```
//18111C_Master
//-----------------------------------------------------------
// App.c
//-----------------------------------------------------------
//
//
//  Auteur      :
//  Date        :
//  Version     :
//  Modifications : MDS 26.09.2022
//  Description :
//          Application principal de la carte ticketing Master 1811C
//
//
/*-----------------------------------------------------------*/


#include "app.h"
#include "Data_Code.h"
#include "Mc32gest_RS232.h"
#include "Gest_Menu.h"


APP_DATA appData;
APP_DATA appData_Old;
APP_DATA appData_callback;


uint8_t Nb_Student = 0;//eleves actuellement en cours de traitement
uint8_t Nb_Student_max = 0;//nombre max de slave/eleve actuellement enregistrer
bool Message_receive_XBEE;// flag de reception de donnee xbee

/*******************************************************************************
  Function:
    void APP_Initialize ( void )

  Remarks:
    See prototype in app.h.
 */

void APP_Initialize ( void )
{
    appData.state = APP_STATE_INIT;
}
/*******************************************************************************
  Function:
    void APP_UpdateState ( void )

  Remarks:
    See prototype in app.h.
 */

void APP_UpdateState (APP_STATES NewState)
{
    appData.state = NewState;
}

/*******************************************************************************
  Function:
    void APP_Tasks ( void )

  Remarks:
    See prototype in app.h.
 */

void APP_Tasks ( void )
{
    static int Count = 0,I,J;
    int32_t RXSize;
    char trash;
    static uint32_t DataCodeToSend = 0;
    static bool Ticket_Refused = false;
    U_32 RXData;
```

```
74        U_32 ADD_M;
75        U_32 ADD_S;
76        bool Waiting_Answer = false;
77
78        /* Check the application's current state. */
79        switch ( appData.state )
80        {
81            /* Application's initial state. */
82            case APP_STATE_INIT:
83            {
84
85                DRV_TMR0_Stop();// stoper le timer 1
86
87                RF_Init();  // initialisation de l Xbee
88
89                TFT_Init(); // initialisation du TFT
90                Init_Menu(); // initialisation du menu
91                InitFifoComm();// initialisation du fifo
92
93                DRV_TMR1_Start(); //Start le timer 2
94                APP_UpdateState (APP_STATE_GEST_MENU);//changement d'etat
95
96                break;
97            }
98
99            case APP_STATE_GEST_MENU:
100           {
101               //Led_QuestToggle();
102
103                Gest_Menu();//gestion du menu
104
105               break;
106           }
107           case APP_STATE_LINK_XBEE:
108           {
109               //AddTarget.val32 = Students_Info[0].ID;
110               //AddTarget.val32 = Add_Master;
111               DataCodeToSend = ARE_U_LINK;//preparation de l'envoie
112
113               //sauvegarde de l'etat precedent
114               appData_callback.state = appData.state;
115               //changement de l'etat
116               APP_UpdateState(APP_STATE_SEND_ID);
117               //sauvegarder l'etat
118               appData_Old.state = APP_STATE_LINK_XBEE;
119
120
121               break;
122           }
123
124           case APP_STATE_SEND_ID:
125           {
126               if(appData_Old.state == APP_STATE_LINK_XBEE)
127               {
128                   //flag start Link
129                   LINK = false;
130                   //flag adresse slave connu
131                   Add_ON = false;
132                   //envoie du message
133                   SendMessage(Add_Master,NULL,DataCodeToSend);
134                   //flag adresse slave connu
135                   Add_ON = true;
136                   //Changement d etat
137                   APP_UpdateState(APP_STATE_GET_DATA);
138               }
139               else if(appData_Old.state == APP_STATE_PRESENCE)
140               {
141                   //envoie du message
142                   SendMessage(Add_Master,Add_Slave,DataCodeToSend);
143                   //Changement d etat
144                   APP_UpdateState(APP_STATE_GET_DATA);
145                   //incrementation de la list
146                   Nb_In_List++;
```

```
147             }
148             else
149             {
150                 //envoie du message
151                 SendMessage(Add_Master,Add_Slave,DataCodeToSend);
152             }
153 
154             break;
155         }
156         case APP_STATE_GET_DATA:
157         {
158                //reception du message et de l expediteur
159                GetMessage(&ADD_S,&ADD_M,&RXData);
160 
161                //on check que l'expediteur est bien le maitre
162                if((ADD_M.val32 == Add_Master)||(Waiting_Answer))
163                 {
164                    //on check si l'on recois un ticket ou si on attend une
165                    //reponse de l'utilisateur
166                     if ((RXData.val32 == ENVOI_TICKET) || (Waiting_Answer))
167                     {
168                         //start du timer 1
169                         DRV_TMR0_Start();
170                         //on met un flag d'attente de reponse
171                         Waiting_Answer = true ;
172                         //si le bounton Acc a ete appuyer
173                         if(appButtons.But_ACC)
174                         {
175                             //on prepare le data a envoyer
176                             DataCodeToSend = TICKET_ACCEPT;
177                             //Changement d'etat
178                             APP_UpdateState(APP_STATE_SEND_DATA);
179                             //on enleve le flag
180                             Waiting_Answer = false ;
181                             //on arrete le timer 1
182                             DRV_TMR0_Stop();
183 
184                         }
185                         //si le bounton Acc a ete appuyer
186                         else if(appButtons.But_DECL)
187                         {
188                             //on prepare le data a envoyer
189                             DataCodeToSend= TICKET_REFUSE;
190                             //Changement d'etat
191                             APP_UpdateState(APP_STATE_SEND_DATA);
192                             //on enleve le flag
193                             Waiting_Answer = false ;
194                             //on arrete le timer 1
195                             DRV_TMR0_Stop();
196                         }
197 //                        else if(appButtons.But_ESC)
198 //                        {
199 //                            DataCodeToSend= TICKET_RESET;
200 //                            APP_UpdateState(APP_STATE_SEND_DATA);
201 //                            Waiting_Answer = false ;
202 //                            DRV_TMR0_Stop();
203 //                        }
204                     }
205                     // si l'on recois une adresse d'un noouveau slave
206                     if((New_student)&&(!Waiting_Answer))
207                     {
208                         //on enleve le falg de nouveau slave
209                         New_student = false;
210                         //on prepare le data a envoyer
211                         DataCodeToSend= ACK;
212                         //Changement d'etat
213                         APP_UpdateState(APP_STATE_SEND_DATA);
214                     }
215                }
216             break;
217         }
218         case APP_STATE_SEND_DATA:
219         {
```

```
220             //envoie de message au Xbee
221             SendMessage(Add_Master,Add_Slave,DataCodeToSend);
222 
223             if(appData_Old.state == APP_STATE_LINK_XBEE)
224             {
225                 //on retourne au debut de la list
226                 Nb_In_List = 0;
227                 //Changement d'etat
228                 APP_UpdateState(APP_STATE_PRESENCE);
229             }
230             else if (appData_Old.state == APP_STATE_PRESENCE)
231             {
232                 //Changement d'etat
233                 APP_UpdateState(APP_STATE_PRESENCE);
234                 //on incremente la list
235                 Nb_In_List++;
236             }
237             else
238             {
239                 //Changement d'etat
240                 APP_UpdateState(APP_STATE_GET_DATA);
241             }
242             break;
243         }
244         case APP_STATE_RESET:
245         {
246             //TODO
247             break;
248         }
249         case APP_STATE_REFUSE:
250         {
251             //Changement d'etat
252             APP_UpdateState(APP_STATE_SEND_ID);
253             //on prepare le data a envoyer
254             DataCodeToSend = TICKET_REFUSE;
255             //on prepare l'adress a envoyer
256             Add_Slave = Students_Info[Nb_In_List].ID;
257             break;
258         }
259         case APP_STATE_ACCEPT:
260         {
261             //Changement d'etat
262             APP_UpdateState(APP_STATE_SEND_ID);
263             //on prepare le data a envoyer
264             DataCodeToSend = TICKET_ACCEPT;
265             //on prepare l'adress a envoyer
266             Add_Slave = Students_Info[Nb_In_List].ID;
267             //on reduit le nombre max de question
268             Students_Info[Nb_In_List].LimitQues --;
269             break;
270         }
271 
272         case APP_STATE_PRESENCE:
273         {
274             //on prepare le data a envoyer
275             DataCodeToSend = ARE_U_LINK;
276             //on prepare l'adress a envoyer
277             Add_Slave =  Students_Info[Nb_In_List].ID;
278             //si on arrive a la fin de la list
279             if(Nb_In_List == Nb_Student_max)
280             {
281                 //Changement d'etat
282                 APP_UpdateState(APP_STATE_SEND_DATA);
283                 //on retourne dans l etat avant le pulling
284                 appData_Old.state == appData_callback.state;
285             }
286             else
287             {
288                 //Changement d'etat
289                 appData_Old.state == APP_STATE_PRESENCE;
290                 //incrémentation de la list
291                 Nb_In_List++;
292             }
```

```
            break;
        }
        default:
        {
            break;
        }
    }
}


/*****************************************************************************
 End of File
 */

```