

Popis ovládání regulátorů řady Novar prostřednictvím dálkové komunikační linky

Příručka pro programátory

Stav 11/2005

Regulátory jalového výkonu Novar-106/114/206/214/314RS mohou být vybaveny dálkovou komunikační linkou s rozhraním RS-232 nebo RS-485. Pomocí této linky mohou být monitorovány a ovládány z nadřazeného řídicího systému (obvykle PC).

Tato příručka popisuje způsob komunikace s regulátorem. Předpokládá základní znalost parametrů regulátoru a jazyka C.

1.1 Popis datových struktur

Data, která lze přenášet mezi regulátorem a nadřazeným systémem, jsou uspořádána do následujících struktur :

- „Status“ ... obsahuje informace o stavu regulátoru (typ, výrobní číslo, stav výstupů, alarmu, chybové příznaky atd.); lze pouze číst
- „EEStatus“ ... obsahuje další stavové informace (zálohované v paměti EEPROM), jako zaznamenané maximální hodnoty veličin, počty a dobu sepnutí výstupních relé atd.; lze pouze číst
- „NovarStatus“ ... obsahuje souhrn základních informací o stavu regulátoru a okamžité hodnoty měřených veličin; určena zejména pro on-line vizualizaci; lze pouze číst
- „Config“ ... obsahuje stav nastavitelných parametrů regulátoru; lze číst i zapisovat
- „NovarSetMap“ ... zápisem do této virtuální struktury lze aktivovat vybrané funkce regulátoru a tím ovlivňovat jeho činnost

Informace o stavu regulátoru získá nadřazený systém přečtením odpovídající struktury z regulátoru, naopak zápisem do odpovídající struktury může nadřazený systém například změnit některý z parametrů, spustit vybranou operaci atd.

Obsah jednotlivých struktur je uveden v samostatné kapitole dále.

1.2 Komunikační protokoly

Přenos informací mezi regulátorem a nadřazeným systémem se provádí přes sériovou asynchronní komunikační linku (COM) s rozhraním RS-232 nebo RS-485. V případě použití rozhraní RS-485 je možné na jednu komunikační linku připojit více přístrojů a na straně nadřazeného systému je možné použít převodník rozhraní RS-232/RS-485, vybavený automatickým přepínáním směru přenosu dat, případně musí přepínání směru přenosu dat zajistit program nadřazeného systému.

Regulátory jsou standardně dodávány s přednastaveným firemním protokolem „KMB“. U regulátorů, vybavených firmware verze 2.0 a vyšším (resp. 1.1 a vyšším v případě regulátoru Novar-314RS) lze volitelně nastavit i protokol Modbus-RTU.

Rychlost komunikace je nastavitelná v rozsahu 300-9600 Bd (standardně 9600 Bd).

1.2.1 Komunikační protokol KMB

Komunikační kanál je nastaven na 8 bitů, bez parity, jeden stop-bit.

Komunikace typu „master-slave“. Na základě přijetí řádné zprávy-příkazu přístroj odešle odpovídající zprávu-odpověď.

Zprávy mají jednotný formát :

1. Adresa přístroje
2. Délka zprávy (v bytech) bez závěrečné checksum. Má vždy hodnotu (3+délka těla zprávy).
3. Typ zprávy (1 byte).
4. Tělo zprávy - má různou délku dle typu zprávy.
5. Závěrečná checksum - součet předchozích byte modulo 256 (1 byte).

Pokud přístroj přijme řádnou zprávu, a podaří se mu ji řádně zpracovat (vykonat příkaz), odešle příslušnou odpověď a na místě typu zprávy bude hodnota 0. Pokud je typ zprávy v odpovědi nenulový, příkaz se z nějakých důvodů nepodařilo vykonat.

Některé zprávy nemají tělo.

Přístroj odešle odpověď do 600ms po obdržení zprávy od mastera. Během příjmu příkazu nebo vysílání odpovědi může nastat mezibytová mezera délky odpovídající době přenosu maximálně 4 znaků (bytů).

1.2.1.1 Popis zpráv

Pro zápis/čtení datových struktur lze použít následující typy zpráv :

č. zprávy (hex)	typ zprávy
0x14	Přečtení stavu přístroje - Status, EEStatus
0x16	Přečtení nastavení přístroje – Config
0x17	Zápis nastavení do přístroje - Config
0x30	Přečtení stavu přístroje – NovarStatus
0x31	Spuštění vybraných funkcí přístroje – zápis do NovarSetMap

1.2.1.1.1 Přečtení stavu přístroje (Status, EEStatus) – 0x14

Zpráva č. 0x14. V odpovědi předá přístroj bezprostředně za sebou tzv. Status (34 bytu) a EEStatus (42 bytu, od verze 2.0 70 bytů), celkem tedy 76, resp. 104 bytů.

Příklad :

Master musí odeslat následující sekvenci (předpokládáme, že adresa přístroje je 1) :

| 0x01| 0x03 | 0x14 | checksum = 0x18 |

Příkaz nemá tělo.

Struktura odpovědi :

| 0x01| 0x6B| 0x00 | tělo zprávy = Status+EEStatus (76, resp. 104 bytů) | checksum |

(druhý byte, tedy délka zprávy bez závěrečné checksum, bude mít hodnotu $76 + 3 = 79 = 0x4F$, resp. pro verzi 2.0 a vyšší $104 + 3 = 107 = 0x6B$).

1.2.1.1.2 Přečtení nastavení přístroje (Config) – 0x16

Zpráva č. 0x16. Přístroj předá tzv. Config (66 bytů).

Příkaz :

| 0x01| 0x03 | 0x16 | checksum = 0x1A |

Odpověď :

| 0x01| 0x45| 0x00 | tělo zprávy = Config (66 bytů) | checksum |

1.2.1.1.3 Zápis nastavení do přístroje (Config) – 0x17

Zpráva č. 0x17. Zápis tzv. Config (66 bytů) do přístroje.

Příkaz :

| 0x01| 0x45 | 0x17 | tělo zprávy = Config (66 bytů) | checksum |

Odpověď :

| 0x01| 0x03| 0x00 | checksum = 0x04 |

Poznámka : Proměnné DeviceAddr a RemoteBDRate nelze přes komunikační linku měnit, zapisované hodnoty těchto proměnných mohou být libovolné.

1.2.1.1.4 Přečtení stavu přístroje (NovarStatus) – 0x30

Zpráva č. 0x30. Přístroj předá informace o stavu přístroje potřebné zejména pro on-line vizualizaci, tzv. NovarStatus (35 bytů).

Příkaz :

| 0x01| 0x03 | 0x30 | checksum = 0x34 |

Odpověď :

| 0x01| 0x26| 0x00 | tělo zprávy =NovarStatus (35 bytů) | checksum |

1.2.1.1.5 Spuštění vybraných funkcí přístroje (přes NovarSetMap) – 0x31

Zpráva č. 0x31. Pomocí tohoto příkazu lze dálkově vyvolat některou z vybraných funkcí regulátoru . Jednotlivé funkce jsou aktivovány hodnotou 1 ve struktuře NovarSetMap po jejím zápisu do přístroje.

Příkaz :

| 0x01| 0x0B | 0x31 | tělo zprávy = NovarSetMap (6, resp.8 bytů) | checksum |

(druhý byte, tedy délka zprávy bez závěrečné checksum, bude mít hodnotu $6 + 3 = 9 = 0x09$, resp. pro verzi 2.0 a vyšší $8 + 3 = 11 = 0x0B$).

Odpověď :

| 0x01| 0x03| 0x00 | checksum = 0x04 |

1.2.2 Komunikační protokol Modbus-RTU

Přístroje s firmware 2.0 a vyšší (resp. 1.1 a vyšší pro Novar-314RS) lze nastavit na komunikační protokol Modbus-RTU. Vedle adresy a rychlosti komunikace lze nastavit i funkci paritního bitu (sudá / lichá / žádná parita).

Přístroj odešle odpověď nejpozději do 600ms po obdržení zprávy od mastera. Během příjmu příkazu nebo vysílání odpovědi může nastat mezibytová mezera délky odpovídající době přenosu maximálně 1,5 znaku (bytů).

Režim "broadcast " není podporován.

Jsou implementovány následující funkce :

kód funkce	název funkce	aplikace
03	Read Holding Registers	čtení Config – registry 40101-40133 (adresovány 100 – 132)
04	Read Input Registers	čtení Status+EEStatus – registry 30101-30152 (adr. 100 – 151) čtení NovarStatus – registry 30201-30218 (adr. 200 – 217)
06	Preset Single Register	zápis Config – registry 40101-40133 (adr. 100 – 132) zápis NovarSetMap – registry 40201-40203 (adr. 200 – 202)
08	Diagnostics – 00 – Return Query Data 01 – Restart Comm Option 02 – Return Diagnostic Register 04 – Force Listen Only Mode 10 – Clear Ctrs & Diag. Register 11 – Return Bus Message Count	základní diagnostické funkce
16	Preset Multiple Registers	obdobně jako 06 - Preset Single Register
17	Report Slave ID	identifikace přístroje

Přístup ke strukturám je realizován pomocí čtení/zápisu z/do odpovídajících registrů dle tabulky. Každá struktura přitom odpovídá souvislé skupině registrů v uvedeném rozsahu.

Příklad :

Načtení stavu regulátoru (NovarStatus) pomocí funkce Read Input Registers, předpokládáme adresu přístroje 1 :

Příkaz :

| 0x01| 0x04 | 0x00 | 0xC8| 0x00 | 0x12 | CRCLo | CRCHi |

NovarStatus je uložen počínaje input –registrem č. 30201(tzn. adresa 200 = 0xC8), délka struktury je 35 bytů, tedy nutno načíst 18 registrů (= 0x12 = 36 bytů, poslední byte je nevýznamný).

Odpověď :

| 0x01| 0x04 | 0x24 | registr 30201 Hi| reg. 30201 Lo| reg. 30202 Hi | reg. 30202 Lo |

..... | reg. 30218 Hi | reg. 30218 Lo | CRCLo | CRCHi |

Za adresou (0x01) a číslem funkce (0x04) následuje počet předávaných bytů (0x24 = 36) a obsah jednotlivých registrů. Registry obsahují strukturu NovarStatus následovně :

reg. 30201 Hi	-	SoftVersion –Hi
reg. 30201 Lo	-	SoftVersion –Lo
reg. 30202 Hi	-	DeviceNo - Hi
reg. 30202 Lo	-	DeviceNo – Lo
.....	-
reg. 30218 Hi	-	RegTime
reg. 30218 Lo	-	bez významu

(konec příkladu)

1.3 Datové struktury

Forma popisu odpovídá konvenci jazyka C. Vícebytové proměnné jsou uloženy v pořadí high-low (nejprve nejvyšší byte, poslední nejnižší byte).

```
//=====
Status :
typedef struct {
    uchar    HWEError;        /* chybovy priznak */
                                /* bit0...chyba checksum EPROM */
                                /* bit1...chyba RAM */
                                /* bit2...chyba checksum SEEPROM */
                                /* bit3...chyba kal. konstant v SEEPROM */
    uchar    OutputSwitchNo[14]; /* in switch-ons */
                                /* pocet sepnuti za poslednich max. 6 hodin */
                                /* jedenkrat za cca 6 hodin se "osvezuje" */
                                /* stav do OutputSwitchNo64 */
    uint     Event;           /* jev */
                                /* bit0...1=podproud */
                                /* bit1...1=nadproud */
                                /* bit2...1=prekomp/podkomp */
                                /* bit3...1=ztrata mericiho napet.signalu */
                                /* bit4...1=prekrojeni meze THD */
                                /* bit5...1=prekrojeni meze poctu sepnuti */
                                /* bit6...1=rezerva */
                                /* bit7...1=zpetne napajeni */
                                /* bit8...1=vypadek reg. kondenzatoru */
    uint     ActRelayState;   /* aktualni stav regulacnich rele, 1=sepnuto */
    uint     ReqRelayState;   /* pozadovany stav regulacnich rele */
    uchar    State;          /* lisi se od aktualniho, pokud se cekna na RelayOffTimeCnt */
                                /* stav */
                                /* bit0-3...Stav regulace */
                                /* bit4...1= zpusob pripojeni neznamy */
                                /*          tzn. nezadano nebo */
                                /*          neuspech rozpoznani UIMode */
                                /* bit5...1= hodnoty stupnu nezname */
                                /*          tzn. nezadano nebo */
                                /*          neuspech rozpoznani Ck */
    uint     AlarmSigActive;  /* indikuje okamžity stav signalizace alarmu */
                                /* bit0...1=alarm od podproudu */
                                /* bit1...1=alarm od nadproudu */
                                /* bit2...1=alarm od prekomp/podkomp */
                                /* bit3...1=alarm od ztraty napet.signalu */
                                /* bit4...1=alarm od prekrojeni meze THD */
                                /* bit5...1=alarm od prekrojeni meze poctu sepnuti */
                                /* bit6...1=rezerva */
                                /* bit7...1=zpetne napajeni */
                                /* bit8...1=vypadek reg. kondenzatoru */
                                /* bit14...1=neznamy zpusob pripojeni, aut. rozp. nasleduje */
                                /* bit15...1=nezname hodnoty stupnu, tzn. aut. rozp. stupnu */
                                /*          neuspesne nebo nulove -tento bit se nahazuje */
                                /*          pouze v AlarmSigActive!!! */
    uint     AlarmActionActive; /* indikuje ovlivnovani cinnost regulatoru vlivem alarmu */
                                /* bit0...0=od podproudu-odepnout */
                                /* bit1...0=od nadproudu-odepnout */
                                /* bit2...0=od prekomp/podkomp-nic */
                                /* bit3...0=od ztraty napet.signalu-vzdy tvrde odepnout */
                                /* bit4...0=od prekrojeni meze THD-odepnout */
                                /* bit5...0=od prekrojeni meze poctu sepnuti-nic */
                                /* bit6...0=rezerva */
                                /* bit7...1=zpetne napajeni */
                                /* bit8...1=vypadek reg. kondenzatoru */

```

```

uint    BadSteps;    /* bitova mapa vystupu, které jsou */
                        /* povazovany za vadne a dle nastaveni */
                        /* prislusne akce alarmu pripadne */
                        /* prechodne vyrazene z regulace */
                        /* bacha, nastavuji se i pri neaktivni */
                        /* prislusne signalizaci/akci, pouze */
                        /* to pak nema zadny dalsi vliv */

uint    SoftVersion; /* lowbyte...softversion */
                        /* highbyte...pokud ruzne od 00 a FF */
                        /* obsahuje cislo specialniho provedeni */

uint    DeviceNo;    /* vyrobni cislo */
uint    DeviceType; /* typ pristroje */
                        /* 2....NOVAR-314 RS*/
                        /* 3....NOVAR-206 */
                        /* 4....NOVAR-214 */
                        /* 5....NOVAR-106 */
                        /* 6....NOVAR-114 */
                        /* Status-34 bytu */
} SType;
//=====

```

EESStatus :

```

typedef struct {
uint    PrecisedSteps;
                        /* bitova mapa vystupu, které jsou již zpresnena */
                        /* 1=zpresneny vystup */
                        /* 0=dosud nezpresneny vystup */
                        /* vystupy se zpresnuji pouze pri nastavenem */
                        /* aut. rozpoznani stupnu */
char    MinCos; /* zaznamenane minimum kosinu */
uchar   Res0;  /* rezerva */
uchar   MaxTHD; /* zaznamenane maximum THD */
                        // kodovani THD:
                        // 0 -100 ... 0.00 az 50.0 po 0,5%
                        // 101-200 ... 52.5 az 300.0 po 2,5%
                        // 201-250 ... 310 az 800% po 10%
                        // 255 ... hodnota nedefinovana
uchar   Res1; /* rezerva */
uchar   MaxHar[6]; /* zaznamenane maximum 3-5-7-11-13-17 harmonicke */
                        // kodovani Har :
                        // 0 -100 ... 0.00 az 10.0 po 0,1%
                        // 101-200 ... 10.5 az 60.0 po 0,5%
                        // 201-254 ... 62.5 az 195.0 po 2,5%
                        // 255 ... hodnota nedefinovana
uint    OutputSwitchNo64[14]; /* pocet sepnuti */
                        /* v jednotkach 64 sepnuti */
                        /* pokud hodnota presahuje hodnotu limitu-nahodi */
                        /* se indikace alarmu */
                        /* rozsah 64000, tj. 4000 "kilosepnuti" */
uint    ManualStepValue; /* bitova mapa stavu vystupu v man. rezimu */
                        /* 0=vystup sepnut, 1=rozepnut */

                        /* od v.2.0 EESStatus rozsiren o nasledujicich 28 bytes : */
uint    OutputSwitchOnTime2H[14]; /* doba sepnuti vystupu */
                        /* v jednotkach 2 hodin */
                        /* rozsah 65000, tj. 130000 hodin */
} EESType; /* EESStatus-42 bytu, od v.2.0 70 bytu */
//=====

```

NovarStatus :

```

typedef struct {
uint    SoftVersion; /* lowbyte...softversion */
                        /* highbyte...pokud ruzne od 00 a FF */
                        /* obsahuje cislo specialniho provedeni */

```

```

uint DeviceNo; /* vyr.cislo */
uint DeviceType;
/* 2....NOVAR-314RS */
/* 3....NOVAR-206 */
/* 4....NOVAR-214 */
/* 5....NOVAR-106 */
/* 6....NOVAR-114 */
uint MTP; /* prevod proudoveho menice 1-6000 */
/* bity 14-0...primar je v jednotkach 5A */
/* bit 15...0 sekundar = 1A */
/* bit 15...1 sekundar =5A */
uchar Fr; /* bez vyznamu */
uint I; /* eff.hodnota proudu v 0,25mA*/
uint I50; /* eff.hodnota zakl.harm. 50Hz v 0,25mA*/
int Ir; /* real.slozka zakl.harm. 50Hz v 0,25mA*/
int Ii; /* im. slozka zakl.harm. 50Hz v 0,25mA*/
/* obe maji znamenko, pro C bude Ii zaporna */
int Fi; /* bez vyznamu */
char Kos; /* účinník cos fi, kódování v procentech : */
/* if(Kos == 0) -> cos fi = 0.00L */
/* . */
/* if(Kos == 99) -> cos fi = 0.99L */
/* if(Kos == 100) -> cos fi = 1.00 */
/* if(Kos == -99) -> cos fi = 0.99C*/
/* . */
/* if(Kos == -1) -> cos fi = 0.01C */
/* if(Kos == -100) -> cos fi = 0.00C*/
/* . */
/* if(Kos == 127)-> cos fi nedefinován (např. při nulovém I) */
uchar THD; /* THD, kodovani viz EEStatus */
uchar Har[6]; /*3-5-7-11-13-17. harmonicka, kodovani viz EEStatus */
uchar Res0;
uint ActRelayState; /* stav vystupu, 1=zapnuty */
uchar Res1;
uchar Res2;
uchar RegState; /* stav regulace */
// STATEMASK 0x0F /* spodni 4 bity-stav regulace : */
/* rozlisuje stav regulatoru v rezimu automat */
// STATEINIT 0x00 /* po resetu */
// STATETEST 0x01 /* probiha test */
// STATEUIMODERE 0x02 /* probiha UIMode-recognition („AF“) */
// STATEUIMODEUK 0x03 /* UIMode-unknown („F=0“) */
// STATECLVALUESRE 0x04 /* probiha CLValues-recognition („AC“) */
// STATECLVALUESUK 0x05 /* CLValues-unknown („C=0“) */
// STATERUN 0x06 /* probiha regulace*/
// STATESTANDBYCLOFF 0x07 /* nelze regulovat- vypnout vse mimo pevných */
// STATESTANDBYALLOFF 0x08 /* nelze regulovat- vypnout vse */
// STATEIDDL 0x09 /* nelze regulovat- nejsou merena data */
// STATEMANUAL 0x0F /* nelze regulovat- je v manualu */
/* horni nibble-masko nestandardnich stavu */
// STATEUIMODEUNKNOWN 0x10 /* nezadano / neusp. aut. rozp. („F=0“) */
// STATECLVALUESUNKNOWN 0x20 /* nezadano / neusp. aut. rozp. („C=0“) */
// STATEVOLTAGEBAD 0x40 /* neni merici napeti („U=0“) */
// STATECURRENTLOW 0x80 /* neni merici proud („I=0“) */
uchar StateLEDs;
/* bit0...TrendL */
/* bit1...TrendLBlik */
/* bit2...TrendC */
/* bit3...TrendCBlik */
/* bit4...PwrReverse */
/* bit5...Alarm */
/* bit6...nic */
/* bit7...Error */

```

```

    uchar   RegTime; /* doba do dalsiho reg. zasahu v % */
                /* klesa z 100 az do 0 */
} NSType; /* NovarStatus-35 bytu */
//=====

```

Config :

```

typedef struct {
    char     ReqCos; /* nastaveny cos- rozsah -90 az +80 */
                /* 0x7F-hodnota dosud nezadana */
    uchar    SwitchDelayL; /* pro nedokompenzovani */
    uchar    SwitchDelayC; /* pro prekompenzovani */
                /* udava dobu reg. zasahu pri reg. odchylce rovne Ck */
                /* pri vetsi potrebe se tato hodnota snizuje kvadraticky */
                /* kodovani : */
                /*      0...5 sec */
                /*      1...10 sec */
                /*      2...15 sec */
                /*      3...20 sec */
                /*      4...30 sec */
                /*      5...60 sec */
                /*      6...120 sec */
                /*      7...180 sec */
                /*      8...300 sec */
                /*      9...600 sec */
                /*     10..1200 sec */
                /*     jinak...neplatna hodnota */
                /* zmena v.2.1: */
                /*     navic bit 7...0= kvadraticke zkrac d. regulace */
                /*     ...1=linearni zkracovani */
    char     Res0; /* rezerva, nastavit konstantne na 0x01 ! */
    char     Res1; /* rezerva, nastavit konstantne na 0x01 ! */
} RegParType;

typedef struct {
    uchar    RegMode; /* nastaveni reg. modu */
                /* bit0...0=manual, 1=automat */
                /* bit1...0=evaluation of tarif2 input */
                /* bit2...1=automaticke rozpoznavani Ck */
                /* bit3...1= password not retained- required */
                /*     0= password retained - not required */
                /*     pro 0 se po zapnuti nemusí heslo zadavat! */
                /* bit4...res. */
                /* bit5...res. */
                /* bit6...res. */
                /* bit7...res. */
    uchar    Res0; /* rezerva */
                /* nyni parametry regulace pro 2 tarify */
    RegParType RegPar[2]; /* hodnoty [1] plati pro tarif c.2!!! */
    uint     MTP; /* prevod proudoveho menice 1-9950 */
                /* bity 14-0...primar je v jednotkach 5A */
                /* sekundar : */
                /* bit 15...0= xxx/1A */
                /* bit 15...1= xxx/5A */
    uchar    SwitchBlockDelay; /* blokovani znovuzapnuti */
                /* kodovani : */
                /*      0...5 sec */
                /*      1...10 sec */
                /*      2...20 sec */
                /*      3...30 sec */
                /*      4...60 sec */
                /*      5...120 sec */
                /*      6...300 sec */
                /*      7...600 sec */
                /*      8...1200 sec */

```

```

/*          jinak...neplatna hodnota          */
uchar    UIMode;
/* zpusob pripojeni fazi U a I */
/* udava, mezi jake faze je zapojeno U */
/* predpoklada se, ze MTN je ve fazi 1 */
/* a je orientovan spravne vzhledem ke k,l */
/* bity 2-0:          */
/* kodovani Ukl (0...stredni, 1-2-3...faze*/
/* pro bit 3=1, tzn. fazove napeti : */
/* 1...U10 (k..na fazi 1, l...na stredni */
/* 2...U20 */
/* 3...U30 */
/* 4...U01 */
/* 5...U02 */
/* 6...U03 */
/* pro bit 3=0, tzn. sdruzene napeti : */
/* 1...U12 */
/* 2...U23 */
/* 3...U31 */
/* 4..U21 */
/* 5..U32 */
/* 6..U13 */
/* horni nibble(bity 7-4) : */
/* pokud bity 0-3 mimo povol. rozsah: */
/* horni nibble =0...nerozpoznano aut. */
/* vyhodnocovacem!!!- ceká se na dalsi */
/* zasah obsluhy ! */
/* pokud bity 0-3 mimo povol. rozsah a horni */
/* nibble je 1-F...UiMode dosud nezadano */
/*          a bude se rozpoznavat */
uchar    CSRatio;
/* compensation step ratio */
/* 0x00...-individuální nastavení CLVal */
/*      při této hodnotě zůstanou hodnoty CLVal */
/*      i při zavolání funkce SetCLValues */
/*      nedotčeny(lze je indiv. editovat) */
/*      Možný postup: zadat hodnotu > 0 (předpřipravit) */
/*      následně v editaci CLVal upravit -> přitom */
/*      se CSRatio automat. nastaví na 0x00! */
/*      -rovněž úspěšný průběh automat. regulace */
/* 0xFF-neúspěšný průběh automat. rozpoznání Ck! */
/* nastaví CSRatio na 0xFF ! */
/* 1...1:1:1:1 */
/* 2...1:1:2:2 */
/* 3...1:1:2:2:4 */
/* 4...1:1:2:3:3 */
/* 5...1:1:2:4:4 */
/* 6...1:1:2:4:8 */
/* 7...1:2:2:2:2 */
/* 8...1:2:3:3:3 */
/* 9...1:2:3:4:4 */
/* 10...1:2:3:6:6 */
/* 11...1:2:4:4:4 */
/* 12...1:2:4:8:8 */
uchar    Ck;
/* 0,02 až 2A po 0,01A */
/* pokud je CSRatio 0, nezobrazuje se */
uchar    Steps;
/* počet použitých indukt. a kapacitních stupňů */
/* nibble 3-0...CSteps=pocet kap. stupňů */
/* bity 5-4...LSteps=pocet ind.stupňů-max 3 */
/* induktivní stupně jsou vždy vzadu, za posledním */
/* kapacitním stupněm */
/* CSteps+LSteps <= MAXSTEPS */
uchar    QuickSteps;
/* pouze pro Novar-314RS: počet použitých stupňů */
/* tranzistorové sekce; pro ostatní typy bez významu */
int       CLVal[14];
/* measured Ck */
/* při automat. rozpoznání Ck obsahuje */
/* jednotlivé změřené Ck- při induktivním */

```

```

/* stupni muze byt i zaporne */
/* obsahuje hodnotu im. slozky proudu 50Hz */
/* v jednotkach 0,25mA, kondik kladny */
/* maximalni rozsah je +/-32000, tj. +/-8A */
uint FixedSteps; /* bitova mapa vystupu, které jsou */
uint FixedStepValue; /* nastavene jako pevne; 0=pevny vystup */
/* bitova mapa hodnot pevnych */
/* 0=vystup sepnut, 1=rozeprnut */
/* pevne nastavene vystupy jsou vyrazeny */
/* z regulace, ale rozlozeni CSteps a LSteps */
/* zustava-nic se neposouva, vsechny */
/* vystupy, které zustavaji v regulaci */
/* maji puvodni chrakter(C/L) i vahu dle */
/* CSRatio bez ohledu na to, kolik vystupu */
/* je nastaveno jako pevnych */
char LCosMargin; /* mezni kosinus pro prip. odp. tlumivky */
uchar QuickControlSpeed; /* pouze pro Novar-314RS: rychlost regulace tranz. */
/* sekce / blokovani znovuzapnuti vystupu tranz. sekce */
/* hodnota pocet reg./sec blok.doba[s]
// 0 1 10 (default)
// 1 1 5.0
// 2 1 2.0
// 3 1 1.0
// 4 2 10
// 5 2 5.0
// 6 2 2.0
// 7 2 1.0
// 8 2 0.5
// 9 3 10
// 10 3 5.0
// 11 3 2.0
// 12 3 1.0
// 13 3 0.6
// 14 3 0.3
// 15 4 10
// 16 4 5.0
// 17 4 2.0
// 18 4 1.0
// 19 4 0.7
// 20 4 0.5
// 21 4 0.2
// 22 5 10
// 23 5 5.0
// 24 5 2.0
// 25 5 1.0
// 26 5 0.8
// 27 5 0.6
// 28 5 0.4
// 29 5 0.2
uint AlarmSig; /* kdy se bude signalizovat pomoci rele */
/* bit0...0=alarm od podproudu-hned */
/* bit1...0=alarm od nadproudu-hned */
/* bit2...0=alarm od prekomp/podkomp trvajici >= 15 minut! */
/* bit3...0=alarm od ztraty napet.signalu-hned */
/* bit4...0=alarm od prekroceni meze THD trvajici >= 5minut! */
/* bit5...0=alarm od prekroceni meze poctu sepnuti-hned */
/* bit6...0=rezerva */
/* bit7...0=pri zpetnem nap. trvajicim >= 5minut */
/* bit8...0=pri vypadku reg. kondenzatoru */
uint AlarmAction; /* kdy se bude ovlivnovat cinnost regulatoru */
/* bit0...0=od podproudu-po 10 sec odepnout */
/* bit1...0=od nadproudu-nelze nastavit */
/* bit2...0=od prekomp/podkomp-nelze nastavit */
/* bit3...0=od ztraty napeti-vzdy tvrde hned (5sec) odepnout */
/* nejde shodit!, vzdy aktivni! */

```

```

/* bit4...0=od prekročení meze THD-po 5 min.odepnout */
/* bit5...0=od prekročení meze počtu sepnutí-nelze nastavit*/
/* bit6...0=od nap. nesymetrie-vždy tvrdě hned odepnout */
/* bit7...0=při trvajícím zpet. nap. po 5min odepnout */
/* bit8...0=při výpadku reg. kondenzátoru-nelze nastavit */
uchar THDLimit; /* mezní povolená hodnota; kodováni viz EEStatus */
uchar SwitchNoLimit; /* pokud překročena po dobu 5 minut, odepne stupně */
/* od 10000 do 2000000 */
/* v jednotkách 10000 sepnutí*/
uchar Res3; /* rezerva */
uchar Res4; /* rezerva */
/* nyní část, která se nedá menit přes komunikační linku */
uchar DeviceAddr; /* pro komunikaci */
uchar RemoteBdRate; /* low nibble = Bd-rate*/
/* 2...300 Bd */
/* 3...600 Bd */
/* 4..1200 Bd */
/* 5..2400 Bd */
/* 6..4800 Bd */
/* 7..9600 Bd */
// od v.1.9H v high-nibble protokol:
// bity 7-6-5-4:
// bit7...rezerva (0)
// bit6...0 = prokol KMB
//      1 = ModBus RTU
// bity 5,4...parita pro ModBus RTU:
// bit5.....0=zadná parita(tzn. 2stopbity)
//      1=parita:
// bit4.....0=suda
//      1=lichá
uint ConfigCRC; /* config má vlastní CRC, ale ten si nastavuje Novar sam, az, */
/* když si ukládá Config do EEPROM. Sem není nutno nic */
/* nastavovat- obsah libovolný */
} CType; /* Config - 66 bytů */

```

```

//=====

```

NovarSetMap :

```

typedef struct { /* nulování provozních maxim a nastavení režimu */
uchar ClearLimit; // bit0= 1 ... nulovat MinCos
// bit1= 1 ... nulovat MaxTHD
// bit2= 1 ... nulovat MaxHar
uint ClearSwitchNo; // bit0-13... snulovat počet sepnutí odpovídajících výstupů
uchar Switch; // bit0=zablokovat editaci ( při následné editaci
// bude vyžadováno heslo)
// bit1=prepnout do regulace(pokud je v manuálním režimu)
// bit2=reinicializace regulatoru
// bit3=smazat chybu HWEError
// od v.2.0 rozšířeno o 2 byty:
uint ClearSwitchOnTime; // bit0-13... nulovat doby sepnutí odpovídajících výstupů
} NovarSetMapType; /* NovarSetMap -6 bytů, od verze 2.0 8 bytů */

```

```

//=====

```