PROJECT ARCHITECTURE REVIEW

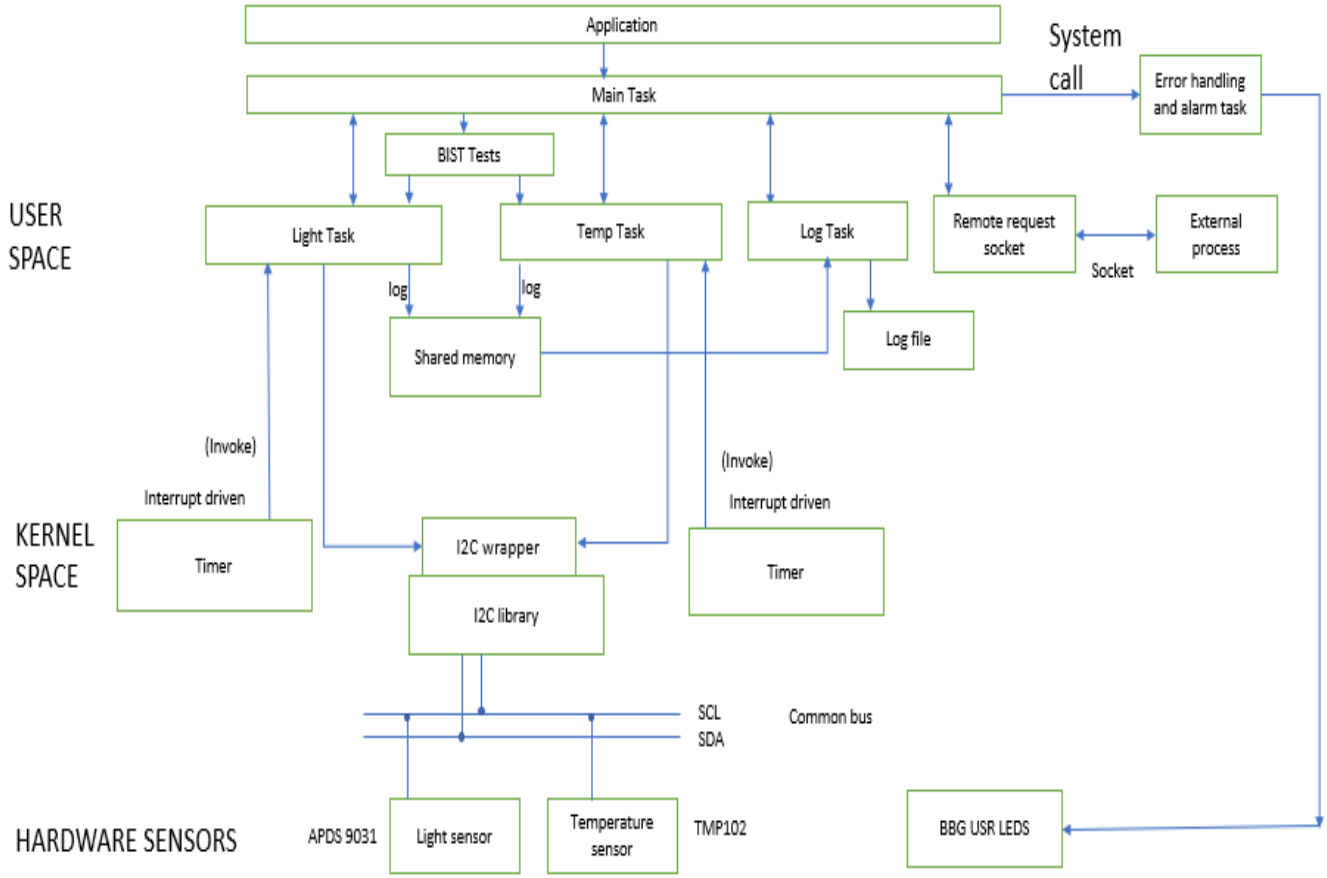
SMART REFRIGERATOR

-BY MONISH NENE

AND

SANIKA DONGRE

System and Software Architecture Diagram:



Documentation:

The project involves interfacing two off-board sensors – light and temperature sensors that are connected to the same I2C bus. The smart refrigerator will give alert indications using the two sensors – for example

– internal temperature is rising, or the fridge door is left open etc. The output data of both the sensors will be logged to a same file.

Tasks and their Responsibilities:

1) Main Task: main() (project1.elf);
    - Create child threads/processes and acts as a parent thread/process.
    - The child threads/processes are:
        - Light sensor task (light.elf)
        - Temperature task (temperature.elf)
        - Synchronized logger task (data_logger.elf)
        - Socket request task (socket_handler())
        - Error handling and Alarm task (error_handler.elf)
    - Responsibilities:
        - Check that tasks are alive and running at regular intervals using request response Mechanism (Heartbeat).
        - Start child processes according to interrupts.
        - Kill child processes after work is done.

2) Light sensor task:
    - Communicate with I2C light sensor using I2C library.
        i. Write to the command register
        ii. Read/Write the Control Register
        iii. Read/Write the Timing register
            1. Set the Integration time
            2. Set Gain
        iv. Enable and disable the Interrupt Control Register
        v. Read the Identification Register.
        vi. Read/Write to the Interrupt threshold registers
        vii. Read sensor LUX data using the ADC registers (Data0 and Data1).
    - Figure out whether the door is open or closed based on the measured LUX value.
    - Save the data to shared memory.
    - Trigger an interrupt to start logging.
    - Trigger an interrupt to turn on LED if the door is open.
    - Turn the light sensor on and off to save power in between the interrupts.

3) Temperature task:
    - Communicate with I2C temperature sensor using I2C library.
        i. Write the Pointer Register
        ii. Read Sensor Tlow register
        iii. Read Sensor Thigh register
        iv. Read Sensor Temperature data register
        v. Read/Write the Configuration Register
            1. Configure the sensor to go in and out of shutdown mode
            2. Configure the sensor resolution
            3. Read the Fault bits

4. Set/Read the EM operation
5. Set/Read Conversion Rate

- Save the data to shared memory.
- Trigger an interrupt to start logging.
- Trigger an interrupt to turn on LED if the measured temperature is above a certain threshold.
- Convert the temperature measured into Kelvin, Celsius and Fahrenheit.
- Turn the temperature sensor on and off to save power in between the interrupts.

4) Synchronized logger task:
    - Log the data that is stored in the shared memory into the log file.
    - The following format will be followed for logging the data:
        i. Timestamp
        ii. Log Level
        iii. Logger Source ID
        iv. Log Message

5) Socket request task:
    - Second process will be used.
    - Handle a request from a remote socket.
    - Authentication.
    - Send Data to the remote socket according to the Request.
        i. System Status.
        ii. Active LEDS
        iii. Latest Temperature Reading.
        iv. Latest Door Status.
    - Modify any of the running tasks according to command send from remote socket.

6) Error handling and Alarm task:
    - Respond to an interrupt generated by Light or Temperature sensor.
    - Control the LEDs on BBG using GPIO control library.

Testing:

- BIST test:
    - These Built-in-self-tests will be executed when the system starts.
    - Communicate with the Temperature sensor and check if it is working.
    - Communicate with the Light sensor and check if it is working.
    - Ensure that threads are running.
    - Log messages will be sent to the logger task when individual BIST tests have finished along with an indicator if the hardware is connected and in working order.
    - If something does not startup correctly, an error message will be logged and both LEDs will start blinking.
- Unit test:
    - These Tests will be performed on individual codes to check functionality before integration.

- Inter thread and Inter process Communication.
- Data Logger.
- Temperature Sensor Measurement and control.
- Light Sensor Measurement and control.
- Alarm conditions and LEDs.

API functions:

I2C:

- i2c_write(): It will take the address of the register as the parameter and write to the register of the temperature and light sensor respectively by using an I2C wrapper function.
- i2c_read(): It will take the address of the register as a parameter and read the respective sensor by using an I2C wrapper function.

Temperature task:

- get_temperature(): get the latest temperature value and convert into required temperature units.

Light task:

- get_luminosity(): get the latest luminosity/lux value

Error handling task:

- led_write(): control the leds on the BBG

Remote request task:

- socket_handler() : handle request from remote

Log synchronization task:

- log_instance(): data will be logged to the log file.

Main task:

- get_heartbeat(): To check if the tasks/processes are alive and return error if any process/thread has stopped.

Shared memory (between Light, Temperature and synchronized logger task)

- put_shared_memory(): To put the temperature and light data into shared memory.
- get_shared_memory(): To get the data (temperature and light) from shared memory.

User and Kernel Space:

- The main task and BIST tests are included in the USER SPACE.

- The I2C wrapper functions are used for provide access to the I2C bus exclusively for two different sensor tasks are included in the KERNEL SPACE.

The timer Kernel module will be used. The error handling and alarm task will use system call.