Principles of Embedded Software: Project Proposal

HEADLAMP BEAM COLOUR ASSISTANCE

Author: Venkata Janakirama Pradyumna Gudluru

Project Outline:

Headlamp beam colour assistance system changes the colour of the headlamp beam based on the PITCH and ROLL angles of the vehicle. When the vehicle is moving from a higher elevation to a lower level, the low beam is updated with a low wavelength colour. When the vehicle is moving up from a lower elevation point, a higher beam is required, hence for a sample project a higher wavelength colour is replaced.

Similarly, for the case of roll left and right different colours are updated. In the following proposed solution, I will use a KL25Z and the on-board accelerometer(MMA8451Q). This feature of headlamp beam colour assistance can be initialized by command line interface or by pressing a button-switch. Reducing the scale of the device by limiting the parameters to angles or positions, the direction of the vehicle and the colour to be updated, the following conditional table is developed.

Command	Position	Colour
LEFT(Roll Axis)	-THRESHOLD<X<0	BLUE
RIGHT(Roll Axis)	0<X< THRESHOLD	GREEN
UP(Pitch Axis)	0<Y< THRESHOLD	ORANGE
DOWN(Pitch Axis)	- THRESHOLD <Y<0	VIOLET
CRASH	ALL ELSE CONDITIONS	RED

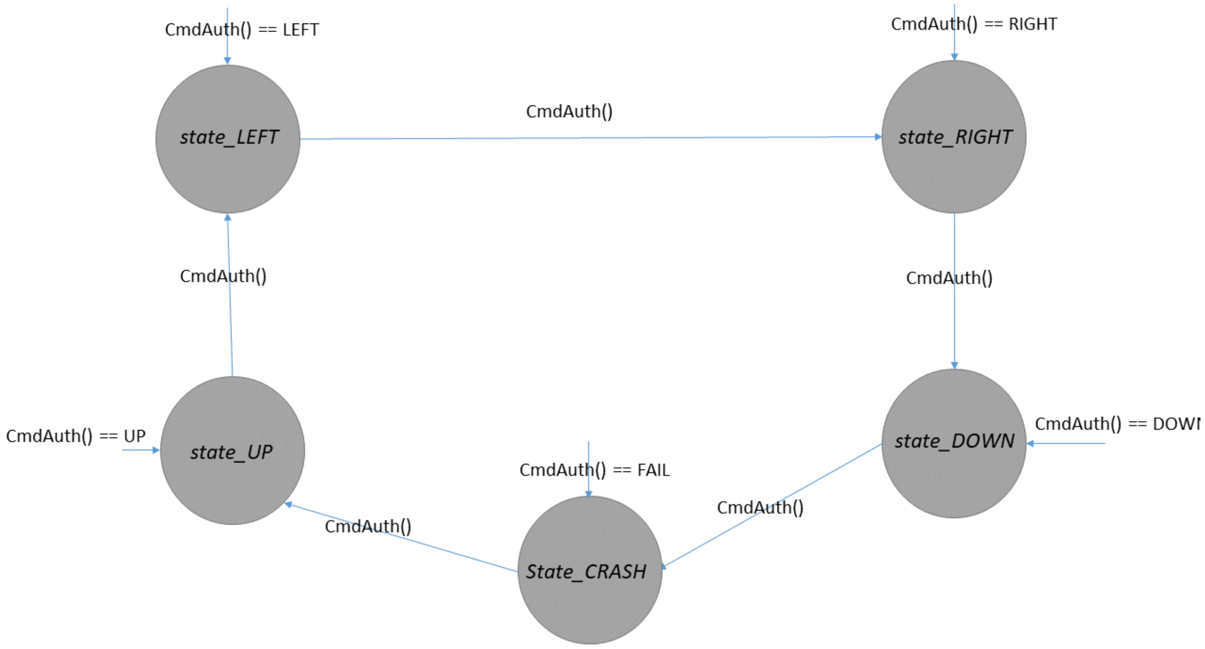
Note: THRESHOLD = 35^0 (based on typical toppling angle of vehicle – considered for ROV’s)[1]

Functionality:

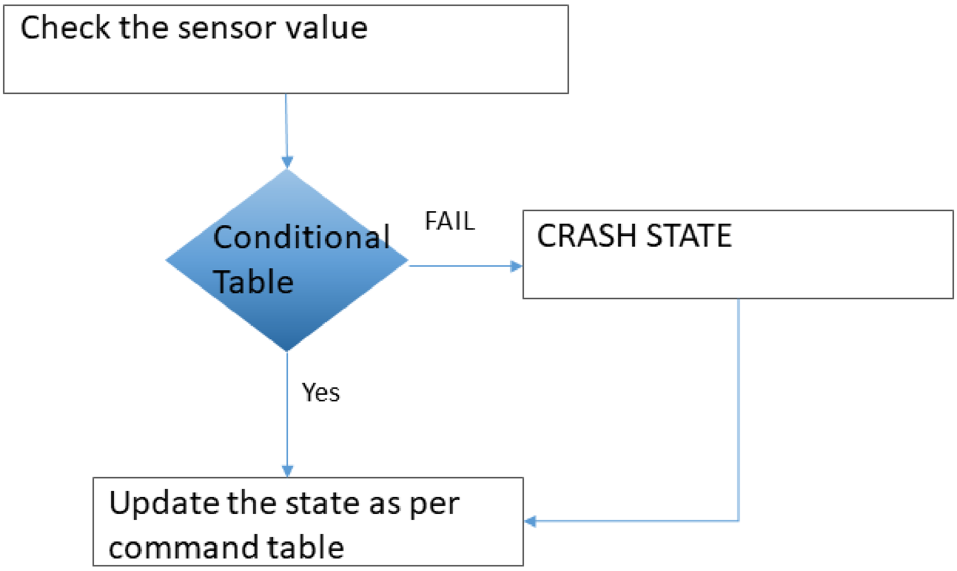
My code will use the I2C communication for the transmission of data from the on-board KL25Z(MMA8451Q). The angle is sensed from the sensor and based on the PITCH and the ROLL thresholds, the LED is switched to different colours. If the pitch value is 0-35^0, the colour is blue. If the pitch value is -35^0 – 0, the colour is green. If the rollover is 0 - 35^0, the colour of the LED is turned orange and if the rollover is -35^0 – 0, the colour is violet. If the pitch or the rollover exceeds the threshold, the vehicle will go to crash state with RED colour blinking.

By sending a command through a command line interface using UART, one can initiate the calibration of the angles indicated by LED with white colour. Commands “calibrate” and “display” are used as input commands for the board to calibrate pitch and roll angles and display the value respectively. For every 3 seconds, the rollover and the pitch values are printed on to the terminal window. This feature can also be initiated using the external switch, when you press the switch, the rollover and pitch are printed on to the terminal, immediately. PWM signalling is used for switching ON and OFF LED’s for the colours to vary and change in brightness of colour.

StateChart:



CmdAuth():



Technologies:

1) UART implementation using circular buffers for communication of terminal to board and vice versa.

2) Circular Buffers for UART data buffering
3) I2C for accelerometer sensor data.
4) GPIO Interrupts for updating the display
5) PWM for LED colour change
6) State Machine for four different directions.
7) Interactive Command processing for calibrate and display

Learnings Required:

1) To understand the calibration and working of MMA8451Q.

2) Multiple interrupt handling and priority

3) I2C communication.

4) Tilt angle measurement according to the MMA8451Q.

References Required:

1) Embedded Systems Fundamentals with ARM Cortex-M based Microcontrollers by Alexander G.Dean
2) Making Embedded systems by Elecia White
3) MMA8451Q datasheet
4) KL25Z manual/ datasheet
5) I will use the source code from the previous assignments by working on all my review points provided to me.

Additional Hardware:

An external button switch is required for the execution of the GPIO external interrupt functionality.

Testing Strategy:

Since the accelerometer has to be tested mechanically, I would be using a mixture of both automated tests and manual tests.

1) The circular buffer is tested automatically for multiple instances from the test cases provided for cbfifo.
2) UART input and output is tested, based on the interactive terminal with pre-assigned inputs to the UART and verified as per the outputs to the command terminal.
3) Checking the UART prints of accelerometer readings manually for verifying the I2C functionality.
4) Checking the PWM by giving arbitrary values into the input and validating the LED output.
5) If the angle from the accelerometer sensor increases the threshold, the LED goes to RED and the State to CRASH.

[1] https://www.cpsc.gov/s3fs-public/pdfs/SEAReportTiltTableResults22ROVsSept2015.pdf