světelné panely.cz

FORMÁTY DAT

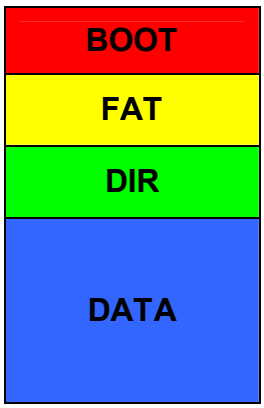
Obsah:

1. FORMÁT PAMĚTI PANELU

Paměť EEPROM slouží pro uchování programu a to i při vypnutí panelu. Paměť umožňuje asi 100000 přepisů, z tohoto důvodu není vhodná pro časté přepisování programu. Pokud je potřeba neustále program měnit je lepší program zapisovat přímo do RAM paměti. Paměť EEPROM je sériová a proto není možné, aby program běžel přímo z této paměti (čtení je pomalejší a zatěžuje procesor), program je při spuštění přesunut do RAM paměti. Je to podobné jako u PC disketa – EEPROM a operační paměť – RAM.

Paměť pro programy je typu EEPROM. Je rozdělena na segmenty o velikosti 256 bajtů. Těchto segmentů může být až 256. Formát paměti je podobný jako například u diskety (FAT velikost položky ve FAT tabulce je 8bitů. Paměť může mít až 64k).

První segment paměti obsahuje informace o velikosti paměti, druhý obsazenost segmentů a třetí tabulku pro jména programů. Zbylé segmenty můžou být použity pro data programů. Jeden program obsadí vždy minimálně jeden segment.



Informace o paměti

Tabulka obsazenosti paměti

Jména programů

Data programů

1.1 BOOT

První segment obsahuje informace o velikosti paměti, jméno programu, který se má spustit po zapnutí panelu a intenzitu svitu, jakou bude panel svítit po zapnutí. Adresa je 0x0000 – 0x00FF (segment číslo 0).

BOOT záznam			
Adresa	**Délka**	**Hodnota**	**Popis**
0 - 7	8	0x00	Volné
8	1	0x80	Celkový počet segmentů paměti (128 segmentů)
9	1	0x01	Nevyužito – Počet sektorů v jedno segmentu
10	1	0x01	Číslo segmentu pro FAT
11	1	0x02	Číslo segmentu pro DIR
12	1	0x00	Volné
13	1	0x80	Počet segmentů jedné paměti (jedna paměť 128)
14 - 15	2	0x00	Volné
16 - 25	10		Jméno spouštěcího programu
26 - 47	21	0x00	Volné
48 - 57	10		Jméno druhého programu
58 - 141	83	0x00	Volné
142	1		MSB Horních 8 bitů kontrolního součtu BOOT

143	1		LSB Dolních 8 bitů kontrolního součtu BOOT
144 - 239	95		Nepožito
Intenzita svitu panelu			
240	1	0x64	Svit panelu (100)
241	1	0x64	Kopie svitu panelu (100)
242	1	0x64	Kopie svitu panelu (100)
242	1	0x2C	Součet hodnot bajtů 240 + 241 + 242 (svit * 3)
243 - 255			

1.2 FAT

Do Fat tabulky se zapisuje pořadí segmentů, ve kterých je program uložen, tím se zároveň určuje, který segment je obsazen. Číslo prvního segmentu programu je uloženo v „DIR“ u jména daného programu. Pozice segmentu je shodná z pozicí ve FAT tabulce (segment 5 = adresa 0x0500 – 0x05FF).

Tabulka významu hodnot ve FAT tabulce.

Hodnota	Popis
0x00	Segment je volný.
0x01	Poslední segment programu.
0x02 – 0xFF	Číslo dalšího segmentu programu

1.3 DIR

Jeden záznam „DIR“ obsahuje jméno, velikost a číslo prvního segmentu uloženého programu. Délka záznamu je 16 bajtů, a v jednom segmentu může být 16 záznamů. Maximálně může mít direktorář 255 záznamů, což je 16 segmentů. Po naformátování programem „Panel2“ je pro DIR vyhrazen jeden segment hned za FAT (od adresy 0x0200). Ve FAT tabulce má bajt s adresou 0x02 hodnotu 0x01, to je obsazený segment a poslední v řetězci segmentů. Je li třeba seznam jmen rozšířit, musí se najít volný segment (většinou první volný). Číslo tohoto segmentu se zapíše do FAT tabulky na adresu 0x02, a na pozici volného bloku ve FAT se zapíše 0x01. Přiřazený segment se vynuluje.

Pro vymazání programu stačí vynulovat první bajt ve jméně programu. Dále se musejí všechny segmenty programu uvolnit a to tím, že ve FAT tabulce se nastaví na 0x00.

Jeden záznam jména souboru

Adresa	Popis
0 – 9	Jméno programu, je-li kratší než 10 znaků zakončené nulou.
10	Atribut 0 = zobrazovací program. 1 = řídící program. 2 = ini. soubor
11	Volné
12	MSB skutečná velikost programu v bajtech.
13	LSB skutečná velikost programu v bajtech.
14	Číslo prvního segmentu programu.
15	CRC - Součet všech bajtů záznamu (0 – 14).

1.4 DATA

Segmenty volné pro zápis programů. Jeden segment má velikost 256 bajtů z toho 254 bajtů obsahuje data, poslední dva bajty jsou kontrolní součet segmentu.

1.5 Zápis programu do panelu.

1. prohledání direktoráře, jestli již program tohoto jména neexistuje.
   a. Pokud již existuje musí se nastavit ve FAT sektory programu na nulu.
2. výpočet, zda se program do paměti vejde
3. nalezení prvního volného segmentu.
4. uložení 254 bajtů včetně CRC do volného segmentu
   a. nastavení ve FAT na číslo dalšího volného segmentu.
   b. Postupně uložit do jednotlivých segmentů, poslední segment ve FAT nastavit na 1.
5. vytvoření položky direktoráře.
   a. Jméno programu.
   b. Atribut.
   c. Velikost programu v bajtech
   d. Číslo prvního segmentu
   e. Kontrolní součet
6. zapsání položky direktoráře. Pokud program existoval, přepíše se původní záznam, v opačném případě zapsat do volného místa. Pokud je direktorář plný, musí se přidat za poslední segment direktoráře nový segment.

Oba zdrojové výpisy jsou pouze názorné a neobsahují výpis všech funkcí. Také v nich nejsou ošetřeny chyby v komunikaci. Například při zápisu je možné doplnit volbu přepsat nebo přeskočit v případě že soubor již existuje. Funkce pro správu paměti EEPROM jsou obsaženy v knihovně PDriver.dll, popis naleznete v „***Popis funkcí knihovny PDriver.dll***“

```
void WriteFile( char *name, BYTE *data, int size )
{
        // načteni FAT a direktoráře

    BYTE record[16];            // záznam v direktoráři
    record[10] = 0;             // atribut zobrazovacího programu
    record[12] = size >> 8;     // velikost souboru
    record[13] = size;
        // hledá toto jméno v direktoráři
        // vrací index jména, pokud neexistuje vrací -1
    int iName = SearchName( name );
        // vymazaní existujícího souboru
    if (iName != -1) DeleteFile( iName );
    int     seg = SearchFree( );    // hlede volny segment;
    record[14] = seg;           // první segment do direktoráře
    int index = 0;
    int prev_seg = -1;
    for ( ; ; ) {
                                    // nastaveni FAT tabulky
```

```
            if (prev_seg != -1) FAT[prev_seg] == seg;
            WriteSegment( seg, &data[index] );      // zápis segmentu
            FAT[seg] = 0x01;                        // poslední segment
            if (size < 254) break;                  // konec
            size -= 254;                            // odečteni velikosti
            index += 254;
            prev_seg = seg;
            seg = SearchFree( );                    // hledá volný segment
        }
        WriteSegment( 1, FAT );                     // zápis FAT tabulky
        memcpy( record, name, 10 );                 // jméno souboru
        WriteName( record );              // zápiše nové jméno souboru
}
```

1.6 Formátování paměti panelu.

1. *Zjištění velikosti jedné paměti.*

Příkazem „Nastavení velikosti EEPROM“ se nastaví nulová velikost EEPROM
Do prvního segmentu se zapíše BOOT záznam se základními hodnotami.
Nalézt tento BOOT záznam postupným prohledáním segmentů.
Číslo shodného segmentu je počet segmentů jedné paměti.

2. *Zjištění počtu osazených pamětí.*

Příkazem „Nastavení velikosti EEPROM“ nastavit skutečnou velikost EEPROM
Zapsat BOOT záznam do sektoru za první paměť
Načíst a zkontrolovat BOOT záznam v sektoru, pokud souhlasí, je paměť osazena

3. Zapsat BOOT záznam s nastavením velikosti paměti:

256 - bloků paměti.
1 - sektor v bloku paměti.
1 - druhy blok pro FAT tabulku.
3 - třetí blok pro DIR.
128 - sektorů jedné paměti.

4. *Nulovat FAT tabulku*

bajt: 1		0x01
2		0x01
3		0x01
4	- 255	0x00

5. Nulování DIR

Do celého sektoru nahrát 0x00. Pro zrychlení stačí nulovat první bajt u každého jména.

Všechny funkce zápisu, čtení i formátování jsou obsaženy v knihovně PDriver.dll, jejich popis najdete.

```
int  Format( )
{
  BYTE data[256];
         // default záznam BOOT sektoru
  BYTE BOOT[256];
  memset( BOOT, 0, 256 );
  BOOT[P_BOOT_SEG] = 0;
  BOOT[P_BOOT_COUNTSEG] = 1;
  BOOT[P_BOOT_FAT] = 1;
  BOOT[P_BOOT_DIR] = 2;
  BOOT[P_BOOT_RAM] = 0;
  BOOT[P_BOOT_COUNT_EE] = 0;
         // zápisem BOOT sektoru se otestuje paměť EEPROM
  WriteSegment( 0, BOOT );                  // zapíše BOOT sektor
  LoadSegment( 0, data );                   // přečte BOOT sektor
  if (memcmp( BOOT, data ) != 0) return;    //
         // nastaveni nulové velikosti EEPROM
  pDirectSendPanel( CMD_PANEL_SEND, eeprom, MSG_SET_EEPROM, 0x50000 );
         // adresní prostor paměti se opakuje, pokud tedy budeme
         // postupně číst jednotlivé segmenty, narazíme na BOOT záznam
         // tím se zjistí velikost jednoho čipu EEPROM
  for (int i=2; i<256; i*=2) {
       LoadSegment( 0, data );              // přečte BOOT sektor
       if (memcmp( BOOT, data ) != 0) break; //
  }
         // nastavení skutečné velikosti EEPROM
  memcpy( data, eeprom, 5 );
  data[4] = i;
  pDirectSendPanel( CMD_PANEL_SEND, eeprom, MSG_SET_EEPROM, 0x50000 );
         // nyní je možné otestovat zda je osazena druhá paměť
  int size = i;
  WriteSegment( 0, BOOT );                  // zapíše BOOT sektor
  LoadSegment( 0, data );                   // přečte BOOT sektor
  if (memcmp( BOOT, data ) == 0) i*=2;      // pokud je osazena velikost
                                            // je dvojnásobná
         // nyní je možné nastavit skutečnou velikost paměti
  memcpy( data, eeprom, 5 );
  data[0] = i;                              // celková velikost
  data[4] = size;                           // velikost jedné paměti
  pDirectSendPanel( CMD_PANEL_SEND, eeprom, MSG_SET_EEPROM, 0x50000 );
         // Zápis konečného stavu BOOT sektoru
     BOOT[0]                = 1;
     BOOT[1]                = 0;
     BOOT[2]                = 1;
     BOOT[3]                = 1;
     BOOT[P_BOOT_SEG]       = i;          // celková velikost
     BOOT[P_BOOT_COUNTSEG]  = 1;
     BOOT[P_BOOT_FAT]       = 1;
     BOOT[P_BOOT_DIR]       = 2;
     BOOT[P_BOOT_RAM]       = 0;
     BOOT[P_BOOT_COUNT_EE]  = size;       // velikost jeden EEPROM
```

```
    WriteSegment( 0, BOOT );            // zápis BOOT
    memset( data, 0, 256 );
    WriteSegment( 2, data );            // nulovaní direktoráře
    data[0] = 1;
    data[1] = 1;
    data[2] = 1;
    WriteSegment( 1, data );            // zápis FAT tabulky
        // v BOOT sektrou je i nastavení jasu
        // po naformátováni je možno nastavit do základu
}
```

2. FORMÁT FONTU

Fonty jsou uloženy ve FLASH paměti, za prostorem pro firmware. Adresu tabulky fontů a dat fontů získáte z identifikátoru firmware. Bližší informace získáte z „**Popisu komunikačního protokolu – Načtení identifikátoru firmware**“ . Konec prostrou pro fonty záleží na typu firmware:

PANEL-CY - 0xFA00
PANLE1-CY - 0xF800

Tento prostor není vhodné přesáhnou, jinak může dojít k zablokování panelu. V tomto prostoru jsou uloženy některé důležité informace pro správný běh programu.

V prostoru pro fonty je nejprve uložena tabulka fontů, za ní následuje prostor pro data fontů. Tabulka fontů obsahuje 16 záznamů, jeden záznam obsahuje informace o jednom fontu. Poslední záznam je vyhrazen pro inicializační soubor. Jeden znak fontu může být široký 8 bodů, od verze firmware 9,70 je možné použít fonty i širší. Počet fontů, které je možné použít je omezen velikostí volné paměti. Na umístění dat jednoho fontu v paměti nezáleží, musí se ovšem uložit v paměti souvisle bez rozdělení.

V identifikátoru firmware je adresa tabulky fontů a dat fontů. U starších verzí jak 9.00 byla tabulka fontů umístěna od adresy 0x0200, u novějších verzí je umístěna na začátku prostoru pro fonty.

Adresy verze starší jak 9.00
Tabulka fontů = Adresa tabulky fontu (identifikátor firmware)
Data Font; = Adresa dat pro fonty (identifikátor firmware)

Adresy verze od verze 9.00
Tabulka fontů = Adresa tabulky fontu (identifikátor firmware)
Data Font; = Adresa dat pro fonty + 0x200 (identifikátor firmware)

Funkce pro zápis a čtení fontů jsou obsaženy v knihovně „**PDriver.dll**“ Program **Panel II** obsahuje kompletní Editor Fontů. Další informace o fontech pak naleznete v dokumentaci „***Popis funkcí knihovny PDriver.dll***“ a „***Editor Fontů***“

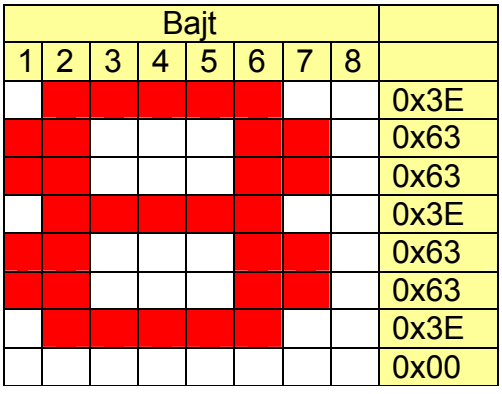
Relativní adresa záznamů v tabulce fontů

Č fontu	Adresa	Č fontu	Adresa
1	0x0000	9	0x0100
2	0x0020	10	0x0120
3	0x0040	11	0x0140
4	0x0060	12	0x0160
5	0x0080	13	0x0180
6	0x00A0	14	0x01A0
7	0x00C0	15	0x01C0
8	0x00E0	16	0x01E0

Jeden záznam v tabulce fontů

Adresa	Popis
0	Velikost záznamu.
1-20	Jméno fontu – je-li jméno kratší jak 20 znaku řetězec zakončený nulou
21	Verze fontu dekadicky cele číslo 5 (v 5.35)
22	desetiny 35 (v 5.35).
23	Adresa dat fontu - Horních 8 bitů
24	Dolních 8 bitů
25	Adresa tabulky zúžení fontu – Horních 8 bitů
26	Dolních 8 bitů
27	Výška znaku v bodech
28	Šířka znaku v celých bajtech.
29	Šířka znaku v bodech
30-31	Volné

Data fontu jsou uložena po jednotlivých znacích za sebou Jeden znak je uložen podle následujícího schématu. Velikost datového pole pro font 8x8 bodů je 8x256 = 2048bajtů. Font 16x8 bodů má dvojnásobnou velikost.



Data jsou uložena v následujícím pořadí

(Znak „7“) 0x3E**, 0x63, 0x63, 0x3E, 0x63, 0x63, 0x3E, 0x00** (Znak „9“)

Bajt 1								Bajt 2								
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	
		■	■	■	■	■	■	■	■	■	■					0xFC, 0x0F
		■	■	■	■	■	■	■	■	■	■					0xFC, 0x0F
■	■	■	■							■	■	■	■			0x0F, 0x3C
■	■	■	■							■	■	■	■			0x0F, 0x3C
■	■	■	■							■	■	■	■			0x0F, 0x3C
■	■	■	■							■	■	■	■			0x0F, 0x3C
		■	■	■	■	■	■	■	■	■	■					0xFC, 0x0F
		■	■	■	■	■	■	■	■	■	■					0xFC, 0x0F
■	■	■	■							■	■	■	■			0x0F, 0x3C
■	■	■	■							■	■	■	■			0x0F, 0x3C
■	■	■	■							■	■	■	■			0x0F, 0x3C
■	■	■	■							■	■	■	■			0x0F, 0x3C
		■	■	■	■	■	■	■	■	■	■					0xFC, 0x0F
		■	■	■	■	■	■	■	■	■	■					0xFC, 0x0F
																0x00, 0x00
																0x00, 0x00

Data jsou uložena v následujícím pořadí:

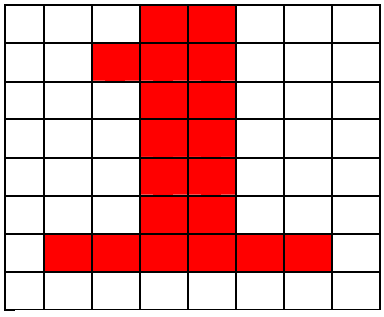
(Znak „7“)

0xFC, 0x0F, 0xFC, 0x0F, 0x0F, 0x3C, 0x0F, 0x3C, 0x0F, 0x3C, 0x0F, 0x3C, 0xFC, 0x0F, 0xFC, 0x0F, 0xFC, 0x0F, 0xFC, 0x0F, 0xFC, 0x0F, 0xFC, 0x0F, 0xFC, 0x0F, 0xFC, 0x0F, 0x00, 0x00, 0x00, 0x00

(Znak „9“)

2.1 Proporcionální písmo

Aby bylo možno zobrazit písmo proporcionálně je k znakům vytvořena tabulka o velikosti 256 bajtu (jeden znak = jeden bajt). Jeden bajt udává, kolik prázdných sloupců je před znakem. Pořadí je stejné jako v datovém poli znaku



V případě znaku jedna je číslo posunutí znaku **„0x01“**

2.2 Inicializační soubor

Poslední záznam v tabulce je vyhrazen pro data ini souboru. Data tohoto souboru jsou většinou uložena na konci paměti, může být samozřejmě uložen kdekoliv v prostrou pro fonty. Platí stejné jako u fontů, konec použitelné paměti je 0xFA00 nebo 0xF800 podle typu firmware.

Záznam ini souboru

Adresa	Popis
0	Velikost záznamu.
1-20	Jméno ini souboru – je-li jméno kratší jak 20 znaku řetězec zakončený nulou
21	
22	
23	Adresa dat ini souboru - Horních 8 bitů
24	Dolních 8 bitů
25	Velikost ini souboru – Horních 8 bitů
26	Dolních 8 bitů
27	
28	
29	
30-31	

3. FORMÁT HODIN

Panel obsahuje hodiny reálného času RTC, tyto hodiny jsou zálohované a odměřují čas i při vypnutém panelu. Čas těchto hodin je možné zobrazovat nebo podle něj spouštět programy.

Formát RTC

Adres	Hodnota	Popis
1	0x00 – 0x59	Vteřiny
2	0x00 – 0x59	Minuty
3	0x00 – 0x23	Hodiny
4	0x01 – 0x08	Den v týdnu
5	0x01 – 0x31	Den
6	0x01 – 0x12	Měsíc
7	0x00 – 0x99	Rok dvoumístně

4. ZOBRAZOVACÍ PROGRAM

Program se skládá z textu a příkazů, příkazy mají kód od 0 do 15. Program je prováděn od prvního bajtu do bajtu konce 0xFF, pak je prováděn opět od začátku. Zobrazení je tvořeno dvěma hlavními příkazy zobrazení textu - posunem a staticky. Ostatní příkazy jsou řídící.

Kódy příkazů

Kód	Funkce	Parametry
0x00	REZERVA	
0x01	REZERVA	
0x02	Posouvání textu	1b - horní 4 bity funkce vlevo, dolů atd). spodní 4 byty volba řádky. 2b - rychlost v milisekundách.
0x03	Pauza	Dva bajty číslo po 0,1s
0x04	Statický text	1b - horní 4 bity funkce. spodní 4 byty volba řádky. 2b - čas zobrazení.
0x05	Smazání všech řádků	
0x06	Smazání řádku	
0x07	Zobrazení datumu a času, teploty analogového vstupu a f vstupu	1b – funkce zobrazení
0x08	REZERVA	
0x09	REZERVA	
0x0A	Volba řádky	1b – číslo řádky
0x0B	Volba Barvy	3b - 1 – volné 2 - červená 3 - zelená 4 - modrá
0x0C	Nastavení znakového fontu	1b - bity 0-3 = vypnutí/zapnutí redukce mezer 0/1 bity 4-7 = číslo znakového fontu.
0x0D	Intenzita svitu	1b - 0-100 v hex formátu
0x0E	Provede jiný program	1b - kolikrát se má program provést 10b jméno programu, který se má provést
0x0F	Spustí jiný program	10b Jméno programu, který se má spustit
0xFF	Konec programu	Žádný parametr

Pořadí příkazů může být jakékoliv, ovšem musí se brát v úvahu, aby daný příkaz plnil svou funkci, jsou příkazy řazeny v určitém pořadí. Používají se pouze příkazy, které jsou třeba pro daný případ zobrazení, ostatní se můžou vynechat.

Před zobrazením mohou být použít tyto příkazy:

Volba řádky	změnit řádku, na které chcete text zobrazit
Volba fontu	nastavit font pro zobrazení – může se vložit i do textu
Nastavení jasu	nastavit intenzitu svitu zobrazení
Nastavení barvy	nastavení barvy

Zobrazovací funkce

Posun	zobrazení textu posouváním
Staticky	statické zobrazení textu

A za příkazy zobrazení:

Pauza	doba, po kterou má být text zobrazen
Smazání celého panelu	smaže celý panel
Smazání řádku	smaže zvolený řádek

Příkazy zobrazeni hodin, teploty a vstupů se vkládají do textu podle toho, jak požadujete hodnoty zobrazit. Do textu je možné též vložit příkaz volby barvy, tím je možné vytvořit barevný text. Také je možné do textu vložit příkaz volby fontu tím zobrazíme jeden text různým písmem.

Příkaz „provedení programu“ je možné vložit do jakékoliv části programu Příkaz „spuštění programu“ se vkládá na konec, pokud je vložen dříve nebudou všechny následující příkazy provedeny. Pomocí tohoto příkazu je možné program rozdělit na několik menších programů, „ladění“ jednoho programu je pak jednodušší.

4.1 Příkaz posouvání znaku - 0x02.

Příkaz má dva parametry. První parametr udává funkci (viz. Tabulka funkcí) a řádek, druhý rychlost posunu. Spodní čtyři bity prvního parametru udávají, na kterém řádku se má text posouvat, význam nemá pouze u funkce „vše nahoru a dolů“. Pokud jsou tyto bity nulové, použije se řádka nastavená příkazem volba řádky. Rychlost je čas posunu o jeden bod.

Tabulka funkcí prvního parametru.

		FUNKCE				ZAROVNÁNÍ		
COD	**1b**	**vlevo**	**dolů**	**nahoru**	**vše**	**vlevo**	**střed**	**vpravo**
0x02	0x00	■						
	0x10		■			■		
	0x20			■		■		
	0x30		■				■	
	0x40			■			■	
	0x50		■					■
	0x60			■				■
	0x70		■		■			
	0x80			■	■			

Tabulka rychlosti ve standardním programu

Rychlost	vlevo	Nahoru, dolů
1	80	80
2	30	40
3	15	30
4	10	20
5	7	15
6	4	10
7	3	5
8	2	3
9	1	1

Příklad zobrazení „Text“ posouváním vlevo maximální rychlostí
< **0x02 0x00 0x01** Text >

4.2 Příkaz pauza - 0x03.

Příkaz má jeden dvoubajtový parametr. Provádění programu se na tomto příkazu zastaví na dobu danou číslem v parametrech. Číslo je uloženo podle následujícího příkladu.

Pauza délky 60 sekund = 600 dekadicky a 0x0258 hex.

Příkaz: < **0x03 0x02 0x58** >

4.3 Příkaz zobrazení statického textu - 0x04.

Příkaz má dva parametry. První udává funkci (viz tabulka), po jakou dobu má být text zobrazen. Spodní čtyři bity prvního parametru udávají na kterém řádku se má text zobrazit, Pokud jsou tyto bity nulové, použije se řádka nastavená příkazem volba řádky. Význam nemá pouze u funkce „vše static“.

Tabulka funkcí prvního parametru.

		FUNKCE			ZAROVNÁNÍ		
COD	**1b**	**static**	**text**	**vše**	**vlevo**	**střed**	**vpravo**
0x04	0x00	■			■		
	0x10	■				■	
	0x20	■					■
	0x30		■		■		
	0x40		■			■	
	0x50		■				■
	0x60	■		■			

Funkce **ext** provede zobrazení textu do paměti, příkazem **vše static**, **vše nahoru** a **vše dolu**, se zobrazí na všech řádcích současně. U funkce text nemá druhý parametr význam a dává se nulový.

Příklad zobrazení „Text“ staticky uprostřed po dobu 2 vteřin.

< **0x04 0x10 0x14 Text** >

4.4 Příkaz smazání všech řádek - 0x05.

Příkaz nemá žádný parametr, provede smazání všech řádků, na kterých bylo zobrazeno programem. To pro případ, že například u dvouřádkového panelu na druhý řádek zobrazuje druhý program tak, aby první program nesmazal druhou řádku, na kterou nezobrazuje. Tento příkaz je funkční pouze u panelu s verzí 8.60 a nižší. U panelů s nižší verzí je tento příkaz přeskočen.

4.5 Příkaz smazání jedné řádky - 0x06.

Příkaz smaže aktuelně zvolenou řádku (Příkazem volba řádky} nemá žádný parametr a je funkční od verze 8.80.

4.6 Příkaz zobrazení hodin, teploty a ostatních vstupů - 0x07.

Příkaz má jeden parametr udávající co se má zobrazit. Tento příkaz je možno vložit do textu statického zobrazení.

Tabulka kódů zobrazení

Čas a datum		Teplota		Analog	
COD	zobrazení	COD Cele / des	Čidlo	COD	Vstup
0	vteřiny	16 / 32	1	48	1
1	minuty	17 / 33	2	49	2
2	hodiny	18 / 34	3	50	3
3	týden	19 / 35	4	51	4
4	den	20 / 36	5	52	5
5	měsíc	21 / 37	6	53	6
6	rok	22	7	54	7
		23	8		
		24	9	64	Frekvence
		25 / 41	CPU	65	počítadlo

Celé - teplota zobrazena bez desetiny.
Des – teplota zobrazena s jedním desetinným místem.
MCU – teplota procesoru panelu.

U funkce zobrazení analogového vstupu jsou za funkcí dva dvoubajtové parametry, udávající spodní a horní konstantu zobrazení.
Spodní konstanta – hodnota, která bude zobrazena při nule na vstupu.
Horní konstanta – hodnota, která bude zobrazena při maximálním napětí na vstupu (podle vstupního děliče) na CPU 1.2V.

Přiklad zobrazení analog 0-1500 0x0000 – 0x05DC < **0x07 0x48 0x00 0x00 0x05 0xDC** >

Aby se údaj zobrazil, musí před tímto příkazem být příkaz zobrazení posunem 0x02 nebo staticky 0x04.

4.7 Příkaz volba řádky - 0x0A.

Příkaz má jeden jednobajtový parametr obsahující **číslo řádky a segmentu**. Řádek u panelu od verze 8.41 je možno rozdělit na segmenty a do těch samostatně zobrazovat. Rozdělení řádku je dané při výrobě. Je to vhodné například pokud chcete, aby panel zobrazoval na 16 znacích reklamní text, vlevo by byla zobrazena teplota a vpravo čas. V panelu je pro takovýto případ možno spustit dva programy.

Parametr:
Bit 0-3 číslo řádky 1 – 15 (0 = řádka nebude měněna).
Bit 4-7 číslo segmentu 1 – 5 (0 = segment nebude měněn)
Pokud je zvoleno větší číslo řádky nebo segmentu, než má panel, je zvoleno nejvyšší číslo řádky nebo segmentu

Příklad přepnutí na druhou řádku < **0x0A 0x02** >

4.8 Příkaz změna barvy - 0x0B.

Příkaz má čtyři jednobajtové parametry. První bajt je volný, zbylé tři obsahují nastavení barev RGB. Příkaz je možné vkládat do textu a tím vytvářet barevný text. Příkaz je funkční pouze u panelů, které mají více barev. Použití tohoto příkazu u panelu s verzí nižší jak **8.80** může způsobit **chybné zobrazení**.

Příklad: změna na zelenou barvu < **0x0B 0x00 0x00 0xFF 0x00** >

Bajt	Hodnota	Popis
1	**0B**	Kód přikazu změny barvy
2	00	Volné
3	00 – FF	Červená 0 – ne 255 použít tuto barvu
4	00 – FF	Zelená 0 – ne 255 použít tuto barvu
5	00 – FF	Modrá 0 – ne 255 použít tuto barvu

4.9 Příkaz nastavení znakového fontu - 0x0C.

Příkaz má jeden jednobajtový parametr. Od verze 8.40 je možno tento příkaz vložit do statického zobrazení, tím je možno staticky text zobrazit různým písmem.

Parametr:
Bit **0-3 redukce mezer** 0 = vypnuto
1 = zapnuto – proporcionální písmo.
15 = nastavení redukce mezer nebude měněno.

Bit **4-7 číslo fontu** 1 – 14 = číslo fontu
15 = nastavený font nebude měněn.

Příklad volba druhého fontu s redukcí mezek (proporcionální písmo) < **0x0C 0x21** >

4.10 Příkaz změny intenzity svitu - 0x0D.

Příkaz má jeden jednobajtový parametr, který obsahuje intenzitu svitu od 0 do 100. Příkaz změní intenzitu svitu celého panelu.

Příklad nastavení jasu na 80% < **0x0D 0x50** >

4.11 Příkaz provedení programu - 0x0E.

Příkaz má dva parametry, první jednobajtový udává kolikrát se má daný program provést, druhý dlouhý 10 bajtů obsahuje jméno spouštěného programu. Pokud je program s tímto příkazem spuštěn jako druhý program, je přeskočen.

Příkaz zastaví provádění programu a spustí jiný program, tento program provede minimálně jednou nebo vícekrát podle prvního parametru. Po ukončení pokračuje provádění původního programu. Pokud program v paměti panelu není, je příkaz přeskočen.

Parametry:
1 - počet provedení programu -- 1 – 200 počet provedení
254 zkopíruje následující příkaz do druhého programu a spustí jej
255 spustí program jako druhý
2 - jméno programu, je-li kratší než 10 znaků, musí být zakončené nulou

Spuštění příkazu v druhém programu je pro případ, pokud chcete na dvou řádcích posouvat text vlevo.

V případě spuštění příkazu jako druhý musí být za příkazem provedení programu příkaz zobrazení (staticky nebo posun) před zobrazením může být příkaz nastavení fontu. Z obou těchto příkazů včetně následného textu je vytvořen nový program, který je spuštěn jako druhý. Hlavní program provede další funkci zobrazení a počká na dokončení zobrazení druhého programu.

Příklad 0x0E 0xFE jméno nulové - příkaz přesunutí příkazu
0x02 0x00 0x01 Text řádky 1 - posouvej text na řádce 1 (druhý program)
0x02 0x01 0x01 Text řádky 2 - posouvej text na řádce 2
0xFF - Konec programu

Příklad provedení 5x programu „test“ < **0x0F 0x05 test 0x00 0x00 0x00 0x00 0x00 0x00** >

4.12 Příkaz spuštění programu - 0x0F.

Příkaz má jeden parametr o délce 10 bajtů, obsahující jméno spouštěného programu. Příkaz spustí jiný program, spuštěný program je cyklicky prováděn dokud není přerušen. Příkaz se

používá na konci programu, pokud by byl použit uprostřed, všechny následující příkazy by nebyly provedeny. Pokud je jméno kratší než 10 znaků, je zakončeno nulou.

Parametr:
2 - jméno programu, jeli kratší než 10 znaků, musí být zakončené nulou.

Příklad spuštění programu „test“ < **0x0F test 0x00 0x00 0x00 0x00 0x00 0x00** >

4.13 Konec programu - 0xFF.

Tento kód označuje konec programu. **Musí být vždy uveden**.

4.14 Příklady

1, Posouvání textu vlevo s pauzou a jasem na 70%

0x0D 0x46	*Nastavení jasu 70%*
0x0C 0x11	*Redukce mezer a font 1*
0x02 0x00 0x01	*Posun textu vlevo*
Text 1	*Zobrazovaný text*
0x03 0x00 0x1E	*Pauza 3 vteřiny*
0x0C 0x21	*Redukce meze a font 2*
0x02 0x00 0x01	*Posun textu vlevo*
Text 2	
0xFF	*Konec programu*

Tento program lze uložit do panelu příkazem „přímý zápis do paměti“ od adresy 10 a spustit příkazem „spuštění programu“ (0xFF 0x00 0xFD 0x00). Na displeji se bude zprava posouvat Text 1 a Text 2, každý text bude zobrazen jiným fontem (pokud panel obsahuje dva fonty). Před zobrazením Text 2 bude 3 vteřiny pauza. Text bude zobrazován jasem 70%.

2, Statické zobrazení textu u víceřádkového panelu.

Pokud je třeba u víceřádkového panelu zobrazit staticky text na všech řádcích najednou, použije se příkaz „statické zobrazení“ do paměti (text). Zobrazení se pak provede příkazem „**vše**“.

0x04 0x40 0x01	**řadka 1**	*Text na řádce 1*
0x04 0x40 0x02	**řadka 2**	*Text na řádce 2*
0x04 0x40 0x03	**řadka 3**	*Text na řádce 3*
0x04 0x40 0x04	**řadka 4**	*Text na řádce 4*
0x04 0x60 0x1E		Zobrazení textu (vše)
0xFF		*Konec programu*

Podobné funkce lze dosáhnou i použitím čtyř příkazů statického zobrazení (**0x04 0x00 0x00**), ovšem text by byl v krátkých intervalech zobrazován od první řádky. Je to způsobeno tím, že zobrazení textu nějaký čas trvá. Příkazem „text“ se text připraví a příkazem „vše“ se jenom přesune na zobrazení.

U příkazu zobrazení textu lze použít i „**vše nahoru**“ pak text najede na jednotlivých řádcích zespodu, nebo „**vše dolů**“ kdy text najede na jednotlivých řádcích z hora.

3, Statické zobrazení času.

0x04 0x10 0x32 0x07 0x02 : 0x07 0x01 *čas s dvojtečkou*
0x04 0x10 0x32 0x07 0x02 0x07 0x01 *čas s mezetrou*
0xFF *Konec programu*

Tento program zobrazí uprostřed řádky čas ve formátu **< HH : MM >,** tečka uprostřed bude blikat v intervalu půl vteřiny.

9. ŘÍDÍCÍ PROGRAM

Tento program nelze spustit po nahrání přímo do paměti panelu. Po spuštění by jej panel začal zpracovávat jako zobrazovací program. Do paměti EEPROM panelu se program nahrává s atributem „**řídící program**“ Jeden údaj spuštění programu má délku 16 bajtů. Obsahuje čas zapnutí a vypnutí, případně hodnotu paralelního vstupu a jméno programu, který má být spuštěn.

Formát záznamu časového spouštění.

Bajt	Hodnota	Popis
0	0x00 – 0x23	Hodiny zapnutí programu
1	0x00 – 0x59	Minuty zapnutí programu
2	0x01 – 0x31	Den zapnutí programu den nebo týden
3	0x00 – 0x23	Hodiny vypnutí programu
4	0x00 – 0x59	Minuty vypnutí programu den nebo týden
5	0x01 – 0x31	Den vypnutí programu
6	text	10 znaků dlouhé jméno, je-li kratší zakončené nulou

Formát bajtu 2 a 3 při nastavení spouštění podle dnů v týdnu

bit	Hodnota	Popis
0	0/1 ne/ano	Pondělí
1	0/1 ne/ano	Úterý
2	0/1 ne/ano	Středa
3	0/1 ne/ano	Čtvrtek
4	0/1 ne/ano	Pátek
5	0/1 ne/ano	Sobota
6	0/1 ne/ano	Neděle
7	1	Identifikační bit dnů v týdnu

Formát záznamu spouštění paralelními vstupy.

Bajt	Hodnota	Popis
0	0xFD	Identifikační kód paralelního spouštění
1	0x00 – 0xFF	Hodnota na vstupu
2	0x01 – 0x31	Funkce 0 – spuštěn po dobu hodnoty na vstupu 1 – spuštěn do spuštění jiného programu 2 – počet spuštění uveden v bajtu 4
3	0x00, 0xFD	Funkce 2 = 0xFD jinak 0x00
4	0x00 – 0x59	Funkce 2 = počet provedení, jinak 0x00
5	0x01 – 0x31	
6	text	10 znaků dlouhé jméno, je-li kratší, zakončené nulou

Jednotlivé 16 bajtové záznamy jsou za sebou, na pořadí nezáleží a maximálně jich může být 256. Tento program lze spustit pouze jako hlavní program. Panel cyklicky kontroluje časy nebo hodnoty vstupů a podle toho spouští hlavní programy.

Výrobce: **RTG Mělník** tel/fax. 315624739 mob. 603261914
www.rtg-tengler.cz email: rtg@rtg-tengler.cz
www.svetelnepanely.cz

Software: **Vacek Luboš** tel. 606485842 email: vacek@svetelnepanely.cz