```c
/*
 * sd_card_func.c
 *
 *  Created on: Jul 7, 2023
 *      Author: christop.grobety
 */

#include "sd_card_func.h"
#include <math.h>

FRESULT res_sd;
uint32_t byteswritten_sd, bytesread_sd;
uint8_t rtext_ [_MAX_SS] ;
uint8_t wtext_2 [100] ;
uint8_t test [2] ;
double init_Value [4] ;

void SD_create_file(FIL sd_file, TCHAR* fileName){
    res_sd = f_mount(&SDFatFS, (TCHAR const*)SDPath, 0) ;
    if(res_sd != FR_OK){
        LED_on(RED);
    }
    else{
      //Open file for writing (Create)
      res_sd = f_open(&sd_file, fileName, (FA_CREATE_ALWAYS | FA_WRITE));
      if(res_sd != FR_OK){
          LED_on(RED);
      }
      else{
          //LED_on(BLUE);
          f_close(&sd_file);
      }
    }
    f_mount(&SDFatFS, (TCHAR const*)NULL, 0);
}

void SD_write_data(FIL sd_file, TCHAR* fileName, uint16_t* data){
    //sprintf((char*)wtext_, "%d \n",data);
    res_sd = f_mount(&SDFatFS, (TCHAR const*)SDPath, 0) ;
    if(res_sd != FR_OK){
        LED_on(RED);
    }
    else{
    //Open file for writing (Create)
      res_sd = f_open(&sd_file, fileName, (FA_OPEN_ALWAYS | FA_WRITE));
      if(res_sd != FR_OK){
          LED_on(RED);
      }
      else{
        //Write to the text file
          if(f_lseek(&sd_file, f_size(&sd_file)) == FR_OK){
              //res_sd = f_write(&sd_file, wtext_, strlen((char *)wtext_), (void
              *)&byteswritten_sd);
              res_sd = f_write(&sd_file, data, (TAB_SIZE), (void *)&byteswritten_sd);
            if((byteswritten_sd == 0) || (res_sd != FR_OK)){
                  LED_on(RED);
            }
            else{
                //LED_on(BLUE);
                f_close(&sd_file);
            }
          }
      }
    }
    f_mount(&SDFatFS, (TCHAR const*)NULL, 0);
}

uint8_t* SD_read_data(FIL sd_file, TCHAR* fileName){
    res_sd = f_mount(&SDFatFS, (TCHAR const*)SDPath, 0) ;
        if(res_sd != FR_OK){
            LED_on(RED);
        }
        else{
```

```
73          //Open file for writing (Create)
74            res_sd = f_open(&sd_file, "INIT.txt", FA_OPEN_ALWAYS | FA_READ);
75            if(res_sd != FR_OK){
76                LED_on(RED);
77            }
78            else{
79              //Write to the text file
80                if(f_lseek(&sd_file, 0) == FR_OK){
81                    while(bytesread_sd == 0){
82                        res_sd = f_read(&sd_file, rtext_, sizeof(rtext_), (UINT *)&
                          bytesread_sd);
83                    }
84                    //f_gets(wtext_, sd_file.fptr, &sd_file);
85                  if((bytesread_sd == 0) || (res_sd != FR_OK)){
86                        LED_on(RED);
87                  }
88                  else{
89                      //LED_on(BLUE);
90                      f_close(&sd_file);
91                  }
92                }
93              }
94          }
95          f_mount(&SDFatFS, (TCHAR const*)NULL, 0);
96          return (uint8_t*)&rtext_;
97    }
98    double* initValue(FIL sd_file, TCHAR* fileName){
99        uint8_t* data = SD_read_data(sd_file, fileName);
100       uint8_t value[_MAX_SS/2];
101       uint8_t number = 0;
102       uint8_t number_ = 0;
103       uint8_t pow_ = 0;
104       uint8_t div_ = 0;
105       bool fraction = false;
106       for(uint32_t i = 0 ; i<_MAX_SS; i++){
107           if(data[i] >= '0' && data[i] <= '9' ){
108               value[number] = data[i]-'0';
109               number++;
110               pow_++;
111           }
112           else if(data[i] == '.' || data[i] == ','){
113               fraction = true;
114               div_ = pow_;
115           }
116           else if(data[i] == '\n' || data[i] == '\0'){
117               for(uint8_t j = 0; j<number ; j++){
118                   init_Value[number_] = init_Value[number_] + value[j]*pow(10,number-j-1
                      );
119               }
120               if(fraction){
121                   fraction = false;
122                   init_Value[number_] = init_Value[number_]/(pow(10,number-div_));
123               }
124               pow_ = 0;
125               div_ = 0;
126               number = 0;
127               number_++;
128               if(data[i] == '\0'){
129                   break;
130               }
131           }
132       }
133       return (double*)&init_Value;
134   }
135
136
```