

Popis ovládání regulátorů řady Novar-1xxx prostřednictvím dálkové komunikační linky

Příručka pro programátory

Stav 06/2011

Regulátory jalového výkonu Novar-1106/1114/1206/1214/1312 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“. Lze je nastavit i protokol Modbus-RTU. Rychlost komunikace je nastavitelná dle technických parametry regulátoru.

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 (110 bytů), celkem tedy 144 bytů.

Příklad :

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

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

Příkaz nemá tělo.

Struktura odpovědi :

| 0x01| 0x93| 0x00 | tělo zprávy = Status+EEStatus (144 bytů) | checksum |

Druhý byte, tedy délka zprávy bez závěrečné checksum, má hodnotu $144 + 3 = 147 = 0x93$.

1.2.1.1.2 Přečtení nastavení přístroje (Config) – 0x16

Zpráva č. 0x16. Přístroj předá tzv. Config (80 bytů, od verze 1.3 100 bytů).

Příkaz :

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

Odpověď :

| 0x01| 0x53| 0x00 | tělo zprávy = Config (80 bytů) | checksum |

1.2.1.1.3 Zápis nastavení do přístroje (Config) – 0x17

Zpráva č. 0x17. Zápis tzv. Config (80 bytů, od verze 1.3 100 bytů) do přístroje.

Příkaz :

| 0x01| 0x53 | 0x17 | tělo zprávy = Config (80 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 (60 bytů, pro modely Novar1414 100 bytů).

Příkaz :

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

Odpověď :

| 0x01| 0x3F| 0x00 | tělo zprávy =NovarStatus (60 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| 0x09 | 0x31 | tělo zprávy = NovarSetMap (6 bytů) | checksum |

(druhý byte, tedy délka zprávy bez závěrečné checksum, bude mít hodnotu $6 + 3 = 9 = 0x09$).

Odpověď :

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

1.2.2 Komunikační protokol Modbus-RTU

Přístroje 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.

Poznámka : Dle definice protokolu Modbus se používá přenos 9 bitů i při nastavení „bez parity“, paritní bit se přitom ignoruje. Některé aplikace však při tomto nastavení používají 8-bitový přenos a přístroj pak na takovéto zprávy nereaguje. V takovém případě může pomoci nastavit v aplikaci protokol na 2 stopbity.

Jsou implementovány následující funkce :

| kód funkce | název funkce | aplikace |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 03 | Read Holding Registers | čtení Config : – registry 40101-40140 (adresovány 100 – 139, pro verzi firmware do 1.2) – registry 40101-40150 (adresovány 100 – 149, pro verzi firmware od 1.3) |
| 04 | Read Input Registers | čtení Status+EEStatus – registry 30101-30172 (adr. 100 – 171) čtení NovarStatus : – registry 30201-30230 (adr. 200 – 229 , platí pro všechny modely mimo Novar1414) – registry 30201-30250 (adr. 200 – 249 , platí pro modely Novar1414) |
| 06 | Preset Single Register | zápis Config – rozsah registrů viz výše zápis NovarSetMap – registry 40201-40203 (adr. 200 – 202) |
| 08 | Diagnositics – 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 1 :

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

Příkaz :

| 0x01| 0x04 | 0x00 | 0xC8| 0x00 | 0x1E | 0xF1 = CRCLo | 0xFC = CRCHi |

NovarStatus je uložen počínaje input –registrem č. 30201(tzn. adresa 200 = 0xC8), délka struktury je 60 bytů, tedy nutno načíst 30 registrů (= 0x1E = 60 bytů).

Odpověď :

| 0x01| 0x04 | 0x3C | registr 30201 Hi| reg. 30201 Lo| reg. 30202 Hi | reg. 30202 Lo |
 | reg. 30230 Hi | reg. 30230 Lo | CRCLo | CRCHi |

Za adresou (0x01) a číslem funkce (0x04) následuje počet předávaných bytů (0x3C = 60) 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. 30230 Hi | - | RegTime |
| reg. 30230 Lo | - | bez významu |

Příklad 2 :

Načtení okamžitého účinníku (proměnná „Kos“ ze struktury NovarStatus) pomocí funkce Read Input Registers, předpokládáme adresu přístroje 1 :

Příkaz :

| 0x01| 0x04 | 0x00 | 0xD1| 0x00 | 0x01 | 0x61| 0xF3 |

Proměnná „Kos“ je uložena ve struktuře NovarStatus v bytu č. 20, to znamená že je uložena ve spodní části (šestnáctibitového) input-registru č. 30210 (tzn. adresa 209 = 0xD1).

Odpověď :

| 0x01| 0x04 | 0x02 | 0x8B| 0x4B | 0x9F| 0xF7 |

Registr obsahuje hodnotu 0x8B4B; ve spodním byte je hodnota proměnné „Kos“ = 0x4B = 75. Hodnota okamžitého účinníku je tedy 0.75 .

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 */
                                /* vyznam bitu dle AlarmSigActive */
    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 okamžitý stav signalizace alarmu */
                                /* bit0...0=alarm od podproudu */
                                /* bit1...0=alarm od nadproudu */
                                /* bit2...0=alarm od ztraty napet.signalu */
                                /* bit3...0=alarm od podpeti */
                                /* bit4...0=alarm od prepeti */
                                /* bit5...0=alarm od prekroceni meze THDI */
                                /* bit6...0=alarm od prekroceni meze THDU */
                                /* bit7...0=alarm od prekroceni meze CHL */
                                /* bit8...0=alarm od chyby kompenzace */
                                /* bit9...0=alarm pri zpetnem nap. */
                                /* bit10..0=alarm od prekroceni meze poctu sepnuti */
                                /* bit11..0=alarm pri vypadku reg. kondenzatoru */
                                /* bit12..0=alarm od prehrati */
                                /* bit13..0=alarm od externiho vstupu (pouze pro N12xx) */
                                /* 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 */
                                /* vyznam bitu dle AlarmSigActive */
    uint    BadSteps;           /* bitova mapa vystupu, ktere 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;           /* vyrobní číslo */

```

```

uint DeviceType; /* typ pristroje */
/* 0x12...N1312 */
/* 0x13...N1206 */
/* 0x14...N1214 */
/* 0x15...N1106 */
/* 0x16...N1114 */
} SType; /* Status-34 bytu */
//=====

EEStatus :

typedef struct {
uint PrecisedSteps;
/* bitova mapa vystupu, ktere jsou jiz zpresnene */
/* 1=zpresneny vystup */
/* 0=dosud nezpresneny vystup */
/* vystupy se zpresnuji pouze pri nastavenem */
/* aut. rozpoznani stupnu */
uchar MaxTHD[2]; /* maximalna dosazena hodnota */
// 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 MaxCHL; /* maximalni dosazena hodnota, kodovani viz CHL */
// kodovani CHL:
// 0 -150 ... 0 az 150 po 1%
// 151-200 ... 155 az 400 po 5%
// 201-250 ... 410 az 900 po 10,0%
// 255 ... hodnota nedefinovana
uchar MaxHar[9]; /* max. dos. hodnota 3-5-7-9-11-13-15-17-19 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
uchar Res0; /* rezerva */
uchar Res1; /* rezerva */
char MaxT; /* maximalna dosazena hodnota, kodovani viz T*/
char MinKos; /* minimalni dosazena hodnota Kos*/

int MaxAveP; // maximalni dosazena hodnota prumerneho P
int MaxAveQ; // maximalni dosazena hodnota prumerneho Q
int MaxAveDeltaQ; // maximalni dosazena hodnota prumerneho chybejiciho DeltaQ
// rozmer jako proud, nutno nasobit nom. napetim

float AveP[2]; // klouz. prumer P (pro vnitri potrebu pristroje)
float AveQ[2]; // klouz. prumer Q (pro vnitri potrebu pristroje)
float AveDeltaQ; // klouz. prumer DeltaQ (pro vnitri potrebu pristroje)
ulong AvePQCounter[2]; // citac delky klouz. okna (pro vnitri potrebu pristroje)

uint OutputSwitchNo64[14]; /* pocet sepnuti */
/* v jednotkach 64 sepnuti */
/* pokud hodnota presahuje hodnotu limitu, nahodi se indikace alarmu */
/* rozsah 64000, tj. 4000 "kilosepnuti" */
uint OutputSwitchOnTime2H[14]; /* doba sepnuti vystupu */
/* v jednotkach 2 hodin */
/* rozsah 65000, tj. 130000 hodin */
uint ManualStepValue; /* bitova mapa stavu vystupu v man. rezimu */
/* 0=vystup sepnut, 1=rozepnut */
} EESType; /* EEStatus-110 bytu */
//=====

```

NovarStatus :

```

typedef struct {
uint   SoftVersion; /* lowbyte....softversion */
                /* highbyte...pokud rozne od 00 a FF , obsahuje c. spec. provedeni */
uint   DeviceNo; /* vyr.cislo */
uint   DeviceType;
                /* 0x12...N1312 */
                /* 0x13...N1206 */
                /* 0x14...N1214 */
                /* 0x15...N1106 */
                /* 0x16...N1114 */
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; /* frekvence
                // kodovani :
                // po 0,1Hz, 55Hz=0x80, 0= 42,2Hz, 254=67,6Hz, 255=nezmereno
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*/
                // pozn. : pro Novar-1414 se jedna o prumer proudu vsech 3 fazi
int    Fi; /* uhel Ir a Ij ve stupnich */
char   Kos; /* ucinnik cos fi, kodovani 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) */
                // pozn. : pro Novar-1414 se jedna o trojfazovy ucinik
uchar  THD[2]; /* 0...U, 1...I
                // 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  Har[2][9]; /* 0...U, 1...I, hodnota 3-5-7-9-11-13-15-17-19 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
                // pozn. : pro Novar-1414 vyse uvedene THDI a proudove harmonicke
                // nemaji vyznam; viz jednotlivé fazové hodnoty níže
uint   U; /* efektivni hodnota U, kodovani v 0,1V, 0xFFFF...nedefinovano
uint   U50; /* hodnota napeti zakl harmonicke slozky jednotkach 0,1V
uchar  CHL; /* Capacitor Harmonic Load factor
                // kodovani CHL:
                // 0 -150 ... 0 az 150 po 1%
                // 151-200 ... 155 az 400 po 5%
                // 201-250 ... 410 az 900 po 10,0%
                // 255 ... hodnota nedefinovana
int    DeltaI; /* chybejici jalovy proud, rozmer stejný jako I
char   T; /* teplota ve stupnich C
uchar  Input; /* bit 0: ....0, vstup 2 tarifu neseprnut
                // ....1, vstup 2 tarifu seprnut

```

```

uchar   Res0;   /* rezerva */
uchar   MTN;    // prevod MTN
           // kodovani :
           //      0=1(tj.bez MTN), zobrazeni U ve voltech, jinak v kV
           //      1=10 az 100=1000, tedy 1000/100 az 100000/100
           //      101=1100 az 140=5000, tedy 110000/100 az 500000/100
           //      140...500kV/100V, maximum
           //      >140...bez MTN, prime mereni, stejne jako 0
uchar   Unom;   // nominalni merici napeti (bez prevodu MTN)
           // kodovani :
           // 9           ...50V
           // 10          ...55V
           // 11          ...58V
           // 12 az 150   ...60V az 750V po 5 V
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 */
           /* pevnych vystupu */
           // 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 */
uchar   Res3;

```

// pro vsechny modely mimo Novar1414 je zde konec struktury. Pro modely Novar1414 struktura
// pokracuje nasledujicimi fazovymi hodnotami :

```

uint    I[3]; /* eff.hodnota fazovych proudu L1, L2, L3 v 0,25mA*/
char    Kos[3]; /* jednotlivé fazové účinníky */
uchar   THDI[3]; /* jednotlivá fazová THDI
uchar   HarI[9][3]; /* jednotlivé fazové harmonické složky proudu
uchar   Res4;

```

```

} NSType; /* NovarStatus-60 bytu ; pro Novar-1414 100 bytu*/

```

```

//=====

```

Config :

```

typedef struct {
    char    ReqCos;          /* nastaveny cos- rozsah -80 az +80 */
                          /* 0x7F-hodnota dosud nezadana */
                          /* nebo ve stupnich : 101= +10 st. , az 121= -10 st.
    uchar   SwitchDelayL;   /* pro nedokompenzovani */
    uchar   SwitchDelayC;   /* pro prekompenzovani */
                          /* udava dobu reg. zasahu pri reg. odchylce rovne Ck */
                          /* pri vetsi odchylce se tato hodnota koriguje-snizuje dle
                          /* bitu 7 takto :
                          /* bit 7 ...0= kvadraticke zkracovani doby regulace(standard)
                          /*          ...1=linearni zkracovani

                          /* kodovani : bity 3-0          */
                          /*    0...5  sec          */
                          /*    1...10 sec          */
                          /*    2...15 sec          */
                          /*    3...20 sec          */
                          /*    4...30 sec          */
                          /*    5...45 sec          */
                          /*    6... 1  min          */
                          /*    7... 1 min 30 sec      */
                          /*    8... 2  min          */
                          /*    9... 3  min          */
                          /*   10... 4  min          */
                          /*   11... 5  min          */
                          /*   12... 7  min          */
                          /*   13... 10 min          */
                          /*   14... 15 min          */
                          /*   15... 20 min          */
                          /*   jinak...neplatna hodnota      */

    uchar   ReqCosBandWidth; // sirka regulacniho pasma
                          /* v jednotkach 0,005, rozsah 0-8, tj. 0.000 az 0.040
                          /* (ReqCos+/-0.02) pasma necitlivosti, od ReqCos na obe strany,
    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 rozpoznani Ck */
                          /* bit3...1= password not retained- required*/
                          /*    0= password retained - not required*/
                          /*    pro 0 se po zapnuti nemusi heslo zadavat! */
                          /* bit 4...pokud je TARIF2MASK aktivni,
                          /* ...1(default)...tarif 2 se uplatni pri aktivnim vstupu 2.tarifu
                          /* ...0.....tarif2 se uplatni pri zpetnem napajeni
                          /* bit5...1...pokud bit 2 zaroven v 1, aut. rozpoznani Ck typu
                          /* AUTO, tedy jen pokud vsechny stupne nulove ci nezname */
                          /* bit 6...= 1...standardni regulace
                          /*    = 0...linearni spinani, pro filtry harmonickych
                          /* 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 shodne jako SwitchDelayL/C
    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 */
/* compensation step ratio */
/* 0x00...-individuální nastavení CLVal */
/* při této hodnotě zůstane hodnota CLVal */
/* i při zavolání funkce SetCLValues */
/* nedotčený(lze je indiv. editovat) */
/* Možný postup: zadat hodnotu > 0 (předpřipravit) */
/* následně v editaci CLVal upravit -> přitom */
/* se CSRatio autom. nastaví na 0x00! */
/* -rovněž úspěšný průběh autom. regulace */
/* 0xFF-neúspěšný průběh autom. rozpoznání Ck! */
/* nastaví CSRatio na 0xFF ! */
/* 1...1:1:1:1 */
/* 2...1:1:2:2 */
/* 3...1:1:2:4 */
/* 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 */
uchar QuickSteps; /* pouze pro Novar-1312: počet stupňů pro rychlou regulaci */
/* pro ostatní typy bez významu */
int CLVal[14]; /* hodnoty stupňů */
/* obsahuje hodnotu im. složky proudu 50Hz */
/* v jednotkách 0,25mA, kondik kladný */
/* maximální rozsah je +/-32000, tj. +/-8A */
uint FixedSteps; /* bitová mapa výstupu, které jsou */
/* nastaveny jako pevné; 0=pevný výstup */
uint FixedStepValue; /* bitová mapa hodnot pevných */
/* 0=výstup sepnut, 1=rozepnut */
/* pevné nastavené výstupy jsou vyraženy */

```

```

/* 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 pevných */
char    LCoMargin; /* mezni kosinus pro prip. odp. tlumivky */
uchar   QuickControlSpeed; /* pouze pro Novar-1312: rychlost regulace rychle */
/* sekce / blokovani znovuzapnuti vystupu rychle. 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                  5.0
//      5              2                  2.5
//      6              2                  1.0
//      7              2                  0.5
//      8              3                  3.3
//      9              3                  1.7
//      10             3                  0.7
//      11             3                  0.3
//      12             5                  2.0
//      13             5                  1.0
//      14             5                  0.4
//      15             5                  0.2
//      16             10                 1.0
//      17             10                 0.5
//      18             10                 0.2
//      19             10                 0.1
uint     AlarmSig; /* kdy se bude signalizovat pomoci rele */
uint     AlarmAction; /* kdy se bude ovlivnovat cinnost regulatoru */
/* bit0...0=alarm od podproudu */
/* bit1...0=alarm od nadproudu */
/* bit2...0=alarm od ztraty napet.signalu */
/* bit3...0=alarm od podpeti */
/* bit4...0=alarm od prepeti */
/* bit5...0=alarm od prekroceni meze THDI */
/* bit6...0=alarm od prekroceni meze THDU */
/* bit7...0=alarm od prekroceni meze CHL */
/* bit8...0=alarm od chyby kompenzace */
/* bit9...0=alarm pri zpetnem nap. */
/* bit10...0=alarm od prekroceni meze poctu sepnuti */
/* bit11...0=alarm pri vypadku reg. kondenzatoru */
/* bit12...0=alarm od prehrati */
uchar    FixedStepsFH; // alternativni funkce pevných stupnu dvou posledních stupnu
// pokud je prislusny stupen nastaven na pevny, definuje
// alternativni funkci tohoto vystupu
// bit 1,0...pro posledni stupen
// bit 0...1=alt. funkce vypnuta(zaroven by mel byt i bit 1 v 1)
//      0=alternativni funkce zapnuta, bit urcuje typ alt. funkce
// bit 1...1=F(fan)
//      0=H(heating)
// bit 3,2...dto pro predposledni stupen
uchar    MTN; // prevod MTN
// kodovani :
// 0=1(tj.bez MTN), zobrazeni U ve voltech, jinak v kV
// 1=10 az 100=1000, tedy 1000/100 az 100000/100
// 101=1100 az 140=5000, tedy 110000/100 az 500000/100
// 140...500kV/100V, maximum
// >140...bez MTN, prime mereni, stejne jako 0
uchar    Unom; // nominalni merici napeti, bez MTN
// kodovani :
// 9      ...50V
// 10     ...55V

```

```

// 11 ...58V
// 12 az 150...60V az 750V po 5 V
char TFHLimit[2]; // 0...teplota zapnuti vetraku,
// 1...teplota zapnuti topeni, ve stupnich C
uchar ULimit[2]; // mez napeti v prostych procentech Unom:
// 0... pro podpeti, 1...prepeti
// rozsah 10% - 150%
uchar THDLimit[2]; // mez THD, 0...pro napeti, 1...pro proud
// kodovano jako ActData.THDLimit
// pro 0xFF- vypnuto */
uchar CHLLimit; // mez CHL, zakodovano jako ActData.CHL
uchar TLimit; // mez teploty, zakodovano jako ActData.T
uchar SwitchNoLimit; /* mez poctu sepnuti, od 10000 do 2000000 */
/* v jednotkach 10000 sepnuti*/
uchar TCF; // bit 0...1=zobrazeni teploty Celsius, 0=Fahrenheit
uchar ScanFreq; // bity 1, 0 :
// 1 x auto
// 0 1 pevne 50 Hz
// 0 0 pevne 60 Hz
uchar Res3; /* rezerva */
uchar Res4; /* rezerva */
/* nyni cast, ktera se neda menit pres komunikacni linku */
uchar DeviceAddr; /* pro komunikaci */
uchar RemoteBdRate; /* low nibble = Bd-rate*/
/* 6..4800 Bd */
/* 7..9600 Bd */
/* 8..19200 Bd */
// 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=zadna parita(tzn. 2stopbity)
// 1=parita:
// bit4.....0=suda
// 1=licha
uchar AvePQWindowLength; // delky klouz. oken pro prum. P/Q/cos a
// minima/maxima P/Q/dQ/cos
// low nibble...pro prumery, high nibble...pro maxima/minima
// vyznam: 0...1min.klouzani 1/12
// 1...15min.....1/180
// 2...1hod.....1/720
// 3...8hod.....1/5760
// 4...1den.....1/17280
// 5...7dni.....1/120960
// jinak...7 dni
uchar UIMode23; // vyznam pouze pro Novar-1414 :
// rozsireni UIMode : bity 2-0 ...znaci pozici U2
// : bity 6-4 ...znaci pozici U3

```

// od verze firmware 1.3 vlozeno nasledujicich 20 bytu, obsahujicich nastaveni offsetu :

```

uchar RemoteControl; // bez významu
char ExtCosValue[5]; // bez významu
uchar ExtCosValueRes[4]; // bez významu
int OffsetCLVal[2]; // hodnota offsetu; pro kazdy tarif zvlast
uchar OffsetMode; // bit0 = 1 ... normalni regulace funkce bez offsetu
// 0 ... regulace s offsetem
uchar RemoteControlTimeout; // bez významu
uchar OffsetRes[4]; // bez významu
// konec vložky
uint ConfigCRC; /* CRC Configu. Neni nutno nastavovat, obsah libovolny */
} CType; /* Config - 80 bytu ; od verze firmware 1.3 100 bytu */

```

```
//=====
```

NovarSetMap :

```
typedef struct {
    /* nulovani provoznich maxim a nastaveni rezimu */
    uchar ClearLimit; // bit0= Acos, APac, Apre
                    // bit1= minCos,maxPac,maxPre,maxdPre
                    // bit2= MaxTemp
                    // bit3=maxCHL,maxTHDU,maxharU
                    // bit4=maxTHDI
    uint ClearSwitchNo; // bit0-13... snulovat pocet sepnuti odpovidajich vystupu
    uchar Switch; // bit0=zablokovat editaci ( pri nasledne editaci
                // bude vyžadovano heslo)
                // bit1=prepnout do regulace(pokud je v manualnim rezimu)
                // bit2=reinicializace regulatoru
                // bit3=smazat chybu HWError
    uint ClearSwitchOnTime; // bit0-13... nulovat doby sepnuti odpovidajich vystupu
} NovarSetMapType; /* NovarSetMap -6 bytu */
```

```
//=====
```