Module – Microcontroller

Car Black Box
Contents

1 Abstract......................................................................................................................................... 1
2 Requirements................................................................................................................................. 2
3 Prerequisites.................................................................................................................................. 4
4 Design............................................................................................................................................. 5
5 Sample Output................................................................................................................................. 10
6 Artifacts........................................................................................................................................... 12
   6.1 Skeleton Code......................................................................................................................... 12
   6.2 References............................................................................................................................... 12
1 Abstract

The concept of Black Box is mainly heard by us in case of Aero-planes. Upon a catastrophe the Black Box is used to analyze the root cause of the issue. However, the purpose of Black Box can go beyond catastrophe analysis. Also known as an event data recorder (EDR) or Accident data recorder (ADR) the Black Box can be installed in some automobiles to record various events. These events which are electronically sensed can often pro-actively detect any issue in the vehicle (ex: Engine fault) or monitor the fleet (ex: Drive over speeding), thereby doing pro-active maintenance of the Automotive vehicle.

By considering today's busy life, every one wants to reach the destination as soon as possible by ignoring the rules and regulations. By implementing a Black Box which would log critical activities on the car and take appropriate actions in case of rash driving. As mentioned above the root cause of the negligence in the driving would be meeting the daily schedule and go off duty, or to earn extra money by illegal trips etc… So by implementing the mentioned solution it would be easy to keep track of how the vehicle is being used, handled and control the efficiency of the vehicle.

The proposed solution is to log all the critical events like the gear shifts with current speed, the engine temperature, fuel consumption per trip, trip distance etc., The system should allow a password based access to the transport managers to view or download the log to PC if required. Here is a video which gives a working idea about this solution.

As an extension of this idea, the data can be exported continuously to a Cloud based server where a centralized monitoring can be done. This means the black box implementation is extendable as a IoT based solution as well. The goal of this project is to implement a Black Box using the given Micro-controller (PIC in our case) and simulate various events by interfacing the Micro Controllers with sensors etc.
2 Requirements

• Default Screen
  ◦ When the system is in Operation Mode, it would act like a dashboard which would show the current time, speed of vehicle and the latest occurred event.

• Login Screen
  ◦ On press of the UP or DOWN (User Keys) keys the system should prompt for password
  ◦ The password would be the combination of 4 presses (User Keys).
  ◦ Each press should be denoted a "***" symbol
  ◦ Every wrong entry would be re prompted for password (Max 3 times for every 15 minutes)
  ◦ Incomplete key press (pause of 3 seconds) would lead to Default Screen

• Main Menu
  ◦ The main menu should contain 2 options
    ▪ View Log
    ▪ Set Time
  ◦ The UP / DOWN keys are used to navigate
  ◦ A long press of UP Key should enter the selected menu
  ◦ A long press of DOWN Key should logout
  ◦ Idle screen for more than 5 secs should logout

• View Log
  ◦ Should display all the events captured with log index starting from 0, like
    "EVENT NUMBER" "EVENT SIGNATURE" "EVENT TIME" "SPEED AT THE EVENT"
  ◦ The UP and DOWN key will be used to scroll the entries
  ◦ Rollover on reaching the max log entries
  ◦ The system should be live (capture events occurred) even while viewing the log
  ◦ A long press of UP Key should take you back to main menu
  ◦ A long press of DOWN Key should logout
• Set Time
  ◦ Should show the current time. The Secs field should blink indicating the field to be changed
  ◦ The UP key should be used to increment the time. Rollover on reaching max
  ◦ The DOWN key will be used to choose the field.
  ◦ A long press of UP Key should take you back to main menu

• Event Capture
  ◦ Required events have to be captured and stored in the memory
  ◦ Every event should have a format as
    “EVENT SIGNATURE” “EVENT TIME” “SPEED AT THE EVENT”
  ◦ The events should be captured real time, no matter which mode you are in
3 Prerequisites

- Embedded C programming
- I2C protocols
4 Design

• Block Diagrams

Fig 4 1: Block Diagram - Level 0

Fig 4 2: Block Diagram - Level 1

Fig 4 3: Block Diagram - Level 2
• Schematic
• Flowchart

Start

0

Check for user key activity

1

Key press detected?

No

Create default screen

Yes

Default screen?

2

Logon screen?

Yes

Display

No

Main menu?

3

View log screen?

Yes

No

Set time screen?

4

5

Init gpio

0

Init keypad

Init clcd

Init timers

Init icc

Init rtc

Init can

Init system defaults

Enable global interrupts

1
Note: Partial flowchart. You may follow the similar approach to implement remaining part of menu
5 Sample Output

Fig 5: Default Screen. Should display the current time, the latest event captured and the current speed.

Fig 5: User Login. The user needs to enter the password to browse through the available menu.

Fig 5: The main menu.

Fig 5: The main menu.
Fig 5: The Log screen. Pressing holding the UP key in the main menu should take you this screen

Fig 6: Time Setting Screen
6 Artifacts

6.1 Skeleton Code

The skeleton code is a very interesting concept used in Emertxe. By looking into the skeleton code, you will get a clear picture into converting the given requirement into a working solution. This will also take care of important aspects like modularity, clean coding practices, re-usability etc.

• TBD

6.2 References

• https://en.wikipedia.org/wiki/Event_data_recorder