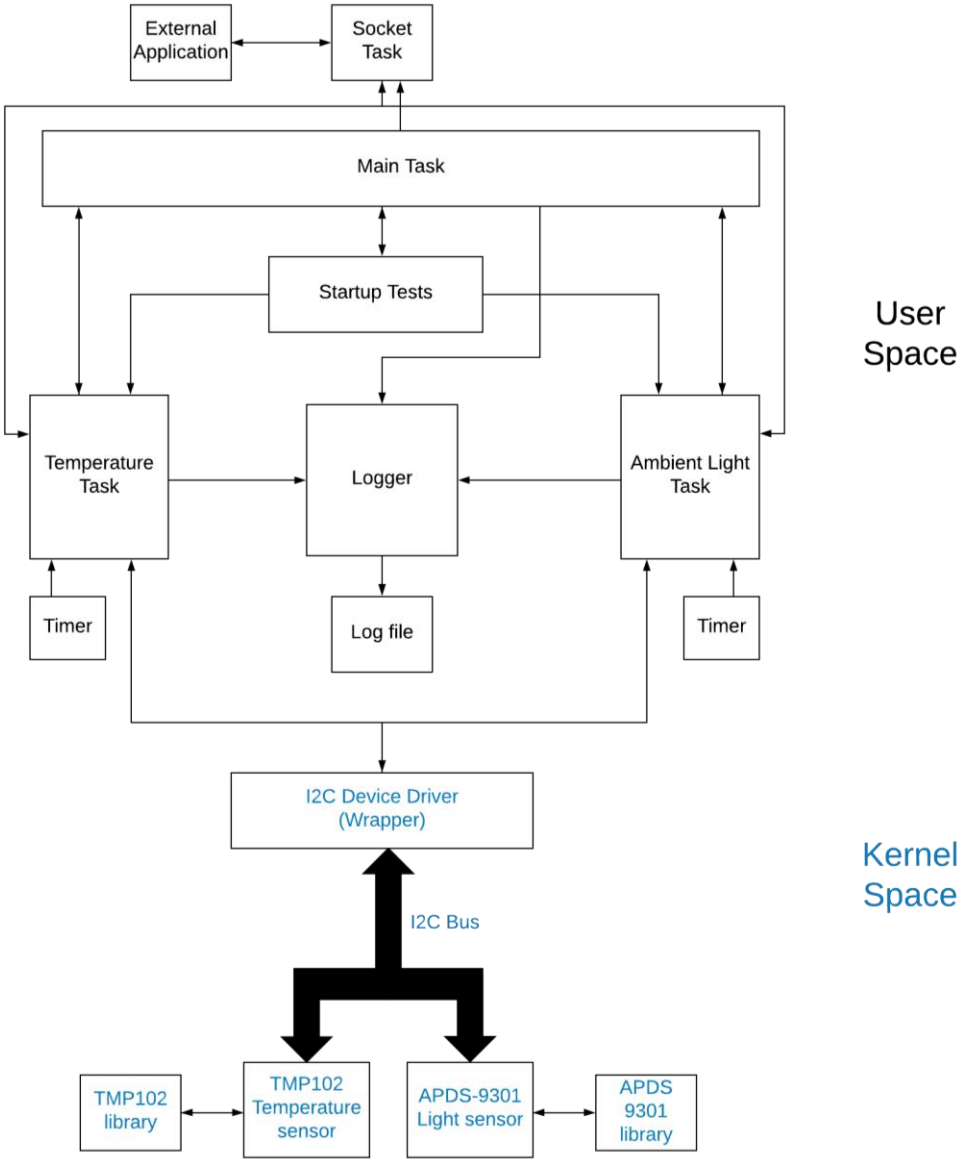
Advanced Practical Embedded Software Design

Project 1 Software Architecture Diagram

Group Members:

1. Nikhil Divekar
2. Vipraja Patil

Software Architecture Diagram



Tasks and their Functionalities:

Main Task:

The main task is the parent task which is responsible for creating all the child tasks which include temperature sensor task, light sensor task, synchronized logger task, socket request task. Error task is included in the main task itself. The other responsibility of the main task is to ensure all the tasks are alive and running after specific interval of time. To ensure this, the main function will use 'heartbeat' functionality which is request-response mechanism. The main task will close the application gracefully if user wishes to stop the task. Also, it will report any error to the user and indicate an error condition using usr LEDs.

Temperature Sensor Task:

The temperature sensor task is the user space task which will interact with temperature sensor TMP 102. This task is responsible to communicate with the temperature sensor through I2C wrapper function which is present in the kernel space. This task is responsible for reading, writing or configuring some of the registers of the temperature sensor and to collect the current value of temperature from the sensor. The temperature recorded should support any of the 3 temperature formats i.e Celsius, Fahrenheit or Kelvin.

Light Sensor Task:

The light sensor task is the user space task which will interact with light sensor APDS-9301. This task is responsible to communicate with the light sensor through I2C wrapper function which is present in the kernel space. This task is responsible for reading, writing or configuring some of the registers of the light sensor and to collect the current value of luminosity or lux value from the sensor. This task is also responsible to decide whether it is daytime or nighttime based on the data collected by sensor and to log if there is any abrupt change.

Logger Task:

This task accepts the data log from various sources on the regular intervals. This interface will need to have protection from multiple log source. The logger writes the data to the log file, the file name and path of which should be kept configurable, which means it can be modified during runtime. Every logger file must contain Timestamp, Log level, Logger source ID, Log message. Logs should be able to print character strings as well as integer/float data. Logger should efficiently get the data from the two sensors exclusively. Logger task will have a message queue which will collect data from both sensors.

Socket Task:

This task works as a bridge between the external requests and the sensor data. The external application can request data from the sensors by making the API calls to the sensors through this socket connection. This task can be used to test the functionality of our application in its live state by requesting/injecting API command into the internal messaging structure.

Test Conditions:

Startup Tests:

These tests must be run before the application runs in the steady state operation to validate that hardware and software is functioning properly. Some of these startup tests are as below:

- Communicate with temperature sensor to confirm I2C and hardware functioning properly.
- Communicate with light sensor to confirm I2C and hardware functioning properly.
- Communicate with main task to ensure that all threads have started and running.

Unit Tests:

These tests are performed to check whether the data received from tasks and APIs are same as the expected value and to check whether the data is logged into the logger properly. Some of the unit tests are as follows:

- Inter-thread communication.
- Logger.
- Temperature sensor conversions with mocked data.
- Light sensor conversions with mocked data.

Important Functions/Message APIS:

Some of the important functions or message APIs that will be included are listed below

Temperature task:

1) temp_sensor_setup():

   This API will perform all the setup configurations required to initialize this sensor.

2) get_temp(int format):

   This API is used to get the current temperature reading for TMP102 sensor. This function takes some int or enum as a parameter which will decide the format in which temperature is recorded i.e Celsius, Fahrenheit, Kelvin, etc

Light task:

1) light_sensor_setup():

   This API will perform all the setup configurations required to initialize this sensor.

2) get_luminosity():

   This API is used to get the current lux value from sensor.

3) getDayorNight(int luxValue):

   This API is used to determine whether it's daytime or night time based on the current lux value recorded by the get_luminosity API. It takes luxValue as a parameter which is current lux value.

I2C Wrapper Function:

1) i2c_read():

This API is used to read the sensor data or register using I2C wrapper function. This function will have register address as a parameter from which we are supposed to read. It may also have some other parameter depending on application.

2) i2c_write():

This API is used to write to the registers of the corresponding sensor through I2C wrapper function. This function will have register address as a parameter to which we are supposed to write. It may also have some other parameter depending on application.

Logger:

1) get_log():

This function will log the data to some log file. The filename and path will be kept configurable and can be modified at the run time.

Startup tests:

1) int check_setup():

This API will run the startup tests and return success or error if any. The error enum is also returned to indicate the error condition

Main Task:

1) check_heartbeat():

This API will be used by main task to ensure whether all the child tasks are up and running. It will return failure if any task is stopped due to any circumstances.

User Space Vs Kernel Space:

The main task and all the children tasks are included in the user space. Startup tests is also included in the user space which will check up all the functionality at the startup for the function. The I2C wrapper function along with the temperature sensor and light sensor library functions are include in the kernel space. Temperature task and light task will communicate to the corresponding sensors through I2C wrapper functions which will read from or write to the registers of the respective sensors. I2C wrapper function will ensure the exclusive access to the I2C bus with two independent tasks trying to read or write to the two different sensors.