```c
/*
 * File:   menu.c
 * Author: ricch
 *
 * Created on August 30, 2023, 10:15 AM
 */

#include "menu.h"
#include "lcd_spi.h"
#include "stdio.h"
#include "pec12.h"
#include "stepperDriver.h"


MENU menu;
extern APP_DATA appData;
bool isInModifMode = 0;
bool isFirstDataProcessPass = false;


void initMenuParam(){

    menu.menuPage = 0;
    menu.menuSize = 2;
    menu.menuState = MAIN_MENU;
}


//_____________________________________________________________________________//
menuManagementProcess
void menuManagementProcess(void){

    static int32_t pec12RotationValue = 0;

    /* Get PEC12 increments or decrements if there are */
    int incrOrDecr = getPec12IncrOrDecr();
    //-------------------------------------------------------------------------//
    isInModifMode == true
    if(isInModifMode){

        pec12RotationValue += incrOrDecr;
        menuDataProcess(&pec12RotationValue, getMyStepperStruct());
        menuPrintProcess(getMyStepperStruct());

        /* PEC12 switch pressed */
        if(getPec12SwitchEvent()){

            /* Leave modification mode */
            isInModifMode = false;
            /* Put the cursor on the first line */
            pec12RotationValue = 0;
        }
    }
    //-------------------------------------------------------------------------//
    isInModifMode == false
    else if(isInModifMode == false){

        pec12RotationValue += incrOrDecr;

        if(pec12RotationValue > menu.menuSize) pec12RotationValue =
                menu.menuSize;
        else if(pec12RotationValue < 0) pec12RotationValue = 0;

        if(pec12RotationValue <= 3){

            menu.menuPage = 0;
            menuPrintProcess(getMyStepperStruct());
            printCursor(pec12RotationValue);
        }
```

```
67          else{
68
69              menu.menuPage = 1;
70              menuPrintProcess(getMyStepperStruct());
71              printCursor(pec12RotationValue - 4);
72          }
73
74          /* PEC12 switch pressed */
75          if(getPec12SwitchEvent()){
76
77              menuActionProcess(pec12RotationValue);
78              menuDataProcess(&pec12RotationValue, getMyStepperStruct());
79              menuPrintProcess(getMyStepperStruct());
80
81              /* Put the cursor on the first line */
82  //            pec12RotationValue = 0;
83          }
84      }
85      if(getSwitchEvent()){
86
87          //startFiveShotsSequence();
88  //        startFullImagingSequence();
89
90          switch (menu.menuState){
91
92              case MANUAL_MODE_MENU:
93                  /* Start a sequence of 5 pictures */
94                  //startFiveShotsSequence();
95                  startFiveShotsSeqProcess();
96                  break;
97
98              default:
99                  break;
100         }
101     }
102 }
103
104
105 //_____________________________________________________________________//
106 void menuActionProcess(int32_t pec12RotationValue){
107
108     /* Menu action switch */
109     if(isInModifMode == false){
110         switch(menu.menuState){
111
112             //-------------------------------------------------------------// Main
                menu
113             case MAIN_MENU:
114
115                 switch(pec12RotationValue){
116
117                     case CAPTURE_MODE_SEL:
118                         menu.menuState = CAPTURE_MODE_MENU;
119                         menu.menuSize = 2;
120                         break;
121
122                     case SETTINGS_SEL:
123                         menu.menuState = SETTINGS_MENU;
124                         menu.menuSize = 5;
125                         break;
126
127                     case ABOUT_SEL:
128                         menu.menuState = ABOUT_MENU;
129                         menu.menuSize = 0;
130                         break;
131                 }
132                 break;
133
134             //-------------------------------------------------------------// Main
```

```
                menu -> Choice menu
135             case CAPTURE_MODE_MENU:
136
137                 switch(pec12RotationValue){
138
139                     case RETURN_SEL:
140                         menu.menuState = MAIN_MENU;
141                         menu.menuSize = 2;
142                         break;
143
144                     case MANUAL_MODE_SEL:
145                         menu.menuState = MANUAL_MODE_MENU;
146                         menu.menuSize = 2;
147                         break;
148
149                     case AUTOMATIC_MODE_SEL:
150                         menu.menuState = AUTOMATIC_MODE_MENU;
151                         menu.menuSize = 1;
152                         break;
153                 }
154                 break;
155
156             //-----------------------------------------------------------------// Main
                menu -> Choice menu -> Manual Mode
157             case MANUAL_MODE_MENU:
158                 switch(pec12RotationValue){
159
160                     case RETURN_SEL:
161                         menu.menuState = CAPTURE_MODE_MENU;
162                         menu.menuSize = 2;
163                         break;
164
165                     case AUTO_HOME_SEL:
166                         menu.menuState = AUTO_HOME_MENU;
167                         menu.menuSize = 1;
168                         break;
169
170                     case ANGLE_SEL:
171                         menu.modifState = ANGLE_MODIF;
172                         isInModifMode = true;
173                         isFirstDataProcessPass = true;
174                         break;
175                 }
176                 break;
177
178             //-----------------------------------------------------------------// Main
                menu -> Auto menu
179             case AUTOMATIC_MODE_MENU:
180                 switch(pec12RotationValue){
181
182                     case RETURN_SEL:
183                         menu.menuState = CAPTURE_MODE_MENU;
184                         menu.menuSize = 2;
185                         break;
186
187                     case AUTOMATIC_MODE_START_SEL:
188                         menu.modifState = AUTOMATIC_MODE_START;
189                         isInModifMode = true;
190                         isFirstDataProcessPass = true;
191                         break;
192                 }
193                 break;
194
195             //-----------------------------------------------------------------// Main
                menu -> Manual menu -> Auto home
196             case AUTO_HOME_MENU:
197                     switch(pec12RotationValue){
198
199                     case RETURN_SEL:
```

```
                        menu.menuState = MANUAL_MODE_MENU;
                        menu.menuSize = 2;
                        break;

                    case AUTO_HOME_START_SEL:
                        menu.modifState = AUTO_HOME_START;
                        isInModifMode = true;
                        isFirstDataProcessPass = true;
                        break;
                    }
                break;

                //returnToHome(); PEUT ETRE METTRE AILLEUR
                break;

            //-----------------------------------------------------------------// Main menu -> Settings menu
            case SETTINGS_MENU:

                switch(pec12RotationValue){

                    case RETURN_SEL:
                        menu.menuState = MAIN_MENU;
                        menu.menuSize = 2;
                        break;

                    case MOTOR_SEL:
                        menu.menuState = MOTOR_MENU;
                        menu.menuSize = 4;
                        break;

                    case LEDS_SEL:
                        menu.menuState = LIGHT_MENU;
                        menu.menuSize = 1;
                        break;

                    case BACKLIGHT_SEL:
                        menu.menuState = BACKLIGHT_MENU;
                        menu.menuSize = 1;
                        break;

                    case CAMERA_SEL:
                        menu.menuState = CAMERA_MENU;
                        menu.menuSize = 3;
                        break;

                    case SAVE_DATA_SEL:
                        menu.menuState = SAVE_DATA_MENU;
                        menu.menuSize = 1;
                        break;

                }
                break;

            //-----------------------------------------------------------------// Main menu -> Settings menu -> Motor menu
            case MOTOR_MENU:
                switch(pec12RotationValue){

                    case RETURN_SEL:
                        menu.menuState = SETTINGS_MENU;
                        menu.menuSize = 5;
                        break;

                    case SPEED_SEL:
                        menu.modifState = SPEED_MODIF;
                        isInModifMode = true;
                        isFirstDataProcessPass = true;
                        break;
```

```
267
268                     case GEAR_SEL:
269                         menu.modifState = GEAR_MODIF;
270                         isInModifMode = true;
271                         isFirstDataProcessPass = true;
272                         break;
273
274                     case STEP_PER_TURN_SEL:
275                         menu.modifState = STEP_PER_TURN_MODIF;
276                         isInModifMode = true;
277                         isFirstDataProcessPass = true;
278                         break;
279
280                     case POWER_SEL:
281                         menu.modifState = POWER_MODIF;
282                         isInModifMode = true;
283                         isFirstDataProcessPass = true;
284                         break;
285                 }
286                 break;
287
288             //-------------------------------------------------------------// Main menu -> Settings menu -> Light menu
289             case LIGHT_MENU:
290                 switch(pec12RotationValue){
291
292                     case RETURN_SEL:
293                         menu.menuState = SETTINGS_MENU;
294                         menu.menuSize = 5;
295                         break;
296
297                     case LIGHT_INTENSITY_SEL:
298                         menu.modifState = LIGHT_INTENSITY_MODIF;
299                         isInModifMode = true;
300                         isFirstDataProcessPass = true;
301                         break;
302
303 //                    case LIGHT_TIME_SEL: // <--- in camera param
304 //                        menu.modifState = LIGHT_TIME_MODIF;
305 //                        isInModifMode = true;
306 //                        isFirstDataProcessPass = true;
307 //                        break;
308                 }
309                 break;
310
311             //-------------------------------------------------------------// Main menu -> Settings menu -> Back-light menu
312             case BACKLIGHT_MENU:
313                 switch(pec12RotationValue){
314
315                     case RETURN_SEL:
316                         menu.menuState = SETTINGS_MENU;
317                         menu.menuSize = 5;
318                         break;
319
320                     case LIGHT_INTENSITY_SEL:
321                         menu.modifState = BL_INTENSITY_MODIF;
322                         isInModifMode = true;
323                         isFirstDataProcessPass = true;
324                         break;
325                 }
326                 break;
327
328             //-------------------------------------------------------------// Main menu -> Settings menu -> Camera
329             case CAMERA_MENU:
330                 switch(pec12RotationValue){
331
332                     case RETURN_SEL:
```

```
                        menu.menuState = SETTINGS_MENU;
                        menu.menuSize = 5;
                        break;

                    case EXPOSURE_TIME_SEL:
                        menu.modifState = EXPOSURE_TIME_MODIF;
                        isInModifMode = true;
                        break;

                    case TIME_BW_PICTURES_SEL:
                        menu.modifState = TIME_BW_PICTURES_MODIF;
                        isInModifMode = true;
                        break;
                }
                isFirstDataProcessPass = true;
                break;

            //-----------------------------------------------------------------// Main menu -> Settings menu -> Save data
            case SAVE_DATA_MENU:
                switch(pec12RotationValue){

                    case RETURN_SEL:
                        menu.menuState = SETTINGS_MENU;
                        menu.menuSize = 5;
                        break;

                    case SAVE_DATA_SEL - 4:
                        menu.modifState = SAVE_DATA_START;
                        isInModifMode = true;
                        break;
                }
                isFirstDataProcessPass = true;
                break;

            //-----------------------------------------------------------------// Main menu -> About menu
            case ABOUT_MENU:

                switch(pec12RotationValue){

                    case RETURN_SEL:
                        menu.menuState = MAIN_MENU;
                        menu.menuSize = 2;
                        break;
                }
                break;

            default:
                break;
        }
    }
}

//_____________________________________________________________________//
void menuDataProcess(int32_t *pec12RotationValue, STEPPER_DATA *pStepperData){

    /* Data action switch */
    if(isInModifMode){
        switch(menu.modifState){

            //-----------------------------------------------------------//
            ANGLE_MODIF
            case ANGLE_MODIF:
                if(isFirstDataProcessPass){

                    isFirstDataProcessPass = false;
                    /* A TESTER ET VALIDER, PERTE DE PAS POSSIBLE */
                    *pec12RotationValue = getRotationToDo(pStepperData);
```

```
399                 }
400                 setRotationToDo(pStepperData, pec12RotationValue);
401                 break;
402 
403             //-----------------------------------------------------------------//
                SPEED_MODIF
404             case SPEED_MODIF:
405                 if(isFirstDataProcessPass){
406 
407                     isFirstDataProcessPass = false;
408                     *pec12RotationValue = getSpeed(pStepperData);
409                 }
410                 setSpeed(pStepperData, pec12RotationValue);
411                 break;
412 
413             //-----------------------------------------------------------------//
                GEAR_MODIF
414             case GEAR_MODIF:
415                 if(isFirstDataProcessPass){
416 
417                     isFirstDataProcessPass = false;
418                     *pec12RotationValue = getGearReduction(pStepperData);
419                 }
420                 setGearReduction(pStepperData, pec12RotationValue);
421                 break;
422 
423             //-----------------------------------------------------------------//
                STEP_PER_TURN_MODIF
424             case STEP_PER_TURN_MODIF :
425                 if(isFirstDataProcessPass){
426 
427                     isFirstDataProcessPass = false;
428                     *pec12RotationValue = getAnglePerStep(pStepperData);
429                 }
430                 setAnglePerStep(pStepperData, pec12RotationValue);
431                 break;
432 
433             //-----------------------------------------------------------------//
                POWER_MODIF
434             case POWER_MODIF:
435                 if(isFirstDataProcessPass){
436 
437                     isFirstDataProcessPass = false;
438                     *pec12RotationValue = getStepperPower(pStepperData);
439                 }
440                 setStepperPower(pStepperData, (uint16_t*)pec12RotationValue); ///
                    ????_____Dwaf-ad-***
441                 break;
442 
443             //-----------------------------------------------------------------//
                BL_INTENSITY_MODIF
444             case BL_INTENSITY_MODIF :
445                 if(isFirstDataProcessPass){
446 
447                     isFirstDataProcessPass = false;
448                     *pec12RotationValue = getBlIntensity();
449                 }
450                 setBlIntensity(pec12RotationValue);
451                 break;
452 
453             //-----------------------------------------------------------------//
                LIGHT_INTENSITY_MODIF
454             case LIGHT_INTENSITY_MODIF:
455                 if(isFirstDataProcessPass){
456 
457                     isFirstDataProcessPass = false;
458                     *pec12RotationValue = getLightIntensity();
459                 }
460                 setLightIntensity(pec12RotationValue);
```

```
461                 break;
462
463             //-----------------------------------------------------------------//
                EXPOSURE_TIME_MODIF
464             case EXPOSURE_TIME_MODIF:
465                 if(isFirstDataProcessPass){
466
467                     isFirstDataProcessPass = false;
468                     *pec12RotationValue = getExposureTime();
469                 }
470                 setExposureTime(pec12RotationValue);
471                 break;
472
473             //-----------------------------------------------------------------//
                TIME_BW_PICTURES_MODIF
474             case TIME_BW_PICTURES_MODIF:
475                 if(isFirstDataProcessPass){
476
477                     isFirstDataProcessPass = false;
478                     *pec12RotationValue = getTimeBwPictures();
479                 }
480                 setTimeBwPictures(pec12RotationValue);
481                 break;
482
483             //-----------------------------------------------------------------//
                SAVE_DATA_START
484             case SAVE_DATA_START:
485                 if(isFirstDataProcessPass){
486
487                     isFirstDataProcessPass = false;
488 //                    isInModifMode = false; // AFFICHER .. ECRAN
489                     saveDataInEeprom(pStepperData);
490                     /* Once the data are saved, back to previous menu */
491                     isInModifMode = false;
492                     menu.menuState = SETTINGS_MENU;
493                     menu.menuSize = 5;
494                 }
495                 break;
496
497             //-----------------------------------------------------------------//
                AUTO_HOME_START
498             case AUTO_HOME_START:
499                 if(isFirstDataProcessPass){
500
501                     isFirstDataProcessPass = false;
502                     /* Start the auto home seq. */
503                     startAutoHome(pStepperData);
504                     /* Once auto home seq. is started, back to previous menu */
505                     isInModifMode = false;
506                     menu.menuState = MANUAL_MODE_MENU;
507                     menu.menuSize = 2;
508                 }
509                 break;
510
511             //-----------------------------------------------------------------//
                AUTOMATIC_MODE_START
512             case AUTOMATIC_MODE_START:
513                 if(isFirstDataProcessPass){
514
515                     isFirstDataProcessPass = false;
516                     /* Start the auto home seq. */
517                     startFullImagingSequence();
518                     /* Once auto home seq. is started, back to previous menu */
519                     isInModifMode = false;
520                     menu.menuState = AUTOMATIC_MODE_MENU;
521                     menu.menuSize = 1;
522                 }
523                 break;
524         }
```

```
    }
}



//_____________________________________________________________________________//
void menuPrintProcess(STEPPER_DATA *pStepperData){
    /* Print switch */
    switch(menu.menuState){

        case MAIN_MENU:
            printMainMenu();
            break;

        case SETTINGS_MENU:
            switch (menu.menuPage){
                case 0: printParameterMenuPage0();
                    break;
                case 1: printParameterMenuPage1();
                    break;
            }
            break;

        case MOTOR_MENU:
            switch (menu.menuPage){
                case 0: printMotorMenu0(pStepperData);
                    break;
                case 1: printMotorMenu1(pStepperData);
                    break;
            }
            break;

        case LIGHT_MENU:
            printLedsMenu();
            break;

        case BACKLIGHT_MENU:
            printBackLightMenu();
            break;

        case CAMERA_MENU:
            printCameraMenu();
            break;

        case SAVE_DATA_MENU:
            printSaveDataMenu();
            break;

        case CAPTURE_MODE_MENU:
            printChoiceSeqMenu();
            break;

        case MANUAL_MODE_MENU:
            printManualModeMenu(pStepperData);
            break;

        case AUTOMATIC_MODE_MENU:
            printAutoModeMenu(pStepperData);
            break;

        case ABOUT_MENU:
            printAboutMenu();
            break;

        case AUTO_HOME_MENU:
            printAutoHomeMenu();
            break;

```

```
            default:
                break;
        }
    }
    
    
    
    
    
    void printLcdInit(void){
    
        char str[2];
        ClrDisplay();
        DisplayOnOff(DISPLAY_ON); //Disable cursor
        SetPostion(LINE1);
        WriteString("Auto RTI Capt System");
        SetPostion(LINE2);
        WriteString("08-09 2023");
        SetPostion(LINE3);
        WriteString("Meven Ricchieri");
        SetPostion(LINE4);
    
        int i;
        for (i = 0; i < 20; i++){
    
            APP_Delay_ms(75);
            SetPostion(LINE4 + i);
            sprintf(str, "%c", 0xD0);
            WriteString(str);
        }
        APP_Delay_ms(150);
    }
    
    void printMainMenu(void){
    
        ClrDisplay();
        SetPostion(LINE1);
        WriteString("  Capture mode");
        SetPostion(LINE2);
        WriteString("  Settings");
        SetPostion(LINE3);
        WriteString("  About");
        SetPostion(LINE4);
        WriteString("  ");
    }
    
    void printParameterMenuPage0(void){
    
        ClrDisplay();
        SetPostion(LINE1);
        WriteString("  Return");
        SetPostion(LINE2);
        WriteString("  Motor");
        SetPostion(LINE3);
        WriteString("  Power light");
        SetPostion(LINE4);
        WriteString("  Back-light");
    }
    
    void printParameterMenuPage1(void){
    
        ClrDisplay();
        SetPostion(LINE1);
        WriteString("  Camera");
        SetPostion(LINE2);
        WriteString("  Save data");
    }
```

```
663  void printMotorMenu0(STEPPER_DATA *pStepperData){
664
665      char str[21];
666      ClrDisplay();
667      SetPostion(LINE1);
668      WriteString("  Return");
669      SetPostion(LINE2);
670      sprintf(str, "  Speed: %4dsteps/s", pStepperData->stepPerSec);
671      WriteString(str);
672      SetPostion(LINE3);
673      sprintf(str, "  Gear:        1:%3d", pStepperData->gearValue);
674      WriteString(str);
675      SetPostion(LINE4);
676      sprintf(str, "  Step angle: %1.2f%c", pStepperData->anglePerStep, 0x01);
677      WriteString(str);
678  }
679
680  void printMotorMenu1(STEPPER_DATA *pStepperData){
681
682      char str[21];
683      ClrDisplay();
684      SetPostion(LINE1);
685      /* A changer, en Duty pure, ensuite en %%% */
686      sprintf(str, "  Power : %03d", pStepperData->dutyCycleStepper);
687      WriteString(str);
688  }
689
690  void printLedsMenu(void){
691
692      char str[21];
693      ClrDisplay();
694      SetPostion(LINE1);
695      WriteString("  Return");
696      SetPostion(LINE2);
697      /* 0.04 = 100 / 2500 */
698      sprintf(str, "  Intensity : %03.0f%%", ((float)appData.lightIntensity * 0.04));
699      WriteString(str);
700  //    SetPostion(LINE3);
701  //    sprintf(str, "  Light time: %03dms", appData.lightTime);
702  //    WriteString(str);
703  }
704
705  void printChoiceSeqMenu(void){
706
707      ClrDisplay();
708      SetPostion(LINE1);
709      WriteString("  Return");
710      SetPostion(LINE2);
711      WriteString("  Manual mode");
712      SetPostion(LINE3);
713      WriteString("  Auto mode");
714      SetPostion(LINE4);
715      WriteString("  ");
716  }
717
718  void printAboutMenu(void){
719
720      ClrDisplay();
721      SetPostion(LINE1);
722      WriteString("  Return");
723      SetPostion(LINE2);
724      WriteString("  Version 1.0.0");
725      SetPostion(LINE3);
726      WriteString("  Meven Ricchieri");
727      SetPostion(LINE4);
728      WriteString("  08-09 2023");
729  }
730
731  void printManualModeMenu(STEPPER_DATA *pStepperData){
```

```
732
733         char str[21];
734         ClrDisplay();
735         SetPostion(LINE1);
736         WriteString("  Return");
737         SetPostion(LINE2);
738         if(pStepperData->isIndexed == true){
739             sprintf(str, "  Auto home    :%s", "DONE");
740         } else {
741             sprintf(str, "  Auto home     :%s", "NOK");
742         }
743         WriteString(str);
744         SetPostion(LINE3);
745         sprintf(str, "  Des. angle :%03.1f%c", (((float)pStepperData->stepToReach * 1.8)
746                 / pStepperData->gearValue), 0x01);
747     //    sprintf(str, "  Steps      : %05d", stepperData.stepToDoReach);
748         WriteString(str);
749         SetPostion(LINE4);
750         sprintf(str, "  Real angle :%03.1f%c", (((float)pStepperData->performedSteps * 1.8)
751                 / pStepperData->gearValue), 0x01);
752     //    sprintf(str, "  Steps       :%05d", pStepperData->performedStep);
753         WriteString(str);
754     }
755
756     void printAutoModeMenu(STEPPER_DATA *pStepperData){
757
758         char str[21];
759         ClrDisplay();
760         SetPostion(LINE1);
761         WriteString("  Return");
762         SetPostion(LINE2);
763         if(appData.isFullImaginSeqEnable == false) {
764             sprintf(str, "  Start sequence");
765         } else {
766             sprintf(str, "  Sequence is ON");
767         }
768         WriteString(str);
769         SetPostion(LINE3);
770         sprintf(str, "  Pictures:      %03d", appData.nbrOfShotsPerformed);
771         WriteString(str);
772         SetPostion(LINE4);
773         sprintf(str, "  Real angle :%03.1f%c", (((float)pStepperData->performedSteps * 1.8)
774                 / pStepperData->gearValue), 0x01);
775         WriteString(str);
776     }
777
778     void printAutoHomeMenu(void){
779
780         ClrDisplay();
781         SetPostion(LINE1);
782         WriteString("  Return");
783         SetPostion(LINE2);
784         WriteString("  Press to index");
785         SetPostion(LINE3);
786         WriteString("  ");
787         SetPostion(LINE4);
788         WriteString("  ");
789     }
790
791     void printBackLightMenu(void){
792
793         char str[21];
794         ClrDisplay();
795         SetPostion(LINE1);
796         WriteString("  Return");
797         SetPostion(LINE2);
798         /* 0.04 = 100 / 2500 */
799         sprintf(str, "  Intensity : %03.0f%%", ((float)appData.backLightIntensitiy * 0.04));
800         WriteString(str);
```

```
801 }
802
803 void printCameraMenu(void){
804
805     char str[21];
806     ClrDisplay();
807     SetPostion(LINE1);
808     WriteString("  Return");
809     SetPostion(LINE2);
810     sprintf(str, "  Expos time: %04dms", appData.exposureDuration);
811     WriteString(str);
812     SetPostion(LINE3);
813     sprintf(str, "  Time bw pic:%04dms", appData.timeBetweenPictures);
814     WriteString(str);
815     SetPostion(LINE4);
816     WriteString("  Trigger : cable"); // <-- or IR but not ready
817 }
818
819 void printSaveDataMenu(){
820
821     ClrDisplay();
822     SetPostion(LINE1);
823     WriteString("  Return");
824     SetPostion(LINE2);
825     WriteString("  Confirm to save");
826     SetPostion(LINE3);
827     WriteString("  ! Old values will ");
828     SetPostion(LINE4);
829     WriteString("  be overwritten ! ");
830 }
831
832
833
834
835 /* Clear the first row all 4 lines */
836 void clearFirstRow(void){
837
838     SetPostion(LINE1);
839     WriteString(" ");
840     SetPostion(LINE2);
841     WriteString(" ");
842     SetPostion(LINE3);
843     WriteString(" ");
844     SetPostion(LINE4);
845     WriteString(" ");
846 }
847
848 /* Print cursor */
849 void printCursor(int32_t cursor){
850
851     char str[2];
852     clearFirstRow();
853     SetPostion(cursor * 0x20);
854     sprintf(str, "%c", RIGHT_ARROW);
855     WriteString(str);
856 }
857
858
859
860 //-------------------------------------------------------------------------//
    saveDataInEeprom
861 bool saveDataInEeprom(STEPPER_DATA *pStepperData){
862
863     DATA_IN_EEPROM dataToSaveInEeprom;
864
865     /* Set the structure value for saving in EEPROM */
866     dataToSaveInEeprom.stepPerSec   = pStepperData->stepPerSec;
867     dataToSaveInEeprom.stepPerTurn  = pStepperData->stepPerTurn;
868     dataToSaveInEeprom.gearValue    = pStepperData->gearValue;
```

```
869       dataToSaveInEeprom.anglePerStep = pStepperData->anglePerStep;
870
871       dataToSaveInEeprom.lightIntensity       = appData.lightIntensity;
872       dataToSaveInEeprom.timeBetweenPictures  = appData.timeBetweenPictures;
873       dataToSaveInEeprom.exposureDuration     = appData.exposureDuration;
874
875       dataToSaveInEeprom.backLightIntensitiy  = appData.backLightIntensitiy;
876
877       dataToSaveInEeprom.controlValue = CONTROL_VALUE;
878
879       Init_DataBuff();
880       /* Write in the EEPROM */
881       NVM_WriteBlock((uint32_t*)&dataToSaveInEeprom, sizeof(dataToSaveInEeprom));
882
883       return 0;
884   }
885
886   //-------------------------------------------------------------------------//
      readDataFromEeprom
887   /* Read the parameters from the EEPROM */
888   bool readDataFromEeprom(STEPPER_DATA *pStepperData){
889
890       DATA_IN_EEPROM dataReadFromEeprom;
891
892       Init_DataBuff();
893       /* Read in the EEPROM */
894       NVM_ReadBlock((uint32_t*)&dataReadFromEeprom, sizeof(dataReadFromEeprom));
895
896       /* Check if the control value is already inside the EEPROM */
897       if(dataReadFromEeprom.controlValue == CONTROL_VALUE){
898
899           /* Save data from EEPROM */
900           pStepperData->stepPerSec    = dataReadFromEeprom.stepPerSec;
901           pStepperData->stepPerTurn   = dataReadFromEeprom.stepPerTurn;
902           pStepperData->gearValue     = dataReadFromEeprom.gearValue;
903           pStepperData->anglePerStep  = dataReadFromEeprom.anglePerStep;
904
905           appData.lightIntensity      = dataReadFromEeprom.lightIntensity;
906           appData.timeBetweenPictures = dataReadFromEeprom.timeBetweenPictures;
907           appData.exposureDuration    = dataReadFromEeprom.exposureDuration;
908
909           appData.backLightIntensitiy = dataReadFromEeprom.backLightIntensitiy;
910
911       } else {
912
913           // SAVE INIT VAL
914           saveDataInEeprom(pStepperData);
915       }
916   }
```