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

  Company:
    Microchip Technology Inc.

  File Name:
    app.h

*******************************************************************************/

//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 "bno055.h"

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

extern "C" {

#endif
// DOM-IGNORE-END


#define TIME_OUT                    80000000U
#define BTN_HOLD_SHUTDOWN_x10ms     200
#define NB_MEASURES                 2

#define ACCEL_ACTIV_DETECT_msq      0.3
#define T_CONFIG_TIMEOUT            20
#define T_INACTIVE_PERIOD_DEFAULT   20UL
#define T_INTERVAL_GNSS_DEFAULT     5000UL
#define T_INTERVAL_IMU_DEFAULT      500UL
#define LED_STATE_DEFAULT           (uint8_t)1

```

```
74  #define LED_PERIOD                  5
75
76  #define CHAR_READ_BUFFER_SIZE       30
77
78  #define G                 9.81
79
80
81  // *****************************************************************************
82  // *****************************************************************************
83  // Section: Type Definitions
84  // *****************************************************************************
85  // *****************************************************************************
86  typedef struct {
87      s32 comres;
88      bool flagMeasReady;
89      uint8_t flagImportantMeas;
90      struct bno055_gravity_double_t gravity;
91      struct bno055_linear_accel_double_t linear_accel;
92      struct bno055_euler_double_t euler;
93      struct bno055_gyro_double_t gyro;
94      struct bno055_mag_double_t mag;
95      struct bno055_quaternion_t quaternion;
96      unsigned long time;
97      unsigned long l_time;
98      uint16_t d_time;
99  }s_bno055_data;
100 // *****************************************************************************
101 /* Application states
102
103   Summary:
104     Application states enumeration
105
106   Description:
107     This enumeration defines the valid application states.  These states
108     determine the behavior of the application at various times.
109 */
110
111 typedef enum
112 {
113     /* Application's state machine's initial state. */
114     APP_STATE_INIT=0,
115     APP_STATE_CONFIG,
116     APP_STATE_LOGGING,
117     APP_STATE_FLAG_MEAS,
118     APP_STATE_COMM_LIVE_GNSS,
119     APP_STATE_COMM_LIVE_IMU,
120     APP_STATE_CONFIGURATE_BBX,
121     APP_STATE_SHUTDOWN
122     /* TODO: Define states used by the application state machine. */
123
124 } APP_STATES;
125
126
127 // *****************************************************************************
128 /* Application Data
129
130   Summary:
131     Holds application data
132
133   Description:
134     This structure holds the application's data.
135
136   Remarks:
137     Application strings and buffers are be defined outside this structure.
138  */
139
140 typedef struct
141 {
142     /* The application's current state */
143     APP_STATES state;
144
145     bool ledState;
146
```

```
147         /* TODO: Define any additional data used by the application. */
148
149     } APP_DATA;
150
151     typedef struct
152     {
153         /* DELAY DATA */
154         bool tmrTickFlag;
155         unsigned long delayCnt;
156
157         /* MEASURES DATA */
158         unsigned long measCnt[NB_MEASURES];
159         unsigned long ltime[NB_MEASURES];
160         bool measTodo[NB_MEASURES];
161         unsigned long measPeriod[NB_MEASURES];
162
163         unsigned long inactiveCnt;
164         uint32_t inactivePeriod;
165
166         /* DISPLAY DATA */
167         uint32_t ledCnt;
168
169         /* BUTTON DATA */
170         bool flagCntBtnPressed;
171         uint32_t cntBtnPressed;
172     }TIMER_DATA;
173
174     /* Measures index */
175     enum measure{BNO055_idx, GNSS_idx};
176
177     // *****************************************************************************
178     // *****************************************************************************
179     // Section: Application Callback Routines
180     // *****************************************************************************
181     // *****************************************************************************
182     /* These routines are called by drivers when certain events occur.
183     */
184
185     // *****************************************************************************
186     // *****************************************************************************
187     // Section: Application Initialization and State Machine Functions
188     // *****************************************************************************
189     // *****************************************************************************
190
191     /*******************************************************************************
192       Function:
193         void APP_Initialize ( void )
194
195       Summary:
196          MPLAB Harmony application initialization routine.
197
198       Description:
199         This function initializes the Harmony application.  It places the
200         application in its initial state and prepares it to run so that its
201         APP_Tasks function can be called.
202
203       Precondition:
204         All other system initialization routines should be called before calling
205         this routine (in "SYS_Initialize").
206
207       Parameters:
208         None.
209
210       Returns:
211         None.
212
213       Example:
214         <code>
215         APP_Initialize();
216         </code>
217
218       Remarks:
219         This routine must be called from the SYS_Initialize function.
```

```
220   */
221
222   void APP_Initialize ( void );
223
224
225   /*******************************************************************************
226     Function:
227       void APP_Tasks ( void )
228
229     Summary:
230       MPLAB Harmony Demo application tasks function
231
232     Description:
233       This routine is the Harmony Demo application's tasks function.  It
234       defines the application's state machine and core logic.
235
236     Precondition:
237       The system and application initialization ("SYS_Initialize") should be
238       called before calling this.
239
240     Parameters:
241       None.
242
243     Returns:
244       None.
245
246     Example:
247       <code>
248       APP_Tasks();
249       </code>
250
251     Remarks:
252       This routine must be called from SYS_Tasks() routine.
253    */
254
255   void APP_Tasks( void );
256
257   // CALLBACKS
258   void delayTimer_callback( void );
259   void stateTimer_callback( void );
260
261   void appStateSet( APP_STATES newState );
262
263   #endif /* _APP_H */
264
265   //DOM-IGNORE-BEGIN
266   #ifdef __cplusplus
267   }
268   #endif
269   //DOM-IGNORE-END
270
271   /*******************************************************************************
272    End of File
273    */
274
275
```