Indoor Exhaust System

By

Sanika Dongre

and

Monish Nene

Github repo link: https://github.com/monishnene/AESD_2

Project Description:

Our project is a simulation of an indoor exhaust system. We will use exhaust fans to remove harmful gases from the chamber and alert with a buzzer if the gases and/or temperature goes above a certain threshold.

Remote Node: Tiva TM4C1294XL

Control Node: Beagle bone green

The indoor exhaust system will be designed in a box that comprises of

Sensors:

1) Adafruit Temperature Si7021 Temperature and Humidity Sensor (I2C interface).
2) Adafruit Indoor Air Quality (IAQ) sensor evaluation board that is used for detecting various gases in ppm level.

Control outputs:

1) A set of four exhaust fans
2) A buzzer

The sensors will be placed in a box and we are planning to check it with an incense stick or a lighter. The count of exhaust fans on and their speed can be varied according to the feedback from the sensor.

Requirements:

1) State machine on the remote node to alert in case of emergency or failure of sensors, switch the fans on and off according to feedback (closed loop state machine).
2) PID control (PWM) to control the speed of the exhaust fans.
3) LEDs on the remote node to indicate the current mode of operation.

Use cases:

1) Default: The value of temperature and the gas sensor is within the normal range and the exhaust fan is off
2) Smoke detected: Turn on the fans according to the level
3) High Temperature detected: Turn exhaust fans on based on the level
4) Failure of smoke sensor: Run the fans on maximum speed and send alert with the buzzer
5) Failure of Temperature sensor: Run the fans on maximum speed and send alert with the buzzer

System Configuration:

1) Tiva board: Remote Node
2) Beagle bone green: Control Node

3) The alerting mechanisms by the beagle bone green to indicate:

We will be using two buzzers – One buzzer for TIVA board and One for Beagle bone green to indicate the operational status of the system. The exhaust fans (actuators) will be connected on the TIVA board.

- Default/Normal: The value of temperature and the gas sensor is within the normal range and the exhaust fan is off

- Degraded: Status LEDs on the beagle bone green will be used for indication of degraded mode
- One of the sensors fail
- One of the fans fail
- Buzzer fails

Failure:

- Smoke detected: Turn on the fans according to the level
- High Temperature detected: Turn exhaust fans on based on the level
- Failure of smoke sensor: Run the fans on maximum speed and send alert with the buzzer
- Failure of Temperature sensor: Run the fans on maximum speed and send alert with the buzzer

Control-Remote Node connectivity:

- UART for connecting the remote node (TIVA Board) to the control node (Beagle bone green)
- A message queue for communicating the control/sensor information for the transmission of the temperature and gas sensor data and logs.

System Functionality:

Automatic startup:

Control Node startup

Remote Node startup

BIST/startup tests: To test the hardware and software is up and functioning. If the BIST fails then the tasks related to the respective sensors which caused the failure won't be turned on.

Remote Node sensing:

State diagram:

1) 0 fan
2) 1 fan and speed control
3) 2 fan and speed control
4) 3 fan and speed control
5) 4 fan and speed control
6) Buzzer

The control algorithm – If the temperature goes above 20 degree Celsius then run the fan at 40% of its full speed and if the temperature goes above 100 degrees – (alert condition): All the fans will run at full speed and the buzzer will be turned on.

Failure and Fault detection Behavior:

Using BIST (startup test) If the sensors or one of the sensor fails then the corresponding task won't be executed.

Testing based on threshold values for temperature and gas: by activating the exhaust fans and buzzers.

If the control node cannot detect the Remote node then the control node will keep trying to get connected until it can get connected to the remote node.

If the sensor fails or the temperature and gas sensor values are beyond a threshold then the fans will be turned on accordingly as a feedback mechanism.

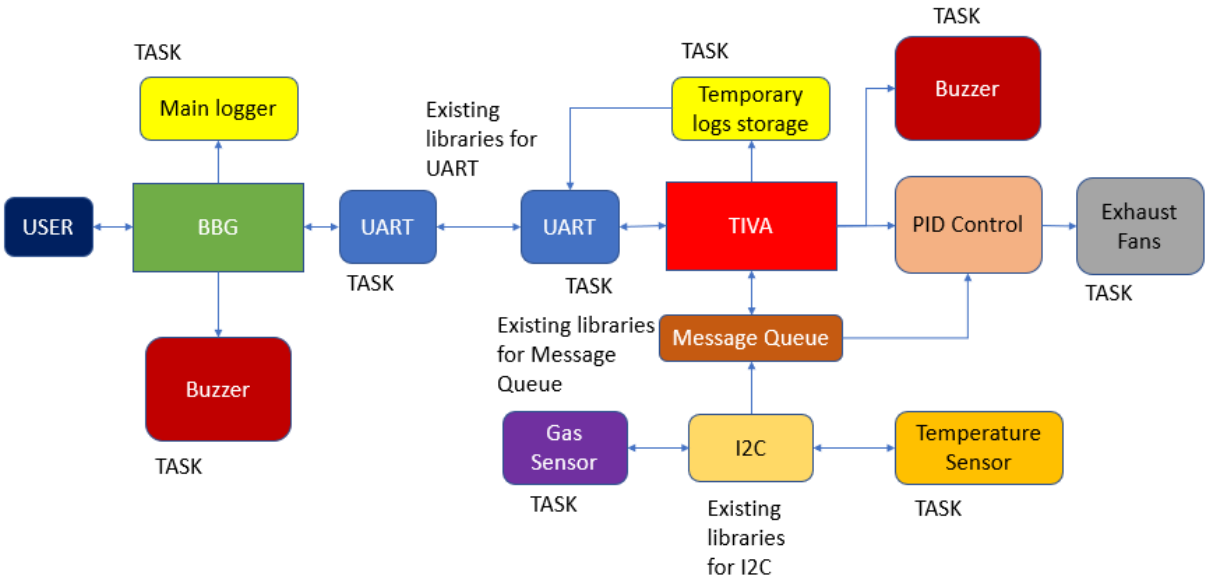
If the control node and the remote node connection fails then the buzzer on the control node side will turn on.

Logging:

The logging will be done on a logfile in the control node side. The data will be collected by the remote node and sent to the Beagle bone via UART. If the connection between the control node and remote node fails, then the real time data from the sensors will be saved on a logfile on remote node side and once the connection is retrieved then the temporary file will be sent to the beagle bone.

Architecture Description:

Software Diagram:



States:

1) 0 fan
2) 1 fan and speed control
3) 2 fan and speed control
4) 3 fan and speed control
5) 4 fan and speed control
6) Buzzer

Tasks for Remote Node:

1) Buzzer task: Buzzer task is used for turning on the buzzer on the TIVA board
2) Logger task: For collecting data and sending it over UART and back up data in case of connection failure
3) UART task: uart send and receive options
4) Temperature sensor task: To get the temperature values from the sensor using I2C interface
5) Gas sensor task: To get the gas readings from the sensor using I2C interface.

6) Exhaust fan control task: To activate the exhaust fans in case of fault/failure conditions

Tasks for Control Node:

1) Buzzer task: Buzzer task is used for turning on the buzzer on the beagle bone green
2) Logger task: To save the data in a log file
3) UART task: To send/receive data over UART

API Description:

Remote-Node:

1) get_temperature() : Read the data from I2C sensor
2) get_gas(): Read the data from I2C sensor
3) log_init(): Initialization for log file
4) log_data(): logging data
5) logger_creator(): For creating logs with string and log level as the parameters
6) pid_control(): control the speed of the exhaust fans
7) uart_send(): to send the data over UART
8) uart_receive(): To receive data over UART
9) buzzer_task(): create a varying PWM to control the buzzer
10) bist_setup(): It checks the if the BIST setup is done properly (I2C connection up and running, hardware and software working).

Control-Node:

Pthreads for uart_send, uart_receive, logger_thread, buzzer_thread

1) uart_send(): to send the data over UART
2) uart_receive(): To receive data over UART
3) buzzer_task(): create a varying PWM to control the buzzer.
4) log_init(): Initialization for log file
5) log_data(): logging data
6) logger_creator(): For creating logs with string and log level as the parameters
7) logfile_setup(): create the logfile with command line argument.

Project Plan:

Using Excel Gantt:

AESD_PROJECT_2