Linux Kernel Basecamp Project

Kyiv, 2023
Vlad Degtyarov

Inclinometer

An inclinometer is a device used to measure the angle of slope or inclination of an object with respect to the horizontal plane. It is also known as a tilt sensor, clinometer or slope meter. Inclinometers work by detecting changes in gravitational forces or changes in the orientation of an object. Some inclinometers use a pendulum or a bubble level to indicate the slope angle, while others use electronic sensors such as accelerometers and gyroscopes to measure the angle of inclination with greater precision.

The purpose of the project

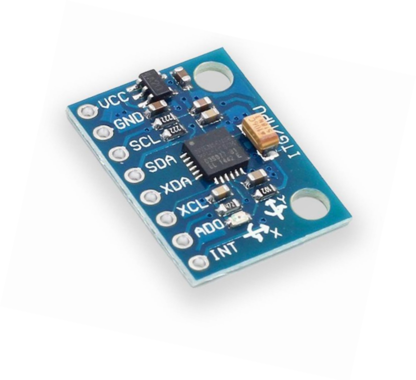
The main goal of this project is to practice writing Linux kernel modules by creating a simple device and writing its drivers.

Components

Raspberry Pi Zero W

The Raspberry Pi Zero W is a compact single-board computer.
It features a 1 GHz single-core ARM11 processor, 512MB RAM, and a microSD card slot for storage. It has mini-HDMI and micro-USB ports, along with a 40-pin GPIO header for interfacing with external devices. Additionally, it includes built-in Wi-Fi and Bluetooth connectivity for wireless communication. The Broadcom BCM2835 SoC, used in the Raspberry Pi Zero models, features a 1 GHz single-core ARM11 processor. The ARM11 is an ARMv6 architecture, which is a 32-bit instruction set architecture. With its small form factor, GPIO pins, and integrated features, it is widely used in IoT projects and other applications.

Components

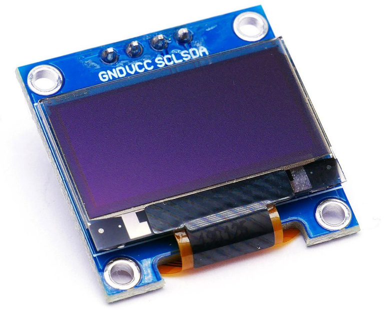


MPU6050 based sensor module

The MPU6050 is a popular 6-axis gyroscope and accelerometer sensor module. It combines a 3-axis gyroscope and a 3-axis accelerometer into a single chip, allowing it to measure both angular velocity and linear acceleration in three dimensions. The MPU6050 is widely used in projects involving motion tracking, orientation sensing, and stabilization.

Our MPU6050 sensor module is equipped with an I^2C (Inter-Integrated Circuit) interface.

Components

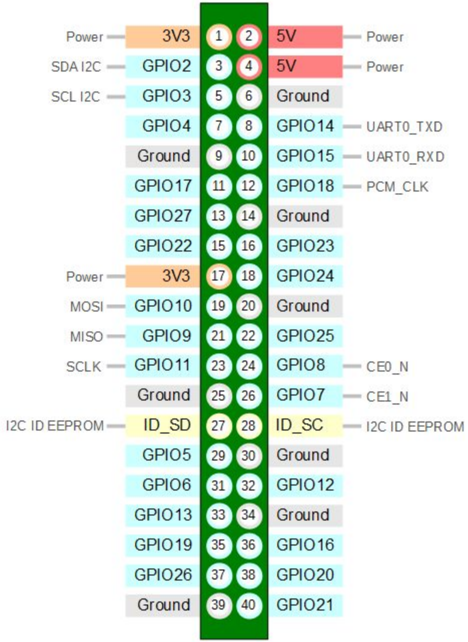


SSD1306 based OLED 0.96 inch display module

SSD1306 is a versatile and popular chip features integrated RAM and a built-in display controller for easy implementation. Its low power consumption makes it ideal for battery-powered devices. Overall, it is a popular choice for compact and power-efficient displays. It supports both I2C and SPI communication interfaces, making it compatible with a wide range of microcontrollers.

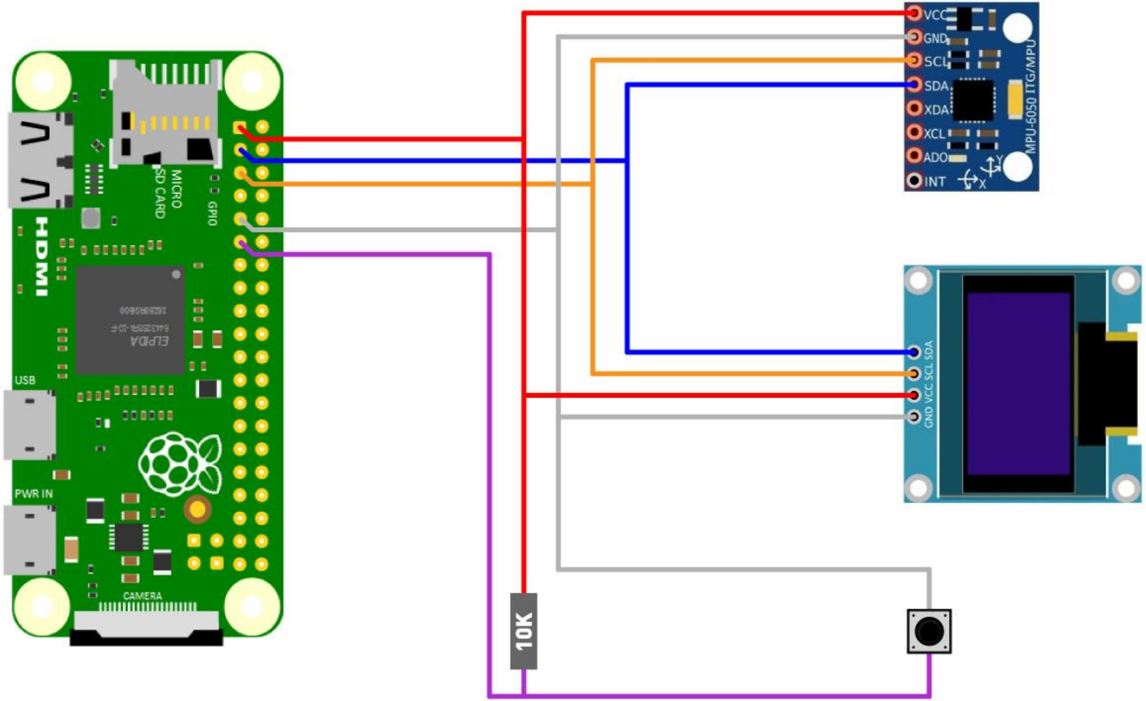
Our display module is equipped with an I^2C (Inter-Integrated Circuit) interface and has a monochrome screen with a resolution of 128x64 pixels.

Wiring diagram

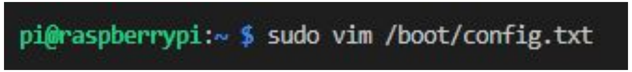


Power	3V3	1	2	5V	Power
SDA I2C	GPIO2	3	4	5V	Power
SCL I2C	GPIO3	5	6	Ground	
	GPIO4	7	8	GPIO14	UART0_TXD
	Ground	9	10	GPIO15	UART0_RXD
	GPIO17	11	12	GPIO18	PCM_CLK
	GPIO27	13	14	Ground	
	GPIO22	15	16	GPIO23	
Power	3V3	17	18	GPIO24	
MOSI	GPIO10	19	20	Ground	
MISO	GPIO9	21	22	GPIO25	
SCLK	GPIO11	23	24	GPIO8	CE0_N
	Ground	25	26	GPIO7	CE1_N
I2C ID EEPROM	ID_SD	27	28	ID_SC	I2C ID EEPROM
	GPIO5	29	30	Ground	
	GPIO6	31	32	GPIO12	
	GPIO13	33	34	Ground	
	GPIO19	35	36	GPIO16	
	GPIO26	37	38	GPIO20	
	Ground	39	40	GPIO21	

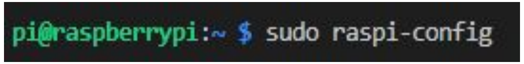
Raspberry Pi Zero Pinout



Enable I^2C on Raspberry Pi



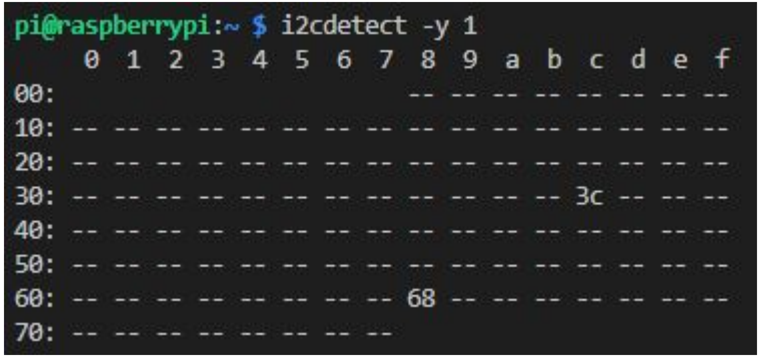
```
pi@raspberrypi:~ $ sudo vim /boot/config.txt
```



```
pi@raspberrypi:~ $ sudo raspi-config
```

Add or uncomment **`dtparam=i2c_arm=on`**
Save and reboot Raspberry Pi

Detect devices

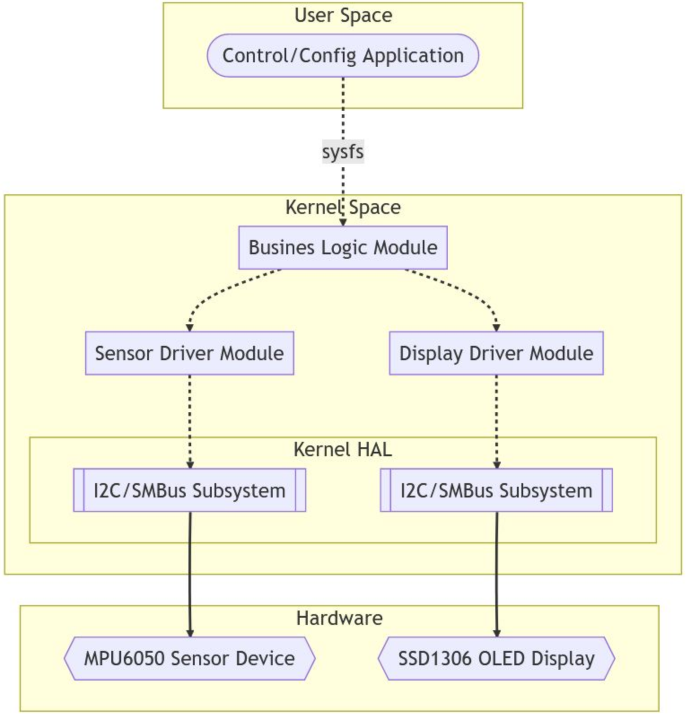


```
pi@raspberrypi:~ $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- 3c -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

If the i2cdetect command is not found, install it with:

```
sudo apt install i2c-tools
```

Project Design Diagram

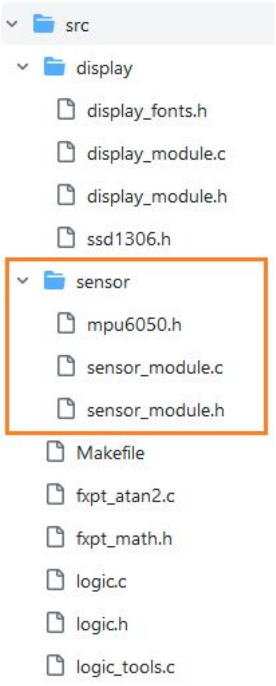


1. **Sensor driver module** for reading data from the mpu6050 sensor

2. **Display driver module** to display information on the OLED screen

3. **Business Logic module** for serving interfaces, polling and managing data, scheduling and deciding what to display

Sensor Driver Module



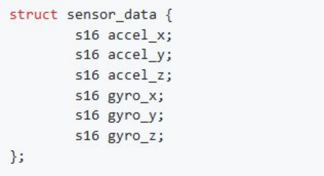
```
src
  display
    display_fonts.h
    display_module.c
    display_module.h
    ssd1306.h
  sensor
    mpu6050.h
    sensor_module.c
    sensor_module.h
  Makefile
  fxpt_atan2.c
  fxpt_math.h
  logic.c
  logic.h
  logic_tools.c
```

The sensor driver module creates an i2c client for working with the bus and adds the correspondingly described mpu6050 device driver to the kernel.

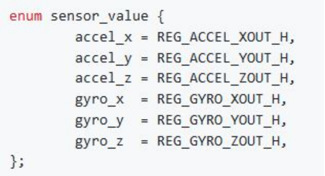
This module does not have an interface with the user space.

The purpose of the module is to poll the sensor device via I²C linux kernel subsystem and transfer information to the business logic module.

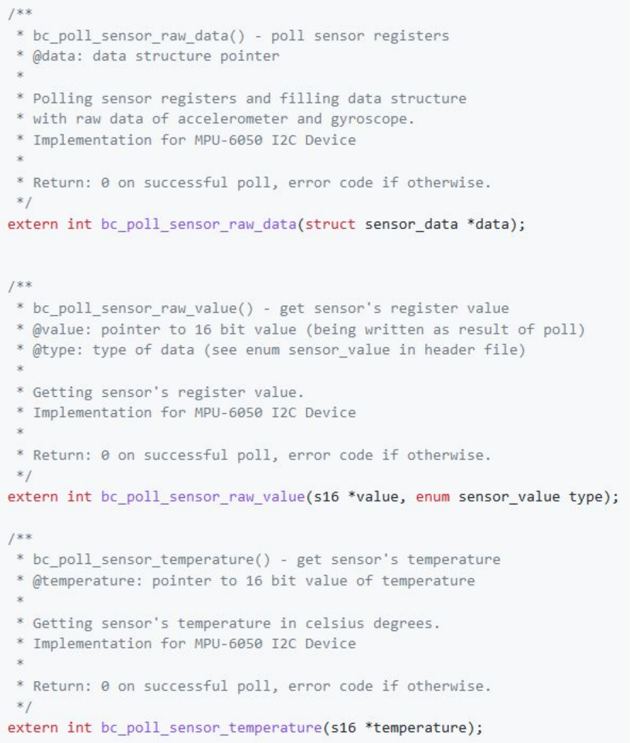
The Business Logic module uses this module's API, which is passed to the kernel using EXPORT_SYMBOL



```
struct sensor_data {
        s16 accel_x;
        s16 accel_y;
        s16 accel_z;
        s16 gyro_x;
        s16 gyro_y;
        s16 gyro_z;
};
```

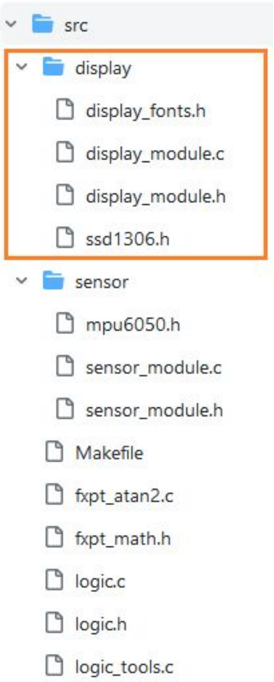


```
enum sensor_value {
        accel_x = REG_ACCEL_XOUT_H,
        accel_y = REG_ACCEL_YOUT_H,
        accel_z = REG_ACCEL_ZOUT_H,
        gyro_x  = REG_GYRO_XOUT_H,
        gyro_y  = REG_GYRO_YOUT_H,
        gyro_z  = REG_GYRO_ZOUT_H,
};
```



```
/**
 * bc_poll_sensor_raw_data() - poll sensor registers
 * @data: data structure pointer
 *
 * Polling sensor registers and filling data structure
 * with raw data of accelerometer and gyroscope.
 * Implementation for MPU-6050 I2C Device
 *
 * Return: 0 on successful poll, error code if otherwise.
 */
extern int bc_poll_sensor_raw_data(struct sensor_data *data);


/**
 * bc_poll_sensor_raw_value() - get sensor's register value
 * @value: pointer to 16 bit value (being written as result of poll)
 * @type: type of data (see enum sensor_value in header file)
 *
 * Getting sensor's register value.
 * Implementation for MPU-6050 I2C Device
 *
 * Return: 0 on successful poll, error code if otherwise.
 */
extern int bc_poll_sensor_raw_value(s16 *value, enum sensor_value type);

/**
 * bc_poll_sensor_temperature() - get sensor's temperature
 * @temperature: pointer to 16 bit value of temperature
 *
 * Getting sensor's temperature in celsius degrees.
 * Implementation for MPU-6050 I2C Device
 *
 * Return: 0 on successful poll, error code if otherwise.
 */
extern int bc_poll_sensor_temperature(s16 *temperature);
```

Display Driver Module



```
src
  display
    display_fonts.h
    display_module.c
    display_module.h
    ssd1306.h
  sensor
    mpu6050.h
    sensor_module.c
    sensor_module.h
  Makefile
  fxpt_atan2.c
  fxpt_math.h
  logic.c
  logic.h
  logic_tools.c
```

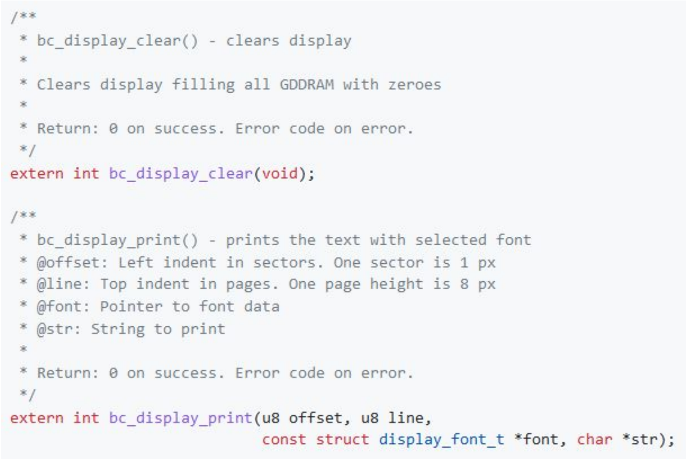
The display driver module creates an i2c client for working with the bus and adds the correspondingly described ssd1306 device driver to the kernel. After initialization, the display is ready for use.

This module also does not have an interface with the user space.

The purpose of the driver is to control display module over I²C linux kernel subsystem.

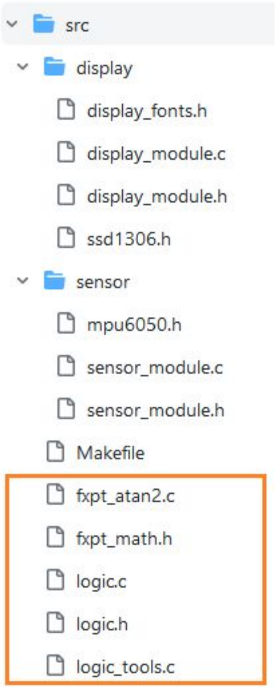
It is able to print text messages using different predefined fonts.

The Business Logic module uses this module's API, which is passed to the kernel using EXPORT_SYMBOL



```c
/**
 * bc_display_clear() - clears display
 *
 * Clears display filling all GDDRAM with zeroes
 *
 * Return: 0 on success. Error code on error.
 */
extern int bc_display_clear(void);

/**
 * bc_display_print() - prints the text with selected font
 * @offset: Left indent in sectors. One sector is 1 px
 * @line: Top indent in pages. One page height is 8 px
 * @font: Pointer to font data
 * @str: String to print
 *
 * Return: 0 on success. Error code on error.
 */
extern int bc_display_print(u8 offset, u8 line,
                            const struct display_font_t *font, char *str);
```

Inclinometer: Business Logic Module



- src
  - display
    - display_fonts.h
    - display_module.c
    - display_module.h
    - ssd1306.h
  - sensor
    - mpu6050.h
    - sensor_module.c
    - sensor_module.h
  - Makefile
  - fxpt_atan2.c
  - fxpt_math.h
  - logic.c
  - logic.h
  - logic_tools.c

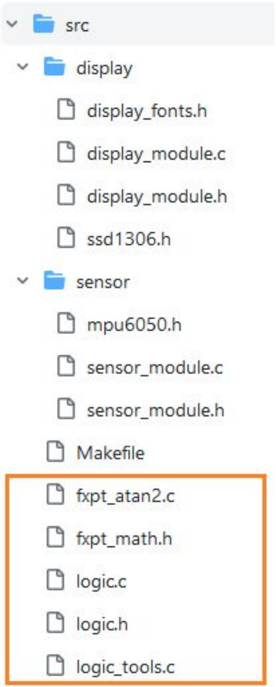
The purpose of the module:

- Connect all together
- Make sysfs interface to user space
- Register IRQ handlers
- Read sensor calibration information
- Receive and manage sensor data
- Select the information to display according to the selected mode
- Schedule the work loop

Typical loop cycle

1. Read sensor data
2. Prepare data due to selected mode
3. Display data

Modes



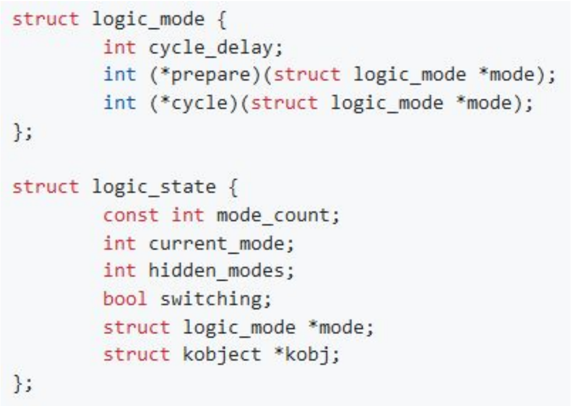
- src
  - display
    - display_fonts.h
    - display_module.c
    - display_module.h
    - ssd1306.h
  - sensor
    - mpu6050.h
    - sensor_module.c
    - sensor_module.h
  - Makefile
  - fxpt_atan2.c
  - fxpt_math.h
  - logic.c
  - logic.h
  - logic_tools.c

Every logic mode have two callbacks:

`prepare` – used for prepare the mode
`cycle` – used to perform work in a loop

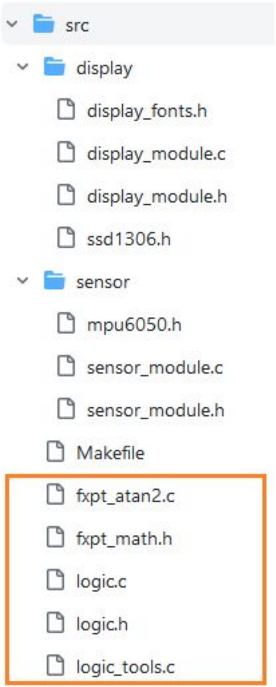
For switching the mode module will execute prepare callback and schedule the loop for cycle callback.

We can select modes from user space via sysfs or by pushing action button which will select next mode.



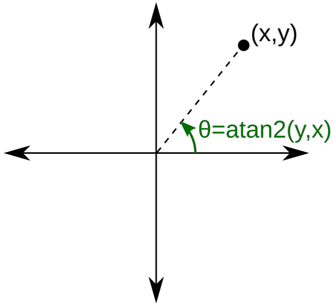
```
struct logic_mode {
        int cycle_delay;
        int (*prepare)(struct logic_mode *mode);
        int (*cycle)(struct logic_mode *mode);
};

struct logic_state {
        const int mode_count;
        int current_mode;
        int hidden_modes;
        bool switching;
        struct logic_mode *mode;
        struct kobject *kobj;
};
```

Fixed point math

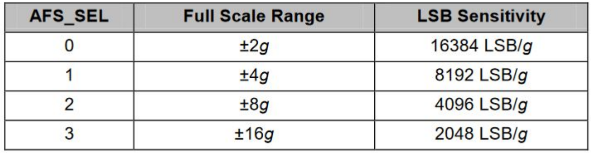


- src
  - display
    - display_fonts.h
    - display_module.c
    - display_module.h
    - ssd1306.h
  - sensor
    - mpu6050.h
    - sensor_module.c
    - sensor_module.h
  - Makefile
  - fxpt_atan2.c
  - fxpt_math.h
  - logic.c
  - logic.h
  - logic_tools.c

Accelerometer



$$\operatorname{atan2}(y,x)=\begin{cases}\arctan\left(\frac{y}{x}\right) & x>0\\ \arctan\left(\frac{y}{x}\right)+\pi & y\geq 0, x<0\\ \arctan\left(\frac{y}{x}\right)-\pi & y<0, x<0\\ +\frac{\pi}{2} & y>0, x=0\\ -\frac{\pi}{2} & y<0, x=0\\ \text{undefined} & y=0, x=0\end{cases}$$

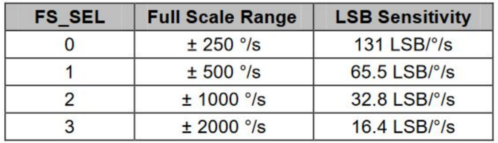


AFS_SEL	Full Scale Range	LSB Sensitivity
0	±2*g*	16384 LSB/*g*
1	±4*g*	8192 LSB/*g*
2	±8*g*	4096 LSB/*g*
3	±16*g*	2048 LSB/*g*

Gyroscope

$$\varphi\ (t) = \int \omega(t) * dt$$

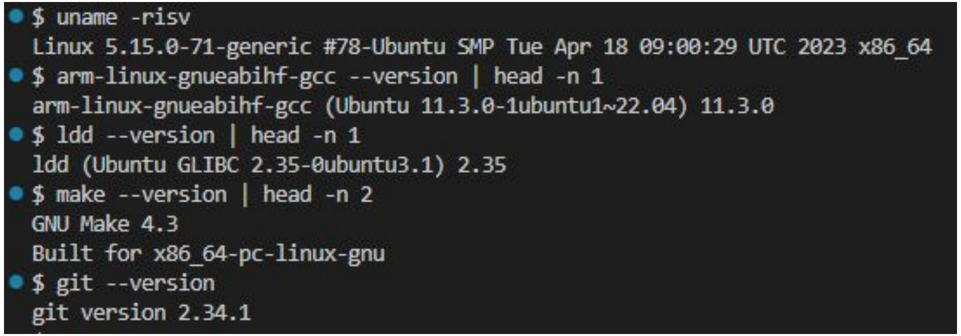
$$\varphi\ (t) = \varphi\ (0) + \sum \omega * \Delta t$$



FS_SEL	Full Scale Range	LSB Sensitivity
0	± 250 °/s	131 LSB/°/s
1	± 500 °/s	65.5 LSB/°/s
2	± 1000 °/s	32.8 LSB/°/s
3	± 2000 °/s	16.4 LSB/°/s

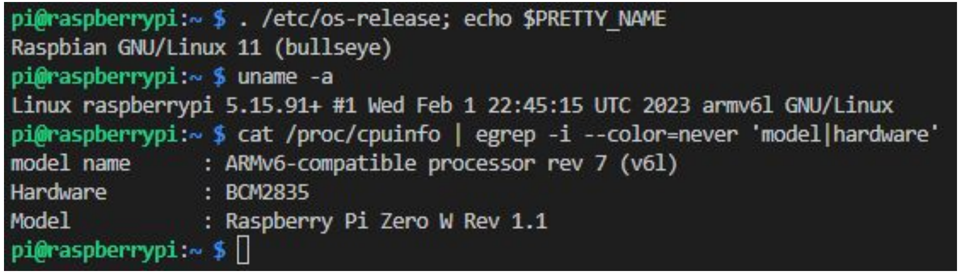
Environment

Cross compile toolchain on build machine



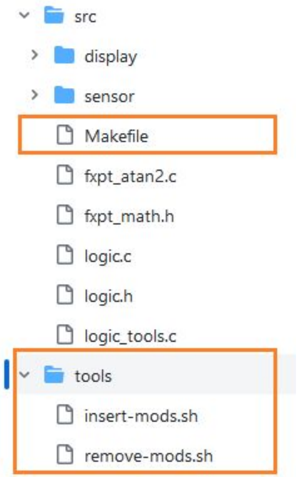
```
$ uname -risv
Linux 5.15.0-71-generic #78-Ubuntu SMP Tue Apr 18 09:00:29 UTC 2023 x86_64
$ arm-linux-gnueabihf-gcc --version | head -n 1
arm-linux-gnueabihf-gcc (Ubuntu 11.3.0-1ubuntu1~22.04) 11.3.0
$ ldd --version | head -n 1
ldd (Ubuntu GLIBC 2.35-0ubuntu3.1) 2.35
$ make --version | head -n 2
GNU Make 4.3
Built for x86_64-pc-linux-gnu
$ git --version
git version 2.34.1
```

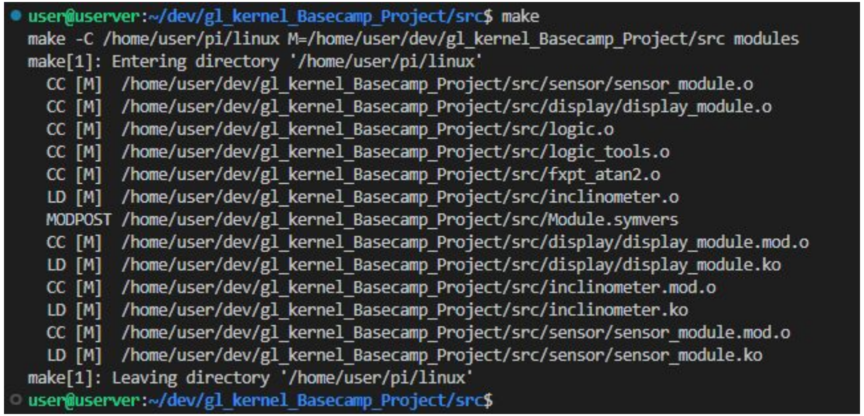
Raspberry Pi



```
pi@raspberrypi:~ $ . /etc/os-release; echo $PRETTY_NAME
Raspbian GNU/Linux 11 (bullseye)
pi@raspberrypi:~ $ uname -a
Linux raspberrypi 5.15.91+ #1 Wed Feb 1 22:45:15 UTC 2023 armv6l GNU/Linux
pi@raspberrypi:~ $ cat /proc/cpuinfo | egrep -i --color=never 'model|hardware'
model name      : ARMv6-compatible processor rev 7 (v6l)
Hardware        : BCM2835
Model           : Raspberry Pi Zero W Rev 1.1
pi@raspberrypi:~ $
```

Make and tools

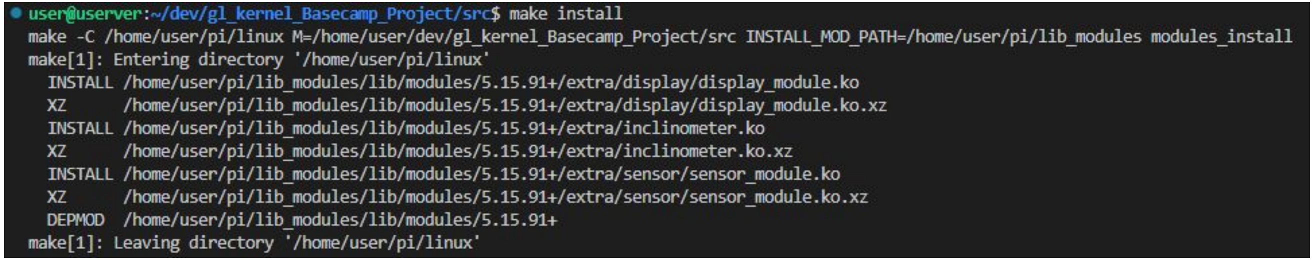




```
user@userver:~/dev/gl_kernel_Basecamp_Project/src$ make
make -C /home/user/pi/linux M=/home/user/dev/gl_kernel_Basecamp_Project/src modules
make[1]: Entering directory '/home/user/pi/linux'
  CC [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/sensor/sensor_module.o
  CC [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/display/display_module.o
  CC [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/logic.o
  CC [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/logic_tools.o
  CC [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/fxpt_atan2.o
  LD [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/inclinometer.o
  MODPOST /home/user/dev/gl_kernel_Basecamp_Project/src/Module.symvers
  CC [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/display/display_module.mod.o
  LD [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/display/display_module.ko
  CC [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/inclinometer.mod.o
  LD [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/inclinometer.ko
  CC [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/sensor/sensor_module.mod.o
  LD [M]  /home/user/dev/gl_kernel_Basecamp_Project/src/sensor/sensor_module.ko
make[1]: Leaving directory '/home/user/pi/linux'
user@userver:~/dev/gl_kernel_Basecamp_Project/src$
```

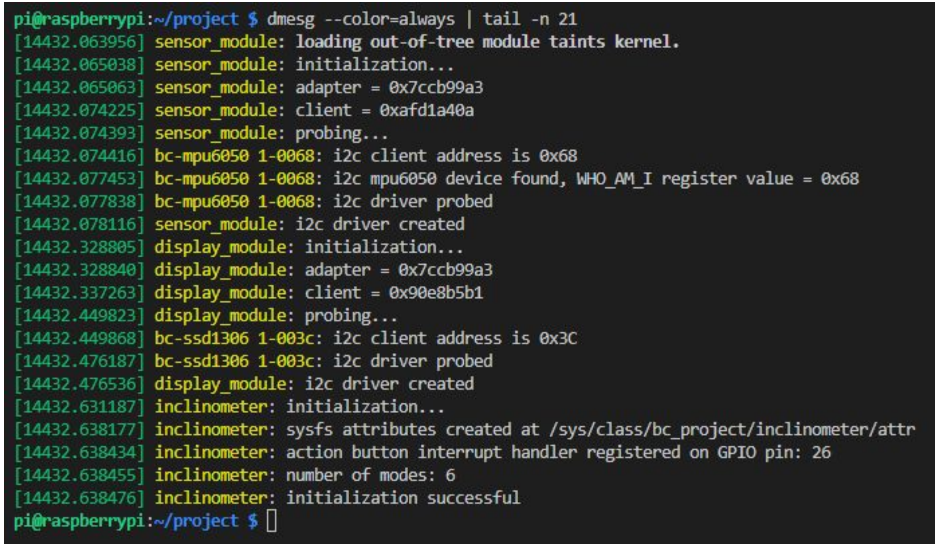
There are also several scripts for safely inserting modules during development process. These scripts remove already inserted modules and then insert the modules in the correct order.

To directly install modules into the system, you can use the **`make install`** command.

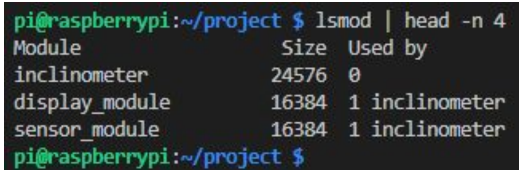


```
user@userver:~/dev/gl_kernel_Basecamp_Project/src$ make install
make -C /home/user/pi/linux M=/home/user/dev/gl_kernel_Basecamp_Project/src INSTALL_MOD_PATH=/home/user/pi/lib_modules modules_install
make[1]: Entering directory '/home/user/pi/linux'
  INSTALL /home/user/pi/lib_modules/lib/modules/5.15.91+/extra/display/display_module.ko
  XZ      /home/user/pi/lib_modules/lib/modules/5.15.91+/extra/display/display_module.ko.xz
  INSTALL /home/user/pi/lib_modules/lib/modules/5.15.91+/extra/inclinometer.ko
  XZ      /home/user/pi/lib_modules/lib/modules/5.15.91+/extra/inclinometer.ko.xz
  INSTALL /home/user/pi/lib_modules/lib/modules/5.15.91+/extra/sensor/sensor_module.ko
  XZ      /home/user/pi/lib_modules/lib/modules/5.15.91+/extra/sensor/sensor_module.ko.xz
  DEPMOD  /home/user/pi/lib_modules/lib/modules/5.15.91+
make[1]: Leaving directory '/home/user/pi/linux'
```

Inserting Modules



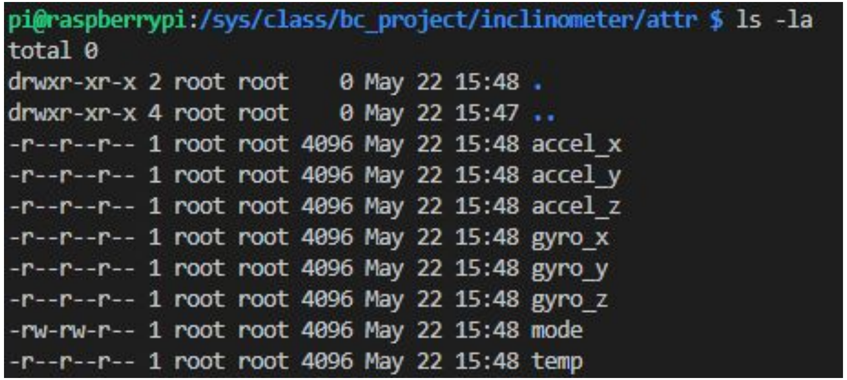
```
pi@raspberrypi:~/project $ dmesg --color=always | tail -n 21
[14432.063956] sensor_module: loading out-of-tree module taints kernel.
[14432.065038] sensor_module: initialization...
[14432.065063] sensor_module: adapter = 0x7ccb99a3
[14432.074225] sensor_module: client = 0xafd1a40a
[14432.074393] sensor_module: probing...
[14432.074416] bc-mpu6050 1-0068: i2c client address is 0x68
[14432.077453] bc-mpu6050 1-0068: i2c mpu6050 device found, WHO_AM_I register value = 0x68
[14432.077838] bc-mpu6050 1-0068: i2c driver probed
[14432.078116] sensor_module: i2c driver created
[14432.328805] display_module: initialization...
[14432.328840] display_module: adapter = 0x7ccb99a3
[14432.337263] display_module: client = 0x90e8b5b1
[14432.449823] display_module: probing...
[14432.449868] bc-ssd1306 1-003c: i2c client address is 0x3C
[14432.476187] bc-ssd1306 1-003c: i2c driver probed
[14432.476536] display_module: i2c driver created
[14432.631187] inclinometer: initialization...
[14432.638177] inclinometer: sysfs attributes created at /sys/class/bc_project/inclinometer/attr
[14432.638434] inclinometer: action button interrupt handler registered on GPIO pin: 26
[14432.638455] inclinometer: number of modes: 6
[14432.638476] inclinometer: initialization successful
pi@raspberrypi:~/project $
```



```
pi@raspberrypi:~/project $ lsmod | head -n 4
Module                  Size  Used by
inclinometer           24576  0
display_module         16384  1 inclinometer
sensor_module          16384  1 inclinometer
pi@raspberrypi:~/project $
```

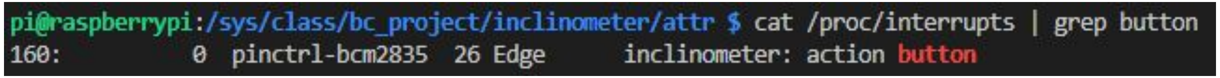
Checking Interfaces

Sysfs



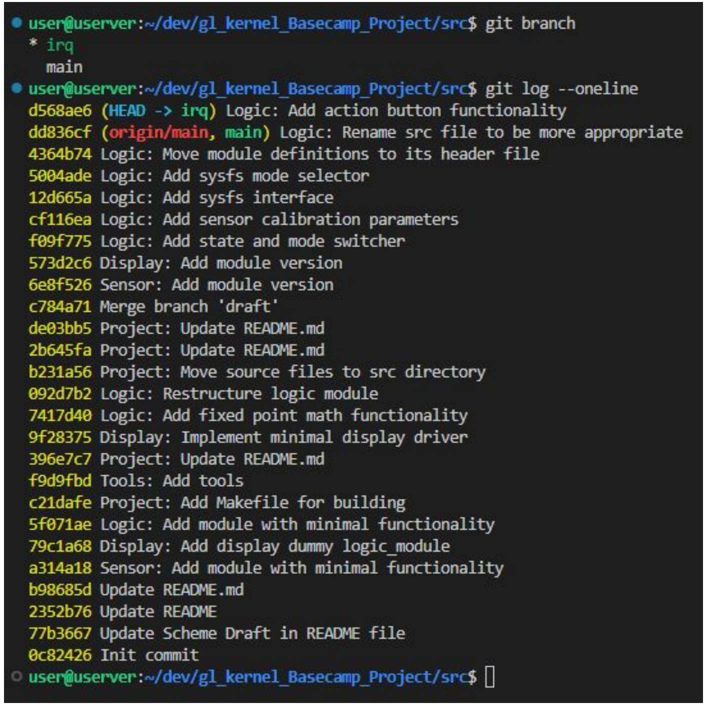
```
pi@raspberrypi:/sys/class/bc_project/inclinometer/attr $ ls -la
total 0
drwxr-xr-x 2 root root    0 May 22 15:48 .
drwxr-xr-x 4 root root    0 May 22 15:47 ..
-r--r--r-- 1 root root 4096 May 22 15:48 accel_x
-r--r--r-- 1 root root 4096 May 22 15:48 accel_y
-r--r--r-- 1 root root 4096 May 22 15:48 accel_z
-r--r--r-- 1 root root 4096 May 22 15:48 gyro_x
-r--r--r-- 1 root root 4096 May 22 15:48 gyro_y
-r--r--r-- 1 root root 4096 May 22 15:48 gyro_z
-rw-rw-r-- 1 root root 4096 May 22 15:48 mode
-r--r--r-- 1 root root 4096 May 22 15:48 temp
```

Interrupts



```
pi@raspberrypi:/sys/class/bc_project/inclinometer/attr $ cat /proc/interrupts | grep button
160:          0  pinctrl-bcm2835  26 Edge      inclinometer: action button
```

Work with the repository

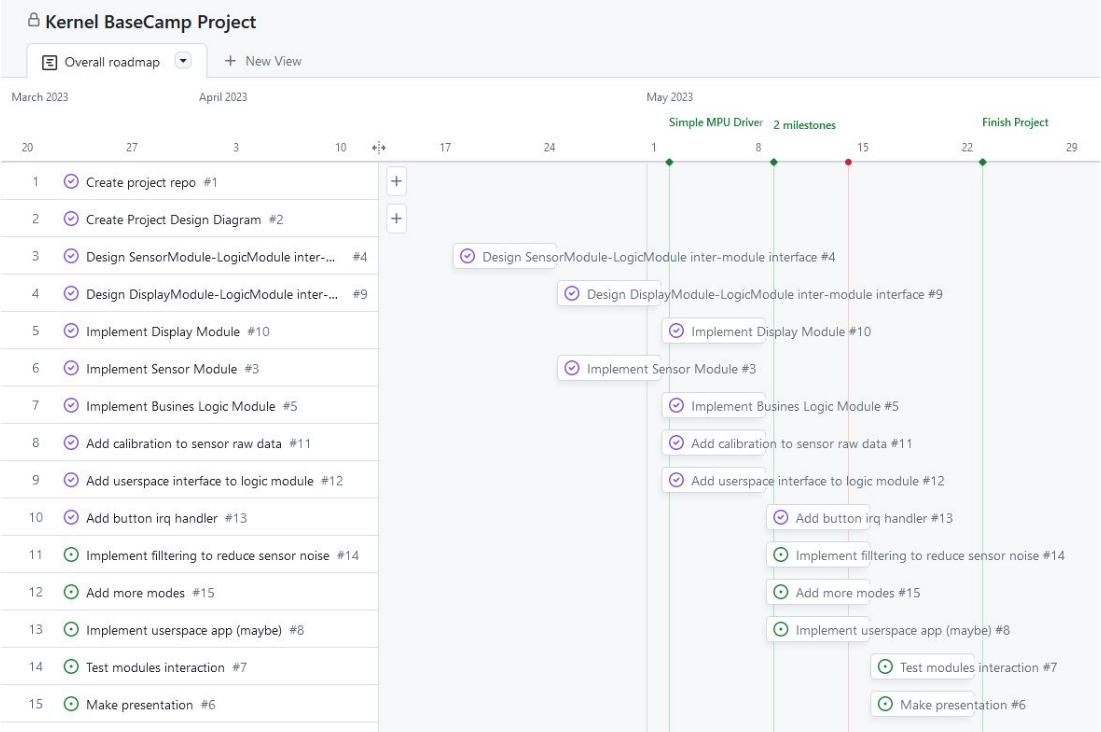


```
user@userver:~/dev/gl_kernel_Basecamp_Project/src$ git branch
* irq
  main
user@userver:~/dev/gl_kernel_Basecamp_Project/src$ git log --oneline
d568ae6 (HEAD -> irq) Logic: Add action button functionality
dd836cf (origin/main, main) Logic: Rename src file to be more appropriate
4364b74 Logic: Move module definitions to its header file
5004ade Logic: Add sysfs mode selector
12d665a Logic: Add sysfs interface
cf116ea Logic: Add sensor calibration parameters
f09f775 Logic: Add state and mode switcher
573d2c6 Display: Add module version
6e8f526 Sensor: Add module version
c784a71 Merge branch 'draft'
de03bb5 Project: Update README.md
2b645fa Project: Update README.md
b231a56 Project: Move source files to src directory
092d7b2 Logic: Restructure logic module
7417d40 Logic: Add fixed point math functionality
9f28375 Display: Implement minimal display driver
396e7c7 Project: Update README.md
f9d9fbd Tools: Add tools
c21dafe Project: Add Makefile for building
5f071ae Logic: Add module with minimal functionality
79c1a68 Display: Add display dummy logic_module
a314a18 Sensor: Add module with minimal functionality
b98685d Update README.md
2352b76 Update README
77b3667 Update Scheme Draft in README file
0c82426 Init commit
user@userver:~/dev/gl_kernel_Basecamp_Project/src$
```

Typical project workflow

1. Branch issues
2. Work on issues
3. Commit changes
4. Merge
5. Push
6. Repeat

Project Roadmap



1	✓	Create project repo #1
2	✓	Create Project Design Diagram #2
3	✓	Design SensorModule-LogicModule inter-... #4
4	✓	Design DisplayModule-LogicModule inter-... #9
5	✓	Implement Display Module #10
6	✓	Implement Sensor Module #3
7	✓	Implement Busines Logic Module #5
8	✓	Add calibration to sensor raw data #11
9	✓	Add userspace interface to logic module #12
10	✓	Add button irq handler #13
11	○	Implement filltering to reduce sensor noise #14
12	○	Add more modes #15
13	○	Implement userspace app (maybe) #8
14	○	Test modules interaction #7
15	○	Make presentation #6

Design SensorModule-LogicModule inter-module interface #4

Design DisplayModule-LogicModule inter-module interface #9

Implement Display Module #10

Implement Sensor Module #3

Implement Busines Logic Module #5

Add calibration to sensor raw data #11

Add userspace interface to logic module #12

Add button irq handler #13

Implement filltering to reduce sensor noise #14

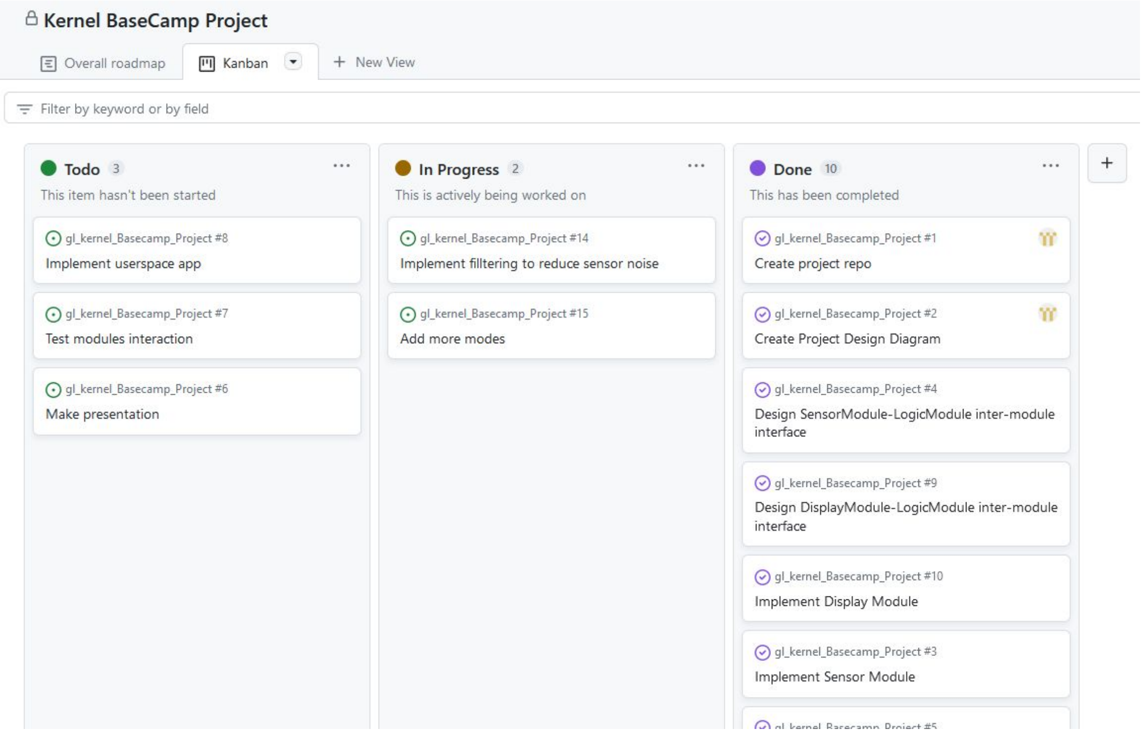
Add more modes #15

Implement userspace app (maybe) #8

Test modules interaction #7

Make presentation #6

Project Kanban Board



Kernel BaseCamp Project

Todo 3

This item hasn't been started

- gl_kernel_Basecamp_Project #8
  Implement userspace app
- gl_kernel_Basecamp_Project #7
  Test modules interaction
- gl_kernel_Basecamp_Project #6
  Make presentation

In Progress 2

This is actively being worked on

- gl_kernel_Basecamp_Project #14
  Implement filltering to reduce sensor noise
- gl_kernel_Basecamp_Project #15
  Add more modes

Done 10

This has been completed

- gl_kernel_Basecamp_Project #1
  Create project repo
- gl_kernel_Basecamp_Project #2
  Create Project Design Diagram
- gl_kernel_Basecamp_Project #4
  Design SensorModule-LogicModule inter-module interface
- gl_kernel_Basecamp_Project #9
  Design DisplayModule-LogicModule inter-module interface
- gl_kernel_Basecamp_Project #10
  Implement Display Module
- gl_kernel_Basecamp_Project #3
  Implement Sensor Module
- gl_kernel_Basecamp_Project #5

Thoughts

- Maybe it's better to implement asynchronous independent threads for polling the sensor and working with the display
- One might use device tree to describe hardware configuration

GitHub

Project repository

https://github.com/deesync/gl_kernel_Basecamp_Project

Datasheet references

MPU6050

https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Register-Map1.pdf

https://product.tdk.com/system/files/dam/doc/product/sensor/mortion-inertial/imu/data_sheet/mpu-6000-datasheet1.pdf

SSD1306

https://cdn-shop.adafruit.com/datasheets/SSD1306.pdf

Thank you!