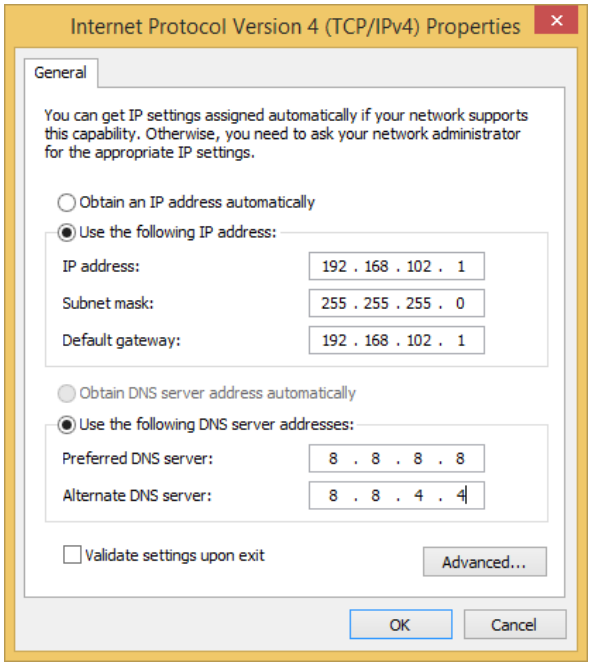
Connecting to Pi:

- SSH (Raspberry Pi's IP address is 192.168.102.3, so 'default gateway' and 'ip address' in TCP/IP v4 settings in Windows should be 192.168.102.1 as seen here:



- With this set up on the ethernet adapter, it should be possible to connect to the Pi via direct ethernet connection and then use a program such as PuTTY to connect over SSH to IP address 192.168.102.3 over port 22.
    - "MobaXTerm" is an alternative to PuTTY which combines SCP (file-transfer) functionality, among others
    - Log into the Pi using username: **pi** and password: **mpu9150**

Program notes:

- Program is located in **./home/pi/SP14** (SP14 folder visible once logged in as user **pi**
- The program was made to auto-start after power-up in the **/etc/rc.local** script (edit with command "sudo nano /etc/rc.local"

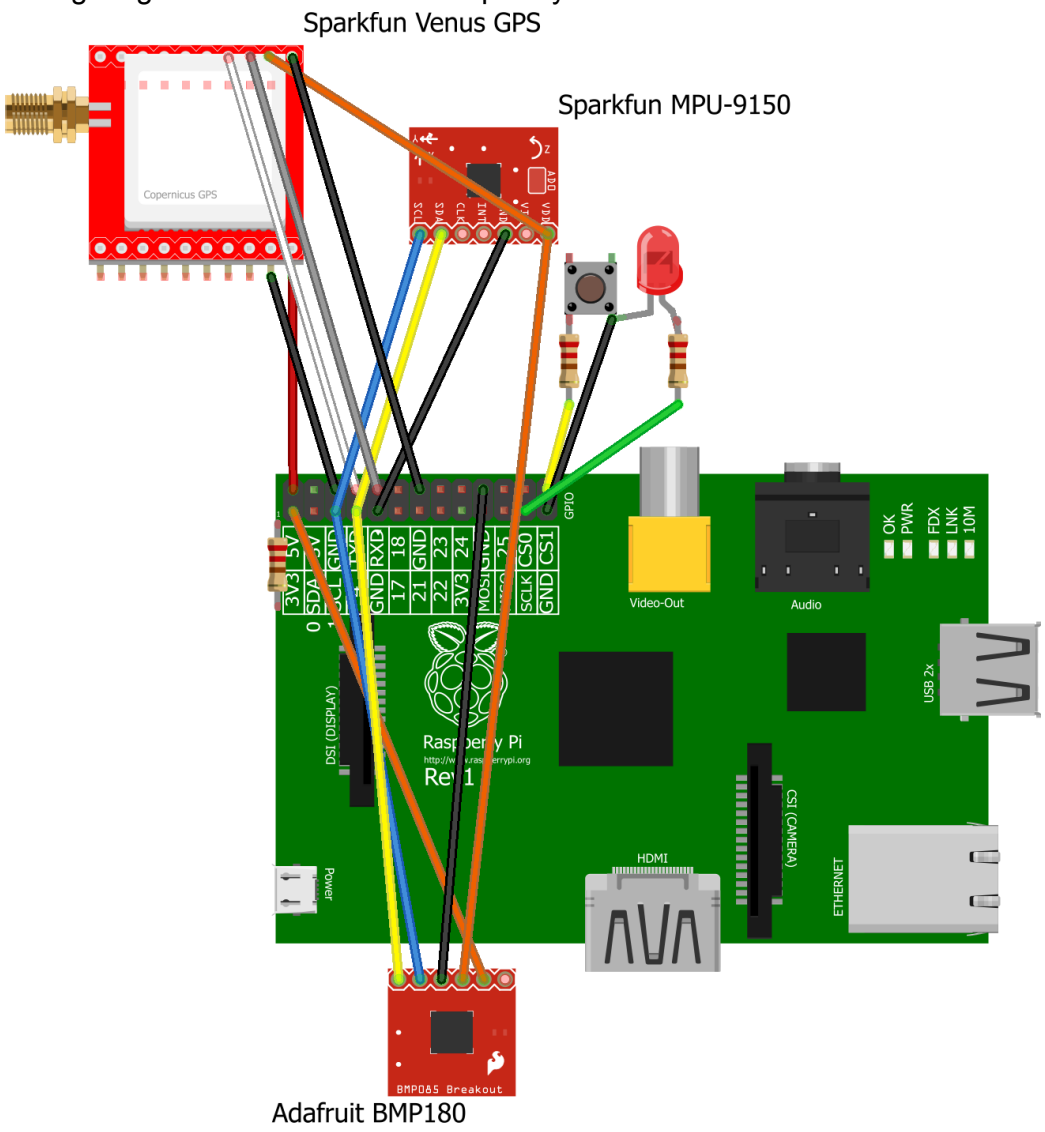
```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi

# auto-start GISF program
/home/pi/SP14/test1

exit 0
```

- File transfers can be accomplished via many ways, including any of the following:
    - SCP (Mac, Linux) or PSCP (Windows) - a command-line file transfer utility
    - MobaXTerm has integrated SCP functionality, greatly reducing time to view files on the Raspberry Pi ← preferred method when using Windows
    - Samba File sharing (adding the Pi as a “shared network drive” and then accessing files that way)
    - Edit the program to write to a file on an attached USB drive (with FAT32 file system) - this drive can then be unplugged and read by Windows
    - Read the SD card via Linux (untested, may be possible)
- Compile line for the main.cpp program:

```
g++ -Wall main.cpp Adafruit_BMP085_U.cpp I2Cdev.cpp MPU6050.cpp
-o test1 -lncurses -pthread -lrt -lwiringPi
```

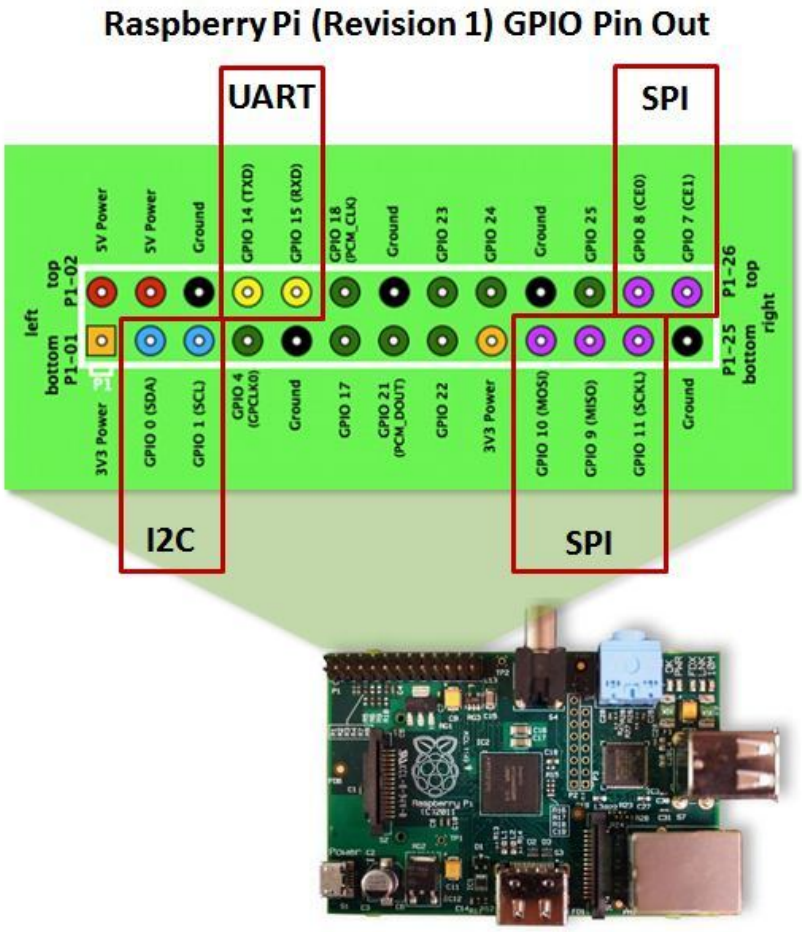
    - Items required for compilation in the main directory:
        - main.cpp
        - Adafruit_BMP085_U.cpp
        - I2Cdev.cpp
        - MPU6050.cpp
        - “include” directory and contents:
            - Adafruit_BMP085_U.h
            - helper_3dmath.h
            - I2Cdev.h
            - MPU6050.h
            - MPU6050_6Axis_MotionApps20.h
        - “Data” directory
        - “DataRaw” directory
    - Output file name: test1
        - Run via **sudo ./test1** when in same directory (sudo required for GPIO)

- Wiring diagram for sensors and Raspberry Pi



  - Note that sensor models are not exact, however Raspberry Pi connections are accurate. Simply connect:
    - RPi's 3.3V source to BMP180 and GPS 3.3V
    - BMP180 3Vo to MPU-9150 VCC
    - I^2C SDA and SCL to both MPU-9150 and BMP180 SDA and SCL pins
    - GPS TX0 and RX0 to Raspberry Pi's RX0 and TX0, respectively
    - RPi's 5V source to GPS' VBAT
    - LED+ via 1kΩ resistor to RPi GPIO 11 (SCLK)
    - PB+ via 1kΩ resistor to RPi GPIO 7 (CE1)
    - PB- and LED- to GND
    - 1x MPU-9150, 2x GPS, 1x BMP180 GNDs to available RPi GND pins

- For reference, Raspberry Pi pinout:



  - OK to use any of the GND pins
  - Second 3.3V Power (P1-17) should not be used