EMERTXE TRAINING PROJECT DOCUMENTATION FRAMEWORK

REQUIREMENTS & DESIGN DOCUMENT

Module – Microcontroller

Car Black Box

ΣMERTXE

Contents

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 dash board 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, re prompt 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 option
    - 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

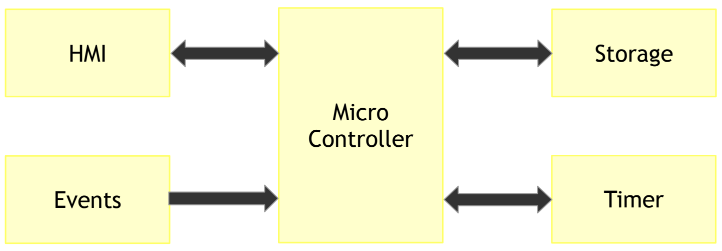


Fig 4 1: Block Diagram - Level 0

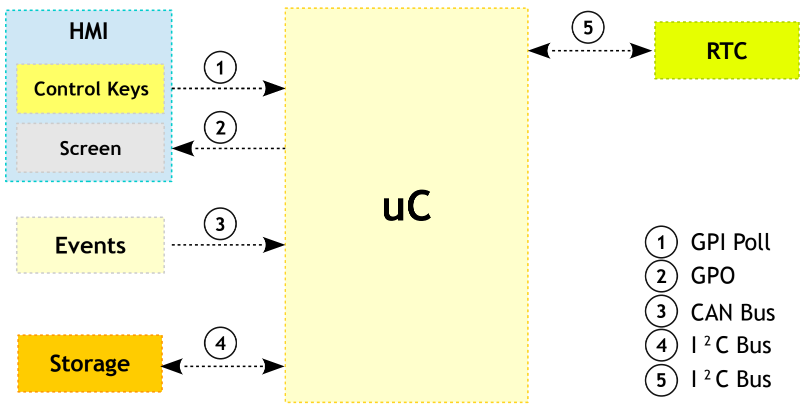


Fig 4 2: Block Diagram - Level 1

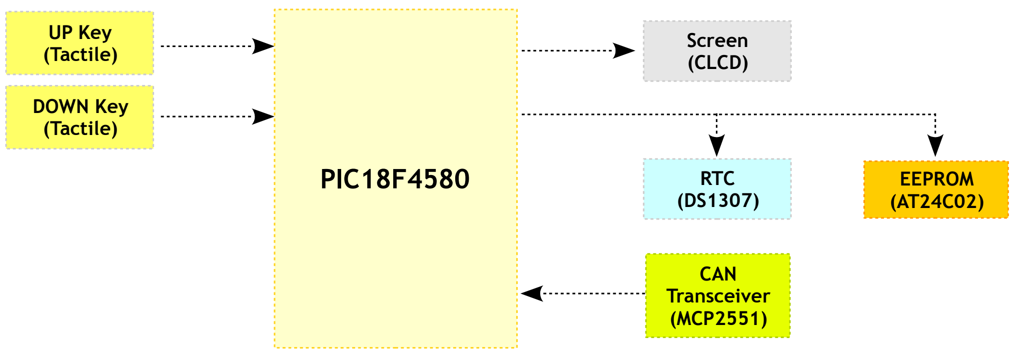
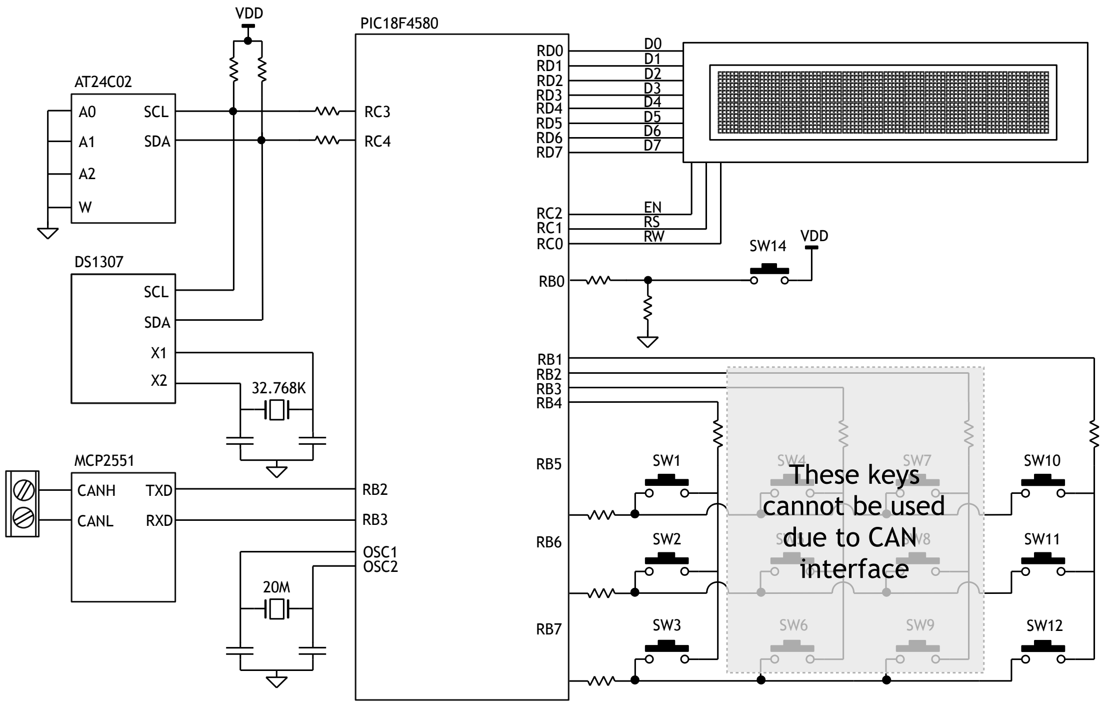
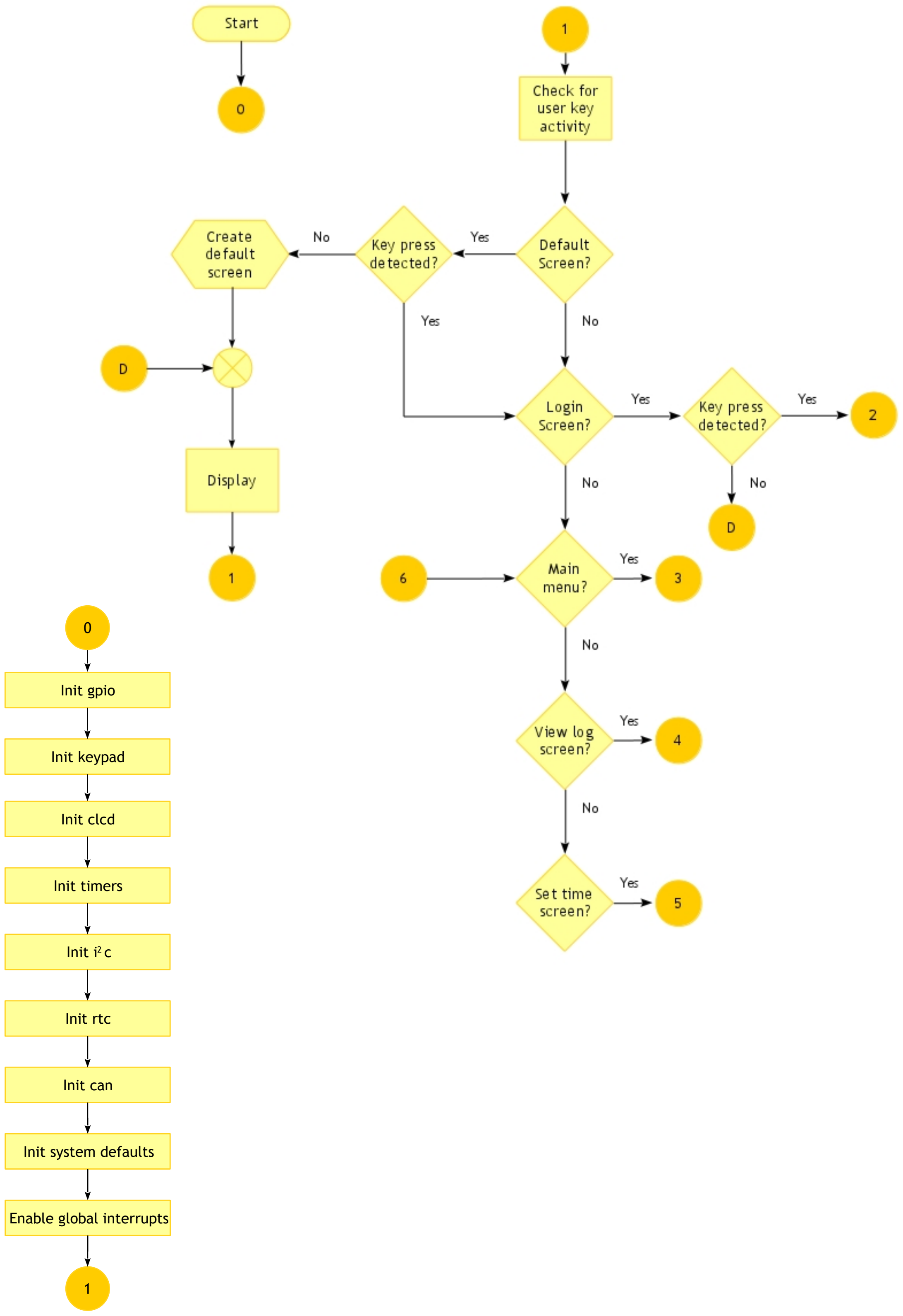


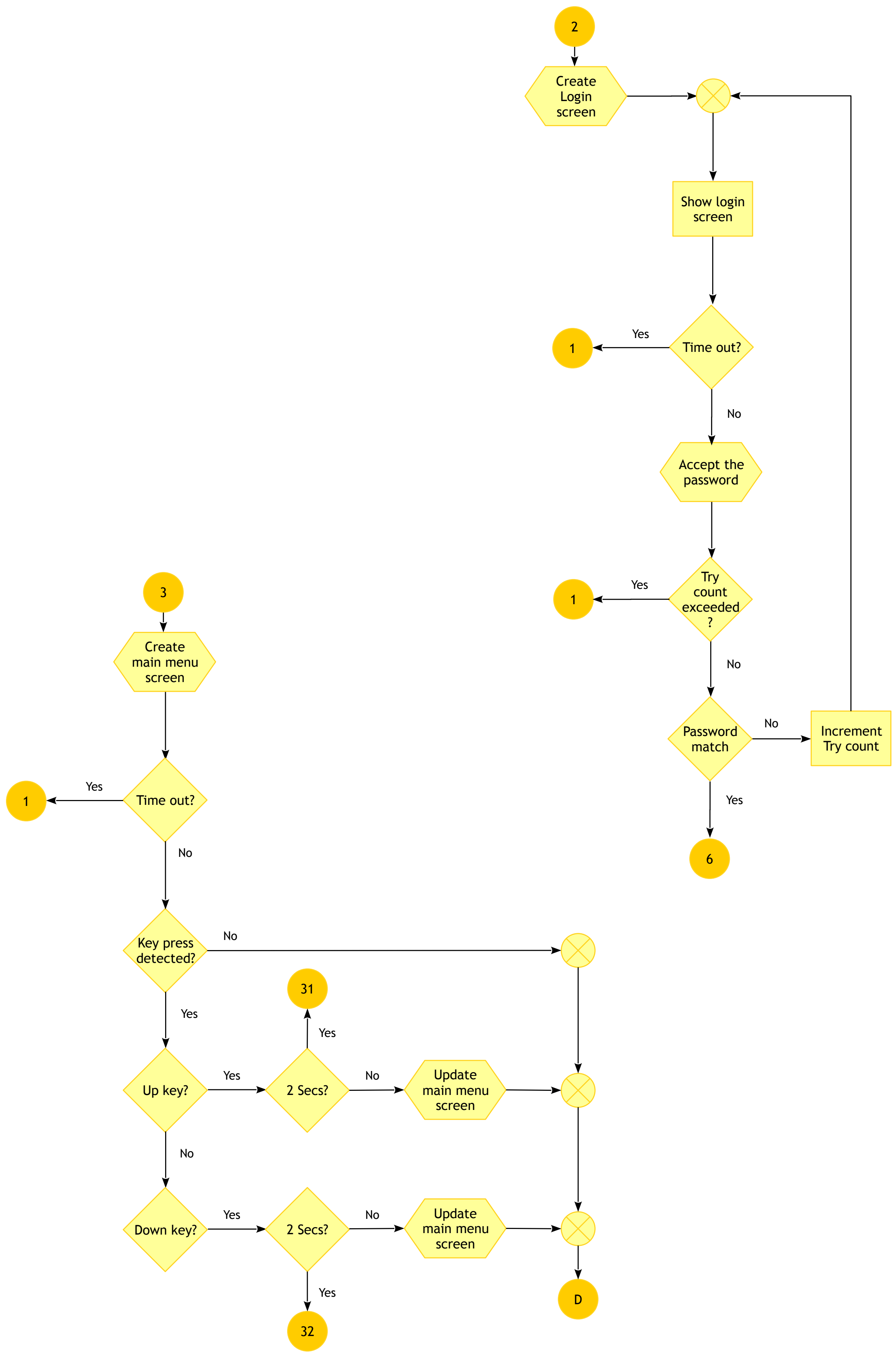
Fig 4 3: Block Diagram - Level 2

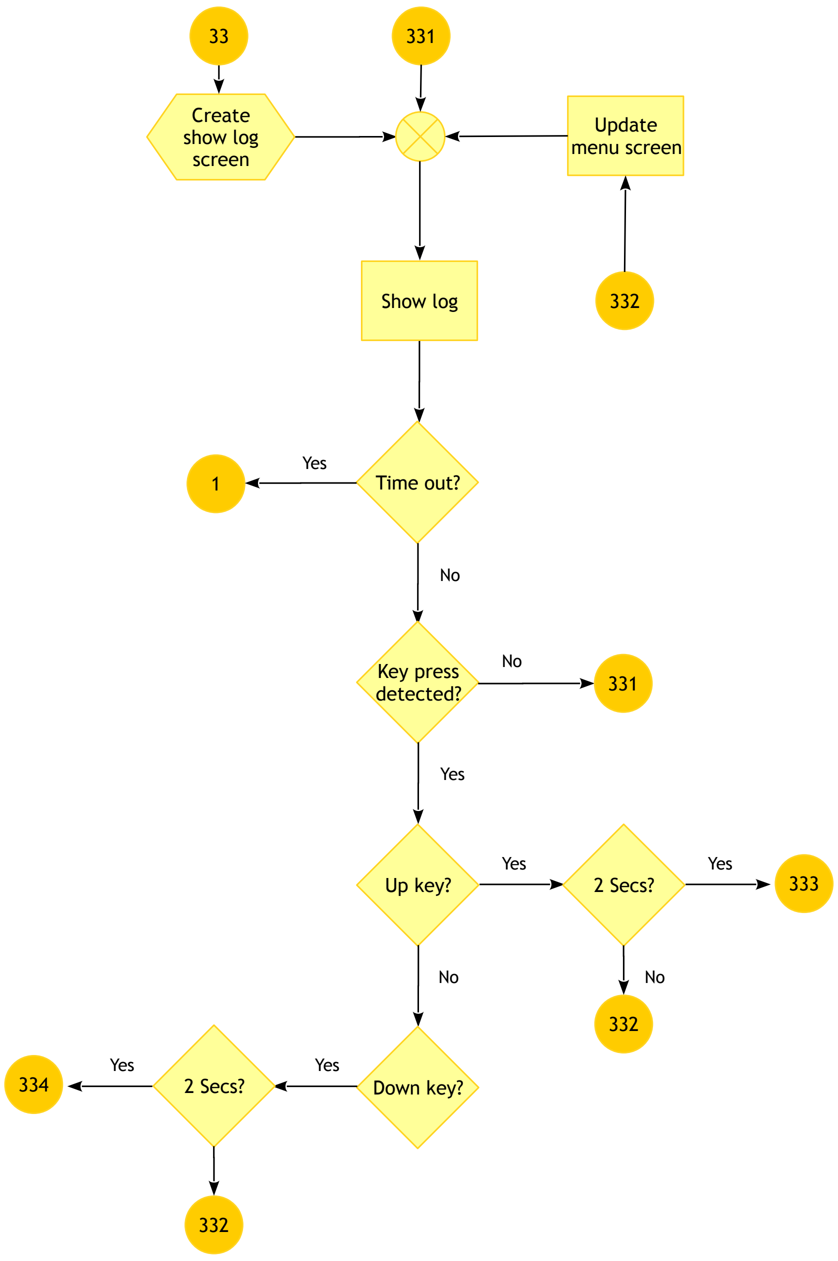
- Schematic


VDD
PIC18F4580
AT24C02
A0
A1
A2
W
SCL
SDA
RC3
RC4
DS1307
SCL
SDA
X1
X2
32.768K
MCP2551
CANH
CANL
TXD
RXD
RB2
RB3
OSC1
OSC2
20M
RD0
RD1
RD2
RD3
RD4
RD5
RD6
RD7
D0
D1
D2
D3
D4
D5
D6
D7
RC2
RC1
RC0
EN
RS
RW
SW14
VDD
RB0
RB1
RB2
RB3
RB4
RB5
RB6
RB7
SW1
SW2
SW3
SW4
SW5
SW6
SW7
SW8
SW9
SW10
SW11
SW12
These keys cannot be used due to CAN interface

- Flowchart


Start
0
1
Check for user key activity
Default Screen?
Yes
Key press detected?
No
Create default screen
Yes
No
D
Display
1
Login Screen?
Yes
Key press detected?
Yes
2
No
D
No
6
Main menu?
Yes
3
No
View log screen?
Yes
4
No
Set time screen?
Yes
5
0
Init gpio
Init keypad
Init clcd
Init timers
Init i²c
Init rtc
Init can
Init system defaults
Enable global interrupts
1


2
Create Login screen
Show login screen
Time out?
Yes
1
No
Accept the password
Try count exceeded ?
Yes
1
No
Password match
No
Increment Try count
Yes
6
3
Create main menu screen
Time out?
Yes
1
No
Key press detected?
No
Yes
Up key?
Yes
2 Secs?
Yes
31
No
Update main menu screen
No
Down key?
Yes
2 Secs?
No
Update main menu screen
Yes
32
D



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

5 Sample Output

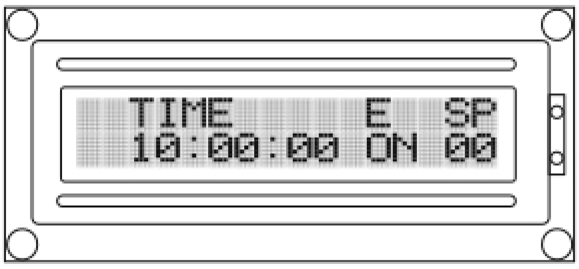


Fig 5 1: Default Screen. Should display the current time, The latest event captured and the current speed

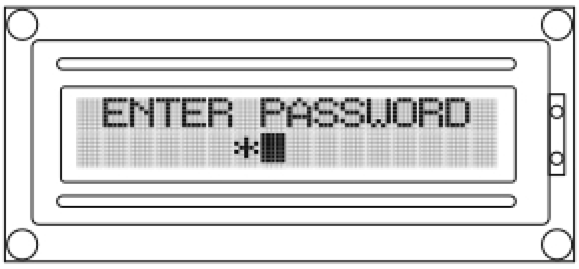


Fig 5 2: User Login. The user needs to enter the password to browse through the available menu

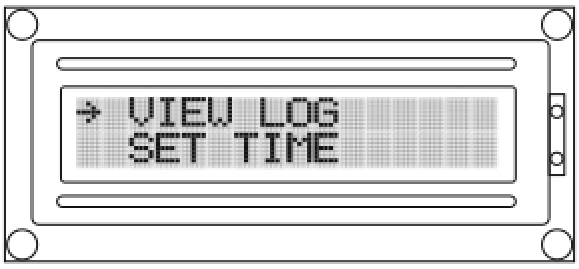


Fig 5 3: The main menu

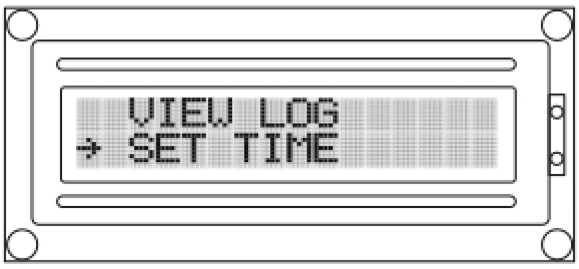


Fig 5 4: The main menu

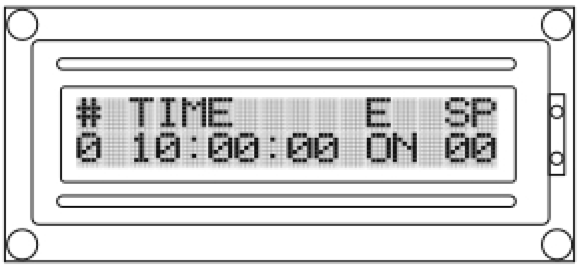


Fig 5 5: The Log screen. Pressing holding the UP key in the main menu should take you this screen

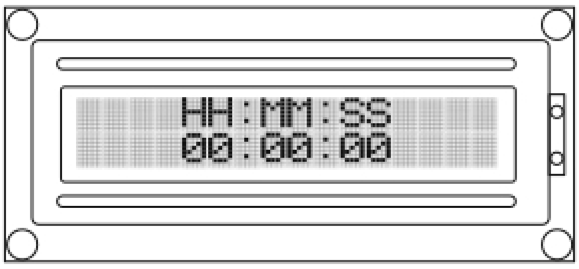


Fig 5 6: Time Setting Screen

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