How to use the STSW-L99615C PC software GUI for L9961 based evaluation board

Introduction

This document will cover the various aspects of the L9961 PC software GUI (Graphical User Interface) and the features that can be used to interact with the STEVAL-L99615C BMS evaluation board.

The L9961 GUI utilizes the PC serial communication port to interface with the STM32 microcontroller. The STM32 then communicates directly with the L9961 via I^2C.

For technical details of the L9961 BMS device, the L9961 datasheet should be referenced.

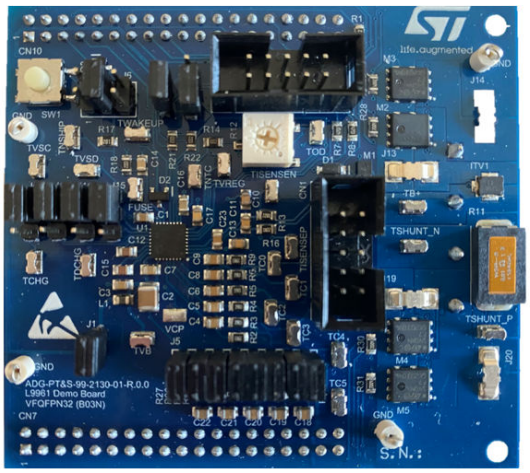
1 STSW-L99615C software package

1.1 Package content

The STEVAL-L99615C software package includes the L9961 GUI .exe installer, the application sample scripts to enable key parameters such as cell voltage, current and battery temperature acquisitions, and the binary code to flash the NUCLEO-G071RB STM32 Nucleo-64 development board. This board is used to control the L9961 device via I2C and GPIO and also to interface to the PC via the USB connection.

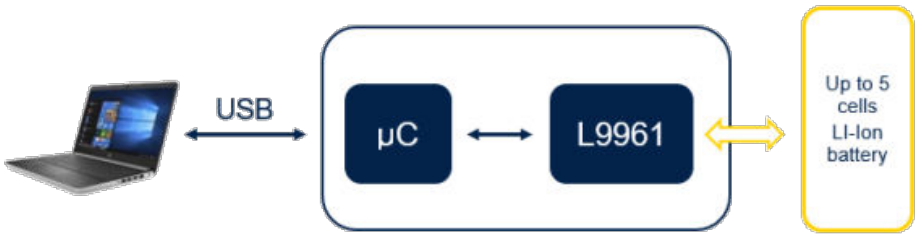
The software GUI exploits the features of the L9961 BMS IC, through the STEVAL-L99615C evaluation board.

Figure 1. STEVAL-L99615C demo board



Note: Please refer to the UM2951 on how to flash the NUCLEO-G071RB MCU board.

Figure 2. Demo board block diagram

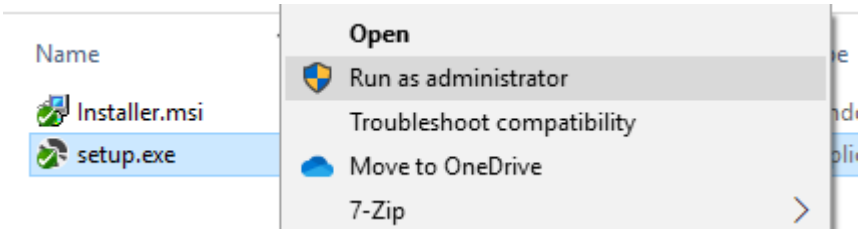


2 GUI installation

The GUI software comes packaged as a standard Windows installer. The program must be installed as administrator. Therefore, the user must have administrative rights in order to properly install this application. This can be verified by your local IT department.

To initiate installation as administrator, navigate to the setup.exe file, right-click, and select "Run as administrator" (see the Figure 3).

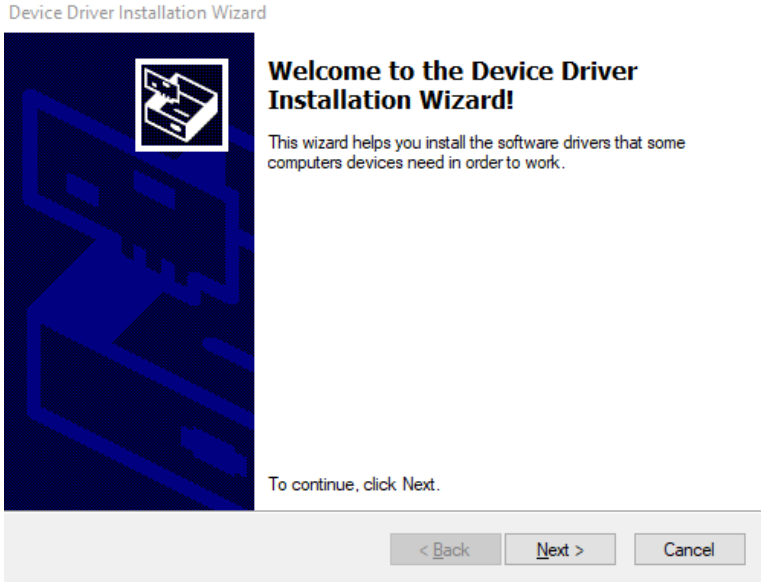
Figure 3. Run GUI installation as administrator



The GUI installer will then go through a series of dialog boxes. Unless there is a need to change one of the default settings, the default values can be used.

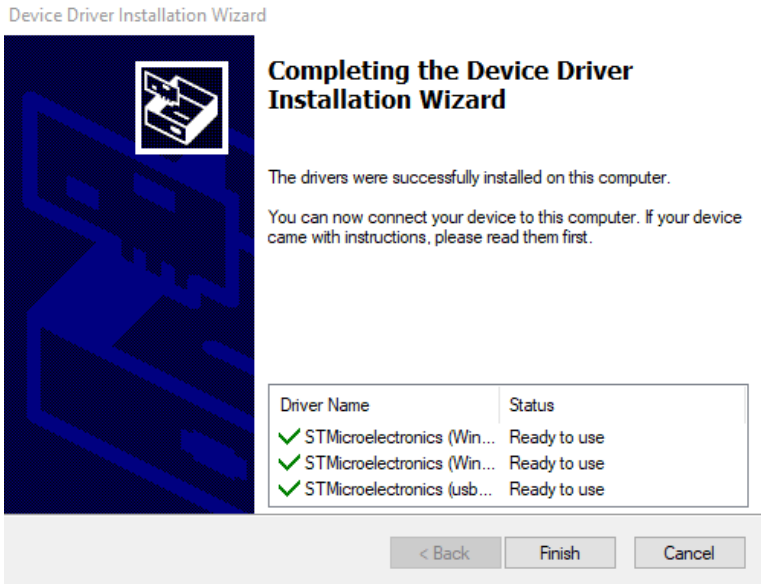
Following the installation of the GUI, the installer for the necessary device drivers will be launched, and the following window will appear (see the Figure 4):

Figure 4. Device driver installation wizard



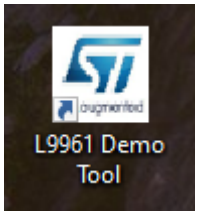
As with the GUI installer, select the default values to complete the installation, and then press the "Finish" button (see the Figure 5):

Figure 5. Successful installation of mandatory device drivers



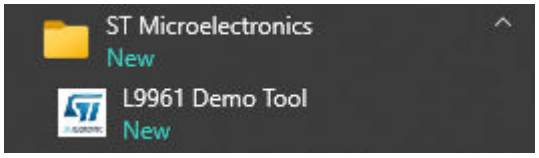
Once installation is complete, an L9961 Demo Tool shortcut icon will appear on the desktop (see the Figure 6):

Figure 6. L9961 GUI desktop shortcut



Additionally, the GUI can be located using the Windows start menu (see the Figure 7):

Figure 7. L9961 GUI Windows start menu location



3 GUI operation

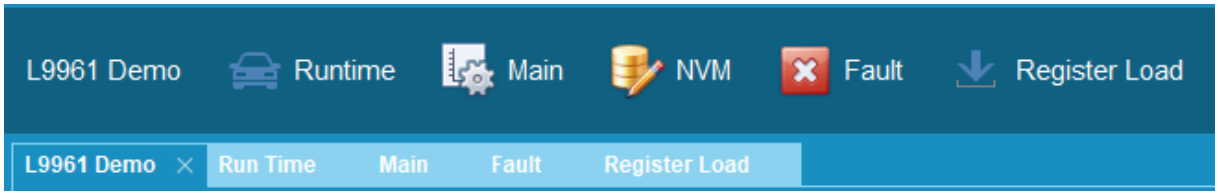
3.1 Overview

Figure 8. **L9961 evaluation GUI**

The L9961 GUI consists of six tabs that can be selected at the top left of the GUI (see the Figure 9). These tabs will be reviewed in more detail later in this document. The tabs are as follows:

- L9961 Demo
- Runtime
- Main
- NVM
- Fault
- Register Load

Figure 9. **L9961 GUI tab selections**



The tabs can be arranged in any order the user would like by grabbing a tab (left-click and hold) and dragging to the desired window location (see the Figure 10). It is possible to display all six tabs at one time (see the Figure 11).

Figure 10. Dragging a tab to the top window location

Figure 11. All six tabs displayed at once

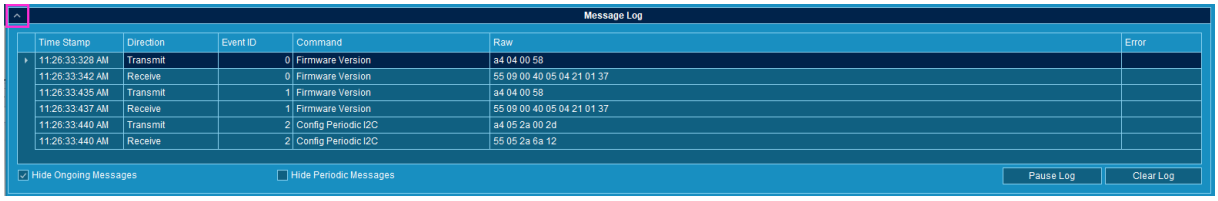
There are three sections of the GUI that can remain visible across all six tabs (highlighted above in the Figure 8):

- Message log - can be expanded/collapsed using the carrot.
- Board connection status - always visible.
- I^2C Read/Write array - can be expanded/collapsed using the carrot.

3.1.1 Message log

The message log provides a history of the command communication between the STM32G0 MCU and the PC. It can be expanded or collapsed by pressing the carrot shown in the Figure 12:

Figure 12. L9961 GUI message log



The fields in the message log are as follows:

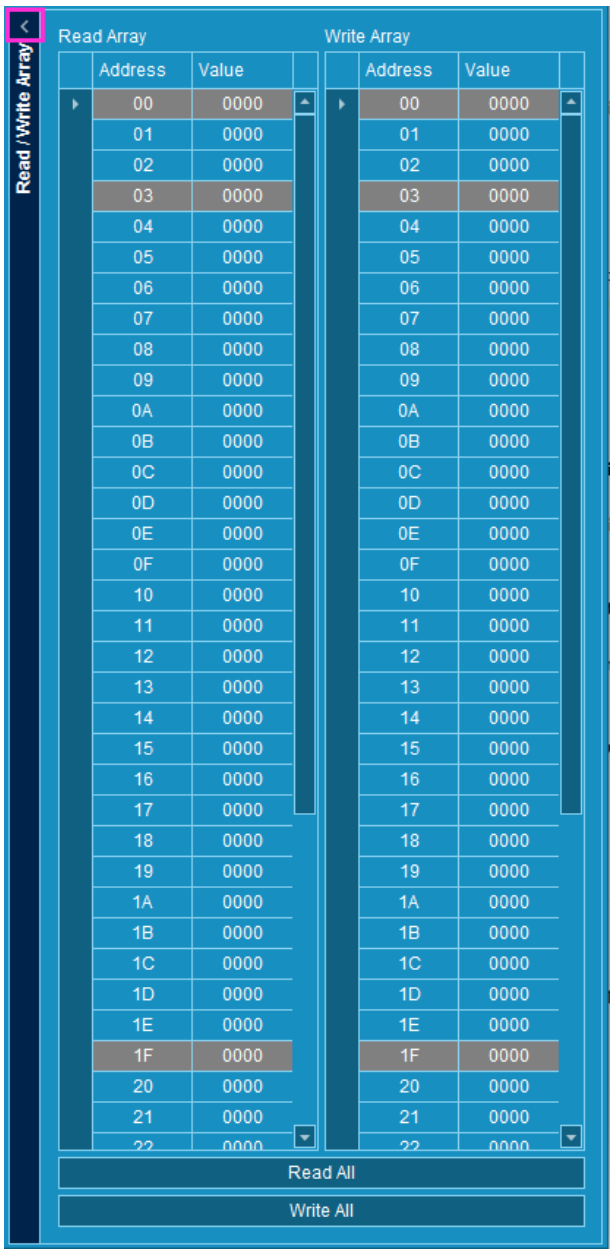
- Time Stamp - The time that the transaction occurred
- Direction - Whether the message was transmitted or received
- Event ID - An incremental number indicating the transaction number. For each transmit ID, there should be a matching receive ID
- Command - The command that was sent or is being responded to
- Raw - The raw communication between the STM32/L9961 and the PC

Once the message log is expanded, the options of pausing and clearing of the log are available. Additionally, the ability to enable/disable the logging of ongoing or periodic messages is available.

3.1.2 I²C Read/Write array log

Similarly, the I²C read/write array log can be collapsed or expanded by pressing the carrot shown in the Figure 13:

Figure 13. I²C read/write array log



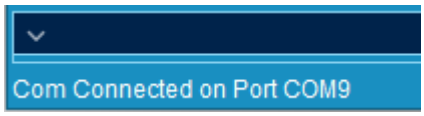
The user can also send read/write commands to all of the registers in the array log by pressing "Read All" or "Write All". The write data (value) field can be edited by double-clicking the text field (see the Figure 14):

Figure 14. Editing the data field of register 0x0D in the Write array

3.2 Connecting to the evaluation board

As soon as the GUI is opened, the PC will attempt to connect to the STM32 Nucleo board. If a valid COM port is found, and a connection is made successfully, the lower left corner of the GUI will show that the connection has been established (see the Figure 15):

Figure 15. GUI connected to evaluation board successfully



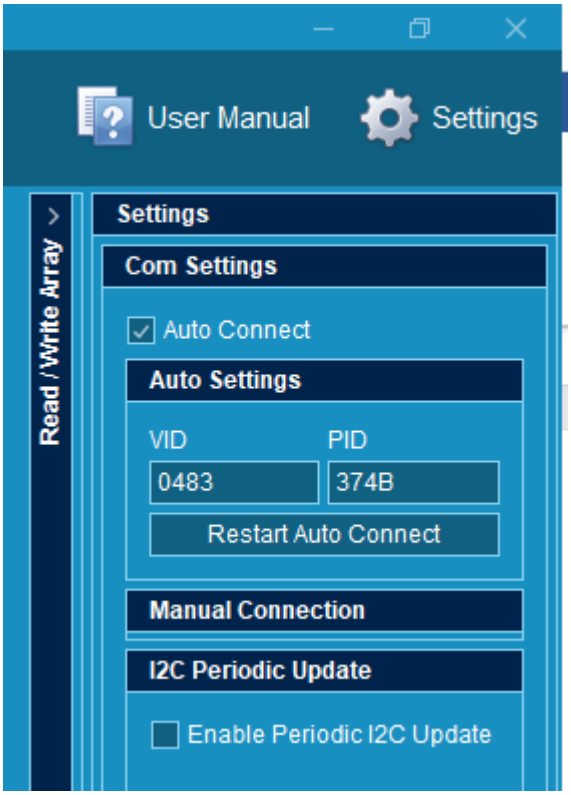
If a connection between the PC and the evaluation board is not established, the GUI will actively look for the device and display the following message (see the Figure 16):

Figure 16. COM failure

In the event that the GUI cannot detect the board (and it has been confirmed it is powered on, plugged in, and all device drivers are installed), the user can adjust the COM setting manually.

Navigate to the Settings menu, located in the upper right corner of the GUI (see the Figure 17):

Figure 17. L9961 GUI Settings window

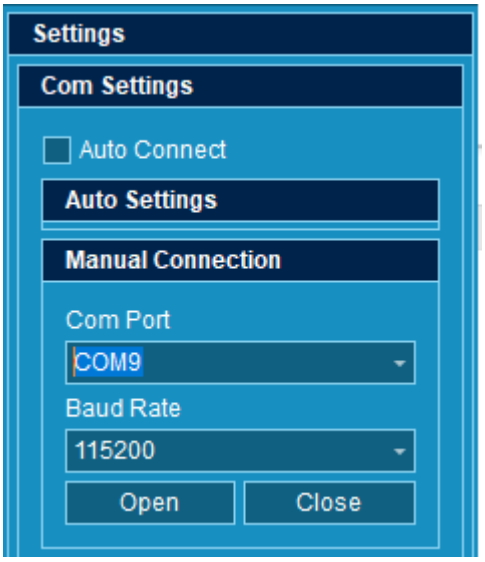


When the Auto Connect checkbox is enabled, the application will be able to automatically detect when a unit is plugged in or removed. If the Auto Connect is not enabled, it will be necessary to press the Restart Auto Connect button.

The VID/PID values are the default values for the USB to serial interface on the STM32/L9961 evaluation board. There should be no need to change these values, but the option is available should the need arise.

If reconfiguration of the VID/PID values does not fix the communication problem, the port can be manually selected by turning off the auto connect feature (see the Figure 18).

Figure 18. Manual COM port selection



3.3 GUI tab overview

3.3.1 L9961 Demo tab

When the GUI is launched, this tab will be displayed by default (see the Figure 19):

Figure 19. L9961 Demo tab display

The L9961 Demo tab is used to work with the device in an easy to visualize way. All aspects of the device are laid out in this tab which consists of seven sections:

1. Script Selection
2. Voltage Conversion and Cell Balancing
3. Current Conversion

4. HS/LS Pre-Drivers
5. Status Outputs
6. Die Temp and NTC
7. Device State

3.3.1.1 *Script Selection*

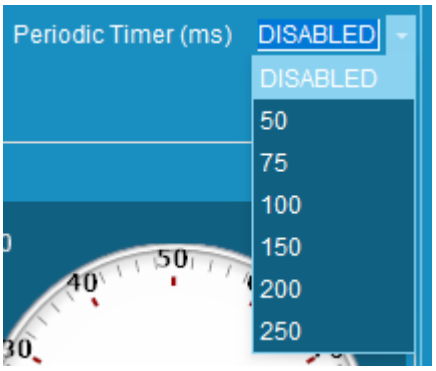
Figure 20. Script Select drop-down box

The L9961 GUI is provided with a set of scripts that can be used to get the user up and running quickly. However, if the user wishes to create their own script, then Section 3.3.6 Register Load tab should be referenced.

To access a script, press the belonging to the "Script Select" box (see the Figure 20), and click on the desired script.

Next, change the Periodic Timer (see the Figure 21) to "200 ms" for any of the sample scripts. This can be less for custom scripts depending on the timings selected.

Figure 21. L9961 demo tab periodic update rate selection



Following the selection of the periodic update rate, the selected script will run. Once the script has completed, the components on the tab will begin to update per the script's instructions.

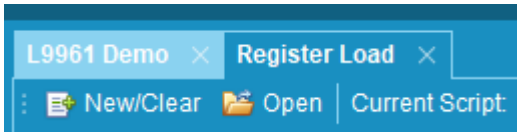
Example:

If the user selects the script titled "Voltage Acquisition Init - 5Cell + VB + NTC.csv" and selects a periodic update of 200 ms, the components for cells, Vbat, Vsum, NTC, and Die Temperature (along with their associated fault LEDs) will all display their respective data at a 200 ms update rate (see the UM2951 for the board configuration).

Note: In order to have the scripts appear in the "Script Select" drop-down box, the file extension must be configured properly in the Register Load tab. This only needs to be done for first-time users - the settings will be retained for future.

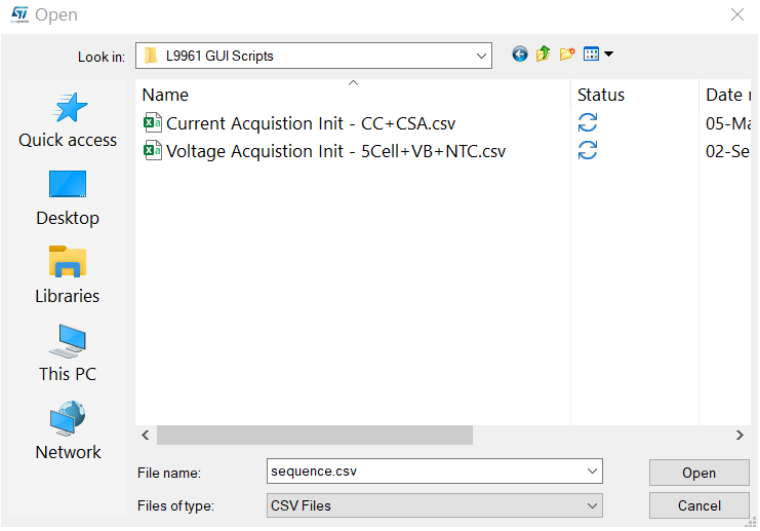
To do this, navigate to the Register Load tab and click "Open" (see the Figure 22).

Figure 22. Register Load file open



Next, navigate to the folder where the sample scripts (or custom scripts) are located, click on one of the scripts, and press "Open" (see the Figure 23).

Figure 23. Script folder selection



Now the "Current Script" should show the file extension of the selected script.

Navigating back to the L9961 Demo tab and pressing the  for the Script Select box, it should now show the scripts located in the folder selected in the Register Load tab.

3.3.1.2 *Voltage Conversion and Cell Balancing*

Figure 24. Voltage Conversion and Cell Balancing section

The voltage conversion and cell balancing section of the L9961 Demo tab (see the Figure 24) allows the user to easily read the cell voltages and associated fault statuses. Additionally, the Vbat and Vsum voltages and associated fault statuses are also available.

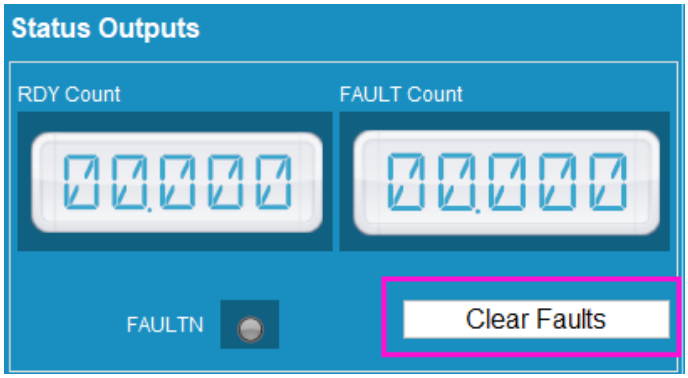
The fault statuses are indicated by LEDs. Each LED is associated with a specific I^2C register bit. When no faults are present, all LEDs will be illuminated in green (see the Figure 25). In the event that one or more of the LEDs changes to red, this indicates a fault is present (see the Figure 26).

Figure 25. No fault present

Figure 26. Fault present

In order to clear the fault, first resolve the condition causing the fault, then press the "Clear Faults" button located in the "Status Outputs" section of the tab (see the Figure 27).

Figure 27. Clear faults button location



Note: In order for this section to properly update the visual components, the selected script must properly configure the device for voltage acquisition. Details on properly configuring the device are outlined in the L9961 datasheet. To quickly access this functionality of the device, the user can select from one of the sample scripts.

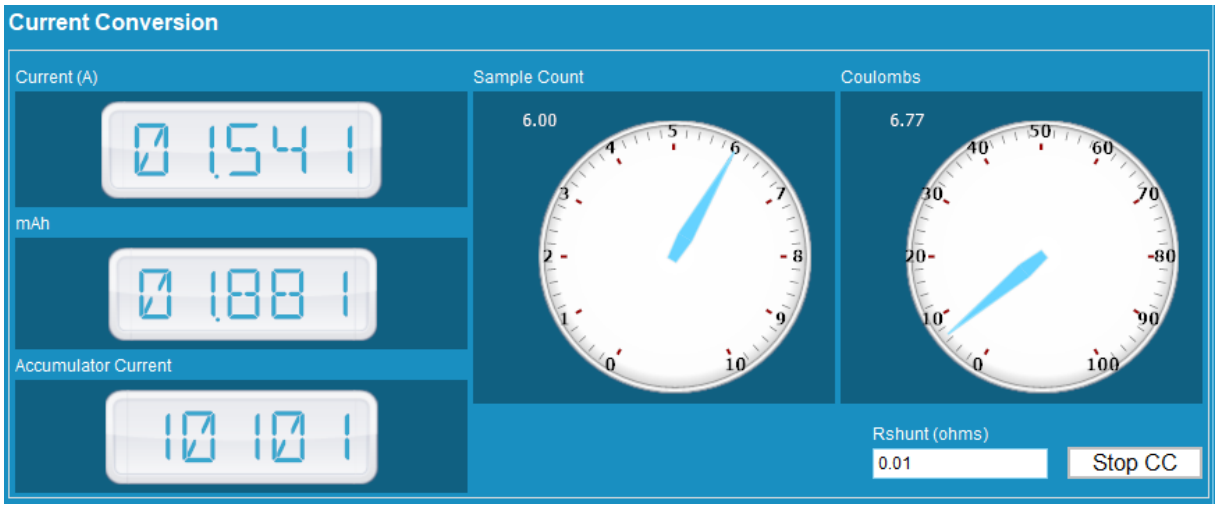
This section of the L9961 Demo tab also allows for the enabling and disabling of the balancing current on Cells 1-5. This is done by pressing the Balance ON/OFF buttons belonging to each cell (see the Figure 28).

Figure 28. Balance ON, Balance OFF buttons

Note: These buttons will change their text based on the current state of the BALx_ON I^2C bits. If the bit is "0" (default/ OFF), the button will read "Balance ON" to turn balancing on. If the bit is "1" (ON) the button will read "Balance OFF" to turn balancing off.

3.3.1.3 *Current conversion*

Figure 29. Current Conversion section



The Current Conversion section of the L9961 demo tab (see the Figure 29) is used to clearly display the data acquired by the current conversion routine of the L9961.

When the current conversion routine is commanded through the script, the accumulator current, sample count, and current values will be collected from the device.

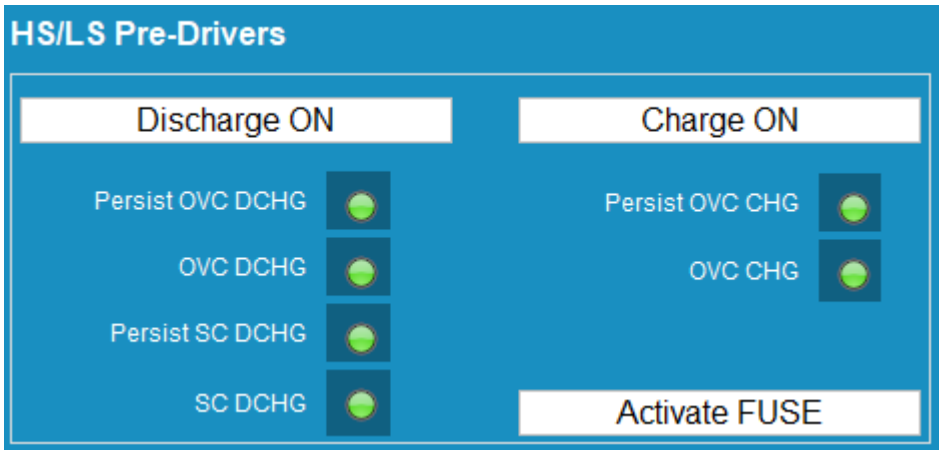
In order to enable the Coulomb Counting routine and obtain the mAh, and Coulomb count, the user must enter the correct Rshunt value (default value matches demo board value) and press "Start CC" (see the Figure 30).

Figure 30. Start Coulomb Counting routine

Note: In order for this section to properly update the visual components, the selected script must adequately configure the device for current conversion acquisition. Details on appropriately configuring the device are outlined in the L9961 datasheet. To quickly access this functionality of the device, the user can select from one of the sample scripts.

3.3.1.4 HS/LS Pre-Drivers

Figure 31. **HS/LS Pre-Drivers section**

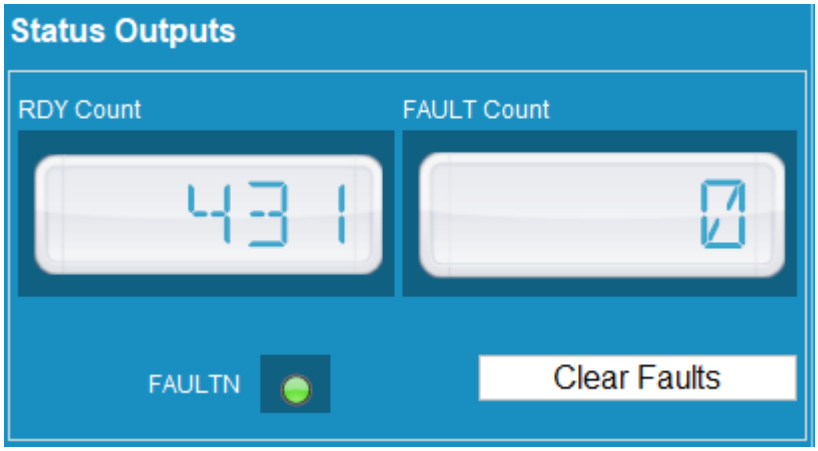


The HS/LS Pre-Drivers section of the demo tab (see the Figure 31) is used to easily control the three pre-driver stages of the L9961 - external Charge, Discharge, and Fuse switches. The relevant fault statuses are indicated by the LEDs. The behavior of the LEDs is described in detail in the Section 3.3.1.2 Voltage Conversion and Cell Balancing.

Note: In order for this section to properly update the visual components, the selected script must adequately configure the device by programming the pre-driver stages. Details on appropriately configuring the device are outlined in the L9961 datasheet.

3.3.1.5 Status Outputs

Figure 32. **Status Outputs section**



The Status Output section of the L9961 Demo tab (see the Figure 32) is used to display the count value for the data-ready interrupt pin (RDY) and the fault interrupt pin (FAULT). Additionally, this section of the tab is used to clear all faults and displays the status of the FAULTN pin with a LED.

To clear all existing faults, press "Clear Faults". Note that this will only clear the faults that have not saturated their counters. In the case a fault will not clear, the fault counter has saturated, and the user must follow the steps outlined in section 3.4.5 of the L9961 datasheet.

3.3.1.6 Die Temp and NTC

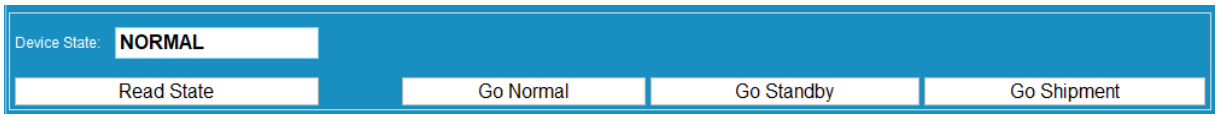
Figure 33. Die Temp and NTC section

The Die Temp + NTC section of the L9961 Demo tab (see the Figure 33) is used to display the NTC voltage and the die temperature of the device. The relevant fault statuses are indicated by the LEDs. The behavior of the LEDs is described in detail in the Section 3.3.1.2 Voltage Conversion and Cell Balancing.

Note: In order for this section to properly update the visual components, the selected script must adequately configure the device for NTC voltage acquisition. Details on appropriately configuring the device are outlined in the L9961 datasheet. To quickly access this functionality of the device, the user can select from one of the sample scripts.

3.3.1.7 Device State

Figure 34. Device State section



The Device State section of the L9961 Demo tab is used to read and select the state of the device (see the Figure 34).

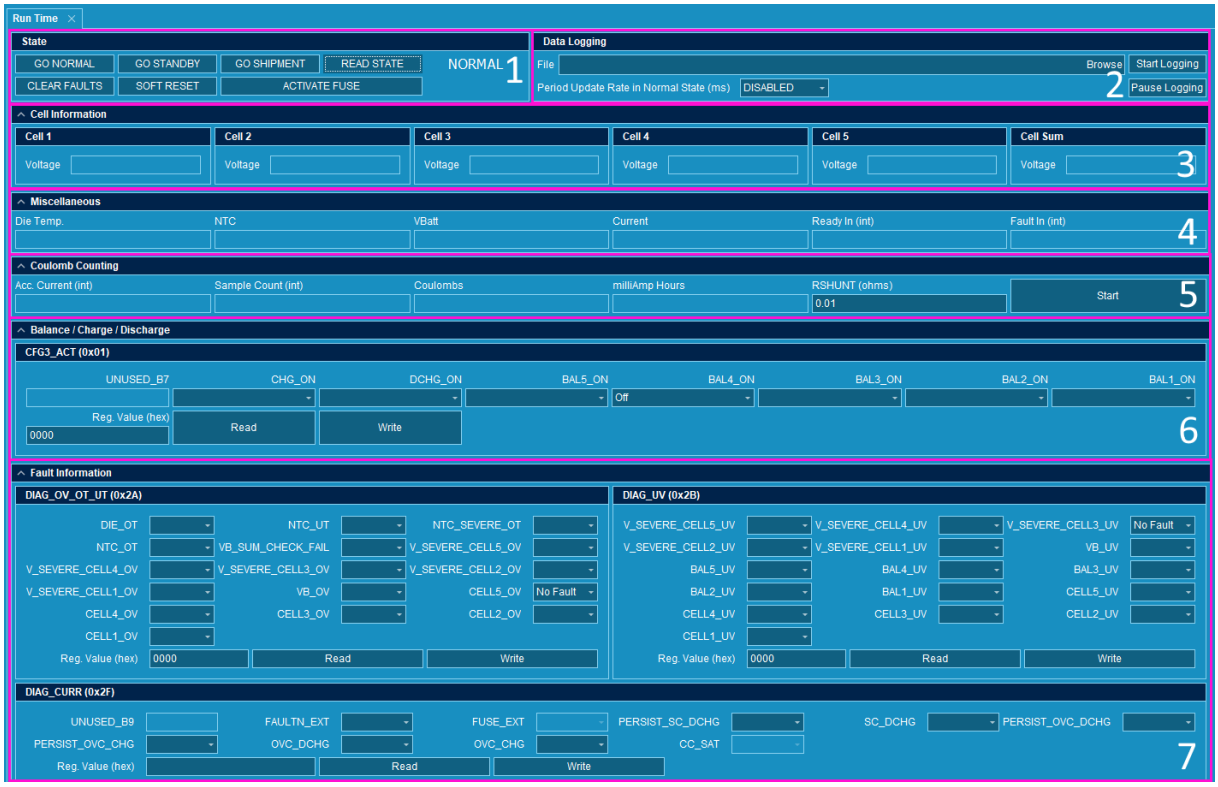
The L9961 has three states: Shipment, Standby, and Normal (for details on the different states, refer to the datasheet section "Device Functional States"). To read the current state of the device, press the "Read Mode" button. To change the state of the device, press one of the state buttons (see the Figure 35):

Figure 35. Device state selection

Note that any externally driven state changes cannot be tracked by the GUI and may result in the incorrect state being depicted in the GUI. To avoid this, the state should be controlled in the GUI using the "GO" and "READ" buttons.

3.3.2 Run Time tab

Figure 36. **Run Time tab display**



This page consists of seven sections:

1. State
2. Data Logging
3. Cell Information
4. Miscellaneous
5. Coulomb Counting
6. Balance/Charge/Discharge
7. Fault Information

Each section can be collapsed or expanded by pressing the  next to the name of the section.

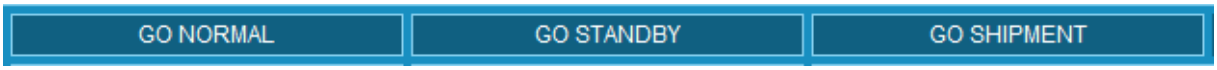
3.3.2.1 *State*

In this section the state of the device can be read and selected. Additionally, the user can clear all of the device faults and activate the fuse (see the Figure 37).

Figure 37. State section

The L9961 has three states: Shipment, Standby, and Normal (for details on the different states, refer to the datasheet section "Device Functional States"). To read the current state of the device, press the "Read State" button. To change the state of the device, press one of the state buttons (see the Figure 38):

Figure 38. Device state selection



Note that any externally driven state changes cannot be tracked by the GUI and may result in the incorrect state being depicted in the GUI. To avoid this, the state should be controlled in the GUI using the "GO" and "READ" buttons.

To clear all existing faults, press "Clear Faults". Note that this will only clear the faults that have not saturated their counters. In the case a fault will not clear, the fault counter has saturated, and the user must follow the steps outlined in section 3.4.5 of the L9961 datasheet.

3.3.2.2 *Data Logging*

Figure 39. Data Logging section

This section of the tab is used to update the Run Time tab at a set periodic rate. It is also used to log the Run Time tab data to a specified file location.

To save the Run Time tab data to a log file, press "Browse" and create the csv file in a selected folder location. Then press "Start Logging" to begin recording the data to the csv file. To pause logging data, press "Pause Logging". To stop data logging, press “Stop Logging”. The csv file will update at the specified update rate described next.

To periodically update the Run Time tab, press the drop-down arrow and select the desired refresh rate time (Disabled, 50 ms, 75 ms, 100 ms, 150 ms, 200 ms, 250 ms). The Run Time tab will then update all of the fields at the specified rate and log the results to the data log file (if "Start Logging" has been pressed).

Note that this feature will only work if the device has first been initialized as described in the following sections.

Opening the log file from the specified save location will yield the following (see the Figure 40):

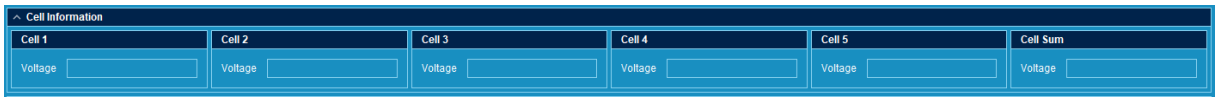
Figure 40. Run Time tab data log file

L9961 Eval Board Periodic Read Log
Collection Start: 2021-05-05 11:08:50

Read Timestamp (ms)	Cell 1(mV)	Cell 2(mV)	Cell 3(mV)	Cell 4(mV)	Cell 5(mV)	Battery(mV)	Current(A)	NTC(Raw	Die Temp	CHG_ON	DCHG_ON	BAL5_ON	BAL4_ON	BAL3_ON	BAL2_ON	BAL1_ON	DIAG_OV_	DIAG_UV	DIAG_CUR	FAULTN	CNT
0	2397.3	2408.28	2399.74	2404.62	2403.4	11998.7	0	0	22.509	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
204	2397.3	2409.5	2398.52	2405.84	2402.18	11992.6	0	0	22.509	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
401	2398.52	2410.72	2398.52	2405.84	2403.4	11992.6	0	0	22.313	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
600	2397.3	2410.72	2398.52	2405.84	2404.62	12004.8	0	0	21.725	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
800	2398.52	2408.28	2399.74	2404.62	2404.62	12004.8	0	0	22.117	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
1002	2398.52	2407.06	2400.96	2404.62	2405.84	12004.8	0	0	21.921	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
1201	2398.52	2408.28	2399.74	2403.4	2404.62	12004.8	0	0	22.117	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
1400	2397.3	2409.5	2398.52	2405.84	2403.4	11998.7	0	0	22.313	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
1601	2397.3	2408.28	2399.74	2404.62	2403.4	11998.7	0	0	22.117	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
1800	2397.3	2409.5	2398.52	2405.84	2403.4	11998.7	0	0	22.313	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
2001	2396.08	2410.72	2397.3	2407.06	2402.18	11992.6	0	0	22.901	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
2201	2398.52	2409.5	2399.74	2405.84	2403.4	11998.7	0	0	22.117	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0
2400	2398.52	2408.28	2399.74	2404.62	2404.62	11998.7	0	0	22.117	0	0	0	0	0	0	0	0000 0000	0000 0000	0 0000 000		0

3.3.2.3 *Cell Information*

Figure 41. Cell Information section



This section of the tab displays the cell voltages in millivolts (mV).

In order to acquire the cell and stack voltage measurements, the L9961 must first be configured in the following order using the NVM tab (see the Section 3.3.3 Main tab):

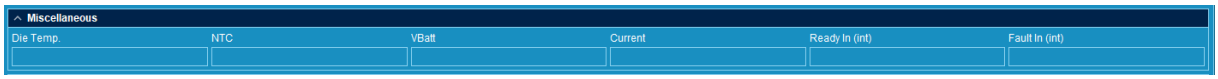
1. Set UV/OV thresholds (REG: 0x06, 0x07, 0x09, 0x0A, 0x0B, 0x0C)
2. Set threshold failure counters (REG: 0x06, 0x07, 0x09, 0x0A, 0x0B)
3. Enable cells and VB to be read (at least 3 cells must be enabled) (REG: 0x04)
4. Set measurement times (REG: 0x02)

See section 3.4 of the L9961 datasheet for details on configuring the voltage conversion routine execution period (T_{MEAS_CYCLE}).

The user also has the option to create a script that will handle these steps using the Register Load tab (see the Section 3.3.5 Fault tab). This is a more advanced method and only recommended if the user is familiar with the I^2C registers.

3.3.2.4 *Miscellaneous*

Figure 42. Miscellaneous section



This section of the tab is used for miscellaneous measurements. Specifically:

- Die temperature measurement
- NTC voltage measurement
- VBatt voltage measurement
- Current measurement
- Ready In pulse counter
- Fault In input counter

3.3.2.4.1 Die temperatuere measurement

The die temperature measurement is enabled and running by default. When the user selects the periodic update rate (see the Section 3.3.1.2 Voltage Conversion and Cell Balancing), this measurement will update at that rate.

3.3.2.4.2 NTC voltage measurement

The NTC voltage measurement is not enabled by default and must be configured by the user if measurement is needed. To configure the L9961 to perform NTC measurements, the following steps must be taken (in order):

1. Set UT/OT thresholds (REG: 0x0D, 0x0E, 0x0F)
2. Set threshold failure counters (REG: 0x0D, 0x0E)
3. Enable cells and NTC to be read (at least 3 cells must be enabled) (REG: 0x04)
4. Set measurement times (REG: 0x02)

3.3.2.4.3 VBatt voltage measurement

The VBatt voltage measurement is not enabled by default and must be configured by the user if measurement is needed. To configure the L9961 to perform VBatt measurements, the following steps must be taken (in order):

1. Set UV/OV thresholds (REG: 0x0A, 0x0B, 0x0C)
2. Set threshold failure counters (REG: 0x0A, 0x0B)
3. Enable cells and VB to be read (at least 3 cells must be enabled) (REG: 0x04)
4. Set measurement times (REG: 0x02)

3.3.2.4.4 Current measurement

The current measurement is not enabled by default and must be configured by the user if measurement is needed. To configure the L9961 to perform current measurements, the following steps must be taken (in order):

1. Set OVC thresholds (REG: 0x10, 0x11)
2. Enable cells and CSA to be read (at least 3 cells must be enabled) (REG: 0x04)
3. Set measurement times (REG: 0x02)

3.3.2.4.5 Ready in and Fault In counters

The Ready In and Fault In counters monitor the RDY and FAULTN pins of the L9961.

The Ready In counter tracks all pulses that occur on the RDY pin (section 3.8.1 of the L9961 datasheet).

The Fault In counter tracks all instances of the FAULTN line going LOW (section 3.8.2 of the L9961 datasheet).

3.3.2.5 Coulomb Counting

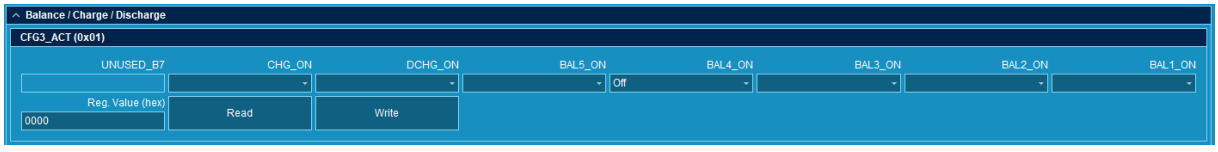
Figure 43. Coulomb Counting section

This section of the Run Time tab is used for the Coulomb Counting feature of the L9961. This measurement is disabled by default and must be configured by the user if measurement is needed. To configure the L9961 to perform Coulomb Counting, the following steps must be taken (in order):

1. Set OVC thresholds (REG: 0x10, 0x11)
2. Enable cells and CC_ACC to be read (at least 3 cells must be enabled) (REG: 0x04)
3. Set measurement times (REG: 0x02)

3.3.2.6 Balance/Charge/Discharge (CFG3_ACT:0x01)

Figure 44. Balance/Charge/Discharge section

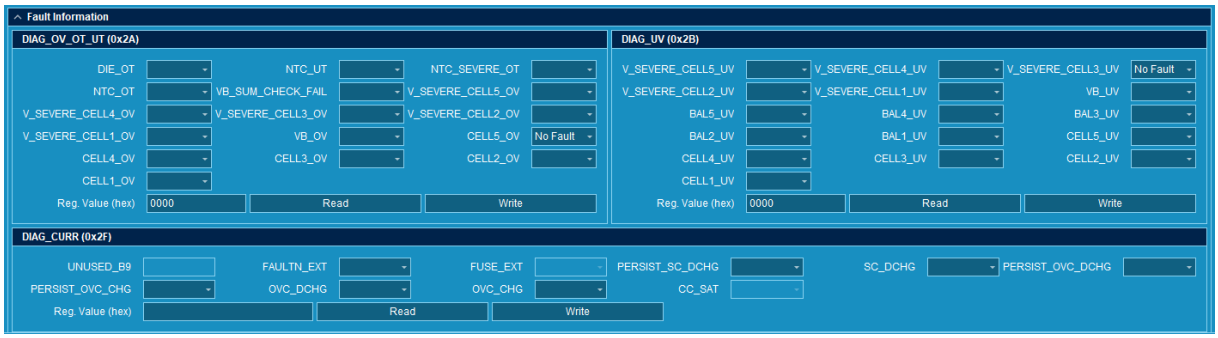


This section provides access to the balance, charge, and discharge fields in the register CFG3_ACT. This section allows the user to turn on or off the respective field.

Pressing read will read the values for the register. Pressing write will write the values for the register. After reading/writing or modifying a field, the "Reg. Value" will contain the register value in hexadecimal format.

3.3.2.7 *Fault Information*

Figure 45. Fault Information section



This section of the Run Time tab displays all of the available fault information for the L9961.

The user can press "Read" to retrieve the current fault status, for any of the three fault registers, at any time. All of the faults will be displayed as "Fault" and highlighted in red (see the Figure 46). All fields with no faults present will be displayed as "No Fault" (see the Figure 47). Additionally, the returned register data will be shown in the "Reg. Value (hex)" field in hexadecimal format (see the Figure 48).

Figure 46. Example of fault bit flagged

Figure 47. Example of no fault present

Figure 48. Example of returned DIAG_OV_OT register data

In order to clear faults, press "Write" to clear the flagged fault bits. Note that faults can only be cleared if the fault counters have not saturated, and the fault is no longer present. See section 3.4.5 of the L9961 datasheet for details.

All grayed out fields are read-only and cannot be modified by the user (see the Figure 49).

Figure 49. Grayed out field in Fault Information section

3.3.3 Main tab

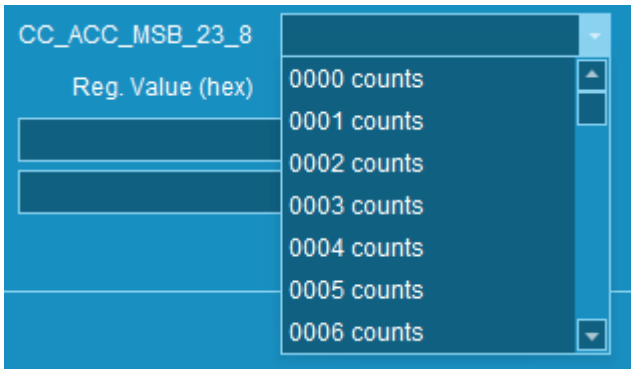
Figure 50. **Main tab display**

The Main tab provides access to all of the registers not part of the NVM group, or fault group. All register values are presented in hexadecimal format. For more information on I^2C registers, refer to the L9961 datasheet.
As mentioned previously, grayed out fields are read-only and cannot be modified by the user (see the Figure 51).

Figure 51. **Read only register bit**

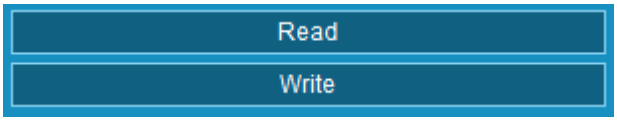
Fields that are writable are indicated by white drop-down boxes. The selection lists only valid entries (see the Figure 52).

Figure 52. **Selectable bit write options**



For each register section, a button to read will be available. Registers that allow write operations will display both write and read buttons (see the Figure 53).

Figure 53. Read and Write register buttons



Following a read or write command, the register data will be displayed in the corresponding "Reg. Value (hex)" field in hexadecimal format (see the Figure 54).

Figure 54. Register value field

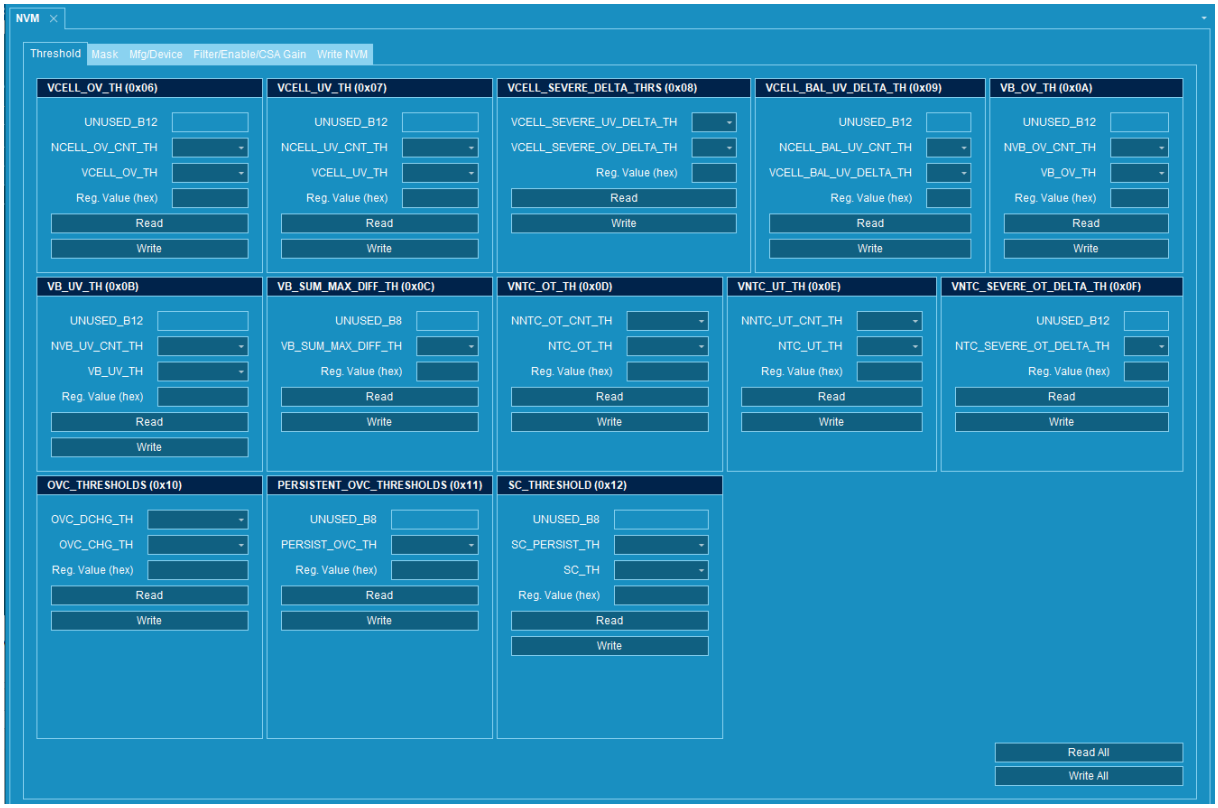
The option to read or write all registers within the Main tab is also available. In the lower right corner of the tab the "Read All" and "Write All" buttons can be used to perform this operation (see the Figure 55).

Figure 55. Read All and Write All buttons

Note that the "Write All" button only applies to registers on the tab that support write operations.

3.3.4 NVM tab

Figure 56. **NVM tab display**



The NVM tab provides access to all of the registers that are part of the NVM group. All register values are presented in hexadecimal format. For more information on I^2C registers, refer to the L9961 datasheet.
The NVM page consists of five tabs (see the Figure 57):

- Threshold
- Mask
- Manufacturer/Device
- Filter/Enable/CSA Gain
- Write NVM

Figure 57. **NVM tab selections**

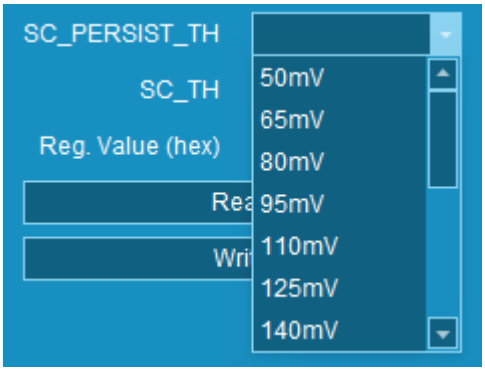
3.3.4.1 *Threshold, Mask, Mfg/Device, Filter/Enable/CSA Gain tabs*

As mentioned previously, grayed out fields are read-only and cannot be modified by the user (see the Figure 58).

Figure 58. **Read only register bit**

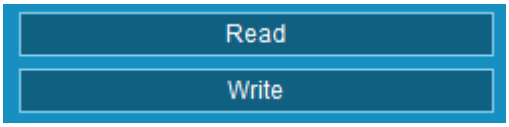
Fields that are writable are indicated by white drop-down boxes. The selection lists only valid entries (see the Figure 59).

Figure 59. **Selectable bit write options**



For each register section, a button to read will be available. Registers that allow write operations will display both write and read buttons (see the Figure 60).

Figure 60. **Read and Write register buttons**

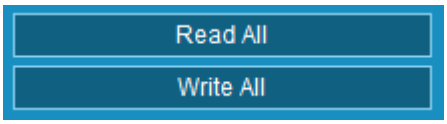


Following a read or write command, the register data will be displayed in the corresponding "Reg. Value (hex)" field in hexadecimal format (see the Figure 61).

Figure 61. **Register value field**

The option to read or write all registers within the selected NVM tab is also available. In the lower right corner of the tab the "Read All" and "Write All" buttons can be used to perform this operation (see the Figure 62. Read All and Write All buttons).

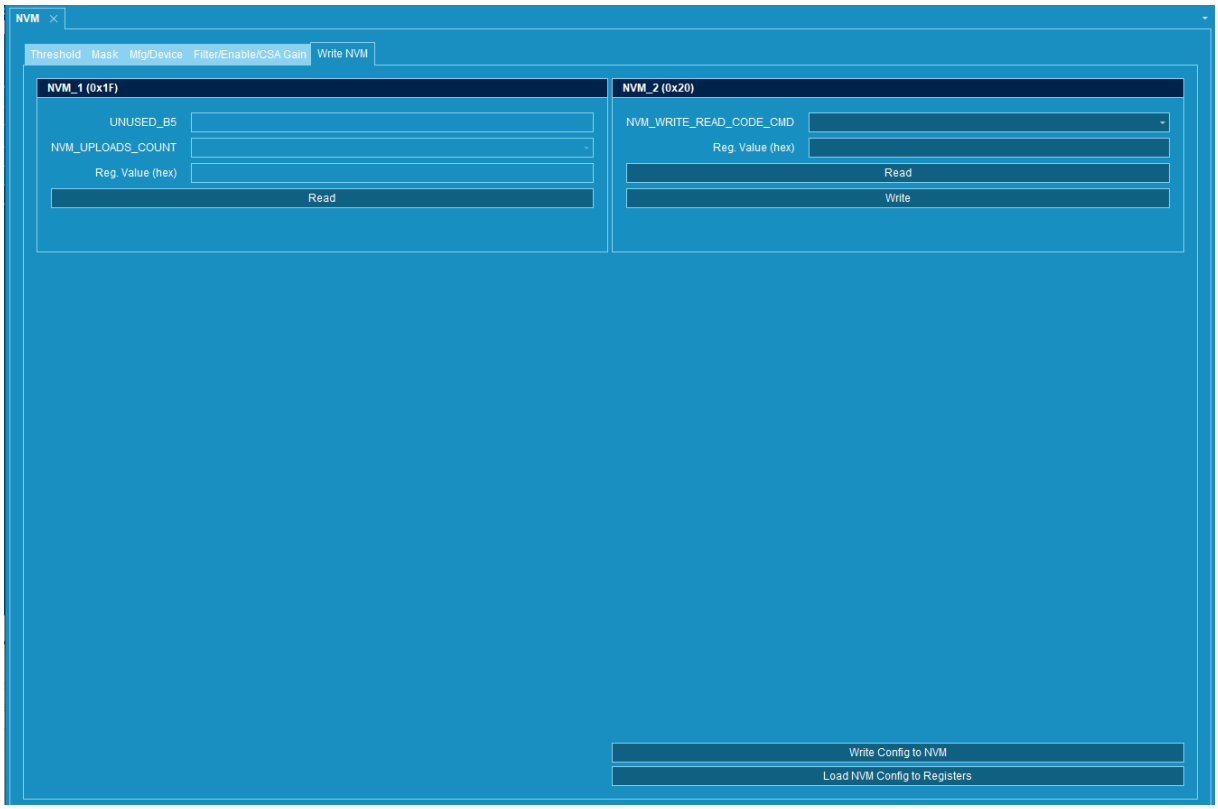
Figure 62. **Read All and Write All buttons**



Note that the "Write All" button only applies to registers on the tab that support write operations.

3.3.4.2 *Write NVM tab*

Figure 63. NVM tab display



Caution: This tab should only be used by those users that are well-versed with the L9961 device. The L9961 NVM is limited to a total of 31 write cycles. The user can read the number of NVM uploads using the NVM_1 "Read" button. If the limit is exceeded, data retention is not guaranteed!

See section 3.10 of the L9961 datasheet for details on writing to the NVM.

3.3.5 Fault tab

Figure 64. **Fault tab display**

The fault tab provides access to all of the registers that are part of the fault group. All register values are presented in hexadecimal format. For more information on I^2C registers, refer to the L9961 datasheet.

The Fault tab performs the same as the fault section of the Run Time tab (see the Section 3.3.1.7 Device State), and the information is duplicated here:

The user can press "Read" to retrieve the current fault status, for any of the three fault registers, at any time. All of the faults will be displayed as "Fault" and highlighted in red (see the Figure 65). All fields with no faults present will be displayed as "No Fault" (see the Figure 66). Additionally, the returned register data will be shown in the "Reg. Value (hex)" field in hexadecimal format (see the Figure 67).

Figure 65. **Example of fault bit flagged**

Figure 66. **Example of no fault present**

Figure 67. **Location of returned DIAG_OV_OT register data**

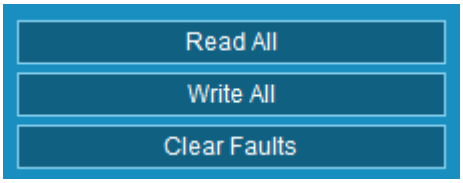
In order to clear faults, press "Write" to clear the flagged fault bits. Note that faults can only be cleared if the fault counters have not saturated, and the fault is no longer present. See section 3.4.5 of the L9961 datasheet for details.

As mentioned previously, grayed out fields are read-only and cannot be modified by the user (see the Figure 68).

Figure 68. Read only register bit

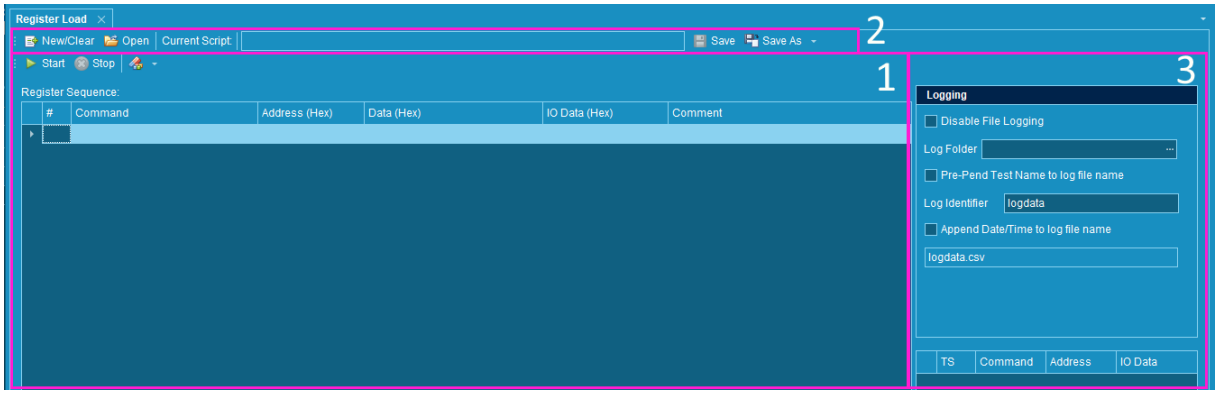
The option to read all, write all, and clear faults within the Fault tab is also available. In the lower right corner of the tab the "Read All", "Write All", and "Clear Faults" buttons can be used to perform this operation (see the Figure 69).

Figure 69. Read All and Write All buttons



3.3.6 Register Load tab

Figure 70. Register Load tab display



This tab allows the user to create a sequence of commands, or script, that can be run on the device (see the Figure 70. Register Load tab display - 1). The user can save and recall previously created register sequences/ scripts see the Figure 70. Register Load tab display - 2). Additionally, when the command sequence is running, the user has the option of logging the data/results (see the Figure 70. Register Load tab display - 3).

Figure 71. Register sequence display

The user can create scripts by entering commands into the register sequence grid (see the Figure 71). Each line consists of the following six columns:

1. **#** - The line number of the command.
   – This is available for reference and is auto generated by the GUI.
2. **Command** - The command to be issued.
   – This is selected by the user. The available commands are outlined in more detail in the Section 3.3.6.1 Commands of this document.
3. **Address (Hex)** - The address of the register to be used in the command.
   – Not all commands require the Address field and are outlined in the Section 3.3.6.1 Commands.
4. **Data (Hex)** - The data for the command to be used.
   – Not all commands require the Data field and are outlined in the Section 3.3.6.1 Commands.
5. **IO Data (Hex)** - Commands that generate data will display the results here.
6. **Comment** - If needed, a comment can be added to each line.

3.3.6.1 *Commands*

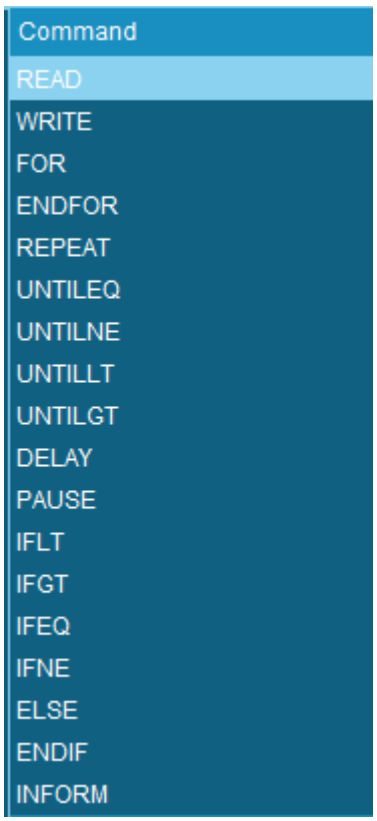
The section of the document outlines and explains the available commands that can be used in the register sequence.

The available commands are as follows:

1. **READ** - Read a value from a register.
2. **WRITE** - Write a value to a register.
3. **FOR** - Begin a FOR loop that starts at 1 and goes to set value "n".
4. **ENDFOR** - Close of FOR loop.
5. **REPEAT** - Begin of until-equal-to loop.
6. **UNTILEQ** - Close until-equal-to loop when result equals "n".
7. **UNTILNE** - Close until-equal-to loop when result does not equal "n".
8. **UNTILLT** - Close until-equal-to loop when result is less than "n".
9. **UNTILGT** - Close until-equal-to loop when result is greater than "n".
10. **IFLT** - Perform proceeding lines of commands if result is less than "n".
11. **IFGT** - Perform proceeding lines of commands if result is greater than "n".
12. **IFNE** - Perform proceeding lines of commands if result is not equal to "n".
13. **IFEQ** - Perform proceeding lines of commands if result is equal to "n".
14. **ELSE** - Perform proceeding lines of commands in case prior IF statement is not met.
15. **ENDIF** - Close IF statement.
16. **DELAY** - Delay for a specified number of milliseconds.
17. **INFORM** - Output user supplied information.
18. **PAUSE** - Allows for pausing of the script with a prompt to continue.

These commands can be selected by clicking in the "Command" field of a selected row:

Figure 72. Command selection in Register Sequence



Once the command is selected, the Address or Data may need to be filled in.

Note: The values for Data and Address are assumed to be hexadecimal numbers. There is one exception to this and that is when using the INFORM command. In this case, the Data field can be any alphanumeric character.

In the following sections, commands requiring text for the address and/or data fields will have the following descriptor after the command label:

- <Address>
- <Data>
  - *Example:* The READ command requires the user to enter a valid hexadecimal address. The section for the READ command will read as follows: READ <Address>.

If either of these fields are not required, the above descriptor(s) will not be present.

3.3.6.1.1 READ <Address>

The READ command is used to request a read of a specified address. The address must be entered in hexadecimal format in order to be acknowledged as a valid command. The returned data will be available in the "IO Data (Hex)" column on the same line.

Figure 73. Valid READ command

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	READ	21			

3.3.6.1.2 WRITE <Address> <Data>

The WRITE command is used to command a write of data to a specified address. The address and data must be entered in hexadecimal format in order to be acknowledged as a valid command. The returned data will be available in the "IO Data (Hex)" column on the same line.

Figure 74. Valid WRITE command

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	WRITE	6	AFF		N=10, Cell OV = 4.9976V

3.3.6.1.3 DELAY <Data>

The DELAY command is used to set a delay for a set number of milliseconds. The time of the delay is entered in the "Data (Hex)" field.

Figure 75. Valid DELAY command

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	DELAY		A		

Note: Depending on the PC hardware and software the evaluation software is running on, there is potential for the delay to exhibit some variances from PC to PC.

3.3.6.1.4 FOR <Data>

The FOR command is used to build a for-loop within the register sequence. The value entered in the "Data (Hex)" field specifies the number of times the loop will be executed (must be entered in hexadecimal format). An ENDFOR command must follow a FOR command to close the loop. Failure to do so will result in errors.

Figure 76. Valid FOR loop structure

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	FOR		5		
1	READ	A			
2	WRITE	E	AF		
3	ENDFOR				

Note: It is not possible to interleave FOR loops with other REPEAT/IF/FOR statements. In order to repeat a block containing a FOR loop, the REPEAT command would have to be inserted before (or after) the FOR loop command (see the Figure 77):

Figure 77. Valid FOR loop with repeat until equal statement

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	REPEAT				
1	FOR		5		
2	READ	A			
3	WRITE	E	AF		
4	ENDFOR				
5	UNTILEQ	A	5		

3.3.6.1.5 ENDFOR

The ENDFOR command is used to close a FOR loop within the register sequence. A FOR command must proceed an ENDFOR command to complete the FOR loop. Failure to do so will result in errors.

Figure 78. Valid ENDFOR command structure

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	FOR		5		
1	READ	A			
2	WRITE	E	AF		
3	ENDFOR				

3.3.6.1.6 REPEAT

The REPEAT command is used to create a repeat-until-equal conditional statement within the register sequence. A REPEAT command must be followed by a UNTILEQ/UNTILNE/UNTILLT/UNTILGT command to complete the repeat-until-equal conditional statement. Failure to do so will result in errors.

Figure 79. Valid REPEAT command structure

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	REPEAT				
1	READ	A			
2	UNTILEQ	A	5		

3.3.6.1.7 UNTILEQ/UNTILNE/UNTILLT/UNTILGT <Address> <Data>

The UNTILEQ/UNTILNE/UNTILLT/UNTILGT commands are used to close a repeat-until-equal conditional statement. A REPEAT command must proceed an UNTILxx command to complete the conditional statement. Failure to do so will result in errors.

The UNTILxx command definitions are as follows:

- UNTILEQ = Until address result is equal to "n"
- UNTILNE = Until address result is not equal to "n"
- UNTILLT = Until address result is less than "n"
- UNTILGT = Until address result is greater than "n"

The value entered in the "Address (Hex)" field specifies the register to perform the repeat-until-equal conditional statement on. The value entered in the "Data (Hex)" field specifies comparison value "n".

Note: The address to be evaluated must be READ prior to an UNTILxx command.

Figure 80. Valid UNTILxx command structure

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	REPEAT				
1	READ	A			
2	UNTILEQ	A	5		

3.3.6.1.8 IFLT/IFGT/IFEQ <Address> <Data>

The IFLT/IFGT/IFEQ commands are used to create a conditional IF statement within the register sequence. An IFxx command must be followed by an ENDIF or ELSE command to complete the conditional IF statement. Failure to do so will result in errors.

The IFxx command definitions are as follows:

- IFLT = If address result is less than "n"
- IFGT = If address result is greater than "n"
- IFEQ = If address result is equal to "n"
- IFNE = If address result is not equal to "n"

The value entered in the "Address (Hex)" field specifies the register to perform the conditional IF statement. The value entered in the "Data (Hex)" field specifies comparison value "n".

If the IFxx condition is met, then the next series of commands are run.

Note: The address to be evaluated must be READ prior to an IFxx command.

Figure 81. Valid IFxx command structure

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	READ	A			
1	IFGT	A	5		
2	READ	D			
3	READ	B			
4	ENDIF				

3.3.6.1.9 ELSE

The ELSE command is used to create a branch from a previous conditional IF statement if the IFxx conditions are not met. An IFxx command (and its series of commands) must precede an ELSE command, and an ENDIF command must follow an ELSE command. Failure to do so will result in errors.

Figure 82. Valid ELSE command structure

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	READ	A			
1	IFGT	A	5		
2	READ	D			
3	READ	B			
4	ELSE				
5	WRITE	A	5		
6	ENDIF				

3.3.6.1.10 ENDIF

The ENDIF command is used to close a conditional IF statement within the register sequence. An IFxx or ELSE command must precede an ENDIF command to complete the IF statement. Failure to do so will result in errors.

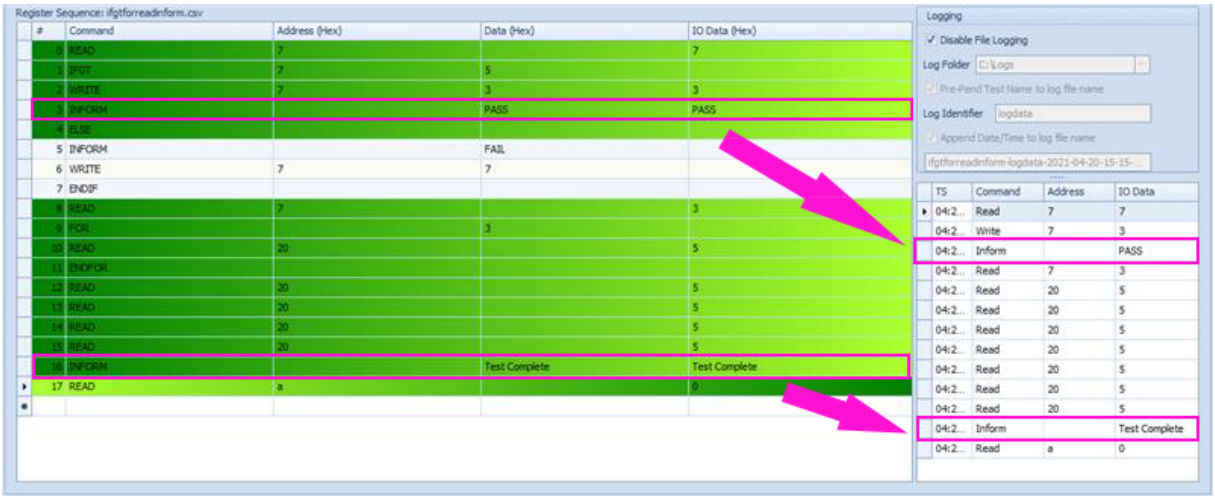
Figure 83. Valid ENDIF command structure

#	Command	Address (Hex)	Data (Hex)	IO Data (Hex)	Comment
0	READ	A			
1	IFGT	A	5		
2	READ	D			
3	READ	B			
4	ENDIF				

3.3.6.1.11 INFORM <Data>

The INFORM command is used to pass text to the log file. The text entered in the "Data (Hex)" field will be passed to the "IO Data (Hex)" field in the event that the script reaches the line. This instruction does not require data to be in hexadecimal format.

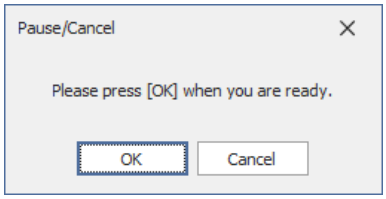
Figure 84. Valid INFORM command



3.3.6.1.12 PAUSE <Data>

The PAUSE command is used to pause the execution of the script. Once the script reaches this line, it will pause and prompt the user to continue or cancel (see the Figure 85). This can be useful in instances where a hardware adjustment needs to be made before proceeding to the next line in the script.

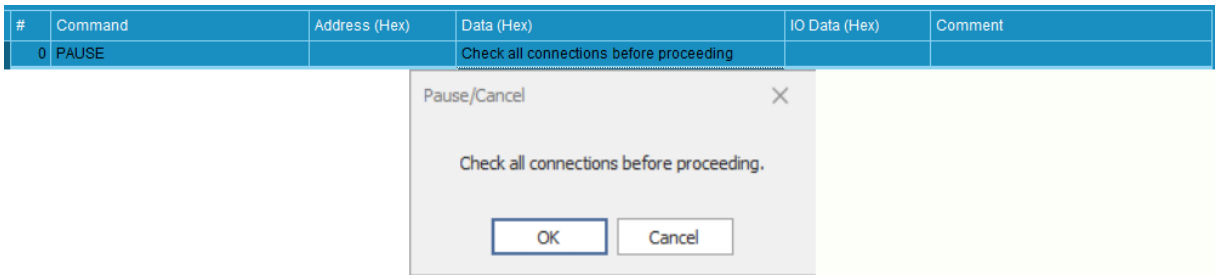
Figure 85. Pause prompt window



To proceed to the next line in the script, select "OK". Otherwise, if the user wishes to exit the script, press "Cancel."

The user also has the ability to change the text shown in the popup window. In order to change from the default, enter the desired text in the Data column. For example, if the user needs to check hardware connection before proceeding, the text "Check all connections before proceeding" can be entered in the Data column. This will then appear in the popup window with the script executes (see the Figure 86).

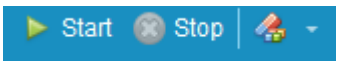
Figure 86. Customizing the text of a Pause popup window



3.3.6.2 *Run/Abort Register sequence*

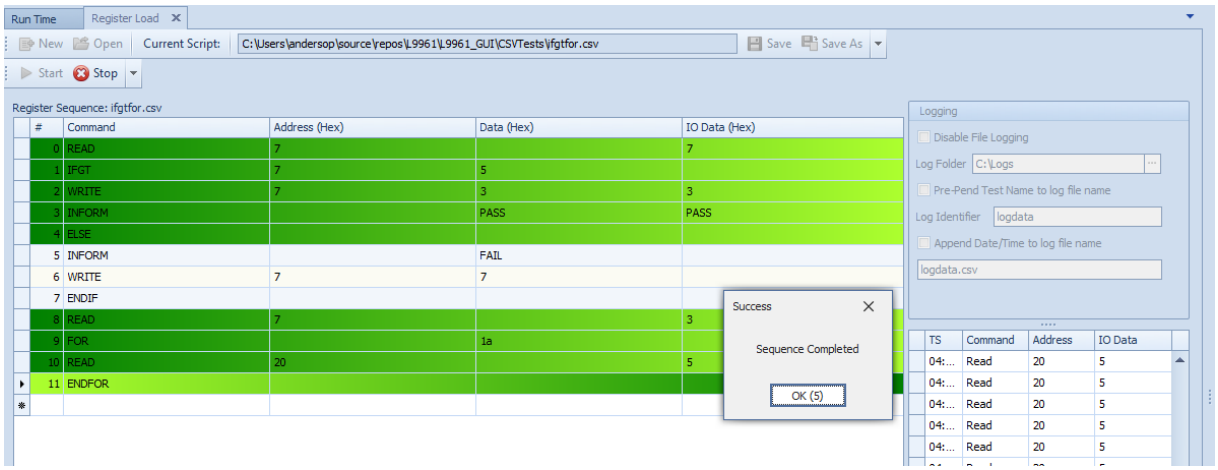
Once the user has completed the register sequence, the script can be run using the ‘Start’ button. Should it become necessary to abort a register sequence while it is running, the "Stop" button can be used (see the Figure 87).

Figure 87. Script "Start" button



After completing a sequence, a popup window with a six second timer will appear indicating the script ran successfully. Additionally, the lines that executed successfully will be indicated by turning green (see the Figure 88).

Figure 88. Completion of sequence



3.3.6.3 *Clear, Save and Load sequence*

Figure 89. Clear, Save and Load

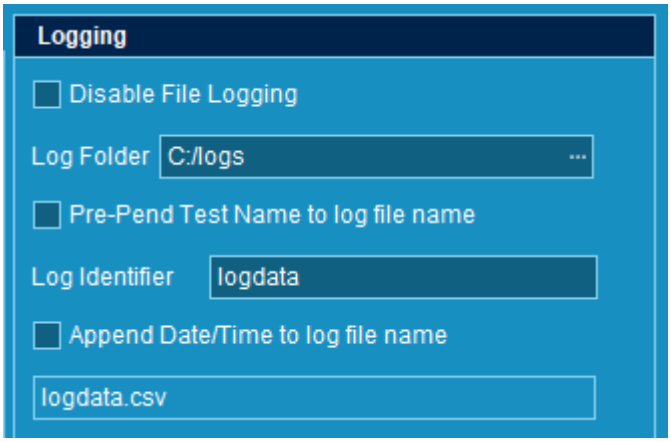
To clear the register sequence table and create a new sequence, press the "New/Clear" button.

Sequences can be saved via the “Save” button. A file dialog box will pop up if no filename has been previously selected. The file is saved as a comma separated file (csv).

To load a pre-existing sequence file, press the "Open" button and select a valid csv file through the file dialog window. Once the file has been selected, the register sequence will be loaded into the register sequence table.

3.3.6.4 *Logging sequence*

Figure 90. Sequence logging



The user has the option to log the results from a sequence execution.

Sequence logging can be enabled or disabled on demand through the "Disable File Logging" check box. By default, the check box is cleared, logging data to the "Log Folder" location. If the check box is highlighted, the file logging will be disabled.

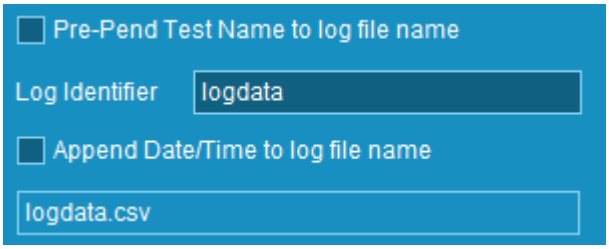
To configure the "Log Folder" location, press the three dots next to the text box, and navigate to the desired save location.

A valid log file name can contain the following:

- Current script name
- Log Identifier
- Date/time (YYYY-MM-DD-HH-MM-SS)

The fields above can be added/removed by using the following fields (see the Figure 91):

Figure 91. Log file name fields



Pre-pending the test script name to the identifier can be turned on or off via the checkbox.

If "Append Data/Time to log file name" is checked, the data time will be appended to the file name.

If neither the pre-pend nor append checkboxes are selected, the log identifier will be used as the file name. If both checkboxes are selected, the file name will have the following format:

TestName-LogIdentifier-DataTime.csv.

Opening the log file from the specified save location will yield the following (see the Figure 92):

Figure 92. Register Load tab data log file

TimeStamp	Command	Address	Data
05:05:21:10:05:53:118	Write	6	aff
05:05:21:10:05:53:141	Write	7	a00
05:05:21:10:05:53:155	Write	8	0
05:05:21:10:05:53:171	Write	9	a00
05:05:21:10:05:53:186	Write	a	aff
05:05:21:10:05:53:205	Write	b	a00
05:05:21:10:05:53:218	Write	c	ff
05:05:21:10:05:53:240	Write	d	a000
05:05:21:10:05:53:251	Write	e	afff
05:05:21:10:05:53:266	Write	f	1
05:05:21:10:05:53:282	Write	10	0
05:05:21:10:05:53:304	Write	11	0
05:05:21:10:05:53:314	Write	12	0
05:05:21:10:05:53:340	Write	4	383f
05:05:21:10:05:54:752	Write	2	7ff

The header in the log grid contains the timestamp (TS), the command (Command), the address (Address), and the IO data (IO Data) (see the Figure 93. Sequence log grid). The grid and the log will only provide IO Data for READ, WRITE, and INFORM functions.

Figure 93. Sequence log grid

TS	Command	Address	IO Data
04:...	Read	20	5
04:...	Read	20	5
04:...	Read	20	5

Revision history

Table 1. **Document revision history**

Date	Version	Changes
02-May-2023	1	Initial release.

Contents

List of figures