

```

1  /*
2  * File:   lights.c
3  * Author: ricch
4  *
5  * Created on September 5, 2023, 7:15 PM
6  */
7
8  #include "app.h"
9
10 extern APP_DATA appData;
11
12 //-----//
13 lightManagementProcess
14 void sequenceManagementProcess(void){
15     static int32_t order = 5; //= angleDesired / gear;
16
17     if(appData.isFiveShotsSeqEnable){
18         /* Sequence of 5 pictures is enable */
19         //fiveShotsSeqProcess();
20         //startFiveShotsSeqProcess();
21     }
22     if(appData.isFullImaginSeqEnable){
23         /* Full sequence is enable */
24         switch (appData.valSeq){
25             case 0:
26                 appData.valSeq += fiveShotsSeqProcess();
27                 break;
28             case 1:
29                 setRotationToDo(getMyStepperStruct(), &order);
30                 if(getPerformedSteps(getMyStepperStruct()) == order){
31                     order += 5; // appData.angleBwEachSeq;
32                     appData.valSeq = 0;
33                     appData.seqClock1_ms = 0;
34                     appData.seqClock2_ms = 0;
35                     startFiveShotsSeqProcess();
36                 }
37                 break;
38             }
39         }
40     }
41 }
42
43 //-----//
44 turnOffAllPwrLeds
45 void turnOffAllPwrLeds(void){
46     /* Turn off all power LED */
47     LED1_CMDOff();
48     LED2_CMDOff();
49     LED3_CMDOff();
50     LED4_CMDOff();
51     LED5_CMDOff();
52 }
53
54 //-----//
55 startFiveShotsSequence
56 /* Start a sequence for 5 shots */
57 void startFiveShotsSequence(void){
58     appData.seqClock1_ms = 0;
59     appData.seqClock2_ms = 0;
60     appData.isFiveShotsSeqEnable = true;
61 }
62
63 //-----//

```

```

startFullImagingSequence
67 void startFullImagingSequence(void){
68
69     appData.seqClock1_ms = 0;
70     appData.seqClock2_ms = 0;
71     appData.isFullImaginSeqEnable = true;
72     appData.valSeq = 0;
73     appData.nbrOfShotsPerformed = 0;
74     startFiveShotsSeqProcess();
75 }
76
77 //-----//
simpleShotProcess
78 void startSimpleShotProcess(void){
79
80     appData.seqClock2_ms = 0;
81     DRV_TMR4_Start();
82 }
83
84 void startFiveShotsSeqProcess(void){
85
86     appData.seqClock1_ms = 0;
87     DRV_TMR0_Start();
88 }
89
90 //-----//
imagingSeqProcess
91 /* This function takes 5 pictures with 5 different LEDs */
92 bool fiveShotsSeqProcess(void){
93
94     // if(appData.seqClock1_ms == 0){
95     //     appData.ledId = PWR_LED1;
96     //     startSimpleShotProcess();
97     //
98     // } else if(appData.seqClock1_ms == 1 * appData.timeBetweenPictures){
99     //     appData.ledId = PWR_LED2;
100    //     startSimpleShotProcess();
101    //
102    // } else if(appData.seqClock1_ms == 2 * appData.timeBetweenPictures){
103    //     appData.ledId = PWR_LED3;
104    //     startSimpleShotProcess();
105    //
106    // } else if(appData.seqClock1_ms == 3 * appData.timeBetweenPictures){
107    //     appData.ledId = PWR_LED4;
108    //     startSimpleShotProcess();
109    //
110    // } else if(appData.seqClock1_ms == 4 * appData.timeBetweenPictures){
111    //     appData.ledId = PWR_LED5;
112    //     startSimpleShotProcess();
113    // }
114    // if(appData.seqClock1_ms >= 5 * appData.timeBetweenPictures){
115    //
116    //     appData.seqClock1_ms = 0;
117    //     appData.seqClock2_ms = 0;
118    //     appData.isFiveShotsSeqEnable = false;
119    //     return 1;
120    // }
121    // return 0;
122 }
123
124
125 //-----//
setLighIntensity
126 void setLightIntensity(int32_t *lightIntensity){
127
128     // Limit values to avoid problems
129     if(*lightIntensity < LIGHT_INTENSITY_MIN) *lightIntensity
130         = LIGHT_INTENSITY_MIN;
131     if(*lightIntensity > LIGHT_INTENSITY_MAX) *lightIntensity

```

```

132         = LIGHT_INTENSITY_MAX;
133
134     /* 25 = 2500 / 100 */
135     appData.lightIntensity = *lightIntensity * 25;
136     PLIB_MCPWM_ChannelPrimaryDutyCycleSet(MCPWM_ID_0, PWM_DIM_CH, appData.lightIntensity
    );
137 }
138 int32_t getLightIntensity(void){
139
140     return appData.lightIntensity / 25;
141 }
142
143 //-----//
144 setTimeBwPictures
145 void setTimeBwPictures(int32_t *timeBwPictures){
146
147     int32_t time_bw_pictures_min = appData.exposureDuration +
148         3 * MARGIN_LED_DELAY;
149     // Limit values to avoid problems
150     if(*timeBwPictures < time_bw_pictures_min) *timeBwPictures
151         = time_bw_pictures_min;
152     if(*timeBwPictures > TIME_BW_PICTURES_MAX) *timeBwPictures
153         = TIME_BW_PICTURES_MAX;
154
155     appData.timeBetweenPictures = *timeBwPictures;
156 }
157 int32_t getTimeBwPictures(void){
158
159     return appData.timeBetweenPictures;
160 }
161 //-----//
162 setExposureTime
163 void setExposureTime(int32_t *exposureTime){
164
165     // Limit values to avoid problems
166     if(*exposureTime < EXPOSURE_TIME_MIN) *exposureTime = EXPOSURE_TIME_MIN;
167     if(*exposureTime > EXPOSURE_TIME_MAX) *exposureTime = EXPOSURE_TIME_MAX;
168
169     appData.exposureDuration = *exposureTime;
170 }
171 int32_t getExposureTime(void){
172
173     return appData.exposureDuration;
174 }

```