J4210 Arduino Interfacing

Version: 1.1 Date: 2025-06-11

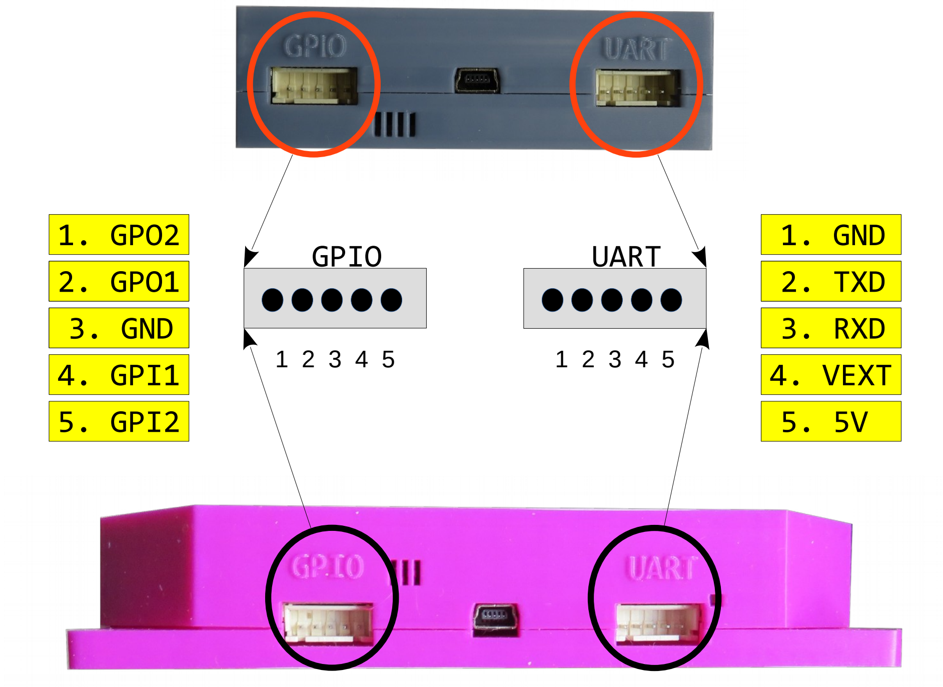
Contents

Getting Started

This documentation is your comprehensive guide to understand and utilize the full potential of **Jence Uhf RFID Reader**. Whether you are a developer, system integrator, or end-user, we have tailored this guide to assist you in setting up, configuring, and integrating our UHF RFID reader into your specific applications.

Hardware Description:

Aside from USB interface our reader provides UART communication interface via Rx and Tx pin. The configuration picture is given below



UART:
VEXT – This is the Logic High, V_{OH} voltage. In 3.3V logic, tie it to 3.3V, in 5V logic, tie it to 5V.
5V – This is supply voltage. Do not supply 3.3V here.

GPIO:
GPO1 and **GPO2** – Active Low Open Drain. Must provide at pullup resistor. Maximum continuous load current 35mA.
GPI1 and **GPI2** – High Impedance input.

UART – RXD and **TXD** will be crossed with the MCU’s **TXD** and **RXD**.

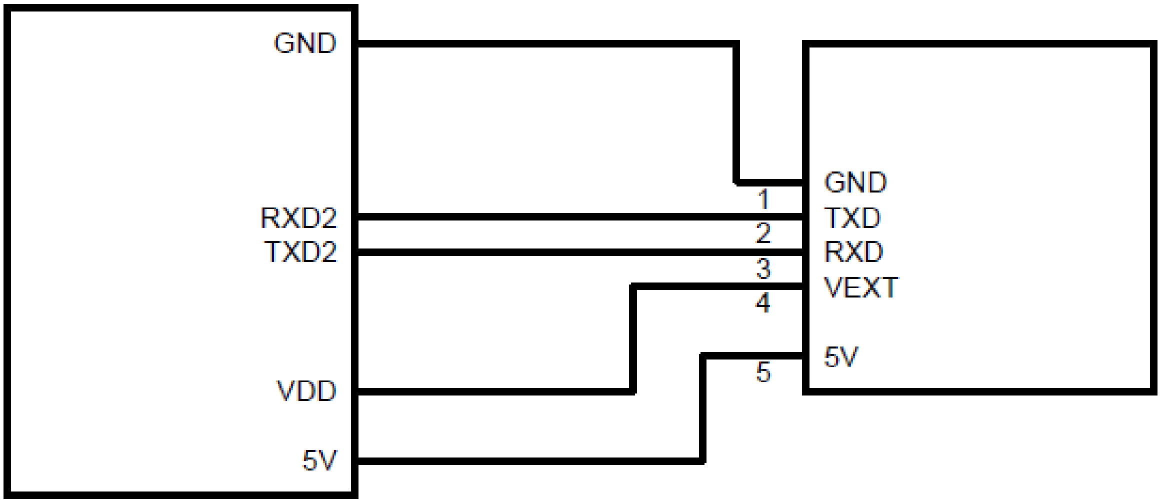
VEXT – This is the Logic High, VOH voltage. In 3.3V logic, tie it to 3.3V, in 5V logic, tie it to 5V.

5V – This is supply voltage. Do not supply 3.3V here.

The basic idea is to connect the uhf reader module with an MCU like below configuration

MCU

UHF module



Hardware Setup:

Connection basic

The key is MCU needs 2 sets of UART port. One is to connect with the PC through USB-Serial device to monitor the status of the MCU through Serial Monitor. Another set of UART is to communicate with the module to send command and retrieve information. In the example, we are using UART2 for the Module – MCU communication but it is configurable as the user wants. Only 1 set of UART pins can do the communication if Serial Monitor is not needed. Our example uses 2 sets of UART port.

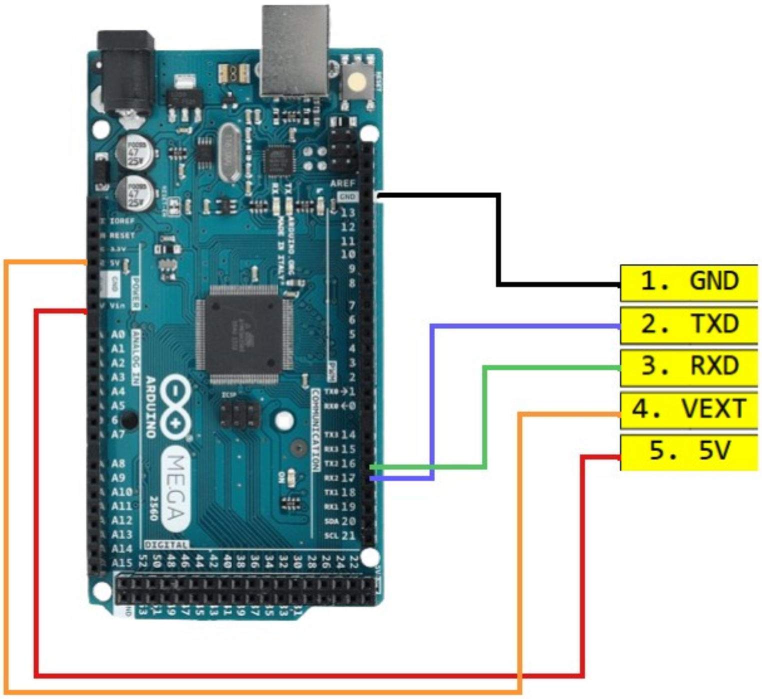
Power Requirements

UHF RFID Module requires 400mA, 5V power while scanning. If the MCU is used as power source for the module it will take around 400mA of current while scanning. While scanning if the MCU can’t provide that much current then scan will fail. So, if the MCU can’t provide 400mA 5V then external power source is needed.

Connection Schemes for different Microcontrollers:

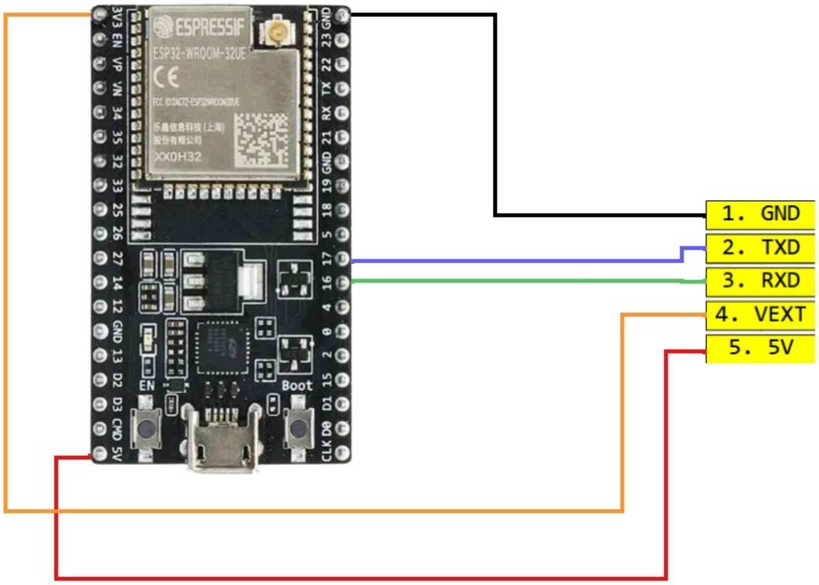
Connection configuration example for some common microcontroller that can be used with Arduino platform is provided below. You should have one extra UART other than the default UART, which is used for PC console and/or debug. Note that pin configuration is just an example and an expert user can configure their pins however they want reflecting their code.

For Arduino Mega:

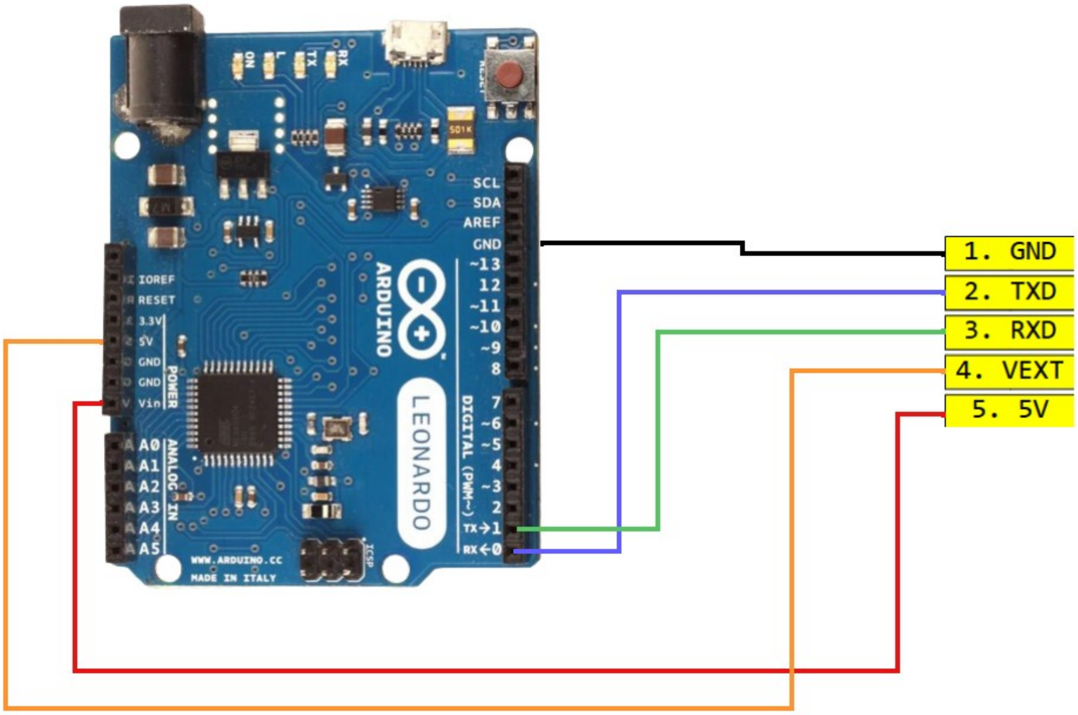


f

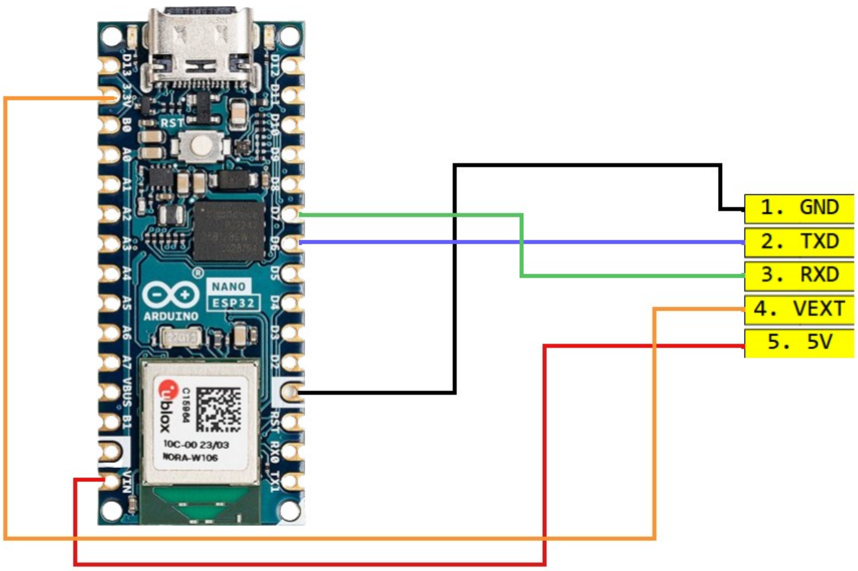
FOR ESP32 dev module (wroom):



For Arduino Leonardo



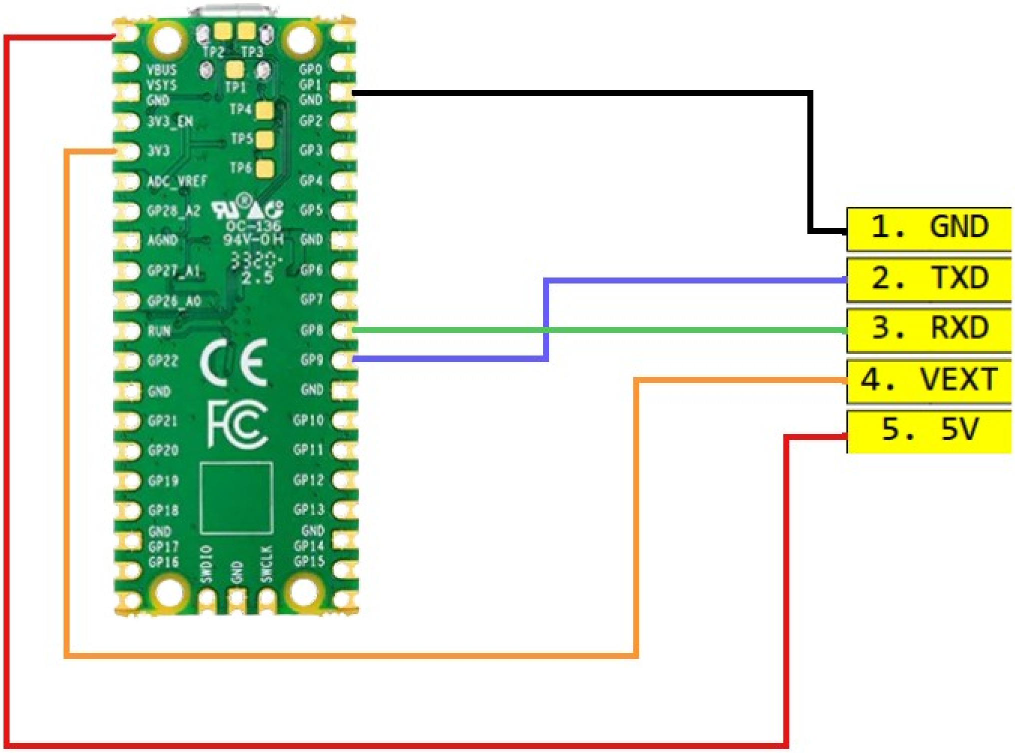
For Arduino Nano ESP32



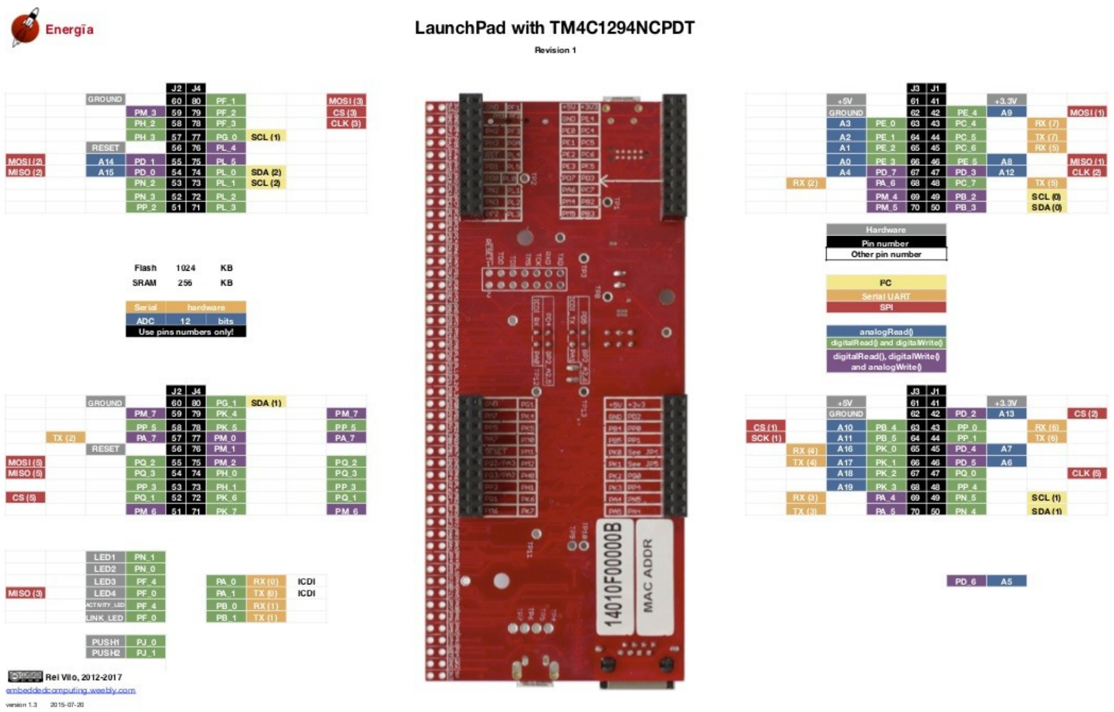
For RPI Pico

Add the RP2040 support in the Arduino IDE.

Link:https://www.upesy.com/blogs/tutorials/install-raspberry-pi-pico-on-arduino-ide-software



For TivaC Launchpad with TM4C1294NCPDT



LaunchPad with TM4C1294NCPDT

Revision 1

				J2	J4				
		GROUND		60	80	PF_1			MOSI (3)
			PM_3	59	79	PF_2			CS (3)
			PH_2	58	78	PF_3			CLK (3)
			PH_3	57	77	PG_0	SCL (1)		
		RESET		56	76	PL_4			
MOSI (2)		A14	PD_1	55	75	PL_5			
MISO (2)		A15	PD_0	54	74	PL_0	SDA (2)		
			PN_2	53	73	PL_1	SCL (2)		
			PN_3	52	72	PL_2			
			PP_2	51	71	PL_3			

Flash	1024	KB
SRAM	256	KB
Serial	hardware	
ADC	12	bits
Use pins numbers only!		

				J3	J1				
		+5V		61	41		+3.3V		
		GROUND		62	42	PE_4	A9		MOSI (1)
		A3	PE_0	63	43	PC_4		RX (7)	
		A2	PE_1	64	44	PC_5		TX (7)	
		A1	PE_2	65	45	PC_6		RX (5)	
		A0	PE_3	66	46	PE_5	A8		MISO (1)
		A4	PD_7	67	47	PD_3	A12		CLK (2)
	RX (2)		PA_6	68	48	PC_7		TX (5)	
			PM_4	69	49	PB_2		SCL (0)	
			PM_5	70	50	PB_3		SDA (0)	

Hardware
Pin number
Other pin number

I²C
Serial UART
SPI

analogRead()
digitalRead() and digitalWrite()
digitalRead(), digitalWrite() and analogWrite()

				J2	J4				
		GROUND		60	80	PG_1	SDA (1)		
			PM_7	59	79	PK_4			PM_7
			PP_5	58	78	PK_5			PP_5
	TX (2)		PA_7	57	77	PM_0			PA_7
		RESET		56	76	PM_1			
MOSI (5)			PQ_2	55	75	PM_2			PQ_2
MISO (5)			PQ_3	54	74	PH_0			PQ_3
			PP_3	53	73	PH_1			PP_3
CS (5)			PQ_1	52	72	PK_6			PQ_1
			PM_6	51	71	PK_7			PM_6

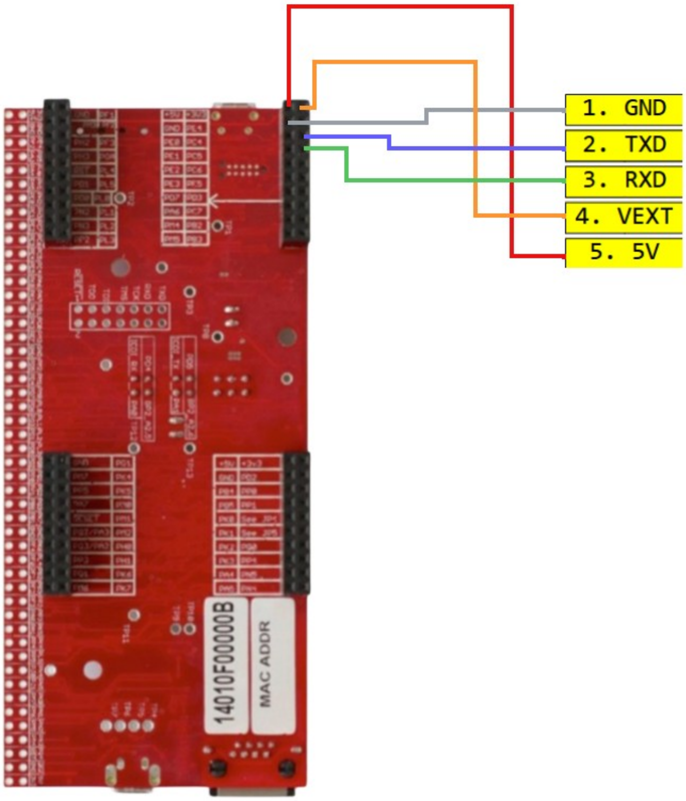
				J3	J1				
		+5V		61	41		+3.3V		
		GROUND		62	42	PD_2	A13		CS (2)
CS (1)		A10	PB_4	63	43	PP_0		RX (6)	
SCK (1)		A11	PB_5	64	44	PP_1		TX (6)	
	RX (4)	A16	PK_0	65	45	PD_4	A7		
	TX (4)	A17	PK_1	66	46	PD_5	A6		
		A18	PK_2	67	47	PQ_0			CLK (5)
		A19	PK_3	68	48	PP_4			
	RX (3)		PA_4	69	49	PN_5		SCL (1)	
	TX (3)		PA_5	70	50	PN_4		SDA (1)	

PD_6	A5

		LED1	PN_1
		LED2	PN_0
		LED3	PF_4
MISO (3)		LED4	PF_0
		ACTIVITY_LED	PF_4
		LINK_LED	PF_0
		PUSH1	PJ_0
		PUSH2	PJ_1

PA_0	RX (0)	ICDI
PA_1	TX (0)	ICDI
PB_0	RX (1)	
PB_1	TX (1)	

Rei Vilo, 2012-2017
embeddedcomputing.weebly.com
version 1.3 2015-07-20



Software Installation:

UHF RFID readers has an Arduino framework library class named J4210U. In the J4210U.h file the public methods are to be used to communicate with the RFID reader. For the library class, an example sketch is also provided that does some operation to demonstrate some of the functionality of the library.

Software setup:

1. Download, Install and Setup Arduino IDE normally.
2. Open Arduino IDE.
3. In the menu bar, go to Sketch > Include Library > Add .ZIP Library...
4. Select the bundled j4210_driver_arduino_lib.zip file in the file explorer prompt.
5. The library should install in Output Terminal at the bottom
6. Restart the Arduino IDE
7. In the menu bar, go to File > Examples > j4210u Uhf RFID Reader Driver > choose one

8. You can find 3 examples from there choose one of 3 to run and check the example usage.

9. Uncomment // #define NANO_ESP32 line from the example sketches if Arduino Nano ESP32 is begin used.

10. Connect the wires accordingly based on the MCU

11. Configure Arduino IDE (Or Code Composer Studio for TivaC board) for right MCU board configuration from TOOLS tab

12. Build and upload the sketch.

13. Check in serial monitor for messages. The default Serial monitor baud of the examples are 9600

14. A successful integration will show example specific output in the Serial monitor. For continuous scan example the output will be the reader settings in the serial monitor and the reader will keep try to scan nearby cards after 3s. keep some uhf tag nearby and EPC and TID will be visible after each scan in the serial monitor.

15. After testing the examples, you can start using the library for Standard application such as GetSettings, SetSettings, Inventory Scan, Read, Write, Filter GPIO, SetPassword, Lock, Kill, operation etc. by using the public methods of J4210U class.

Common Troubleshooting:

1. Check the device if it behaves intendedly with platform specific desktop software before using it with MCU.
2. While connecting the wires sometimes Rx and Tx might get mixed up by mistake and that can cause failed communication. Check if the **Rx** and **Tx** is connected properly and try swapping the Rx and Tx.
3. VEXT is the logic voltage. If VEXT doesn’t get matched with logic HIGH level then the communication won’t work. Check the voltage level of VEXT
4. Module is connected but scan can’t find any tags. It might be due to the module is not getting enough power to scan. Connect an external power source with 5V and GND pin.
5. Check the baud Rate from desktop software to be 100% sure. Default baud rate and Arduino example has also set to 57600bps. If that baud rate fails then try changing it to 115200bps.