Měřič kvality oblohy
(amatérské SQM)

http://sqm.AstroMiK.org

Návod pro úpravy SW

SW:	STM32	2024-12-18 (STM32F4x1)
	ATmega328	2024-12-14..int

20.1.2025

OBSAH

Úvod

Tento návod obsahuje detailní informace o programu, jeho možných úpravách a přizpůsobení.

Organizace vnitřní EEPROM

Uvnitř SQM se nachází paměť, do které se ukládají jednotlivé záznamy. Tato paměť je trvalá a pamatuje si zapsané údaje i při vypnutí napájení.

Pro ukládání záznamů slouží oblast od buňky s adresou 700 až do konce paměti (adresa 131071).

Na začátku paměti jsou uložené systémové parametry a kalibrační tabulky.

Adresy v EEPROM jsou nadefinované na začátku programu v souboru "stm411.ino" v bloku #doc#10:

```
#define eeaddr_RTC_set                0UL
#define eeaddr_RTC_korekce            4UL
#define eeaddr_leto_zima              8UL
#define eeaddr_oddelovace             9UL
#define eeaddr_automat               10UL
#define eeaddr_prumerovani           11UL
#define eeaddr_modbus_LED            12UL
#define eeaddr_LED                   13UL
#define eeaddr_SLAVE_baud            14UL
#define eeaddr_CRC_tset_lock         15UL
#define eeaddr_stab                  16UL
#define eeaddr_pocet_cidel           17UL
#define eeaddr_10kB_znacka           18UL
#define eeaddr_uroven_EEPROM         19UL
#define eeaddr_teplotni_kalibrace    20UL
#define eeaddr_TS_stanoviste         48UL
#define eeaddr_svetelna_kalibrace    50UL
#define eeaddr_TS_aktual            470UL
#define eeaddr_kompas_offset        471UL
#define eeaddr_perif_bity           472UL
#define eeaddr_beep_bity            474UL
```

```
#define eeaddr_elevace_soumrak      476UL
#define eeaddr_GPS_track_id         477UL
#define eeaddr_GPS_track_interval   479UL
#define eeaddr_rezerva_1            480UL
#define eeaddr_GEO_lat              485UL
#define eeaddr_GEO_lon              487UL
#define eeaddr_zima_hod             489UL
#define eeaddr_leto_hod             490UL
#define eeaddr_menu_bity            491UL
#define eeaddr_posledni_den         495UL
#define eeaddr_AfD                  499UL
#define eeaddr_stanoviste           500UL
#define eeaddr_txt_zony             575UL
#define eeaddr_luxmetr              583UL
#define eeaddr_rezerva_2            589UL
#define eeaddr_test_write           599UL
#define eeaddr_kompas               600UL
#define eeaddr_naklon_horiz         612UL
#define eeaddr_naklon_vert          618UL
#define eeaddr_alarmy               624UL
#define eeaddr_TEST_alarm           636UL
#define eeaddr_rezerva_3            637UL
#define eeaddr_MIN_EEPROM           700UL
```

Konkrétně tedy vypadá obsazení paměti takto:

adresa	**popis**
0 až 3	čas posledního seřízení RTC v sekundách od 1.1.1970
4 až 7	za jak dlouho se má přidávat nebo ubírat autokalibrační sekunda k vnitřnímu času (detailně popsáno v kapitole " Vnitřní hodiny (RTC)")
8	značka pro aktuální časovou zónu zimní (0) nebo letní (1).
9	nastavení oddělovačů položek a desetinných oddělovačů v CSV souborech na SD kartě. (detailně rozepsáno pod touto tabulkou)
10	interval automatického spouštění měření v minutách

11	počet vzorků k průměrování
12	zapnutý (0bx1), nebo vypnutý (0bx0) modbus LED menu (0b0x), nebo displejove menu (0b1x)
13	případné vypínání funkcí RGB LED
14	nastavená SLAVE adresa a rychlost komunikace (detailně rozepsáno pod touto tabulkou)
15	bit 0: CRC bajt se při příjmu dat z RS485 (0 = netestuje) (1 = testuje) bit 6: soubor RTC_set.csv (0 = zakázán; 1 = povolen) bit 7: Tlačítko pro TimeStamp (0 = odemčené) (1 = blokované)
16	úroveň pro určení nestabilního jasu
17	počet čidel světla na expanzní desce (jedno SQM dokáže číst hodnotu světla až ze 7 čidel připojených přes I^2C expander)
18	značka při každých obsazených 10kB paměti pro rychlejší hledání volného místa po zapnutí napájení.
19	Hranice světla pro ukládání do EEPROM. V desetinách mag/arcsec^2 (platí jen pro spouštění tlačítkem, automatem, nebo po sériové lince. V případě kalibrace se ignoruje)
20 až 47	kalibrační tabulka pro teplotu (2 sloupce * 7 dvojbajtových hodnot = 28 bajtů)

48 až 49	značka pro nulování pořadí v záznamech časových razítek a aktuální přednastavené pozorovací stanoviště (nejvyšší 3 bity)
50 až 469	kalibrační tabulky pro čidla světla (7 čidel * 2 sloupce * 15 dvojbajtových hodnot = 420 bajtů)
470	aktuální pořadové číslo časového razítka (pořadí se nuluje vždycky v poledne)
471	doladění kompasu při špatně nalepeném čidle (+/- 12,7°)
472 až 473	informace o dočasně deaktivovaých periféríích
474	informace o funkcích, které mají povolenou akustickou signalizaci (2 bajty)
476	elevace Slunce pod horizontem, při které se hlásí uživatelsky nastavitelný soumrak
477	2 bajty: stav GPS trasování (nejvyšší bit15) bit 14 nepoužit pořadové cislo trasovacího souboru (bity 13 až 0)
479	interval záznamu GPS souřadnic do trasovacího souboru v sekundách (defaultně 20 sekund)
480 až 484	rezerva (5 bajtů)
485 až 486	zeměpisná šířka pro astro výpočty v desetinách stupně
487 až 488	zeměpisná délka pro astro výpočty v desetinách stupně
489	posun časové zóny pro "zimní" čas proti UTC
490	posun časové zóny pro letní čas proti UTC
491 až 494	povolené, nebo zakázané položky v menu (bitově)

495 až 498	poslední den, kdy probíhal nějaký záznam. Používá se pro výpis posledního souboru se záznamy do sériové linky.
499	hodnota jasu pro funkci Alarm for Darkness v desetinách mag/arcsec^2.
500 až 574	5 přednastavených pozorovacích stanovišť (vždycky 10 znaků názvu a 5 bajtů grafiky pro displej)
575 až 582	2x4 znaky pro uživatelské nastavení označení časových zón (např. "SEC " / "SELC")
583 až 588	pro kalibraci luxmetru (6 bajtů)
589 až 598	rezerva 10 bajtů
599	pro testovací zápis na kontrolu, že EEPROM funguje
600 až 611	kalibrace kompasu (minima a maxima magnetické síly ve všech 3 osách)
612 až 617	kalibrace náklonoměru LSM303DLHC (hodnoty zrychlení os x, y, z v horizontální poloze)
618 až 623	kalibrace náklonoměru LSM303DLHC (hodnoty zrychlení os x, y, z ve vertikální poloze)
624 až 635	časy 5 alarmů (budíků) a jedné odpočtové funkce. Každý cílový čas je uložen jako počet minut od půlnoci.
636	značka pro testovací alarm
637 až 699	rezerva (63 bajtů)

Detail adresy 9

(oddělovače položek a desetinných míst v CSV souborech)

bit 1 a bit 0
'0' '0' ... oddělovač položek mezera
'0' '1' ... oddělovač položek čárka
'1' '0' ... oddělovač položek středník [default]
'1' '1' ... oddělovač položek tabulátor

bit 2 = '0' ... oddělovač desetinných míst: tečka
bit 2 = '1' ... oddělovač desetinných míst: čárka [default]

bit 3 = '0' ... položky mezi oddělovači neuzavírat do uvozovek [default]
bit 3 = '1' ... každou položku mezi oddělovači zavřít do uvozovek

bit 4 = '0' ... nevytvářet hlavičku v souboru
bit 4 = '1' ... na začátku každého souboru vytvořit hlavičku [default]

formát bajtu:
(0bxxxhudoo)
xxx = nevyužito; h = hlavička; u = uzavírání položek do uvozovek;
d = desetinný oddělovač; oo = oddělovač položek

Detail adresy 14

(SLAVE adresa a rychlost komunikace)

bity 0 až 3 = SLAVE adresa pro komunikaci (1 až 15)
bity 4 a 5 = rychlost sériové komunikace (0 až 3)
formát bajtu:
(0bxx00AAAA = 9600)
(0bxx01AAAA = 19200)
(0bxx10AAAA = 38400)
(0bxx11AAAA = 115200)
AAAA = SLAVE adresa; xx = nevyužito (rezerva)

Většina parametrů je uživatelsky nastavitelná pomocí příkazů přes USB sériovou komunikaci, nebo přes menu.

Popis uložených záznamů v EEPROM

Každý záznam se ukládá do EEPROM v minimalizovaném formátu ve stejně dlouhých blocích. Délka bloku závisí na tom, co všechno se má zaznamenávat.
Nastavení zaznamenávaných dat se provádí přímo v programu před kompilací (v soboru "stm411.ino" v části #doc#05). Po každé změně je nutné celou paměť přeformátovat (je na to určený příkaz odesílaný přes USB sériovou linku - HARD formát: **#FH**).

V minimální variantě, kdy se zaznamenává pouze datum, čas, úroveň jasu a informační bajty (informace o způsobu spouštění měření, časová zóna a SLAVE adresa, značka stability), je velikost jednoho záznamu 8 bajtů. v tom případě se do 128kB paměti vejde asi 16 000 záznamů.

V maximální variantě, kdy se ukládá navíc ještě teplota, tlak, vlhkost (musí být osazené čidlo BME280), zeměpisné souřadnice, nadmořská výška (musí být připojen GPS modul), detailní informace o změřeném světle (kanály Infra a Full, zesílení, doba měření) údaje o poloze Slunce a Měsíce, azimut a náklon je jeden záznam dlouhý 36 bajtů a do paměti se tak vejde asi 3 400 záznamů.

Po zaplnění paměti se starší záznamy začnou přepisovat novými naměřenými hodnotami.

Organizace jednoho záznamu světla vypadá následovně:

1. bajt
   - značka obsazeného nebo volného bloku (bit 7),
   - typ záznamu (světlo, stopky, časová značka) (bity 6 a 5)
   - (bit 4 nepoužitý)
   - index přepnutého čidla na rozšiřující desce (bity 3, 2, 1)
   - značka zima (SEČ) / léto (SELČ) (bit 0),

2. až 5. bajt - časový údaj v sekundách od 1.1.1970

6. a 7. bajt - údaj o naměřeném plošném jasu v tisícinách mag/arcsec^2

8. bajt
   - SLAVE adresa zařízeni pro komunikaci (bity 7, 6, 5, 4)
   - značka stability měřeného jasu (bit 3)
   - popis co způsobilo spuštění měření (bity 2, 1, 0) (tlačítko, sériová linka, automat, kalibrace ...)

Volitelně se pak k 8 bajtům přidávají ještě následující údaje:

- informace o teplotě (plus 2 bajty na záznam)
- informace o tlaku (plus 2 bajty na záznam)
- informace o vlhkosti (plus 2 bajty na záznam)
- informace o INFRA složce světla (plus 2 bajty na záznam)
- informace o FULL kanálu světla (plus 2 bajty na záznam)
- informace o nastavení Control registru čidla světla
  (zesílení, doba snímání) (plus 1 bajt na záznam)
- informace o zeměpisných souřadnicích
  (délka, šířka, výška) (plus 10 bajtů na záznam)
- informace o úhlu náklonu v době měření
  (desetiny stupně) (plus 2 bajty na záznam)
- informace o azimutu měření (plus 2 bajty na záznam)
- informace o poloze Slunce a Měsíce a osvětlení Měs. v době měření
  (stupně a procenta) (plus 3 bajty na záznam)

Detailní informace k organizaci 1. bajtu záznamu:

bit 7 = '0' = následující blok bajtů je volný, je možné ho přepsat
bit 7 = '1' = následující blok bajtů je obsazený

bit 6	bit 5	
'0'	'0'	= běžný záznam světla
'1'	'0'	= záznam byl pořízen v režimu "Stopky"
'0'	'1'	= jedná se o záznam časové značky
'1'	'1'	= nemůže nastat (rezervováno pro speciální případy)

bit 4 = nepoužito

bity 3, 2, 1 = index čidla světla na rozšiřující desce
(když není deska použita má vnitřní čidlo index 0)
index je o 1 nižší, než je zobrazované číslo čidla
0bxxxx010x označuje číslo čidla 3

bit 0 = '0' = zaznamenaný čas je v "zimní" zóně (SEČ)
bit 0 = '1' = zaznamenaný čas je v "letní" zóně (SELČ)

Detailní informace k organizaci 8. bajtu záznamu:

bity 7, 6, 5, 4 = SLAVE adresa zařízení (1 až 15) - 0 není použita

bit 3 = '0' = měření světla nebylo stabilní
bit 3 = '1' = měření světla bylo stabilní

V režimu měření světla:
bity 2, 1, 0 = co způsobilo spuštění měření

0bxxxxx000	= nepoužito
0bxxxxx001	= tlačítko [nahoru] (u)
0bxxxxx010	= tlačítko [dolu] (d)
0bxxxxx011	= tlačítko [OK] (o)
0bxxxxx100	= spuštěno komunikací (M)
0bxxxxx101	= spuštěno automatem (A)
0bxxxxx110	= kalibrační měření (c)
0bxxxxx111	= kalibrační průměr (C)

V režimu stopek:
bity 2, 1, 0 = typ záznamu

0bxxxxx000	= nepoužito
0bxxxxx001	= spuštění stopek
0bxxxxx010	= zastavení stopek
0bxxxxx011	= mezičas
0bxxxxx100	= pokračování bez nulování

V režimu rychlé časové značky:
bity 2, 1, 0 = vždycky 0bxxxxx000

V případě povoleného ukládání **Control registru** čidla světla je formát jednotlivých bitů následující:

bity 7 a 6 určují zesílení (AGAIN)
- zesílení se upravuje automaticky podle aktuálního osvětlení.

0b00xxxxxx	=	1x
0b01xxxxxx	=	25x
0b10xxxxxx	=	428x
0b11xxxxxx	=	9876x

bity 5, 4 a 3 jsou nepoužité

bity 2,1 a 0 určují dobu snímání světla (ATIME)

0bxxxxx000	=	100ms
0bxxxxx001	=	200ms

0bxxxxx010 = 300ms
0bxxxxx011 = 400ms
0bxxxxx100 = 500ms
0bxxxxx101 = 600ms [v programu nastavené napevno]
0bxxxxx110 = nepoužito
0bxxxxx111 = nepoužito

Teplota se do EEPROM ukládá v upraveném formátu, aby se vešla do 16-bitového rozsahu:
Změřená hodnota z čidla se nejdřív vynásobí 100x, takže je výsledek celé číslo v setinách °C. K této hodnotě se přičte 50°C (v setinách °C), aby byla zaznamenávaná hodnota vždycky kladná.
Výsledek se pak ještě přepočte přes kalibrační tabulku.
Ukázka kódu:

```
float cidlo_teploty = cidlo_BME.readTemperature();
unsigned int teplota_BME  = 5000 + (100 * cidlo_teploty);
teplota_BME = korekce_teploty(teplota_BME);
```

Příklad:
Teplota z čidla 26,71°C se vynásobí 100×, výsledek je číslo 2671.
K tomuto číslu se přidá 50°C v setinách °C (2671+5000 = 7671).
Pokud není nastavená korekce přes kalibrační tabulku, uloží se číslo 7671 do EEPROM jako 16-bitové číslo.

Vlhkost se do EEPROM ukládá také jako celé číslo, ale protože nemůže být nikdy záporná, stačí hodnotu z čidla vynásobit 100×.
Ukázka kódu:

```
float cidlo_vlhkosti = cidlo_BME.readHumidity();
unsigned int vlhkost_BME = cidlo_vlhkosti * 100;
```

Příklad:
Vlhkost z čidla 51,23% se vynásobí 100×, výsledek je číslo 5123.
Toto číslo se uloží do EEPROM jako 16-bitové číslo.

Protože se normální tlak pohybuje kolem 100 000 Pa, nebylo by možné ukládat hodnotu do EEPROM jako 16-bitové číslo.
Tlak se proto do EEPROM ukládá až po odečtení konstanty 60000 Pa.
Konstanta se definuje v souboru "stm411.ino" v bloku #doc#09 jako 'posun_tlaku'.

Ukázka kódu:

```
#define posun_tlaku          60000
....
unsigned long cidlo_tlaku = cidlo_BME.readPressure();
unsigned int tlak_BME = cidlo_tlaku - posun_tlaku;
```

I **náklon** se do EEPROM ukládá v upraveném formátu.
Nejdřív se hodnota z čidla zaokrouhlí na desetiny stupně.
Pak se připočte 1000° aby byla hodnota vždycky kladná.
Výsledek se pak vynásobí 10×, takže ukládaná hodnota je náklon s rozlišením na desetiny stupně.

Ukázka kódu:

```
float naklon = naklon_z_cidla();
if (naklon < 0)    naklon = (naklon +  999.95) * 10.0;
else               naklon = (naklon + 1000.05) * 10.0;
```

Azimuty (namíření SQM, Slunce, Měsíc) se do EEPROM ukládají jen se zaokrouhlením na celé stupně. Ukládá se tedy hodnota 0 až 359.

Elevaci Slunce a Měsíce bylo nutné dostat každou do 1 bajtu, proto se ukádá jen hodnota ve stupních, která je navíc posunuta o 90°, aby byla vždycky kladná.
Příklady:
Elevace Slunce je -23,7° (23,7° pod horizontem),
do EEPROM se uloží číslo -24 + 90 = 66.
V EEPROM je hodnota 131,
skutečná elevace je 131 - 90 = 41° nad horizontem

Záznamy GPS souřadnic se ukládají v upraveném unsigned long formátu.

Zeměpisná délka se ukládá ve 4 bajtech v miliontinách stupně.
Pro západní polokouli se ukládá číslo přímo, pro východní polokouli se do EEPROM přičítá 180 000 000 (východní polokoule bude vždycky větší než 180 000 000).

Příklady:
v EEPROM je uloženo číslo 194366646 a to znamená 14.366646° v.d.
v EEPROM je uloženo číslo 54366646 a to znamená 54.366646° z.d.

Zeměpisná šířka se ukládá podobně
(také ve 4 bajtech s rozlišením na miliontiny stupně).
Pro jižní polokouli se ukládá číslo přímo, pro severní polokouli se do EEPROM přičítá číslo 90 000 000 (severní polokoule bude vždycky větší než 90 000 000).

V případě použití některého z přednastavených pozorovacích stanovišť se ukládá informace o stanovišti do nejvyšších 3 bitů zeměpisné šířky.

Příklady:
v EEPROM je uloženo číslo 139444356 a to znamená 49.444356° s.š.
v EEPROM je uloženo číslo 19444356 a to znamená 19.444356° j.š.

Nadmořská výška se ukládá jen dvojbajtově. Rozlišení je 1 metr.
Ke změřené hodnotě se ale do EEPROM přidává +500 metrů.

Příklady:
v EEPROM je uloženo číslo 987 - to znamená 487 m n.m.
v EEPROM je uloženo číslo 200 - to znamená 300 m pod mořem.

Stopky

U záznamů pořízených ve funkci stopky záleží i na tom, co se zaznamenává:

V prvním bajtu záznamu se vždycky nastaví bit 6 do '1'
- to je informace, že záznam neobsahuje světlo, ale stopky.

Zbytek bitů prvního bajtu je uložený stejně jako při záznamu světla.

Stopky - START

Při startu stopek je ve 2. až 5. bajtu uložen datum a čas spuštění stopek v sekundách od 1.1.1970 (tak jako u záznamu světla)

6. bajt se naplní hodnotou 0xFF.

7. bajt obsahuje aktuální počítadlo mezičasů

Do 8. bajtu se zapíše číslo 1 (značka pro START).

Ostatní bajty (teplota, tlak, GPS ...) se neukládají - nic se neměří.

Stopky - LAP (mezičas)

Do 2. až 5. bajtu se ukládá čas v milisekundách od spuštění stopek.

6. bajt se naplní hodnotou 0xFF.

7. bajt obsahuje počítadlo mezičasů.
Toto počítadlo se nuluje společně s nulováním stopek.

Do 8. bajtu se zapíše číslo 3 (značka pro mezičas).

Stopky - STOP

Do 2. až 5. bajtu se ale ukládá čas v milisekundách od spuštění stopek.

6 bajt se naplní hodnotou 0xFF.

7 bajt obsahuje počítadlo mezičasů.
Při zastavení stopek se toto počítadlo také zvětší o 1,
stejně jako u mezičasů.

Do 8. bajtu se zapíše číslo 2 (značka pro STOP).

Stopky - CONT

Poslední typ záznamu pro režim "Stopky" je záznam "Cont".
K tomuto záznamu dochází, když při spuštění stopek nebyl předchozí zaznamenaný čas vynulovaný.
Záznam v EEPROM je stejný jako při startu, akorát do 8. bajtu se zapisuje číslo 4 (značka pro pokračování bez nulování).
Počítadlo mezičasů také pokračuje v řadě bez vynulování.

Příklady uložených záznamů stopek v EEPROM (data v HEX formátu)

START:
```
C0 60 39 5F 8E FF 00 01 00 00 00 00 00 00 00 00 00 00 00
```

Mezičas 1 (LAP):
```
C0 00 00 0D 52 FF 01 03 00 00 00 00 00 00 00 00 00 00 00
```

Mezičas 2 (LAP):
```
C0 00 00 18 46 FF 02 03 00 00 00 00 00 00 00 00 00 00 00
```

Mezičas 3 (LAP):
```
C0 00 00 23 34 FF 03 03 00 00 00 00 00 00 00 00 00 00 00
```

STOP:
```
C0 00 00 2F 29 FF 04 02 00 00 00 00 00 00 00 00 00 00 00
```

Pokračování bez vynulování (CONT):
```
C0 60 39 5F 9E FF 04 04 00 00 00 00 00 00 00 00 00 00 00
```

STOP:
```
C0 00 00 39 AC FF 05 02 00 00 00 00 00 00 00 00 00 00 00
```

Základní popis programu a jeho nastavení

Program je psaný v jazyku Wiring pro prostředí Arduino IDE.
Do Arduino IDE je nutné doinstalovat podporu pro procesory STM32 a některé knihovny pro čidla (ke stažení v příloze).

Celý program je rozdělen na několik samostatných souborů:

astro.ino	podprogramy pro výpočty poloh Slunce a Měsíce
bme280.ino	podprogramy pro obsluhu čidla BME280
displej.ino	podprogramy pro obsluhu 7-segmentových displejů
ds3231.ino	podprogramy pro řídavný RTC obvod s funkcí budíku
eeprom.ino	podprogramy pro čtení a zápis do EEPROM
expander.ino	podprogramy pro ovládání rozšiřující desky
gps.ino	podprogramy pro práci s GPS
jazyky.h	jazykové verze (zatím jen CZ - angličtina nedodělána)
kompas.ino	podprogramy pro náklonoměr s kompasem LSM303
konverze.ino	podprogramy pro převod dat z EEPROM na čitelné textové řetězce s konstantní délkou (pro CSV soubory nebo textové výstupy)
led.ino	obsluha LED pro verzi bez displeje (různé blikání)
menu_dis.ino	kompletní podprogramy pro menu s displejem
menu_LED.ino	zjednodušené menu pro verzi bez displeje (s LED)
mereni.ino	hlavní podprogram pro různé varianty spouštění měření
rezervy.ino	připravené funkce pro rozšíření záznamů o další 4 měřené hodnoty (UV, vítr, sníh ....)
rezim30s.ino	rozšíření o 30-sek. záznam počasí do druhé EEPROM
rs485.ino	komunikace přes RS485
rtc.ino	obsluha vnitřních hodin včetně autokalibrace
sd_karta.ino	podprogramy pro práci s SD kartou
ser_kom.ino	USB sériová komunikace
stm411.ino	základní nastavení systému, hlavní smyčka, výpisy záznamů, formátování EEPROM, systémové informace...
stopky.ino	speciální podprogram pro stopky
svetlo.ino	základní podprogramy pro měření jasu
test.ino	podprogramy pro testovací funkce HW
timestamp.ino	podprogram pro záznam časové značky

Značené části programu

Některé části programu určené pro úpravy jsou označeny unikátním komentářem ve formátu:

```
// #doc#nn
```

kde "nn" je pořadové číslo.

Zde je seznam a popis těchto částí:

`// #doc#01` soubor "stm411.ino"

Seznam osazených periférií.
Ve webovém konfigurátoru se tento blok generuje automaticky jako textový soubor podle zaškrtnuté varianty osazení.
Text z konfigurátoru se před kompilací překopíruje do programu.
Periférie, které jsou ozančené jako neosazené se pak v programu vůbec nepoužívají.

`// #doc#02` soubor "stm411.ino"

Přídavná I2C čidla. Zatím nepoužitá. Jsou to jen připravené rezervy.
Každé z těchto čidel může ukládat do EEPROM dva bajty.
Odkomentováním čidel se zpřístupní měřící podprogramy v souboru "rezervy.ino". Detailní postup doplňování čidel je popsán v kapitole "Připravené rozšíření o další 4 čidla".

`// #doc#03` soubor "stm411.ino"

Výběr jazyka CZ / EN. Použitý jazyk se odkomentuje,
nepoužitý jazyk se zakomentuje.
Definice se projeví v souboru "jazyk.h", kde je pro každý jazyk zvolen patřičný blok textů, grafických definic znaků pro displej a ovládací příkazy pro sériovou linku.
Podrobnosti v kapitole "Jazyky"

`// #doc#04` soubor "stm411.ino"

Nastavení analogových úrovní pro signalizaci stavu baterie.
Detailní popis v kapitole "Nastavení úrovní pro test stavu baterie"

// #doc#05 soubor "stm411.ino"
// #doc#06 soubor "stm411.ino"
Seznam měřených hodnot, které se ukládají do EEPROM.
Program je přednastaven pro ukládání všech hodnot (i z neosazených čidel). Zakomentováním je možné některé z volitelných hodnot přestat ukládat. Zvětší se tím počet záznamů, které je možné v EEPROM zachovat.

V #doc#06 jsou rezervní čidla, která se defaultně neukládají.
Po zaplnění EEPROM se staré záznamy začnou přepisovat.
Při změně v těchto blocích je nutné provést "HARD FORMÁT" příkazem "#FH" přes sériovou linku.

// #doc#07 soubor "stm411.ino"
Nastavení počtu případně použitých rezervních čidel. Když jsou nějaká čidla použita, zobrazí se při výpisu v "sys_info()" a mění se i velikost tisknuté hlavičky do CSV souborů na SD kartě.

Při hodnotě 0 se skryjí všechny 4 rezervy.
Při hodnotě 1 se zobrazí rezerva 1 a ostatní rezervy se skryjí.
Při hodnotě 2 se zobrazí rezervy 1 a 2 a ostatní rezervy se skryjí.
Při hodnotě 2 se zobrazí rezervy 1, 2 a 3 a rezerva 4 se skryje.
Při hodnotě 4 se zobrazí všechno (nic se neskrývá)

// #doc#08 soubor "stm411.ino"
Doladění AGING OFFSET registru v obvodu DS3231.
Detaily nastavení v katalogovém listu DS3231.

// #doc#09 soubor "stm411.ino"

Posun atmosférického tlaku, který je ukládán do EEPROM.
Tlak se obvykle pohybuje kolem 100kPa. Takhle velké číslo by se ale v EEPROM nevešlo do 2 bajtů. Proto se od změřeného tlaku odečítá konstanta 60000Pa. Před zobrazením, nebo uložením do CSV souboru se zase tato konstanta přičte.
Pokud by se SQM používalo v místech, kde je absolutní atmosférický tlak nižší než 60kPa (na vysokých horách), způsobovalo by to problémy s podtečením výpočtu tlaku do záporných čísel. V tom případě je nutné konstantu snížit.
V opačném případě (pokud by se SQM používalo v místech, kde je absolutní atmosférický tlak vyšší než 125535Pa) by bylo nutné konstantu zvýšit (maximálně ale na hodnotu 65535). Pak by bylo možné měřit tlak až do hodnoty 65535 + 65535Pa = 131070Pa. Při vyšším tlaku by došlo k přetečení rozsahu.
Normálně by ale takový extrém nastat neměl.
Rekordní tlak (přepočtený na hladinu moře) byl necelých 109kPa (Mongolsko 2004). Když se k tomu přičte ještě teoretických 430m pod hladinu moře (Mrtvé moře), tak by absolutní tlak neměl nikdy překročit s rezervou 115kPa.

// #doc#10 soubor "stm411.ino"

Definice adres v systémové části EEPROM.
Při úpravách programu je možné použít adresy označené jako rezervy.
Při každé úpravě ake nezapomenout aktualizovat verzi rozložení adres v paměti.
Označení je použito při zálohování nastavení na SD kartu a obnovování nastavení zpátky do EEPROM. Když se verze v programu neshoduje s verzí v zálohovacím souborum, k obnovení zálohy nedojde.

// #doc#11 soubor "stm411.ino"
// #doc#14 soubor "menu_dis.ino"

Nastavení přepočtu pro měření napětí hlavního zdroje (9V baterie).
Ve vzorci je možné upravit přesnou hodnotu děliče napětí z odporů R7 a R8, aby byla zobrazovaná hodnota přesnější.

```
//  DZ1=1.8V                                                R8   / R7+R8
    ((((1.8 / ref_in) * 10 * Vcc * bat_in) / 4096.0) / (8.2 / 108.2)) + 0.5;
```

// #doc#12 soubor "stm411.ino"

Defaultní parametry, které se uloží do EEPROM při prvním zapnutí, nebo po spuštění příkazu "**@DP**" přes sériovou linku.
Jedná se hlavně o možnost změny defaultní letní a zimní časové zóny včetně popisků a domácích zeměpisných souřadnic pro astrovýpočty pro případ, že bude SQM použito v cizině.

// #doc#13 soubor "stm411.ino"

Podprogram, který poskládá poslední změřený jas a teplotu do formátu, kterému rozumí "Unihedron Device Manager".
Je to odpověď do sériové linky na příkaz "**R**" nebo "**rx**".
Výstup vypadá nějak takto:

```
r, 10.77m,0000022921Hz,0000000020c,0000000.000s, 24.3C
```

(červeně je označený poslední změřený jas a zeleně poslední teplota)

// #doc#15 soubor "ser_kom.ino"

Odeslání identifikace SQM do programu Unihedron Device Manager (UDM) po sériovém příkazu "**ix**".
Tady se jedná hlavně o změnu sériového čísla, které se podvrhne do UDM. Detaily jsou uvedeny v manuálu k UDM.
Výstup má následující formát:

```
i,00000002,0000003,00000001,00012345
```

Červeně je zvýrazněn "*Protocol number*".
Zeleně "*Model number*".
Modře "*Feature number*".
Fialově "*Serial number*".

// #doc#16 soubor "stm411.ino"

Konfigurace pořadí přenosu barevných složek do neopixelové LED.
LED funguje tak, že si po komunikaci přečte 24 bitů, které obsahují jas jednotlivých barevných složek.
Každá barva má jas zakódovaný do 8 bitů.
Problém je, že různé LED mohou mít přeházené pořadí těchto složek.
Program je připraven pro 2 verze komunikace s LED.
Odesílání v pořadí složek R-G-B, nebo pořadí G-R-B.
Volba pořadí se provádí odkomentováním požadované řádky:

```
Adafruit_NeoPixel strip = Adafruit_NeoPixel(1, pin_NEO, NEO_GRB + NEO_KHZ800);
Adafruit_NeoPixel strip = Adafruit_NeoPixel(1, pin_NEO, NEO_RGB + NEO_KHZ800);
```

// #doc#17 soubor "ser_kom.ino"

Ruční oprava jména souboru ve složce /HELP/ na SD kartě pro případ, že by byl kód fukce znak, který v souborovém systému SD karty zakázaný (<,>,*,",?).
Detaily v kapitole "Nápovědy na SD kartě".

Když je příkaz zakázaný znak, musí se v kódu ručně přepsat na povolený, ale nepoužitý znak.

Příklad náhrady jména souboru pro funkci **@>**
Znak '>' je ve jméně souboru zakázaný. Proto je nutné vytvořit jiné jméno souboru, v kterém se znak '>' přepíše nějakým nepoužitým znakem ('[').

```
    case '>':                       // puvodni jmeno souboru "2_>.txt" pro
                                    //   funkci "@>" se nahradi za "2_[.txt"
      helpfile_name[7] = '[';       // znak '[' ale nesmi byt pouzit pro
                                    //   jinou funkci
      break;
```

// #doc#18 soubor "rezim30s.ino"

Možnost nastavení desetinných oddělovačů v CSV souborech pro záznamy dat v režimu "R30s".

```
// #doc#18
      vystupni_retezec30[52] = ',';   // desetinny oddelovac v teplote
      vystupni_retezec30[64] = ',';   // desetinny oddelovac ve vlhkosti
      vystupni_retezec30[90] = ',';   // desetinny oddelovac v osvetleni
```

Připravené rozšíření o další 4 čidla

Program je připraven na měření a záznam dalších 4 hodnot, které se dají získávat například z bočního konektoru přes I2C.
Hodnoty jsou už teď stažitelné přes modbus, ale když nejsou čidla použita, vrací se v rezervních registrech [30028] až [30031] čísla 65535

Pro budoucí zprovoznění je nutné:

---

1) v souboru "hlavni411.ino" v bloku #doc#02 odkomentovat použitá čidla na řádkách:

```
//#define cidlo_rezerva_1                   // cidlo _____
//#define cidlo_rezerva_2                   // cidlo _____
//#define cidlo_rezerva_3                   // cidlo _____
//#define cidlo_rezerva_4                   // cidlo _____
```

a dopsat do komentářů, o jaká čidla se jedná

---

2) v souboru "hlavni411.ino" v bloku #doc#06 povolit ukládání do záznamů do EEPROM na řádkách:

```
//#define ukladat_rezerva_1    // do EEPROM se bude ukladat informace o ...
//#define ukladat_rezerva_2    // do EEPROM se bude ukladat informace o ...
//#define ukladat_rezerva_3    // do EEPROM se bude ukladat informace o ...
//#define ukladat_rezerva_4    // do EEPROM se bude ukladat informace o ...
```

---

3) v souboru "hlavni411.ino" v bloku #doc#07 zadat počet zobrazených rezervních čidel na řádce:

```
#define zobraz_rezervy           0 // zobrazi nebo skryje rezervni cidla
      // pri hodnote 0 se skryji vsechny rezervy
      // pri hodnote 1 se zobrazi rezerva 1, a ostatni se skryji
      // pri hodnote 2 se zobrazi rezervy 1 a 2, a ostatni se skryji
      // pri hodnote 3 se zobrazi rezervy 1, 2 a 3, a rezerva 4 se skryje
      // pri hodnote 4 se zobrazi vsechno (nic se neskryva)
```

---

4) v souboru "jazyk.h" přepsat hlavičky na skutečně měřené hodnoty:

```
char hlavicka[]      = ".........; rez.1 ; rez.2 ; rez.3 ; rez.4 \0";
```

(například na: `= ".........;   UV  ;  snih ;  vitr ; barva \0";`

Dodržet přesnou šířku 5 znaků na každý sloupec.

---

5) v souboru "jazyk.h" upravit popisky do "sys_info" na řádkách:

```
#define lng305    "  - rezerva 1            "
#define lng306    "  - rezerva 2            "
#define lng307    "  - rezerva 3            "
#define lng308    "  - rezerva 4            "
```

Dodržet zarovnání s ostatními čidly (na začátku řetězce dvě mezery, pak pomlčka, mezera a popis čidla).

---

6) v souboru "rezervy.ino" pro použitá čidla doplnit podprogramy pro získávání "unsigned int" hodnot i I^2C komunikace:

```
//-----------------------
// tady cist data pres I2C z nejake adresy a registru
//-----------------------
return 11111;       // vrati zmerenou hodnotu
                    // (pro testovani je tu natvrdo hodnota 11111)
```

---

7) V případě, že by bylo třeba zobrazovat v tabulkách (CSV souborech) místo `unsigned int` rozsahu hodnot něco jiného (znaménkové `int / float`), musely by se upravit části konverzního podprogramu v souboru "konverze.ino":

```
#ifdef ukladat_rezerva_1
  unsigned int pom_rezerva_1;
  pom_rezerva_1 = (EEPROM_read(indexpole) * 256) + EPROM_read(indexpole+1);
  formatuj_string("     "     , 257,  5);
  formatuj_ulong(pom_rezerva_1 , 261,  5);
  indexpole = indexpole + 2;
#else  // kdyz se rezerva neuklada do EEPROM, zobrazi se jen pomlcky
  formatuj_string("-----"     , 257,  5);
#endif
```

Místo podprogramu `formatuj_ulong(...)`
by pak byly použité podprogramy
`formatuj_int(...)` nebo `formatuj_float(...)`

Nastavení úrovní pro test stavu baterie

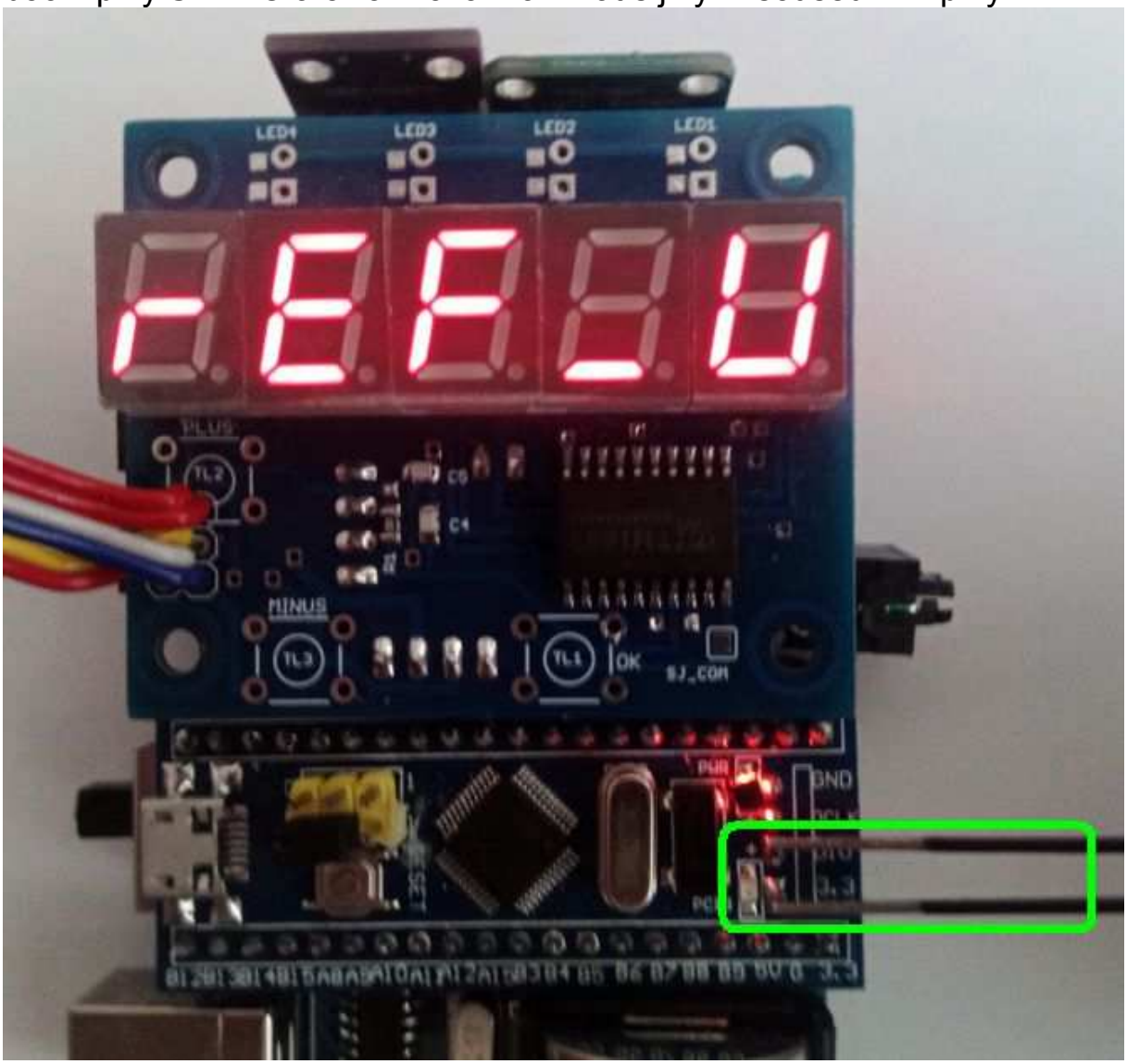
Program je přednastavený na signalizaci několika úrovní stavu baterie. Tyto úrovně ale závisí na přesnosti odporů v napěťovém děliči R7, R8. Pokud by bylo nutné úrovně posunout, postupuje se takto:

- Odpojit napájení přes USB konektor.

- Místo baterie připojit regulovatelný zdroj.

- Na zdroji nastavit takové napětí, které má signalizovat hranici mezi blikáním 1 a 2 teček na displeji (při vyšším napětí bude blikat 1 tečka, při nižším napětí 2 tečky). Program je přednastavený na hodnotu 7V.
- Zapnout hlavní vypínač

- Pomocí šroubováku nebo pinzety zkratovat na procesorové desce boční piny SWDIO a 3V3. Pozor na zkrat s jinými sousedními piny.



Obrázek je ještě z původní verze SQM s deskou BluePill. Novější deska BlackPill s procesorem STM32F401 (nebo 411) má ale piny umístěné stejně.

Na displeji se na objeví nápis "rEF_U" následovaný číslem a po chvíli nápis "U_bAt" následovaný jiným číslem.
Obě tato čísla se někam poznamenají.

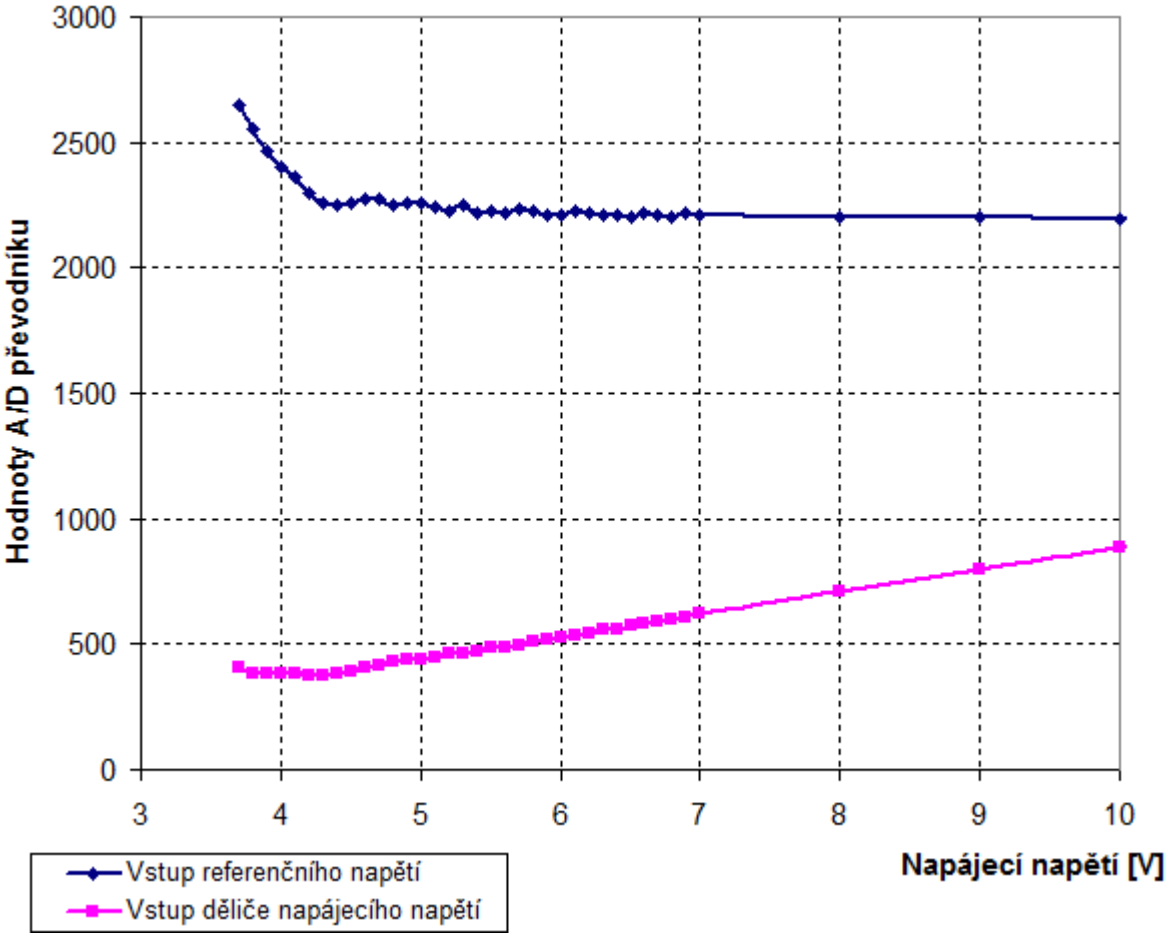
- Postup se opakuje pro další úrovně vstupního napájecího napětí
  (pro mez mezi 2 a 3 blikajícími tečkami: 6V)
  (pro mez mezi 3 a 4 blikajícími tečkami: 5V)
  (pro mez mezi 4 a 5 blikajícími tečkami: 4V)

- Zaznamenané hodnoty by měly vypadat nějak takto:

7V	rEF_U = 2208	U_bAt = 620
6V	rEF_U = 2214	U_bAt = 528
5V	rEF_U = 2258	U_bAt = 441
4V	rEF_U = 2401	U_bAt = 382

- Zkontrolovat, že se hodnota "rEF_U" mezi 5V a 4V výrazně zvýšila a že se hodnota "U_bAt" od 7V do 5V postupně snižuje.

Při detailním proměření by vypadal graf zjištěných hodnot zhruba takto:



Při 4V už přestává fungovat Step-Down zdroj (AP1501) a tím dochází ke snížení napájecího napětí procesoru. Protože vnitřní A/D převodník v procesoru používá jako referenci právě napájecí napětí, začnou obě měřená napětí při tomto stavu vykazovat vyšší hodnoty, než ve skutečnosti mají.Výrazný nárůst na vstupu s referenčním napětím je pak signálem pro rozblikání všech 5 teček na displeji (stav velice slabé baterie).

- Zjištěné hodnoty "U_bAt" pro 7V, 6V a 5V se pak přepíší do programu (soubor "stm411.ino" #doc#04). Hodnota "rEF_U" pro 4V se přepíše na konec bloku.

```
44 // nastaveni analogovych urovni pro test baterie (meri se za odporovym delicem R7, R8, R6)
45 //     merici rozsah je 0 az 3,3V a to odpovida cislu 0 az 4095.
46 //  Uvedene meze vychazi ze skutecne zjistenych hodnot - v priloze je dokument "test_baterie.
47 #define bat_level_3     620   // (7V) nad level_3 je vyborny stav baterie
48 #define bat_level_2     528   // (6V) mezi level_3 a level_2 (7V az 6V)je dobry stav baterie
49 #define bat_level_1     441   // (5V) mezi level_2 a level_1 (6V az 5V) je stredni stav bate
50                               // pod mezi level_1 (pod 5V)
51 #define ref_level      2401   // kdyz je napajeni tak nizke, ze referencni napeti prekroci t
52
```

- Upravený program se pak znova nahraje do procesoru

Poznámka:
Při napájení SQM přes USB se dělič napětí z odporů R7 a R8 obchází, a proto je zobrazené napětí hlavní baterie "U." někde mezi 16 až 17V.

Jazyky

Případné změny textů, které vypisují přes sériovou linku, nápisy na displeji a ovládací příkazy pro USB sériovou linku se provádí v souboru "jazyky.h".

Tento soubor je rozdělen na 2 části. Blok CZ a blok EN.
Je ale možné přidávat i další jazyky. Každý takto vytvořený jazyk musí mít ale v #doc#03 vlastní řádku s definicí.

V každém jazykovém bloku jsou nastavené texty hlaviček.
U těch je nutné dodržet přesnou šířku jednotlivých sloupců i za cenu zkracování nápisů.

Dále soubor obsahuje texty, které se vypisují do sériové linky.
Ve většině případů tam není žádné omezení na délku.
Některé texty se ale vypisují do sériové linky ve formě přehledné tabulky a tam je třeba si dát pozor na správné zarovnání (doplnění mezerami zleva - někdy i zprava).

Příklady nutného dodržení zarovnání některých položek:

```
#define lng150    " obsahuje tyto polozky: "
#define lng151    "  - plosny jas             ANO"
#define lng152    "  - teplota                "
#define lng155    "  - tlak                   "

#define lng339    "@L           ... vypnout / zapnout RGB LED"
#define lng282    "     @Lb (B) ... vypnout (zapnout) LED pro test baterie"
#define lng283    "     @Lm (M) ... vypnout (zapnout) LED mereni"
#define lng284    "     @Le (E) ... vypnout (zapnout) LED pri chybe"

#define lng_test_001     "a .... sken I2C"
#define lng_test_002     "b .... test A/D"
#define lng_test_003     "c .... test EEPROM 128k"
#define lng_test_004     "d .... test extra EEPROM (r30s)"
```

V bloku popisků vkládaných do CSV záznamů je nutné dodržet přesný počet znaků.

```
#define lng254    " STABIL "            // znacka stabilniho jasu
#define lng255    " NESTAB "            // znacka nestabilniho jasu
#define lng256    "tl_UP"               // znacka, ze bylo mereni spuste...
#define lng257    "tl_DN"               // znacka, ze bylo mereni spuste...
```

Následují definice ovládacích kódů pro USB sériovou linku.
Kódy jsou rozděleny do několika skupin podle prvního ovládacího kódu:

Příkazy bez kódu jsou ve skupině `#define USB_fce_1_nn`
Příkazy, které začínají znakem # jsou ve skupině `#define USB_fce_2_nn`
Příkazy, které začínají znakem @ jsou ve skupině `#define USB_fce_3_nn`
Příkazy, které začínají znakem % jsou ve skupině `#define USB_fce_4_nn`

Příkazy, které začínají znakem *, které se používají pro komunikaci s PC programem jsou nastavené na pevno pro všechny jazyky a není možné je upravovat.

V definicích ovládacích znaků je třeba dodržet pravidlo, že v každém sloupci je použit unikátní znak.Pak se nemůže stát, že by dvě funkce měly stejné ovládací kódy.

```
#define USB_fce_3_10 '@'                    // ... @@ = reset
#define USB_fce_3_11 'G'                    // ... @G = vypis dat z GP...
#define USB_fce_3_19          's'           // ... @Gs = nastaveni dom...
#define USB_fce_3_20          'd'           // ... @Gd = nastaveni dom...
#define USB_fce_3_21          'z'           // ... @Gz = nastaveni dom...
```

Při úpravách ovládacích kódů nezapomenout provést i úpravy v příslušných řádkách nápovědy:

```
#define lng076    "@@           ... softwarovy RESET"
#define lng208    "@G           ... vypis dat z GPS modulu"
#define lng240    "@Gs nnn      ... domaci zemepisna sirka pro astro...
#define lng241    "@Gd nnn      ... domaci zemepisna delka pro astro...
#define lng242    "@Gz nn       ... domaci zimni casova zona"
```

Úpravy ovládacích kódů se musí zanést i do souborů ve složce /HELP/ na SD kartě. Detaily o rozšířené nápovědě na SD kartě popsány v odstavci "Nápovědy na SD kartě".

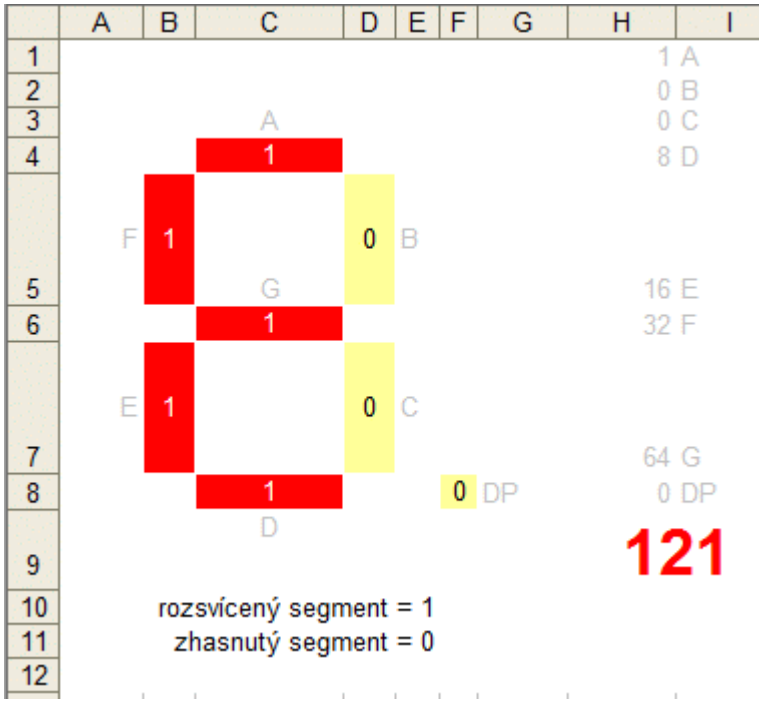
Soubor "jazyky.h" obashuje i nekolk definic řetězcových polí. Tam je nutné dodržet správný počet znaků.

```
(2 znaky na položku)
String dny[] = {"--", "Po" , "Ut" , "St" , "Ct" , "Pa" , "So" , "Ne"};

(8 znaků na položku)
String planety[] = {"------- ", "Merkur  " , "Venuse  " , "Mars    " , ...

(10 znaků na položku - kromě nulté položky, která se nepoužívá)
String poz_st_txt[] = {"--", "Poz.Stan.1" , "Poz.Stan.2" , "Poz.Stan.3" ...
```

Řetězcové pole "popisy3231[]" je bez omezení délek - jsou to popisy, které se přidávají na konce rádek k výpisu registrů RTC obvodu do sériové linky.

Nápisy na displeji jsou nadefinované pomocí 5 čísel (pro každou sedmisegmentovku jednotku jedno). Číslo vyjadřuje binární kód pro rozsvícené segmenty každé sedmisegmentovky.
Pro jednodušší přepočet je v příloze Excelovský dokument "fontgen.xls", ve kterém se na segmenty, které mají svítit, zapíše číslo 1.
Na segmenty, které mají být zhasnuté, se zapíše číslo 0.
Program pak automaticky potřebné číslo vypočítá.

Ukázka pro velké 'E':



Sekvence takto získaných 5 čísel se pak zadává do programu (do souboru "jazyky.h") oddělená čárkami a uzavřená do složených závorek.
Pro nápis "Err-3" vypadá zápis takto:

```
{ 121,  80,  80,  64,  79} , // "Err-3"
```

Podobným způsobem jsou nadefinovány i samostatné znaky pro zobrazení a nastavování položek v menu.
Tady se ale jedná jen o jedno číslo pro jednu zobrazovací jednotku:

```
byte znak_V =       62      ;  // "V"   zobrazeni vlhkosti
byte znak_A =      119      ;  // "A"   nastaveni automatu
byte znak_P =      115      ;  // "P"   nastaveni prumerovani
```

Nápovědy na SD kartě

Na SD kartě jsou ve složce /HELP/ umístěné soubory s detailními nápovědami pro každý příkaz odesílaný do sériové linky.

Nápověda z těchto souborů se vyvolá příkazem **?**, za kterým následuje jeden nebo dva znaky příslušné funkce.

Zadaný kód se převede na tříznakové jméno souboru a obsah souboru se pak vypíše do sériové linky.

První znak ve jméně souboru je číslo, které udává skupinu funkcí:

- 0 = základní skupina bez úvodního znaku
- 1 = skupina, která začíná znakem '#'
- 2 = skupina, která začíná znakem '@'
- 3 = skupina, která začíná znakem '%'
- 4 = skupina, která začíná znakem '*'

Druhý znak ve jméně souboru udává, jestli je ovládací kód velké, nebo malé písmeno (případně nepísmenný znak)

- l (malé L) = kód je malé písmeno (lower case)
- u = kód je velké písmeni (upper case)
- _ = kód je nepísmenný znak

Poslední znak v souboru je konkrétní ovládací znak (vždycky jako malý znak).
Ve speciálních případech, kdy je jako znak použit zakázaný kód (/ " \ > < ? *) se tento kód převádí na nějaký povolený znak (například podtržítko).

Speciální případ souboru je "mmm.txt", který se zobrazí po příkazu **??** a obsahuje seznam všech funkcí.

Příklady jmen souborů ve složce HELP/ na SD kartě:

0ln.txt	příkaz pro nápovědu	**?n**	funkce **n**
1um.txt	příkaz pro nápovědu	**?#M**	funkce **#Mk, #Mo**
2up.txt	příkaz pro nápovědu	**?@P**	funkce **@P ...**
2_@.txt	příkaz pro nápovědu	**?@@**	funkce **@@**
2__.txt	příkaz pro nápovědu	**?@>**	funkce **@>**
3ur.txt	příkaz pro nápovědu	**?%R**	funkce **%R**
4lf.txt	příkaz pro nápovědu	**?*f**	funkce ***f**
mmm.txt	příkaz pro nápovědu	**??**	funkce **??**

Nahrání programu a jeho případná aktualizace

Je detailně popsáno v návodu pro výrobu "sqm_vyroba.doc" (včetně přípravy vývojového prostředí)

Změny v návodu

20.1.2025
- doplnění odkazu na úpravy desetinných oddělovačů (#doc#18)

18.12.2024
- První zveřejněná verze návodu pro procesor STM32F4x1.