

QBot Platform

User Manual – Software: Simulink

v 1.1 – 1st Oct 2024

Quanser Consulting Inc. info@quanser.com
119 Spy Court Phone : 19059403575
Markham, Ontario Fax : 19059403576
L3R 5H6, Canada printed in Markham, Ontario.

This document and the software described in it are provided subject to a license agreement. Neither the software nor this document may be used or copied except as specified under the terms of that license agreement. Quanser Consulting Inc. ("Quanser") grants the following rights: a) The right to reproduce the work, to incorporate the work into one or more collections, and to reproduce the work as incorporated in the collections, b) to create and reproduce adaptations provided reasonable steps are taken to clearly identify the changes that were made to the original work, c) to distribute and publicly perform the work including as incorporated in collections, and d) to distribute and publicly perform adaptations. The above rights may be exercised in all media and formats whether now known or hereafter devised. These rights are granted subject to and limited by the following restrictions: a) You may not exercise any of the rights granted to You in above in any manner that is primarily intended for or directed toward commercial advantage or private monetary compensation, and b) You must keep intact all copyright notices for the Work and provide the name Quanser for attribution. These restrictions may not be waived without express prior written permission of Quanser.



Caution

This equipment is designed to be used for educational and research purposes and is not intended for use by the public. The user is responsible for ensuring that the equipment will be used by technically qualified personnel only. **NOTE:** While the GPIO, Ethernet and USB ports provide connections for external user devices, users are responsible for certifying any modifications or additions they make to the default configuration.

FCC Notice This device complies with Part 15 of the FCC rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

Note: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

Contains FCC ID: SQG-60SIPT

Industry Canada Notice This Class A digital apparatus complies with CAN ICES-3 (A). Cet appareil numérique de la classe A est conforme à la norme NMB-3 (A) du Canada.

Contains IC: ST60-2230C-PU

Waste Electrical and Electronic Equipment (WEEE)



This symbol indicates that waste products must be disposed of separately from municipal household waste, according to Directive 2012/19/EU of the European Parliament and the Council on waste electrical and electronic equipment (WEEE). All products at the end of their life cycle must be sent to a WEEE collection and recycling center. Proper WEEE disposal reduces the environmental impact and the risk to human health due to potentially hazardous substances used in such equipment. Your cooperation in proper WEEE disposal will contribute to the effective usage of natural resources.

CE Compliance 

This product meets the essential requirements of applicable European Directives as follows:

- 2014/35/EU; Low-Voltage Directive (safety)
- 2014/30/EU; Electromagnetic Compatibility Directive (EMC)
- 2014/53/EU; Radio Equipment Directive (RED)

Warning: This is a Class A product. In a domestic environment this product may cause radio interference, in which case the user may be required to take adequate measures.

The Leishen LiDAR M10P and the Intel RealSense D435 RGB-D camera are both classified as Class 1 Laser Products. The laser safety of both products meets the following standards:

- IEC 60825-1:2014
- 21 CFR 1040.10 and 1040.11 standards, except for conformance with IEC 60825-1 Ed. 3 as described in Laser Notice No. 56, dated May 8, 2019.



Caution

Do not power on the product if any external damage is observed. Do not open or modify any portion of any laser product as it may cause the emissions to exceed Class 1. Invisible laser radiation when opened. Do not look directly at the transmitting laser through optical instruments such as a magnifying glass or microscope. Do not update laser product firmware unless instructed by Quanser.



Caution

ESD Warning. The QBot Platform internal components are sensitive to electrostatic discharge. Before handling the QBot Platform, ensure that you have been properly grounded.

Table of Contents

A. Overview	3
B. Model Configuration Settings	4
C. Multiple sampling rates	5
D. Driver Model	5
Send packet message	6
Receive packet message	6
Built in Driver LEDs	6
E. Code generation, deployment, and monitoring	7
F. QUARC Monitor and Console	9
G. Upgrading QUARC	10
H. Basic Troubleshooting	10

A. Overview

MATLAB/Simulink is used as an open canvas to design your applications as you see fit. It is common to setup the Simulink model for real-time deployment (section B) prior to developing the code (Section C, D). Once ready, you will build a real-time application from the Simulink Code, download it and deploy it on the QBot Platform (Section E, F).

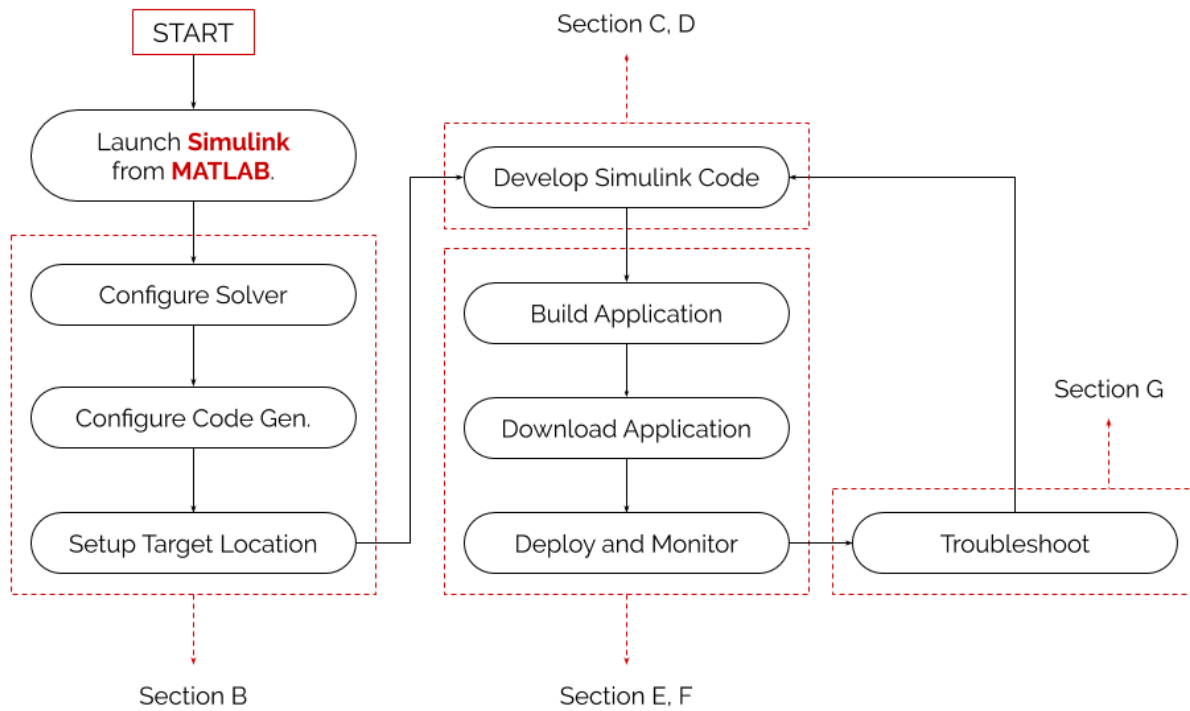



Figure 1. Process diagram for MATLAB/Simulink code deployment

B. Model Configuration Settings

1. Once you open **Simulink**, open the **Hardware Settings** using the icon  or pressing **Ctrl+E**.
2. Under **Solver > Solver selection**, set the **Type** to **Fixed-step** and set the **Solver** to the desired one. If you are unsure, select **ode1 (Euler)**.
3. Under **Solver > Solver details**, set the **Fixed-step size (fundamental sample time)** to the desired value. For most QBot Platform user applications talking to a QBot Platform Driver, select 60 Hz, set the value to **1/60**.
4. Under **Code Generation > Target Selection**, set **System target file** as applicable to your deployed application, for example:
 - a. For applications running on the development machine, use **quarc_win64.tlc** for a windows-based platform. This may be used for real-time simulations, robust communications proxies to QBot Platforms, or infrastructure servers to coordinate the actions of multiple QBot Platforms.
 - b. For applications to be deployed to a QBot Platform target, select the **quarc_linux_qbot_platform.tlc** for the QBot Platform. This will be the typical target for most of the Simulink examples and courseware provided.
5. If the target is the QBot Platform, you must also specify the location of the target on the network so that the application can be downloaded there automatically by QUARC. To do this, navigate to **Code Generation > Interface**, and edit the **MEX-file arguments** by adding the following code:

```
'-w -d /tmp -uri %u', 'tcpip://IP_ADDRESS:PORT'
```

Do not exclude the comma or the single quotations. Here, **IP_ADDRESS** refers to the IPv4 address of the QBot Platform shown on the QBot Platform's LCD screen and **PORT** refers to a number between 17001 and 17999. Do not use 17099 as it is reserved for the driver application (see below). Simulink will use the port specified to communicate with the deployed application and display any data connected to Simulink sinks in your code on your local machine itself. For example, if the **IP_ADDRESS** of the QBot Platform is **192.168.2.12** and we are targeting the model to port **17001**, then the MEX-file arguments string will be:

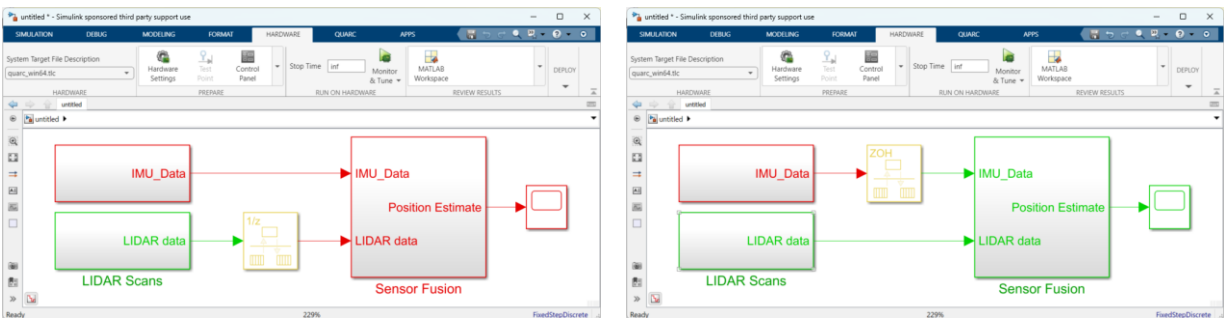
```
'-w -d /tmp -uri %u', 'tcpip://192.168.2.12:17001'
```

Typically port 17001 will always be used, but if you are running multiple Simulink models on the QBot Platforms simultaneously, then you will need to specify a separate port for each simultaneous model you run.

C. Multiple sampling rates

The **Fixed-step size (fundamental sample time)** parameter in the **Model Configuration Settings** corresponds to the fastest sample time in your model, denoted as D1 in Figure 2. This also corresponds to the **continuous** sample time. The current value of this parameter can be accessed in the model by using the constant `qc_get_step_size`.

Most Simulink/QUARC blocks have a **Sample Time** parameter. When set to -1, this makes the block automatically acquire its sample time based on other blocks that are connected to it. However, it is good practice to explicitly specify the sample time, which makes it easier to read the diagram for future developers and debugging. Integer multiples of the time step constant `qc_get_step_size` can be used to specify other sample rates.



a. Sensor Fusion at 500Hz

b. Sensor Fusion at 10Hz

Figure 2. Using Rate Transition to combine signals at various rates

Tip: Under **DEBUG > Information Overlays**, click **Colors** to turn on rate colors

Consider the example shown in Figure 2 where the **LIDAR Scans** subsystem provides data at 10Hz and the **IMU Acquisition** subsystem provides data at 500Hz. The subsystem labelled **Sensor Fusion** needs to use both the signals to estimate position. To do so without Simulink errors, you must use a **Rate Transition** block on the signal depending on the rate at which you expect sensor fusion to run. For example, if **Sensor Fusion** must be executed at 500Hz, insert a **Rate Transition** on the lidar signal line with sample time set to -1 as in Figure 2a. To run **Sensor Fusion** at 10Hz, insert a **Rate Transition** block on the IMU signal line, with sample time set to -1 as shown in Figure 2b.

D. Driver Model

Along with the libraries, the QBot Platform also comes with a driver model. The driver model has been created for safety purposes with driving the QBot Platform. We strongly encourage all user applications to go through the driver model for safety purposes. There are several different inputs that get sent to driver model.

Send packet message

The message format to the drive application includes the following data structure. Use a stream client at 60Hz connected to the server running on the following URL: `tcpip://IP_ADDRESS_OF_QBOT:18888`. The packet structure contains 10 doubles:

Function	Doubles (float64)	Details
Mode	1	0 or 1 (teaching mode) 2 or 3 (research mode) (see Vel Cmd)
Enable User LED	1	Enable color override
User LED color	3	Color value
Arm	1	Arm the motors
Hold	1	Not implemented yet
Vel Cmd	2	Forward/turn speed (mode 0 or 2) (m/s, rad/s) left/right wheel speeds (mode 1 or 3) (rad/s, rad/s)
Timestamp	1	Timestamp signal for loopback

Receive packet message

The QBot Platform driver also returns a 17 double data packet with useful information. This is summarized below. Use the same stream client to receive this data at 60Hz.

Function	Doubles (float64)	Details
Wheel position	2	Wheel encoder positions (in rads)
Wheel speeds	2	Wheel tachometer speeds (in rad/s)
Cmd Voltage	2	Voltage commanded by onboard controllers (in Volts)
Accelerometer	3	Tri-axis Accelerometer data (m/s/s)
Gyroscope	3	Tri-axis Gyroscope data (rad/s)
Current	2	Left/right motor current draw from the battery (Amps)
Battery level	1	Active battery level (Volts)
Watchdog status	1	Status information regarding watchdog expiry. If expired, re-arm the QBot Platform to resume function.
Timestamp	1	Loopback timestamp signal returned by the driver

Built in Driver LEDs

The driver and QBot Platform also have some LED colors built in to let you know when the QBot Platform is in different states.

Running the driver model will initially change the QBot Platform lights to pulse on and off white. This shows that your driver model is running, and you can now run your own application to connect to the driver model. If you connect a model to the robot, the lights will turn blue. This shows that the driver application is connected to a client and waiting for commands (but the robot is not yet armed). Arming the robot in this state will cause the lights to turn green. Setting the hold command to 1 in this state will cause the lights to flash warning you that this feature is not yet implemented in the driver.

You can override the blue, green or white pulse colors with your own input by setting the color variable to a RGB color of choice and setting the Enable User LED command to 1.

If your battery is too low, the lights will pulse purple to let you know you need to charge your robot. If your lights pulse yellow, it means you've caused a stall or overcurrent condition. Simply disarm and rearm the robot to continue.

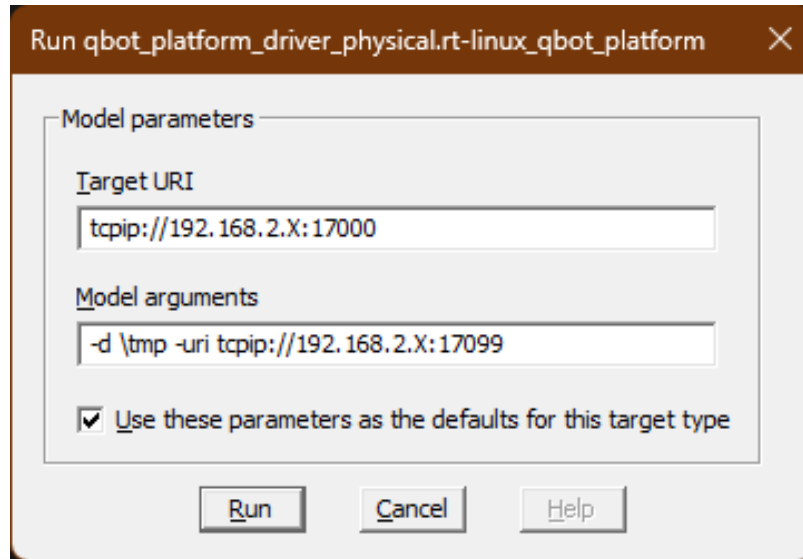
Finally, if your robots' lights are red or flashing red, these states are set by the firmware on the robot. Solid red lights show the robot is on, or the application has stopped normally without any error conditions. Red flashing lights show that the embedded computer has shut down and the robot must be power cycled to work again.

Color	State	Level	Description
White	Pulse	Model	Driver application is healthy and waiting for an incoming client connection
Blue	Solid	Model	Driver application is connected to a client and waiting for commands (the robot is not armed).
Green	Solid	Model	Driver application has armed the robot and motor controllers are active.
Green	Pulse	Model	Driver application has armed the robot and has received the hold command (not implemented).
Other	N/A	Model	Driver application is applying a user LED color.
Yellow	Pulse	Model	Motor overcurrent or stall being detected (this is not an error but indicates strain)
Magenta	Pulse	Model	Low battery warning (this is not an error but indicates that you should stop the model soon and change the batteries).
Red	Solid	Firmware	Driver application was stopped normally without any error conditions, or no application has been deployed yet.
Red	Pulse	Firmware	Embedded Computer has shut down and the robot must be turned OFF using the power switch.

E. Code generation, deployment, and monitoring

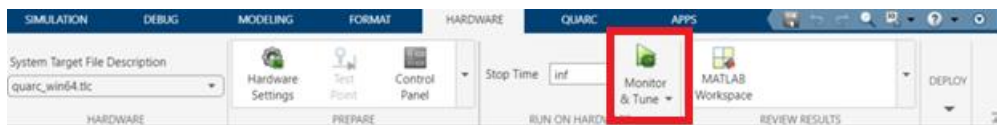
Once you are done with code development, you must generate C code before deploying it to the target. Follow these steps:

First run the `qbot_platform_driver_physical.rt-linux_qbot_platform` file provided to you. To run the driver model on the physical hardware, right-click on the driver file in your explorer outside MATLAB, click on 'show more options', followed by 'Run on target'. Use the following settings and hit Run.




For MATLAB r2023a and newer:


1. In the Simulink ribbon under the **HARDWARE** tab, click the **Monitor and Tune** button to start the compilation, download and model execution process:



Once your code is running, you will see an increasing timestamp in your model (bottom right).

2. **Stop your model:** When ready to stop execution, click on the stop icon  or press **Ctrl+Shift+W**.

For MATLAB r2022a and older:

1. **Build and download your code:** click build icon  or pressing **Ctrl+B**. Open the **Diagnostic Viewer** (View menu) to view the progress. In case you are deploying the code to an external target, this also downloads the code.
2. **Connect to the target:** click on the connect icon  or press **Ctrl+Shift+O**.
3. **Start your model:** click on the start icon  or press **Ctrl+Shift+Q**. Your model should start running and the simulation time should advance (bottom right corner of the model).
4. **Stop your model:** When ready to stop execution, click on the stop icon  or press **Ctrl+Shift+W**.

F. QUARC Monitor and Console

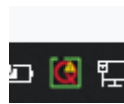
From the **Start** menu, find and launch the **QUARC Monitor** app. It will appear on the bottom-right corner of your screen as a tray icon as shown in Figure 4a. Right-click this icon and set the **Target** to **Remote** or **Local Windows** depending on where your QUARC application is being deployed. For the **Remote** case, set the **Target URI** to the following,

```
tcpip://IP_ADDRESS:17000
```

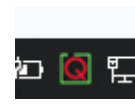
where IP_ADDRESS corresponds to the IPv4 address of the QBot Platform (e.g., 192.168.2.11). When the monitor searches for the target, a yellow warning symbol is also displayed as shown in Figure 4b. Once connected, the warning sign disappears as in Figure 4c. Right-click the icon now and click on **Console....** This opens a QUARC console that shows additional information to validate the model being downloaded, connected to and started (Figure 5).



a. QUARC monitor tray icon



b. connecting to remote target



c. connected to remote target

Figure 4. QUARC Monitor tray icon and connecting to target

```
QUARC Console for * at tcpip://192.168.2.18:17000
---- model 'test' downloaded ----
---- model 'test' loading ----
---- model 'test' loaded ----
Entered main(argc=6, argv=0x7fef1f2c38)
  argv[0] = test
  argv[1] = -w
  argv[2] = -d
  argv[3] = /tmp
  argv[4] = -uri
  argv[5] = tcpip://192.168.2.18:17001
Waiting for start packet from host.
** starting the model **
Creating a multithreaded model
Creating subrate thread 1 with priority 20...
Creating main thread with priority 21 and period 0.002...
Main thread exited
Subrate thread 1 returned exit_code: 0xffffffffffff
Invoking model termination function...
Exiting real-time code
---- model 'test' terminated (exit code 0) ----
```

connected {

started {

stopped {

application downloaded

Figure 5. QUARC Console for additional debugging information

G. Upgrading QUARC

Quanser regularly updates QUARC with new features. Many of these features only require that you update your development PC with the latest version of QUARC. To upgrade QUARC on the QBot Platform, connect to it remotely or directly as indicated in the Connectivity User Manual, and run the following commands in terminal on the QBot,

```
nvidia@qbp-XXXXX:~$ sudo apt update
nvidia@qbp-XXXXX:~$ sudo apt upgrade
```

Note: The password for sudo operations is nvidia. A valid internet connection is required. Consider connecting the QBot Platform to a Wi-Fi network with internet connectivity or use a LAN wired connection directly.

H. Basic Troubleshooting

This non-exhaustive list consists of common errors when building code or connecting to and starting applications.

Note: If your software error is not present here, please contact tech@quanser.com.

1. The 'Build' step completes successfully but the MATLAB Command Window displays

```
'??? Model ***model_name*** cannot be downloaded to target
'tcpip://IP_ADDRESS:17000?keep_alive=1'. It was not possible to connect to the specified URI.'
```

This indicates that even though the build succeeded, and the application is ready, the 'download' step has failed due to QUARC not being able to find the target.

- a. Check your connection to the target - both your local machine and the target should be on the same network. For more details, see the [User Guide - Connectivity](#).
- b. Check the **IP_ADDRESS** of your target in the **MEX-file arguments**
- c. Check the Windows Firewall setting to ensure the requested port numbers in the model URI and in the QUARC Stream API blocks such as Stream Server and Stream Client (e.g., 17000-17020 and 18000-18999) are allowed to pass through (for both incoming and outgoing TCP/IP and UDP traffic).

Once your connection to the target resumes, download the application already built by clicking on **Deploy** in the **QUARC** tab in Simulink.

2. The 'Connect' step returns the following error in the Diagnostic Viewer

```
'Error occurred while executing External Mode MEX-file 'quarc_comm':
Unable to establish connection with QUARC Target manager for external mode communications. It was
not possible to connect to the specified URI. Verify that the target is serving on URI
tcpip://IP_ADDRESS:17000?keep_alive=1'. Use the QUARC Console for debugging.'
```

This mirrors issue (1) described above. Communication with the **Quarc Target Manager** application could not be established because the target could not be found. Follow the instructions given in the troubleshooting steps for issue (1) to

establish the connection.

3. **The 'Connect' step returns the following error in the Diagnostic Viewer**

'Error occurred while executing External Mode MEX-file 'quarc_comm':
Unable to establish connection with real-time model for external mode communications. The remote peer refused the connection, most likely because no server application was listening for connections. Verify that your real-time model is serving on URI 'tcpip://IP_ADDRESS:PORT'.

The connection between your development machine and the target was successful, however, the built application was not downloaded successfully and hence cannot be found. Click on **Download** in the **QUARC** menu to download the built application to your target and **Connect** again.

Note: It is common to get this error after the error in 1.b. If the connection to the target wasn't established when you used Build to build your application, it will fail to download the model to the target. After resuming connection, trying to Connect directly will also fail, resulting in this error. Try downloading the model again should resolve this issue.

4. **Error occurred while executing External Mode MEX-file 'quarc_comm': The card specific option specified is not recognized.**

This sometimes occurs when models are moved between MATLAB versions. Find the HIL Initialize block in the Essentials subsystem and double click on it to open the dialog. Confirm that QBot Platform is selected then click on the Defaults button at the bottom of the dialog to restore the default board specific options.

© Quanser Inc. All rights reserved.



Solutions for teaching and research. Made in Canada.