

POLOLU L3GD20H 3-AXIS GYRO CARRIER WITH

VOLTAGE REGULATOR

USER'S GUIDE

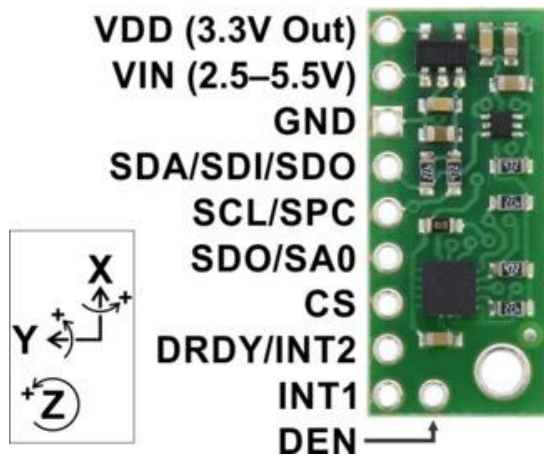
CONNECTIONS

Regardless of the interface being used to communicate with the L3GD20H, its VIN pin should be connected to a 2.5 V to 5.5 V source, and GND should be connected to 0 volts. (Alternatively, if you are using the gyro 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.)

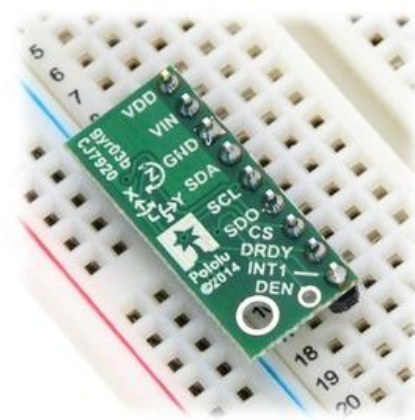
A minimum of two logic connections are necessary to use the L3GD20H in I²C mode (this is the default mode): SCL and SDA. These pins are connected to built-in level-shifters that make them safe to use at voltages over 3.3 V; they should be connected to an I²C bus operating at the same logic level as VIN. The remaining pins are not connected to level-shifters on the board and are not 5V-tolerant, but our 4-channel bidirectional logic level shifter can be used externally with those pins to achieve the same effect.

To use the L3GD20H in SPI mode, four logic connections are typically used: SPC, SDI, SDO, and CS. These should be connected to an SPI bus operating at the same logic level as VIN. The SPI interface operates in 4-wire mode by default, with SDI and SDO on separate pins, but it can be configured to use 3-wire mode so that SDO shares a pin with SDI.

PINOUT



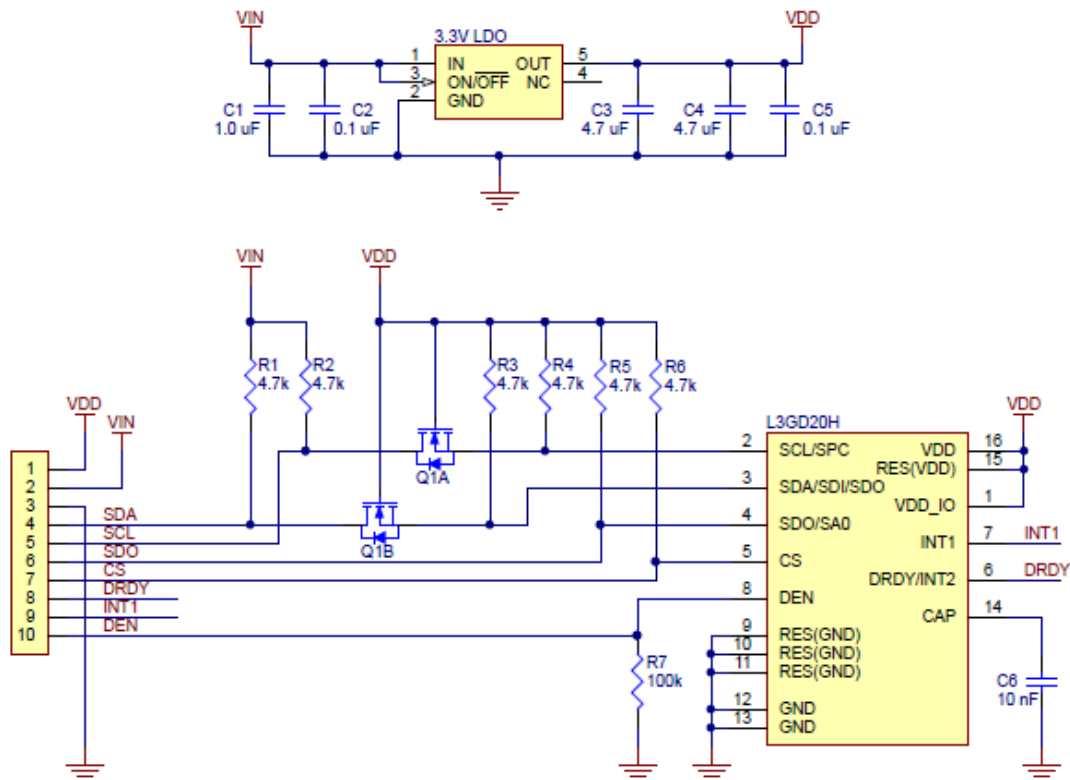
L3GD20H 3-axis gyro carrier with voltage regulator, labeled top view.



L3GD20H 3-axis gyro carrier with voltage regulator in a breadboard.

PIN	Description
VIN	This is the main power supply connection, and it should be powered at 2.5 V to 5.5 V. The SCL/SPC and SDA/SDI level shifters pull the I ² C and SPI bus high bits up to this level.
GND	The ground (0 V) connection for your power supply. Your I ² C or SPI control source must also share a common ground with this board.
VDD	Regulated 3.3 V output. Almost 150 mA is available to power external components. (If you want to bypass the internal regulator, you can instead use this pin as a 3.3 V input with VIN disconnected.)
SCL/SPC	Level-shifted I ² C/SPI clock line: HIGH is VIN, LOW is 0 V
SDA/SDI	Level-shifted I ² C data line and SPI data-in line (also doubles as SDO in 3-wire mode): HIGH is VIN, LOW is 0 V
SDO	SPI data-out line in 4-wire mode: HIGH is VDD, LOW is 0 V. Also used as a 3.3V-logic-level input to determine I ² C slave address (see below). <i>This pin is not level-shifted and is not 5V-tolerant.</i>
CS	SPI enable (chip select). Pulled up to VDD to enable I ² C communication by default; drive low to begin SPI communication.
DRDY/INT2	Data ready indicator, a 3.3V-logic-level output. HIGH (3.3 V) indicates angular rate data can be read. Can also be configured as a FIFO interrupt. <i>This output is not level-shifted.</i>
INT1	Programmable interrupt, a 3.3V-logic-level output. <i>This output is not level-shifted.</i>
DEN	Data enable trigger, a 3.3V-logic-level input. <i>This input is not level-shifted and is not 5V-tolerant.</i>

SCHEMATIC DIAGRAM



The above schematic shows the additional components the carrier board incorporates to make the L3GD20H easier to use, including the voltage regulator that allows the board to be powered from a 2.5 V to 5.5 V supply and the level-shifter circuit that allows for I²C and SPI communication at the same logic voltage level as VIN. This schematic is also available as a downloadable PDF (174k PDF).

I²C COMMUNICATION

With the CS pin in its default state (pulled up to VDD), the L3GD20H can be configured and its angular velocity readings can be queried through the I²C bus. Level shifters on the I²C clock (SCL) and data (SDA) lines enable I²C communication with microcontrollers operating at the same voltage as VIN (2.5 V to 5.5 V). A detailed explanation of the I²C

interface on the L3GD20H can be found in its datasheet (3MB pdf), and more detailed information about I²C in general can be found in NXP's I²C-bus specification (371k pdf).

In I²C mode, the gyro's 7-bit slave address has its least significant bit (LSb) determined by the voltage on the SDO pin. The carrier board pulls SDO to VDD through a 4.7 kΩ resistor, making the LSb 1 and setting the slave address to 1101011b by default (this is the same slave address as the L3GD20). If the gyro's selected slave address happens to conflict with some other device on your I²C bus, you can drive SDO low to set the LSb to 0.

In our tests of the board, we were able to communicate with the chip at clock frequencies up to 400 kHz; higher frequencies might work but were not tested.

SPI COMMUNICATION

To communicate with the L3GD20H in SPI mode, the CS pin (which the board pulls to VDD through a 4.7 kΩ resistor) must be driven low before the start of an SPI command and allowed to return high after the end of the command. Level shifters on the SPI clock (SPC) and data in (SDI) lines enable SPI communication with microcontrollers operating at the same voltage as VIN (2.5 V to 5.5 V).

In the default 4-wire mode, the gyro transmits data to the SPI master on a dedicated data out (SDO) line. If the SPI interface is configured to use 3-wire mode instead, the SDI line doubles as SDO and is driven by the L3GD20H when it transmits data to the master. A detailed explanation of the SPI interface on the L3GD20H can be found in its datasheet (3MB pdf).

SAMPLE CODE

We have written a basic Arduino library for the L3G that makes it easy to interface this sensor with an Arduino or Arduino-compatible board like an A-Star. The library makes it simple to configure the L3GD20H and read the raw gyro data through I²C.

PROTOCOL HINTS

The datasheet provides all the information you need to use this sensor, but picking out the important details can take some time. Here are some pointers for communicating with and configuring the L3GD20H that we hope will get you up and running a little bit faster:

- The gyro is in power down mode by default. You have to turn it on by writing the appropriate value to the CTRL1 register.
- You can read or write multiple registers in a single I²C command by asserting the most significant bit of the register address to enable address auto-increment.
- You can enable the same auto-increment feature in SPI mode by asserting the second bit (bit 1, called the MS bit in the datasheet) of an SPI command.