```
/****************************************************************************
 Module
     ES_Configure.h
 Description
     This file contains macro definitions that are edited by the user to
     adapt the Events and Services framework to a particular application.
 Notes

 History
 When           Who     What/Why
 -------------- ---     --------
 01/15/12 10:03 jec      started coding
****************************************************************************/

#ifndef CONFIGURE_H
#define CONFIGURE_H

/****************************************************************************/
// The maximum number of services sets an upper bound on the number of
// services that the framework will handle. Reasonable values are 8 and 16
// HOWEVER: at this time only a value of 8 is supported.
#define MAX_NUM_SERVICES 8

/****************************************************************************/
// This macro determines that nuber of services that are *actually* used in
// a particular application. It will vary in value from 1 to MAX_NUM_SERVICES
#define NUM_SERVICES 3

/****************************************************************************/
// These are the definitions for Service 0, the lowest priority service
// every Events and Services application must have a Service 0. Further
// services are added in numeric sequence (1,2,3,...) with increasing
// priorities
// the header file with the public fuction prototypes
#define SERV_0_HEADER "CheckValidSM.h"
// the name of the Init function
#define SERV_0_INIT InitCheckValidSM
// the name of the run function
#define SERV_0_RUN RunCheckValidSM
// How big should this service's Queue be?
#define SERV_0_QUEUE_SIZE 6

/****************************************************************************/
// The following sections are used to define the parameters for each of the
// services. You only need to fill out as many as the number of services
// defined by NUM_SERVICES
/****************************************************************************/
// These are the definitions for Service 1
#if NUM_SERVICES > 1
// the header file with the public fuction prototypes
#define SERV_1_HEADER "TransmitXbeeSM.h"
// the name of the Init function
#define SERV_1_INIT InitTransmitXbeeSM
```

```
// the name of the run function
#define SERV_1_RUN RunTransmitXbeeSM
// How big should this services Queue be?
#define SERV_1_QUEUE_SIZE 4
#endif

/****************************************************************************/
// These are the definitions for Service 2
#if NUM_SERVICES > 2
// the header file with the public fuction prototypes
#define SERV_2_HEADER "InterpretSM.h"
// the name of the Init function
#define SERV_2_INIT InitInterpretSM
// the name of the run function
#define SERV_2_RUN RunInterpretSM
// How big should this services Queue be?
#define SERV_2_QUEUE_SIZE 8
#endif

/****************************************************************************/
// These are the definitions for Service 3
#if NUM_SERVICES > 3
// the header file with the public fuction prototypes
#define SERV_3_HEADER "TestService.h"
// the name of the Init function
#define SERV_3_INIT TestServiceInit
// the name of the run function
#define SERV_3_RUN TestServiceRun
// How big should this services Queue be?
#define SERV_3_QUEUE_SIZE 1
#endif

/****************************************************************************/
// These are the definitions for Service 4
#if NUM_SERVICES > 4
// the header file with the public fuction prototypes
#define SERV_4_HEADER "TestService.h"
// the name of the Init function
#define SERV_4_INIT TestServiceInit
// the name of the run function
#define SERV_4_RUN TestServiceRun
// How big should this services Queue be?
#define SERV_4_QUEUE_SIZE 1
#endif

/****************************************************************************/
// These are the definitions for Service 5
#if NUM_SERVICES > 5
// the header file with the public fuction prototypes
#define SERV_5_HEADER "TestService.h"
// the name of the Init function
#define SERV_5_INIT TestServiceInit
// the name of the run function
```

```
#define SERV_5_RUN TestServiceRun
// How big should this services Queue be?
#define SERV_5_QUEUE_SIZE 1
#endif

/****************************************************************************/
// These are the definitions for Service 6
#if NUM_SERVICES > 6
// the header file with the public fuction prototypes
#define SERV_6_HEADER "TestService.h"
// the name of the Init function
#define SERV_6_INIT TestServiceInit
// the name of the run function
#define SERV_6_RUN TestServiceRun
// How big should this services Queue be?
#define SERV_6_QUEUE_SIZE 1
#endif

/****************************************************************************/
// These are the definitions for Service 7
#if NUM_SERVICES > 7
// the header file with the public fuction prototypes
#define SERV_7_HEADER "TestService.h"
// the name of the Init function
#define SERV_7_INIT TestServiceInit
// the name of the run function
#define SERV_7_RUN TestServiceRun
// How big should this services Queue be?
#define SERV_7_QUEUE_SIZE 1
#endif

/****************************************************************************/
// the name of the posting function that you want executed when a new
// keystroke is detected.
// The default initialization distributes keystrokes to all state machines
#define POST_KEY_FUNC PostCheckValidSM //ES_PostAll

/****************************************************************************/
// Name/define the events of interest
// Universal events occupy the lowest entries, followed by user-defined events
typedef enum {  ES_NO_EVENT = 0,
                ES_ERROR,  /* used to indicate an error from the service */
                ES_INIT,   /* used to transition from initial pseudo-state */
                ES_NEW_KEY, /* signals a new key received from terminal */
                        ES_TIMEOUT, /* signals that the timer has expired */
                /* User-defined events start here */
                ES_XB_Msg_Rec,
                ES_BALLOON_POPPED,
                ES_CONTROLLER_FOUND,
                ES_COMMAND_REC,
                ES_TRMT_REG_EMPTY,
                ES_FRAME_REC,
                ES_POST_DIR_BYTE,
```

```
                ES_POST_FLAG_BYTE} ES_EventTyp_t ;

/****************************************************************************/
// These are the definitions for the Distribution lists. Each definition
// should be a comma seperated list of post functions to indicate which
// services are on that distribution list.
#define NUM_DIST_LISTS 0
#if NUM_DIST_LISTS > 0
#define DIST_LIST0 PostTemplateSM
#endif
#if NUM_DIST_LISTS > 1
#define DIST_LIST1 PostTemplateSM
#endif
#if NUM_DIST_LISTS > 2
#define DIST_LIST2 PostTemplateFSM
#endif
#if NUM_DIST_LISTS > 3
#define DIST_LIST3 PostTemplateFSM
#endif
#if NUM_DIST_LISTS > 4
#define DIST_LIST4 PostTemplateFSM
#endif
#if NUM_DIST_LISTS > 5
#define DIST_LIST5 PostTemplateFSM
#endif
#if NUM_DIST_LISTS > 6
#define DIST_LIST6 PostTemplateFSM
#endif
#if NUM_DIST_LISTS > 7
#define DIST_LIST7 PostTemplateFSM
#endif

/****************************************************************************/
// This are the name of the Event checking funcion header file.
#define EVENT_CHECK_HEADER "EventCheckers.h"

/****************************************************************************/
// This is the list of event checking functions
#define EVENT_CHECK_LIST TransmitBufferEmpty,Check4BalloonPopped

/****************************************************************************/
// These are the definitions for the post functions to be executed when the
// correspnding timer expires. All 8 must be defined. If you are not using
// a timers, then you can use TIMER_UNUSED
#define TIMER_UNUSED ((pPostFunc)0)
#define TIMER0_RESP_FUNC PostCheckValidSM
#define TIMER1_RESP_FUNC PostInterpretSM
#define TIMER2_RESP_FUNC TIMER_UNUSED
#define TIMER3_RESP_FUNC TIMER_UNUSED
#define TIMER4_RESP_FUNC TIMER_UNUSED
#define TIMER5_RESP_FUNC TIMER_UNUSED
#define TIMER6_RESP_FUNC TIMER_UNUSED
#define TIMER7_RESP_FUNC TIMER_UNUSED
```

```
/*****************************************************************************/
// Give the timer numbers symbolc names to make it easier to move them
// to different timers if the need arises. Keep these definitons close to the
// definitions for the response functions to make it easire to check that
// the timer number matches where the timer event will be routed

#define _1_SEC_TIMER_CV 0
#define _1_SEC_TIMER_IP 1


#endif /* CONFIGURE_H */
```