Application Note:
Interfacing MMA8452Q Triple Axis Accelerometer Breakout board with I2C on Altera DE2

Group 8: Caitlin Smart and Troy Davis

OVERVIEW

The Freescale Semiconductor MMA8452Q Triple Axis Accelerometer is available on a breakout board through Sparkfun with part number SEN-10955. Here is a link to the part:

https://www.sparkfun.com/products/10955

This accelerometer uses interrupts, INT1 and INT2, which correspond to pins I1 and I2 on the breakout board and are customized through registers to choose the type of motion you want to measure. For our project, we chose to measure motion through the freefall/motion registers.

This accelerometer uses the I2C interface for communication. There are two I2C pins on the board, SDA and SCL. These correspond to the I2C pins I2C_SDAT and I2C_SCLK on the DE2 board. These pins are connected to the board through the GPIO.

We strongly support the use of an oscilloscope to verify I²C signals on SDA and SCL lines.

HARDWARE SETUP

Solder the pins of the breakout board to a small pin header to ensure a good connection. Attach the pin header and a GPIO pin header to a piece of punch board. Wire wrap the accelerometer pins to GPIO pins. We used GPIO_0(3),GPIO_0(5), and GPIO_0(7) for pins SDA, SCL, and I2. Connect 3.3V and GND pins to corresponding GPIO pins. We attached I1 but do not use it. Take care when choosing GPIO pins because the numbers are not in order and you will need to look at the DE2 schematics to find out which pin you are connected to. We have two accelerometers for redundancy connected in Fig. 1 and we are using the red accelerometer. The complete GPIO attachment with ribbon cable is shown in Fig. 2.

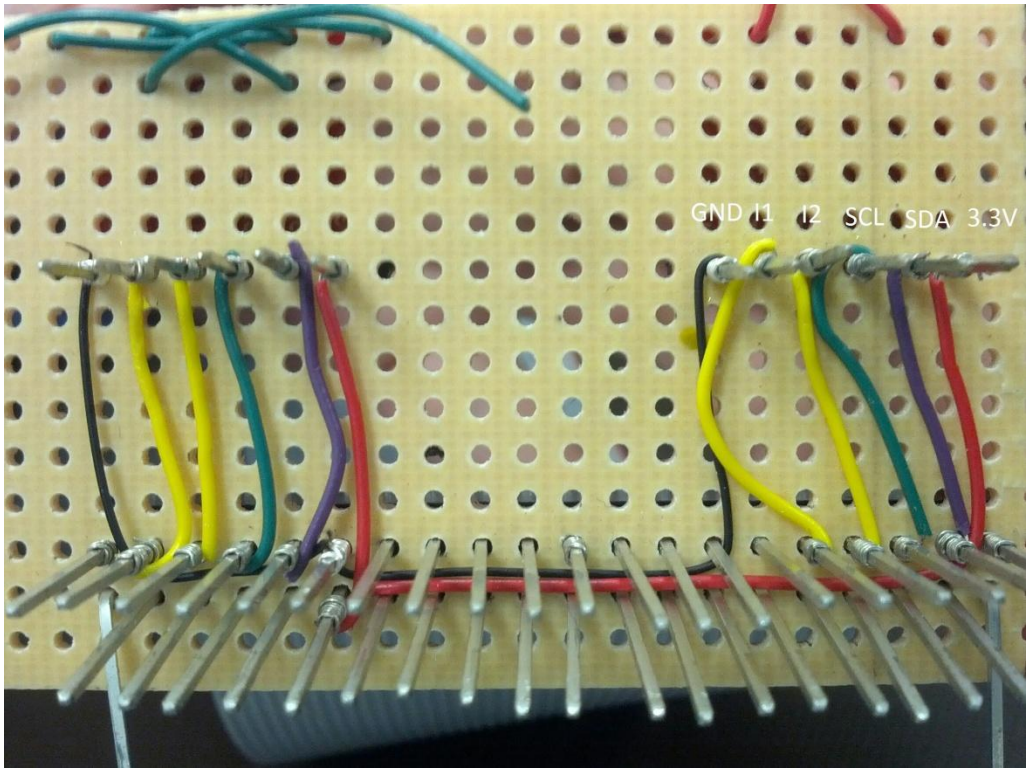


Fig. 1: Wire wrapping

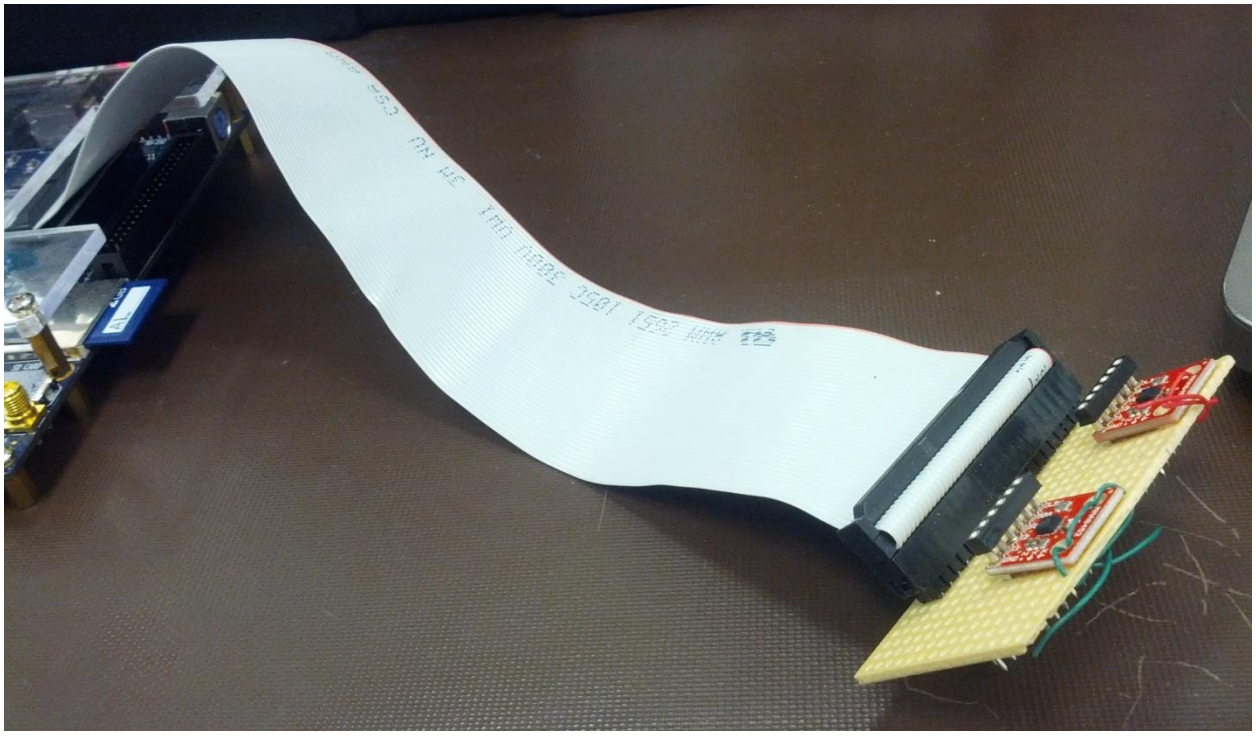


Fig. 2: Hardware setup

SOPC: Accelerometer Component and I²C Master

The accelerometer.vhd file in the ip folder is an implementation of an Avalon slave component. It contains the hardware implementation of the accelerometer device. It implements a tristate bus for the sda I²C line as well as a hardware counter for the interrupts generated by the accelerometer device.

The opencores_i2c_master.vhd file in the ip folder is an implementation of an Avalon slave component. It contains the hardware implementation of an I²C master device. It implements three other I²C vhdl components in order to provide I²C bus master functionality.

Other standardized Altera components are instantiated in the accel_demo.sopc file for demonstration purposes.

VHDL Top Level

The top level file is named accel_demo_TopLevel.vhd. The relevant mappings for the accelerometer are between the accelerometer component, the I²C component, and the GPIO pins. As explained earlier, the accelerometer facing signals are: interrupt 2, tied to GPIO_0(7), scl_i2c_to_accel, tied to GPIO_0(5), sda_i2c_to_accel, tied to GPIO_0(3). The I²C master facing signals are scl_i2c_to_bus and sda_i2c_to_bus , both of which are tied to the imaginary i2c bus signals created for this demo. In addition to these, the sda_padoen_o signal is rerouted from the I²C master

to the accelerometer component to help with the tristate design. Values for implementation for the other accelerometer device using GPIO pins 31, 29, and 27 are also included.

C CODE

The file main.c initializes µC-OS II data structures and creates a task, task_acceleromer, to handle accelerometer functionality. This demo calculates how often the accelerometer is shaken on a per minute basis. It does this by averaging 50 samples taken over the course of 25 seconds. This was done to increase accuracy while providing a robust design. The results are printed to the DE2 LCD as well as the IDE console.

Accelerometer and I²C driver code is also included:

accelerometer.c implements five functions:

accelerometer_write formats data into an I²C capable format and perfoms an I²C write.

accelerometer_read performs an I²C read and returns the value.

init_accelerometer validates communication with accelerometer by reading WHO_AM_I register and comparing it to known value. Contains all initialization of accelerometer registers. For this demo, transient motion detection was used with a threshold value of 0.375g and a debounce setting to 150ms to stop interrupt generation for 150ms after an interrupt was detected to prevent jitter.

MMA8452Standby sets the accelerometer into standby mode which is required for register changes.

MMA8452Active sets the accelerometer into active mode which is required to output data.

i2c_ctrl.c implements five functions:

short_delay generates a short delay. Used to space out many I²C commands.

i2c_wait_tip prevents further code execution when a transfer is in progress.

i2c_write formats data into I²C bytes and performs an I²C write.

i2c_read performs I²C read operation with a forced NAK to signify read operation completion.

init_i2c sets pre-scaling for I²C master and enables core.