VelvetApp

Kamalov Marat

Версия 1.0

# Список файлов

## Файлы

Полный список документированных файлов.

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/adc.c (Исходный код функций для чтения АЦП )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/app.c (Исходный код функции запуска задач по таймауту )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/appInit.c (Исходный код функций инициализации ПО )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/esp.c (Исходный код функций для отправки и приёма сообщений от микроконтроллера ESP )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/main.c (: Main program body )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/rfid.c (Исходный код функций для чтения RFID меток )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/stm32f4xx\_hal\_msp.c (This file provides code for the MSP Initialization and de-Initialization codes )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/stm32f4xx\_hal\_timebase\_tim.c (HAL time base based on the hardware TIM )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/stm32f4xx\_it.c (Interrupt Service Routines )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/system\_stm32f4xx.c (CMSIS Cortex-M4 Device Peripheral Access Layer System Source File )** pagenum

**C:/Projects/Velvet/Software/VelvetApp/Core/Src/uartCallback.c (Исходный код функции обработки прерываний по интерфейсу UART )** pagenum

# Файлы

## Файл C:/Projects/Velvet/Software/VelvetApp/Core/Src/adc.c

Исходный код функций для чтения АЦП

#include "adc.h"

#include "hx711.h"

#include "cmsis\_os2.h"

#include "app.h"

#include "esp.h"

### Макросы

#define **RECEIVE\_OK**  0

#define **SAMPLE\_NUMBER**  10

### Функции

void **initADC** (void)

*Функция инициализации АЦП*

void **readWeightTask** (void \*argument)

uint8\_t **getAdcState** (void)

*Функция получения состояния АЦП*

void **setAdcState** (uint8\_t state)

*Функция установки состояния АЦП*

float **getWeightValByIndex** (uint8\_t index)

*Функция чтения веса из массива весов по индексу*

### Переменные

osMessageQueueId\_t **adcQueueHandle**

### Подробное описание

Исходный код функций для чтения АЦП

##### Дата

19.04.2023

##### Автор

Kamalov Marat

### Функции

#### uint8\_t getAdcState (void )

Функция получения состояния АЦП

##### Аргументы

|  |  |
| --- | --- |
| *Отсутствует* |  |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *ADC\_BUSY* | если АЦП занято, ADC\_FREE если свободно |

#### float getWeightValByIndex (uint8\_t *index*)

Функция чтения веса из массива весов по индексу

##### Аргументы

|  |  |
| --- | --- |
| *index* | - индекс в массиве |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Вес* |  |

#### void initADC (void )

Функция инициализации АЦП

##### Аргументы

|  |  |
| --- | --- |
| *Отсутствует* |  |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void readWeightTask (void \* *argument*)

@brief Задача чтения веса

Задача производит инициализацию АЦП. После инициализации АЦП и поступления команды от задачи чтения RFID меток производится чтение АЦП в количестве 60 раз для заполнения массива весов

##### Аргументы

|  |  |
| --- | --- |
| *Отсутствует* |  |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void setAdcState (uint8\_t *state*)

Функция установки состояния АЦП

##### Аргументы

|  |  |
| --- | --- |
| *state* | - состояние АЦП |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

## Файл C:/Projects/Velvet/Software/VelvetApp/Core/Src/app.c

Исходный код функции запуска задач по таймауту

#include "app.h"

#include "cmsis\_os2.h"

#include "esp.h"

### Функции

void **checkTasksTimeout** ()

*Функция запуска задач по таймауту.*

### Переменные

volatile uint16\_t **secondsCounter**

volatile uint32\_t **timestamp** = 0

### Подробное описание

Исходный код функции запуска задач по таймауту

##### Дата

23.02.2023

##### Автор

Kamalov Marat

### Функции

#### void checkTasksTimeout ()

Функция запуска задач по таймауту.

1 раз в 15 минут вызывается синхронизация системного времени 1 раз в минуту вызывается проверка наличия обновлений ПО

##### Аргументы

|  |  |
| --- | --- |
| *Отсутствует* |  |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

## Файл C:/Projects/Velvet/Software/VelvetApp/Core/Src/appInit.c

Исходный код функций инициализации ПО

#include "appInit.h"

#include "app.h"

### Функции

void **initApp** (void)

*Функция инициализации ПО: запускает аппаратный таймер 6.*

### Переменные

TIM\_HandleTypeDef **htim6**

### Подробное описание

Исходный код функций инициализации ПО

##### Дата

23.02.2023

##### Авторы

Kamalov Marat

### Функции

#### void initApp (void )

Функция инициализации ПО: запускает аппаратный таймер 6.

##### Аргументы

|  |  |
| --- | --- |
| *None* |  |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *None* |  |

## Файл C:/Projects/Velvet/Software/VelvetApp/Core/Src/esp.c

Исходный код функций для отправки и приёма сообщений от микроконтроллера ESP

#include "esp.h"

#include "cmsis\_os2.h"

#include "app.h"

#include "adc.h"

#include "rfid.h"

#include "main.h"

#include "string.h"

### Структуры данных

union **weight**

### Макросы

#define **RECEIVE\_OK**  0

### Перечисления

enum **EspResponse\_t** { **RESPONSE\_ERROR**, **RESTART\_STM**, **TRANSMIT\_PREPARE\_OK**, **SEND\_WEIGHT\_OK**, **TRANSMIT\_WIFI\_ERROR**, **TIMESTAMP\_RESPONSE\_OK** }

*Набор возможных ответов микроконтроллера ESP*

### Функции

**EspResponse\_t** **checkResponse** (uint8\_t \*buf)

*Функция проверки ответа от микроконтроллера ESP*

void **prepareSendBuffer** (void)

*Функция подготовки буфера измеренных весов для отправки в микроконтроллер ESP*

void **readEspResponse** (uint8\_t \*buf)

*Функция чтения ответа от микроконтроллера ESP.*

void **sendMsgToESP** (EspMsg\_t sendMessageType)

*Функция отправки сообщения микроконтроллеру ESP.*

void **sendMsgToESPTask** (void \*argument)

*Задача для отправки сообщений микроконтроллеру ESP*

### Переменные

osMessageQueueId\_t **espSendQueueHandle**

osMessageQueueId\_t **espReceiveQueueHandle**

### Подробное описание

Исходный код функций для отправки и приёма сообщений от микроконтроллера ESP

##### Дата

12.03.2023

##### Автор

Kamalov Marat

### Функции

#### EspResponse\_t checkResponse (uint8\_t \* *buf*)

Функция проверки ответа от микроконтроллера ESP

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив с ответом микроконтроллера ESP |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Код* | ответа микроконтроллера ESP |

#### void prepareSendBuffer (void )

Функция подготовки буфера измеренных весов для отправки в микроконтроллер ESP

Также отправляется RFID метка животного и метка времени

##### Аргументы

|  |  |
| --- | --- |
| *Отсутствует* |  |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void readEspResponse (uint8\_t \* *buf*)

Функция чтения ответа от микроконтроллера ESP.

В соответствии с ответом ESP запускаются функции:

Перезагрузка микроконтроллера

Отправка массива с измеренными весами

Разблокировка АЦП

Обновление системного времени

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив с ответом микроконтроллера ESP |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void sendMsgToESP (EspMsg\_t *sendMessageType*)

Функция отправки сообщения микроконтроллеру ESP.

В соответствии типом сообщения производится отправка соответствующей последовательности байт:

Подготовка ESP к приёму массива весов

Проверка наличия обновлений ПО

Синхронизация системного времени

Отправка массива измеренных весов

##### Аргументы

|  |  |
| --- | --- |
| *sendMessageType* | Тип сообщения, отправляемого микроконтроллеру ESP |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### \_\_weak void sendMsgToESPTask (void \* *argument*)

Задача для отправки сообщений микроконтроллеру ESP

Function implementing the sendMsg thread.

Задача производит отправку сообщений для микроконтроллера ESP. Сообщения принимаются через очередь сообщений. Также задача производит приём сообщений от ESP

##### Аргументы

|  |  |
| --- | --- |
| *argument* | Указатель на структуру интерфейса UART, подключенного к ESP |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

##### Аргументы

|  |  |
| --- | --- |
| *argument* | Not used |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *None* |  |

## Файл C:/Projects/Velvet/Software/VelvetApp/Core/Src/main.c

: Main program body

#include "main.h"

#include "cmsis\_os.h"

#include "appInit.h"

#include "app.h"

#include "esp.h"

#include "rfid.h"

#include "adc.h"

### Функции

void **SystemClock\_Config** (void)

*System Clock Configuration.*

void **sendMsgToESPTask** (void \*argument)

*Function implementing the sendMsg thread.*

void **readWeightTask** (void \*argument)

void **readRfidTask** (void \*argument)

*Задача чтения RFID меток.*

int **main** (void)

*The application entry point.*

void **HAL\_TIM\_PeriodElapsedCallback** (TIM\_HandleTypeDef \*htim)

*Period elapsed callback in non blocking mode.*

void **Error\_Handler** (void)

*This function is executed in case of error occurrence.*

### Переменные

IWDG\_HandleTypeDef **hiwdg**

TIM\_HandleTypeDef **htim6**

UART\_HandleTypeDef **huart4**

UART\_HandleTypeDef **huart6**

DMA\_HandleTypeDef **hdma\_uart4\_rx**

DMA\_HandleTypeDef **hdma\_usart6\_rx**

osThreadId\_t **sendMsgHandle**

const osThreadAttr\_t **sendMsg\_attributes**

osThreadId\_t **readWeightHandle**

const osThreadAttr\_t **readWeight\_attributes**

osThreadId\_t **readRfidHandle**

const osThreadAttr\_t **readRfid\_attributes**

### Подробное описание

: Main program body

##### Внимание

Copyright (c) 2023 STMicroelectronics. All rights reserved.

This software is licensed under terms that can be found in the LICENSE file in the root directory of this software component. If no LICENSE file comes with this software, it is provided AS-IS.

### Функции

#### void Error\_Handler (void )

This function is executed in case of error occurrence.

##### Возвращаемые значения

|  |  |
| --- | --- |
| *None* |  |

#### void HAL\_TIM\_PeriodElapsedCallback (TIM\_HandleTypeDef \* *htim*)

Period elapsed callback in non blocking mode.

##### Заметки

This function is called when TIM1 interrupt took place, inside HAL\_TIM\_IRQHandler(). It makes a direct call to HAL\_IncTick() to increment a global variable "uwTick" used as application time base.

##### Аргументы

|  |  |
| --- | --- |
| *htim* | : TIM handle |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *None* |  |

#### int main (void )

The application entry point.

##### Возвращаемые значения

|  |  |
| --- | --- |
| *int* |  |

#### void readRfidTask (void \* *argument*)

Задача чтения RFID меток.

Задача производит инициализацию RFID модуля. После инициализации с периодом 1 с производится чтение массива RFID меток и запуск соответствующих обработчиков

##### Аргументы

|  |  |
| --- | --- |
| *argument* | Указатель на структуру интерфейса UART, подключенного к RFID модулю |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void readWeightTask (void \* *argument*)

@brief Задача чтения веса

Задача производит инициализацию АЦП. После инициализации АЦП и поступления команды от задачи чтения RFID меток производится чтение АЦП в количестве 60 раз для заполнения массива весов

##### Аргументы

|  |  |
| --- | --- |
| *Отсутствует* |  |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void sendMsgToESPTask (void \* *argument*)

Function implementing the sendMsg thread.

##### Аргументы

|  |  |
| --- | --- |
| *argument* | Not used |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *None* |  |

#### void SystemClock\_Config (void )

System Clock Configuration.

##### Возвращаемые значения

|  |  |
| --- | --- |
| *None* |  |

Configure the main internal regulator output voltage

Initializes the RCC Oscillators according to the specified parameters in the RCC\_OscInitTypeDef structure.

Initializes the CPU, AHB and APB buses clocks

### Переменные

#### const osThreadAttr\_t readRfid\_attributes

**Инициализатор**= {

.name = "readRfid",

.stack\_size = 128 \* 4,

.priority = (osPriority\_t) osPriorityNormal1,

}

#### const osThreadAttr\_t readWeight\_attributes

**Инициализатор**= {

.name = "readWeight",

.stack\_size = 128 \* 4,

.priority = (osPriority\_t) osPriorityNormal,

}

#### const osThreadAttr\_t sendMsg\_attributes

**Инициализатор**= {

.name = "sendMsg",

.stack\_size = 128 \* 4,

.priority = (osPriority\_t) osPriorityBelowNormal7,

}

## Файл C:/Projects/Velvet/Software/VelvetApp/Core/Src/rfid.c

Исходный код функций для чтения RFID меток

#include "rfid.h"

#include "cmsis\_os2.h"

#include "stm32f4xx\_hal.h"

#include "main.h"

#include "adc.h"

#include "uartCallback.h"

#include "string.h"

### Макросы

#define **PRESET\_VALUE**  0xFFFF

#define **POLYNOMIAL**  0x8408

#define **RECEIVE\_OK**  0

#define **TRANSMIT\_OK**  0

#define **GET\_READER\_INFO\_CMD**  0x21

#define **INVENTORY\_G2\_CMD**  0x01

### Функции

uint16\_t **uiCrc16Calc** (uint8\_t \*buf, uint8\_t length)

*Функция расчёта контрольной суммы CRC16.*

void **setCRC16** (uint8\_t \*buf)

*Функция записи контрольной суммы CRC16 в массив.*

void **getReaderInfo** (uint8\_t \*buf)

*Функция записи в массив команды на чтение настроек RFID модуля.*

void **writeCMDToBuf** (uint8\_t \*buf, uint8\_t cmdNum)

*Функция записи команды в массив.*

void **sendCmd** (uint8\_t \*buf, uint8\_t cmdNum)

*Функция отправки команды RFID модулю.*

int8\_t **checkBufCRC** (uint8\_t \*buf)

*Функция проверки контрольной суммы буфера.*

void **readEPCData** (uint8\_t \*buf)

*Функция чтения массива данных, полученных от RFID модуля.*

void **readRfidResponse** (uint8\_t \*buf)

*Функция чтения ответа, полученного от RFID модуля.*

void **rfidInit** (void)

*Функция инициализации RFID модуля.*

void **rfidReInit** (void)

*Функция реинициализации RFID модуля.*

void **readRfidTask** (void \*argument)

*Задача чтения RFID меток.*

void **getCurrentTag** (uint8\_t \*tagBuf)

*Функция записи текущей метки в массив.*

### Переменные

IWDG\_HandleTypeDef **hiwdg**

osMessageQueueId\_t **rfidReceiveQueueHandle**

### Подробное описание

Исходный код функций для чтения RFID меток

##### Дата

02.04.2023

##### Автор

Kamalov Marat

### Функции

#### int8\_t checkBufCRC (uint8\_t \* *buf*)

Функция проверки контрольной суммы буфера.

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив для проверки |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *RFID\_OK* | если CRC корректная, RFID\_ERROR если нет |

#### void getCurrentTag (uint8\_t \* *tagBuf*)

Функция записи текущей метки в массив.

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив данных для записи |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void getReaderInfo (uint8\_t \* *buf*)

Функция записи в массив команды на чтение настроек RFID модуля.

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив для записи команды |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void readEPCData (uint8\_t \* *buf*)

Функция чтения массива данных, полученных от RFID модуля.

В функии проверяется наличие меток в массиве данных, отличие новой метки от текущей. Также проверяется если соответствует ли метка калибровочной, либо предназначенной для установки нуля. При получении соответсвующей метки производится отправка команды на калибровку АЦП, установку нуля и чтение массива весов.

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив данных, полученных от RFID модуля |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void readRfidResponse (uint8\_t \* *buf*)

Функция чтения ответа, полученного от RFID модуля.

В функии код команы, полученной от RFID модуля и производится вызов функии для обработки массива

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив данных, полученных от RFID модуля |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void readRfidTask (void \* *argument*)

Задача чтения RFID меток.

Задача производит инициализацию RFID модуля. После инициализации с периодом 1 с производится чтение массива RFID меток и запуск соответствующих обработчиков

##### Аргументы

|  |  |
| --- | --- |
| *argument* | Указатель на структуру интерфейса UART, подключенного к RFID модулю |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void rfidInit (void )

Функция инициализации RFID модуля.

##### Аргументы

|  |  |
| --- | --- |
| *Отсутствует* |  |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void rfidReInit (void )

Функция реинициализации RFID модуля.

##### Аргументы

|  |  |
| --- | --- |
| *Отсутствует* |  |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void sendCmd (uint8\_t \* *buf*, uint8\_t *cmdNum*)

Функция отправки команды RFID модулю.

В функии также сразу включается приемник команд от RFID модуля

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив для записи команды |
| *cmdNum* | Код команды |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void setCRC16 (uint8\_t \* *buf*)

Функция записи контрольной суммы CRC16 в массив.

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив для расчёта контрольной суммы |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### uint16\_t uiCrc16Calc (uint8\_t \* *buf*, uint8\_t *length*)

Функция расчёта контрольной суммы CRC16.

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив для расчёта контрольной суммы |
| *length* | Размер массива |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Контрольная* | сумма массива |

#### void writeCMDToBuf (uint8\_t \* *buf*, uint8\_t *cmdNum*)

Функция записи команды в массив.

Записываются команды на чтение настроек RFID модуля и массива считанных меток

##### Аргументы

|  |  |
| --- | --- |
| *buf* | Массив для записи команды |
| *cmdNum* | Код команды |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

## Файл C:/Projects/Velvet/Software/VelvetApp/Core/Src/uartCallback.c

Исходный код функции обработки прерываний по интерфейсу UART

#include "uartCallback.h"

#include "stm32f4xx\_hal.h"

#include "rfid.h"

#include "esp.h"

#include "cmsis\_os2.h"

#include "app.h"

### Функции

void **HAL\_UART\_TxCpltCallback** (UART\_HandleTypeDef \*huart)

*Функция обработки прерываний по завершению передачи по интерфейсу UART.*

void **HAL\_UART\_RxCpltCallback** (UART\_HandleTypeDef \*huart)

*Функция обработки прерываний по завершению приёма по интерфейсу UART.*

void **setReceiveStage** (uint8\_t stage)

*Функция установки стадии приёма от RFID модуля.*

### Подробное описание

Исходный код функции обработки прерываний по интерфейсу UART

##### Дата

03.04.2023

##### Автор

Kamalov Marat

### Функции

#### void HAL\_UART\_RxCpltCallback (UART\_HandleTypeDef \* *huart*)

Функция обработки прерываний по завершению приёма по интерфейсу UART.

В соответствии с выбранным интерфесом производится вызов обработчиков через очередь для сообщений

##### Аргументы

|  |  |
| --- | --- |
| *huart* | - Указатель на структуру интерфейса UART |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void HAL\_UART\_TxCpltCallback (UART\_HandleTypeDef \* *huart*)

Функция обработки прерываний по завершению передачи по интерфейсу UART.

##### Аргументы

|  |  |
| --- | --- |
| *huart* | - Указатель на структуру интерфейса UART |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |

#### void setReceiveStage (uint8\_t *stage*)

Функция установки стадии приёма от RFID модуля.

##### Аргументы

|  |  |
| --- | --- |
| *stage* | - стадия приёма |

##### Возвращаемые значения

|  |  |
| --- | --- |
| *Отсутствует* |  |