

```

1  /*****
2  *
3  *   Buzzer Sequences
4  *   _____
5  *   |   |   |   |   |   |   |   |   |   |   |   |   |
6  *   |   |   |   |   |   |   |   |   |   |   |   |   |
7  *   |   |   |   |   |   |   |   |   |   |   |   |   |
8  *   |   |   |   |   |   |   |   |   |   |   |   |   |
9  *****/
10 *
11 * File      : buzzer.h
12 * Version   : 1.0
13 *
14 *****/
15 *
16 * Description : Managing buzzer state machine and sequences
17 *
18 *           Thanks to robsoncouto on GitHub for musics !
19 *           https://github.com/robsoncouto/arduino-songs.git
20 *
21 *****/
22 *
23 * Author      : Miguel Santos
24 * Date        : 25.09.2023
25 *
26 *****/
27 *
28 * MPLAB X      : 5.45
29 * XC32         : 2.50
30 * Harmony      : 2.06
31 *
32 *****/
33
34 #ifndef _BZR_H
35 #define _BZR_H
36
37 /*****
38
39 #include <stdint.h>
40 #include <stdbool.h>
41 #include "system_config.h"
42 #include "system_definitions.h"
43 #include "modules/counter.h"
44
45 *****/
46
47 /* Enumeration of sequences allow sequence
48  * to be called by their name out of this library */
49 typedef enum
50 {
51     BZR_SEQ_TEST,
52     BZR_SEQ_MARIO,
53     BZR_SEQ_IMPERIAL,
54 }E_BZR_SEQ;
55
56 *****/
57
58 /* Struct to hold informations about each sequence */
59 typedef struct
60 {
61     /* Music tempo in BPM */
62     uint16_t tempo;
63
64     /* Size of the sequence (use sizeof) */
65     uint16_t size;
66
67     /* Point to the array holding the notes and timings */
68     int16_t *notes;
69
70 }S_BZR_SEQ;
71
72 *****/
73

```

```

74  /* Buzzer state machine */
75  typedef enum
76  {
77      /* Buzzer waiting for new sequence */
78      BZR_STATE_IDLE,
79
80      /* Buzzer getting the note to play */
81      BZR_STATE_NOTE,
82
83      /* Buzzer playing the note and waiting */
84      BZR_STATE_PLAYING,
85
86  } BZR_STATES;
87
88  /*****
89
90  /* Buzzer structure of global datas */
91  typedef struct
92  {
93      /* The buzzer current state */
94      BZR_STATES state;
95
96      bool newSequence;
97
98      uint32_t tmrFrequency;
99
100     uint16_t tempo;
101     int16_t *currentNote;
102     int16_t *lastNote;
103     int16_t *sequence;
104
105     S_Counter counterPlay;
106
107 } BZR_DATA;
108
109 /*****
110
111 /**
112  * @brief BZR_Initialize
113  *
114  * Initialize buzzer state machine
115  *
116  * @param void
117  * @return void
118  */
119 void BZR_Initialize ( void );
120
121 /*****
122
123 /**
124  * @brief BZR_Tasks
125  *
126  * Execute buzzer state machine, should be called cyclically
127  *
128  * @param void
129  * @return void
130  */
131 void BZR_Tasks ( void );
132
133 /*****
134
135 /**
136  * @brief BZR_PlaySequence
137  *
138  * Play a music sequence using the state machine
139  *
140  * @param E_BZR_SEQ song Call the music you wann play !
141  * @return void
142  */
143 void BZR_PlaySequence(E_BZR_SEQ song);
144
145 /*****

```

```
147  #endif /* _BZR_H */
148
149  /* End of File *****/
150
```