G-4513 Series

3G Power Saving PAC with Solar Charger

User's Manual V1.0.1



High Quality, Industrial Data Acquisition, and Control Products

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assumes no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Copyright 2014 by ICP DAS CO., LTD. All rights reserved worldwide.

Trademark

The names used for identification only may be registered trademarks of their respective companies.

Contact us

If you have any problem, please feel free to contact us.
You can count on us for quick response.

Email：service@icpdas.com

Table of Contents

1. Introduction 1
2. Hardware Specifications 2
2.1 G-4513-3GWA Series 2
2.2 G-4513-3GWA Series Specifications 3
3. Application Architecture 6
3.1 Hydrologic / Wind Monitoring Application 6
3.2 Car Monitor / Tracking System 6
3.3 Redundancy Communication system 7
4. Hardware 8
4.1 Pin Assignments 8
4.2 Dimensions 9
4.3 Operation Mode Switch 10
4.4 LED Indicators 11
4.5 Wire Connection 12
4.6 Wake up from Sleep Mode 14
4.7 Installation 15
5. Power Saving and Charger 16
5.1 Power Saving 16
5.2 How to Choose the Battery 18
5.3 How to Choose the Solar Panel 21
6. APIs and Demo References 24
6.1 API for Local I/O 24
6.1.1 X305IO_Init 25
6.1.2 X305IO_GetLibVersion 26
6.1.3 X305IO_Read_AD_CalibrationGain 27
6.1.4 X305IO_Read_AD_CalibrationOffset 28
6.1.5 X305IO_AnalogIn 29
6.1.6 X305IO_Read_All_DI 30
6.1.7 X305IO_Read_One_DI 31
6.1.8 X305IO_Write_All_DO 32
6.1.9 X305IO_Write_One_DO 33
6.1.10 X305IO_Read_All_DO 34
6.1.11 X305IO_Read_One_DO 35
6.1.12 X305IO_AnalogIn_SetChannel 36
6.1.13 X305IO_AnalogIn_Hex 37
6.1.14 X305IO_AnalogIn_HexToFloat 38
6.2 API for MMC/SD 39
6.3 API for LCD 44

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

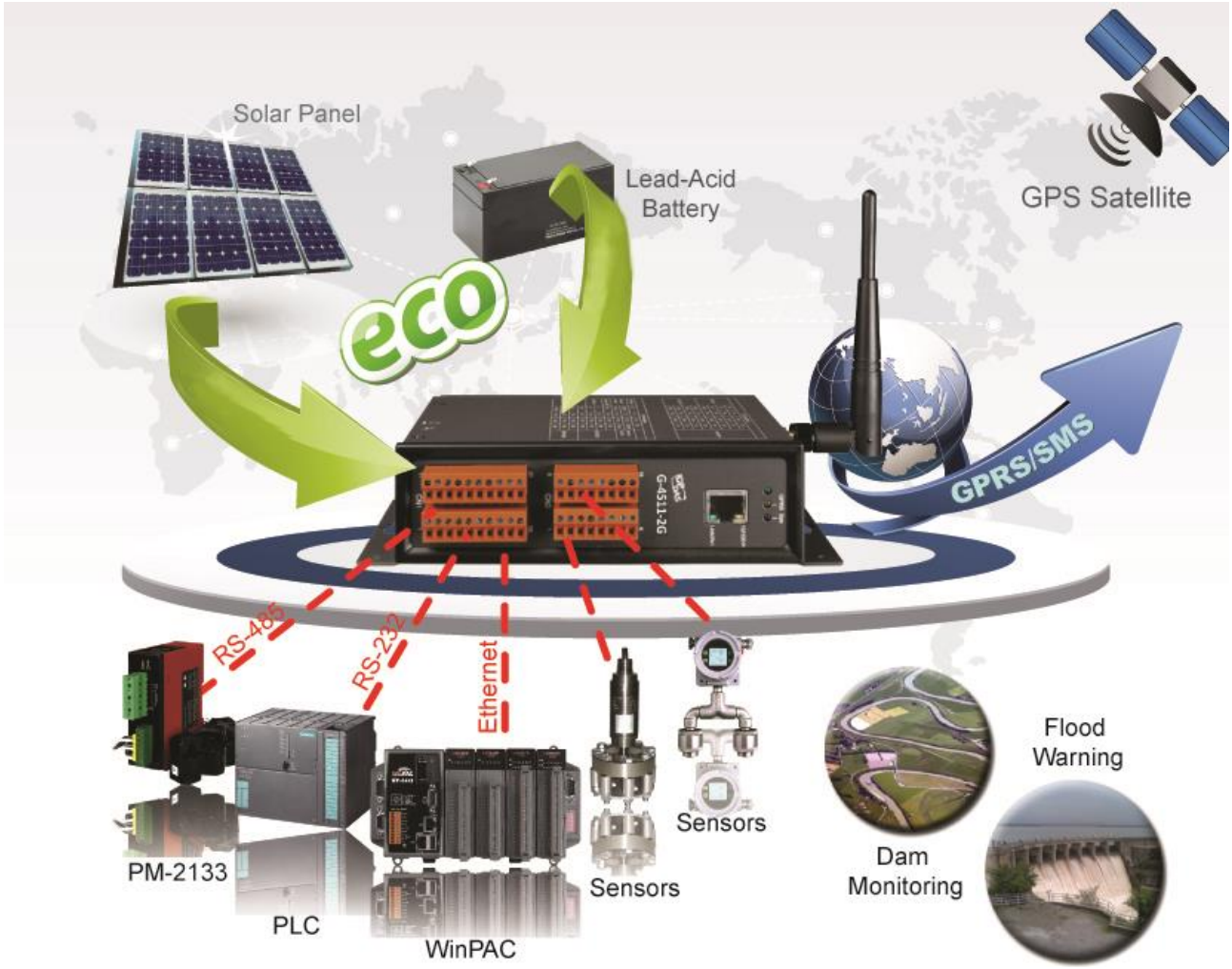
6.3.1 LCD_Init 45
6.3.2 LCD_BackLight_On 46
6.3.3 LCD_BackLight_Off 47
6.3.4 LCD_ShowText 48
6.3.5 LCD_ClrScrn 49
6.3.6 LCD_StandByMode 50
6.3.7 LCD_NormalMode 51
6.3.8 LCD_GotoPosition 52
6.3.9 LCD_CursorDisplay 53
6.3.10 LCD_LineReverse 54
6.3.11 LCD_LineRestore 55
6.3.12 LCD_GetLibDate 56
6.3.13 LCD_GetLibVersion 57
6.4 API for GSM 58
6.4.1 GSM Design Flowchart 59
6.4.2 GM_SYS_InitModem 60
6.4.3 GM_SYS_CloseModem 61
6.4.4 GM_SYS_CheckModemStatus 62
6.4.5 GM_SYS_CheckSignal 63
6.4.6 GM_SYS_CheckReg 64
6.4.7 GM_SMS_SendMsg 65
6.4.8 GM_SMS_GetNewMsg 66
6.4.9 GM_NET_SetNet 67
6.4.10 GM_NET_InstallLink 68
6.4.11 GM_NET_CloseNet 69
6.4.12 GM_NET_CloseLink 70
6.4.13 GM_NET_Send 71
6.4.14 GM_NET_GetNewPacket 72
6.4.15 GSM Demo References 73
7. Program Download Procedure 81
8. Revision History 87

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

1. Introduction

The G-4513 series are M2M (Machine to Machine) Power Saving PAC with a cellular transceiver and Solar Charger. It can be used in hydrologic monitoring or mudslide monitoring system. With optional GPS model, the G-4513 can also be a GPS tracking system for vehicle management system or maritime system.

The features of G-4513 series: Solar Charger, 3G module, Ethernet interface, optional GPS module, 3 digital inputs, 3 digital outputs, 8 analog inputs, 1 relay, 1 RS-232 and 1 RS-485 port. That can be used in various application fields to transfer data over 3G, GPRS, SMS, Ethernet or serial bus. The G-4513 series built-in MiniOS7 provide the same development environment with I-7188/I-7186 series. It is easier for I-7188/I-7186 users to apply the G-4513 series.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2. Hardware Specifications

2.1 G-4513-3GWA Series

G-4513-3GWA	G-4513D-3GWA	G-4513P-3GWA	G-4513PD-3GWA
			

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.2 G-4513-3GWA Series Specifications

Item	G-4513-3GWA	G-4513D-3GWA	G-4513P-3GWA	G-4513PD-3GWA
CPU	80 MHz internal microprocessor			
SRAM/Flash	512K/512K , real time clock, watchdog timer			
NVRAM	31 bytes, battery backup, data valid up to 10 years			
EEPROM	16 KB, retention > 40 years. 1,000,000 erase/write cycles			
Comm. Interface				
COM ports	COM1:5-wire RS-232; COM2: RS-485			
Ethernet	10/100 Base-TX Ethernet controller			
GSM Interface				
Frequency Band	GSM 850/900/1800/1900 MHz			
GPRS connectivity	GPRS class 10/8; GPRS station class B			
SMS	MT, MO, CB, Text and PDU mode			
3G Interface				
Frequency Band	WCDMA 850/900/1900/2100 MHz			
Power Class	Class 3 (250mW @ WCDMA/HSPA)			
Digital Input				
Input Channel	3			
Input Type	Source(Dry Type), Common Ground			
Off Voltage Level	+1 V max.			
On Voltage Level	+3.5 ~ +30 V			
Isolated Voltage	Non-isolated			
Digital Output				
Output Channel	3			
Output Type	3 Open Collector (Sink/NPN)			
Load Voltage	+30 VDC max.			
Load Current	100 mA max.			
Isolated Voltage	Non-isolated			
Analog Input				
Input Channel	8			
Resolution	12 - bit			
Input Range/Type	0 ~ 20 mA			
Sample Rate	1 KHz max. (Read one channel)			
Accuracy	+/- 2 LSB (+/- 0.0097 mA)			
Isolated Voltage	2500Vrms 3000Dc to DC			
Relay				
Output Channel	1			
Type	Form C			

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Input Range		2A@30 Vdc ; 0.25 A @250 Vac			
Mechanical endurance		typ. 10^8 operations			
GPS Interface					
Support Channels		-		32	
Sensitivity		-		Tracking = up to -159 dBm (with external LNA)	
				Cold start = up to -146 dBm (with external LNA)	
Acquisition Time		-		Hot start (Open Sky) = 2 s(typical)	
				Cold start (Open Sky) = 36 s(typical)	
Protocol Support		-		NMEA 0183 version 3.01	
LCD Interface					
General	Effective display area	-	80.61 mm x 14.37 mm (W x H)	-	80.61 mm x 14.37 mm (W x H)
	Module Dimension	-	93 mm x 70 mm x 1.6 mm (W x H x T)	-	93 mm x 70 mm x 1.6 mm (W x H x T)
Life Time		-	Expected life is more than 100,000 hours under normal operation	-	Expected life is more than 100,000 hours under normal operation
Power (Solar Input)					
Protection		Power reverse polarity protection			
Frame Ground Protection		ESD, Surge, EFT, Hi-Pot			
Power Requirement		+10 V_{DC} ~ +30 V_{DC} , (Max. Voltage of Solar Panel must less +30V)			
Power Consumption		Deep Sleep: < 10 mA@12V_{DC}; Deep Sleep(With LCD): < 11 mA@12V_{DC}; Sleep: < 15 mA@12V_{DC}; Idle: 77 mA @ 24 V_{DC}; Data Link: 150 ~ 400 mA (peak) @ 24 V_{DC}			
Lead Acid Battery Requirement					
Battery		12V Lead-Acid Battery			
Charging Voltage		Voltage of Power Input must be over +16V			
Low Voltage Protect		Low Voltage disconnect = 11.1V / Low Voltage reconnect = 12.6V			
LED Indicators					
System		Red			

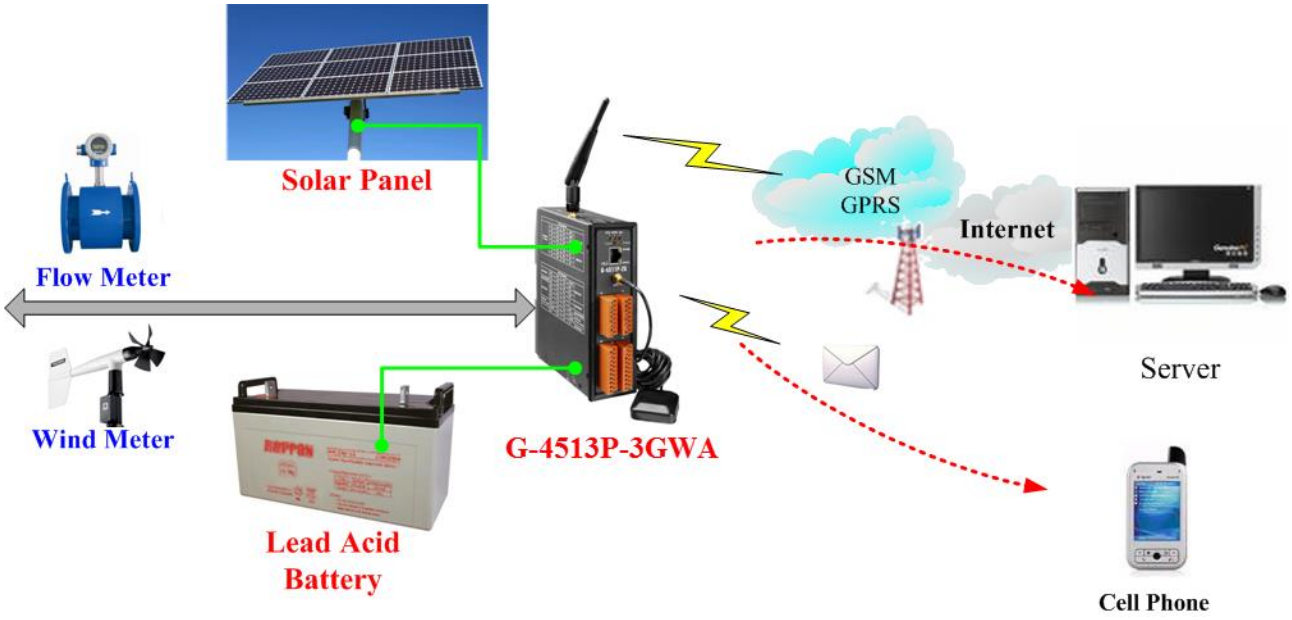
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3G	Yellow			
GPS	-		Green	
Charging / Fault	Green / Red			
Mechanical				
Casing	Metal			
Dimensions	47 mm x 142 mm x 168 mm (W x L x H)			
Installation	DIN-Rail and Wall mount			
Environment				
Operating Temperature	-20 ~ +70 °C	-15 ~ +55 °C	-20 ~ +70 °C	-15 ~ +55 °C
Storage Temperature	-40 ~ +80 °C	-20 ~ +70 °C	-40 ~ +80 °C	-20 ~ +70 °C
Humidity	5~90% RH, non-condensing			

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3. Application Architecture

3.1 Hydrologic / Wind Monitoring Application

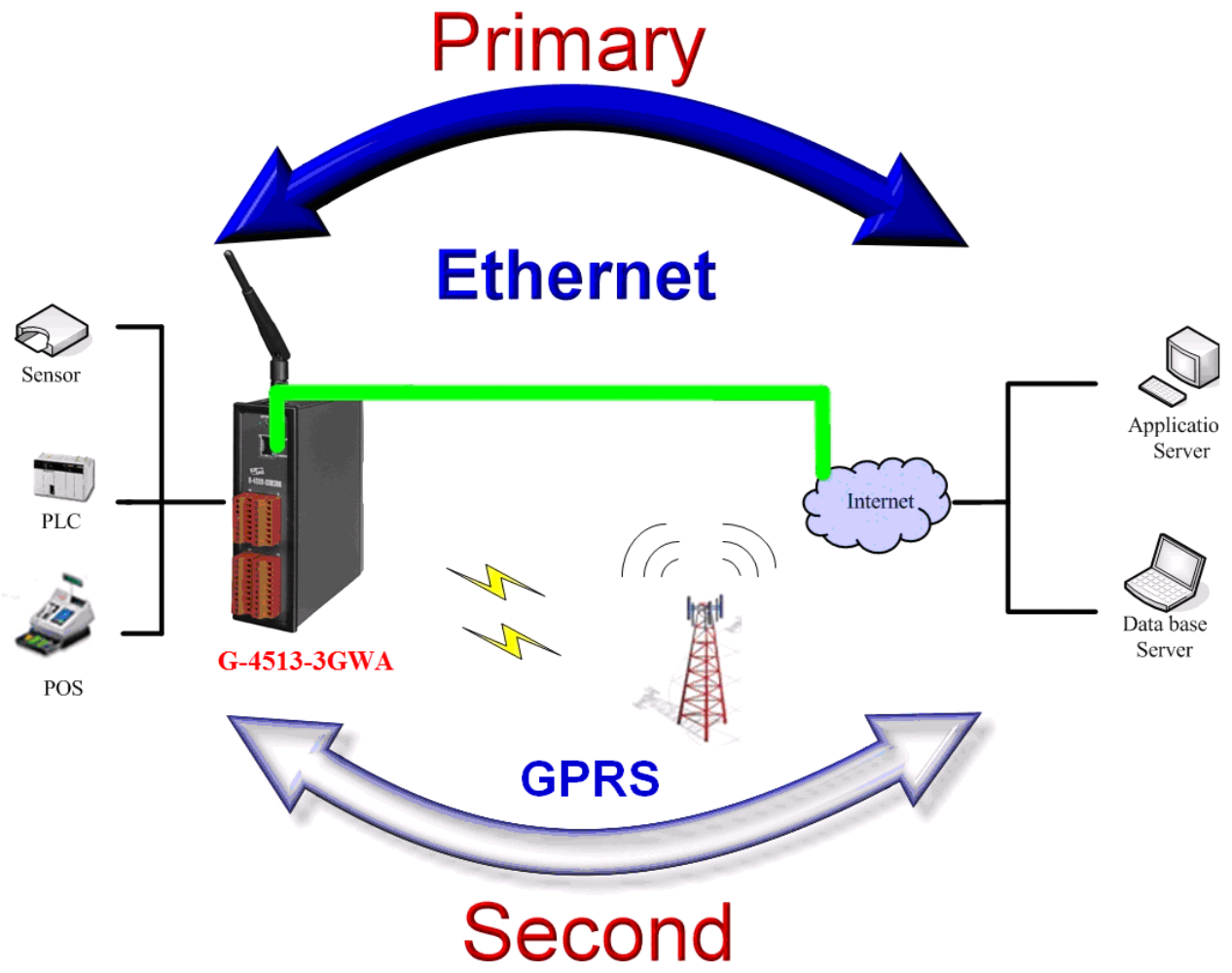


3.2 Car Monitor / Tracking System



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

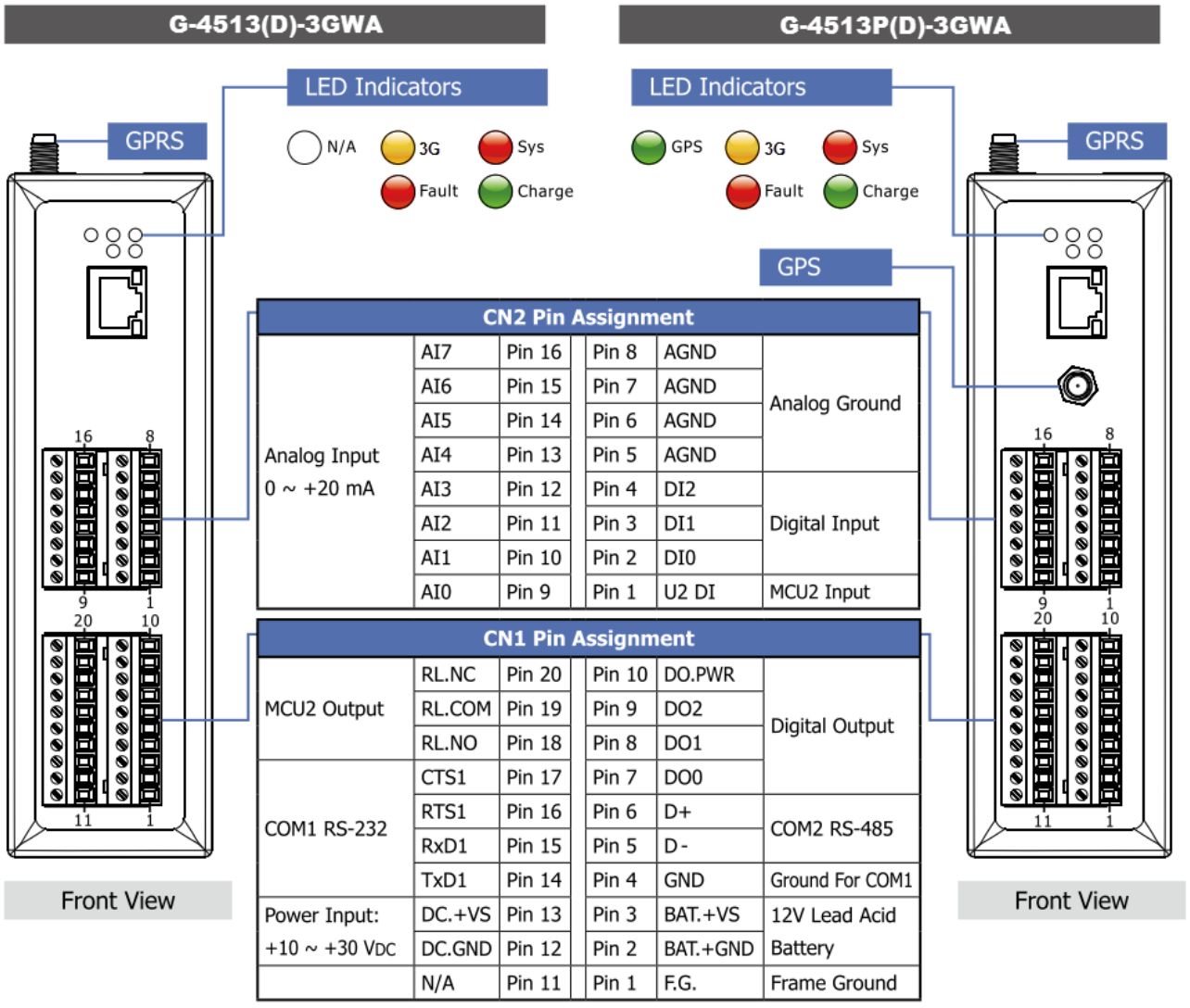
3.3 Redundancy Communication system



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4. Hardware

4.1 Pin Assignments

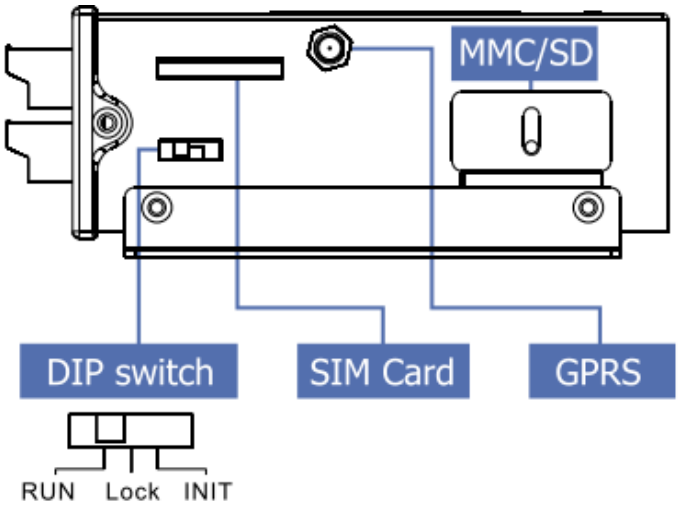


CN2 Pin Assignment

Analog Input 0 ~ +20 mA	AI7	Pin 16	Pin 8	AGND	Analog Ground
	AI6	Pin 15	Pin 7	AGND	
	AI5	Pin 14	Pin 6	AGND	
	AI4	Pin 13	Pin 5	AGND	
	AI3	Pin 12	Pin 4	DI2	Digital Input
	AI2	Pin 11	Pin 3	DI1	
	AI1	Pin 10	Pin 2	DI0	
	AI0	Pin 9	Pin 1	U2 DI	MCU2 Input

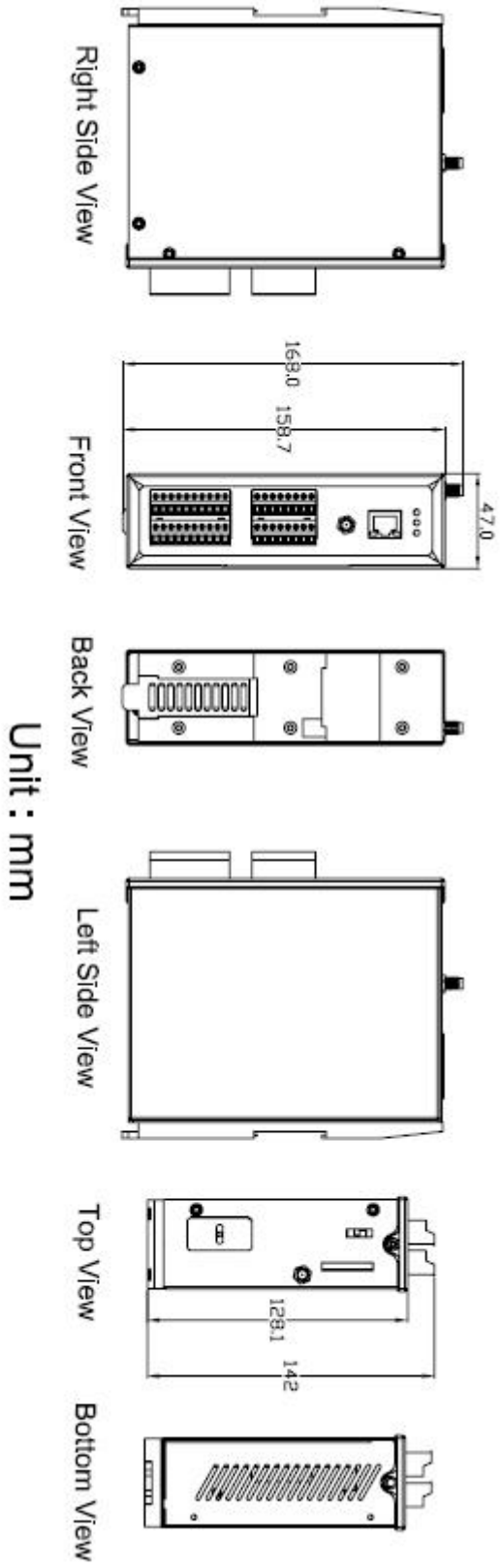
CN1 Pin Assignment

MCU2 Output	RL.NC	Pin 20	Pin 10	DO.PWR	Digital Output
	RL.COM	Pin 19	Pin 9	DO2	
	RL.NO	Pin 18	Pin 8	DO1	
COM1 RS-232	CTS1	Pin 17	Pin 7	DO0	
	RTS1	Pin 16	Pin 6	D+	COM2 RS-485
	RxD1	Pin 15	Pin 5	D-	
	TxD1	Pin 14	Pin 4	GND	Ground For COM1
Power Input: +10 ~ +30 VDC	DC.+VS	Pin 13	Pin 3	BAT.+VS	12V Lead Acid
	DC.GND	Pin 12	Pin 2	BAT.+GND	Battery
	N/A	Pin 11	Pin 1	F.G.	Frame Ground



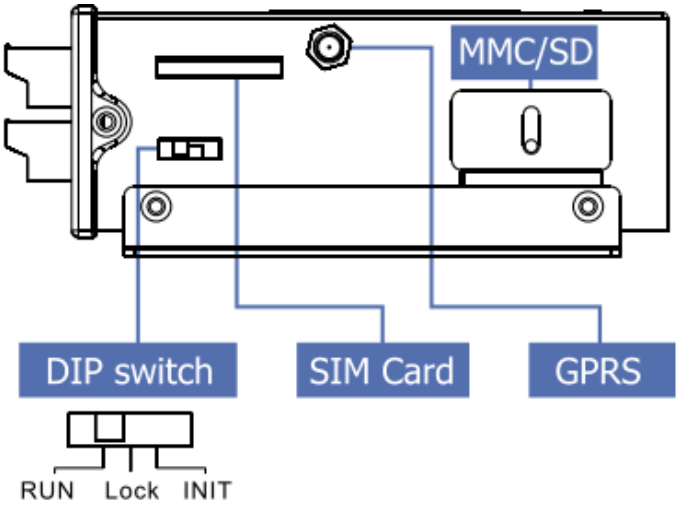
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.2 Dimensions



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

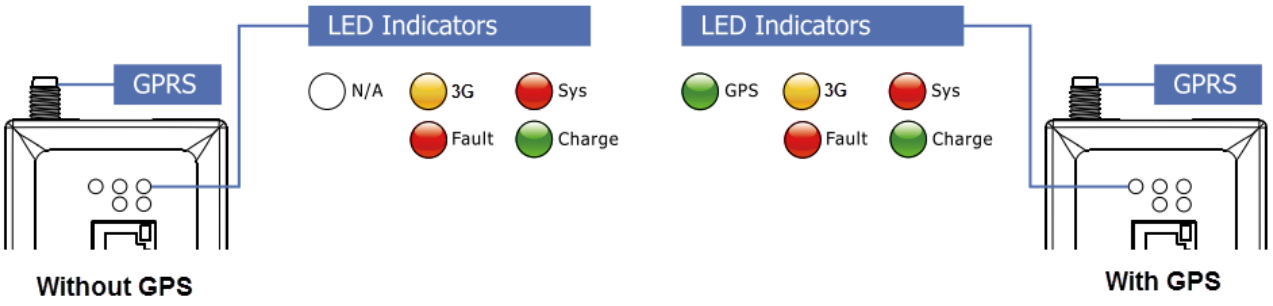
4.3 Operation Mode Switch



Operation Mode Switch	
RUN	OS can execute autoexec.bat
	Flash can be read/write.
LRun	OS can execute autoexec.bat
	Flash is read only (lock).
INIT	OS can not execute autoexec.bat
	Flash can be read/write.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.4 LED Indicators



There are five LED indicators to help users to judge the various conditions of G-4513 Series. The description is as follows：

A. Sys (Red)：System LED is programmable.

B. 3G (Yellow)：The modem LED can indicate the status of 3G module.

	3G module normal	3G module fail
2G Mode	Blinking per 2 sec	OFF or Blinking (not 3 sec)
3G Mode	Blinking twice per 2 sec	

C. GPS (Green)(Option)：The GPS LED can indicate the status of GPS module.

GPS Fail	Search GPS	Receive GPS data
Always OFF	Always ON	Blinking (1 sec)

D. Charging (Green)：Charging status indicator.

Charging	Not Charging
Always ON	Always OFF

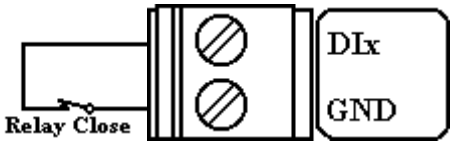
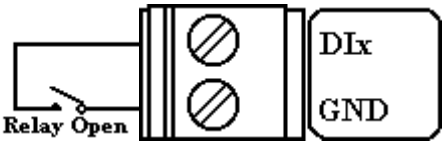
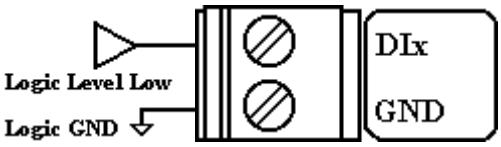
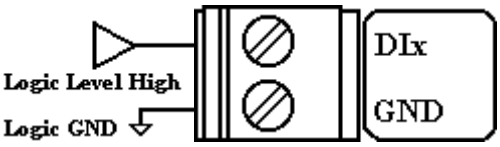
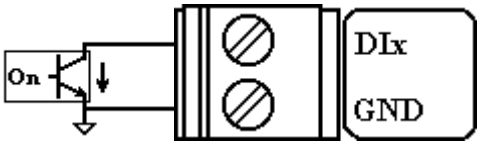
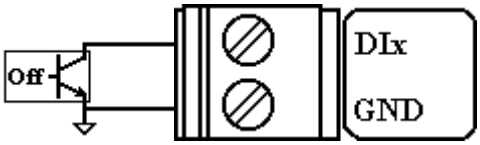
E. Fault (Red)：Charging Fault indicator.

Normal	Fault
Always OFF	Always ON

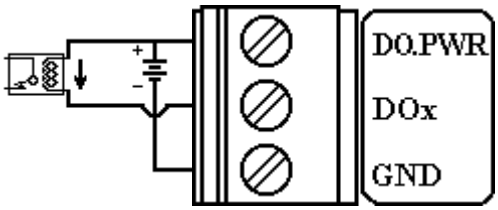
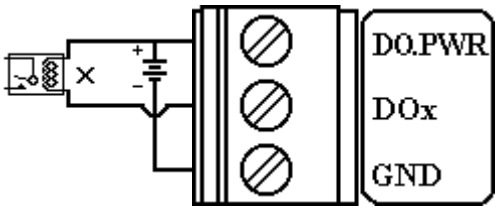
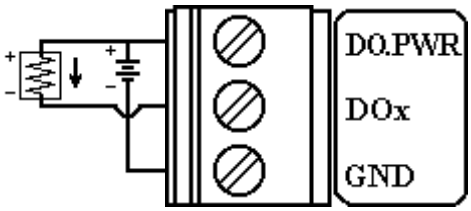
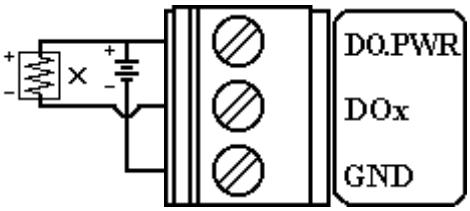
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.5 Wire Connection

➢ Digital Input Wire Connection

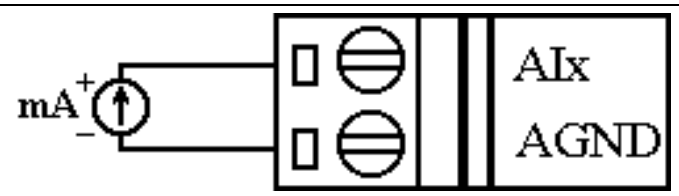
Input Type	ON State DI value as 0	OFF State DI value as 1
Relay Contact		
TTL/CMOS Logic		
Open Collector		

➢ Digital Output Wire Connection

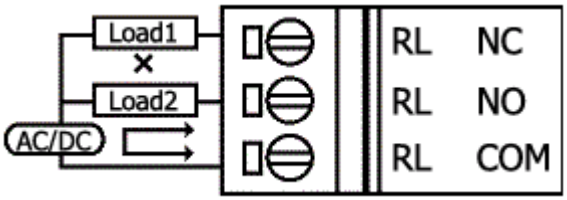
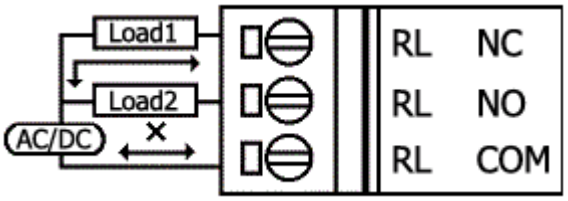
Input Type	ON State DO value as 1	OFF State DO value as 0
Drive Relay		
Resistance Load		

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- Current Input Wire Connection

Input Type	

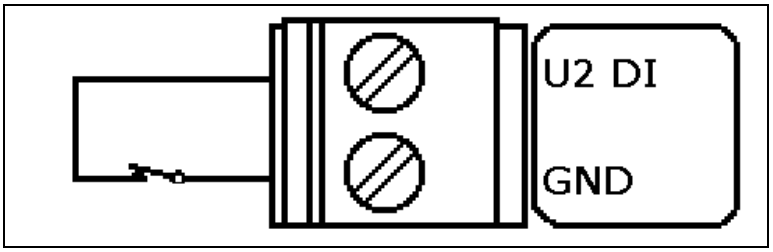
- Relay Wire connection

Relay Output ON	Relay Output OFF
	

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.6 Wake up from Sleep Mode

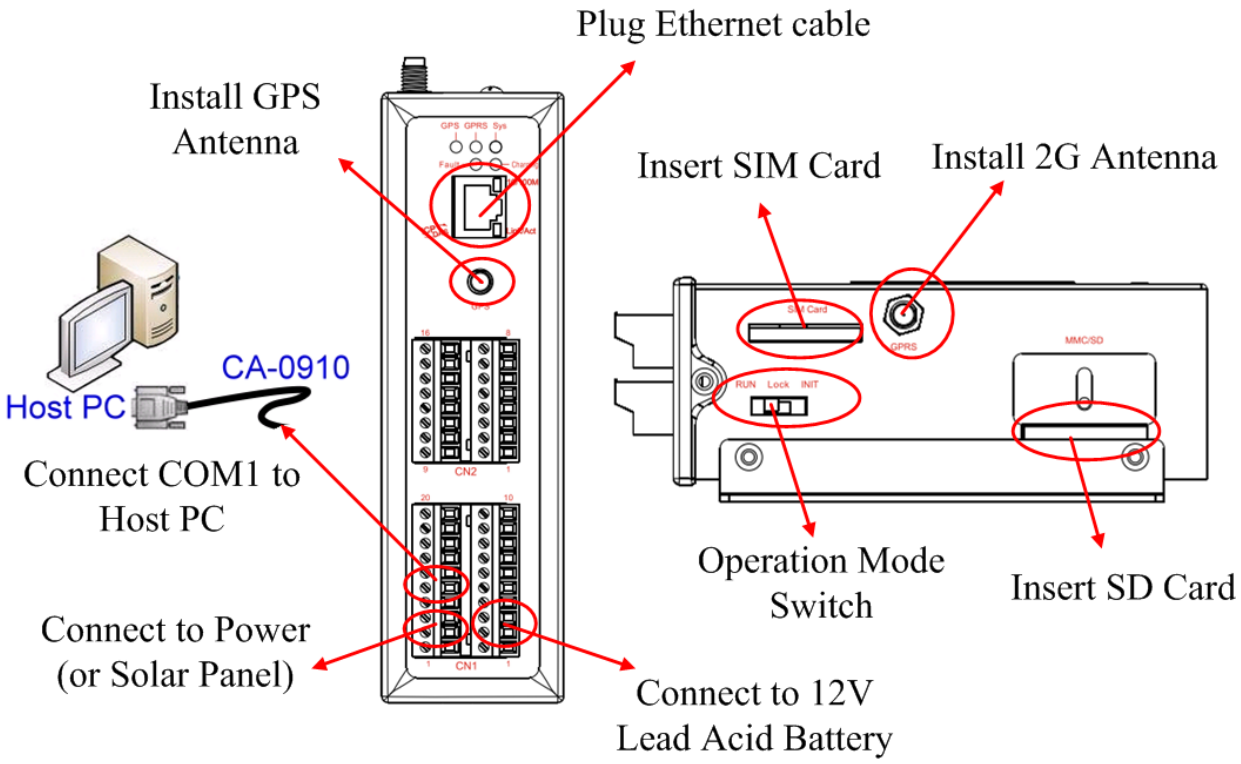
1. When G-4513 was in sleep mode, you can connect the Pin "U2 DI" (MCU2 DI) to GND. to awaking G-4513 from sleep mode.
2. You will read "U2 DI" as 0, when you connect "U2 DI" to GND.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4.7 Installation

1. Install 2G/3G antenna.
2. Plug in the normal SIM card (Before apply the SIM card, confirm it is OK by mobile phone.)
3. Connect the DC.+VS and DC.GND to the power supply or Solar Panel.
4. Connect BAT.+VS and BAT.GND to the 12V Lead Acid Battery.



WARNING! HOT SURFACE DO NOT TOUCH

The product's enclosure may be with high temperature, do not touch before cooling or else will be burned.

SAFETY INSTRUCTION NOTES

The unit installation to final system and the DC source (SELV, Limited Power Source) that is intended to connect with power input pins (DC.+VS / DC.GND) should be complied with requirements of EN 60950-1. Be sure before connect to input pins.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5. Power Saving and Charger

5.1 Power Saving

Sleep Mode

This mode will shut down 7186 CPU, all I/O(3DI, 3DO, 8AI, exclude MCU2 I/O) and GPS, but 3G module still works.

- Power Consumption：14~15 mA@12V
- How to awake G-4513：
  (1) Sleeping Time is finished.
  (2) Trigger U2_DI.(connect U2_DI to GND.; read U2_DI as 0)
  (3) Make a phone call to G-4513

Deep Sleep Mode

This mode will shut down all interface, but exclude MCU2 I/O.

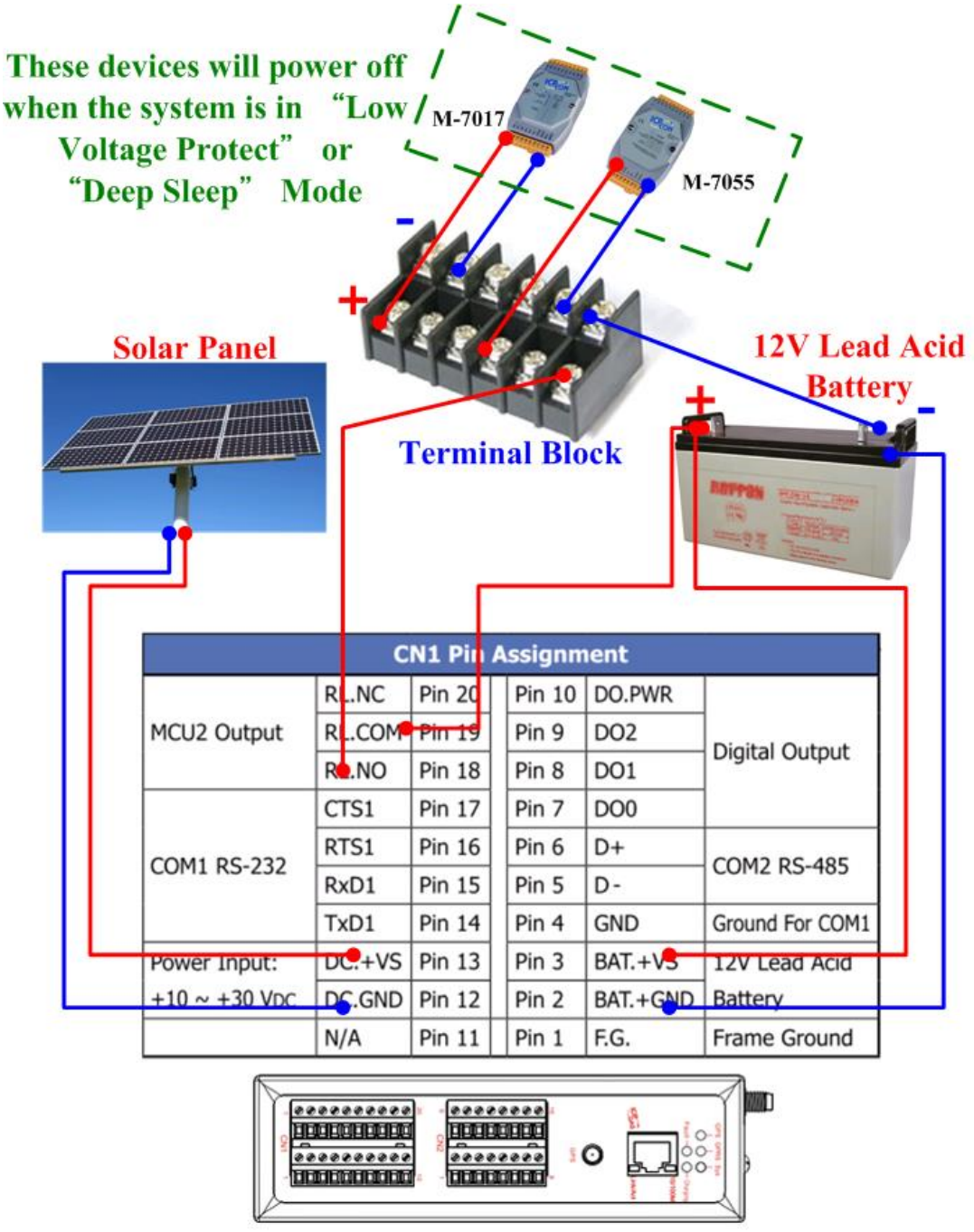
- Power consumption：9~10 mA@12V
- How to awake G-4513：
  (1) Sleeping Time is finished.
  (2) Trigger U2_DI.(connect U2_DI to GND.; read U2_DI as 0)

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Low Voltage Protection

Default value is disabled, and you can enable this function by your program. This function will prevent the battery to over-discharging. When the voltage of the battery is less 11.1V, G-4513 will go into Low Voltage Protect Mode that will turn off all system power. And then G-4513 will wake up if the voltage of the battery is over 12.6V after charging.

- Low Voltage disconnect Voltage = 11.1 V
- Low Voltage reconnect Voltage = 12.6 V
- How to use：please refer to the figure' s wire connection below, and MCU2 library demo code.

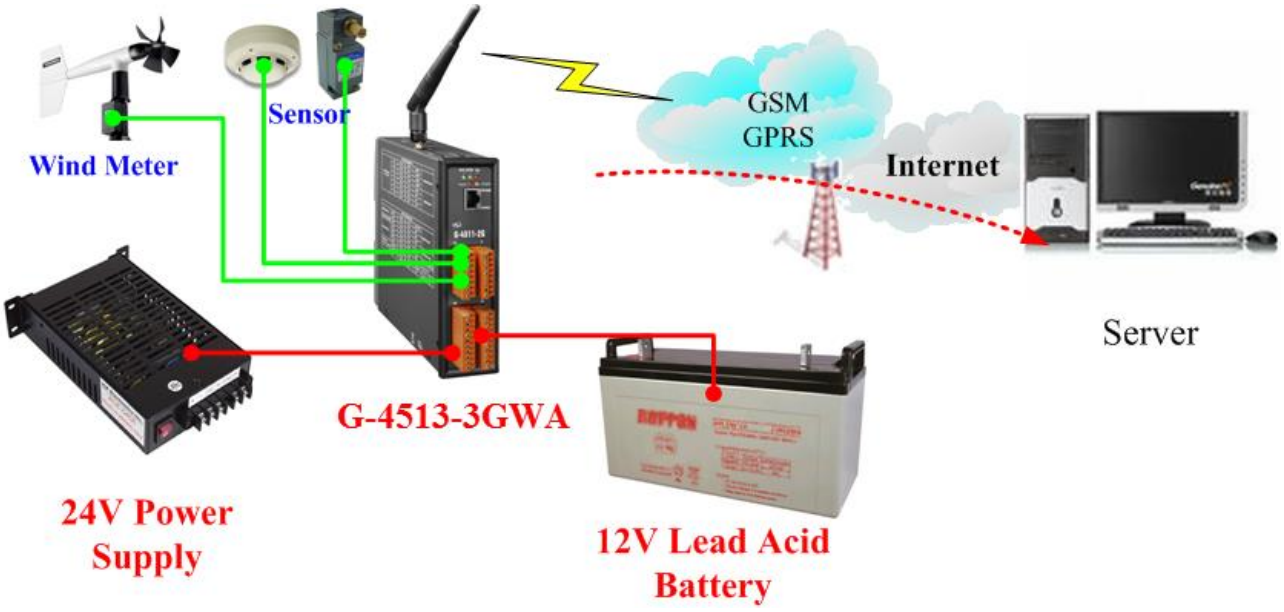


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

5.2 How to Choose the Battery

This section will discuss how to choose a suitable battery for your system. Because the alive time of the system is depending on your system power consumption and your battery capacity, we will calculate with some conditions below.

➢ **Example 1：**



Conditions：

- The system has a 24V external power
- It must work in 2 week when the external power is shut-down.
- The system transmits the data to the server every 10 minutes. (1 minute for full work, and 9 minutes for sleeping)
- Power consumption of deep sleep mode is 7.2 mA@12V
- Average power consumption of full work is 245 mA@12V

Calculation：

- Average power consumption = 245 x (1/10) + 7.2 x (9/10) = 31 (mA)
- 31 (mA) x 24 (hours) x 14 (days) = 10416 mAh
- We may choose "12V, 14Ah Lead Acid Battery" for this system.
- Because 10% battery capacity is low battery voltage state, we don't work in this state. We use 90% battery capacity to calculate.
- Double check the battery capacity：
  14Ah x 90% x 1000 = 12600 mAh > 10416 mAh
  → We will choose a **"12V, 14Ah Lead Acid Battery"** for this system.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Please refer Table 5.2.1 to choose the battery for other report frequency：

Table 5.2.1

Report Frequency	Average power consumption	power consumption of 14 days (mAh @12V)
Every minute (No Sleep)	245	82320
Every 10 minutes	31	10416
Every hour	11.2	3763.2
Every day	7.4	2486.4
Every Month	7.2	2419.2

➢ **Example 2：**



Conditions：

- The system has a 24V external power.
- It must work in 2 week when the external power is shut-down.
- The system transmits 3 Modbus devices data to the server every 10 minutes. (1 minute for full work, and 9 minutes for sleeping)
- The system will power off all Modbus devices by "MCU2 Relay Output" when it's in Deep Sleep Mode.
- Power consumption of deep sleep mode is 7.2 mA@12V
- Average power consumption of full work is 424 mA@12V

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Calculation：

- Average power consumption = 424 x (1/10) + 7.2 x (9/10) = 49 (mA)
- 424 (mA) x 24 (hours) x 14 (days) = 16430.4 mAh
- We may choose "12V, 22Ah Lead Acid Battery" for this system.
- Because 10% battery capacity is low battery voltage state, we don't work in this state. We use 90% battery capacity to calculate.
- Double check the battery capacity:
  22Ah x 90% x 1000 = 19800 mAh > 16430.4 mAh
  → We will choose a **"12V, 22Ah Lead Acid Battery"** for this system.

Please refer Table 5.2.2 to choose the battery for other report frequency：

Table 5.2.2

Report Frequency	Average power consumption	power consumption of 14 days (mAh @12V)
Every minute (No Sleep)	424	142464
Every 10 minutes	48.9	16430.4
Every hour	14.1	4737.6
Every day	7.5	2520
Every Month	7.2	2419.2

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

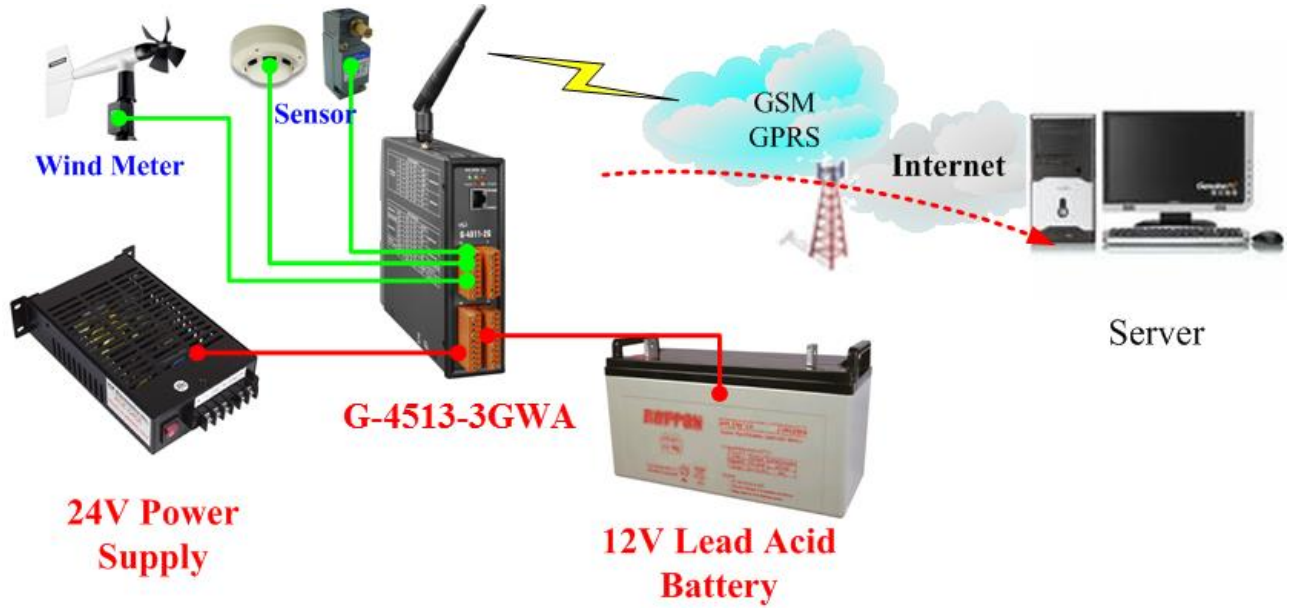
5.3 How to Choose the Solar Panel

This section will discuss how to choose a suitable solar panel for your system. The power of solar panel must be more than the power consumption of the system, and we will calculate with some conditions below.

Things you must know before you calculate

Charging voltage：must be more than +16V
Max. Charge Current：2A
Sun hours：an average value. If the average daily solar radiation of the area is 3 kW/m2 , the sun hours of this area is 3 hours

➢ **Example：**



Conditions：

- Power consumption of deep sleep mode is 7.2 mA@12V
- Average power consumption of the system is 245 mA@12V
- The system transmits the data to the server every 10 minutes. (1 minute for full work, and 9 minutes for sleeping)
- Sun hours is 4 hours/day.
- Using 10W solar panel

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Calculation：

- Solar panel Max. current = 10 (w) / 12 (V) = 0.833 (A) = 833 (mA)
- Usually the charging current may be effected by many factor, like angle, building and other environment factor…etc.
- Here we use 1/2 Max. current to calculate
- 1/2 Max. Current = 833/2 = 416 mA
- Average current = 416 x 4 / 24 = 69 (mA/hr)
- Refer to Table 5.3.2, We can know "Average power consumption" is 31 mA for this system. (refer to Table 5.3.2 or section 5.2)
  → We can know **10W solar Panel is suitable** for this system, because 69 > 31

Table 5.3.1

Solar panel (W)	Max. Current (mA)	1/2 Current (mA)	Sun hours	Average Current for a day (mA/hr)
10	833	416	4	69
20	1666	833	4	138
30	2500	1250	4	208
40	3333	1666	4	277
50	4166	2083	4	347

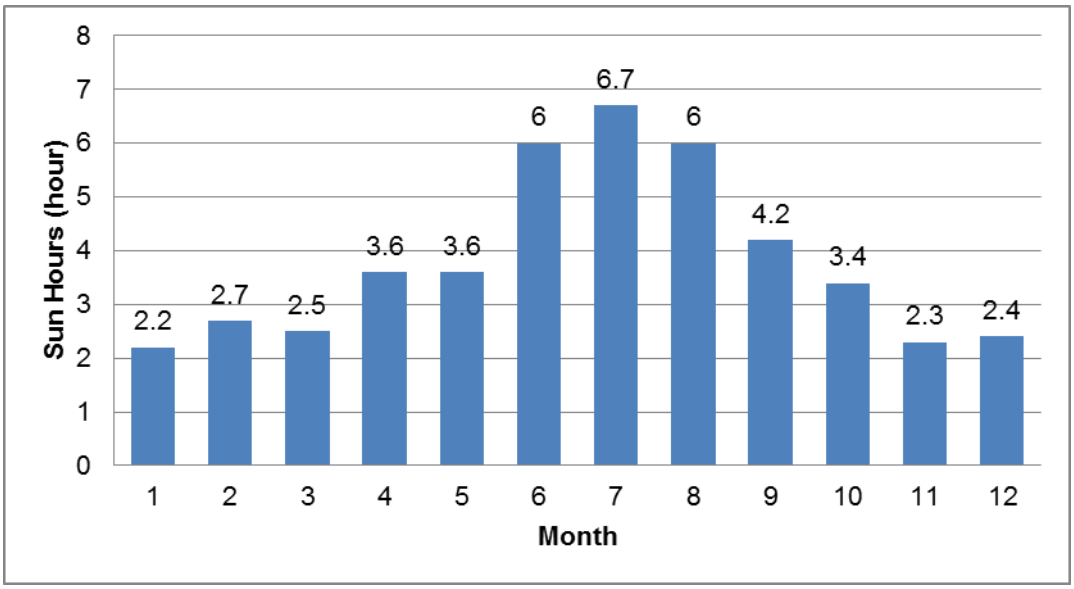
Table 5.3.2

Report Frequency	Average power consumption	power consumption of 14 days (mAh @12V)
Every minute (No Sleep)	245	82320
Every 10 minutes	31	10416
Every hour	11.2	3763.2
Every day	7.4	2486.4
Every Month	7.2	2419.2

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Things you must know about "sun hours"

Usually the "sun hours" is variable with the season. You need use the Min. sun hours to calculate and choose the solar panel and the battery.
For example, if the sun hours of month is like the figure below, you must use 2.2 hours to calculate but not 6.7 hours.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6. APIs and Demo References

6.1 API for Local I/O

Function definition	Description
X305IO_Init	Initial I/O
X305IO_GetLibVersion	Get X305IO_LIB Version
X305IO_Read_AD_CalibrationGain	Read AD Calibration Gain
X305IO_Read_AD_CalibrationOffset	Read AD Calibration Offset
X305IO_AnalogIn	Read value from assign AI channel
X305IO_Read_All_DI	Read All DI
X305IO_Read_One_DI	Read the value form assign DI channel
X305IO_Write_All_DO	Write All DO
X305IO_Write_One_DO	Write the value to the assign DO channel
X305IO_Read_All_DO	Read All DO state
X305IO_Read_One_DO	Read the DO state form the assign DO channel.
X305IO_AnalogIn_SetChannel	Set the AI channel that users want to read.
X305IO_AnalogIn_Hex	Read the value from the specific A/D channel (12 bits)
X305IO_AnalogIn_HexToFloat	Transfer the AI value from 12 bits to float

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.1 X305IO_Init

Initial X305IO.

Syntax

```
int X305IO_Init(void);
```

Parameters

None

Return values

0：success
<>0：error

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.2 X305IO_GetLibVersion

Get X305IO_Lib Version.

Syntax

```
unsigned X305IO_GetLibVersion(void);
```

Parameters

None

Return values

Version Number

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.3 X305IO_Read_AD_CalibrationGain

Read the A/D Calibration Gain.

Syntax

```
float X305IO_Read_AD_CalibrationGain(void);
```

Parameters

None

Return values

Calibration Gain of the AD channels

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.4 X305IO_Read_AD_CalibrationOffset

Read the A/D Calibration Offset.

Syntax

```
float X305IO_Read_AD_CalibrationOffset(void);
```

Parameters

None

Return values

Calibration Offset of the AD channels

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.5 X305IO_AnalogIn

Read the value from the assign AI channel.

Syntax

```
float X305IO_AnalogIn(
    int iChannel
);
```

Parameters

iChannel

0：channel 0
1：channel 1
2：channel 2
3：channel 3
4：channel 4
5：channel 5
6：channel 6
7：channel 7

Return values

0.0mA ~ 20.0mA

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.6 X305IO_Read_All_DI

Read all DI values of the G-4513 series.

Syntax

```
int X305IO_Read_All_DI(void);
```

Parameters

None

Return values

0x00~0x07

Example

When DI0 Ground
DI1 Open
DI2 Open

value = X305IO_Read_All_DI( );

value = 0x6

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.7 X305IO_Read_One_DI

Read the value from the assign DI channel.

Syntax

```
int X305IO_Read_One_DI(
    int iChannel
);
```

Parameters

iChannel

0：channel 0

1：channel 1

2：channel 2

Return values

1：open

Logic high level (+3.5V ~ +30V)

0：close to GND

Logic low level (0V ~ +1V)

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.8 X305IO_Write_All_DO

Write to all DO values of the G-4513 series.

Syntax

```
void X305IO_Write_All_DO(
    int iOutValue
);
```

Parameters

iOutValue
0x0~0x7

Return values

None

Example

X305IO_Write_All_DO(6);

After function execute：
DO0 OFF
DO1 ON
DO2 ON

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.9 X305IO_Write_One_DO

Write the specific value to the assign DO channel.

Syntax

```
void X305IO_Write_One_DO(
    int iChannel, int iStatus
);
```

Parameters

iChannel

0：channel 0
1：channel 1
2：channel 2

iStatus

0：Status is OFF
1：Status is ON

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.10 X305IO_Read_All_DO

Read all DO values of the G-4513 series.

Syntax

```
int X305IO_Read_All_DO(void);
```

Parameters

None

Return values

0x0~0x7

Example

When DO0 OFF
DO1 ON
DO2 ON

Value = X305IO_Read_All_DO( );

Value = 0x6

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.11 X305IO_Read_One_DO

Read the state from the assign DO channel.

Syntax

```
int X305IO_Read_One_DO(
    int iChannel
);
```

Parameters

iChannel

0：channel 0

1：channel 1

2：channel 2

Return values

0：OFF

1：ON

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.12 X305IO_AnalogIn_SetChannel

Set the specific AI channel that users want to read.

Syntax

```
int X305IO_AnalogIn_SetChannel(
    unsigned iChannel
);
```

Parameters

iChannel

0：channel 0
1：channel 1
2：channel 2
3：channel 3
4：channel 4
5：channel 5
6：channel 6
7：channel 7

Return values

0：Set up success
-1：Set iChannel number error

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.13 X305IO_AnalogIn_Hex

Read the value of the assign AI channel assigned by X305IO_AnalogIn_SetChannel function.

Syntax

```
int X305IO_AnalogIn_Hex(void);
```

Parameters

None

Return values

After Read assign AI channel value.

Example

```
X305IO_AnalogIn_SetChannel(0);     // Set channel 0
X305IO_AnalogIn_Hex( );
```


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.1.14 X305IO_AnalogIn_HexToFloat

Set the AI value from 12 bits to float format.

Syntax

```
float X305IO_AnalogIn_HexToFloat(
    int iValue
);
```

Parameters

iValue

A value want to 12 bits transform float.

Return values

The transferred AI value by float format.

Example

Set the channel 0 to read, and then transform the value to float.

```
float AdValue;

X305IO_AnalogIn_SetChannel(0);
AdValue=X305IO_AnalogIn_HexToFloat(X305IO_AnalogIn_Hex( ));
```


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.2 API for MMC/SD



Required library and header files:
SD_Vnnn.LIB and microSD.h

Function definition	Description
pc_init	Initializes the SD socket library
pc_open	1. Open an existing file and return a file handle 2. Creates a new file
pc_close	Closes a file and release a file handle.
pc_read	Reads the specified file
pc_write	Writes the specified file
pc_seek	Moves the file pointer to relative offset from the current offset
pc_tell	Gets current offset of the file pointer
pc_eof	Checks whether the end-of-file is reached
pc_format	Formats the SD card as FAT (FAT32)
pc_mkdir	Creates a directory or subdirectory
pc_rmdir	Removes an existing directory
pc_move	Renames an existing file or a directory, including the subdirectory
pc_del	Deletes the specified file
pc_deltree	Deletes the specified directory or subdirectory
pc_isdir	Checks whether the file is a directory
pc_isvol	Checks if is a volume
pc_size	Gets the size of the specified file
pc_set_cwd	Sets the current working directory

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

pc_get_cwd	Gets the pathname of the current working directory
pc_gfirst	Moves the pointer to the first element
pc_gnext	Moves the pointer to the next element
pc_gdone	Moves the pointer to the last element
pc_get_freeSize_KB	Gets the free space of the SD memory card
pc_get_usedSize_KB	Gets the used space of the SD memory card
pc_get_totalSize_KB	Gets the total size of the SD memory card
pc_get_attributes	Gets the file attributes
pc_set_attributes	Sets the file attributes
pc_get_errno	Gets the error number

API for starting SD card

1. pc_ Init()
   Before using any SD functions, pc_init() must be called to initialize the SD.

API for enabling/disabling SD card

2. pc_open()
   Before writing/reading data to/from the SD card, pc_open() must be called to open the file.

3. pc_close()
   After the data has finished being written/read to/from the SD, pc_close() must be called to close the file with a file handle.

API for writing data to the SD card

4. pc_write()
   After using pc_open() to open the file, pc_write() must be called to read data from the SD.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

For example, writing data to the SD:

```
#include <string.h>
#include <stdio.h>
#include "upac5000.h"
#include "microSD.h"
void main(void)
{
    int fd, iRet;
    InitLib();
    if(pc_init())
        Print("Init microSD ok\r\n");
    else
    {
        Print("Init microSD failed\r\n");
        iRet=pc_get_errno();
        switch(iRet)
        {
            case PCERR_BAD_FORMAT: //1
                Print("Error 01: format is not FAT\r\n");
                break;
            case PCERR_NO_CARD: //2
                Print("Error 02: no microSD card\r\n");
                break;
            default:
                Print("Error %02d: unknow error\r\n", iRet);
                break;
        }
    }
    fd=pc_open("test.txt", (word) (PO_WRONLY|PO_CREAT|PO_APPEND),
    (word) (PS_IWRITE|PS_IREAD));
    if(fd>=0)
    {
        pc_write(fd, "1234567890", 10); /* write 10 bytes */
        pc_close(fd);
    }
}
```


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

API for reading data from the SD card

5. pc_read()
   After using pc_open() to open the file, pc_read() must be called to read data from the SD.

For example, reading data from the SD:

```
#include <string.h>
#include <stdio.h>
#include "upac5000.h"
#include "microSD.h"
void main(void)
{
    int fd, iRet;
    unsigned char Buffer[80];
    InitLib();
    if(pc_init())
        Print("Init microSD ok\r\n");
    else
    {
        Print("Init microSD failed\r\n");
        iRet=pc_get_errno();
        switch(iRet)
        {
            case PCERR_BAD_FORMAT: //1
                Print("Error 01: format is not FAT\r\n");
                break;
            case PCERR_NO_CARD: //2
                Print("Error 02: no microSD card\r\n");
                break;
            default:
                Print("Error %02d: unknow error\r\n", iRet);
                break;
        }
    }
    fd=pc_open("test.txt", (word) (PO_RDONLY), (word) (PS_IWRITE|PS_IREAD));
    if(fd>=0)
    {
        iRet=pc_read(fd, Buffer, 10);    /* reads 10 bytes */
```


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

```
        Buffer[10]=0; /* adds zero end to the end of the string */
        pc_close(fd);
        Print("%s", Buffer);
    }
}
```

For more demo program about the microSD, please refer to:
CD:\napdos\g-4513-3gwa\software\demo\basic\microSD\
http://ftp.icpdas.com/pub/cd/usbcd/napdos/g-4513-3gwa/software/demo/basic/microsd/

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3 API for LCD

Function definition	Description
LCD_Init	Initialize the library
LCD_BackLight_On	Turn on the LCD backlight
LCD_BackLight_Off	Turn off the LCD backlight
LCD_ShowText	Display one character on the LCD panel
LCD_ClrScrn	Clear the LCD panel
LCD_StandByMode	Enter the stand by mode
LCD_NormalMode	Restore the LCD to normal mode
LCD_GotoPosition	Move the cursor to the specified position
LCD_CursorDisplay	Set the Cursor display status
LCD_LineReverse	Select one of four line and reverse the display
LCD_LineRestore	Select one of four line and restore the display
LCD_GetLibDate	Gets the create date of funciton library
LCD_GetLibVersion	Gets the version number of function library

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.1 LCD_Init

Initialize parameters about LCD functions in the library.

Syntax

```
void LCD_Init(void);
```

Parameters

None

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.2 LCD_BackLight_On

Turn on the LCD backlight.

Syntax

```
void LCD_BackLight_On(void);
```

Parameters

None

Return

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.3 LCD_BackLight_Off

Turn off the LCD backlight.

Syntax

```
void LCD_BackLight_Off(void);
```

Parameters

None

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.4 LCD_ShowText

Display one character on the LCD panel, and the cursor will right-shifted by one character position automatically.

Syntax

```
void LCD_ShowText(
    uchar Text
);
```

Parameters

Text

Display character

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.5 LCD_ClrScrn

Clear the LCD panel.

Syntax

```
void LCD_ClrScrn(void);
```

Parameters

None

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.6 LCD_StandByMode

Enter the stand by mode, and it can be terminated by either LCD_NormalMode() or other function.

Syntax

```
void LCD_StandByMode(void);
```

Parameters

None

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.7 LCD_NormalMode

Restore the LCD to normal mode when it is in the stand by mode.

Syntax

```
void LCD_NormalMode(void);
```

Parameters

None

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.8 LCD_GotoPosition

Move the cursor to the specified position.

Syntax

```
void LCD_GotoPosition(
    int Line,
    int Offset
);
```

Parameters

Line

One of four line numbers (1 to 4)

Offset

Cursor position (1 to 8)

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.9 LCD_CursorDisplay

Set the Cursor display status.

Syntax

```
void LCD_CursorDisplay(
    int Display,
    int Blink
);
```

Parameters

Display

Cursor display on/off

1: Display on

0: Display off

Blink

Character blink on/off

1: Display on

0: Display off

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.10 LCD_LineReverse

Select one of four line and reverse the display.

Syntax

```
void LCD_LineReverse(int Line);
```

Parameters

Line

One of four line numbers (0 to 4)

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.11 LCD_LineRestore

Select one of four line and restore the display.

Syntax

```
void LCD_LineRestore(
    int Line
);
```

Parameters

Line

One of four line numbers (0 to 4)

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.12 LCD_GetLibDate

Gets the create date of funciton library.

Syntax

```
void LCD_GetLibDate(
    unsigned char *LibDate
);
```

Parameters

LibDate

Gets the create date of funciton library

Return values

None

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.3.13 LCD_GetLibVersion

Get the version number of function library.

Syntax

```
unsigned LCD_GetLibVersion(void);
```

Parameters

None

Return values

Return the current version number.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4 API for GSM

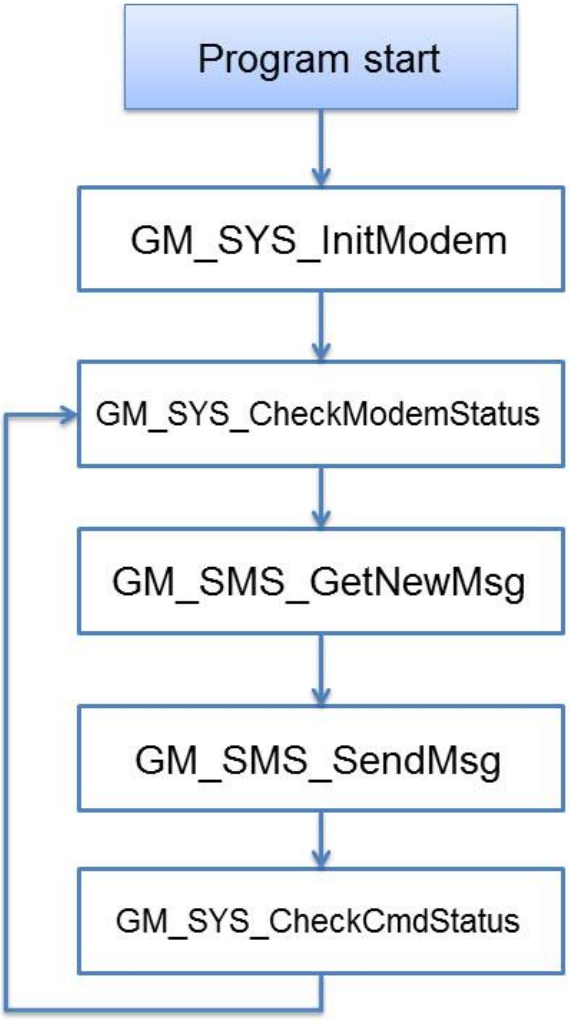
For more detailed API, please refer to GSM_U2_Library_User's Manual

Function definition	Description
GM_SYS_InitModem	Initialize Modem
GM_SYS_CloseModem	Close the modem
GM_SYS_CheckModemStatus	Check modem status, and suggest you check it in your loop every time
GM_SYS_CheckSignal	Check signal quality
GM_SYS_CheckReg	Check register
GM_SMS_SendMsg	Send a message
GM_SMS_GetNewMsg	Get a new sms message
GM_NET_SetNet	Set Net profile data
GM_NET_InstallLink	Built TCP/UDP link
GM_NET_CloseNet	Close Network
GM_NET_CloseLink	Close client link[n]
GM_NET_Send	Send a packet
GM_NET_GetNewPacket	Get the new packet

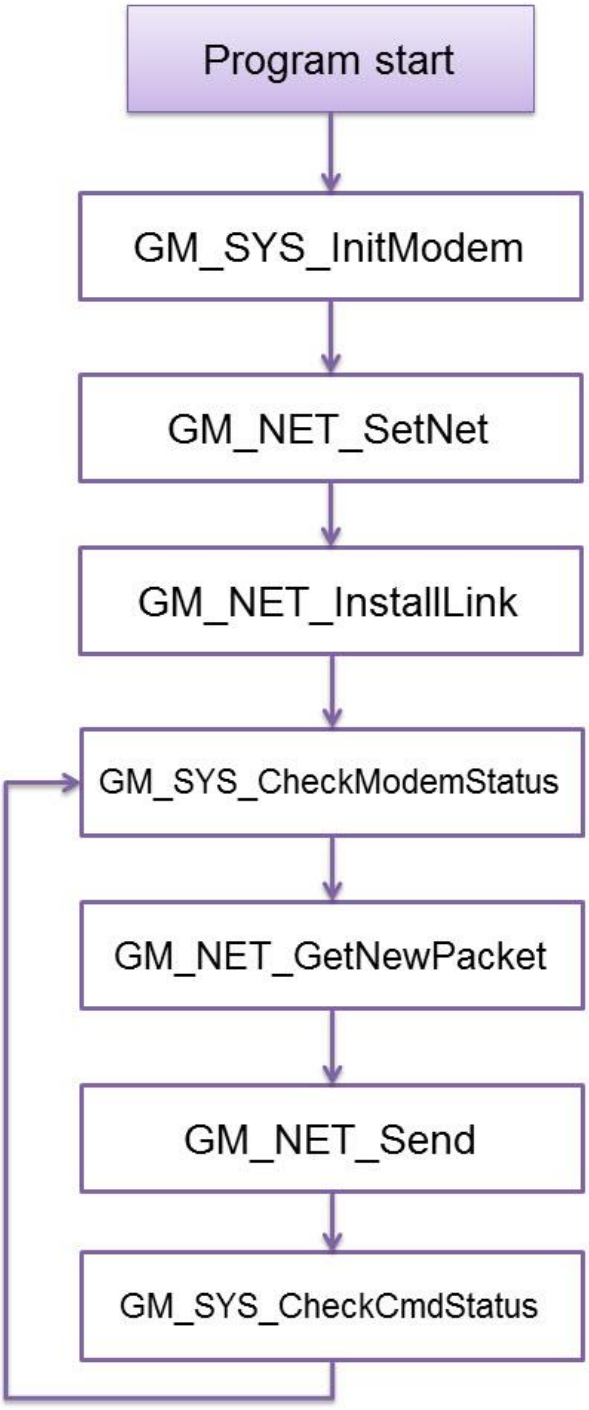
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.1 GSM Design Flowchart

SMS Design Flowchart



GPRS Design Flowchart



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.2 GM_SYS_InitModem

Initialize Modem.

**must use GM_SYS_CheckModemStatus() to check modem status later

Syntax

```
int GM_SYS_InitModem(
    SYSProfile sysProfile
);
```

Parameters

sysProfile
set system profile

Return values

GM_NOERROR：success
GM_COMERROR：comport error
GM_INITERROR：init fail error

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.3 GM_SYS_CloseModem

Close the modem.

**Please call GM_SYS_InitModem() to wake up modem after using GM_SYS_CloseModem(1) to shut down the modem.

Syntax

```
int GM_SYS_CloseModem(
    int mode
);
```

Parameters

mode

0：close modem, but maintain it power on

1：close modem and set it power off

Return values

GM_NOERROR：no error

GM_CMDERROR：command error

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.4 GM_SYS_CheckModemStatus

Check modem status, and suggest you check it in your loop every time.

Syntax

```
int GM_SYS_CheckModemStatus(void);
```

Parameters

None

Return values

GM_NOERROR：modem register success, can service
GM_NOREG：modem not registered, can't service

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.5 GM_SYS_CheckSignal

Check signal quality.

Syntax

```
int GM_SYS_CheckSignal(void);
```

Parameters

None

Return values

signal quality

0	-113 dBm or less
1	-111 dBm
2...30	-109... -53 dBm
31	-51 dBm or greater

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.6 GM_SYS_CheckReg

Check register.

Syntax

```
int GM_SYS_CheckReg(void);
```

Parameters

None

Return values

Register flag
- 0：not registered
- 1：registered, home network
- 2：not registered, and searching...
- 3：registration denied
- 4：unknown
- 5：registered, roaming

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.7 GM_SMS_SendMsg

Send a message.

**must use "GM_SYS_CheckCmdStatus()" to check status later

Syntax

```
int GM_SMS_SendMsg(
    strEncode_Msg* strMsg
);
```

Parameters

strMsg
the message that will be sent.

Return values

GM_NOERROR：no error
GM_NOREG：not registered, or can't service
GM_BUSY：modem busy

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.8 GM_SMS_GetNewMsg

Get a new sms message.

Syntax

```
int GM_SMS_GetNewMsg(
    strEncode_Msg* msg
);
```

Parameters

msg

new sms message

Return values

0：no new message

1：new message coming

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.9 GM_NET_SetNet

Set Net profile data.

Syntax

```
int GM_NET_SetNet(
    NetProfile netProfile
);
```

Parameters

netProfile
Net profile data

Return values

GM_NOERROR：no error
GM_CMDERROR：command error

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.10 GM_NET_InstallLink

Built TCP/UDP link.

Syntax

```
int GM_NET_InstallLink(
    int n,
    int tcp,
    char* serverIP,
    unsigned int serverPort
);
```

Parameters

n

link number (0~6)
3G (G-4513 series)：0~6
2G (G-4511 series)：0

tcp

client type, tcp=1 for TCP client ; tcp=0 for UDP client

serverIP

IP or Domain name of the server, ex: "61.111.222.333", "test.com.tw"

serverPort

TCP/UDP Port of the server (1~65535), ex: 1234

Return

GM_NOERROR：correct parameter to install TCP/UDP link
GM_CMDERROR：command error

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.11 GM_NET_CloseNet

Close Network.

Syntax

```
int GM_NET_CloseNet(void);
```

Parameters

None

Return values

GM_NOERROR：no error
GM_CMDERROR：command error
GM_BUSY：modem busy

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.12 GM_NET_CloseLink

Close client link[n].

Syntax

```
int GM_NET_CloseLink(
    int n
);
```

Parameters

n

3G (G-4513 series)：0~6
2G (G-4511 series)：0

Return values

GM_NOERROR：no error
GM_CMDERROR：command error
GM_BUSY：modem busy

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.13 GM_NET_Send

Send a packet.
**must use "GM_SYS_CheckCmdStatus()" to check status later

Syntax

```
int GM_NET_Send(
    char link,
    char* data,
    int dataLen
);
```

Parameters

link
link number
3G (G-4513 series)：0~6
2G (G-4511 series)：0

data
data that will be sent

dataLen
data length, Max.=1000

Return values

GM_NOERROR：no error
GM_CMDERROR：command error
GM_BUSY：modem busy

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.14 GM_NET_GetNewPacket

Get the new packet.

Syntax

```
int GM_NET_GetNewPacket(
    GPRSData* gprsData
);
```

Parameters

gprsData
new data packet

Return values

0：no new packet
1：new packet coming

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

6.4.15 GSM Demo References

For example, send and receive sms message

```
#include <conio.h>
#include <stdio.h>
#include <malloc.h>
#include <stdlib.h>
#include <string.h>
#include "../lib/G4500.h"
#include "../lib/GSM_U2.h"
#include "../lib/OS7_COM.h"
#include "../lib/MCU2LIB.h"

int main(void)
{
    int iAction=1, quit=1;
    int i, j, tmp;
    int Result=0;
    strEncode_Msg RecMsg, SendMsg;
    char sendNumber[20];
    int send_n;
    int sendStatus = 0;
    SYSProfile sysProfile;

    //-- init
    InitLib();

    /*---- init modem----*/
    strcpy(sysProfile.PINCode, "0000"); /*The pin code of SIM card, ex: "0000"*/
    sysProfile.modemPort = 4;   /*modem port number. G-4500 = 4, uP-5000 = 11*/
    sysProfile.hardware = 0;       /*hardware type. 1: G-4500, 2: uPAC-5000, 0: Other*/
    GM_SYS_SetPowerFunction(powerFunction); /*set power-control function*/


    if((Result = GM_SYS_InitModem(sysProfile)) == GM_NOERROR)
        Print("init_modem success!!\r\n");
    else
    {
        Print("init_modem fail!! return value is %d\r\n", Result);
```


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

```
        return 1;
    }

    /*--check Could the modem service? --*/
    while(GM_SYS_CheckModemStatus() != GM_NOERROR)
    {
        Print("wait modem register...\r\n");
        DelayMs(1000);
    }
    Print("modem registered!!\r\n");

    while(iAction!=0)
    {
        iAction=0;
        quit=0;
        Print("1) Send ASCII messages\r\n");
        Print("2) check signal quality\r\n");
        Print("3) check registered?\r\n");
        Print("0) Quits demo program\r\n");
        Print("Choose an option and press [Enter]: ");
        Scanf("%d", &iAction);

        switch(iAction)
        {
        case 1:     /*Send ASCII messages*/
            Print("Send ASCII messages\r\n");

            Print("Please Input Phone Number =");
            Scanf("%s", sendNumber);

            Print("How many message do you want to send?\r\n");
            Scanf("%d", &send_n);
            i = 0;
            Print("== start to send sms, please press <ESC> to exit ==\r\n");
            while(1)
            {
                /* press "ESC" to exit */
                if(Kbhit())
                {
                    tmp = Getch();
```


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

```
            if( tmp == 27 || tmp == 'q')
                break;
        }

/*--(1) check that could the modem service, if it can't, skip operating the modem below --*/
        if(GM_SYS_CheckModemStatus() != GM_NOERROR)
            continue;

        /*--(2) send messages, when the modem can service--*/
        if(i<send_n)
        {
            switch((sendStatus=GM_SYS_CheckCmdStatus()))
            {
            case GM_READY:
                Print("sending message(%d)....\r\n", i);

                strcpy(SendMsg.phoneNumber, sendNumber);
                SendMsg.mode = GSM_7BIT;
                sprintf(SendMsg.msg, "GSM_Test(%2d)", i);
                SendMsg.dataLen = strlen(SendMsg.msg);
                GM_SMS_SendMsg(SendMsg);
                break;
            case GM_NOERROR:
                Print("send success!!\r\n");
                i++;
                break;
            case GM_BUSY:  //sending, and waiting reply
                break;
            default:
                Print("send error, and skip this one, error code=%d\r\n",
                sendStatus);
                i++;
                break;
            }
        }
        /*--(3) if any sms message come in, print it --*/
        if( GM_SMS_GetNewMsg(&RecMsg) != 0)
        {
            printMsg(RecMsg);
        }
```


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

```
                }
                break;

        case 2://check signal quality
                Result = GM_SYS_CheckSignal();
                Print("signal value = %d\r\n", Result);
                break;

        case 3://check register value
                Result = GM_SYS_CheckReg();
                Print("register value = %d    (0:no register, 1:registered, 2:registering)\r\n",
                Result);
                break;

        case 0:
        default:
                quit=1;
                break;

        }//end switch()

        if(!quit)
        {
                Print("Press any key to continue...\r\n");
                Getch();
        }
    }   /*end while(1)*/

    /*must close before program ending to release you resource*/
    /*-- Close the modem, 0:not turn off modem, 1:turn off modem*/
    GM_SYS_CloseModem(0);

    Print("Please press ENTER to exit...\r\n");
    Getch();
    return 0;
}
```


LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

For example, TCP client Demo

```
#include <conio.h>
#include <stdio.h>
#include <malloc.h>
#include <stdlib.h>
#include <string.h>
#include "../lib/G4500.h"
#include "../lib/GSM_U2.h"
#include "../lib/OS7_COM.h"
#include "../lib/MCU2LIB.h"

int main(void)
{
   int iAction=1, quit=1;
   int i, j,tmp;
   int Result=0;
   int send_n;
   NetProfile netProfile;
   SYSProfile sysProfile;
   GPRSData gprsData;
   char serverIP[16];
   int serverPort;
   long socket_n;
   int netSendStatus = 0;
   char myIP[16];

   InitLib();

   /*---- init modem ----*/
   strcpy(sysProfile.PINCode, "0000");    /*The pin code of SIM card, ex: "0000"*/
   sysProfile.modemPort = 4;    /*modem port number. G-4500 = 4, uP-5000 = 11*/
   sysProfile.hardware = 0;       /*hardware type. 1: G-4500, 2: uPAC-5000, 0: Other*/
   GM_SYS_SetPowerFunction(powerFunction); /*set power-control function*/

   if( (Result = GM_SYS_InitModem(sysProfile)) == GM_NOERROR)
         Print("init_modem success!!\r\n");
   else{
         Print("init_modem fail!! return value is %d\r\n", Result);
         return 1;
```


LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

```
}

/*-- check Could the modem service? --*/
while(GM_SYS_CheckModemStatus() != GM_NOERROR)
{
    Print("wait modem register...\r\n");
    DelayMs(1000);
}
Print("modem registered!!\r\n");

while(iAction!=0)
{
    iAction=0;
    quit=0;
    Print("1) TCP client demo\r\n");
    Print("0) Quits demo program\r\n");
    Print("Choose an option and press [Enter]: ");
    Scanf("%d", &iAction);
    Print("\r\n");

    switch(iAction)
    {
      case 1:   //TCP client demo
      Print("TCP client demo start\r\n");

        /* set Network profile */
        /* APN for network provided by your cellular provider*/
        strcpy(netProfile.APN, "INTERNET");
        /*username for network provided by your cellular provider */
        strcpy(netProfile.pw, "guest");
        /*password for network provided by your cellular provider */
        strcpy(netProfile.user, "guest");
        /* The most basic task of DNS is to translate hostnames such as
        www.icpdas.com to IP address such as 96.9.41.131 */.
        strcpy(netProfile.DnsServerIP, "");      /*empty string = system default value*/
        GM_NET_SetNet(netProfile);

        /*set ip, port of server */
        Print("please input server IP:(ex: 74.125.227.48)\r\n");
        Scanf("%s", serverIP);
```


LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

```
        Print("please input server Port:(ex: 80)\r\n");
        Scanf("%d", &serverPort);

        /*--(1) install link[0], GM_NET_InstallLink(0, 0, serverIP, serverPort) for UDP--*/
        GM_NET_InstallLink(0, 1, serverIP, serverPort);
        Print("linking...\r\n");
        socket_n = 0;    //count for the packets

        while (1)
        {
            /* press "ESC" to exit */
            if(Kbhit())
            {
                tmp = Getch();
                if( tmp == 27 || tmp == 'q')
                    break;
            }

/*-- (2) check that could the modem service, if it can't, skip operating the modem below --*/
            if(GM_SYS_CheckModemStatus() != GM_NOERROR)
                continue;

            if(GM_NET_GetLinkStatus(0)!=1)
                continue;
            else
                GM_NET_GetIP(myIP);

            /*--(3) send the data to server, and when LinkStatus[1]=1 --*/
            switch((netSendStatus=GM_SYS_CheckCmdStatus()))
            {
              case GM_READY:
                Print("sending package[%8ld]..., myIP = %s\r\n", socket_n, myIP);
                gprsData.link = 0;
                sprintf(gprsData.data, "-<%8ld>-TCP send test!!", socket_n);

                gprsData.dataLen = strlen(gprsData.data);
                if(GM_NET_Send(gprsData.link,gprsData.data,gprsData.dataLen)!=G
                M_NOERROR)
                    Print("can't send package[%8ld]\r\n");
                break;
```


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

```
                case GM_NOERROR:
                    Print("send success!!\r\n");
                    socket_n++;
                    break;
                case GM_BUSY:  /*sending, and waiting reply*/
                    break;
                default:
                    Print("send error, and re-send again, error code=%d\r\n",
                    netSendStatus);
                    break;
            }

            /*-- (4) if any new data packet come in, print it --*/
                if(GM_NET_GetNewPacket(&gprsData) != NULL)
                {
                    Print("\n== new data packet come in\r\n");
                    printPacket(gprsData);
                }
        }
        GM_NET_CloseLink(0); /*--Close client link[n], 3G:0~6, 2G:0 --*/
        DelayMs(1000);
        GM_NET_CloseNet();
        DelayMs(1000);

        break;
    }

}
```

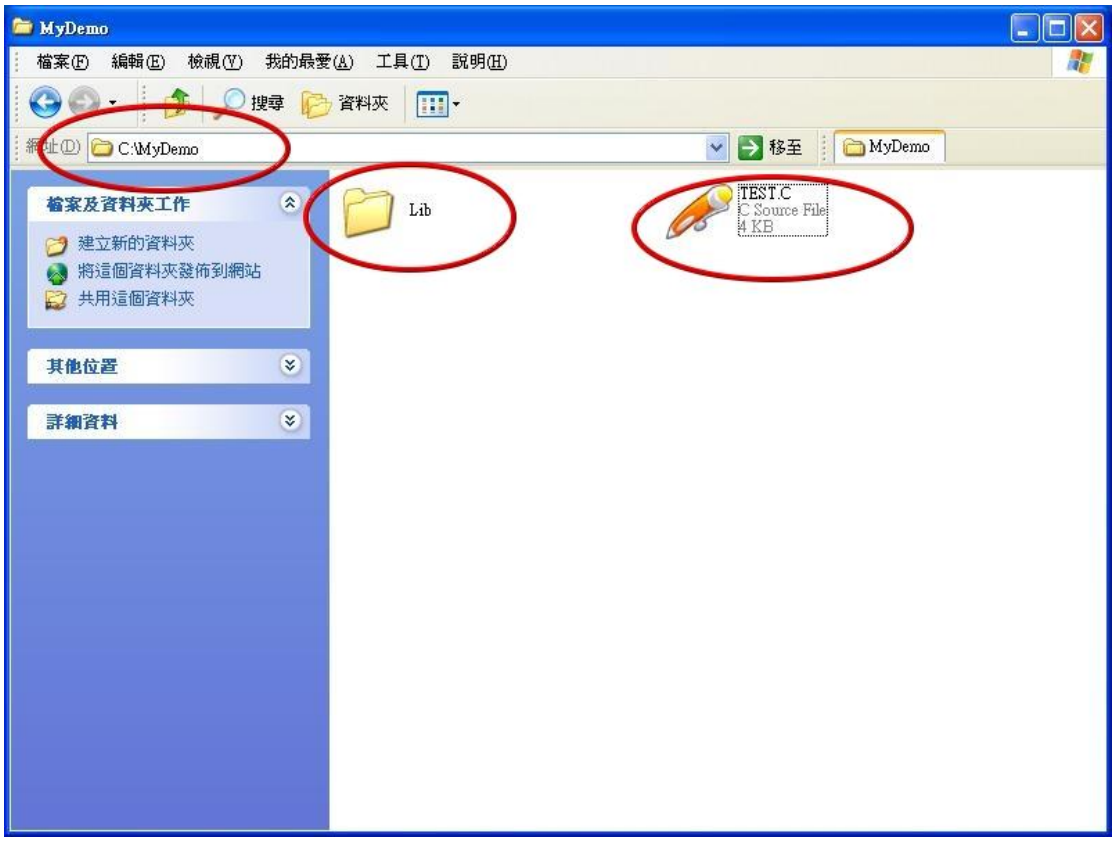

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

7. Program Download Procedure

Here, it is considered that how to build an execution file and how to run this program on the G-4513 series.

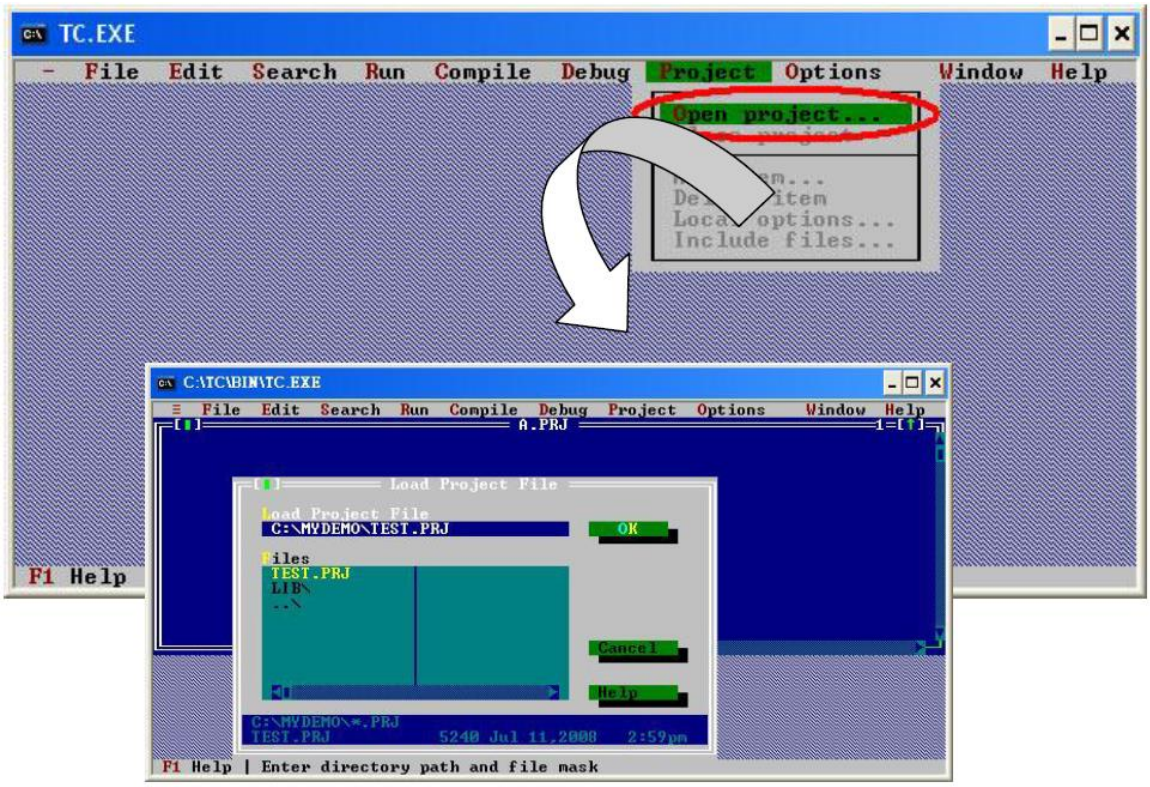
Library	Description
G4500.LIB	G-4513 and DI/O、AI functions
GSM_U2.LIB	GPRS functions
SD_Vnnn.LIB	MMC/SD functions
TCP_DM32.LIB	Ethernet functions
LCD.LIB	LCD functions

Step 1：Create a folder name "MyDemo" in the C disk, and copy the lib folder and users program into the MyDemo folder.

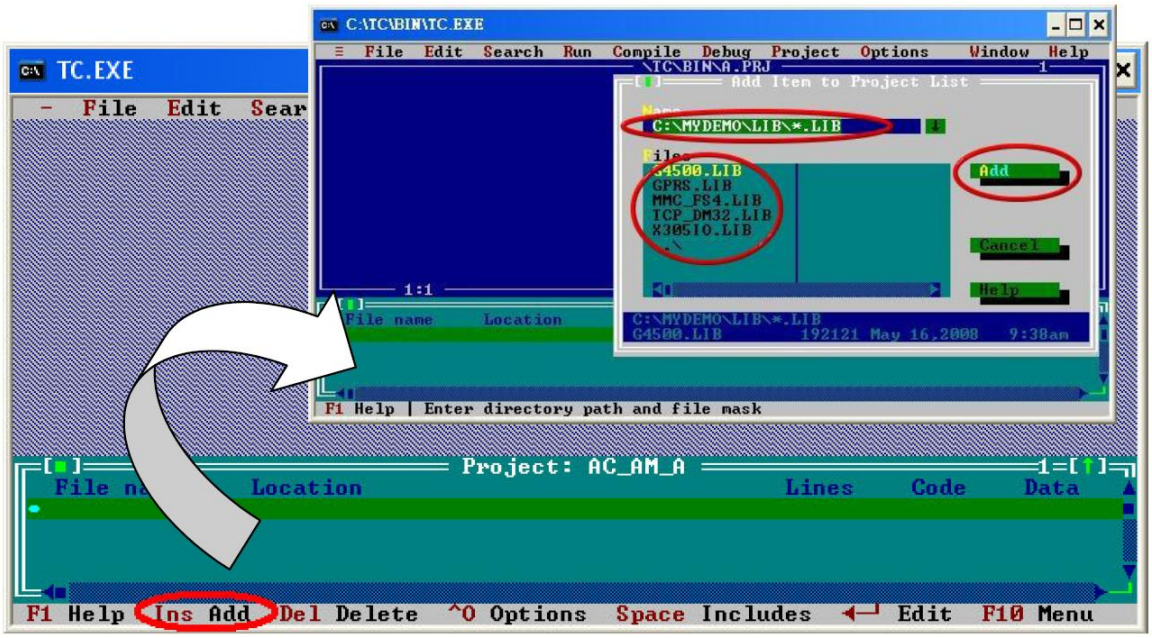


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 2：Run the TC++1.01development. Click the "Project\Open project…" create new project named "TEST.PRJ".

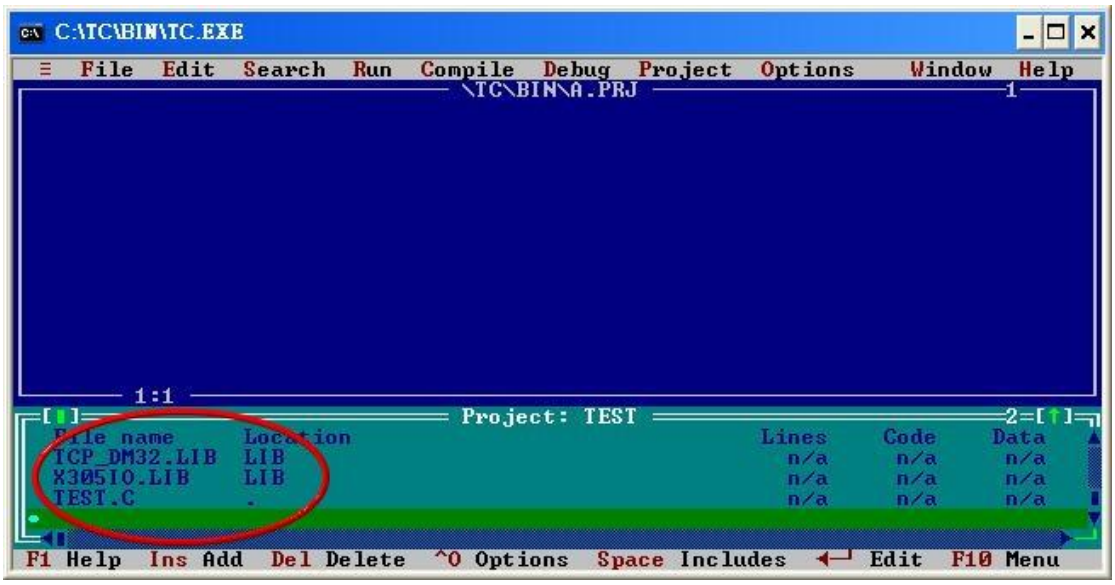


Step 3：Use the "Add" function to add the library file into MyDemo project.

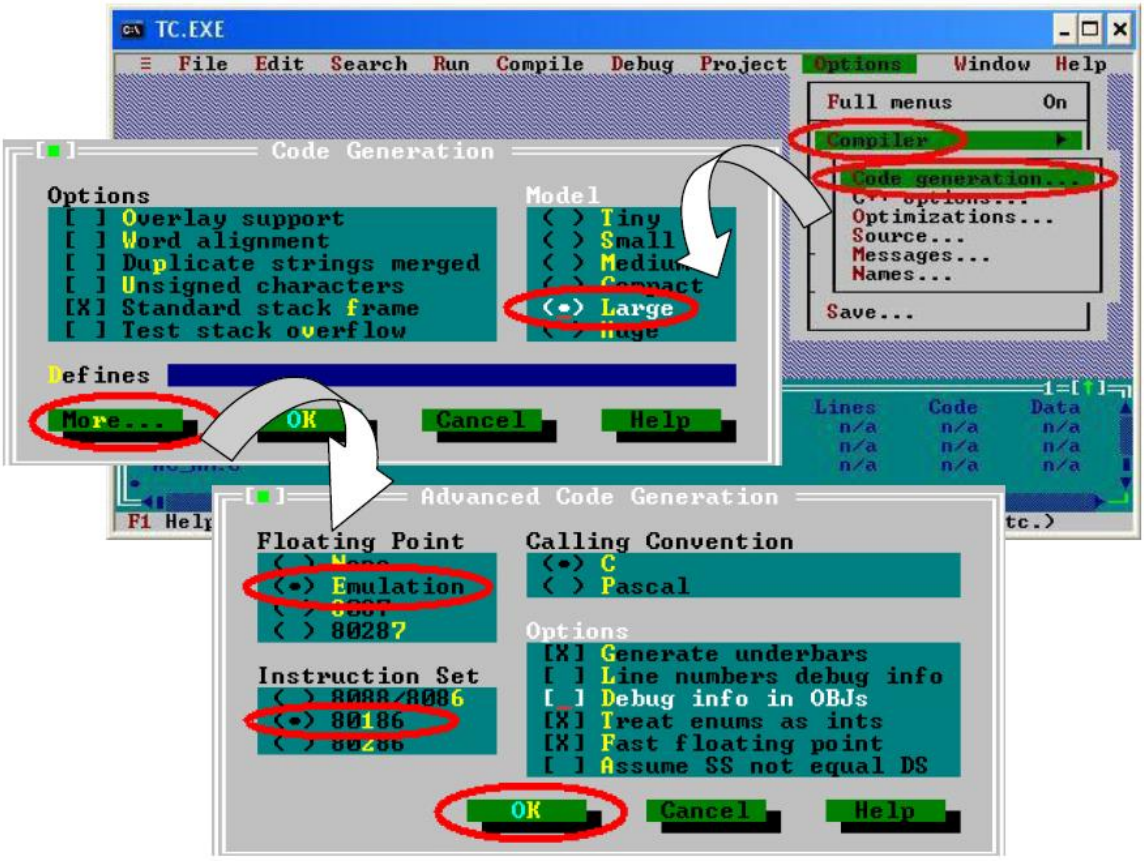


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 4：Following the step3, add another library and TEST.C into MyDemo project.

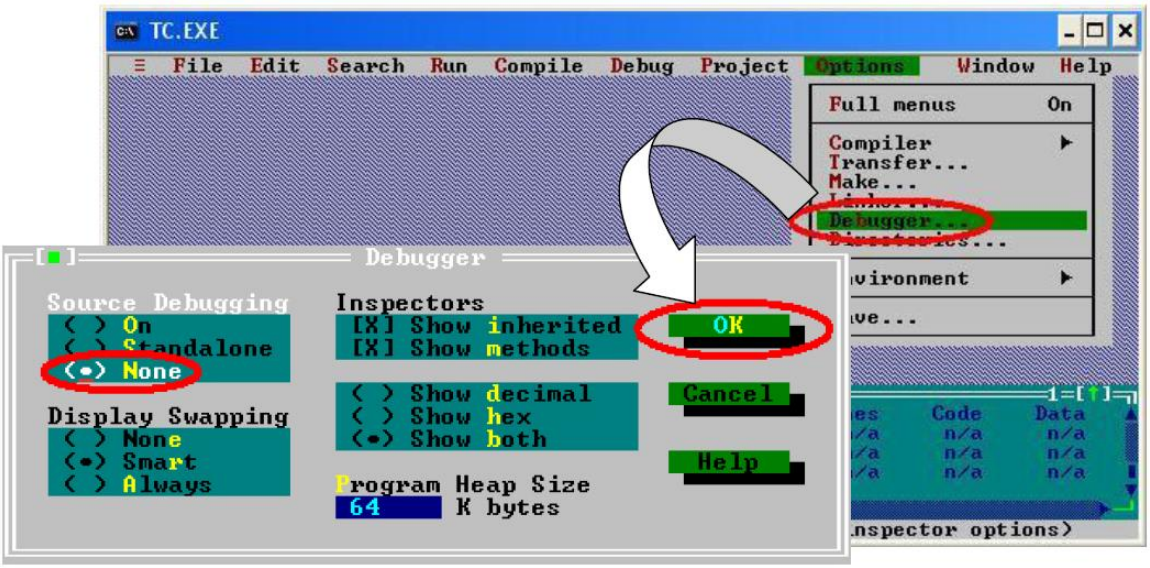


Step 5：Click the "Options/Compiler/Code generation…" to set the compile mode to the large mode. Click "More…" to set the "Floating point" and "Instruction Set" parameters. The Emulation and 80186 will be used respectively. Then, click OK button to save the configuration.

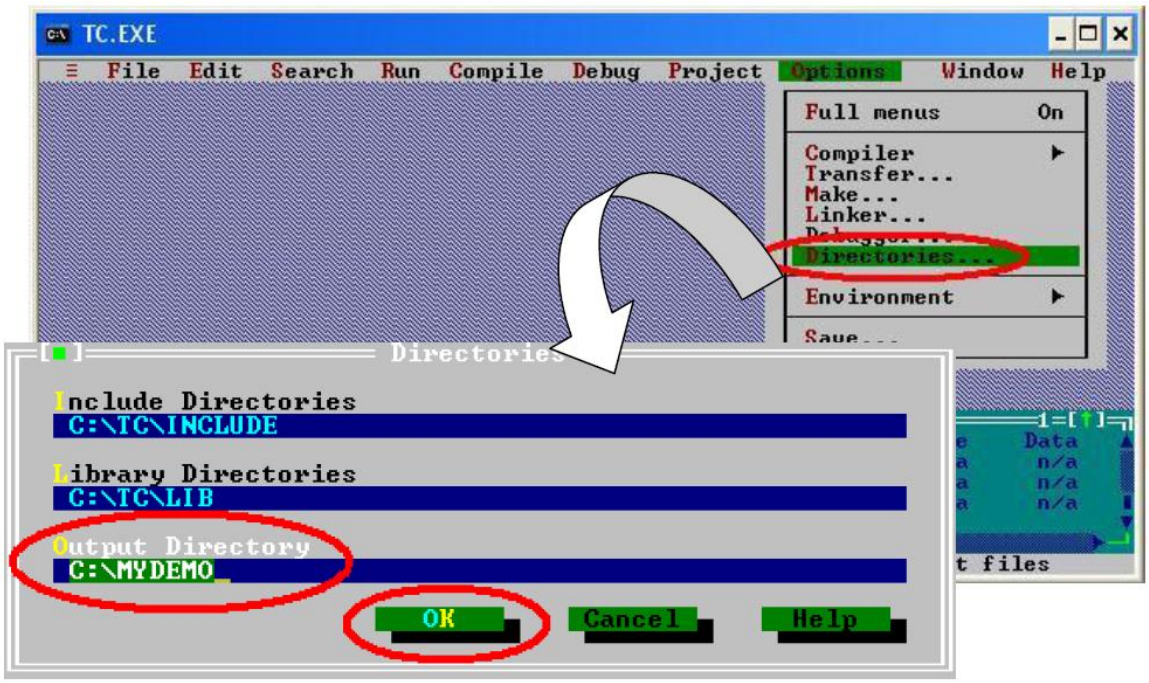


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 6：Click the "Option/Debugger... " to set the "Source Debugging" parameter. Here, select the "None" for the "Source Debugging".

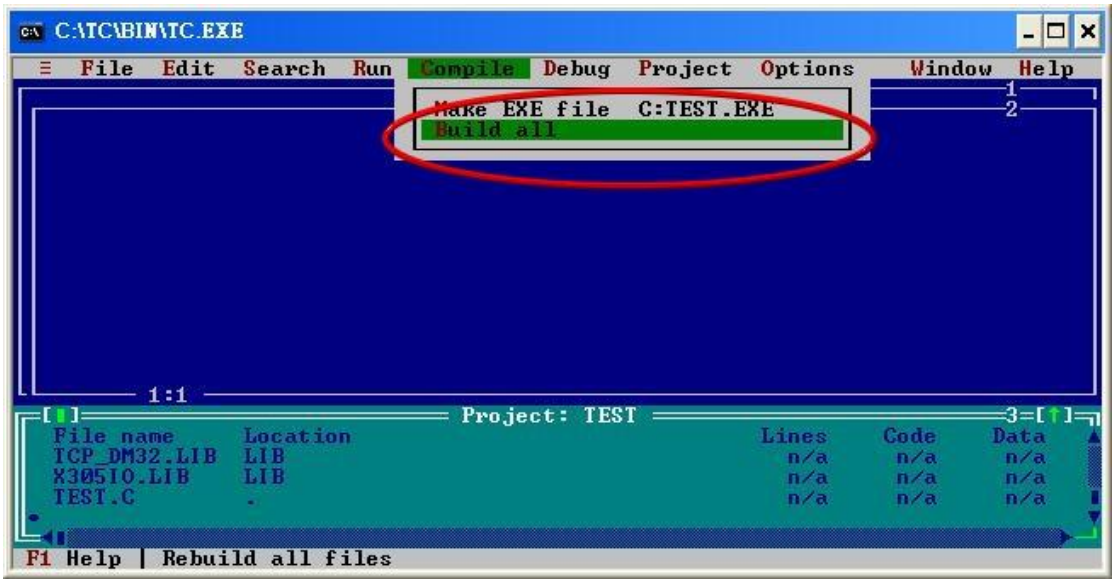


Step 7：Click the "Option/Directories... " to set the "Output Directory" parameter. Here, set the "C:\MyDemo" for the "Output Directory" parameter.

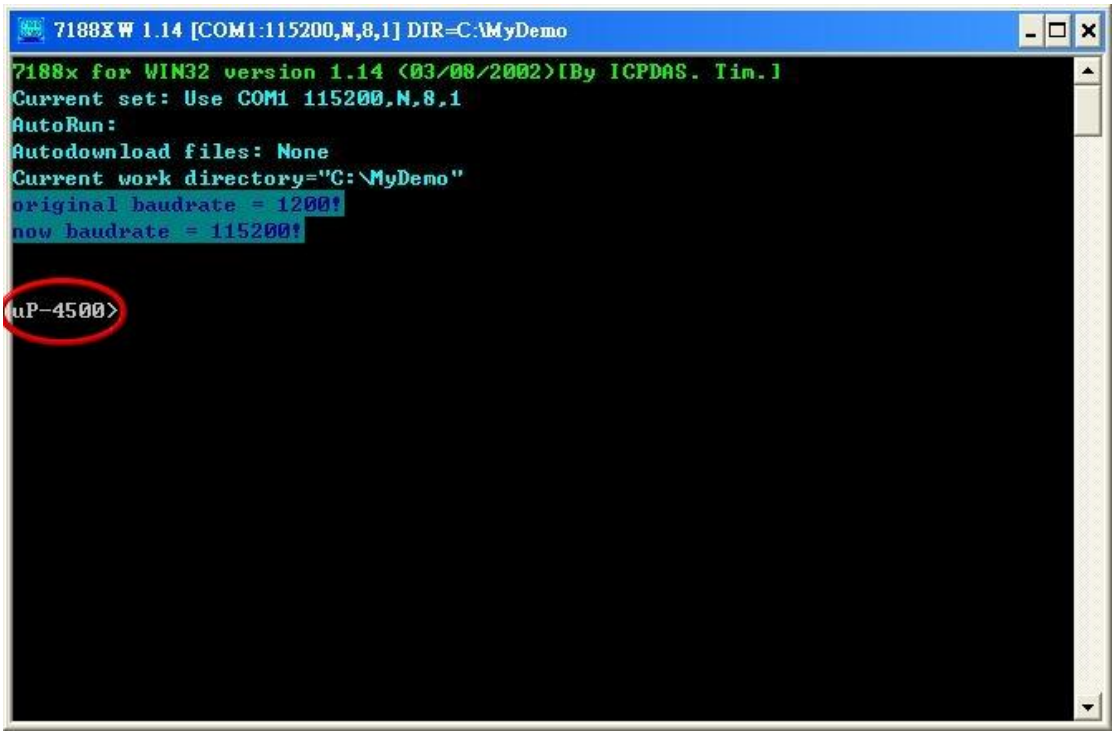


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 8：After finishing all the parameters setting, click the "Compile/build all" to produce the execution file name "TEST.exe".

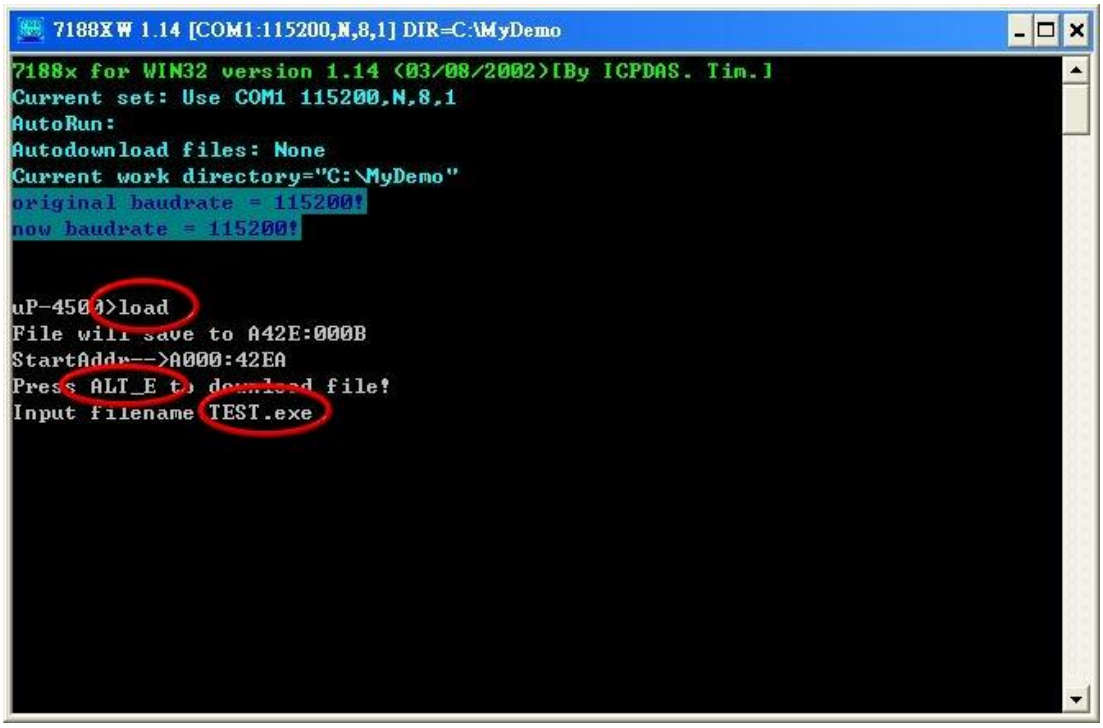


Step 9：Copy the file 7188XW.exe into the MyDemo folder. Then, double-click the 7188XW.exe file.The 7188XW.exe can be found in the Osimage folder. And G-4513 series COM1 connected to the PC RS-232.

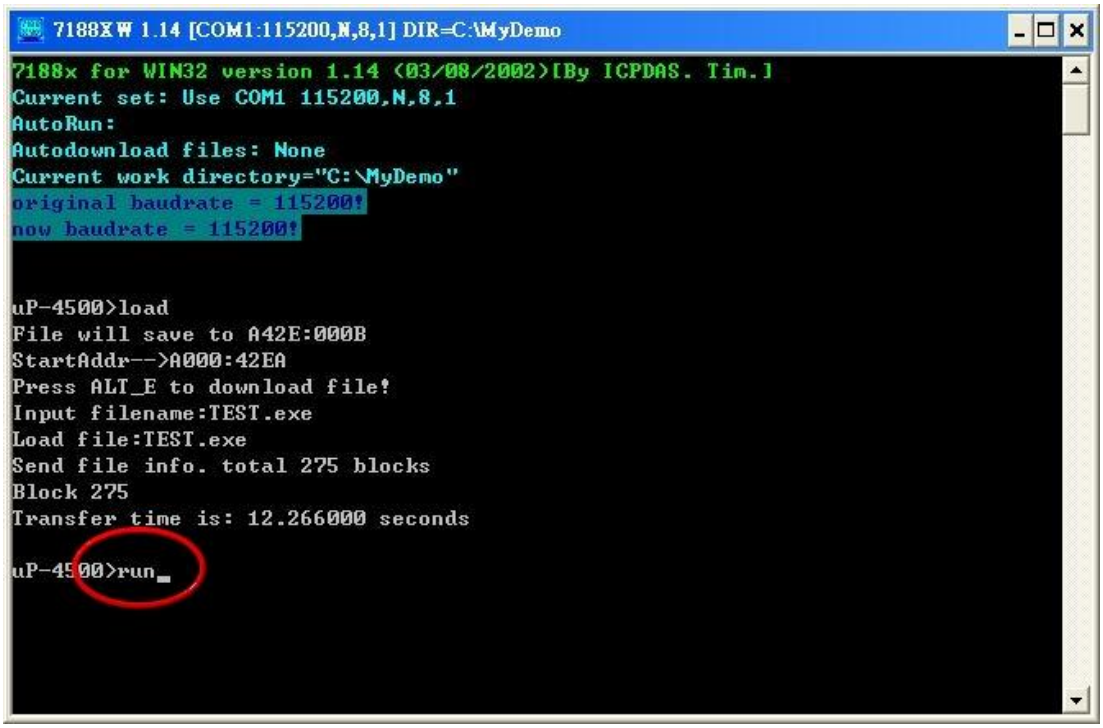


Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Step 10：Key the command, "load" in the 7188xw.exe program. Then, follow the hint command to press "Alt+E" and input the file name, "TEST.exe", to download the execution file.



Step 11：After finishing the download procedure, key in the command, "run", to implement the execution file, "TEST.exe".



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

8. Revision History

Revision	Date	Author	Description
1.0.0	2014/09/01	William	Release version
1.0.1	2015/06/15	William	1. Modify Chapter 6 2. Modify power consumption

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000