. The breakout board also includes a circuit that shifts the I²C clock and data lines to the same logic voltage level as the supplied VIN, making it simple to interface the board with 5 V systems, and the board’s 0.1″ pin spacing makes it easy to use with standard solderless breadboards and 0.1″ perfboards.  
Specifications  
  
    Dimensions: 0.8″ × 0.5″ × 0.1″ (20 × 13 × 3 mm)  
    Weight without header pins: 0.7 g (0.02 oz)  
    Operating voltage: 2.5 to 5.5 V  
    Supply current: 10 mA  
    Output format (I²C):  
        Gyro: one 16-bit reading per axis  
        Accelerometer: one 12-bit reading (left-justified) per axis  
        Magnetometer: one 12-bit reading (right-justified) per axis  
    Sensitivity range (configurable):  
        Gyro: ±250, ±500, or ±2000°/s  
        Accelerometer: ±2, ±4, ±8, or ±16 g  
        Magnetometer: ±1.3, ±1.9, ±2.5, ±4.0, ±4.7, ±5.6, or ±8.1 gauss  
  
Included Components  
  
A 5×1 strip of 0.1″ header pins and a 5×1 strip of 0.1″ right-angle header pins are included, as shown in the picture below. You can solder the header strip of your choice to the board for use with custom cables or solderless breadboards, or you can solder wires directly to the board itself for more compact installations.  
  
Using the MinIMU-9 v2  
Connections  
  
A minimum of four connections are necessary to use the MinIMU-9 v2: VIN, GND, SCL, and SDA. VIN should be connected to a 2.5-5.5 V source, GND to 0 volts, and SCL and SDA should be connected to an I²C bus operating at the same logic level as VIN. (Alternatively, if you are using the board with a 3.3 V system, you can leave VIN disconnected and bypass the built-in regulator by connecting 3.3 V directly to VDD.)  
Pololu MinIMU-9 v2 gyro, accelerometer, and compass pinout.  
  
Pololu MinIMU-9 v2 gyro, accelerometer, and compass in a breadboard.  
Pinout  
PIN     Description  
SCL     Level-shifted I²C clock line: HIGH is VIN, LOW is 0 V  
SDA     Level-shifted I²C data line: HIGH is VIN, LOW is 0 V  
GND     The ground (0 V) connection for your power supply. Your I²C control source must also share a common ground with this board.  
VIN     This is the main 2.5 – 5.5 V power supply connection. The SCL and SDA level shifters pull the I²C bus high bits up to this level.  
VDD     3.3 V regulator output or low-voltage logic power supply, depending on VIN. When VIN is supplied and greater than 3.3 V, VDD is a regulated 3.3 V output that can supply up to approximately 150 mA to external components. Alternatively, when interfacing with a 2.5 – 3.3 V system, VIN can be left disconnected and power can be supplied directly to VDD. Never supply voltage to VDD when VIN is connected, and never supply more than 3.6 V to VDD.  
  
The data ready and interrupt pins of the L3GD20 and the LSM303DLHC are not accessible on the MinIMU-9 v2; if you need these outputs, consider using our L3GD20 carrier and LSM303DLHC carrier boards.  
Schematic Diagram  
  
The above schematic shows the additional components the carrier board incorporates to make the L3GD20 and LSM303DLHC easier to use, including the voltage regulator that allows the board to be powered from a single 2.5-5.5 V supply and the level-shifter circuit that allows for I²C communication at the same logic voltage level as VIN. This schematic is also available as a downloadable pdf: MinIMU-9 v2 schematic (149k pdf).  
I²C Communication  
  
The L3GD20 and LSM303DLHC readings can be queried and the devices can be configured through the I²C bus. The three sensors (the L3GD20 gyro and the LSM303DLHC accelerometer and magnetometer) act as slave devices on the same I²C bus (i.e. their clock and data lines are tied together to ease communication). Additionally, level shifters on the I²C clock (SCL) and data lines (SDA) enable I²C communication with microcontrollers operating at the same voltage as VIN (2.5 – 5.5V). A detailed explanation of the protocols used by each device can be found in the L3GD20 datasheet (2MB pdf) and the LSM303DLHC datasheet (629k pdf), and more detailed information about I²C in general can be found in NXP’s I²C-bus specification (371k pdf).  
  
The gyro, accelerometer, and magnetometer each have separate slave addresses on the I²C bus. The board pulls the gyro’s SA0 pin high, setting its slave address to 1101011b. The accelerometer’s slave address is fixed to 0011001b and the magnetometer’s slave address is fixed to 0011110b.  
  
In our tests of the MinIMU-9 v2, we were able to communicate with both chips at clock frequencies up to 400 kHz; higher frequencies might work but were not tested. The chips themselves and carrier board do not meet of some requirements to make the devices compliant with I²C fast mode. They are missing 50 ns spike suppression on the clock and data lines, and additional pull-ups on the clock and data lines might also be necessary to achieve compliant signal timing characteristics.  
Sample Code  
  
We have written a basic L3GD20 Arduino library and LSM303 Arduino library that make it easy to interface the MinIMU-9 with an Arduino. The libraries make it simple to configure the sensors and read their raw gyro, accelerometer, and magnetometer data.  
  
For a demonstration of what you can do with this data, you can turn an Arduino connected to a MinIMU-9 into an attitude and heading reference system, or AHRS, with this Arduino program. It uses the data from the MinIMU-9 to calculate estimated roll, pitch, and yaw angles, and you can visualize the output of the AHRS with a 3D test program on your PC (as shown in a screenshot above). This software is based on the work of Jordi Munoz, William Premerlani, Jose Julio, and Doug Weibel.  
  
  
  
----------------------------------------  
  
  
Changing the OS  
  
The last thing you might be interested in doing is reloading or changing the operating system on the device. There are two methods for doing this: LiveSuit (sometimes found on the internet as “LiveSuite”) and PhoenixCard. LiveSuit can be used to load either Lubuntu or Android; PhoenixCard can only be used to load Linux. We’ll only be covering LiveSuit, as it is the easier and preferred means for doing this.  
  
LiveSuit is available in a Windows (32-bit) version, a Windows 7 (64-bit) version, or in a Linux (64-bit-only) version. We’ll skip over installation of the tool; the linked website explains how to install the Linux version, and the Windows versions don’t need to be installed, merely extracted.  
  
You’ll also need the appropriate image file. Here’s the Lubuntu image that ships with the pcDuino, and this Android image can be loaded as well.  
Installing drivers (Windows only)  
  
Before going any farther, you should install the drivers for the pcDuino debugging port. Make sure that you have the right version of LiveSuit–the drivers in the 32-bit version won’t work on a 64-bit system!  
  
    Plug a microUSB cable into your computer. While holding the button marked SW2 on your pcDuino, plug the other end of the cable into the USB-OTG port on your pcDuino.  
  
    Labeled hardware for updating flash  
  
    In Windows XP, you’ll see a driver installation query pop up. In Windows 7, you’ll probably have to open the device manager. The device will appear as “USB Device(VID\_1f3a\_PIDefe8)” under the “Universal Serial Bus controllers” section. The drivers are in the LiveSuit directory, in the “UsbDriver” directory.  
  
    Device manager for Windows 7  
  
    Once driver installation has completed, unplug the pcDuino.  
  
Installing Android  
  
    Launch the LiveSuit application.  
    Select the Android image file you downloaded above.  
  
    Plug a microUSB cable into your computer. While holding the button marked SW2 on your pcDuino, plug the other end of the cable into the USB-OTG port on your pcDuino.  
  
    LiveSuit first format message  
  
    After a few seconds, LiveSuit will popup with a window asking if you want to format the device. Since we’re reflashing, choose “Yes”. Note that even if mandatory doesn’t format, you’ll lose anything installed on the device, so back it up!  
  
    LiveSuit second format message  
  
    LiveSuit will double check that you want to format the device. Click “Yes” again.  
  
    Update success message  
  
    Click “Finish” in the next window. Don’t worry about the directions in the window; they’re not important. Or comprehensible.  
  
    LiveSuit updating  
  
    The progress bar will fill in, and if you’ve got a connection on the serial port terminal, you can watch the progress there as well. Don’t be alarmed if there are some long pauses during the update, although the whole process should not take more than five minutes.  
  
    LiveSuit will give you an “Update success” window, and the pcDuino will automatically reboot into Android. Note that when watching the serial debugger port output, Android doesn’t give you any kind of a prompt–just a blank line, into which you can enter commands.  
  
Installing Ubuntu  
  
    Launch the LiveSuit application.  
    Extract the Ubuntu archive you downloaded above. In it are two image files: “a10\_kernel\_20130203.img” and “ubuntu\_20130203.img”. Select the “a10\_kernel\_20130203.img” file.  
    Copy the files “env\_nandd.fex”, “ubuntu\_20130203.img”, and “update.sh” from the Ubuntu directory you just extracted to either a microSD card or a flash drive. Make sure the files are in the root directory of the drive!  
    Insert the flash drive or microSD card into the pcDuino.  
  
    Plug a microUSB cable into your computer. While holding the button marked SW2 on your pcDuino, plug the other end of the cable into the USB-OTG port on your pcDuino.  
  
    LiveSuit first format message  
  
    After a few seconds, LiveSuit will popup with a window asking if you want to format the device. Since we’re reflashing, choose “Yes”. Note that even if mandatory doesn’t format, you’ll lose anything installed on the device, so back it up!  
  
    LiveSuit second format message  
  
    LiveSuit will double check that you want to format the device. Click “Yes” again.  
  
    LiveSuit update complete  
  
    Click “Finish” in the next window. Don’t worry about the directions in the window; they’re not important. Or comprehensible.  
  
    LiveSuit updating  
  
    The progress bar will fill in, and if you’ve got a connection on the serial port terminal, you can watch the progress there, as well. Don’t be alarmed if there are some long pauses during the update, but it won’t take more than five minutes to complete.  
    LiveSuit will open an “Update success” message. Do not disconnect the power from the pcDuino at this time.  
  
Serial monitor after Ubuntu image load complete  
  
    If you’re connected to the serial monitor, you can watch the device update from the external memory. It takes some time–several minutes at least–and at the end, an “update complete” message will appear on the serial monitor.  
  
    If you’re not connected to the serial monitor, you’ll know the update process is completed when the RXLED and TXLED will blink slowly, in unison. While the upgrade is underway, RX will be solid on and TX will be blinking.  
    Once the upgrade is complete, you may restart the pcDuino.