

```

1  /*****
2  *
3  *   _____   _____   _____   _____   _____   _____
4  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
5  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
6  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
7  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
8  *   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
9  * *****/
10 *
11 * File      : fifo.h
12 * Version   : 1.0
13 *
14 *****/
15 *
16 * Description : Managing a FIFO using descriptor and pointers
17 *              Maximal size of FIFO is 255
18 *
19 *****/
20 *
21 * Author      : Miguel Santos
22 * Date        : 14.09.2023
23 *
24 *****/
25 *
26 * MPLAB X      : 5.45
27 * XC32         : 2.50
28 * Harmony      : 2.06
29 *
30 *****/
31
32 #ifndef FIFO_H
33 #define FIFO_H
34
35 /*****/
36
37 #include <stdint.h>
38 #include <stdbool.h>
39
40 /*****/
41
42 /* FIFO descriptor structure */
43 typedef struct fifo {
44     uint16_t size;
45     uint8_t *write;
46     uint8_t *read;
47     uint8_t *start;
48     uint8_t *end;
49 } S_Fifo;
50
51 /*****/
52
53 /**
54  * @brief FIFO_Init
55  *
56  * This function initializes a FIFO with the provided parameters,
57  * setting its size, start address, and initializing all elements
58  * to the given initial value.
59  *
60  * @param fifoDescriptor Pointer to the FIFO descriptor structure.
61  * @param fifoSize       The size of the FIFO.
62  * @param fifoStart      Pointer to the beginning of the FIFO memory.
63  * @param initialValue   The initial value to set for all elements in the FIFO.
64  */
65 void FIFO_Initialize( S_Fifo *fifoDescriptor, uint16_t fifoSize,
66                     uint8_t *fifoStart, uint8_t initialValue );
67
68 /*****/
69
70 /**
71  * @brief FIFO_GetWriteSpace
72  *
73  * This function calculates the available space for writing

```

```

74  * in the provided FIFO descriptor,
75  * taking into account the current read and write positions.
76  *
77  * @param fifoDescriptor Pointer to the FIFO descriptor structure.
78  * @return The available space for writing in the FIFO.
79  */
80  uint8_t FIFO_GetWriteSpace( S_Fifo *fifoDescriptor );
81
82  /*****
83
84  /**
85   * @brief FIFO_GetReadSpace
86   *
87   *
88   * This function calculates the available space for reading
89   * from the provided FIFO descriptor,
90   * taking into account the current read and write positions.
91   *
92   * @param fifoDescriptor Pointer to the FIFO descriptor structure.
93   * @return The available space for reading from the FIFO.
94   */
95  uint8_t FIFO_GetReadSpace( S_Fifo *fifoDescriptor );
96
97  /*****
98
99  /**
100   * @brief FIFO_Add
101   *
102   * This function attempts to put the specified character into the FIFO.
103   * If the FIFO is full, returns 0 (FIFO FULL),
104   * otherwise, it puts the character and returns 1 (OK).
105   *
106   * @param fifoDescriptor Pointer to the FIFO descriptor structure.
107   * @param value           The value to add to the FIFO.
108   * @return 1 if (OK), 0 if (FIFO FULL).
109   */
110  bool FIFO_Add( S_Fifo *fifoDescriptor , uint8_t value );
111
112  /*****
113
114  /**
115   * @brief FIFO_GetData
116   *
117   * This function attempts to get a value from the FIFO.
118   * If the FIFO is empty, returns 0 (FIFO EMPTY),
119   * otherwise, it gets the value and returns 1 (OK).
120   *
121   * @param fifoDescriptor Pointer to the FIFO descriptor structure.
122   * @param value           Pointer to store the retrieved value.
123   * @return 1 if (OK), 0 if (FIFO EMPTY).
124   */
125  bool FIFO_GetData( S_Fifo *fifoDescriptor , uint8_t *value );
126
127  /*****
128
129  /**
130   * @brief FIFO_GetBuffer
131   *
132   * This function attempts to get all the FIFO in a buffer.
133   * If the FIFO is empty, returns 0 (FIFO EMPTY),
134   * otherwise, it gets the value and returns 1 (OK).
135   *
136   * @param fifoDescriptor Pointer to the FIFO descriptor structure.
137   * @param buffer         Pointer to the buffer to store the FIFO.
138   * @return true if (OK), false if (FIFO EMPTY).
139   */
140  bool FIFO_GetBuffer( S_Fifo *fifoDescriptor , uint8_t *buffer );
141
142  /*****
143
144  #endif
145
146  /* End of File *****/

```