RAK13101 WisBlock GSM/GPRS Module Datasheet

Overview

Description

WisBlock RAK13101 GSM/GPRS Module is a part of the WisBlock Wireless series that provides GSM/GPRS capability on the WisBlock platform by using Quectel MC20CE cellular module.

Features

- Quectel MC20CE module
- Supports GSM/GPRS/GNSS: 850/900/1800/1900 MHz
- Supports BeiDou/GPS/GLONASS/QZSS
- Built-in LNA
- IPEX connectors for the GSM and GNSS antenna
- Micro-USB debug and log output connector
- Nano SIM and ESIM options
- 3.3 V Power supply
- Small size: 25 mm x 45 mm

Specifications

Overview

The RAK13101 module can be mounted on the IO slot of the WisBlock Base board. Figure 1 shows the mounting mechanism of the RAK13101 on a WisBlock Base module, such as a RAK5005-O.



Figure 1: RAK13101 mounting mechanism on a WisBlock Base module

Hardware

The hardware specification is categorized into five parts. It shows the chipset of the module and discusses the pinouts and its corresponding functions and diagrams. It also covers the electrical and mechanical parameters that include the tabular data of the functionalities and standard ratings of the RAK13101 WisBlock GSM/GPRS Module.

Chipset

Vendor

Part number

Quectel

MC20CE

Pin Definition

The RAK13101 WisBlock GSM/GPRS module comprises a standard WisBlock IO slot connector. The WisBlock connector allows the RAK13101 module to be mounted on a WisBlock baseboard with IO slot, such as RAK5005-O. The pin order of the connector and the pinout definition is shown in **Figure 2**.





Figure 2: RAK13101 Connector Pin Definition

📝 NOTE

- RAK13101 WisBlock IO slot connector utilizes the **UART** related pins, **PWRKEY** via WB_IO5, **VBAT**, **3V3**, and **GND**.
- **VBAT** is the battery voltage input with max voltage 4.2 V. During GPRS data and GSM dial mode, the peak current is 1170 mA and 769 mA, respectively, which exceeds the USB port supply current. That is why you must use a dedicated battery.

Electrical Characteristics

Absolute Maximum Ratings

Parameter	Minimum	Maximum	Unit
VBAT	-0.3	+4.2	V
GNSS_VCC	-0.3	+4.2	V
Power supply peak current0	0	2	A

Power Supply Ratings

Symbol	Description	Condition	Min.	Nom.	Max.	Unit
VBAT	Supply Voltage	Input voltage must within this range	3.3	4.0	4.2	V
GNSS_VCC	GNSS Part Supply voltage	Input voltage must within this range	2.8	3.3	4.2	V
Ipeak	Peak Current @ VBAT = 3.8 V	Idle	-	-	110	mA

Symbol	Description	Condition	Min.	Nom.	Max.	Unit
	Peak Current @ VBAT = 3.8 V	GSM Dial	-	-	769	mA
	Peak Current @ VBAT = 3.8 V	GPRS Data	-	-	1170	mA
lvbat	Average Supply Current	Power down mode	-	20	-	mA
	Average Supply Current	Sleep mode @DRX = 5	-	1.2	-	mA

Mechanical Characteristics Board Dimensions

Figure 3 shows the dimensions and the mechanical drawing of the RAK13101 module.



Figure 3: RAK13101 Mechanical Drawing

WisConnector PCB Layout



Figure 4: WisConnector PCB Footprint and Recommendations

Schematic Diagram

Power Supply

The Quectel MC20CE module's main power supply comes from **VBAT**, which is a battery voltage connected to the WisBlock Base board. **GNSS_VCC** is the GNSS section supply voltage and is turned on or off via **GNSS_VCC_EN**, which is connected to MC20CE. The power supply of the GNSS part is controlled via the AT command AT+QGNSSC.



Figure 5: RAK13101 Power Supply

GSM and GNSS

J3 is GSM antenna connector and J2 is GNSS antenna connector.



Figure 6: RAK13101 GSM and GNSS Antenna Circuit

Voltage Level Transfer

The GSM/GPRS module operates at 2.8 V, while the WisBlock Core is at 3.3 V. That is why a voltage level shifter is needed.



Figure 7: RAK13101 Voltage Level Shifter Circuit

SIM Circuit

Figure 8 shows the schematic of RAK13101 module as a slot for standard SIM card and an optional **ESIM** pcb footprint.



Figure 8: RAK13101 GSM/GPRS Module SIM Circuit

WisConnector

RAK13101 WisBlock IO slot connector utilizes **UART** related pins, **PWRKEY** via WB_IO5, **VBAT**, **3V3**, and **GND**.



Figure 9: RAK13101 IO Slot Connector

Debug Connector

Figure 10 shows the RAK13101 USB debugging circuit.



Figure 10: RAK13101 USB Debugging

LED/PWRKER Control Circuit

Figure 11 shows the RAK13101 module LED and PWRKEY controlled circuit.



Figure 11: RAK13101 Module LED PWRKEY Control Circuit

RAK13101 Quick Start Guide

Prerequisite

What Do You Need?

Before going through each and every step on using the RAK13101 WisBlock module, make sure to prepare the necessary items listed below:

Hardware

- RAK13101
- Your choice of WisBlock Base I
- Your choice of WisBlock Core
 [™]
- USB Cable
- GNSS and GSM Antennas
- Li-Ion/LiPo battery ☑
- Solar charger (optional) [⊥]

Software

- Download and install Arduino IDE
- To add the RAKwireless Core boards on your Arduino board, install the RAKwireless Arduino BSP. Follow the steps in the GitHub repo □ .

Product Configuration

Hardware Setup

You can integrate the RAK13101 module on your WisBlock project to extend its functionality and have GSM/GPRS/GNSS capability. This is ideal for tracking applications with GSM/GPSR cellular connectivity in the area. For more information about RAK13101, refer to its Datasheet.

RAK13101 module can be mounted to the IO slot of the WisBlock Base and communicates to the WisBlock Core via UART. The module is activated via WB_105 pin of the WisBlock Core. Two antennas must be connected to the module as well, one for the GNSS antenna port and one for the GSM antenna port. An external battery (Li-Ion/LiPo 3.7-4.2V) is also required to power up the module properly.

- Batteries can cause harm if not handled properly.
- Only 3.7-4.2 V Rechargeable LiPo batteries are supported. It is highly recommended not to use other types of batteries with the system unless you know what you are doing.
- If a non-rechargeable battery is used, it has to be unplugged first before connecting the USB cable to the USB port of the board to configure the device. Not doing so might damage the battery or cause a fire.
- Only 5 V solar panels are supported. Do not use 12 V solar panels. It will destroy the charging unit and eventually other electronic parts.
- Make sure the battery wires match the polarity on the WisBlock Base board. Not all batteries have the same wiring.



Figure 1: RAK13101 connection to WisBlock Base



Figure 2: WisBlock Base RAK5005-O battery polarity and connection

Assembling and Disassembling of WisBlock Modules Assembling

As shown in **Figure 3**, the location for Slot A, B, C, and D are properly marked by silkscreen. Follow carefully the procedure defined in RAK5005-O module assembly/disassembly instructions I to attach a WisBlock module. Once attached, carefully fix the module with one or more pieces of M1.2 x 3 mm screws depending on the module.



Figure 3: RAK13101 connection to WisBlock Base

Disassembling

The procedure in disassembling any type of WisBlock modules is the same.

1. First, remove the screws.



Figure 4: Removing screws from the WisBlock module

2. Once the screws are removed, check the silkscreen of the module to find the correct location where force can be applied.



Figure 5: Detaching silkscreen on the WisBlock module

3. Apply force to the module at the position of the connector, as shown in **Figure 6**, to detach the module from the baseboard.



Figure 6: Applying even forces on the proper location of a WisBlock module

NOTE

If you will connect other modules to the remaining WisBlock Base slots, check on the WisBlock Pin Mapper 12 tool for possible conflicts. RAK13101 uses UART communication lines, and it can cause possible conflict especially on some modules that also use UART.

Software Configuration and Example

The RAK13101 WisBlock GSM/GPRS Module uses UART serial communication lines. In this example code, you will be able to send AT commands to the RAK13101 module. This will ensure that your RAK13101 is functional and ready for your IoT project.

Initial Test of the RAK13101 WisBlock Module

If you already installed the RAKwireless Arduino BSP 🖸 , the WisBlock Core and example code should now be available on the Arduino IDE.

1. You need to select first the WisBlock Core you have, as shown in Figure 7 to Figure 9.

iie Edir Sketh Tools Help Adrohe Sketh Sketh Loo21 eeup () // put yout Srial Nontor Ctrl-Shift-I Serial Montor Ctrl-Shift-I Serial Montor Ctrl-Shift-I Serial Montor Ctrl-Shift-I Serial Shoter Ctrl-Shift-I SoftDevice "5140 6.1.1" Debug" Level 0 (Release)" Port Get Board 'WisBlock RAK4631 Programmer Bum Bootloader
Image: Second
sketch_nov21e Fik Encoding & Reload void setup () Manage Libraries Ctrl - Shift - I serial Monitor Ctrl - Shift - I serial Plotter Ctrl - Shift - I void 1cop () Wift101 / WiftNINA Firmware Updater // put yout Serial Plotter Board: WieBlock RAK4631 Boards Manager SoftDevice: "S140 6.1.1" Advine AVR Boards Port RAKwireless RSP32 Modules Port Rakwireless Respberry Modules Programmer Burn Bootloader
Skelin (1902) is fire Encoding & Reload Manage Libraries Ctrl-Shift-I Serial Monitor Ctrl-Shift-I Serial Plotter Ctrl-Shift-I Serial Plotter Ctrl-Shift-I Serial Plotter Ctrl-Shift-I SoftDevice: 'S140 6.1.1' SoftDevice: 'S140 6.1.1' Debug "Level O (Relesse)' Port RAKwireless Raspberry Modules WisBlock RAK4631 Programmer S Bum Boottoader
Manage Libranes Ctrl-Shift-IL Serial Monitor Ctrl-Shift-L void Joop () [] WiFi01 / WiFINIAA Firmware Updater // put: your Board: "WiSBlock RAK4631" Board: "WiSBlock RAK4631" Boards Manager // put: your SoftDevice: "St40 6.1.1" Debug: "Level 0 (Reless)" RAKwireless ESP32 Modules Port Get Board Info Rakwireless Raspberry Modules WisBlock RAK4631 Programmer > Burn Bootloader >
Senial Montor Ctrl+Shift-L Senial Montor Ctrl+Shift-L Vid Loop () [Vid Loop () [Vid Kiblock RAK4631* Boards Manager Arduino AVR Boards Debug: "Level 0 (Release)" Port Get Board Info Programmer Burn Bootloader VisBlock RAK4601 Programmer Burn Bootloader
<pre>void loop() [void loop()</pre>
void loop() [// put your } SoftDevice "SI40 6.1.1" Debug:"Level (Release)" Port Get Board Info Programmer Burn Bootloader
Board: WisBlock RAK4631* Board: Manager SoftDevice: "SI40 6.1.1* Arduino AVK Boards Pott RAKwireless SP32 Modules Port RAKwireless Raspberry Modules Get Board Info Rakwireless Raspberry Modules Programmer > Burn Boottoader >
3 SoftDevice: "S140 6.1.1" Poet Poet G(Relesse)" Poet Geseard Info Programmer Burn Bootloader Burn Bootloader Statistical Rational Statistical Stati
Debug: "Level 0 (Release)" RAKwireless SP32 Modules Port RAkwireless RAF Modules Get Board Info Rakwireless Raspberry Modules Programmer Bum Bootloader
Port Image: Control of the set
Get Board Info Rakwireless Raspberry Modules WirsBlock RAK4601 Programmer > Burn Bootloader
Programmer > Burn Bootloader
Burn Bootloader

Figure 7: Selecting RAK4631 as WisBlock Core

sketch_nov21a A	Arduino 1.8.16		-	٥	×
File Edit Sketch To	Auto Format	Ctrl+T			Ø
	Archive Sketch				
sketch_nov21a	Fix Encoding & Reload				
<pre>void setup()</pre>	Manage Libraries	Ctrl+Shift+I			^
// put your	Serial Monitor	Ctrl+Shift+M	1		
}	Serial Plotter	Ctrl+Shift+L			
void loop() [WiFi101 / WiFiNINA Firmware Updater				
// put your	Board: "WisCore RAK11200 Board"	;	Boards Manager		
}	Upload Speed: "921600"	2	Arduino AVR Boards		
	Flash Frequency: "80MHz"	2	RAKwireless ESP32 Modules VisCore RAK11200 Board		
	Flash Mode: "QIO"	2	RAKwireless nRF Modules		
	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)	r >	Rakwireless Raspberry Modules >		
	Core Debug Level: "None"	>			
	Port	>	>		
	Get Board Info				
	Programmer	>	>		
	Burn Bootloader				
					~
9			WisCore RAK11200 Board, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), QIO, 80MHz, 921800, Non	e on COM	112

Figure 8: Selecting RAK11200 as WisBlock Core

sketch_nov21a Ar File Edit Sketch Too	rduino 1.8.16					-	٥	×
sketch nov21a	Auto Format Archive Sketch	Ctrl+T						•
<pre>void setup() // put your }</pre>	Manage Libraries Serial Monitor Serial Plotter	Ctrl+Shift+I Ctrl+Shift+M Ctrl+Shift+L						,
void loop()	WiFi101 / WiFiNINA Firmware Upo	later						
// put your	Board: "WisBlock RAK11300"		Boards Manager	-				
}	Port Get Board Info	1	Arduino AVR Boards > RAKwireless ESP32 Modules >					
	Programmer Burn Bootloader	3	RAKwireless nRF Modules > Rakwireless Raspberry Modules >	WisBlock RAK11300				
9						WisBlock RAK1130	0 on CON	112

Figure 9: Selecting RAK11300 as WisBlock Core

2. Next, copy the following sample code into your Arduino IDE:

```
@author rakwireless.com
   @version 0.1
#include <Wire.h>
#define POWER_KEY WB_I05
void setup()
    time_t serial_timeout = millis();
   Serial.begin(115200);
   while (!Serial)
        if ((millis() - serial_timeout) < 5000)</pre>
            delay(100);
        else
            break;
    Serial.println("AT CMD TEST!");
   time_t timeout = millis();
   bool moduleSleeps = true;
   Serial1.begin(9600);
   delay(1000);
   Serial1.println("ATI");
   while ((millis() - timeout) < 6000)</pre>
        if (Serial1.available())
            String result = Serial1.readString();
            Serial.println("Modem response after start:");
            Serial.println(result);
            moduleSleeps = false;
   if (moduleSleeps)
    pinMode(POWER_KEY, OUTPUT);
        digitalWrite(POWER_KEY, 0);
        delay(200);
        digitalWrite(POWER_KEY, 1);
        delay(2000);
        digitalWrite(POWER_KEY, 0);
        delay(1000);
    Serial.println("MC20 power up!");
```

```
void loop()
   int timeout = 100;
   String resp = "";
   String snd = "";
   char cArr[128] = {0};
   while (timeout--)
        if (Serial1.available() > 0)
            resp += char(Serial1.read());
        if (Serial.available() > 0)
            snd += char(Serial.read());
       delay(1);
   if (resp.length() > 0)
        Serial.print(resp);
    }
    if (snd.length() > 0)
       memset(cArr, 0, 128);
        snd.toCharArray(cArr, snd.length() + 1);
       Serial1.write(cArr);
       delay(10);
    }
    resp = "";
    snd = "";
```

3. You can now select the right serial port and upload the code, as shown in Figure 10 and Figure 11.

NOTE:

If you are using RAK11200 as WisBlock Core, you need to configure the BOOT0 pin before uploading. You need to short it to the ground then press the reset button of the WisBlock Base before releasing the BOOT0 pin. If not done properly, uploading the source code to RAK11200 will fail. Check the full details on the RAK11200 Quick Start Guide.

💿 sketch nov22a l	Arduino 1.8.16		-	×
File Edit Sketch To	ools Help			
	Auto Format	Ctrl+T		0
	Archive Sketch			
sketch_nov22a	Fix Encoding & Reload			
/**	Manage Libraries	Ctrl+Shift+I		^
Gille Unva Gauthor ra	Serial Monitor	Ctrl+Shift+M		
@brief unv	Serial Plotter	Ctrl+Shift+L		
@version 0 @date 2021	WiFi101 / WiFiNINA Firmware Update	r		
@copyright	Board: "WisBlock RAK4631"	>		
~~/	SoftDevice: "S140 6.1.1"	>		
<pre>#include <wir< pre=""></wir<></pre>	Debug: "Level 0 (Release)"	>		
#define POWER	Port: "COM13 (WisBlock RAK4631)"	2	Serial ports	
	Get Board Info		COM13 (WisBlock RAK4631)	
<pre>void setup() </pre>	Programmer	>		
time_t seri	Burn Bootloader			
Serial.begin(115200);			
while (!Seria	1)			
ł				
if ((millis	<pre>() - serial_timeout) < 5000)</pre>			
del	ay(100);			
}				
{				
bre	ak;			
}				
1				~

Figure 10: Selecting the correct Serial Port

💿 rak13101 Arduino 1.8.16	_	. 🗆	×
<u>File_Edit_Sketch_Iools_H</u> elp			
			ø
rak13101			
/** @file Unvarnished_Transmission.ino @author rakwireless.com @brief unvarnished transmission via USB @version 0.1 @date 2021-6-28 @copyright Copyright (c) 2020 **/			<
<pre>#include <wire.h></wire.h></pre>			
<pre>#define POWER_KEY WB_I05</pre>			
<pre>void setup() { time_t serial_timeout = millis(); Serial.begin(115200); while (!Serial) { if ((millis() - serial_timeout) < 5000) { delay(100); } else { break; } } }</pre>			*
Done uploading.			
Device programmed. DFU upgrade took 3.2168917655944824s			^
			> `
6 WisBlock RAK4831.	S140 6.1.1, Level 0 (F	telease) on C	ом13

Figure 11: Uploading the RAK13101 example code

📝 NOTE

If you experience any error in compiling the example sketch, check the updated code for the RAK13101 Module that can be found on the WisBlock Example Code Repository

4. When you successfully uploaded the example sketch, open the Serial Monitor of the Arduino IDE to see the initial logs, as shown in **Figure 12**. If you do not see any logs, you can try to disconnect the USB cable and

battery, then reconnect again with the battery first. If you see that the LED of the RAK13101 is blinking after a few seconds, the module is now initialized properly.

💿 rak13101 Arduino 1.8.16		- 0	×
<u>F</u> ile Edit Sketch <u>T</u> ools <u>H</u> elp			
			ø
rak13101			
<pre>void setup() { time_t serial Serial.begin AT CMD TEST! Modem response after start: ATI (if ((milli</pre>	- C X		~
<pre>else {</pre>			~
Autoscroll Show timestamp	Both NL & CR 🗸 I15200 baud 🧹 Clear output		
Device programmed. DFU upgrade took 3.2168917655944824s			~
13	WisBlock RAK4831, S140 6.1.1, Lo	evel O (Release) on	сом13

Figure 12: RAK13101 initial logs

There are times that you might not see the initial logs if you open the Serial Monitor. The best way to test if the module is working is by sending actual AT commands, as shown in Figure 13. You can try to send the basic commands, AT and ATI.

🐵 rak13101 Arduino 1.8.16			_	\Box \times	1
File Edit Sketch Tools Help					
				P	
rak13101					
void setup() COM13	-	o x]	,	~
time t serial Serial.begin ar		Send		- 1	
while (!Seri ATI { if ((milli) Quectel_Ltd					
{ Quectel_MC20 Revision: MC20CER01A04 de J or					
else or for the second					
}					
Serial.print: // Check if t time t timeou					
Serial1.begir delay(1000);					
Serial1.print //MC20 init while ((mill:					
{ if (Serial: { }					~
Autoscroll Show timestamp Both NL & CR	115200 baud \lor	Clear output			
Device programmed. DFU upgrade took 3.2168917655944824s					< >
13	WisBlock RAK4	631, S140 6. <u>1.1, L</u>	evel 0 (Rel <u>eas</u> e	> e) on COM1 <u>3</u>	

