Sensor Fusion: Ultrasonic Sensor

Microcontroller --> PGA460 --> Murata Ultrasonic Sensor

**Abstract**

Ultrasonic sensors operate on the principle of time-of-flight for the purpose of distance measurement. The time-of-flight principle refers to a measurement of time taken by a particle to travel through a medium. The duration is then used to calculate the distance between the ultrasonic sensor and the barrier by taking d = v\*t/2, for which v is the speed of sound (a constant) and t is the duration the signal takes to travel to the barrier and back to the emitter. The division by 2 in the calculation ensures that the result is the distance between the sensor and barrier, and not the total distance traveled by the signal.

**Materials**

Microcontroller: Arduino Mega

Ultrasonic Driver/ Transducer: PGA460

Ultrasonic sensor: Murata ma58mf14-7n

Step up transformer and signal amplifier circuitry integrated into the system

**Vision:**

Establish communication between microcontroller, transducer, and ultrasonic sensor. Eventual multiplexing between one transducer and multiple ultrasonic sensor.

**Procedure**

In order to interface with the ultrasonic sensor and process returning signals, an embedded system of a microcontroller, transducer, and ultrasonic senor is needed. Due to specific operating settings (voltage, current, etc) a step-up transformer is needed between PGA460 and ultrasonic sensor. Listed below are summary of specifications of PGA460 and Murata ultrasonic sensor.

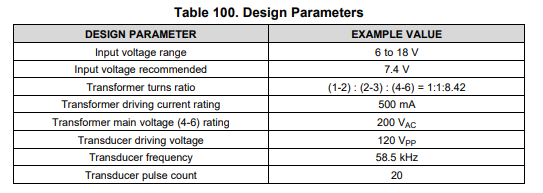
PGA460

Vin; 6-28V, optimum 7.4 V

Baud rate: up to 119200

Serial communication: SERIAL\_8N2

Distance range: Preset 1 (<1 meter) and Preset 2(>1 meter)



Murata ma58mf14-7n

Vin: 80-120Vpp (<20 pulses per second)

Center Frequency: 58 kHz

**Plans going forward:**

First stage:

* + Initializing settings for PGA460.
  + Establishing communication between PGA460 and microcontroller

Second stage:

* + Interfacing PGA460 with sensor
  + Sensor to pulse signals given by PGA460
  + Sensor to detect reflected signal by barrier
  + PGA460 to process reflected signal to derive distance
  + Communicating result back to microcontroller

Third Stage:

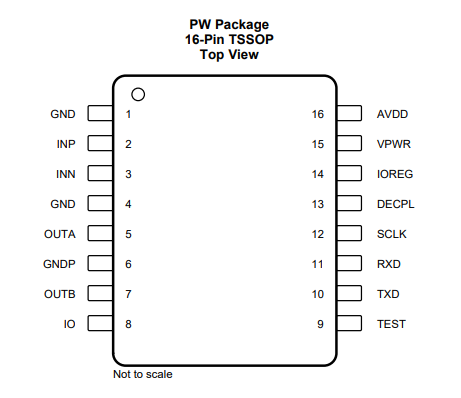
* + Driving multiple sensors with one transducer (Multiplexing)
  + Transmitting distance results back to microcontroller
  + Embedded system to run at rate more than 4Hz

Initialization of PGA460

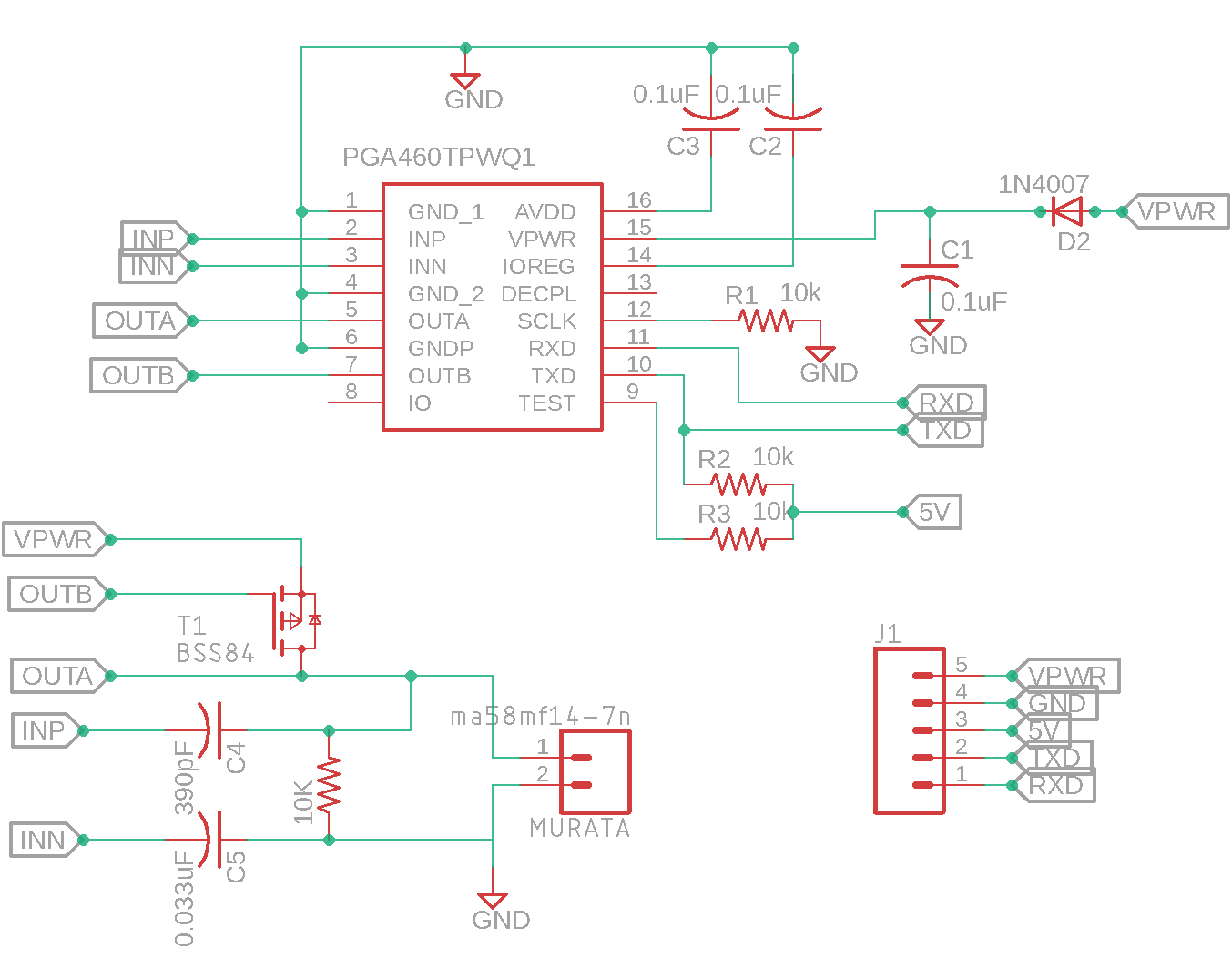
* Register configuration settings for threshold values
* Upon power boot or power cycle, commit configuration settings

Hardware Wiring

PGA460 Pinout



Direct-driven Schematics



Transformer-driven Schematics

**INSERT IMAGE**

For reference on design requirements behind circuitry interfacing with PGA460 with Arduino Mega 2560 and Murata ma58mf14-7n ultrasonic sensor, refer to [PGA460 Ultrasonic Signal Processor and Transducer Driver](http://www.ti.com/lit/ds/symlink/pga460.pdf) datasheet under Section 8 Application and Implementation.

Due to variances in how each component is manufactured, some components used in the circuitry do not exactly follow the suggested values indicated in the PGA460 Ultrasonic Signal Processor and Transducer Driver datasheet. However, the [PGA460 Ultrasonic Module Hardware and Software Optimization](:%20http:/www.tij.co.jp/jp/lit/an/slaa732/slaa732.pdf) guide, under Section 3.4 Passive Tuning, shows flexibility in how these values can be modified in order to best be tuned with the circuitry setup.

**SOFTWARE GUIDE**

PGA460 has four main modes of communication: UART, OWU, SPI, and TCI. For our purpose of measuring distances, we will be using UART, as it is the most common and simplest form of serial communication with Arduino MEGA 2560.

For first-time exposure to send commands and receive data from PGA460, it is recommended to read carefully the [PGA460 Software Development Guide](http://www.ti.com/lit/an/slaa730a/slaa730a.pdf). The same code is also available in the downloadable Energia Library example, which can be ported into the Arduino IDE Library and requires commenting out #include "Energia.h"to make the code compilable. Since SPI mode is not used, it is also suggested that #include "PGA460\_SPI.h" and related SPI functions and variables be commented out in PGA460\_USSC.h and PGA460\_USSC.cpp to prevent need of installing more unnecessary libraries. A cleaned-up version of the h and cpp files, with SPI functions can found PGA460, 0.25-0.5m folder.

The order of instructions for which PGA460 is set up to properly work is as follows:

1. **On power up,** configures EERPOM values to indicate which ultrasonic transducer to being used and how PGA460 should execute its commands. This step is **optional** **if** these EEPROM values have being burned once onto the PGA460.
2. Configures Threshold parameters by using threshold bulk write command (THRBW) or by independently writing a particular parameter by using register write command (SRW).
3. Configures Time-varying gain by using time-varying gain bulk write
4. **Once successfully configured**, program will execute the following commands in a loop:
   1. BURST+LISTEN (Preset1 or Preset 2)
   2. After record interval has expired, issue ultrasonic measurement result command(UMR) to retrieve data
   3. Use time-of-flight calculation on retrieved data to compute distance