```
1
     * fram func.h
 2
 3
4
     * Created on: 6 juil. 2023
 5
            Author: christop.grobety
 6
 7
     #ifndef INC_FRAM_FUNC_H_
8
     #define INC_FRAM FUNC_H
9
10
11
     #define MEMORY SIZE 524288
12
     #include "main.h"
13
     #include <stdbool.h>
14
     #include <stdio.h>
15
16
17
     * @brief Write one or more value in a FRAM
18
     * @param State state : FRAM where to send the value(s)
19
     * @param TX FRAM: Pointer of the table to send
20
     * @param data size: Number of values to send on the FRAM
21
      * @param hspi 1 : SPI of the STM32 to use to send the value
22
23
     void FRAM write one (enum State state , uint8 t*TX FRAM, uint8 t data size,
     SPI HandleTypeDef hspi 1);
24
25
26
     * @brief Write one or more value in all the FRAMs
27
     * @param TX FRAM: Pointer of the table to send
28
     * @param data size: Number of values to send on the FRAMs
     * @param hspi 1 : SPI of the STM32 to use to send the value
29
30
31
    void FRAM write all(SPI HandleTypeDef hspi 1, uint8 t*TX FRAM, uint8 t data size);
32
33
     /**
     ^{\star} {	t @brief} Write in all the FRAMs the instruction of receive datas from ADC DEVICE
34
35
     * \texttt{@param} hspi 1 : SPI of the STM32 to use to send the value
36
37
    void FRAM write ADC to FRAM(SPI HandleTypeDef hspi 1);
38
39
40
     * @brief Write in all the FRAMs the instruction to set the register for write
     instruction
41
      * @param hspi 1 : SPI of the STM32 to use to send the value
42
43
     void FRAM write reg(SPI HandleTypeDef hspi 1);
44
     /**
45
     ^{\star} \mbox{@brief} Write in all the FRAMs the instruction to reset the register
46
     * @param hspi 1 : SPI of the STM32 to use to send the value
47
48
49
    void FRAM_reset_reg(SPI_HandleTypeDef hspi_1);
50
51
52
      * @brief Write in all the FRAMs the instruction to reset the register
53
     * @param State state : FRAM where to read the Register's values
     * @param hspi 1 : SPI of the STM32 to use to send the value
54
55
56
    void FRAM_read_reg(enum State state_, SPI_HandleTypeDef hspi 1);
57
58
     /**
     * @brief Read in a FRAM his Device's value
59
60
     * @param hspi 1 : SPI of the STM32 to use to send the value
61
62
    void FRAM device(SPI HandleTypeDef hspi 1);
63
64
     * @brief Read in a FRAM his Device's value
65
66
     *  * @param State state : FRAM where to read the Register's values
67
      * @param add: Location of the first address to start read the value of a FRAM
     * @param hspi_1 : SPI of the STM32 to use to send the value
     * @param data_size : size of the address where to read value
70
      * @return a pointer of a table with the FRAM's values
71
```

```
uint8_t* FRAM_read(enum State state_, uint32_t add,SPI HandleTypeDef hspi 1, uint8 t
 72
      data size);
 73
      /**
 74
 75
      * @brief Reset the values of the Selector's pins to send in the FPGA
 76
 77
     void PIN reset();
 78
 79
 80
      * @brief Send the value of pretrigger to the FPGA
 81
      * @param hspi_1 : SPI of the STM32 to use to send the value
 82
      * @param pretrig : pretrig's value to send to the FPGA
 83
     void setPreTrigg(SPI HandleTypeDef hspi 1, uint8 t pretrig);
 84
 85
 86
 87
      * @brief Selection of the channel to trigg
 88
      * @param chan: value of the channel to use
 89
 90
      void setTriggChannel(enum trig channel chan);
 91
 92
 93
      * @brief To turn on a Led
 94
      * @param color: color of the Led to turn on
 95
     void LED_on(enum color_Led color);
 96
 97
 98
      * @brief To turn off a Led
 99
     */
100
101
     void LED off();
102
103
     #endif /* INC FRAM FUNC H */
104
105
```