## Main.c

### int main()

inicializuje periférie pomocou inicializačnýh funkcií a zabezpečuje beh hlavného programu a spúšťanie merania periodickým RTC časovačom

## core.c

### uint32\_t millis()

vracia uptime v ms

### void gpio\_init()

inicializuje gpio pre UART a zapínanie gsm modulu

### void uart\_init()

inicializácia UART periférie

### void rtc\_init()

nastavenie cesty hodinového signálu pre RTC ak ešte nie je nastavená a nastavenie periodického časovača

### void rtc\_settimedate(RTC\_TimeTypeDef \* time, RTC\_DateTypeDef \* date)

nastavenie aktuálneho času RTC periférie

### void int\_init()

inicializácia NVIC periférie

## ads1100.c

funkcie na riadenie ADS1100 adc

### Status ADS1100\_ContConvInit()

Spustenie kontinuálnej konverzie

### Status ADS1100\_StartConv()

Spustenie jednej konverzie

### Status ADS1100\_readData(int16\_t \*data)

Vyčítanie dátového registra

### Status ADS1100\_readDataStatus(int16\_t \* data, uint8\_t \* status)

Vyčítanie dátového registra s indikáciou stavu konverzie

## Datasender.c

Stavový stroj na odoslanie nazbieraných dát pomocou gsm modulu, zabezpečuje zapnutie, inicializáciu, pripojenie a vypnutie gsm modulu

### void datasender\_proc()

periodikcky spúšťaná neblokujúca funkcia zabezpečujúca chod stavového stroja

## datastore.c

stavové stroje na obsluhu pripojených senzorov, vyčítanie hodnôt a ich uloženie do RAM

### void datastore\_proc()

periodikcky spúšťaná neblokujúca funkcia zabezpečujúca chod stavových strojov

### void datastore\_collectdata()

spustenie merania a zberu hodnôt zo senzorov

### uint32\_t storeddatanum()

vráti počet uložených vzoriek

### uint8\_t storeddataload(datasample \* ptr)

vráti pointer na štruktúru s nameranými dátami

### void storeddataloadsuccess()

potvrdenie úspešného odoslania dát

### void storeddataloadfail()

obnovenie pointra získaného pomocou storeddataload() na pôvodnú hodnotu v prípade neúspešného odoslania

### float rawtopressure(uint16\_t raw)

prevod adc hodnoty na tlak v kPa

### float rawtotemp(uint16\_t raw)

prevod adc hodnoty na teplotu v C

### float rawtorh(uint16\_t raw)

prevod adc hodnoty na relatívnu vlhkosť v %

## gsm.c

driver pre gsm modul, needs rework

### void gsm\_proc()

periodikcky spúšťaná neblokujúca funkcia zabezpečujúca chod troch stavových strojov, jede obsluhuje príjem znakov, druhý obsluhuje odosielanie príkazov a dát a detekciu odpovedí na ne a tretí obsluhuje postupnosť odosielania príkazov

### void gsm\_poweron()

zapnutie modulu

### void gsm\_poweroff()

vypnutie modulu

### gsm\_status\_t gsm\_status()

získanie aktuálneho stavu gsm modulu

### void gsm\_init()

inicializácia gsm na gprs pipojenie

### void gsm\_gprsconnect()

TCP pripojenie na nastavenú adresu

### void gsm\_gprsdisconnect()

TCP odpojenie

### char \* gsm\_getsendbuffer()

getter na odosielací buffer

### uint16\_t gsm\_getrecvlen()

getter na dĺžku prijatých dát

### char \* gsm\_getrecvbuffer()

getter na prijímací buffer

### void gsm\_senddata(uint16\_t len)

odoslanie dát v odosielacom buffri s uvedenou dĺžkou, ak je dĺžka nulová, na výpočet dĺžky sa použije EOF terminátor reťazca v buffri

### uint8\_t gsm\_socketstate()

getter na stav pripojenia

## i2c.c

opravená knižnica na obsluhu i2c, poskytnutá s prednáškami

## jsonconstrctr.c

generátor json reťazcov na formátovanie odosielaných dát

### void constructhead(char \* str, uint32\_t stationid)

generovanie hlavičky s id meteostanice

### void constructentry(char \* str, datasample \* entry, uint8\_t notlast)

generovanie jednotlivých elementov poľa s nameanými hodntami

### void constructtail(char \* str)

ukončenie json reťazca

## ringbuffer.c

prstencový zásobník pre uart

### char gettxbuff()

vybratie znaku z vysielaciho zásobníka

### void puttxbuff(char c)

vloženie znaku do vysielacieho zásobníka

### char getrxbuff()

vybratie znaku z prijímacieho zásobníka

### void putrxbuff(char c)

vloženie znaku do prijímacieho zásobníka

### uint16\_t gettxfull()

počet znakov vo vysielaciom zásobníku

### uint16\_t gettxfree()

počet voľných miest vo vysielaciom zásobníku

### uint16\_t getrxfull()

počet znakov v prijímaciom zásobníku

### uint16\_t getrxfree()

počet voľných miest v prijímaciom zásobníku

### uint8\_t buffwrite(const uint8\_t \* buff, uint8\_t len)

zápis celého reťazca do vysielacieho zásobníka

### uint8\_t buffwrite(const uint8\_t \* buff, uint8\_t len)

prečítanie celého reťazca z prijímacieho zásobníka

## sht21.c

driver na obsluhu kombinovaného senzora SHT-21

### Status SHT21\_res()

Reset senzora

### Status SHT21\_blockMeasT(uint16\_t \*val)

Spustenie blokujúceho merania teploty

### Status SHT21\_blockMeasRH(uint16\_t \*val)

Spustenie blokujúceho merania relatívnej vlhkosti

### Status SHT21\_trigMeasT()

Spustenie neblokujúceho merania teploty

### Status SHT21\_trigMeasRH()

Spustenie nebokujúceho merania relatívnej vlhkosti

### Status SHT21\_readVal(uint16\_t \*val, uint8\_t \* status)

Prečítanie hodnoty posledného merania

### float SHT21\_CalcRH(uint16\_t u16sRH)

výpočet RH z adc hodnoty

### float SHT21\_CalcTemperatureC(uint16\_t u16sT)

výpočet teploty z adc hodnoty

## stm32l1xx\_it.c

handlery pre prerušenia

### void SysTick\_Handler(void)

spúšťaný každú ms, inkrementuje premenú uint32\_t SysUpTime, používanú vo funkcii millis()

### void USART1\_IRQHandler(void)

prerušenie na obsluhu uart pri odosielaní a prijímaní znakov