

MASTERING EMBEDDED SYSTEM ONLINE DIPLOMA

WWW.LEARN-IN-DEPTH.COM

FIRST TERM { FINAL PROJECT 1 }

[HIGH PRESSURE DETECTION SYSTEM]

Eng.

MOHAMED NABIL MOHAMED

My Profile:

@LEARN-IN-DEPTH.COM/M70.ENG

System Architecting Steps

01

Case Study

02

Method

03

Partitioning

04

System Requirements

05

System Analysis

06

System Design

Case Study

Specifications (from the client):

- A pressure controller informs the crew of a cabin with an alarm when the pressure exceeds 20 bars in the cabin.
- The alarm duration equals 60 seconds.
- keeps track of the measured values.

Assumptions:

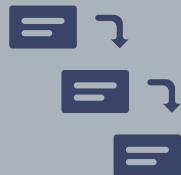
- ☐ The controller set up and shutdown procedures are not modeled.
- ☐ The controller maintenance is not modeled.
- ☐ The pressure sensor never fails.
- ☐ The alarm never fails.
- ☐ The controller never faces power cut.

Versioning :

The “keep track of measured value” option is not modeled in the first version of the design

Method

Waterfall Method



The Waterfall method is the best choice for our High Pressure Detection System due to its structured and sequential approach. It ensures meticulous planning, comprehensive documentation, effective risk management, and thorough testing. These features are crucial for developing a system that requires accuracy, safety, and reliability in high-pressure environments such as the plane navigation cabin.

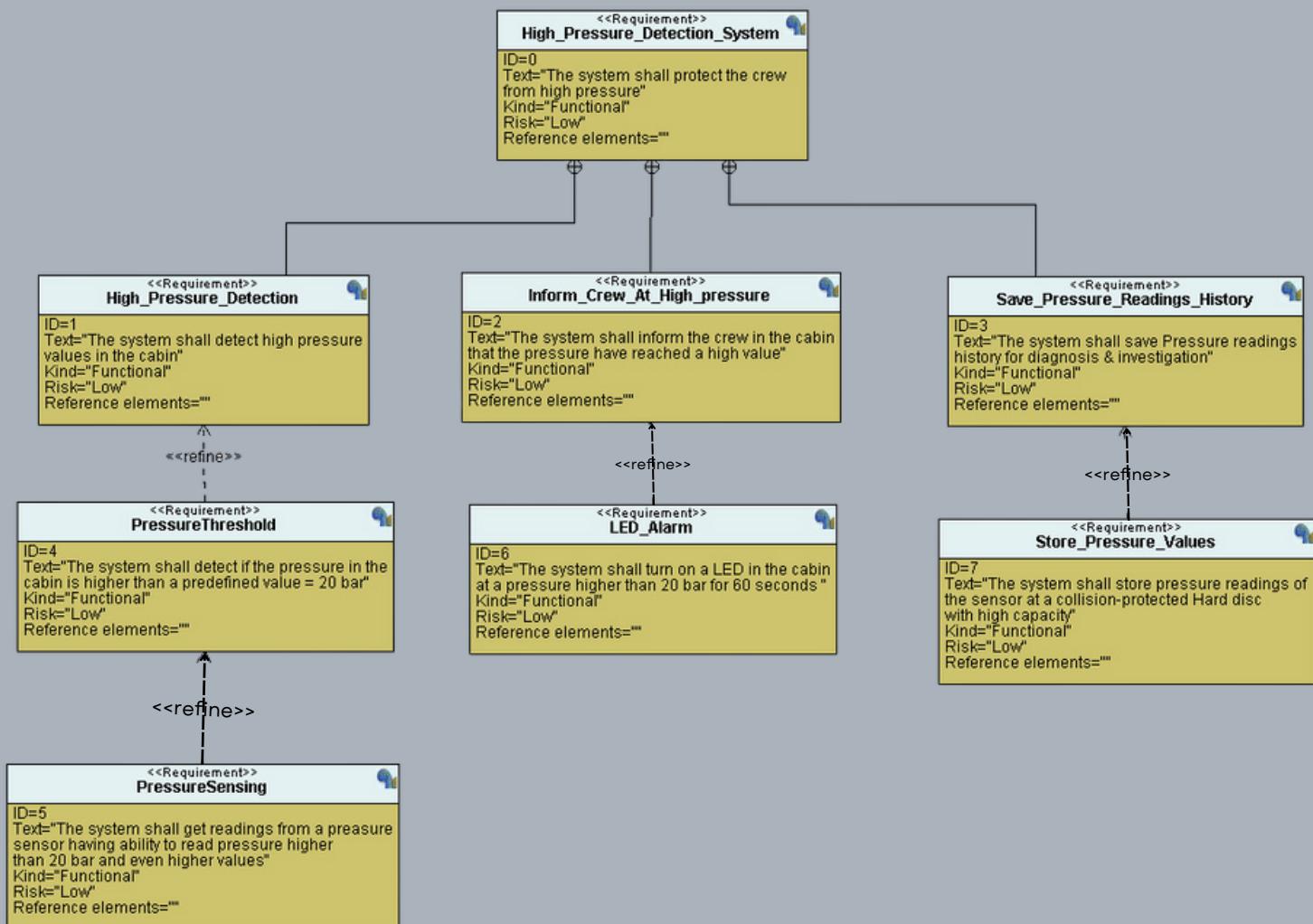
Partitioning

In the System it was determined that the chip that will be used to control the High pressure detection operation is : **STM32F103C6**

System

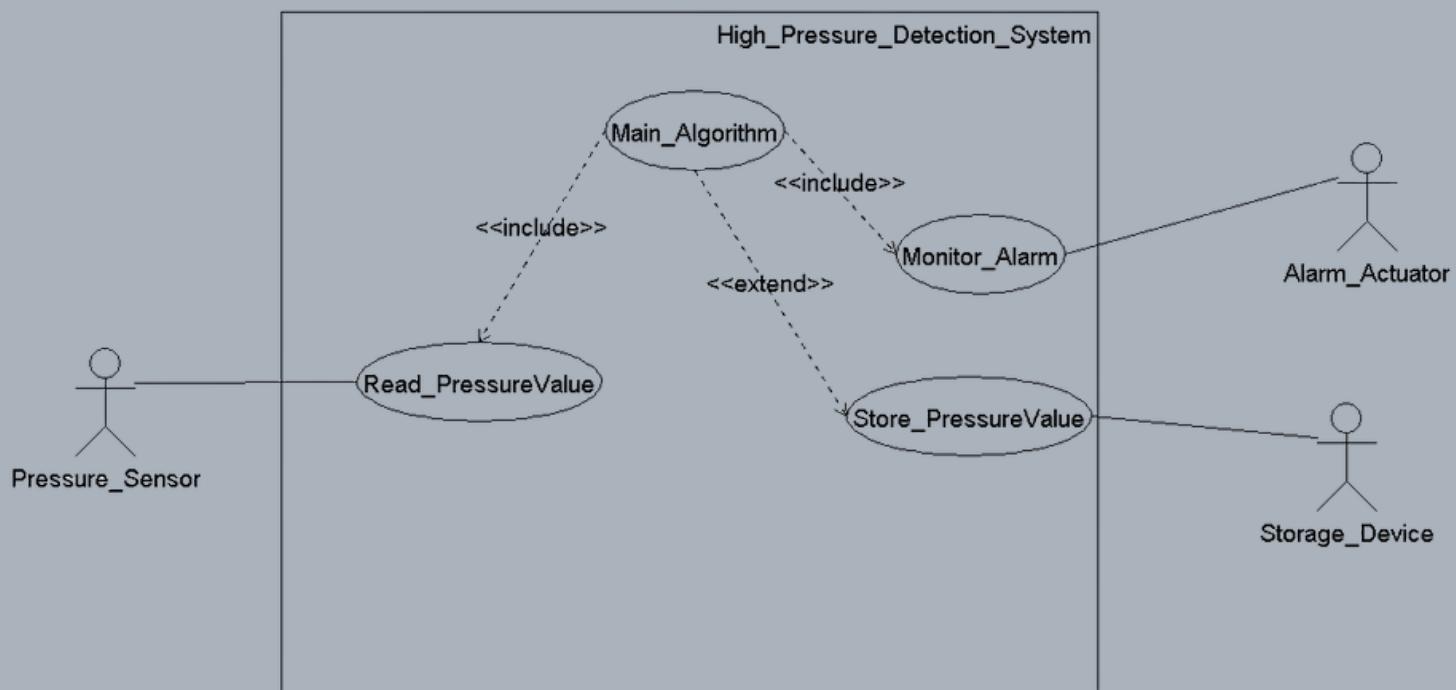
Requirements

Requirement Diagram:

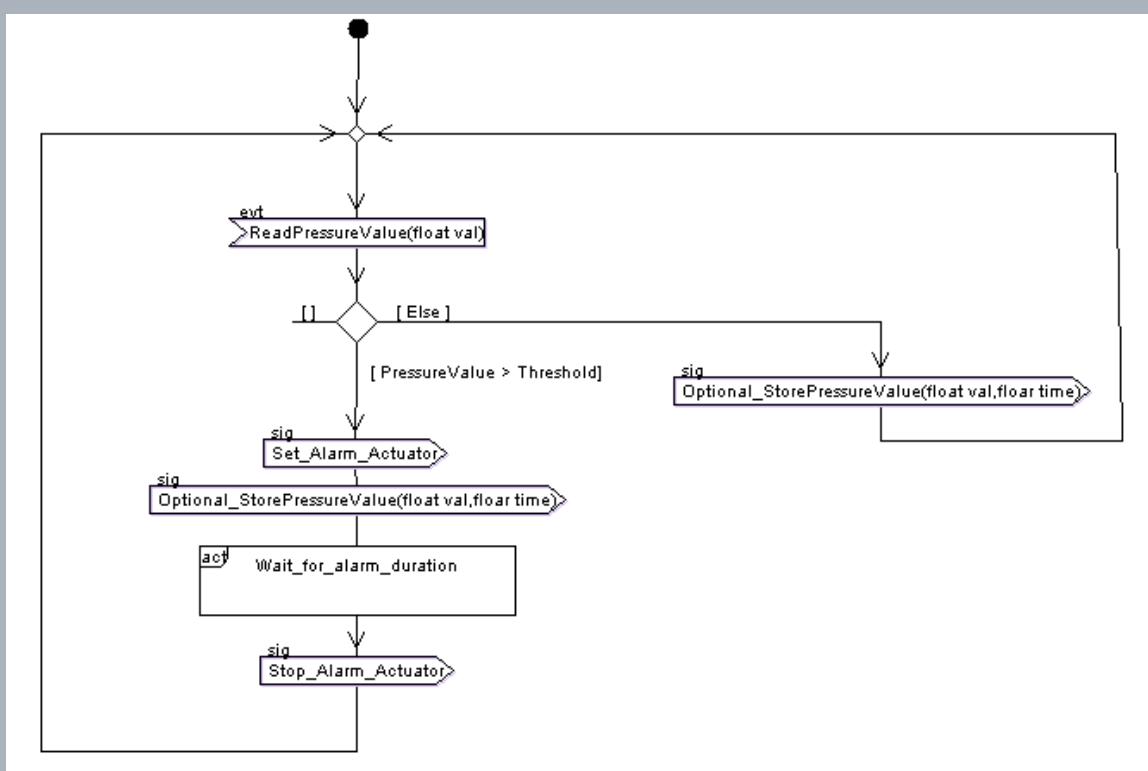


System Analysis

Use Case Diagram

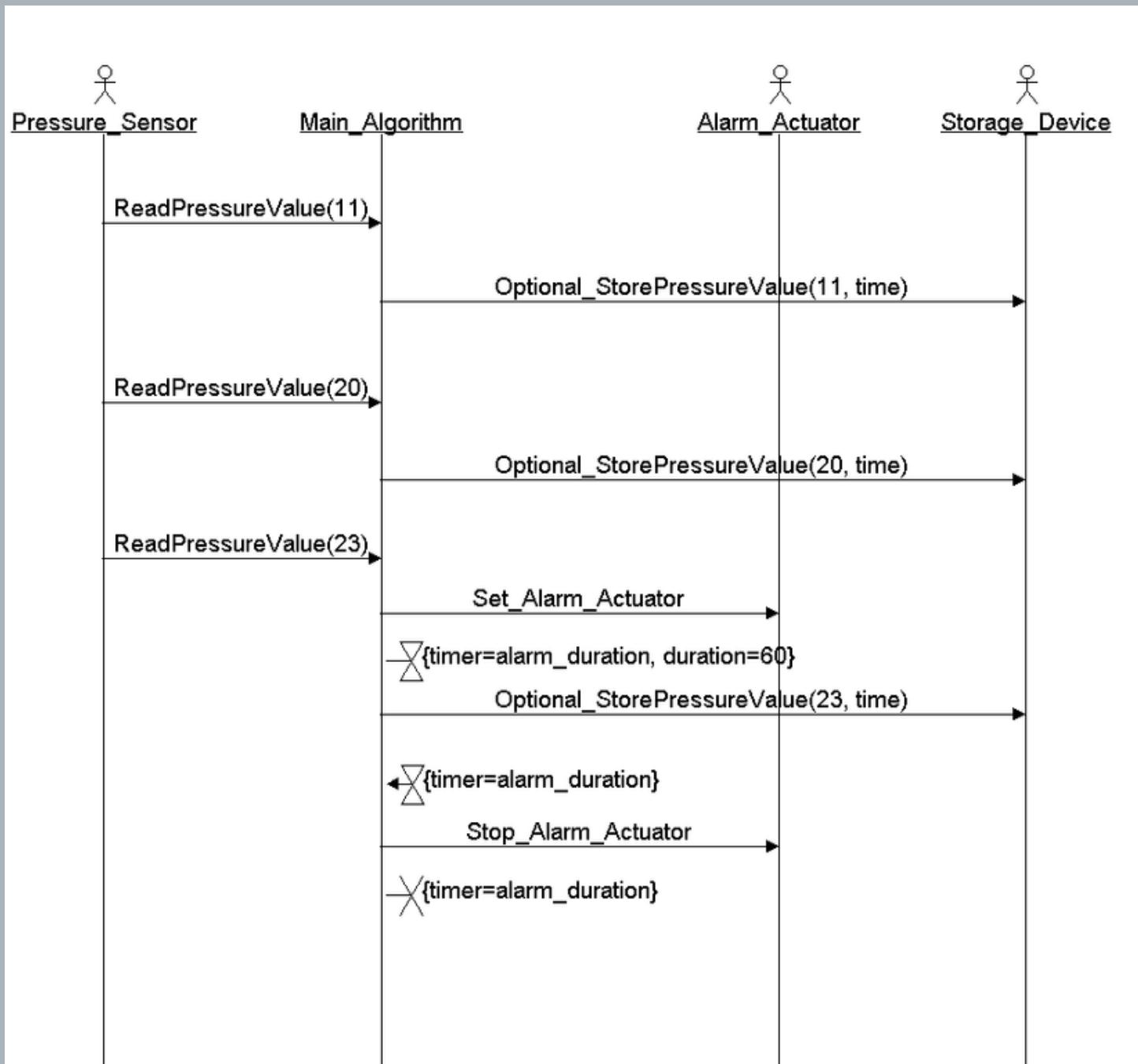


Activity Diagram



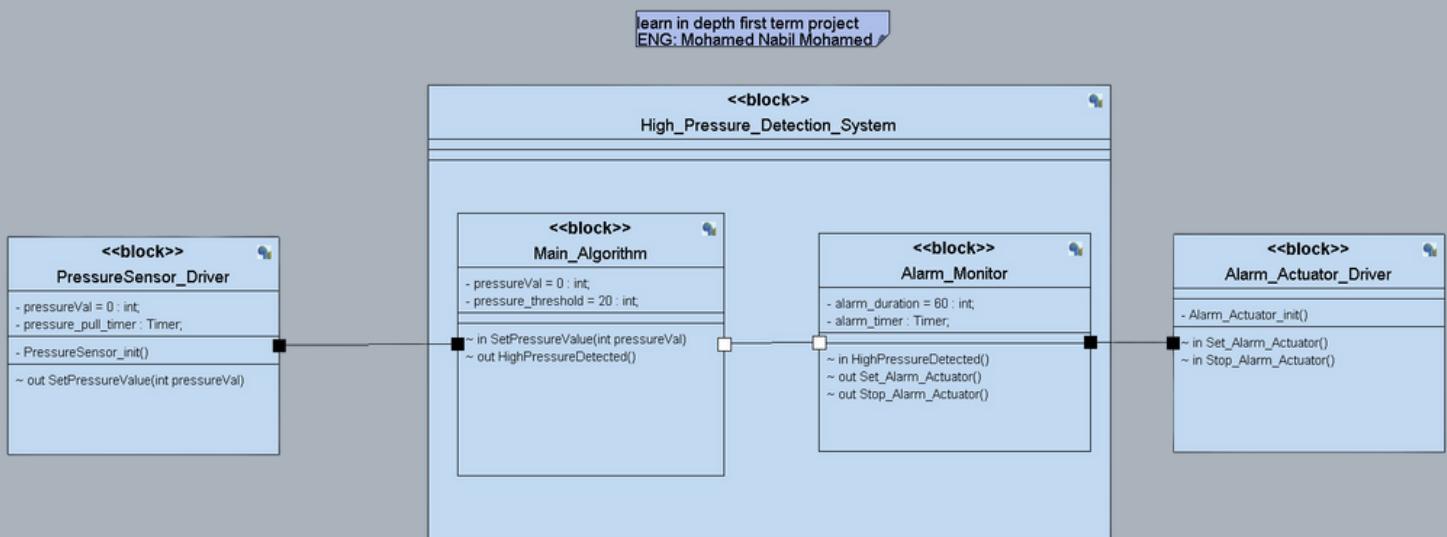
System Analysis

Sequence Diagram



System Design

System Block Diagram



State Machine Diagrams:

1. Pressure Sensor Driver.
2. Main Algorithm.
3. Alarm Monitor.
4. Alarm Actuator Driver.

System Design

Image for each file.C & file.h with the Corresponding state machine:

1-Pressure Sensor Driver.

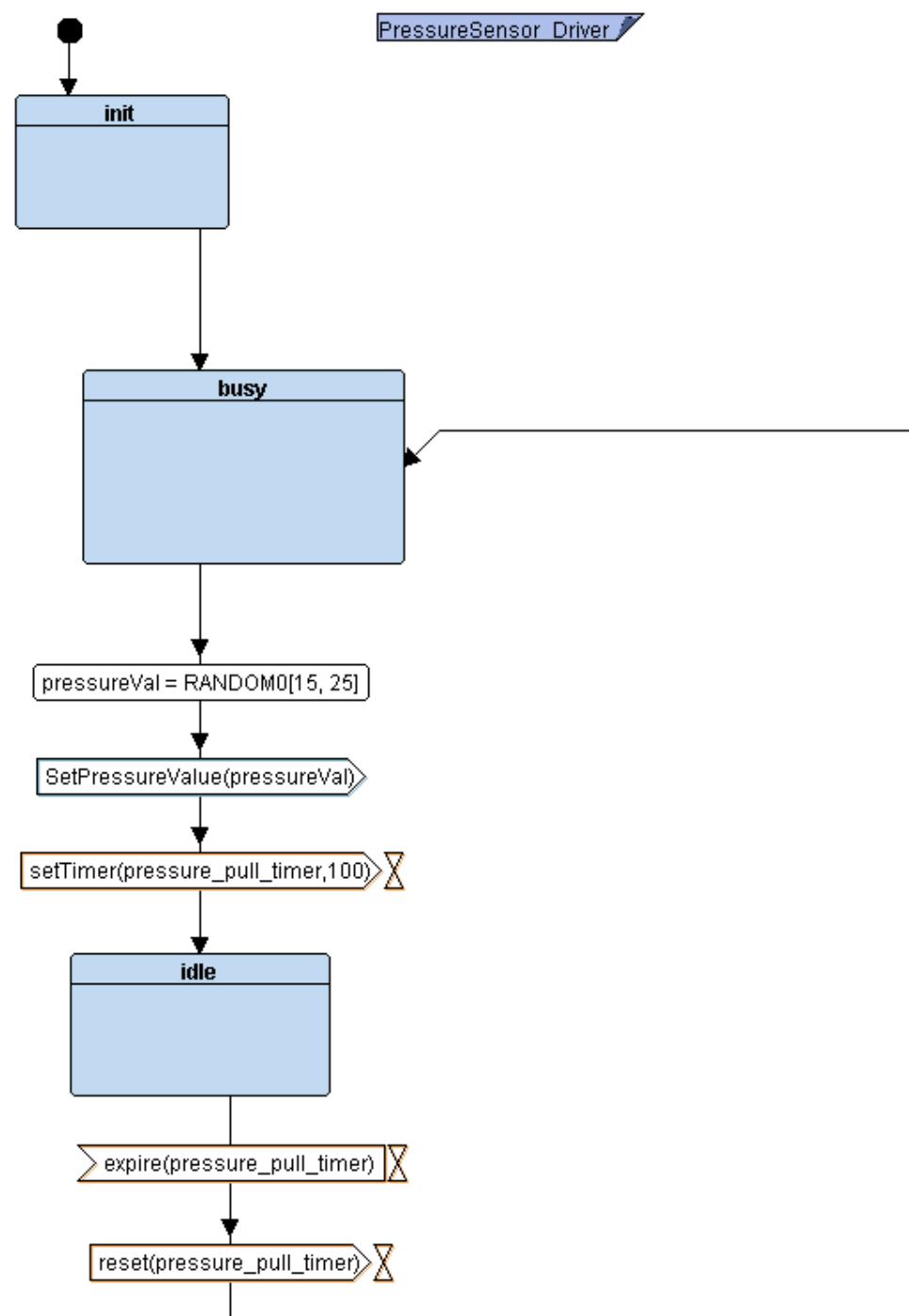
Psensor_Driver.c

```

7 //Includes
8 #include "Psensor_Driver.h"
9 #include "driver.h"
0
1 //Module Variables
2 int Psensor_pressureVal = 0;
3
4 extern void (*Psensor_state)();
5
6 //Functions
7
8 void PressureSensor_init(){
9
10    //Set first state
11    PSensor_state = STATE(Psensor_busy);
12
13 }
14
15 STATE_DEFINE(Psensor_busy){
16
17    //Read pressure value from pressure sensor
18    Psensor_pressureVal = getPressureVal();
19
20    //Send signal (PressureSensor_Driver ---> Main_Algorithm)
21    SetPressureValue(Psensor_pressureVal);
22
23    //Update state
24    PSensor_state = STATE(Psensor_idle);
25
26 }
27
28 STATE_DEFINE(Psensor_idle){
29
30    //Delay between readings
31    Delay(Psensor_pressure_pull_time);
32
33    //Set first state
34    PSensor_state = STATE(Psensor_busy);
35
36 }
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
778
779
780
781
782
783
784
785
786
787
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
877
878
879
880
881
882
883
884
885
886
887
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
917
918
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
938
939
940
941
942
943
944
945
946
947
947
948
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1096
1097
1098
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1166
1167
1168
1169
1170
1171
1172
1173
1173
1174
1175
1176
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1186
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1195
1196
1197
1198
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1207
1208
1209
1210
1211
1212
1213
1214
1215
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1246
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1275
1276
1277
1277
1278
1279
1279
1280
1281
1282
1283
1284
1285
1285
1286
1287
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1305
1306
1307
1307
1308
1309
1309
1310
1311
1312
1313
1314
1314
1315
1316
1316
1317
1318
1318
1319
1319
1320
1321
1321
1322
1323
1323
1324
1325
1325
1326
1327
1327
1328
1328
1329
1329
1330
1331
1331
1332
1333
1333
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1560
1561
1561
1562
1562
1563
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1570
1571
1571
1572
1572
1573
1573
1574
1574
1575
1575
1576
1576
1577
1577
1578
1578
1579
1579
1580
1580
1581
1581
1582
1582
1583
1583
1584
1584
1585
1585
1586
1586
1587
1587
1588
1588
1589
1589
1590
1590
1591
1591
1592
1592
1593
1593
1594
1594
1595
1595
1596
1596
1597
1597
1598
1598
1599
1599
1600
1600
1601
1601
1602
1602
1603
1603
1604
1604
1605
1605
1606
1606
1607
1607
1608
1608
1609
1609
1610
1610
1611
1611
1612
1612
1613
1613
1614
1614
1615
1615
1616
1616
1617
1617
1618
1618
1619
1619
1620
1620
1621
1621
1622
1622
1623
1623
1624
1624
1625
1625
1626
1626
1627
1627
1628
1628
1629
1629
1630
1630
1631
1631
1632
1632
1633
1633
1634
1634
1635
1635
1636
1636
1637
1637
1638
1638
1639
1639
1640
1640
1641
1641
1642
1642
1643
1643
1644
1644
1645
1645
1646
1646
1647
1647
1648
1648
1649
1649
1650
1650
1651
1651
1652
1652
1653
1653
1654
1654
1655
1655
1656
1656
1657
1657
1658
1658
1659
1659
1660
1660
1661
1661
1662
1662
1663
1663
1664
1664
1665
1665
1666
1666
1667
1667
1668
1668
1669
1669
1670
1670
1671
1671
1672
1672
1673
1673
1674
1674
1675
1675
1676
1676
1677
1677
1678
1678
1679
1679
1680
1680
1681
1681
1682
1682
1683
1683
1684
1684
1685
1685
1686
1686
1687
1687
1688
1688
1689
1689
1690
1690
1691
1691
1692
1692
1693
1693
1694
1694
1695
1695
1696
1696
1697
1697
1698
1698
1699
1699
1700
1700
1701
1701
1702
1702
1703
1703
1704
1704
1705
1705
1706
1706
1707
1707
1708
1708
1709
1709
1710
1710
1711
1711
1712
1712
1713
1713
1714
1714
1715
1715
1716
1716
1717
1717
1718
1718
1719
1719
1720
1720
1721
1721
1722
1722
1723
1723
1724
1724
1725
1725
1726
1726
1727
1727
1728
1728
1729
1729
1730
1730
1731
1731
1732
1732
1733
1733
1734
1734
1735
1735
1736
1736
1737
1737
1738
1738
1739
1739
1740
1740
1741
1741
1742
1742
1743
1743
1744
1744
1745
1745
1746
1746
1747
1747
1748
1748
1749
1749
1750
1750
1751
1751
1752
1752
1753
1753
1754
1754
1755
1755
1756
1756
1757
1757
1758
1758
1759
1759
1760
1760
1761
1761
1762
1762
1763
1763
1764
1764
1765
1765
1766
1766
1767
1767
1768
1768
1769
1769
1770
1770
1771
1771
1772
1772
1773
1773
1774
1774
1775
1775
1776
1776
1777
1777
1778
1778
1779
1779
1780
1780
1781
1781
1782
1782
1783
1783
1784
1784
1785
1785
1786
1786
1787
1787
1788
1788
1789
1789
1790
1790
1791
1791
1792
1792
1793
1793
1794
1794
1795
1795
1796
1
```

System Design

Pressure Sensor Driver State Machine



System Design

2-Main Algorithm.

Main_Algorithm.c

```

8 //Includes
9 #include "Main_Algorithm.h"
10
11 //Module Variables
12 int Main_pressureVal = 0;
13 int Main_pressure_threshold = Pthreshold;
14
15 extern void (*Main_Algorithm_state) ();
16
17 //Functions
18
19
20 /*PressureSensor_Driver ---> Main_Algorithm*/
21 void SetPressureValue(int pressureVal){
22
23     Main_pressureVal = pressureVal;
24     Main_Algorithm_state = STATE(High_Pressure_Check);
25 }
26
27
28 STATE_DEFINE(High_Pressure_Check){
29
30     //Receive pressure reading signal (PressureSensor_1)
31
32     //State ID
33     Main_Algorithm_state_id = High_Pressure_Check;
34
35     //Main Algorithm
36     if(Main_pressureVal > Main_pressure_threshold ){
37
38         //Main_Algorithm ---> Alarm_Monitor
39         HighPressureDetected();
40
41     }
42
43
44     //Update state
45     Main_Algorithm_state = STATE(High_Pressure_Check);
46
47 }
```

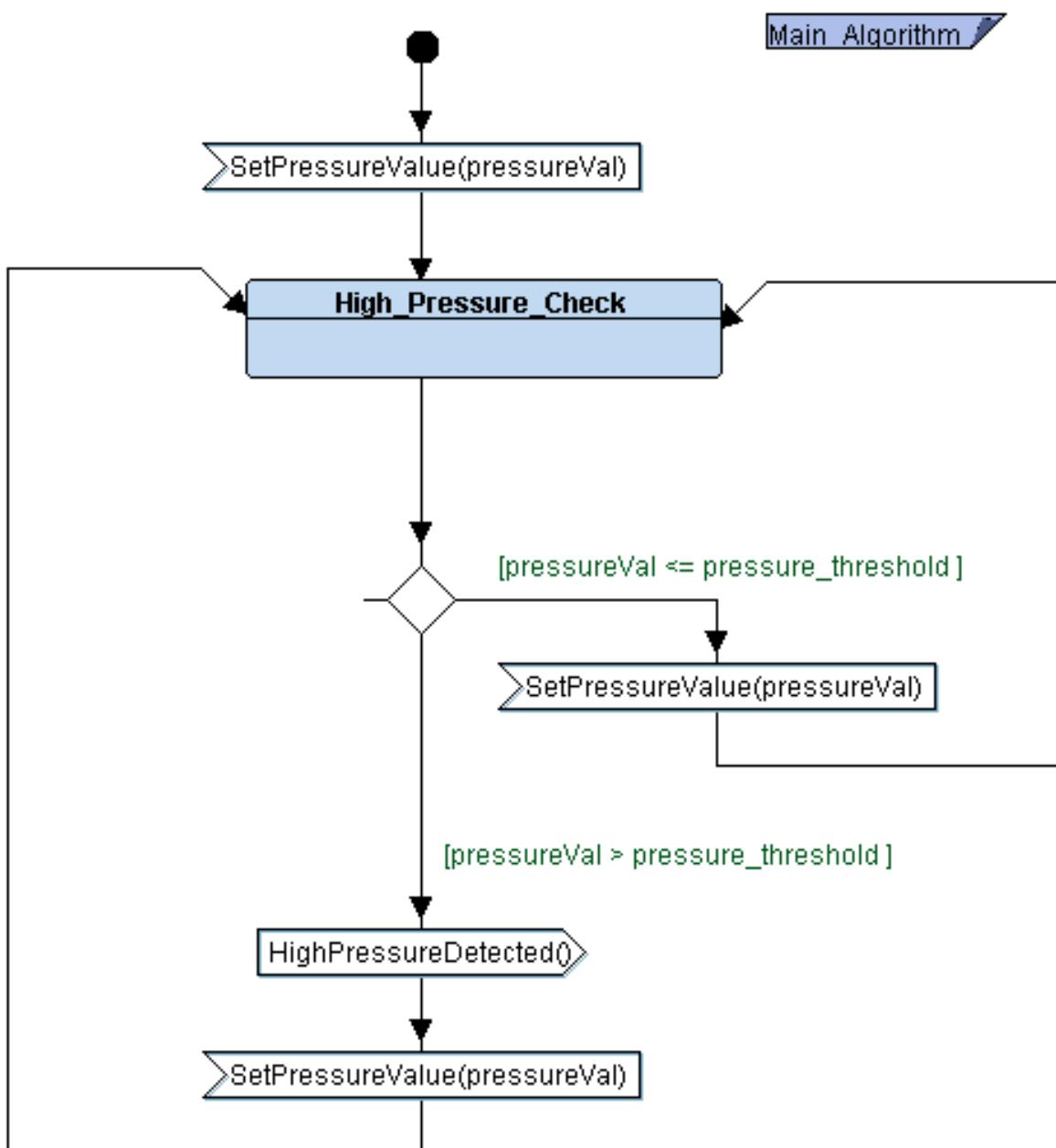
Main_Algorithm.h

```

1  /*
2   * Main_Algorithm.h
3   *
4   * Created on: Aug 29, 2023
5   *           Code By: Mohamed Nabil Mohamed
6   */
7
8 #ifndef MAIN_ALGORITHM_H_
9 #define MAIN_ALGORITHM_H_
10
11 //Includes
12 #include "state.h"
13
14 //Defines
15
16 //select Pressure threshold value in bars
17 #define Pthreshold 20
18
19 //State IDs
20 enum{
21     High_Pressure_Check
22 }Main_Algorithm_state_id;
23
24
25 //Prototypes
26 STATE_DEFINE(High_Pressure_Check);
27
28
29 //State pointer
30 void (*Main_Algorithm_state) ();
31
32
33#endif /* MAIN_ALGORITHM_H_ */
34
```

System Design

Main Algorithm State Machine



System Design

3-Alarm Monitor.

Alarm_Monitor.c

```

9 //Includes
0 #include "Alarm_Monitor.h"
1 #include "driver.h"
2
3 //Module Variables
4 extern void (*Alarm_Monitor_state)();
5
6 //Functions
7
8 //Main Algorithm ---> Alarm_Monitor
9 void HighPressureDetected(){
0     Alarm_Monitor_state = STATE(Alarm_Monitor_AlarmOn);
1 }
2
3
4 STATE_DEFINE(Alarm_Monitor_AlarmOff){
5     Alarm_Monitor_state_id = Alarm_Monitor_AlarmOff;
6     //Do nothing until high pressure is detected
7 }
8
9 STATE_DEFINE(Alarm_Monitor_AlarmOn){
0
1     //State ID
2     Alarm_Monitor_state_id = Alarm_Monitor_AlarmOn;
3
4     //Alarm_Monitor ---> Alarm_Actuator_Driver
5     Set_Alarm_Actuator();
6
7     //set next state
8     Alarm_Monitor_state = STATE(Alarm_Monitor_waiting);
9 }
0
1
2 STATE_DEFINE(Alarm_Monitor_waiting){
3
4     Alarm_Monitor_state_id = Alarm_Monitor_waiting;
5
6     //wait for alarm_duration
7     Delay(alarm_duration);
8
9     //stop alarm
0     Stop_Alarm_Actuator();
1
2     //set next state
3     Alarm_Monitor_state = STATE(Alarm_Monitor_AlarmOff);
4 }
5
6
7
8
9

```

Alarm_Monitor.h

```

# ifndef ALARM_MONITOR_H_
# define ALARM_MONITOR_H_

//Defines
#define alarm_duration 20000

//Includes
#include "state.h"

//States IDs
enum{
    Alarm_Monitor_AlarmOn,
    Alarm_Monitor_AlarmOff,
    Alarm_Monitor_waiting
}Alarm_Monitor_state_id;

//Prototypes
STATE_DEFINE(Alarm_Monitor_AlarmOn);
STATE_DEFINE(Alarm_Monitor_AlarmOff);
STATE_DEFINE(Alarm_Monitor_waiting);

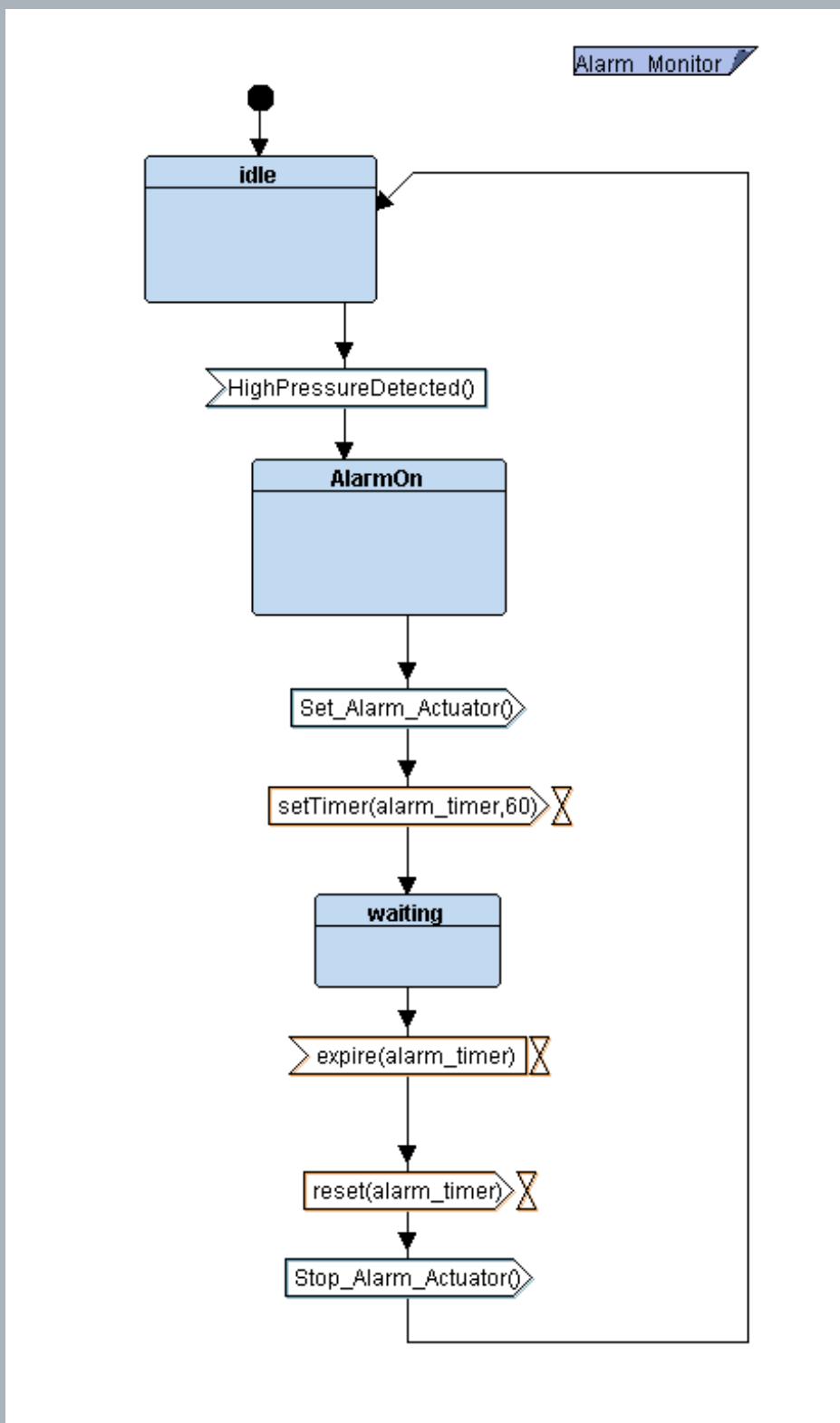
//State pointer
void (*Alarm_Monitor_state)();

#endif /* ALARM_MONITOR_H_ */

```

System Design

Alarm Monitor State Machine



System Design

4-Alarm Actuator Driver.

Alarm_Actuator_Driver.c

```
//Includes
#include "Alarm_Actuator_Driver.h"
#include "driver.h"

//Module Variables
extern void (*Alarm_Actuator_state)();

//Functions

void Alarm_Actuator_init() {
    //Set first state
    Alarm_Actuator_state = STATE(Alarm_Actuator_off);
}

/* Alarm_Monitor ---> Alarm_Actuator_Driver */
void Set_Alarm_Actuator() {
    //Set equivalent state
    Alarm_Actuator_state = STATE(Alarm_Actuator_on);
}

void Stop_Alarm_Actuator() {
    //Set equivalent state
    Alarm_Actuator_state = STATE(Alarm_Actuator_off);
}

STATE_DEFINE(Alarm_Actuator_off) {
    Set_Alarm_actuator(OFF);
}

STATE_DEFINE(Alarm_Actuator_on) {
    Set_Alarm_actuator(ON);
}
```

Alarm_Actuator_Driver.h

```
#ifndef ALARM_ACTUATOR_DRIVER_H_
#define ALARM_ACTUATOR_DRIVER_H_

//Defines
//according to given project connection (Actuator)
#define ON 0
#define OFF 1

//Includes
#include "state.h"

//State IDs
enum{
    Alarm_Actuator_off,
    Alarm_Actuator_on
}Alarm_Actuator_state_id;

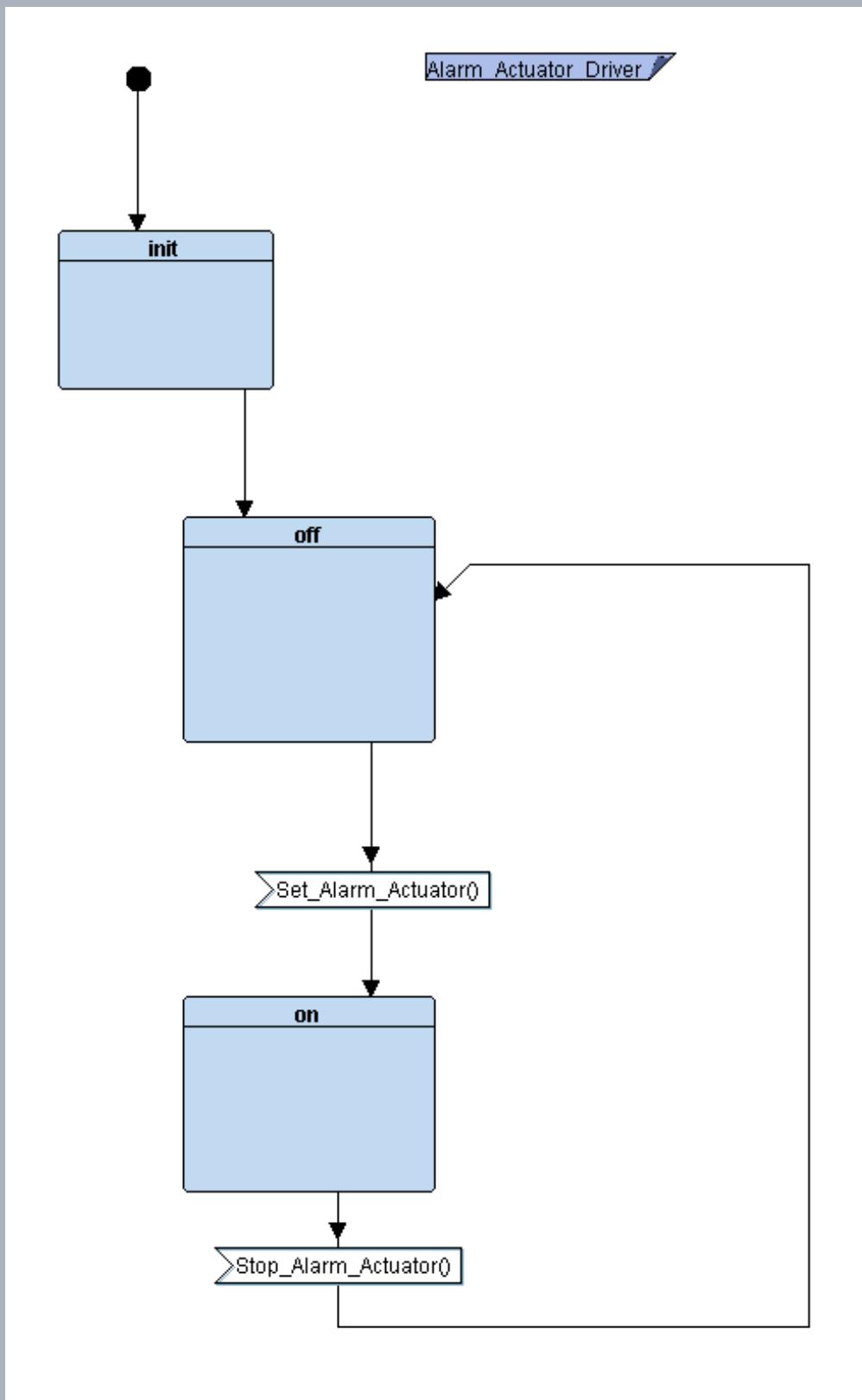
//Prototypes
void Alarm_Actuator_init();
STATE_DEFINE(Alarm_Actuator_on);
STATE_DEFINE(Alarm_Actuator_off);

//State pointer
void (*Alarm_Actuator_state)();

#endif /* ALARM_ACTUATOR_DRIVER_H_ */
```

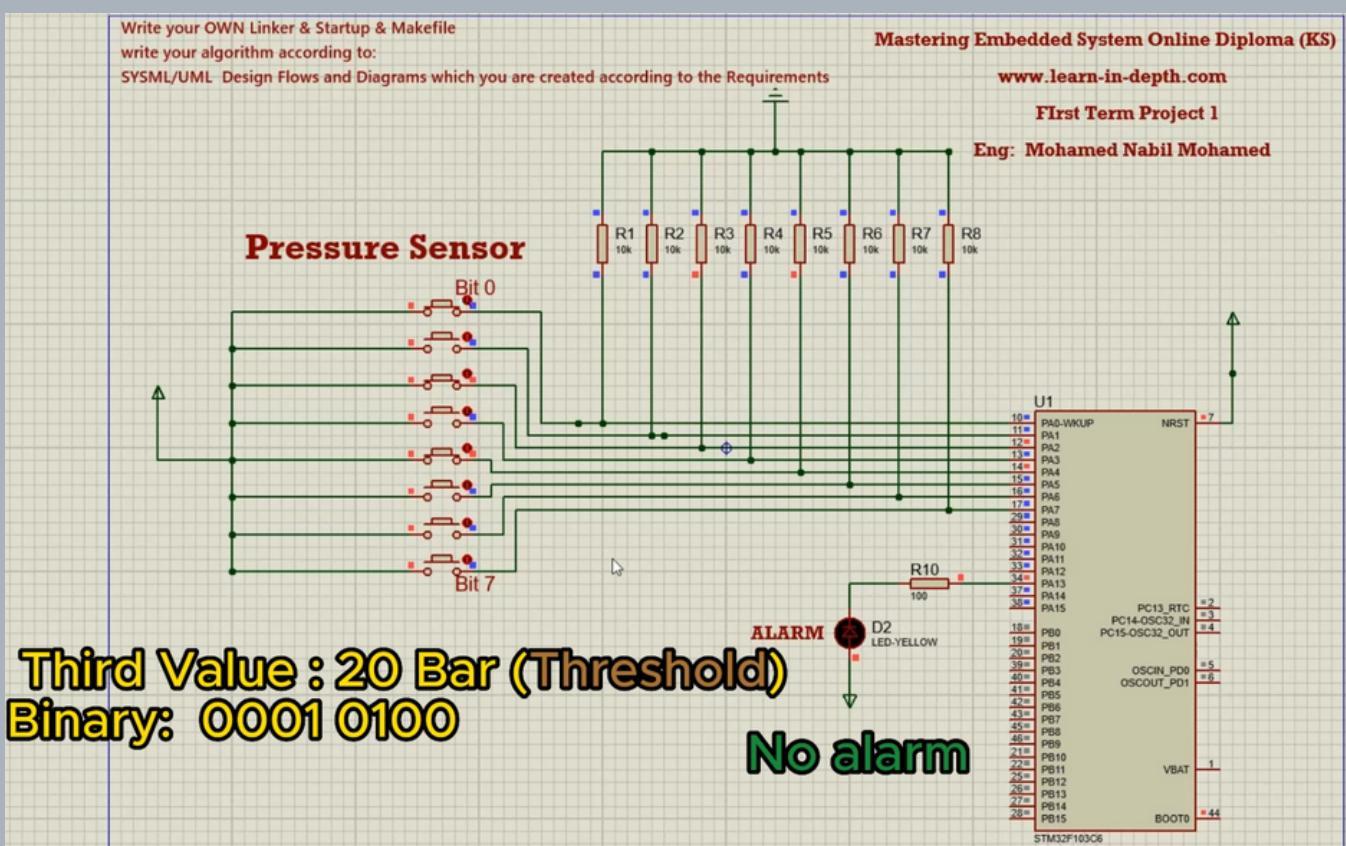
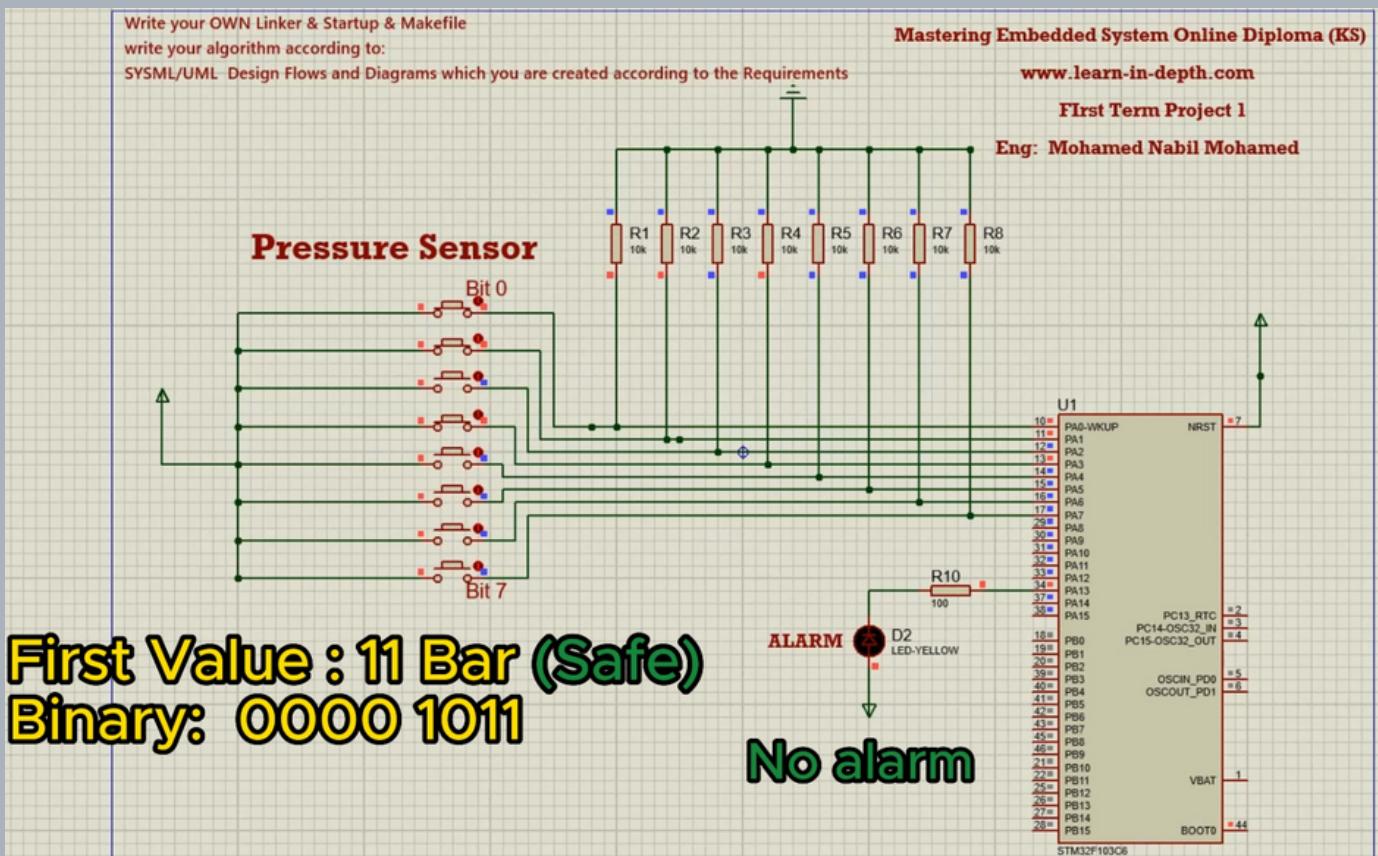
System Design

Alarm Actuator Driver State Machine



System Design

Simulation Results



System Design

Simulation Results

Write your OWN Linker & Startup & Makefile
write your algorithm according to:

SYSML/UML Design Flows and Diagrams which you are created according to the Requirements

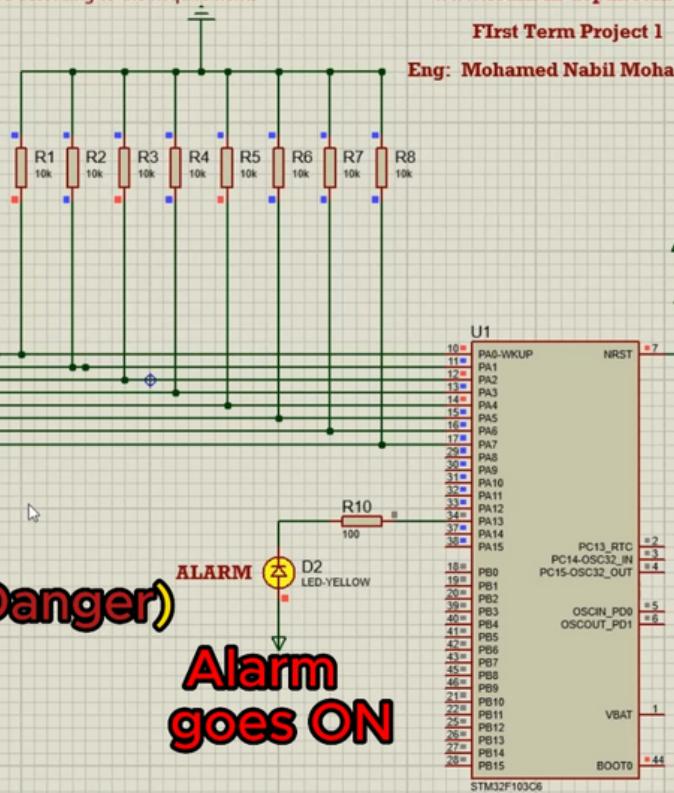
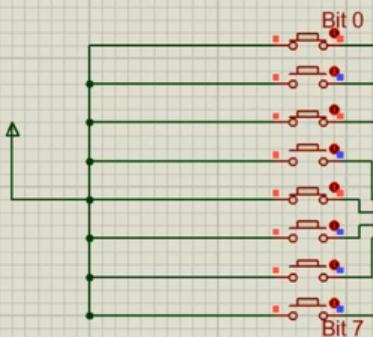
Mastering Embedded System Online Diploma (KS)

www.learn-in-depth.com

FIRrst Term Project 1

Eng: Mohamed Nabil Mohamed

Pressure Sensor



Second Value : 21 Bar (Danger)
Binary: 0001 0101

Alarm goes ON

Write your OWN Linker & Startup & Makefile
write your algorithm according to:

SYSML/UML Design Flows and Diagrams which you are created according to the Requirements

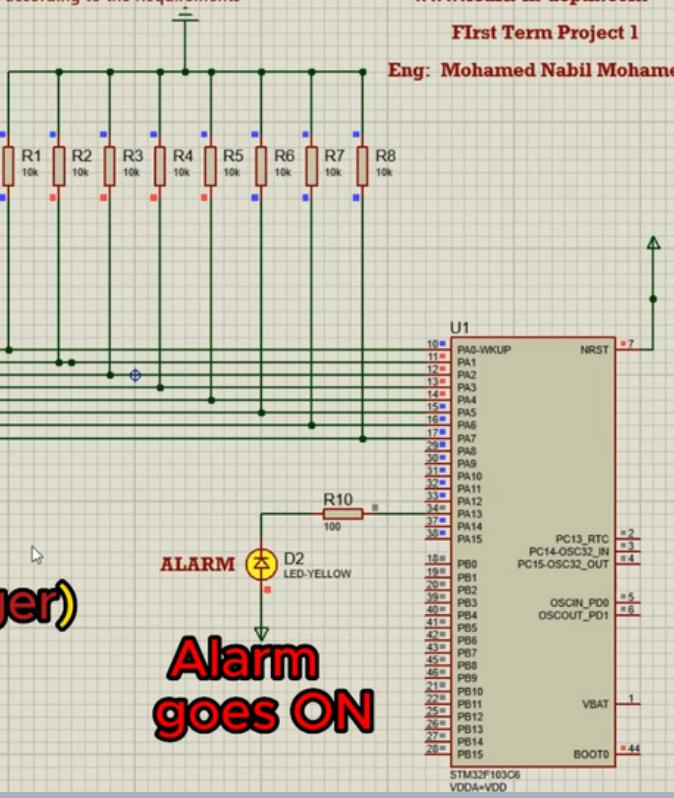
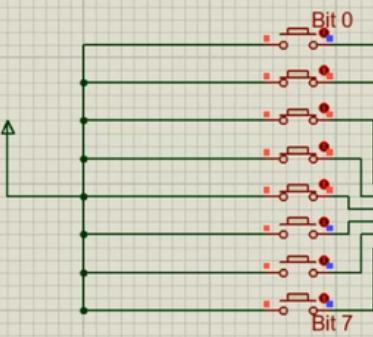
Mastering Embedded System Online Diploma (KS)

www.learn-in-depth.com

FIRrst Term Project 1

Eng: Mohamed Nabil Mohamed

Pressure Sensor



Last Value : 30 Bar (Danger)
Binary: 0001 1110

Alarm goes ON

System Design

Map File

```

2 Allocating common symbols
3 Common symbol      size      file
4
5 Alarm_Actuator_state_id      0x1      Alarm_Actuator_Driver.o
6 PSensor_state      0x4      main.o
7 Alarm_Actuator_state      0x4      Alarm_Actuator_Driver.o
8
9 Main_Algorithm_state      0x4      main.o
10 Main_Algorithm_state_id     0x1      main.o
11 PSensor_state_id      0x1      main.o
12 Alarm_Monitor_state_id    0x1      Alarm_Monitor.o
13 Alarm_Monitor_state      0x4      Alarm_Monitor.o
14
15 Memory Configuration
16
17 Name      Origin      Length      Attributes
18 flash      0x080000000      0x00020000      xr
19 SRAM      0x200000000      0x00005000      xrw
20 *default*  0x000000000      0xffffffff
21
22 Linker script and memory map
23
24
25 .text      0x080000000      0x460
26 *(.vector*)
27 .vectors      0x080000000      0x1c startup.o
28           0x080000000      vectors
29 *(.text*)
30 .text      0x08000001c      0x74 Alarm_Actuator_Driver.o
31           0x08000001c      Alarm_Actuator_init
32           0x080000038      Set_Alarm_Actuator
33           0x080000054      Stop_Alarm_Actuator
34           0x080000070      stateFunction_Alarm_Actuator_off
35           0x080000080      stateFunction_Alarm_Actuator_on
36 .text      0x080000090      0x94 Alarm_Monitor.o
37           0x080000090      HighPressureDetected
38           0x0800000ac      stateFunction_Alarm_Monitor_AlarmOff
39           0x0800000c4      stateFunction_Alarm_Monitor_AlarmOn
40           0x0800000f0      stateFunction_Alarm_Monitor_waiting
41 .text      0x080000124      0x10c driver.o
42           0x080000124      Delay
43           0x080000148      getPressureVal
44           0x080000160      Set_Alarm_actuator
45           0x0800001b0      GPIO_INITIALIZATION
46 .text      0x080000230      0x8c main.o
47           0x080000230      setup
48           0x080000268      main
49 .text      0x0800002bc      0x74 Main_Algorithm.o
50           0x0800002bc      SetPressureValue
51           0x0800002ec      stateFunction_High_Pressure_Check
52 .text      0x080000330      0x74 Psensor_Driver.o
53           0x080000330      PressureSensor_init
54           0x08000034c      stateFunction_PSensor_busy
55           0x080000384      stateFunction_PSensor_idle
56 .text      0x0800003a4      0xbc startup.o
57           0x0800003a4      Reset_Handler
58           0x080000454      MM_Fault_Handler
59           0x080000454      H_Fault_Handler
60           0x080000454      Bus_Fault
61           0x080000454      Default_Handler
62           0x080000454      Usage_Fault_Handler
63           0x080000454      NMI_Handler
64 *(.rodata)
65           0x080000460      _E_text = .
66
67 .glue_7      0x080000460      0x0
68 .glue_7      0x000000000      0x0 linker stubs
69
70 .glue_7t     0x080000460      0x0
71 .glue_7t     0x000000000      0x0 linker stubs
72
73 .vfp11_veneer 0x080000460      0x0
74 .vfp11_veneer 0x000000000      0x0 linker stubs
75
76
77
78
79
80

```

System Design

Map File

```

81 .v4_bx      0x08000460    0x0
82 .v4_bx      0x00000000    0x0 linker stubs
83
84 .iplt       0x08000460    0x0
85 .iplt       0x00000000    0x0 Alarm_Actuator_Driver.o
86
87 .rel.dyn    0x08000460    0x0
88 .rel.iplt   0x00000000    0x0 Alarm_Actuator_Driver.o
89
90 .data        0x20000000    0x4 load address 0x08000460
91             0x20000000    _S_data = .
92 *(.data)
93 .data        0x20000000    0x0 Alarm_Actuator_Driver.o
94 .data        0x20000000    0x0 Alarm_Monitor.o
95 .data        0x20000000    0x0 driver.o
96 .data        0x20000000    0x0 main.o
97 .data        0x20000000    0x4 Main_Algorithm.o
98             0x20000000    Main_pressure_threshold
99 .data        0x20000004    0x0 Psensor_Driver.o
100 .data       0x20000004    0x0 startup.o
101            0x20000004    . = ALIGN (0x4)
102            0x20000004    _E_data = .
103
104 .igot.plt   0x20000004    0x0 load address 0x08000464
105 .igot.plt   0x00000000    0x0 Alarm_Actuator_Driver.o
106
107 .bss         0x20000004    0x1022 load address 0x08000464
108            0x20000004    _S_bss = .
109 *(.bss)
110 .bss         0x20000004    0x0 Alarm_Actuator_Driver.o
111 .bss         0x20000004    0x0 Alarm_Monitor.o
112 .bss         0x20000004    0x0 driver.o
113 .bss         0x20000004    0x0 main.o
114 .bss         0x20000004    0x4 Main_Algorithm.o
115            0x20000004    Main_pressureVal
116 .bss         0x20000008    0x4 Psensor_Driver.o
117            0x20000008    Psensor_pressureVal
118 .bss         0x2000000c    0x0 startup.o
119            0x2000000c    _E_bss = .
120            0x2000000c    . = ALIGN (0x4)
121            0x2000100c    . = (. + 0x1000)
122 *fill*
123           0x2000000c    0x1000
124           0x2000100c    _stack_top = .
125 COMMON       0x2000100c    0x8 Alarm_Actuator_Driver.o
126           0x2000100c    Alarm_Actuator_state_id
127 COMMON       0x20001010    0x8 Alarm_Monitor.o
128           0x20001014    Alarm_Monitor_state_id
129           0x20001018    Alarm_Monitor_state
130 COMMON       0x2000101c    0xa main.o
131           0x2000101c    PSensor_state
132           0x20001020    Main_Algorithm_state
133           0x20001024    Main_Algorithm_state_id
134           0x20001025    PSensor_state_id
135 LOAD Alarm_Actuator_Driver.o
136 LOAD Alarm_Monitor.o
137 LOAD driver.o
138 LOAD main.o
139 LOAD Main_Algorithm.o
140 LOAD Psensor_Driver.o
141 LOAD startup.o
142 OUTPUT(High_Pressure_Detection_System.elf elf32-littlearm)
143
144 .debug_info  0x00000000    0x866
145 .debug_info  0x00000000    0x129 Alarm_Actuator_Driver.o
146 .debug_info  0x00000129    0x11a Alarm_Monitor.o
147 .debug_info  0x00000243    0x103 driver.o
148 .debug_info  0x00000346    0x199 main.o
149 .debug_info  0x000004df    0x10e Main_Algorithm.o
150 .debug_info  0x000005ed    0x111 Psensor_Driver.o
151 .debug_info  0x000006fe    0x168 startup.o
152
153 .debug_abbrev 0x00000000    0x44e
154 .debug_abbrev 0x00000000    0x92 Alarm_Actuator_Driver.o
155 .debug_abbrev 0x00000092    0x92 Alarm_Monitor.o
156 .debug_abbrev 0x00000124    0x9d driver.o
157 .debug_abbrev 0x000001c1    0x94 main.o
158 .debug_abbrev 0x00000255    0xa5 Main_Algorithm.o
159 .debug_abbrev 0x000002fa    0x92 Psensor_Driver.o
160 .debug_abbrev 0x0000038c    0xc2 startup.o

```

System Design

Symbols Table

```
$ arm-none-eabi-nm.exe High_Pressure_Detection_System.elf
2000000c B _E_bss
20000004 D _E_data
08000460 T _E_text
20000004 B _S_bss
20000000 D _S_data
2000100c B _stack_top
0800001c T Alarm_Actuator_init
20001010 B Alarm_Actuator_state
2000100c B Alarm_Actuator_state_id
20001018 B Alarm_Monitor_state
20001014 B Alarm_Monitor_state_id
08000454 W Bus_Fault
08000454 T Default_Handler
08000124 T Delay
08000148 T getPressureVal
080001b0 T GPIO_INITIALIZATION
08000454 W H_Fault_Handler
08000090 T HighPressureDetected
08000268 T main
20001020 B Main_Algorithm_state
20001024 B Main_Algorithm_state_id
20000000 D Main_pressure_threshold
20000004 B Main_pressureVal
08000454 W MM_Fault_Handler
08000454 W NMI_Handler
08000330 T PressureSensor_init
20000008 B Psensor_pressureVal
2000101c B PSensor_state
20001025 B PSensor_state_id
080003a4 T Reset_Handler
08000160 T Set_Alarm_actuator
08000038 T Set_Alarm_Actuator
080002bc T SetPressureValue
08000230 T setup
08000070 T stateFunction_Alarm_Actuator_off
08000080 T stateFunction_Alarm_Actuator_on
080000ac T stateFunction_Alarm_Monitor_AlarmOff
080000c4 T stateFunction_Alarm_Monitor_AlarmOn
080000f0 T stateFunction_Alarm_Monitor_waiting
080002ec T stateFunction_High_Pressure_Check
0800034c T stateFunction_PSensor_busy
08000384 T stateFunction_PSensor_idle
08000054 T Stop_Alarm_Actuator
08000454 W Usage_Fault_Handler
08000000 T vectors
```

System Design

Symbols Table:

```
$ arm-none-eabi-nm.exe High_Pressure_Detection_System.elf
2000000c B _E_bss
20000004 D _E_data
08000460 T _E_text
20000004 B _S_bss
20000000 D _S_data
2000100c B _stack_top
0800001c T Alarm_Actuator_init
20001010 B Alarm_Actuator_state
2000100c B Alarm_Actuator_state_id
20001018 B Alarm_Monitor_state
20001014 B Alarm_Monitor_state_id
08000454 W Bus_Fault
08000454 T Default_Handler
08000124 T Delay
08000148 T getPressureVal
080001b0 T GPIO_INITIALIZATION
08000454 W H_Fault_Handler
08000090 T HighPressureDetected
08000268 T main
20001020 B Main_Algorithm_state
20001024 B Main_Algorithm_state_id
20000000 D Main_pressure_threshold
20000004 B Main_pressureVal
08000454 W MM_Fault_Handler
08000454 W NMI_Handler
08000330 T PressureSensor_init
20000008 B Psensor_pressureVal
2000101c B Psensor_state
20001025 B Psensor_state_id
080003a4 T Reset_Handler
08000160 T Set_Alarm_actuator
08000038 T Set_Alarm_Actuator
080002bc T SetPressureValue
08000230 T setup
08000070 T stateFunction_Alarm_Actuator_off
08000080 T stateFunction_Alarm_Actuator_on
080000ac T stateFunction_Alarm_Monitor_AlarmOff
080000c4 T stateFunction_Alarm_Monitor_AlarmOn
080000f0 T stateFunction_Alarm_Monitor_waiting
080002ec T stateFunction_High_Pressure_Check
0800034c T stateFunction_PSensor_busy
08000384 T stateFunction_PSensor_idle
08000054 T Stop_Alarm_Actuator
08000454 W Usage_Fault_Handler
08000000 T vectors
```

Section Table:

```
$ arm-none-eabi-objdump.exe -h High_Pressure_Detection_System.elf
High_Pressure_Detection_System.elf:      file format elf32-littlearm

Sections:
Idx Name          Size    VMA       LMA       File off  Align
0 .text          00000460 08000000 08000000 00008000 2**2
                CONTENTS, ALLOC, LOAD, READONLY, CODE
1 .data          00000004 20000000 08000460 00010000 2**2
                CONTENTS, ALLOC, LOAD, DATA
2 .bss           00001022 20000004 08000464 00010004 2**2
                ALLOC
3 .debug_info   00000866 00000000 00000000 00010004 2**0
                CONTENTS, READONLY, DEBUGGING
4 .debug_abbrev 0000044e 00000000 00000000 0001086a 2**0
                CONTENTS, READONLY, DEBUGGING
5 .debug_loc    000003f8 00000000 00000000 00010cb8 2**0
                CONTENTS, READONLY, DEBUGGING
6 .debug_aranges 000000e0 00000000 00000000 000110b0 2**0
                CONTENTS, READONLY, DEBUGGING
7 .debug_line   00000038c 00000000 00000000 00011190 2**0
                CONTENTS, READONLY, DEBUGGING
8 .debug_str    000000498 00000000 00000000 0001151c 2**0
                CONTENTS, READONLY, DEBUGGING
9 .comment      00000011 00000000 00000000 000119b4 2**0
                CONTENTS, READONLY
10 .ARM.attributes 00000033 00000000 00000000 000119c5 2**0
                CONTENTS, READONLY
11 .debug_frame 000002bc 00000000 00000000 000119f8 2**2
```