```c
/*******************************************************************************
  MPLAB Harmony Application Header File

  Company:
    Microchip Technology Inc.

  File Name:
    app.h

  Summary:
    This header file provides prototypes and definitions for the application.

  Description:
    This header file provides function prototypes and data type definitions for
    the application.  Some of these are required by the system (such as the
    "APP_Initialize" and "APP_Tasks" prototypes) and some of them are only used
    internally by the application (such as the "APP_STATES" definition).  Both
    are defined here for convenience.
*******************************************************************************/

//DOM-IGNORE-BEGIN
/*******************************************************************************
Copyright (c) 2013-2014 released Microchip Technology Inc.  All rights reserved.

Microchip licenses to you the right to use, modify, copy and distribute
Software only when embedded on a Microchip microcontroller or digital signal
controller that is integrated into your product or third party product
(pursuant to the sublicense terms in the accompanying license agreement).

You should refer to the license agreement accompanying this Software for
additional information regarding your rights and obligations.

SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND,
EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF
MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE.
IN NO EVENT SHALL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER
CONTRACT, NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR
OTHER LEGAL EQUITABLE THEORY ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES
INCLUDING BUT NOT LIMITED TO ANY INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR
CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF
SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES
(INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS.
 *******************************************************************************/
//DOM-IGNORE-END

#ifndef _APP_H
#define _APP_H

// *****************************************************************************
// *****************************************************************************
// Section: Included Files
// *****************************************************************************
// *****************************************************************************

#include <stdint.h>
#include <stdbool.h>
#include <stddef.h>
#include <stdlib.h>
#include "system_config.h"
#include "system_definitions.h"

#include "Data_Code.h"


// DOM-IGNORE-BEGIN
#ifdef __cplusplus  // Provide C++ Compatibility

extern "C" {

#endif
// DOM-IGNORE-END
    //todo for SSOP
```

```c
// *****************************************************************************
// *****************************************************************************
// Section: Type Definitions
// *****************************************************************************
// *****************************************************************************
extern char buffReadName[];// Buffer de reception de l'UART
extern uint8_t Name_Receive ;// Flag de récéption
extern uint8_t countCar;// Compteur du nombre de characters d'un nom
extern bool Ticket_Refused;
/* Application states

  Summary:
    Application states enumeration

  Description:
    This enumeration defines the valid application states.  These states
    determine the behavior of the application at various times.
*/

typedef enum
{
    /* Application's state machine's initial state. */
    APP_STATE_INIT=0,
    APP_RETRIEVE_NAME,
    APP_WAIT_FOR_LINK,
    APP_SEND_DATA,
    APP_SEND_ID,
    APP_ERROR,
    APP_WAIT_FOR_ACK,
    APP_WAIT_FOR_TICKET_ACCEPT,
    APP_ACCEPT,
    APP_REFUSED,
    APP_BLOCKED,
    APP_RESET,
    APP_WAIT_FOR_TICKET,

    /* TODO: Define states used by the application state machine. */

} APP_STATES;
typedef struct
{
    /* The application's current state */
    bool Btn_Tickets;
    /* TODO: Define any additional data used by the application. */

} APP_BUTTON;
APP_BUTTON appButtons;

// *****************************************************************************
/* Application Data

  Summary:
    Holds application data

  Description:
    This structure holds the application's data.

  Remarks:
    Application strings and buffers are be defined outside this structure.
 */

typedef struct
{
    /* The application's current state */
    APP_STATES state;
    /* TODO: Define any additional data used by the application. */

} APP_DATA;


// *****************************************************************************
// *****************************************************************************
```

```c
147    // Section: Application Callback Routines
148    // *****************************************************************************
149    // *****************************************************************************
150    /* These routines are called by drivers when certain events occur.
151    */
152
153    // *****************************************************************************
154    // *****************************************************************************
155    // Section: Application Initialization and State Machine Functions
156    // *****************************************************************************
157    // *****************************************************************************
158
159    /*****************************************************************************
160      Function:
161        void APP_Initialize ( void )
162
163      Summary:
164         MPLAB Harmony application initialization routine.
165
166      Description:
167        This function initializes the Harmony application.  It places the
168        application in its initial state and prepares it to run so that its
169        APP_Tasks function can be called.
170
171      Precondition:
172        All other system initialization routines should be called before calling
173        this routine (in "SYS_Initialize").
174
175      Parameters:
176        None.
177
178      Returns:
179        None.
180
181      Example:
182        <code>
183        APP_Initialize();
184        </code>
185
186      Remarks:
187        This routine must be called from the SYS_Initialize function.
188    */
189
190    void APP_Initialize ( void );
191    void Blink_LED_ACC (void);
192    void APP_UpdateState (APP_STATES NewState);
193    void ALL_LED_ON (void);
194    void ALL_LED_OFF (void);
195
196    /*****************************************************************************
197      Function:
198        void APP_Tasks ( void )
199
200      Summary:
201        MPLAB Harmony Demo application tasks function
202
203      Description:
204        This routine is the Harmony Demo application's tasks function.  It
205        defines the application's state machine and core logic.
206
207      Precondition:
208        The system and application initialization ("SYS_Initialize") should be
209        called before calling this.
210
211      Parameters:
212        None.
213
214      Returns:
215        None.
216
217      Example:
218        <code>
219        APP_Tasks();
```

```
        </code>

  Remarks:
    This routine must be called from SYS_Tasks() routine.
 */

void APP_Tasks( void );


#endif /* _APP_H */

//DOM-IGNORE-BEGIN
#ifdef __cplusplus
}
#endif
//DOM-IGNORE-END

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