor	
	58	Left dock IR far left sensor	

Table 2.5: QBot 2e digital input channels

Channel type	Read channel numbers	Description	Units
Other	12000	Timestamp	s
	16000	Charge state (0=discharging, 2=docking charged, 6=docking charging, 18=adapter charged, 22=adapter charging)	-
	11000	Right wheel PWM	%
	11001	Left wheel PWM	%
	1002	Z-axis angle	rad
	3000	X-axis angular velocity	rad/s
	3001	Y-axis angular velocity	rad/s
	3002	Z-axis angular velocity	rad/s

Table 2.6: QBot 2e other input channels

2.5.2 QBot 2e Kinect

The QBot 2e is integrated with a Microsoft Kinect sensor, which is capable of capturing RGB image data as well as 12-bit depth data at several resolutions and framerates. The Kinect sensor data is available through the QUARC Multimedia blockset under QUARC Targets/Multimedia in the Simulink Library Browser. Table 2.7 shows the various blocks used to interface with the Kinect and describes their purpose.

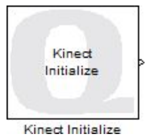
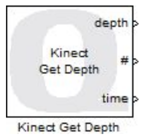
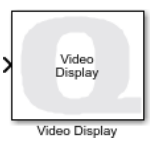
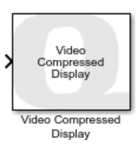
Block	Description
	The Kinect Initialize block is required to initialize the Kinect sensor and provide a name for the Kinect that other blocks use to associate with that sensor.
	The Kinect Get Depth block is used to output a depth-mapped image at the specified resolution. The Kinect can measure depth from 0.5 m to 6 m.
	The Kinect Get Image block is used to acquire RGB or greyscale images from the Kinect. The image sensor type can be specified as the color or infrared image sensor. The image resolution can be selected as either 640×480 or 1280×960. The image frame number is output from the block and can be used to trigger image processing when a new image is available.
	The Display Image block is used to display image data on the host computer. The Display Image block accepts greyscale and RGB image matrices as its input. The Display Image block acts like a scope and is not synchronized with the model; in other words, if the Display Image block cannot keep up with the performance of the model it will not block the model thread(s). Since the Display Image block requires the images to be transmitted between the target and the host it is recommended that the images used for display purposes be downsampled or sent at a slower rate.
	The Video Display block displays video in a window on the host. It is designed for typical video frame rates. The video can be paused and resumed, and the current frame may be saved to disk as an image. This block has much higher performance than the Display Image block because it does not use MATLAB figure windows. The Video Display block supports both RGB (HxWx3) and greyscale (HxW) inputs.
	The Video Compressed Display block displays video in a window on the host. However, it uses image compression internally to minimize the bandwidth required to transmit the raw image from the target model to the host. It is designed for typical video frame rates. The video can be paused and resumed, and the current frame may be saved to disk as an image. This block has much higher performance than the Display Image block because it does not use MATLAB figure windows. The Video Compressed Display block supports both RGB (HxWx3) and greyscale (HxW) inputs.

Table 2.7: Kinect blocks

3 Setup

3.1 Setting up the QBot 2e

Follow these steps to setting up the QBot 2e:

1. Connect the battery inside the Kobuki robot.
2. Press the power button shown in Figure 2.2c. This should turn on both vehicle and QBot 2e DAQ and Raspberry Pi 3 Model B+ computer.

3.2 Establishing Network Connection

The QBot 2e package comes with a pre-configured wireless router and automatically connects to the WiFi network Quanser_UVS-5G. It uses TCP/IP connection for communicating with the host computer and/or other Quanser unmanned vehicles. The Host PC and each of the vehicles must have unique IP addresses and the range of these addresses are defined below:

Host PC(s)	192.168.2.2 to 192.168.2.10
Reserved for DHCP Connections	192.168.2.11 to 192.168.2.19
Quanser QBot 2e (Raspberry Pi 3 Model B+)	192.168.2.20 to 192.168.2.254

Table 3.1: Valid Host PC and QBot 2e IP ranges

These Steps outlined below for setting up the host computer wireless connection only need to be performed once.

1. Power up and turn on the wireless router.
2. After turning on the router that is provided, wait for about 60 seconds for the wireless network to establish and allow the QBot 2e to automatically connect to the wireless router.
3. Connect your PC network card to any of the ports of the router (e.g. port number 1 to 4) using the network cable provided (you can also connect to the Quanser_UVS-5G wireless network if your computer has wireless adapters, however, wired connection between your PC and the router is preferred for better performance). If you choose wireless connectivity between your PC and the router, you should use the password UVS_wifi to connect to the wifi network.
4. Using the Windows Network system icon in the taskbar, open Network and Internet Settings.

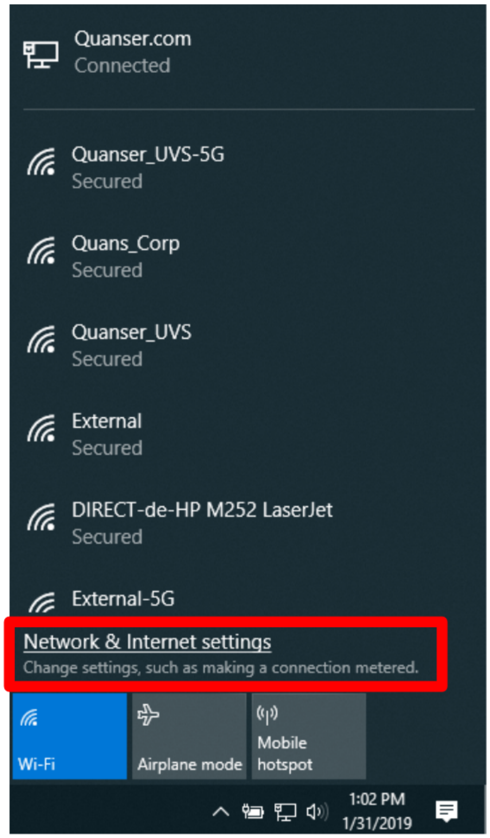


Figure 3.1: Open Network and Internet settings. If you are using a wired connection to the router and you don't have wireless adapter, you will not see Quanser_UVS and Quanser_UVS-5G networks.

5. Click on Change adapter options.
6. Right-click on the Ethernet icon for your connection to the router and click Properties. If you chose wireless connectivity between your PC and the router, right-click the Wi-Fi network connection instead, and right-click on it and select Properties.
7. Under This connection uses the following items: select Internet Protocol Version 4 (TCP/IPv4) and click Properties.
8. Select Use the following IP address: and enter the following settings for the Host PC: IP address: $192.168.2.10$ (alternatively you can use different a IP addresses within the valid range, see Table 3.1), Subnet mask: $255.255.255.0$, and Default gateway: $192.168.2.1$ and press the OK button.

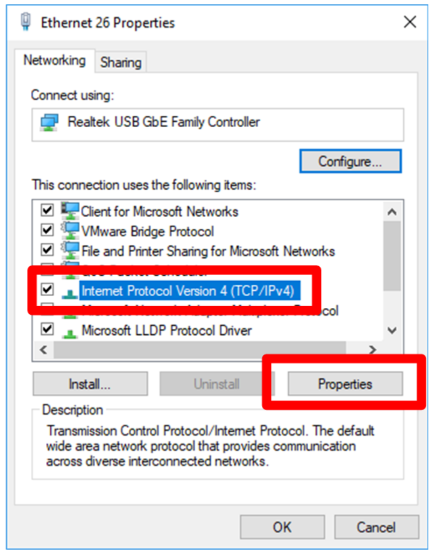


Figure 3.2: Network properties

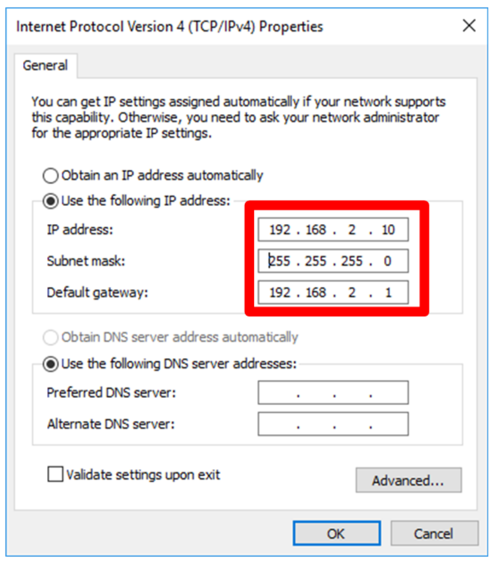


Figure 3.3: IP settings

9. Make sure you can ping the router by typing `ping 192.168.2.1` in the `Run` box in Windows (go to the *Start* menu and search for `Run` and click `Run`, Figure 3.4). If the connection to the router is successful you will see the ping replies in the command window. If you cannot ping the router, check network connectivity and your IP address before going to the next steps.

10. If the QBot 2e is powered on, the QBot 2e can be pinged by typing `ping {IP of the QBot 2e}` in the `Run` box in Windows (go to the *Start* menu and click `Run`, Figure 3.4). The QBot 2e IP address is labeled on the PCB as shown in Figure 3.5. If the connection is successful you will see the ping replies in the command window. If you cannot ping the robot, power cycle the QBot 2e and wait for about 60 seconds to reboot.

    Note: You may need to disable Windows firewall to establish a connection.

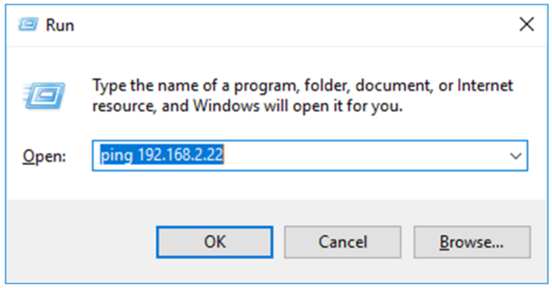


Figure 3.4: Pinging the QBot 2e

Figure 3.5: QBot 2e IP address label

3.3 Configuring Models for the QBot 2e Target

Note: This section applies only to files that are run on the Raspberry Pi 3 Model B+ target (i.e., on the QBot 2e). Simulink should have a new menu item called *QUARC* once QUARC® has been installed. The following steps are required to setup a new QUARC® model for the QBot 2e:

1. Create a new Simulink model, or open an existing model to be run on the Raspberry Pi 3 Model B+.
2. Click on the *QUARC* menu, then select Options.
3. The System target file under *Code Generation* should be `quarc_linux_pi_3.tlc`. Browse through the system target list to locate the proper file if necessary (Figure 3.6).

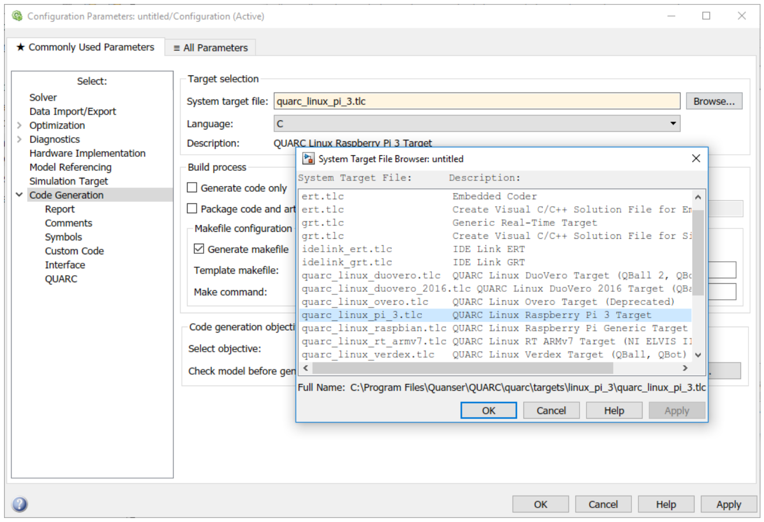


Figure 3.6: QUARC® Option menu

4. In order to run the QUARC® model on the target vehicle, the target's IP address must be specified. To setup the default target address for **all linux-raspberry-pi-3 targets**, go to the *QUARC* menu and select Preferences. The *Target type* parameter should be set to `linux_pi_3`. Replace the *Default Model URI* with the IP address of the desired target vehicle, e.g., `tcpip://{IP of QBot 2e}:17001?nagle=no,keep_alive=1`.

   Alternatively, to set the target address for the **current model only** open the model options under the *QUARC* | *Options* menu and choose *Code Generation* > Interface on the left hand pane. Under the MEX-file arguments, type `'-w -d /tmp -uri %u', 'tcpip://{IP of QBot 2e}:17001'`. Include the single quotation marks (Figure 3.7). Replace {IP of QBot 2e} with the IP of your QBot 2e, e.g. `'tcpip://192.168.2.20:17001'`.

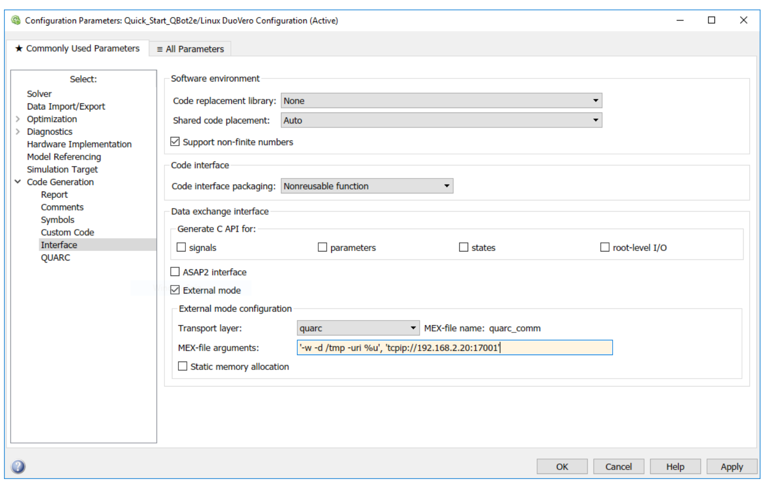


Figure 3.7: Model target IP settings

5. Select External for simulation mode, instead of Normal, which indicates that the model is to be run on the target machine (Raspberry Pi 3 Model B+) rather than simulating the model on the host machine.

6. The model is now ready to be compiled and downloaded to the target. If the wireless connection to the vehicle has been established, a QUARC® console can be opened to show additional messages and progress during model compilation by going to the menu item *QUARC* | Console for all. Building the model (*QUARC* | Build) will begin the code generation and compiling steps. Output from the compilation is shown in the QUARC® console. This step may take a few minutes to complete.

4 Troubleshooting

For any issue, the first and easiest troubleshooting solution on any electronic device is to reboot the device. Turn off the QBot 2e, then turn it back on again. For troubleshooting any problem with the QBot 2e, it is always a good idea to open the QUARC® console in case additional information is printed to the console by going to the QUARC® menu and clicking on Console for all.... The console must be opened after the QBot 2e has booted and established a WiFi connection. If the console is opened successfully it establishes a connection to the target and the console window has the title `QUARC Console for * at tcpip://192.168.2.xxx:17000`, where *xxx* corresponds to the IP address of the QBot 2e.

If you are still unable to resolve the issue after reading through this section, contact tech@quanser.com for further assistance.

Q1 You cannot ping the QBot 2e

Make sure the router is on and the WiFi light on the router is on. Check that the network adapter of the host PC is connected to the router (or the wireless network Quanser_UVS or Quanser_UVS-5G) and is configured according to the network configuration procedure outlined in this manual (Section 3.2). Verify that you can successfully ping the QBot 2e by going to the Windows *Start* | Run and typing `ping 192.168.2.xxx`, where *xxx* corresponds to the IP address of your QBot 2e.

□□□

Q2 The model fails to build/connect or the QUARC console does not successfully open

Turn on the QBot 2e and verify that the LED labeled PWR on the Raspberry Pi 3 Model B+ is lit and has turned solid red. This ensures that ensure power is being supplied to the Raspberry Pi. If the LED does not turn on, check the that the USB-A socket on the PCB is securely connected to the micro USB-B connector on the Raspberry Pi 3 Model B+ using the 0.3 m USB-A to micro USB-B cable as shown in Figure 2.6. Also, check that the network adapter of the host PC is connected to the router (or the wireless network Quanser_UVS or Quanser_UVS-5G) and is configured according to the network configuration procedure outlined in this manual (3.2). Verify that you can successfully ping the QBot 2e by going to the Windows *Start* | Run and typing `ping 192.168.2.xxx`, where *xxx* corresponds to the IP address of your QBot 2e.

□□□

Q3 The QBot 2e sensors are not being read correctly or they are stuck at some constant value

Using the HIL Read block, output all possible channels. Check these outputs using scopes and displays, and determine if the problem lies with a particular sensor, or set of sensors, or if the issue is global across all sensors.

□□□

Q4 The Simulink model appears to run slowly (i.e., the simulation time runs slower than actual time), or the console displays the message *Sampling rate is too fast for base rate*

(a) The maximum sample rate recommended for the QBot 2e is 1000 Hz (sample time 0.001 s). However, if there are complex calculations (such as image processing) performed within the model, then this could potentially limit the sample rate of the model (suggested sample rate of 100 Hz or 50 Hz). Try reducing the model sample rate in the menu *QUARC* | *Options* | *Solver* by increasing the *Fixed-step size (fundamental sample time)* parameter.

(b) To determine the execution time of blocks or subsystems within the model, use the Computation Time block found in the *QUARC library* | *Sources* | *Time*. This block outputs the computation time of a function call subsystem, measured using an independent high-resolution time source. Blocks can be placed inside a function call subsystem and connected to the *Computation Time* block to determine their execution time during each sample instant. This helps identify the bottlenecks in the model (blocks/subsystems with the highest execution time) and can identify blocks/subsystems whose computation time is greater than the sample time of the model. Try increasing the sample time of those blocks whose computation time is greater than the sample time of the model so that the blocks run in a slower rate thread.

(c) If you are using image processing blocks, ensure that they are only executed when there is new image data available or at a slower rate than the image acquisition rate. Otherwise, the embedded computer is spending time processing image data that has already been processed, which can cause the model's computation time to increase undesirably. For complex image processing, consider lowering the rate of the image processing blocks and/or downsampling the source images to a lower resolution to speed up processing time.

(d) If you are using image processing blocks, ensure that signal duration is set to 2 by going to Tools | External Mode Control Panel | Signal & Triggering | Duration menu on the model (The default value is 10000).

□ □ □

Q5 Trying to start the QBot 2e model results in the error *Unable to locate the dynamic link library or shared object*

This error indicates that the QBot 2e driver is not found on the target. Make sure that the model target type is set to quarc_linux_pi_3 by navigating to the QUARC® menu *QUARC* | *Options* | *Code Generation* pane and changing the System target file to `quarc_linux_duovero_2016.tlc`. Open a console through the QUARC® menu *QUARC*| *Console for all*, and verify that the console window displays the target IP of your QBot 2e in the window title.

□ □ □

Q6 Building a model fails when the error *Not enough system resources are available to perform the operation*

When several models are compiled, the disk space on the Raspberry Pi may become full, and you will no longer have space to build models. Using the clean option in the QUARC® menu under *QUARC* | *Clean all* will remove all generated code and compiled code for the current model, but this will only free up the space used by the current model. To view all models currently downloaded on the target select Manage target under the QUARC® menu. The current model's target must be powered on and ready to accept a connection. The target information is displayed including all models that have been downloaded to the target. To clear all downloaded models select all the models in the list and click Remove. Note: this will only remove generated code from the target and will not delete the source models on the host PC.

□ □ □

Q7 MATLAB returns an error 'Model failed to download on target... The code being downloaded or run is not compatible with the type of target referenced by the target' when building the model.

This error is returned by MATLAB when there is a mismatch between the 'Target type' referenced in 'QUARC | Preferences' and the actual target type. Ensure that 'Target type' is set to *linux_pi_3*.

□ □ □

5 Technical Support

To obtain support from Quanser, go to http://www.quanser.com/ and click on the Tech Support link. Fill in the form with all the requested software and hardware information as well as a description of the problem encountered. Also, make sure your e-mail address and telephone number are included. Submit the form and a technical support person will contact you.

Quanser Inc.
119 Spy Court
Markham, Ontario
L3R 5H6
Canada
info@quanser.com
Phone: 1-905-940-3575
Fax: 1-905-940-3576

Printed in Markham, Ontario.

For more information on the solutions Quanser Inc. offers, please visit the web site at:
http://www.quanser.com