GUARDIAN III Programmer Reference Guide

Changhong-Microsoft IoT Innovation Center

Revision History

Version	Description	Author	Date
V0.1	Initial version	Junhui Lan	7-21-2021
V1.0	Update	Junhui Lan	10-11-2021
V1.1	Re-organized and update API	Richard Liang	01-12-2022

Contents

1	Intro	duction		5
2	Get s	tarted Gu	uide	7
	2.1	Con	necting the device	7
	2.2	Inst	all the Azure Sphere SDK	8
	2.3	Inst	all Visual Studio and the Azure Sphere extension	9
	2.4	Hov	v to run the sample code	9
3	Samp	ole and T	est	.12
	3.1	GII	ICM680L	.12
	-	3.1.1	Run the sample code	.12
	-	3.1.2	APP LEDs	.13
	-	3.1.3	Test Lora	.14
		3.1.4	Test UART	.14
	3.2	GII	ICM680E	.15
		3.2.1	Run the sample code	.15
		3.2.2	APP LEDs	.16
		3.2.3	Test Southbound Ethernet (W5500)	.17
	-	3.2.4	Test UART	.18
	3.3	GII	ICM680R	.19
		3.3.1	Run the sample code	.19
		3.3.2	APP LEDs	.20
		3.3.3	Test CAN	.20
	-	3.3.4	Test RS422	.23
		3.3.5	Test RS485	.24
4	Appli	ication A	РІ	.26
	4.1	Con	nmon API	.26
	2	4.1.1	gdn_sys_init()	.26
	2	4.1.2	gdn_daughter_board_power_down()	.26
	2	4.1.3	gdn_daughter_board_power_up()	.26
	2	4.1.4	gdn_change_network_type()	.27
	2	4.1.5	gdn_check_current_wifi_network_status()	.27
	2	4.1.6	gdn_check_network_ready()	.27
	4	4.1.7	gdn_get_network_type()	.27
	4	4.1.8	gdn_init_led()	.27
	2	4.1.9	gdn_set_led_status()	.28
	2	4.1.10	gdn_set_network_by_type()	.28
	4	4.1.11	gdn_watch_dog_feed()	.28
	4	4.1.12	gdn_watch_dog_init()	.28
	4.2	Cell	ular related API	.28
	2	4.2.1	gdn_init_cellular()	.29
	4	4.2.2	gdn_config_cellular()	.29
	4	4.2.3	gdn_get_esim_id()	.29
	4	4.2.4	gdn_get_gps()	.29

	4.2.5	gdn_handle_atcmd()	
	4.2.6	gdn_get_cellular_status()	
	4.2.7	gdn_cellular_connect_state()	
	4.2.8	gdn_get_wwan_info()	31
	4.2.9	gdn_print_cellular_debug_info()	31
4.3	GI	ICM680L	31
	4.3.1	gdn_close_usb_uart()	31
	4.3.2	gdn_open_usb_uart()	31
	4.3.3	gdn_send_message_to_usb()	32
	4.3.4	gdn_lora_recv()	32
4.4	GI	IICM680E	32
	4.4.1	gdn_close_usb_uart()	32
	4.4.2	gdn_open_usb_uart()	33
	4.4.3	gdn_tcp_client_connect()	33
	4.4.4	gdn_tcp_client_recv()	33
	4.4.5	gdn_tcp_client_send()	34
	4.4.6	gdn_southboundeth_recv()	34
	4.4.7	gdn_southboundeth_send()	34
4.5	GI	IICM680R	35
	4.5.1	gdn_close_can()	35
	4.5.2	gdn_close_rs422()	35
	4.5.3	gdn_close_rs485()	35
	4.5.4	gdn_open_can()	35
	4.5.5	gdn_open_rs422()	36
	4.5.6	gdn_open_rs485()	36
	4.5.7	gdn_send_message_to_rs422()	
	4.5.8	gdn_send_message_to_rs485()	37
	4.5.9	gdn_write_can()	37
Sup	port		38

5

1 Introduction

Guardian III is an IoT gateway product series based on Microsoft Azure Sphere OS. It provides end-to-end IoT security solutions and can be used in a variety of industrial fields.

Guardian III has multiple types of southbound interfaces, such as LoRa, Bluetooth, Ethernet, CAN, RS422, RS485, etc., which can be easily connected to many devices or sensors. At the same time, it also includes three types of northbound interfaces (Wi-Fi, Ethernet, Cellular), you can easily connect to the Azure Cloud.

The Guardian III series includes three different product forms, namely GIIICM680L, GIIICM680E, and GIIICM680R. The main difference between the three product forms is the southbound interface.

Product features:

GIIICM680L (GIIICM680L, GIIICM680L-CM2, GIIICM680L-CM6, GIIICM680L-CMA)

- ♦ AI-Link WF-M620-RSD2 module
- Northbound link:
 - Cellular(4G/LTE) data communication (GIIICM680L does not include this feature)
 - Dual-band Wi-Fi, 2.4GHz and 5GHz
 - Ethernet port: RJ45 female, 10Mbps
- Southbound link:
 - Bluetooth 5.0 connectivity
 - LoRa data communication
 - USB port (Power + Data)
- GNSS location report
- MCU programming and debugging port
- ♦ A/C adapter DC 5V/2A
- ◆ 3 LED indicators: 2 programmable Red/Green LEDs, 1 BLE activity LED
- Operating temperature: -40 °C to +85 °C
- Operating humidity: 5% to 95% RH
- Dimensions: 95mm*95mm*25mm
- Enclosure: Mountable, Black color

GIIICM680E (GIIICM680E, GIIICM680E-CE, GIIICM680E-CN)

- AI-Link WF-M620-RSD2 module
- Northbound link:
 - Cellular(4G/LTE) data communication (GIIICM680E does not include this feature)
 - Dual-band Wi-Fi, 2.4GHz and 5GHz
 - Ethernet port: RJ45 female, 10Mbps
- Southbound link:
 - Bluetooth 5.0 connectivity
 - Ethernet port: RJ45 female, up to 1Mbps

- USB port (Power + Data)
- GNSS location report
- MCU programming and debugging port
- ♦ A/C adapter DC 5V/2A
- ♦ 3 LED indicators: 2 programmable Red/Green LEDs, 1 BLE activity LED
- Operating temperature: -40 °C to +85 °C
- Operating humidity: 5% to 95% RH
- Dimensions: 95mm*95mm*25mm
- Enclosure: Mountable, Black color

GIIICM680R (GIIICM680R, GIIICM680R-CE, GIIICM680R-CN)

- ♦ AI-Link WF-M620-RSD2 module
- Northbound link:
 - Cellular(4G/LTE) data communication (GIIICM680R does not include this feature)
 - Dual-band Wi-Fi, 2.4GHz and 5GHz
 - Ethernet port: RJ45 female, 10Mbps
- Southbound link:
 - Bluetooth 5.0 connectivity
 - RS-422 or RS-485
 - CAN
- Modbus protocol supported
- GNSS location report
- MCU programming and debug port
- A/C adapter DC 5V/2A
- 3 LED indicators: 2 programmable Red/Green LED and 1 BLE activity LED,
- Operating temperature: -40 °C to +85 °C
- Operating humidity: 5% to 95% RH
- Dimensions: 95mm*95mm*25mm
- Enclosure: Mountable, Black color

2 Get started Guide

To use Guardian III series products for the first time, you must install Azure Sphere SDK and Dongle debug version driver.

For the establishment of Azure sphere development environment, please refer to: https://docs.microsoft.com/en-us/azure-sphere/

2.1 Connecting the device

Connect Guardian III series products to your PC through USB.



When plugged in, the device exposes four USB Serial Converters, the first time you plug in the device, the drivers should be automatically downloaded and installed. Installation can be slow. If the drivers are not installed automatically, right-click the device name in Device Manager and select Update driver. Alternatively, you can download the drivers from Future Technology Devices International (FTDI). Choose the driver that matches your Windows installation (32-bit or 64-bit).

To verify installation, open Device Manager. Under Universal Serial Bus controllers, look for four USB Serial Converters. Device Manager with four USB serial converters



2.2 Install the Azure Sphere SDK

- Download the SDK(<u>https://aka.ms/AzureSphereSDKDownload/</u>). Save the downloaded file on your PC.
- Run the downloaded .exe to install the SDK. Agree to the license terms, and then select Next.
- Select Install to begin installation.
- Accept the elevation prompt if one appears.
- When setup completes, restart your PC if the setup application requests it.

After installing the CLI for the first time, open the CLI through the start menu.

All Apps Documents Web Mo	ore 🔻	۰۰۰ بې بې
Azure Sphere Developer Command Prompt App App Azure IoT explorer App	>	Azure Sphere Developer Command Prompt
Apps Azure_Sphere_SDK_2010.exe Settings Access work or school Disconnect your device from your work or school	> > >	☐ Open Run as administrator Quen file location Open file location Unpin from Start ⊢ Pin to taskbar Uninstall
A azure - See web results Azure devops Azure lane Azure是什么意思	> > >	

Run azsphere show-version to check whether it is installed, and the correct version is installed.



2.3 Install Visual Studio and the Azure Sphere extension

- Install Visual Studio, please refer to <u>https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?preserve-view=true&view=vs-2019</u>.
- To install the Visual Studio extension for Azure Sphere, see <u>https://marketplace.visualstudio.com/items?itemName=AzureSphereTeam.AzureSphereSDKforVisualStudio2019</u>.

2.4 How to run the sample code

Here is an example of running the GIIICM680E_Demo.zip sample code. Please use the decompression tool to decompress the sample code before running.

GIIICM680E_Demo	2021/7/23 17:26
GIIICM680L_Demo	2021/7/23 17:25
GIIICM680R_Demo	2021/7/23 17:26
FreeRTOS_RTcore_LoRa_signed.imagepackage	2021/6/28 16:20
GIIICM680E_Demo.zip	2021/7/23 17:33
GIIICM680L_Demo.zip	2021/7/23 17:33
GIIICM680R_Demo.zip	2021/7/23 17:33
lan-enc28j60-isu0-int5_signed.imagepackage	2021/6/23 18:13
SouthBound_Ethernet_signed.imagepackage	2021/6/28 16:24

Use the "azsphere login" command to log in to your Microsoft account (Each PC only needs to log in once.)

G:\Guardian III>azsphere device claim warn: You are claiming a device to the following tenant: warn: a87ad796-8231-4e7b-8bd0-d4b362f61b65 warn: acraur90-2291-eer0-3500-ed0502161505 warn: Do you want to claim the device ID afca466b32e2142ea450003531a66fd43c60301a8fc2e24ca66fb07e1b16080c83f7a8e3b7a6ea1 a5526187f111a8bdff2bee9cef0a390952ea7eac189a4791 to this tenant? warn: You cannot change the Azure Sphere tenant this device is claimed to once this action has completed. Enter 'yes' to continue. Enter anything else to exit. l'aiming device. ulaimung device. Successfully claimed device ID 'afca466b32e2142ea450003531a66fd43c60301a8fc2e24ca66fb07e1b16080c83f7a8e3b7a6ea1a3526187f 111a8bdff2bee9cef9a390952eaa7eac189a4791' into tenant 'guanghuipan' with ID 'a87ad796-8231-4e7b-8bd0-d4b362f61b65'.

Declare the device (Each device only needs to be declared once. For more information, please refer to:

https://docs.microsoft.com/en-us/azure-sphere/install/claim-device?tabs=cliv2beta.)

G:\Guardian III>azsphere login Login successful as 'georghuinen@gu Selected Azure Sphere tenant is 'ge (a87ad796-8231-4e7b-8bd0-d4b362f61b65) G:\Guardian III>azsphere device claim warn: You are claiming a device to the following tenant: warn: a87ad796-8231-4e7b-8bd0-d4b362f61b65 warn: asraurge-zz31-gerb-sbdu-debszifulbes warn: Do you want to claim the device ID afca466b32e2142ea450003531a66fd43c60301a8fc2e24ca66fb07e1b16080c83f7a8e3b7a6ea1 a3526187f111a8bdff2bee9cef9a390952ea47eac189a4791 to this tenant? warn: You cannot change the Azure Sphere tenant this device is claimed to once this action has completed. Enter 'yes' to continue. Enter anything else to exit. laiming device. claiming uevice. Successfully claimed device ID 'afca466b32e2142ea450003531a66fd43c60301a8fc2e24ca66fb07e1b16080c83f7a8e3b7a6ea1a3526187: 111a8bdff2bee9cef9a390952eaa7eac189a4791' into tenant 'guanghuipan' with ID 'a87ad796-8231-4e7b-8bd0-d4b362f61b65'. G:\Guardian III> Enable the developer mode of the device G:\Guardian III>azsphere device enable-development warn: The device already has the 'Enable App development' capability. No changes will be applied to its existing capabil ities.

Arges. Application updates have already been disabled for this device. Installing debugging server to device. Deploying 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' to the attached device. Image package 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' has been deployed to the attached device.

Previous device capabilities retained. Ensure that you have the correct development capabilities installed before contin uing.

. Successfully set up device for application development, and disabled application updates. (Device ID: 'afca466b32e2142ea450003531a66fd43c60301a8fc2e24ca66fb07e1b16080c83f7a8e3b7a6ea1a3526187f111a8bdff2bee9cef9a 390952eaa7eac189a4791')

G:\Guardian III>

Use Visual Studio to open the sample code.

Open recent					×	Get sta	inted
→ 👻 🛧 📙 « Guardian	2 > Guardian3D1_Demo	ڻ ~	Search Guardian	3D1_Demo	٩	*	Clone a repository Get code from an online repository like GitHub or
Irganize 🔻 New folder				122 •	0		Peace berops
ame .vs	Date modified 2021/7/23 17:26	Type File folder	Size			Ċ	Open a project or solution Open a local Visual Studio project or .sln file
certs include	2021/7/23 17:26 2021/7/23 17:26 2021/7/23 17:26	File folder File folder File folder				2	Open a local folder Navigate and edit code within any folder
media out sample	2021/7/23 17:26 2021/7/23 17:26 2021/7/23 17:26 2021/7/23 17:26	File folder File folder File folder				*3	Create a new project Choose a project template with code scaffolding
							to get started Continue without code →
Folder Gua	rdian3D1 Demo						

• Compile the sample code.

M Build All Christians	0 - 0 ×
	ADMIN Column Columnia
Gurding Andrew Construction Construction Attribution Attribution	
1 /* Copyright (c) 2020	
2 Build GDB Debugger (HLCore) Chri+B	Search Solution Explorer - Folder View (Cth+)
<pre></pre>	Tenth Statem Faith (1992) T

• Run the sample code.

A De Data Marco de Barlas Build Balans Test Analass Tests Descritors Minders Hale Descritoridad de O Constantint Descri	A B V
PO HE LOT VIEW OR PICKET BUILD DEDUG HER ANBYZE TOOLS EXTERIORS WINDOW HER SAICH (CIT-U) P GUINOMASUL/DEMO	U - D X
3 G • G 1 G • G 1 G • G 1 H 3	ADMIN
sample_cellular.c app_manifest.json sample_cellular.h CMakeLists.tst moinc = X	Solution Explorer - Folder View
Si Guardian2_Demo.out - ARM-Debug - (Global Scope) -	000052600-
1 /* Copyright (c) 2020 CRL. All rights reserved. */	Search Solution Ecolorer - Folder View (Cbrl+:)
<pre>STRICLds corrents statute tight.b statute comparison statute comp</pre>	<pre>Line Constraints and the Line Constraints (Constraints) (Constraint</pre>
101% • € No issues found ↓ ↓ Ln: 1 On: 1 MXED CRUE	
Output	
Show output from: CMake - 😥 🖕 🛬 🖆 🎬 📭	
 (a) Observation of the second s	

When the following log print is entered in the log output window, it indicates that the sample code is running normally.

Output				
Show output from:	Device Output	•		5
Remote debugging [2021-07-23 10:03 [2021-07-23 10:03 [2021-07-23 10:03 [2021-07-23 10:03 [2021-07-23 10:03	<pre>from host 192.168.35.1, port 65203 (47][INFO][main:20]EG1000 Application startin (47][DEBUG][log_debug_device_message:117]This (47][INFO][init_g3_led:932]Open APP LED, state (47][INFO][init_g3_led:959]Set NET LED Off</pre>	g. is a Guardian3D :off :off	2 device	that uses the central Cloud.

3 Sample and Test

In order to demonstrate the use of Guardian III series products, we provide basic sample codes and test methods for each product.

3.1 GIIICM680L

The sample code of GIIICM680L includes LEDs, LoRa, UART demo. Package list: FreeRTOS_RTcore_LoRa_signed.imagepackage

lan-enc28j60-isu0-int5_signed.imagepackage GIIICM680L_Demo.zip

3.1.1 Run the sample code

1. Equipment Recover.

azsphere device recover

G:\Guardian III)azsphere device recover
Downlading recovery images
Starting denies recovery. Place note that this may take up to 10 minutes
Dear tring device recovery. Trease note that this hay take up to rominates.
Bracing flack
Erasing Hash.
Senating 10 images. (3402390 bytes to sena)
Sent 1 of 10 images. (5452512 of 5482530 bytes remaining)
Sent 2 of 10 images. (5330428 of 5482390 bytes remaining)
Sent 3 of 16 images. (5336036 of 5482396 bytes remaining)
Sent 4 of 16 images. (5048636 of 5482396 bytes remaining)
Sent 5 of 16 images. (5032408 of 5482396 bytes remaining)
Sent 6 of 16 images. (5000844 of 5482396 bytes remaining)
Sent 7 of 16 images. (2458896 of 5482396 bytes remaining)
Sent 8 of 16 images. (840740 of 5482396 bytes remaining)
Sent 9 of 16 images. (816164 of 5482396 bytes remaining)
Sent 10 of 16 images. (701264 of 5482396 bytes remaining)
Sent 11 of 16 images. (152180 of 5482396 bytes remaining)
Sent 12 of 16 images. (78240 of 5482396 bytes remaining)
Sent 13 of 16 images. (41164 of 5482396 bytes remaining)
Sent 14 of 16 images. (32768 of 5482396 bytes remaining)
Sent 15 of 16 images. (16384 of 5482396 bytes remaining)
Sent 16 of 16 images. (0 of 5482396 bytes remaining)
Finished writing images, rebooting board.
Device ID: 0668137ae1fd6def5800e5827113d45bb1bb6f5f42d2d76f9ce5eabdacd6305d2142f9086d2d677a623192e1f33c0e6771292c8e0fa74
a8970c3ed6bad57d30f
Device recovered successfully.

2. Enable the developer mode of the device

G:\Guardian III>azsphere device enable-development warn: The device already has the 'Enable App development' capability. No changes will be applied to its existing capabil ities. Application updates have already been disabled for this device. Installing debugging server to device. Deploying 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' to the attached device. Image package 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' has been deployed to the attached device. Previous device capabilities retained. Ensure that you have the correct development capabilities installed before contin uing. Successfully set up device for application development, and disabled application updates. (Device ID: 'afca466b32e2142ea450003531a66fd43c60301a8fc2e24ca66fb07e1b16080c83f7a8e3b7a6ea1a3526187f111a8bdff2bee9cef9a 390952eaa7eac189a4791') G:\Guardian III>

3. Download the internal network card driver package;

azsphere device sideload deploy --imagepackage lan-enc28j60-isu0-int5 signed.imagepackage

:\Guardian III>azsphere device sideload deploy --imagepackage lan-enç22j60-isu0-int5_signed.imagepackage eploying 'G:\Guardian III\lan-enc22j60-isu0-int5_signed.imagepackage' to the attached device. mage package 'G:\Guardian III\lan-enc22j60-isu0-int5_signed.imagepackage' has been deployed to the attached device. he device is rebooting.

4. Download the LoRa driver package;

azsphere device sideload deploy --imagepackage FreeRTOS_RTcore_LoRa_signed.imagepackage G:\Guardian III>azsphere device sideload deploy --imagepackage FreeRTOS_RTcore_LoRa_signed.imagepackage Deploying 'G:\Guardian III\FreeRTOS_RTcore_LoRa_signed.imagepackage' to the attached device. Image package 'G:\Guardian III\FreeRTOS_RTcore_LoRa_signed.imagepackage' has been deployed to the attached device.

5. Run the sample code "GIIICM680L_Demo.zip"; (Refer to section 2.4)

3.1.2 APP LEDs

After the software package is installed, restart the device, and APP LED1 and APP LED2 will flash alternately in accordance with "green" -> "red" -> "off". as figure 23 show.



Figure 1: LED status

3.1.3 Test Lora

In order to complete the Lora test, we need to introduce the third LoRa test module, which mainly completes the conversion of LoRa data and USB serial port data in order to complete the data transmission and reception test. The test module needs to configure the following parameters:

frequency:	910.3M
bandwidth:	125K
spreading factor:	7
coding rate:	4/5
CRC enabled:	true
CRC enabled: preamble length:	true 8bytes
CRC enabled: preamble length: power:	true 8bytes 20dbm

After using the serial port tool putty to connect the test module, the string sent by putty is sent to the device, and the device will return the data as it is after receiving the data, so you can see in putty that you are sending data.

3.1.4 Test UART

Use Type-B to connect the PC and the device. After the connection is complete, open the serial port through the serial port tool putty.

Type-B cable:



Figure 2: Type-B cable

Serial port connection parameters:

Speed(baud):115200bpsData bits:8

Stop bits:	1
Parity:	None
Flow control:	None

After connecting the device with the serial port tool putty, the string sent by putty is sent to the device, and the device will return the data as it is after receiving the data, so you can see in putty that you are sending data.

Putty		×
G3 TEST.		\sim

Figure 3: putty output information

3.2 GIIICM680E

The sample code of GIIICM680E includes LEDs, W5500, UART demo.

Package list:

SouthBound_Ethernet_signed.imagepackage lan-enc28j60-isu0-int5_signed.imagepackage GIIICM680E_Demo.zip

3.2.1 Run the sample code

If you get a box with sample code loaded, do the following to test it:

1. Connect the debug dongle and power on the box, from the powershell of windows, check images installed, it should be like this:

PS C:\Users\richa> a:	zsphere device image list-installed
Installed images:	
> lan-enc28.j60-isu	10-int5
> Image type:	Board configuration
> Component ID:	91ecec4e-7e7e-40da-bb38-01f4d10f3f45
> Image ID:	c4402be3-d851-4035-842d-1e1a4177f944
> gdbserver	
> Image type:	Application
> Component ID:	8548b129-b16f-4f84-8dbe-d2c847862e78
> Image ID:	4d151b4a-a1e6-4530-8347-38468df88834
> SouthboundEther	net
> Image type:	Application
> Component ID:	005180bc-402f-4cb3-a662-72937dbcde47
> Image ID:	aeda318f-66a8-4a2b-b534-3e2ae9d64504
> Guardian3RD2	
> Image type:	Application
> Component ID:	b5cd66c6-8b23-43d8-9991-7e5cc1656918
> Image ID:	3d487e63-836c-4c24-aa69-85fcf219ec50

2. Check interface connection:

PS C:\Users\rid	ha> azsphere	device network lis	st-interface	es			
InterfaceName	InterfaceUp	ConnectedToNetwork	IpAcquired	IpAddresses	ConnectedToInternet	IpAssignment	HardwareAddress
azspheresvc	True	False	False	192.168.35.2	False	None	None
eth0	True	True	True	192.168.86.25	True	dynamic	f6:4e:06:b3:16:83
lo	True	False	False	127.0.0.1	False	None	None
wlanØ	False	False	False	None	False	dynamic	60:1d:9d:f9:46:a6

3. Use the mobile App to watch southbound connection status. Refer 3.2.3 for the test setup.

If you get a brand-new box without code loaded, do the following to install test samples:

1. Equipment Recover.

azsphere device recover

G:\Guardian III>azsphere device recover
Downloading recovery images
Download complete.
Starting device recovery. Please note that this may take up to 10 minutes.
Board found. Sending recovery bootloader.
Erasing flash.
Sending 16 images, (5482396 bytes to send)
Sent 1 of 16 images. (5452312 of 5482396 bytes remaining)
Sent 2 of 16 images. (5336428 of 5482396 bytes remaining)
Sent 3 of 16 images. (5336036 of 5482396 bytes remaining)
Sent 4 of 16 images. (5048636 of 5482396 bytes remaining)
Sent 5 of 16 images. (5032408 of 5482396 bytes remaining)
Sent 6 of 16 images. (5001844 of 5482396 bytes remaining)
Sent 7 of 16 images. (2458896 of 5482396 bytes remaining)
Sent 8 of 16 images. (840740 of 5482396 bytes remaining)
Sent 9 of 16 images. (816164 of 5482396 bytes remaining)
Sent 10 of 16 images. (701264 of 5482396 bytes remaining)
Sent 11 of 16 images. (152180 of 5482396 bytes remaining)
Sent 12 of 16 images. (78240 of 5482396 bytes remaining)
Sent 13 of 16 images. (41164 of 5482396 bytes remaining)
Sent 14 of 16 images. (32768 of 5482396 bytes remaining)
Sent 15 of 16 images. (16384 of 5482396 bytes remaining)
Sent 16 of 16 images. (0 of 5482396 bytes remaining)
Finished writing images; rebooting board.
Device ID: 0668137ae1fd6def5800e5827113d45bb1bb6f5f42d2d76f9ce5eabdacd6305d2142f9086d2d677a623192e1f33c0e6771292c8e0fa7
a 8970 a 3 a d6 ha d57 d 30 f

2. Enable the developer mode of the device

```
G:\Guardian III>azsphere device enable-development
warn: The device already has the 'Enable App development' capability. No changes will be applied to its existing capabil
ities.
Application updates have already been disabled for this device.
Installing debugging server to device.
Deploying 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' to the attached device.
Image package 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' has been deployed to
the attached device.
Previous device capabilities retained. Ensure that you have the correct development capabilities installed before contir
uing.
Successfully set up device for application development, and disabled application updates.
(Device ID: 'atca466b32e2142ea450003531a66fd43c60301a8fc2e24ca66fb07e1b16080c83f7a8e3b7a6ea1a3526187f111a8bdff2bee9cef9a
390952eaa7eac189a4791')
G:\Guardian III>
```

3. Download the internal network card driver package.

azsphere device sideload deploy --imagepackage lan-enc28j60-isu0-int5 signed.imagepackage

G:\Guardian III>azsphere device sideload deploy --imagepackage lan-enc28j60-isu0-int5_signed.imagepackage Deploying 'G:\Guardian III\lan-enc28j60-isu0-int5_signed.imagepackage' to the attached device. Image package 'G:\Guardian III\lan-enc28j60-isu0-int5_signed.imagepackage' has been deployed to the attached device. The device is rebooting.

4. Download the W5500 driver package.

azsphere device sideload deploy --imagepackage SouthBound_Ethernet_signed.imagepackage G:\Guardian III>azsphere device sideload deploy --imagepackage SouthBound_Ethernet_signed.imagepackage Jeploying 'G:\Guardian III\SouthBound_Ethernet_signed.imagepackage' to the attached device. Image package 'G:\Guardian III\SouthBound_Ethernet_signed.imagepackage' has been deployed to the attached device.

5. Run the sample code "GIIICM680E_Demo.zip"; (Refer to section 2.4)

3.2.2 APP LEDs

After the software package is installed, restart the device, and APP LED1 and APP LED2 will flash alternately in accordance with "green" -> "red" -> "off". as figure 26 show.



Figure 4: LED status

3.2.3 Test Southbound Ethernet (W5500)

Use the RJ-45 network cable to connect the device and the PC and set the PC to DHCP client mode. After the connection is completed, W5500 will assign a 192.168.50.X sub-network IP to the PC, and the default gateway address is 192.168.50.1. As figure 27 show.

Ethernet adapter Eth2:		
Connection-specific DNS Suffix Link-local IPv6 Address IPv4 Address Subnet Mask Default Gateway	. : . : fe80::d8b8:57ce:6e38:e0c9%61 . : 192.168.50.2 . : 255.255.255.0 . : 192.168.50.1	

Figure 5: Get Client IP

The W5500 internal test module starts TCP port 5000 at 192.168.50.1 by default and uses putty to connect.

Session	Basic options for your PuTTY session						
- Logging Terminal Keyboard	Specify the destination you want to conn Host <u>Name (or IP address)</u>	ect to Port					
Bell	Bell [192.168.50.1]						
Window	OSH OSerial Other: Telr	net 🗸					
Behaviour Translation Selection Colours Connection	Load, save or delete a stored session Sav <u>e</u> d Sessions Default Settings	Load					
Data Proxy	COM73	Sa <u>v</u> e					
⊡ SSH Serial Telnet		<u>D</u> elete					
Rlogin SUPDUP	Close window on exit: Always Never Only on clean exit						

Figure 6: putty configuration parameters

Any data received by the W5500 internal test module will be returned as it is.



Figure 7: putty output information

3.2.4 Test UART

Use Type-B to connect the PC and the device. After the connection is complete, open the serial port through the serial port tool putty.

Type-B cable:



Figure 8: Type-B cable

Serial port connection parameters:

Speed(baud):	115200
Data bits:	8
Stop bits:	1
Parity:	None
Flow control:	None

After connecting the device with the serial port tool putty, the string sent by putty is sent to the device, and the device will return the data as it is after receiving the data, so you can see in putty that you are sending data.





Figure 9: putty output information

3.3 GIIICM680R

The sample code of GIIICM680R includes LEDs, CAN, RS422, RS485 demonstration. Package list:

lan-enc28j60-isu0-int5_signed.imagepackage GIIICM680R Demo.zip

3.3.1 Run the sample code

1. Equipment Recover.

```
azsphere device recover

3: Chuardian III Jazsphere device recover

Downloading recovery images...

Download complete.

Starting device recovery. Please note that this may take up to 10 minutes.

Board found. Sending recovery bootloader.

Erasing flash.

Sending 16 images. (542296 bytes to send)

Sent 1 of 16 images. (542302 of 542296 bytes remaining)

Sent 2 of 16 images. (5336036 of 542296 bytes remaining)

Sent 3 of 16 images. (5038636 of 542296 bytes remaining)

Sent 4 of 16 images. (5032408 of 542296 bytes remaining)

Sent 5 of 16 images. (502408 of 542296 bytes remaining)

Sent 5 of 16 images. (502408 of 542296 bytes remaining)

Sent 5 of 16 images. (502408 of 542296 bytes remaining)

Sent 7 of 16 images. (501844 of 542296 bytes remaining)

Sent 8 of 16 images. (501844 of 542296 bytes remaining)

Sent 9 of 16 images. (14588096 of 5422396 bytes remaining)

Sent 9 of 16 images. (14582096 bytes remaining)

Sent 10 of 16 images. (15180 of 5422396 bytes remaining)

Sent 10 of 16 images. (152180 of 5422396 bytes remaining)

Sent 11 of 16 images. (770264 of 5422396 bytes remaining)

Sent 12 of 16 images. (770264 of 5422396 bytes remaining)

Sent 13 of 16 images. (152180 of 5422396 bytes remaining)

Sent 13 of 16 images. (16384 of 5422396 bytes remaining)

Sent 14 of 16 images. (16384 of 5422396 bytes remaining)

Sent 15 of 16 images. (0 of 5482396 bytes remaining)

Sent 15 of 16 images. (0 of 5482396 bytes remaining)

Sent 15 of 16 images. (0 of 5482396 bytes remaining)

Sent 15 of 16 images. (0 of 5482396 bytes remaining)

Sent 15 of 16 images. (0 of 5482396 bytes remaining)

Sent 16 if 16 images. (0 of 5482396 bytes remaining)

Sent 17 of 16 images. (0 of 5482396 bytes remaining)

Sent 18 of 16 images. (0 of 5482396 bytes remaining)

Sent 16 of 16 images. (0 of 5482396 bytes remaining)

Sent 17 of 16 images. (0 of 5482396 bytes remaining)

Sent 16 of 16 images. (0 of 5482396 bytes remaining)

Finished writing images, rebooting board.

Device recovered successfully.
```

2. Enable the developer mode of the device

G:\Guardian III>azsphere device enable-development warn: The device already has the 'Enable App development' capability. No changes will be applied to its existing capabil ities. Application updates have already been disabled for this device. Installing debugging server to device. Deploying 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' to the attached device. Image package 'C:\Program Files (x86)\Microsoft Azure Sphere SDK\DebugTools\gdbserver.imagepackage' has been deployed to the attached device. Previous device capabilities retained. Ensure that you have the correct development capabilities installed before contin uing. Successfully set up device for application development, and disabled application updates. (Device ID: 'afca406b32e2142ea450003531a66fd43c60301a8fc2e24ca66fb07e1b16080c83f7a8e3b7a6ea1a3526187f111a8bdff2bee9cef9a 390952eaa7eac189a4791') G:\Guardian III>

3. Download the internal network card driver package.

azsphere device sideload deploy --imagepackage lan-enc28j60-isu0-int5 signed.imagepackage

:\Guardian III>azsphere device sideload deploy --imagepackage 1an-enc28j60-isu0-int5_signed.imagepackage leploying 'G:\Guardian III\lan-enc28j60-isu0-int5_signed.imagepackage' to the attached device. mage package 'G:\Guardian III\lan-enc28j60-isu0-int5_signed.imagepackage' has been deployed to the attached device. he device is rebooting.

Run the sample code "GIIICM680R_Demo.zip". Refer 2.4 for details.

3.3.2 APP LEDs

After the software package is installed, restart the device, and APP LED1 and APP LED2 will flash alternately in accordance with "green" -> "red" -> "off". as figure 32 show.



Figure 10: LED status

3.3.3 Test CAN

Use a third-party CAN test tool to connect the device. The CAN test module used in this example is a CAN module produced by China Chuangxin Technology.



Figure 11: CAN test module Appearance



Figure 12: Device GPIOs

Use CAN connect the device, connect the wire sequence GPIO7---CHNH, GPIO8---CHNL.



Figure 13: CAN connection diagram

After the connection is complete, use the test tool provided with the CAN test module to turn on the device.

USB-CAI	N Tool V9.1	1 The devi	ice does no	t start!							-		×
Device(D)	Operation(C) Settings(S) Informat	tion(<u>I</u>)	View(<u>V</u>) I	Help(<u>H</u>) L	anguage(L)					
Send Da	Stop(T)				00 00 03 7	8 Channel	. 1 🗸	Number	to sen	ł: 1		ID II	10.
Data(HEX)	Register	State(<u>R</u>)					l	ii Se	and Cycl	e: 1	ms	Data	Inc.
CAN Rout	Baudrate Routing (Pro fault USBCAN	Detect(<u>B)</u> Option(<u>O</u>) tolerant can c Test Tool(<u>T</u>)	onfiguratio	n(P)	ettings	Frm	saved: 0 Receive 1	Enable		Stop Cle	send ar	Send fi □Save	1e
Statistic Frm/s R: C	s:Chi D F	rm/s T: 0				Statist Frm/s R:	ics:Ch2	Frm/s	T: 0				
Index Sy	rstem Time	Time Stamp	Channel	Directio	Frame ID	Туре	Format	DLC	Data				
<													,

Figure 14: Open device

Can Open Us	SB Device - CHUANGXIN	Technology	×
)evice Name:	Device Index Number O:Se	rial number: 01900202A91,	~
	ОК	Cancel	

Figure 15: Device connection parameters

vice index: 0	\sim	Channel:	Chi 🗸
CAN Setting			Madrealite
Baud Rate: 500k	bps 🗸 I	3TR0/1:00	1C (HEX)
AccCode: 0x00000	000	Filter: Acce	pt all 🗸
AccMask: 0xFFFFF	FFF F	ilter config	uration tool
Mode Normal		~	

Figure 16: CAN interface parameters

After the test module is successfully connected to the device, send the test data to the device through the tool, and the device will return the test data as it is, as shown in the figure below. as figure 39 show.

Device(D) Operation(<u>C</u>) Settings(S) Inform	ation(<u>l</u>)	View(⊻) He	elp(<u>H</u>) l	Language(<u>l</u>)		- 99-39-37225		
Form	at: Extended 🗸	Type: Dat	a VCAN	ID (HEX) :	00 00 03 78	Channel	l: 1 🗸	Number	to send:	1		D Inc.
Data(HE)	(): 10 00 00 00	00 00 00 ff	Send					Se	nd Cycle	1	ms)ata Inc.
CAN Rot	iting	ID Fil	ter			Fra	saved: 0			Stop se	nd Sen	d file
	Unused	CAN1	settings	CAN2 s	ettings	Ε	Receive F	nable		Clear		Save
Statis Res (c. R	tics:Chi	- (- T- 0.2	_			Statist R (- R-	tics:Ch2	P (T: 0			
rrm/s n	C	T' China	churry 1	Discotion	V TD	Trm/S A.		Prm/s	I. U.			
naex	14.40.17 862	-	chi	Sand	0v00000378	Doto	Format	0×08			47 00 00 1	es.
00001	14:40:17.863	OxC2EA63	chl	Receive	0x00000378	Data	Extende	0x08	x 10 00	00 00 00) 00 D0 Ff	
												>

Figure 17: Data sending and receiving

3.3.4 Test RS422

The RS422 interface test needs to use a standard DB9 serial debugging line. In the example, the USB2.0 RS422/RS485 universal serial debugging line produced by China Z-TEK is used.



Figure 18: RS422/RS485 cable

Device and DB9 connection line sequence:

GIIICM680L	DB9
GPIO4(RXD-)	 T/R-
GPIO5(RXD+)	 T/R+
GPIO10(TXD+)	 RXD+
GPIO11(TXD-)	 RXD-

Connection example diagram:



Figure 19: RS422 connection diagram

Connect the DB9 interface through putty, the baud rate is 115200bps, send data to the device, the device will return the data as it is. as figure 42 show.



Figure 20: putty output information

3.3.5 Test RS485

The RS422 interface test needs to use a standard DB9 serial debugging line. In the example, the USB2.0 RS422/RS485 universal serial debugging line produced by China Z-TEK is used.



Figure 21: RS422/RS485 cable

Device and DB9 connection line sequence:

GIIICM680L		DB9
GPIO4		T/R-
GPIO5		T/R+
GPIO10		T/R+
GPIO11		T/R-

Connect the DB9 interface through putty, the baud rate is 115200bps, send data to the device, the device will return the data as it is. as figure 44 show.



Figure 22: putty output information

٠

4 Application API

4.1 Common API

Common API are the API functions can be called by all Guardian 3 products.

4.1.1 gdn_sys_init()

int gdd_sys_init (int argc,

char * argv[]

)

Init the Guardian system hardware, it should be called before any other API.

Parameters

argc

argv

Returns

int 0 on success, -1 otherwise

4.1.2 gdn_daughter_board_power_down()

int gdn_daughter_board_power_down (void)
Power down daughter board to save power

Returns

int 0 on success, -1 otherwise

4.1.3 gdn_daughter_board_power_up()

int gdn_daughter_board_power_up (void) Power up the daughter board to use its function **Returns**

int 0 on success, -1 otherwise

4.1.4 gdn_change_network_type()

int gdn_change_network_type (int networkType)

set network type and check

Parameters

networkType enum e_network_type

Returns

int 0 on success, -1 otherwisw

4.1.5 gdn_check_current_wifi_network_status()

int gdn_check_current_wifi_network_status (void)
check current wifi status
Returns
int 0 on success, -1 otherwisw

4.1.6 gdn_check_network_ready()

int gdn_check_network_ready (void)
check network ready state
Returns
int 0 Internet available, 1 Internet not available

4.1.7 gdn_get_network_type()

e_network_type gdn_get_network_type (void)
get network type

Returns

e_network_type enum e_network_type

4.1.8 gdn_init_led()

int gdn_init_led (void) init LED Returns int 0 on success, -1 otherwise

4.1.9 gdn_set_led_status()

void gdn_set_led_status (int type, unsigned char status)

Set LED status

Parameters

type led possition state 0 off, 1 on

4.1.10 gdn set network by type()

void gdn_set_network_by_type (int networkType)
set network type

Parameters

networkType enum networkType

4.1.11 gdn_watch_dog_feed()

void gdn_watch_dog_feed (uint32_t watchedIndex)
watch dog feed
Parameters

watchedIndex watch dog index

4.1.12 gdn_watch_dog_init()

int gdn_watch_dog_init (uint32_t watchedNum)

watch dog init

Parameters

watchedNum watch dog id

Returns

int 0 on success, -1 otherwise

4.2 Cellular related API

Cellular related API are API functions can be called by Guardian 3 with cellular modules.

4.2.1 gdn_init_cellular()

int gdn_init_cellular (void)

This function initiates the communication between MT3620 with the cellular module. The application should call this before any other cellular related functions. It should be called after the system module's initialization.

Parameters

None

Returns

int 0 on success, -1 otherwise

4.2.2 gdn_config_cellular()

int gdn_set_cellular_config (CELLULAR_CONFIG_S * cellular_config)

Configure Cellular properties: APN, username, password, pin number.

Parameters

cellular_config Attribute fields that need to be configured

Returns

int 0 on success, -1 otherwise

4.2.3 gdn_get_esim_id()

int gdn_get_esim_id (ESIM_ID_S * esim_id)

get esim_id:IMI,QCCID

Parameters

esim_id The returned esim_id structure requires the user to allocate memory

Returns

int 0 on success, -1 otherwise

4.2.4 gdn_get_gps()

int gdn_get_gps (GPS_INFO_S * gps)

get gps info:utc,longitude,latitude,altitude

Parameters

gps The returned gps structure requires the user to allocate memory

Returns

int 0 on success, -1 otherwise

4.2.5 gdn handle atcmd()

int gdn handle atcmd (char * atcmd,

> unsigned int timeout_ms, char ** at_result, unsigned int * result len

)

AT command transparent transmission to obtain corresponding response data.

Parameters

AT command sent atcmd

timeout ms Timeout waiting for reply

Returned data. The user does not need to allocate memory, but the user needs to release it after at result use

The length of the returned data result len

Returns

int 0 on success, -1 otherwise

4.2.6 gdn get cellular status()

int gdn_get_cellular_status (CELLULAR_STATUS_S * cellular_status)

get cellular status:state,signal-quality,access tech,imei,registration-state,operator-code,operator-name Parameters

cellular status The returned cellular_status structure requires the user to allocate memory

Returns

int 0 on success, -1 otherwise

4.2.7 gdn cellular connect state()

bool gdn_cellular_connect_state (void)

get nxp connect state

Returns

true connected false not connected

4.2.8 gdn_get_wwan_info()

int gdn_get_wwan_info (WWAN_INFO_S * wwan_info)

get wwan infomation: connected,suspended,wwan_if,ip_timeout,apn,roaming,method,address,prefix,prefix,dns,mtu **Parameters**

wwan_info The returned wwan_info structure requires the user to allocate memory

Returns

int 0 on success, -1 otherwise

4.2.9 gdn_print_cellular_debug_info()

void gdn print cellular debug info (void)

One time to print detailed information such as the complete status and attributes of the cellular.

4.3 **GIIICM680L**

Products: GIIICM680L, GIIICM680L-CM2, GIIICM680L-CM6, GIIICM680L-CMA.

4.3.1 gdn_close_usb_uart()

void gdn_close_usb_uart (void)

gdn_close_usb_uart - close usb to uart by params including config and data handler, This is used for the FT234XD

4.3.2 gdn_open_usb_uart()

int gdn_open_usb_uart (UART_Config * config,

UART_DATA_HANDLER handler

)

open usb to uart by params including config and data handler, This is used for the FT234XD

Parameters

config Defined by UART_Config

handler Callback defined by UART_DATA_HANDLER, if set NULL, using built-in handler

Returns

int 0 on success, -1 otherwisw

4.3.3 gdn_send_message_to_usb()

size_t gdn_send_message_to_usb (uint8_t * data,

size t dataLen

)

send data to usb

Parameters

data data - Data buffer to send

dataLen dataLen - Length of data to send

Returns

size_t Bytes of sent data

4.3.4 gdn_lora_recv()

void gdn_lora_recv (char * buff

int buff len

)

LoRa devices' data receiver

Parameters

buff receive data buff buff len receive data buff len

4.4 GIIICM680E

Products: GIIICM680E, GIIICM680E-CN, GIIICM680E-CE

4.4.1 gdn_close_usb_uart()

void gdn_close_usb_uart (void) gdn_close_usb_uart - close usb to uart by params including config and data handler, This is used for the FT234XD

4.4.2 gdn_open_usb_uart()

int gdn_open_usb_uart (UART_Config * config,

UART_DATA_HANDLER handler

)

open usb to uart by params including config and data handler, This is used for the FT234XD

Parameters

config Defined by UART_Config

handler Callback defined by UART_DATA_HANDLER, if set NULL, using built-in handler

Returns

int 0 on success, -1 otherwisw

4.4.3 gdn_tcp_client_connect()

int gdn_tcp_client_connect (int * socket_fd,

const char * ip,

unsigned short port

)

tcp connect

Parameters

socket_fd socket fd

ip ip address

port port number

Returns

int 0 on success, -1 otherwisw

4.4.4 gdn_tcp_client_recv()

int gdn_tcp_client_recv (int *	socket_fd,
	char **	recv_buf,
	unsigned int *	buff_len,
	unsigned int	timeout_second
)		
tcp receive		
Parameters		
socket_fd	socket fd	

recv_buf receive data buff

buff_len receive data buff len

timeout_second time out

Returns

int reality receive data len

4.4.5 gdn_tcp_client_send()

int gdn_tcp_client_send (int * socket_fd, char * buff, unsigned int len) tcp send Parameters socket_fd socket fd buff send data buff len send data buff len

Returns

int reality send data len

4.4.6 gdn_southboundEth_recv()

void gdn_southboundeth_recv (char * buff int buff_len unsigned short port) southbound ethernet data receiver Parameters buff receive data buff buff_len receive data buff len port port number

4.4.7 gdn_southboundEth_send()

void gdn_southboundeth_send (char * buff size_t buff_len unsigned short port) southbound ethernet send

Parameters

buff	send data buff
buff_len	send data buff len
port	port number

4.5 GIIICM680R

Products: GIIICM680R, GIIICM680R-CN, GIIICM680R-CE

4.5.1 gdn_close_can()

void gdn_close_can (int fd)

Parameters

fd CAN fd Indicates which CAN to close

4.5.2 gdn_close_rs422()

void gdn_close_rs422 (void)
close rs422 to uart by params including config and data handler.

4.5.3 gdn_close_rs485()

void gdn_close_rs485 (void)
close rs485 by params including config and data handler.

4.5.4 gdn_open_can()

int gdn_open_can (eInterfaceType interface, eCANPort port, CANParameter_t * parameter_can, CAN_DATA_HANDLER handler)

Open CAN bus by params, baudrate and data handler...

Parameters

interface Use UART or SPI interface to connect to CAN module

port Which channel CAN open, G3 only has port0 by default

parameter_can struct of CANParameter_t

handler Callback defined by CAN_DATA_HANDLER,Used by the user to process the received CAN data

Returns

int 0 on success, -1 otherwise

4.5.5 gdn_open_rs422()

int gdn_open_rs422 (UART_Config * config,

UART_DATA_HANDLER handler

)

open usb to uart by params including config and data handler.

Parameters

config Defined by UART_Config

handler Callback defined by UART_DATA_HANDLER, if set NULL, using built-in handler

Returns

int 0 on success, -1 otherwisw

4.5.6 gdn_open_rs485()

int gdn_open_rs485 (UART_Config * config,

UART_DATA_HANDLER handler

)

open rs485 by params including config and data handler.

Parameters

config Defined by UART_Config

handler Callback defined by UART_DATA_HANDLER, if set NULL, using built-in handler

Returns

int 0 on success, -1 otherwisw

4.5.7 gdn_send_message_to_rs422()

size t gdn send message to rs422 (uint8 t * data,

size_t dataLen

)

Send data to rs422.

Parameters

data Data buffer to send

dataLen Length of data to send

Returns

size_t Bytes of sent data

4.5.8 gdn_send_message_to_rs485()

size_t gdn_send_message_to_rs485 (uint8_t * data,

size_t dataLen

)

send data to rs485

Parameters

data Data buffer to send

dataLen Length of data to send

Returns

size_t Bytes of sent data

4.5.9 gdn_write_can()

void gdn_write_can (int	fd,
	CANWrite_t *	canPara,
	const char *	dataToSend,
	size_t	dataLen
)		
Write data to the CAN	bus.	

Parameters

fdCAN fd Indicates which CAN to writecanParastruct of CANWrite_tdataToSendData buffer to senddataLenLength of data to send

5 Support

For all general, partnership, career, and/or press inquiries, please contact us through email or phone number:

Email: crl@changhong.us Phone: 1-408-970-0349 Address: 2580 North First Street, Suite 100, San Jose, CA 95131