JANUS REMOTE COMMUNICATIONS

Application Note 115

Author: Clayton Knight,
Senior Design Engineer
Janus Remote Communications

Janus Arduino Shield User Manual-Instructions

Overview

The Janus Plug in Modules are versatile, pre-certified devices that allow a user's application be cellularly connected without worrying about many of the complications that accompany cellular integration. The Arduino Platform allows users to be up and running with circuitry quickly, for evaluation and potential testing for integration to a more defined system later on.

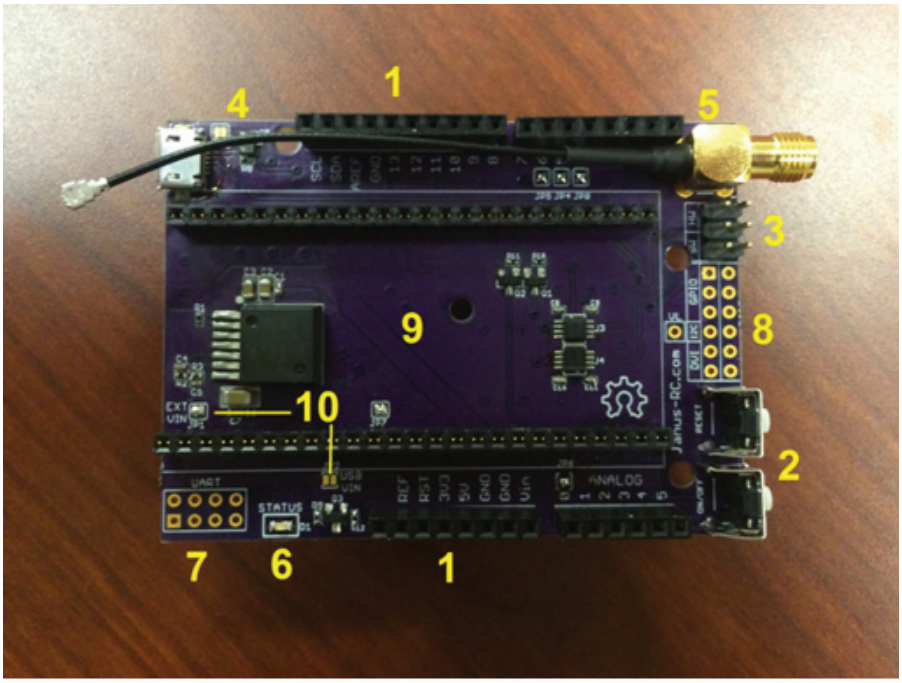
Janus has taken the liberty of merging these two areas with the Arduino Shield for the Plug in Modules, allowing a user to evaluate these devices, and bring up a proof of concept in record time. We are providing customers with both open source hardware designs and software libraries, leaving nothing out for the user to guess. Please be aware that Janus offers these software libraries as is with no warranty or guarantee, and are for evaluation purposes only.

All schematics, layout, and software can be downloaded/forked here: github.com/JanusRC/Arduino

Contents:

Bulletin	JA01-AN115
Revision	00
Date	21 Oct 2015

Connections



1. Arduino compatible stacking headers with top socket for I/O usage.
2. ON/OFF and Reset external buttons for some manual control
3. Hardware and Software translated TX/RX UART selection
4. USB Port directly from the Telit modem
5. SMA to micro coax antenna connection
6. Green cellular status LED, connected to the Stat LED of the Telit Modem.
7. Untranslated modem UART breakout
8. Modem GPIO breakout, see the user manual for more information
9. Terminus Plug in Modem DIP socket, accepts all Plug in Modem designs
10. Power selection jumpers. Defaulted for Barrel Jack/VIN usage.

Jumpers

The shield has several jumpers that are connected by default for the user to adjust or remove connections without needing to cut traces. Red color denotes a defauled normally closed connection.

- JP1 – External VIN : Routes power from VIN through the on board 5V buck regulator
- JP2 – USB VIN: Routes power from the 5V signal/USB, bypassing the regulator
- JP3 – VBUS control: Ensures that VBUS is always ON, bypassing control
- JP4 – Modem Enable: Routes D5 to the Terminus Enable pin
- JP5 – Modem On Off: Routes D6 to the Terminus On/Off pin
- JP6 – Modem PWRMON: Routes A0 to the Terminus Powermon feedback pin
- JP7 – Modem RTS: The RTS UART line jumper to ground, reqired when not using full UART.
- JP8 – Modem VBUS: Routes D4 to the Terminus VBUS control circuitry

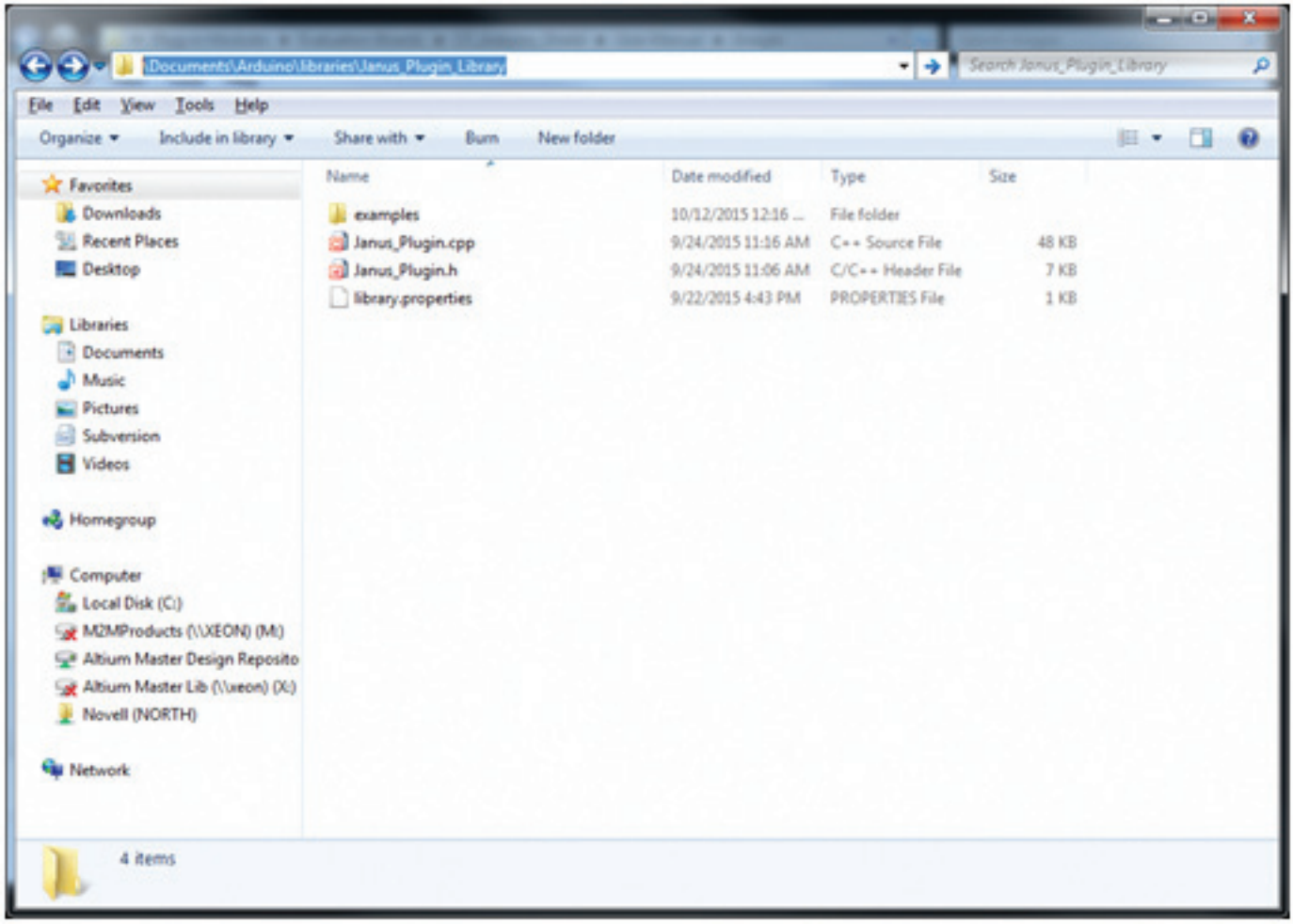
Getting Started

Before you begin

You must have the following for this section

- Janus Plug in Module
- Janus Plugin Module Arduino Shield
- Arduino Uno or Leonardo (this section covers Uno usage)
- Downloaded/cloned the Arduino software repository
- Installed the Arduino IDE and Arduino Uno/Leonardo USB drivers
- Have 1x USB B to A cable
- Have 1x 12 VDC Power Supply

1. The first thing to do is to move the Janus Plugin Library and examples to the proper directory, which is usually "..\Arduino\libraries" for standard Arduino IDE installs.



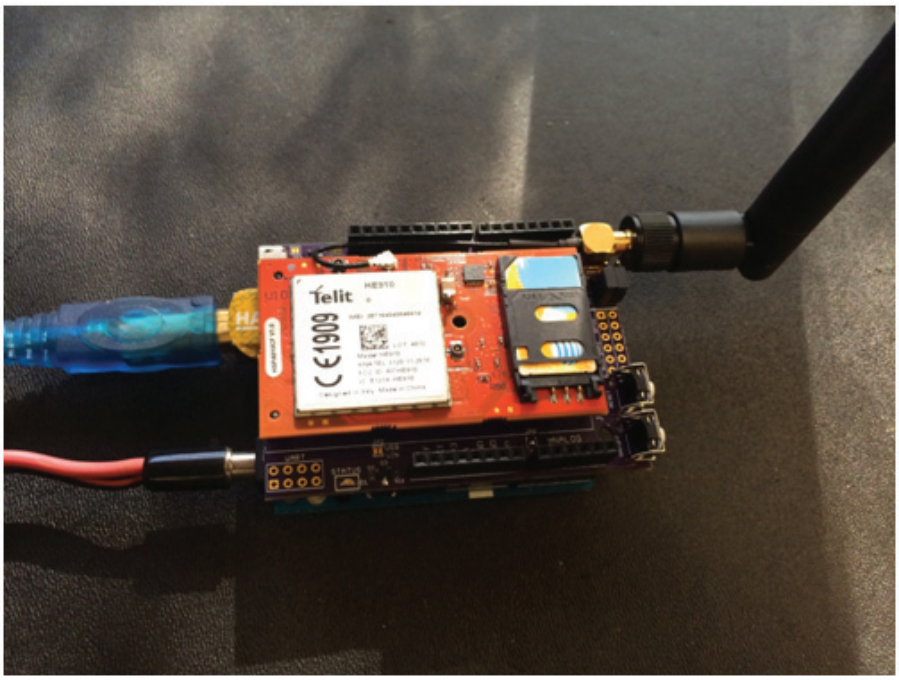
Getting Started continued

2. On your Arduino Uno, the USB B jack will short out two pins on the Plug in Module if not taken into consideration. It's recommended to use one of the included stickers that come with your new Arduino Uno to simply cover the jack. You may also want to utilize the spacers that come with the Arduino Shield Kit from janus to keep the board spacing easy

3. It's easiest to connect the micro coax to the Cellular micro coax connection on your Plug in Module before putting it into the Shield's sockets. Once connected, align your Plug in Module and insert it into the sockets.
4. Insert the Shield into the base Arduino Uno board, taking care to ensure all the long pins are in place.

Getting Started continued

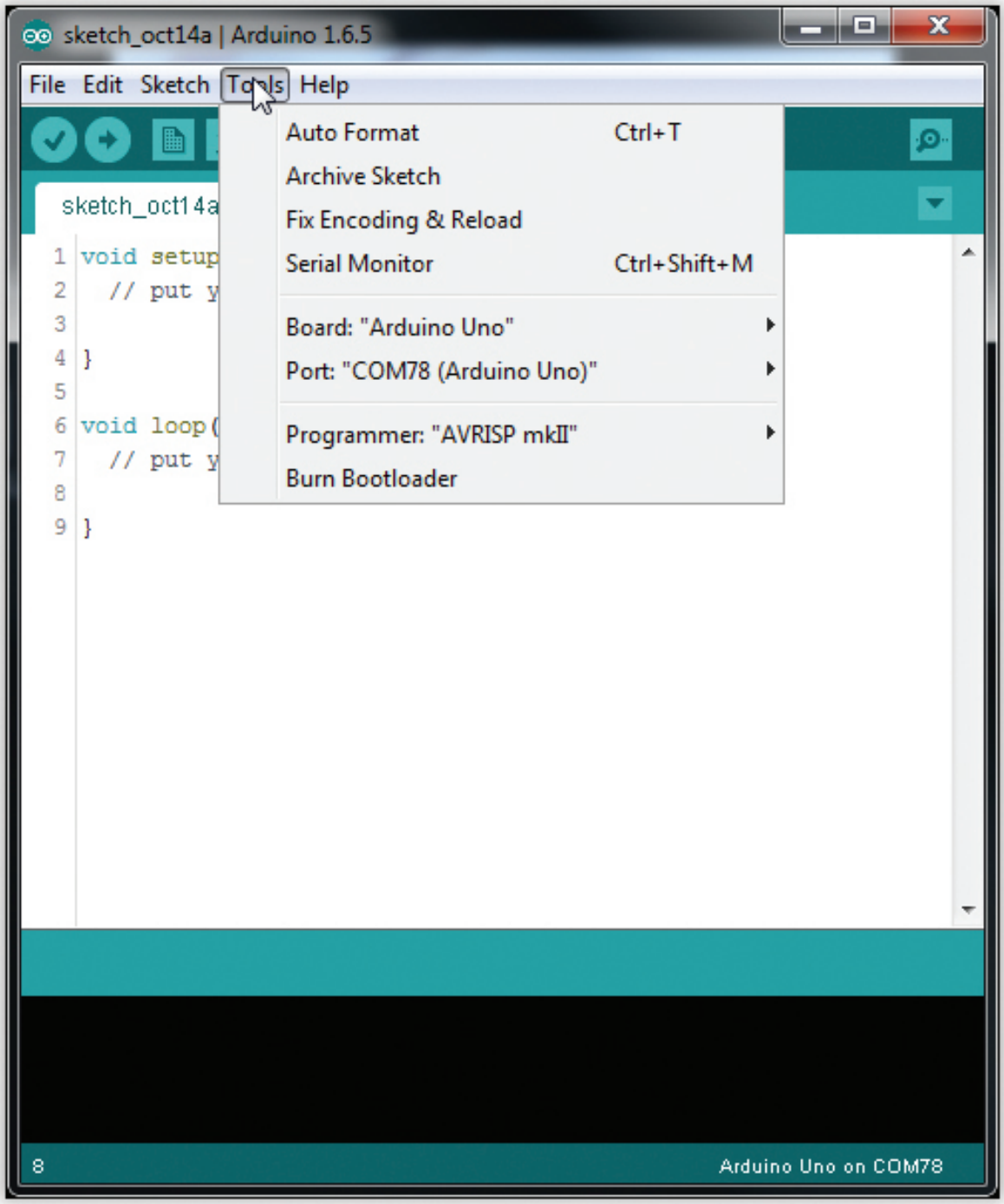
5. If not done already, connect your antenna to the SMA terminal, and place your UART jumpers on the SW end of the 2x6 header, next to the antenna connection.
6. Connect your power supply to the barrel jack, and then connect your USB cable to your host PC that will run the Arduino IDE.



7. Start the Arduino IDE, it's important to have the Plugin library in place before booting the IDE as it autodetects everything, and must be restarted if you adjust the placement of the directories.

Getting Started continued

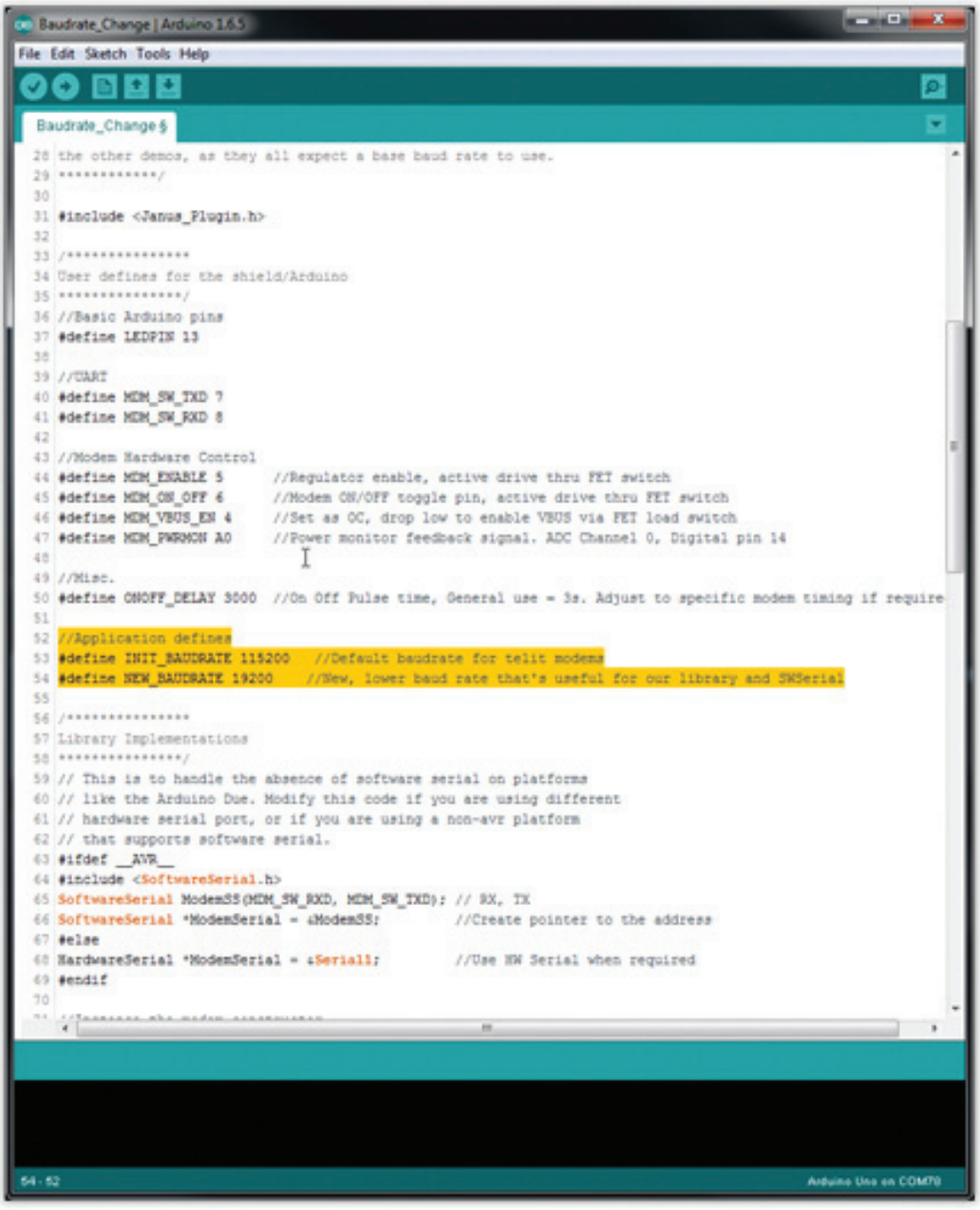
8. Once the IDE is running, ensure to set the board to the Arduino Uno, and the Port to the COM Port that the Arduino Uno installs under.



Getting Started continued

9. Go to File->Examples->Janus Plugin Modem Library->Baudrate_Change. This will bring up the baud rate change sketch that will allow you to adjust the baud rate of the modem to a slower one that the library/sketches will be happy with. They all utilize software serial by default, which requires a slower baud rate for proper functionality, Janus has chosen 19200 as this default baud rate.

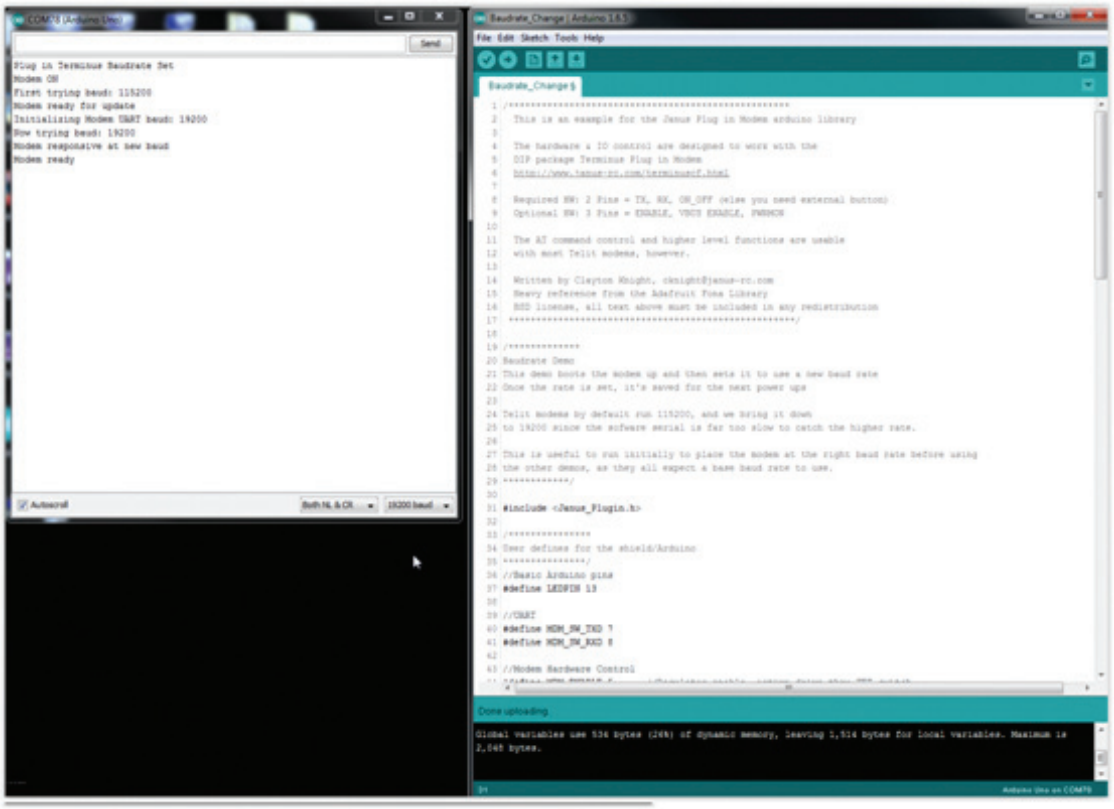
   Although defaulted for proper changeout of a Telit modem, check that INIT_BAUDRATE is 115200, and NW_BAUDRATE is 19200.



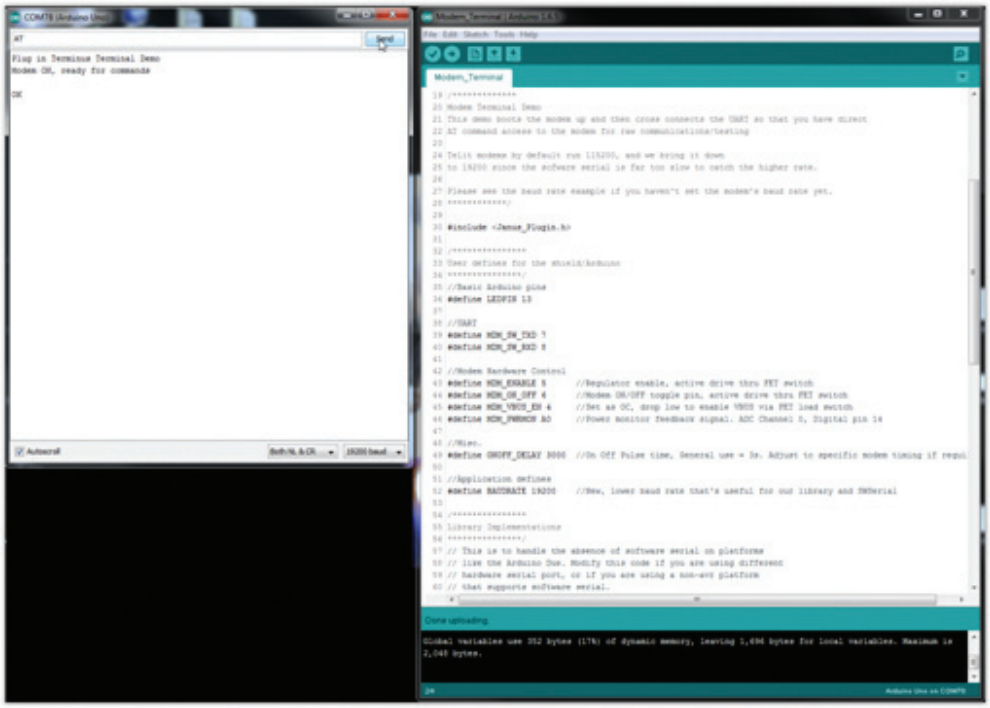
10. Go to Tools->Serial Monitor and open up the serial port monitor, change it to 19200 baud. Once open, upload the sketch through the menu or via the -> arrow indicator.

Getting Started continued

11. The sketch will start immediately and will run through the process, shown below.



12. Once complete, the modem is ready to use at 19200. So we move on and select the Modem_Terminal sketch. Just as before, ensure the BAUDRATE define is 19200, and then upload the sketch. This will cross connect the Atmega328's UARTs and allow you direct terminal communications with the modem.



Getting Started continued

13. From here you play with the modem directly via the AT command interface, or explore the Janus created sketches and library.

Take note of the following:

1. This library is currently optimized for the HE910/HSPA910CF, if using the DE910/EVDO910CF or other modem types, be prepared to adjust the more advanced code that checks for a SIM card or checks for data registration.
2. Although available, it's recommended to do all evaluation with the barrel jack as your power source. You can evaluate using USB as your power source, but be aware that high energy transmissions and modem operations can draw more power than USB is capable of by default. If you have good reception and only want basic things like SMS, this may work for you just fine.
3. In the Janus_Plugin.h file, you can enable debugging by uncommenting the following as necessary:
    1. #define DEBUG //Higher level UART show, both TX and RX
       #define DEBUG_LOWLVL //Low level, show individual received/parsed lines in brackets
4. In the Janus_Plugin.h file, you MUST uncomment USE_SMS or USE_SOCKET to use these functions. They are commented out to save code space when building the sketches that don't need them.

14. You may also use the Arduino Uno as a virtual COM port directly via the hardware serial port for direct terminal communications if necessary (no sketch cross connect in the middle). Simply cross connect the HW section of the 2x3 UART selection header with the center pins, and ensure that the Arduino Uno has a blank sketch on it that does NOT import or use the Serial library (Blinky is a fine built in example for this).