

Lab 9: Digital Level

Digital Design and Computer Architecture: RISC-V Edition (Harris & Harris, Elsevier © 2021)

Objective

The purpose of this lab is to write embedded software to control external peripherals. Specifically, you will build a 2D digital level with an accelerometer and LED matrix. The LED matrix should display a “bubble” indicating whether the accelerometer is level along the x and y axes.

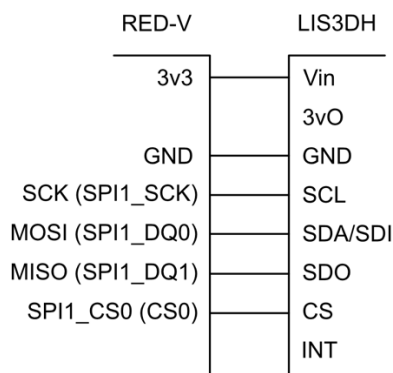
1. System Requirements

Your system should include a RED-V board, a SparkFun LIS3DH triple-axis accelerometer breakout board, a KWM-20882CVB 8x8 LED matrix, and resistors for the LED matrix, all wired together on a breadboard.

The system should use 7x7 of the LEDs to display a bubble indicating the orientation of the accelerometer. When the accelerometer is level, the bubble should be in the center. For each (approximately) 5 degrees that you tilt the accelerometer in the x or y directions, the bubble should move one dot in the corresponding direction on the LED matrix. The bubble should stop at the edge of the matrix if the tilt is too great. You do not need to use a level or protractor to calibrate exactly 5 degrees per dot; just eyeball it to look reasonable.

2. Suggested Approach

- Browse the datasheets for the accelerometer and LED matrix on the class page until you understand how they work.
- Wire the accelerometer to your RED-V board as shown below.



- Create a new PlatformIO project for the RED-V ThingPlus, download the source code files from the course website (lab8starter.c, EasyREDVIO_ThingPlus.h, and REDV_SPI.h), and place them in the src folder of the project. Run the

lab8starter.c SPI sample code. Make sure that you can read the ID register as expected, and then that you can get reasonable readings from the accelerometer. Calibrate your accelerometer by identifying the readings when it is level and how they change as you tilt it.

- d) Turn on an LED in your LED array using a resistor and power and ground to make sure you understand how the LED array operates. Remember that you may burn out the LED array if you don't include the current-limiting resistor. Pay attention to the pin numbering in the datasheet.
- e) Hook the LED array up to the ports of your RED-V board, taking care to use current-limiting resistors appropriately. Write and test a C function to turn on an LED given an (x, y) coordinate.
- f) Write the rest of your program to periodically read the accelerometer, convert the reading to a tilt in the x and y directions, and turn on the appropriate LED. A small delay between reads will reduce flickering.

What to Turn In

- 1. Please indicate how many hours you spent on this lab. This will be helpful for calibrating the workload for next time the course is taught.
- 2. Schematic of the circuit on your breadboard, showing all components and which RED-V pins are connected.
- 3. C code for your digital level.
- 4. Does your program work? Can you tilt the accelerometer and move the LED bubble across the array in both axes as expected and reach every dot in the 7x7 array?

Please indicate any bugs you found in this lab manual, or any suggestions you would have to improve the lab.