# **RAK13002 Quick Start Guide**

#### **Prerequisite**

# What Do You Need?

Before going through each and every step on using the RAK13002 WisBlock module, make sure to prepare the necessary items listed below:

#### Hardware

- RAK13002 Wisblock IO Module
- Your choice of WisBlock Base ☑
- Your choice of WisBlock Core 
  <sup>™</sup>
- USB Cable
- Li-Ion/LiPo battery (optional)
- Solar charger (optional)

#### Software

- Download and install ArduinoIDE C .
- To add the RAKwireless Core boards on your Arduino Boards Manager, install the RAKwireless Arduino BSP

# **Product Configuration**

# Hardware Setup

The RAK13002 WisBlock IO Module is designed as an IO extension module that allows you to connect external digital and analog modules to create a customized IoT solution. It supports two (2) I2C interfaces, two (2) UART interfaces, one (1) SPI Interface, six (6) GPIOs, and two (2) ADC interfaces. For more information about RAK13002, refer to the Datasheet.

The RAK13002 WisBlock IO Module can be mounted on the IO slot of the WisBlock Base board, as shown in **Figure 1**. Also, always secure the connection of the WisBlock module by using compatible screws.



Figure 1: RAK13002 connection to WisBlock Base

#### Assembling and Disassembling of WisBlock Modules

#### Assembling

As shown in **Figure 2**, the location for the IO slot is properly marked by silkscreen. Follow carefully the procedure defined in RAK5005-O module assembly/disassembly instructions  $\Box$  to attach a WisBlock module. Once attached, carefully fix the module with three pieces of M1.2 x 3 mm screws.



Figure 2: RAK13002 assembly to WisBlock Base

#### Disassembling

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

1. First, remove the screws.



Figure 3: 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 4: Detaching silkscreen on the WisBlock module

3. Apply force to the module at the position of the connector, as shown in **Figure 5**, to detach the module from the baseboard.



Figure 5: 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.

After all this setup, you can now connect the battery (optional) and USB cable to start programming your WisBlock Core.

#### **Software Configuration and Example**

The RAK13002 module exposes the IO pins, SPI, I2C, and UART communication ports. You can use these ports to connect sensors or modules, digital I/O, analog I/O, and slave devices. These ports are routed to the WisBlock Core through the IO connector.

For RAK13002, the accessible GPIO pin assignments are defined as follows in the Arduino IDE:

- WB\_I01 for IO1, GPIO1 pin
- WB\_I02 for IO2, GPIO2 pin
- WB\_I03 for IO3, GPIO3 pin
- WB\_I04 for IO4, GPIO4 pin
- WB\_I05 for IO5, GPIO5 pin
- WB\_I06 for IO6, GPIO6 pin
- WB\_SW1 for SW1 pin
- WB\_A0 for AIN1, ADC Input pin
- WB\_A1 for AIN1, ADC Input pin

# **BAK**<sup>®</sup> Documentation Center

Row/Column	Column 1 Column 2		Column 3	Column 4
Row 1	VCC	VCC	VCC	VCC
Row 2	GND	GND	GND	GND
Row 3	101	SCL1	TXD0	CS
Row 4	103	SDA1	RXD0	SDI
Row 5	IO4	SCL2	TXD1	SDO
Row 6	105	SDA2	RXD1	SCK
Row 7	106	LED1	AINO	RST
Row 8	107	LED2	AIN1	SW1

#### I2C Connection on RAK13002

This is just an example and illustration on how to use the RAK13002 for external I2C sensors, modules, or devices. You can use any I2C device as long as it operates at 3.3 V.



Figure 6: Connecting the RAK13002 to the I2C backpack of a 16x2 LCD

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

💿 sketch_aug22b	Arduino 1.8.15						-	đ	$\times$
File Edit Sketch To	ols Help								
	Auto Format	Ctrl+T							<u>.</u>
	Archive Sketch								
sketch_aug2:	Fix Encoding & Reload								
1 void s	Manage Libraries	Ctrl+Shift+I							^
2 // 2	Serial Monitor	Ctrl+Shift+M							
2 // P	Serial Plotter	Ctrl+Shift+L	ice.						
3									
4 }	WiFi101 / WiFiNINA Firmware Updater								
5	Board: "WisBlock Core RAK4631 Board"	>	Boards Manager						
6 void 1	Bootloader: "0.3.2 SoftDevice s140 6.1.1	" >	Arduino AVR Boards	>					
7 // p	Debug: "Level 0 (Release)"	>	Arduino SAMD (32-bits ARM Cortex-M0+) Boards	>					
8	Port: "COM4"	>	ESP8266 Boards (3.0.2)	>					
9}	Get Board Info		ESP8266 Boards (3.1.0-dev)	>					
- ,			RAKwireless ESP32 Modules	>					
	Programmer	,	RAKwireless nRF Modules	•	WisBlock Core RAK4631 Board				
	Burn Bootloader				WisBlock Core RAK4601 Board				
									~
7					WisBlock Co	ore RAK4631 Board, 0.3.2 SoftDevice s140 6.1.1, L	evel 0 (Re	lease) on C	COM4



sketch_aug22c	Arduino 1.8.15				-	٥	×
File Edit Sketch To	pols Help						
	Auto Format	Ctrl+T					<b>9</b>
	Archive Sketch						
sketch_aug2:	Fix Encoding & Reload						•
1 void s	Manage Libraries	Ctrl+Shift+I					^
2 // n	Serial Monitor	Ctrl+Shift+M					
2 // P	Serial Plotter	Ctrl+Shift+L					
4 }	WiFi101 / WiFiNINA Firmware Updater						
5	Board: "WisCore RAK11200 Board"	>	Boards Manager				
6 void 1	Upload Speed: "921600"	>	Arduino AVR Boards	>			
7 // p	Flash Frequency: "80MHz"	>	Arduino SAMD (32-bits ARM Cortex-M0+) Boards	>			
8	Flash Mode: "QIO"	>	ESP8266 Boards (3.0.2)	>			
9}	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"	·	ESP8266 Boards (3.1.0-dev)	>			
	Core Debug Level: "None"	>	RAKwireless ESP32 Modules	WisCore RAK11200 Board			
	Port: "COM4"	>	RAKwireless nRF Modules	>			
	Get Board Info						
	Programmer	>					
	Burn Bootloader						
							~
1			WisCore RAK11200 Board, D	efault 4MB with spiffs (1.2MB APP/1.5MB	SPIFFS), QIO, 80MHz, 921600	, None on CC	DM4

Figure 8: Selecting RAK11200 as WisBlock Core

						_	
Sketch_oct29a /	Arduino 1.8.15				_		X
	Auto Format	Ctrl+T					<mark>.</mark>
	Archive Sketch						
sketch_oct29	Fix Encoding & Reload	Chill Chiffe I					
l void s	Social Monitor	Ctrl+Shift+M					
2 // p	Serial Plotter	Ctrl+Shift+I	once:				
3	Senar Piotter	Cui+Shint+L					
4 }	WiFi101 / WiFiNINA Firmware Update	r					
5	Board: "WisBlock RAK11300"	>	Boards Manager				
6 void l	Port: "COM10 (WisBlock RAK11300)"	>	Arduino AVR Boards				
7 // p	Get Board Info		Arduino SAMD (32-bits ARM Cortex-M0+) Boards >				
8	Programmer	,	ESP8266 Boards (3.0.2)				
9 }	Burn Bootloader		ESP8266 Boards (3.1.0-dev)				
	bum bootouder		RAKwireless ESP32 Modules				
			RAKwireless nRF Modules				
			Rakwireless Raspberry Modules  VisBlock RAK11300				
							~
1				Wist	Block RAK1	300 on C	OM10

Figure 9: Selecting RAK11300 as WisBlock Core

2. On the Arduino IDE, go to **Sketch > Include Library > Manage Libraries**. The Library Manager should open, then install the LiquidCrystal I2C 🖸 library, as shown in **Figure 10**.

.,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	✓ Topic All	<ul> <li>liquid crystal i2c</li> </ul>	
graphic displa	ays (u8glib),		, , , , ,
			Version 2.3.5 V Install
LiquidCrysta	l I2C		
by Marco Sc A library for MIGHT NOT I More info	hwartz r <b>I2C LCD displays.</b> The lib BE COMPATIBLE WITH EXIS <sup>-</sup>	rary allows to control I2C displays with functions extremely similar to Liquid ING SKETCHES.	dCrystal library. THIS LIBRARY
LiquidCrysta	AIP31068		
• •			
by Andriy G A library for to LiquidCrys More info	olovnya r AIP31068 I2C/SPI LCD stal library. THIS LIBRARY MI	displays. The library allows to control AIP31068 based I2C/SPI displays v GHT NOT BE COMPATIBLE WITH EXISTING SKETCHES.	vith functions extremely similar
by Andriy G A library for to LiquidCrys <u>More info</u>	olovnya r <b>AIP31068 12C/SPI LCD</b> stal library. THIS LIBRARY MI	displays. The library allows to control AIP31068 based I2C/SPI displays v GHT NOT BE COMPATIBLE WITH EXISTING SKETCHES.	vith functions extremely similar
by Andriy G A library for to LiquidCrys More info	olovnya r AIP31068 12C/SPI LCD stal library. THIS LIBRARY MI ILI2C_Hangul	displays. The library allows to control AIP31068 based I2C/SPI displays v GHT NOT BE COMPATIBLE WITH EXISTING SKETCHES.	vith functions extremely similar
by Andriy G A library for to LiquidCrys More info LiquidCrysta by Junwha H	olovnya r AIP31068 12C/SPI LCD ttal library. THIS LIBRARY MI al_12C_Hangul Hong	displays. The library allows to control AIP31068 based I2C/SPI displays v GHT NOT BE COMPATIBLE WITH EXISTING SKETCHES.	vith functions extremely similar
by Andriy G A library for to LiquidCrys More info LiquidCrysta by Junwha H A library for library. This L	olovnya r AIP31068 I2C/SPI LCD ttal library. THIS LIBRARY MI nI_I2C_Hangul Hong r printing Hangul on I2C L Library allows to print hangu	displays. The library allows to control AIP31068 based I2C/SPI displays v GHT NOT BE COMPATIBLE WITH EXISTING SKETCHES. CD displays. The library allows to control I2C displays with functions extre on LCDs.	with functions extremely similar

Figure 10: Installing the LiquidCrystal I2C library

3. After successful installation of the library, you can now copy the following sample code into your Arduino IDE:

```
#include LiquidCrystal_I2C.h
#include Wire.h
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup() {
 lcd.init();
 lcd.backlight(); // turn on the backlight
}
void loop() {
 start_display(); // star
 delay(1000); //wait for a second
 lcd.clear(); // clear the LCD content
 delay(1000); //wait for a second
void start_display(){
 lcd.setCursor(0,0); // tell the screen to write on the top row
 lcd.print("RAK13002"); // tell the screen to write "RAK13002" on the top row
 lcd.setCursor(0,1); // tell the screen to write on the bottom row
 lcd.print("EXAMPLE"); // tell the screen to write "EXAMPLE" on the bottom row
```

4. Then select the right Serial Port and upload the code, as shown in Figure 11 and Figure 12.



Figure 11: Selecting the correct Serial Port



Figure 12: Uploading the sample code

5. When you successfully uploaded the sample code, you will now be able to see the "RAK13002 EXAMPLE" in your LCD screen, as shown in **Figure 13**, which means that the module is properly communicating with the WisBlock core using the I2C protocol.

m	$\sim$	ιz.	4	÷.	Ċ.	Ċ.	es.			
12	н	P.	Ŧ.	9	6	Ð	с,			
Ε	Х	Ĥ	Μ	Ρ	L	Е				

Figure 13: RAK13002 EXAMPLE displayed on 16x2 LCD

6. If you are not seeing the same output, check the device's I2C address by using this code:

#### **BAK**<sup>°</sup> Documentation Center

```
* Follow the connection of LCD with I2C Backpack to RAK13002.
#include <Wire.h> //include Wire.h library
void setup()
 Wire.begin(); // Wire communication begin
 Serial.begin(9600); // The baudrate of Serial monitor is set in 9600
 while (!Serial); // Waiting for Serial Monitor
 Serial.println("\nI2C Scanner");
void loop()
  byte error, address; //variable for error and I2C address
  int nDevices;
  Serial.println("Scanning...");
  nDevices = 0;
  for (address = 1; address < 127; address++ )</pre>
   Wire.beginTransmission(address);
   error = Wire.endTransmission();
   if (error == 0)
     Serial.print("I2C device found at address 0x");
     if (address < 16)
       Serial.print("0");
      Serial.print(address, HEX);
     Serial.println(" !");
     nDevices++;
   else if (error == 4)
      Serial.print("Unknown error at address 0x");
     if (address < 16)
       Serial.print("0");
     Serial.println(address, HEX);
  if (nDevices == 0)
   Serial.println("No I2C devices found\n");
   Serial.println("done\n");
 delay(5000); // wait 5 seconds for the next I2C scan
```

7. Your device's I2C address should be displayed on the Serial Monitor, as shown in Figure 14.

© COM8	- 🗆 ×
	Send
22:51:22.058 -> I2C device found at address 0x27 !	^
22:51:22.058 -> done	
22:51:22.058 ->	
22:51:27.073 -> Scanning	
22:51:27.073 $\rightarrow$ I2C device found at address 0x27 $% =$ !	
22:51:27.073 -> done	
22:51:27.073 ->	
22:51:32.088 -> Scanning	
22:51:32.088 $\rightarrow$ I2C device found at address 0x27 !	
22:51:32.088 -> done	
22:51:32.088 ->	
22:51:37.101 -> Scanning	
22:51:37.101 $\rightarrow$ I2C device found at address 0x27 $$ !	
22:51:37.101 -> done	
22:51:37.101 ->	
	~
Autoscroll Show timestamp	Both NL & CR $\ \lor$ 9600 baud $\ \lor$ Clear output

Figure 14: I2C address of your device

#### **GPIO Connection on RAK13002**

This is just an example and illustration on how to use the GPIO pins of RAK13002 for external sensors, modules, or devices. There are six (6) GPIO pins available on the RAK13002. You can use any of the GPIO pins as long as your modules, sensors, or devices operate at 3.3 V.



Figure 15: Connecting Button as your GPIO component

1. You need to select first the WisBlock Core you have, as shown in Figure 16 to Figure 18.

💿 sketch_aug22b   Arduino 1.8.15	-	đ	$\times$
File Edit Sketch Tools Help			
Auto Format Ctrl+T			<u>.</u>
Archive Sketch			
sketch_aug2: Fix Encoding & Reload			
1 void s Manage Libraries Ctrl+Shift+I			^
2 // n Serial Monitor Ctrl+Shift+M			
Serial Plotter Ctrl+Shift+L			
4 ) WiEi101 / WiEiNINA Eirmuara Undator			
4 } Withory Withory Conductor			
5 Board: "WisBlock Core RAK4631 Board"	Boards Manager		
6 void 1 Bootloader: "0.3.2 SoftDevice s140 6.1.1"	> Arduino AVR Boards >		
7 // p Debug: "Level 0 (Release)"	Arduino SAMD (32-bits ARM Cortex-M0+) Boards >		
8 Port: "COM4"	> ESP8266 Boards (3.0.2) >		
9 } Get Board Info	ESP8266 Boards (3.1.0-dev) >		
Programmer	RAKwireless ESP32 Modules		
Burn Bootloader	RAKwireless nRF Modules > • WisBlock Core RAK4631 Board		
	WisBlock Core RAK4601 Board		
			~
7	WisElock Core RAK4831 Board, 0.3.2 SoftDevice s140 6.1.1. Level 0 (R	elease) on (	COM4

Figure 16: Selecting RAK4631 as WisBlock Core

∞ sketch_aug22c	Arduino 1.8.15				-	5 X
File Edit Sketch Too	ols Help					
	Auto Format	Ctrl+T				<u>.</u> 0-
	Archive Sketch					_
sketch_aug2:	Fix Encoding & Reload					
1 void s	Manage Libraries	Ctrl+Shift+I				^
2 // n	Serial Monitor	Ctrl+Shift+M				
2 // P	Serial Plotter	Ctrl+Shift+L				
4 }	WiFi101 / WiFiNINA Firmware Updater					
5	Board: "WisCore RAK11200 Board"	>	Boards Manager			
6 void 1	Upload Speed: "921600"	>	Arduino AVR Boards			
7 // p	Flash Frequency: "80MHz"	>	Arduino SAMD (32-bits ARM Cortex-M0+) Boards			
8	Flash Mode: "QIO"	>	ESP8266 Boards (3.0.2)			
9}	Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"	>	ESP8266 Boards (3.1.0-dev)			
	Core Debug Level: "None"	>	RAKwireless ESP32 Modules	WisCore RAK11200 Board		
	Port: "COM4"	>	RAKwireless nRF Modules			
	Get Board Info			-		
	Programmer	>				
	Burn Bootloader					
						~
1			WisCore RAK11200 Board, De	fault 4MB with spiffs (1.2MB APP/1.5MB \$	SPIFFS), QIO, 80MHz, 921600, No	ine on COM4

Figure 17: Selecting RAK11200 as WisBlock Core

∞ sketch_oct29a   /	Arduino 1.8.15					_	٥	$\times$
File Edit Sketch To	ols Help							
	Auto Format Archive Sketch	Ctrl+T						<b>₽</b>
sketch_oct29	Fix Encoding & Reload							-
1 void s	Manage Libraries	Ctrl+Shift+I						^
	Serial Monitor	Ctrl+Shift+M	an ao .					
2 // p	Serial Plotter	Ctrl+Shift+L	once:					
3 4 }	WiFi101 / WiFiNINA Firmware Update	r						
5	Board: "WisBlock RAK11300"	>	Boards Manager					
6 void 1	Port: "COM10 (WisBlock RAK11300)"	>	Arduino AVR Boards	>				
7 // p	Get Board Info		Arduino SAMD (32-bits ARM Cortex-M0+) Boards	>				
8	_		ESP8266 Boards (3.0.2)	>				
9}	Programmer	>	ESP8266 Boards (3.1.0-dev)	>				
	Burn Bootloader		RAKwireless ESP32 Modules	>				
			RAKwireless nRF Modules	>				
			Rakwireless Raspberry Modules	> • W	/isBlock RAK11300			
								~
1						 WisBlock RAK	11300 on CO	DM10

Figure 18: Selecting RAK11300 as WisBlock Core

2. Copy the following sample code into your Arduino IDE:

```
const int BUTTON_PIN = WB_I01; // the number of the pushbutton pin
const int SHORT_PRESS_TIME = 500; // 500 milliseconds
int lastState = LOW; // the previous state from the input pin
int currentState; // the current reading from the input pin
unsigned long pressedTime = 0;
unsigned long releasedTime = 0;
void setup() {
 Serial.begin(9600);
 pinMode(BUTTON_PIN, INPUT_PULLUP);
void loop() {
 currentState = digitalRead(BUTTON_PIN);
 if(lastState == HIGH && currentState == LOW) // button is pressed
   pressedTime = millis();
 else if(lastState == LOW && currentState == HIGH) { // button is released
   releasedTime = millis();
   long pressDuration = releasedTime - pressedTime;
   if( pressDuration < SHORT_PRESS_TIME )</pre>
     Serial.println("A short press is detected");
 lastState = currentState;
```

3. Then select the right Serial Port and upload the code, as shown in Figure 19 and Figure 20.

💿 RAK13(	002_10	Arduino 1.8.15			- 0	×
File Edit S	ketch To	pols Help				
		Auto Format	Ctrl+T			<b>.</b>
		Archive Sketch				
RAK13	002_1	Fix Encoding & Reload				
1 /*		Manage Libraries	Ctrl+Shift+I			^
2 *	Rea	Serial Monitor	Ctrl+Shift+M	n a Button using RAK13002		
3 *		Serial Plotter	Ctrl+Shift+L	· · · · · · · · · · · · · · · · · · ·		
4 */		WiFi101 / WiFiNINA Firmware Updater				
5		Board: "WisBlock Core RAK4631 Board"	>			
6//	con	Bootloader: "0.3.2 SoftDevice s140 6.1.1	" >	here to set pin numbers:		
7 con	st	Debug: "Level 0 (Release)"	>	umber of the pushbutton pin		
8 con	st	Port: "COM8"	>	Serial ports conds		
9		Get Board Info		COM3		
10//	Var	Programmer	>	V COM8		
11 int	la	Burn Bootloader		tate from the input pin		
12 int	cur	rentState; // the c	urrent re	ading from the input pin		
13 uns	igne	d long pressedTime = 0	;			
14 uns	igne	d long releasedTime = 0	;			
15						
16						
17 voi	d se	tup() {				
18 <b>S</b>	eria	<b>1.begin</b> (9600);				
19 n	inMo		T.T.ITP) -			~
1				WisBlock Core RAK4631 Board, 0.3.2 SoftDevice s140 6.1.1, Le	rei U (Release) on	COM8

Figure 19: Selecting the correct port

© RAK13002_IO   Arduino 1.8.15		- 0	×
RAK13002_IO			
1 /*			^
2 * Reading Long Press and Short Press on a Button using RAK13002			
3 *			
4 */			
5			
6 // constants won't change. They're used here to set pin numbers:			
7 const int BUTTON PIN = WB IO1; // the number of the pushbutton pin			
8 const int SHORT PRESS TIME = 500; // 500 milliseconds			
9			
10 // Variables will change:			
11 int lastState = LOW; // the previous state from the input pin			
12 int currentState; // the current reading from the input pin			
13 unsigned long pressedTime = 0;			
14 unsigned long releasedTime = 0;			
15			
16			
17 void setup() {			
18 Serial.begin(9600);			
19 DIDMODE (RUTTON DIN INDUT PULLTIP) .			~
7	WisBlock Core RAK4631 Board, 0.3.2 SoftDevice s140 6.1.1, Leve	l 0 (Release) on	СОМ8

Figure 20: Uploading your code

4. When you successfully uploaded the sample code, open the Serial Monitor of the Arduino IDE to see the button's reading logs. Try pressing the button, and if you see the logs, as shown in **Figure 21**, then your module or sensor is properly communicating to the WisBlock core using the Digital Interface.

© COM8 —		$\times$
		Send
01:53:11.568 -> A short press is detected		
01:53:12.298 -> A short press is detected		
01:53:13.393 -> A short press is detected		
01:53:13.393 -> A short press is detected		
01:53:14.123 -> A short press is detected		
01:53:14.720 -> A short press is detected		
01:53:15.218 -> A short press is detected		
01:53:15.882 -> A short press is detected		
Autoscroll Show timestamp Both NL & CR v 115200 baud v	Clear	output

Figure 21: Serial Monitor Output

#### Analog Input (ADC) Connection on RAK13002

This is just an example and illustration on how to use the ADC pin of RAK13002 for external sensors, modules, or devices. There are two (2) ADC pins available on the RAK13002 that you can use as long as your modules, sensors, or devices operate at 3.3 V.



Figure 22: Connecting the RAK13002 to the ADC pin of the sensor module

1. You need to select first the WisBlock Core you have, as shown in Figure 23 to Figure 25.

🥯 sketch_aug22b	Arduino 1.8.15			- 0	×
File Edit Sketch To	ools Help				
	Auto Format Archive Sketch Fix Focoding & Poload	Ctrl+T			.Q. ▼
1 void s 2 // p 3	Manage Libraries Serial Monitor Serial Plotter	Ctrl+Shift+I Ctrl+Shift+M Ctrl+Shift+L	nce:		^
4)	without y with with a thin wate opdater				
	Board: "WisBlock Core RAK4631 Board"		Boards Manager		
6 VOId I	Bootloader: "0.3.2 SoftDevice s140 6.1.1		Arduino AVR Boards		
/ // p	Debug: "Level 0 (Release)"		Arduino SAMD (32-bits ARM Cortex-M0+) Boards >		
8	Port: "COM4"		ESP8266 Boards (3.0.2)		
9}	Get Board Info		ESP8266 Boards (3.1.0-dev) >		
	Programmer		RAKwireless ESP32 Modules		
	Burn Bootloader		RAKwireless nRF Modules VisBlock Core RAK4631 Board		
			WisBlock Core RAK4601 Board		
					~
7			WisRinck Core R4K4831 Roard 0.3.2 SoftDevice s140.6.1.1 Lev	al 0 (Release) or	COM

Figure 23: Selecting RAK4631 as WisBlock Core

	- 0	×
File Edit Sketch Tools Help		
Auto Format Ctrl+T		<u>.</u> 0-
Archive Sketch		
sketch_aug2: Fix Encoding & Reload		
1 word s Manage Libraries Ctrl+Shift+1		^
Serial Monitor Ctrl+Shift+M		
2 // P Serial Plotter Ctrl+Shift+L		
4 } WiFi101 / WiFiNINA Firmware Updater		
5 Boards "WisCore RAK11200 Board" Boards Manager		
6 void 1 Upload Speed: "921600"		
7 // p Flash Frequency: "80MHz" > Arduino SAMD (32-bits ARM Cortey-M0+) Boards >		
8 Flash Mode: "OIO" > ESP8266 Boards (3.0.2) >		
Partition Scheme: "Default 4MB with spiffs (1,2MB APP/1.5MB SPIFFS)"     Provide the spiffs (1,2MB APP/1.5MB SPIFFS)"     Provide		
Core Debug Level: "None" RAKwireless FSP32 Modules WisCore RAK11200 Roard		
Port: "COM4" RAKwireless nRF Modules		
Get Board Info		
Programmer		
r roganine a secondar s		
1 WisCore RAK11200 Board, Default-4MB with spiffs (1.2MB APP/1.5MB SPIFFS), OLO, 800	IHz, 921600, None on C	OM4

Figure 24: Selecting RAK11200 as WisBlock Core

💿 sketch_oct29a   Ard	luino 1.8.15			_	٥	$\times$
File Edit Sketch Tools	Help					
sketch_oct29         F           1 void s         2           2         // p	Auto Format Archive Sketch Fix Encoding & Reload Manage Libraries Serial Monitor Serial Plotter	Ctrl+T Ctrl+Shift+I Ctrl+Shift+M Ctrl+Shift+L	once:			
4 } V	WiFi101 / WiFiNINA Firmware Updater					
5 void 1 7 // p 8 9 }	Board: "WisBlock RAK11300" Port: "COM10 (WisBlock RAK11300)" Get Board Info Programmer Burn Bootloader	>	Boards Manager         Arduino AVR Boards         Arduino SAMD (32-bits ARM Cortex-M0+) Boards         ESP8266 Boards (3.0.2)         ESP8266 Boards (3.1.0-dev)         RAKwireless ESP32 Modules         RAKwireless RFR Modules         Rakwireless Raspberry Modules             WisBlock RAK11300			
						Ŷ
1				WisBlock R/	AK11300 on C	OM10

Figure 25: Selecting RAK11300 as WisBlock Core

2. Copy the following sample code into your Arduino IDE:

/* * Reading ADC pin on RAK13002 * using Soil Moisture Sensor * */	
#define SS WB_A0 //Soil Moisture Sensor A0 to AIN0 of RAK13002	
int sensor_value;	
<pre>void setup() {    Serial.begin(9600); // Setting up Serial Monitor to read in 9600 baudrate</pre>	
<pre>} void loop() {   readSensor();   delay(1000); //Read sensor value and print every 1 second. }</pre>	
<pre>void readSensor(){    sensor_value = analogRead(SS);    Serial.println(sensor_value); }</pre>	

3. Then select the right Serial Port and upload the code, as shown in Figure 26 and Figure 27.

# **BAK**<sup>®</sup> Documentation Center

💿 RAK13002_ADC	Arduino 1.8.15			- 0	×
File Edit Sketch To	ools Help				
	Auto Format	Ctrl+T			<b>.</b>
	Archive Sketch				
RAK13002_A	Fix Encoding & Reload				
1 /*	Manage Libraries	Ctrl+Shift+I			^
2 * Read	Serial Monitor	Ctrl+Shift+M			
3 * usin	Serial Plotter	Ctrl+Shift+L			
4 *	WiFi101 / WiFiNINA Firmware Updater				
5 */	Board: "WisBlock Core BAK4631 Board"	>			
6	Rootloader: "0.2.2 SoftDavice s140.6.1.1	/  "			
7#defin	Debug: "Level 0 (Release)"	· ^ ^	A0 to AINO of RAK13002		
8	Port: "COM8"	>	Serial ports		
9 int se	Get Board Info		COM3		
10	D		✓ COM8		
11 woid s	Programmer	,			
12 Soria	bogin (9600) · // Sotti	ng un Cor	al Manitar to road in 0600 baudrato		
12 Seria	L.Degin(9000); // Setti	ing up ser	at Monitor to read in 9000 baudrate		
13					
14 }					
15 void lo	op() {				
16 readS	ensor();				
17 delay	(1000); //Read sensor v	ralue and	print every 1 second.		
18 }					
19					~
1			WisBlock Core RAK4631 Board, 0.3.2 SoftDevice s140 6.1.1, L	evel 0 (Release)	on COM8



<pre>RAK13002_ADC RAK13002_ADC 1 /* 2 * Reading ADC pin on RAK13002 3 * using Soil Moisture Sensor 4 * 5 */ 6 f #define SS WB_A0 //Soil Moisture Sensor A0 to AINO of RAK13002 8 int sensor_value; 10 11 void setup() { 12 Serial horiz (6600); // Satting up Serial Monitor to read in 9600 houdrate </pre>
<pre>RAK13002_ADC  1 /*  2 * Reading ADC pin on RAK13002 3 * using Soil Moisture Sensor 4 * 5 */ 6 7 #define SS WB_A0 //Soil Moisture Sensor A0 to AIN0 of RAK13002 8 9 int sensor_value; 10 11 void setup() { 12 Serial horiz (6600); // Satting up Serial Monitor to read in 9600 houdrate </pre>
<pre>1 //* 2 * Reading ADC pin on RAK13002 3 * using Soil Moisture Sensor 4 * 5 */ 6 7 #define SS WB_A0 //Soil Moisture Sensor A0 to AINO of RAK13002 8 9 int sensor_value; 10 11 void setup() { 12 Serial horiz (6600); // Sotting up Serial Monitor to read in 9600 houdrate </pre>
13
15 void loop() {
<pre>16 readSensor(); 17 delay(1000); //Read sensor value and print every 1 second. 18 } 10</pre>
1 WisBlock Core R4K4631 Board, 0.3.2 SoftDevice s140 6.1.1, Level 0 (Release) on COM8

Figure 27: Uploading the sample code

4. When you successfully uploaded the sample code, open the Serial Monitor of the Arduino IDE to see the module's reading logs. If you see the logs, as shown in **Figure 28**, then your module or sensor is properly communicating to the WisBlock core using the Analog Interface.

© COM8			_		×
					Send
02:37:30.362 -> 933					^
02:37:31.359 -> 941					
02:37:32.369 -> 941					
02:37:33.364 -> 944					
02:37:34.358 -> 940					
02:37:35.354 -> 942					
02:37:36.371 -> 941					
02:37:37.366 -> 944					
02:37:38.357 -> 942					
02:37:39.347 -> 940					
02:37:40.358 -> 939					
02:37:41.353 -> 939					
02:37:42.354 -> 939					
02:37:43.353 -> 942					
02:37:44.354 -> 941					
					~
Autoscroll Show timestamp Both NL	& CR 🕓	9600 baud	$\sim$	Clear	output

Figure 28: FC-28 Soil Moisture Hygrometer data logs

Last Updated: 7/29/2022, 10:17:19 PM

# **RAK13002 WisBlock Adaptor Module Datasheet**

### **Overview**

# Description

The RAK13002 is a WisBlock Core adaptor module that can be mounted to the IO slot of the WisBlock Base board. This module exposed all WisBlock Core signals such as I2C, SPI, UART, GPIO, and ADC to standard 2.54 mm pitch pin header for easy integration of external components and devices.

# Features

- Supports two I2C interfaces
- Supports two UART interfaces
- Supports one SPI interface
- Supports up to six (6) GPIOs
- Supports two (2) ADC interfaces
- 3.3 V power supply interfaces
- Backup battery (super cap) can keep the RTC running for up to 7 days (tested in lab)
- Module size: 25X35 mm

# Specifications

# Overview

# Mounting

The RAK13002 module can be mounted to the IO slot of the WisBlock Base board. Figure 1 shows the mounting mechanism of the RAK13002 on a WisBlock Base module.



Figure 1: RAK13002 WisBlock Adaptor Module Mounting

# Hardware

The hardware specification is categorized into four parts. It discusses the pinouts of the module and its corresponding functions and diagrams. It also covers the electrical and mechanical parameters that include the tabular data of the functionalities and standard values of the RAK13002 WisBlock Adaptor Module.

#### **Pin Definition**

The RAK13002 WisBlock Adaptor Module comprises a standard WisConnector connector. The WisConnector allows the RAK13002 module to be mounted to a WisBlock Base board. The pin order of the connector and the pinout definition is shown in Figure 2.

# **BAK**<sup>®</sup> Documentation Center



Figure 2: RAK13002 WisBlock Adaptor Module Pinout

# Electrical Characteristics

#### **Recommended Operating Conditions**

Symbol	Description	Min.	Nom.	Max.	Unit
VCC	Power supply		3.3		V

# **Mechanical Characteristics**

#### **Board Dimensions**

Figure 3 shows the dimensions and the mechanic drawing of the RAK13002 module.



Figure 3: RAK13002 WisBlock Adaptor Module Mechanic Drawing

#### WisConnector PCB Layout



Figure 4: WisConnector PCB Footprint and Recommendations

# Schematic Diagram

#### Adaptor

Figure 5 shows the RAK13002 adaptor module schematic. VCC: 3.3 V power supply



Figure 5: RAK13002 WisBlock Adaptor Schematic

Last Updated: 6/14/2022, 8:07:39 AM