KMB systems, s.r.o.
Dr. M. Horákové 559, 460 06 Liberec 7, Czech Republic
tel. +420 485 130 314, fax +420 482 736 896
email : kmb@kmb.cz, internet : www.kmb.cz

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

Příručka pro programátory

Verze 01/2019

Regulátory jalového výkonu Novar-1106/1114/1206/1214/1312/1414 mohou být vybaveny dálkovou komunikační linkou s rozhraním RS-232 nebo RS-485. Pomocí této linky mohou být monitorovány a ovládány z nadřazeného řidicího 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

1.2.1.1.1Přečtení 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.

1.2.1.1.2Přečtení nastavení přístroje (Config) – 0x16

Zpráva č. 0x16. Přístroj předá tzv. Config (100 bytů).

Příkaz :

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

Odpověď :

| 0x01| 0x67| 0x00 | ..... tělo zprávy = Config (100 bytů) ..... | checksum |

1.2.1.1.3Zápis nastavení do přístroje (Config) – 0x17

Zpráva č. 0x17. Zápis tzv. Config (100 bytů) do přístroje.

Příkaz :

| 0x01| 0x67| 0x17| ..... tělo zprávy = Config (100 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.

1.2.1.1.4Přečtení stavu přístroje (NovarStatus) – 0x30

Zpráva č. 0x30. Přístroj předá informace o stavu přístroje potřebné zejména pro on-line vizualizaci, tzv. NovarStatus (60 bytů).

Příkaz :

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

Odpověď :

| 0x01| 0x3F| 0x00 | ..... tělo zprávy =NovarStatus (60 bytů) ..... | checksum |

1.2.1.1.5Spuštění 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

V výjimkou typů 1312 a 1414 lze s přístroji komunikovat i pomocí protokolu 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.

POZOR ! Jedním příkazem lze načíst či zapsat maximálně **64** registrů. Pokud je struktura delší, je nutno ji číst či zapisovat nadvakrát.

Jsou implementovány následující funkce :

kód funkce	název funkce	aplikace
03	Read Holding Registers	čtení Config – registry 40101-40140 (adresovány 100 – 139)
04	Read Input Registers	čtení Status+EEStatus – registry 30101-30172 (adr. 100 – 171) čtení NovarStatus – registry 30201-30230 (adr. 200 – 229)
06	Preset Single Register	zápis Config – registry 40101-40150 (adr. 100 – 149) 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 | 0x1E | CRCLo | 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), číslem funkce (0x04) a počtem předávaných bytů (0x3C = 60) následuje obsah jednotlivých registrů. Mapa registrů je uvedena v následující tabulce.

Mapa registrů struktury „NovarStatus”

adresa	registr	horní byte	dolní byte
200	30201	SoftVersion H	SoftVersion L
201	30202	DeviceNo H	DeviceNo L
202	30203	DeviceType H	DeviceType L
203	30204	MTP H	MTP L
204	30205	Fr	I H
205	30206	I L	I50 H
206	30207	I50 L	Ir H
207	30208	Ir L	Ii H
208	30209	Ii L	Fi H
209	30210	Fi L	Kos
210	30211	THDU	THDI
211	30212	HarU3	HarU5
212	30213	HarU7	HarU9
213	30214	HarU11	HarU13
214	30215	HarU15	HarU17
215	30216	HarU19	HarI3
216	30217	HarI5	HarI7
217	30218	HarI9	HarI11
218	30219	HarI13	HarI15
219	30220	HarI17	HarI19
220	30221	U H	U L
221	30222	U50 H	U50 L
222	30223	CHL	DeltaI H
223	30224	DeltaI L	T
224	30225	Input	Res0
225	30226	MTN	Unom
226	30227	ActRelayState H	ActRelayState L
227	30228	Res1	Res2
228	30229	RegState	StateLEDs
229	30230	RegTime	ConfigChangeCnt

H=horní byte, L=dolní byte

Pro nastavení parametrů přístroje slouží struktura Config. Je uložena počínaje holding –registrem č. 40101(tzn. adresa 100), délka struktury je 100 bytů, tedy 50 registrů.

Mapa registrů struktury „Config”

adresa	registr	horní byte	dolní byte
100	40101	RegMode	Res0
101	40102	ReqCos-0	SwitchDelayL-0
102	40103	SwitchDelayC-0	ReqCosBandWidth-0
103	40104	Res1-0	ReqCos-1
104	40105	SwitchDelayL-1	SwitchDelayC-1
105	40106	ReqCosBandWidth-1	Res1-1
106	40107	MTP H	MTP L
107	40108	SwitchBlockDelay	UIMode
108	40109	CSRatio	Ck
109	40110	Steps	QuickSteps
110	40111	CLVal0 H	CLVal0 L
111	40112	CLVal1 H	CLVal1 L
112	40113	CLVal2 H	CLVal2 L
113	40114	CLVal3 H	CLVal3 L
114	40115	CLVal4 H	CLVal4 L
115	40116	CLVal5 H	CLVal5 L
116	40117	CLVal6 H	CLVal6 L
117	40118	CLVal7 H	CLVal7 L
118	40119	CLVal8 H	CLVal8 L
119	40120	CLVal9 H	CLVal9 L
120	40121	CLVal10 H	CLVal10 L
121	40122	CLVal11 H	CLVal11 L
122	40123	CLVal12 H	CLVal12 L
123	40124	CLVal13 H	CLVal13 L
124	40125	FixedSteps H	FixedSteps L
125	40126	FixedStepValue H	FixedStepValue L
126	40127	LCosMargin	QuickControlSpeed
127	40128	AlarmSig H	AlarmSig L
128	40129	AlarmAction H	AlarmAction L
129	40130	FixedStepsFH	MTN
130	40131	Unom	TFHLimit(Fan)
131	40132	TFHLimit(Heating)	Ulimit(Undervoltage)
132	40133	Ulimit(Overvoltage)	THDUlimit
133	40134	THDIlimit	CHLlimit
134	40135	Tlimit	SwitchNoLimit
135	40136	TCF	ScanFreq
136	40137	Res3	Res4
137	40138	DeviceAddr	RemoteBdRate
138	40139	AvePQWindowLength	Res5
139	40140	RemoteControl	ExtCosValue0
140	40141	ExtCosValue1	ExtCosValue2
141	40142	ExtCosValue3	ExtCosValue4
142	40143	Res6[0]	Res6[1]
143	40144	Res6[2]	Res6[3]
144	40145	OffsetCLVal-0 H	OffsetCLVal-0 L
145	40146	OffsetCLVal-1 H	OffsetCLVal-1 L
146	40147	OffsetMode	RemoteControlTimeout
147	40148	Res7[0]	Res7[1]

148	40149	Res7[2]	Res7[3]
149	40150	ConfigCRC H	ConfigCRC L

H=horní byte, L=dolní byte

Další detaily o stavu přístroje lze nalézt ve strukturách Status a EEStatus, uložených za sebou počínaje input –registrem č. 30101(tzn. adresa 100), délka obou struktur dohromady je 144 bytů, tedy 72 registrů.

Mapa registrů struktur „Status" a „EEStatus"

adresa	registr	horní byte	dolní byte
100	30101	HWError	OutputSwitchNo0
101	30102	OutputSwitchNo1	OutputSwitchNo2
102	30103	OutputSwitchNo3	OutputSwitchNo4
103	30104	OutputSwitchNo5	OutputSwitchNo6
104	30105	OutputSwitchNo7	OutputSwitchNo8
105	30106	OutputSwitchNo9	OutputSwitchNo10
106	30107	OutputSwitchNo11	OutputSwitchNo12
107	30108	OutputSwitchNo13	Event H
108	30109	Event L	ActRelayState H
109	30110	ActRelayState L	ReqRelayState H
110	30111	ReqRelayState L	State
111	30112	AlarmSigActive H	AlarmSigActive L
112	30113	AlarmActionActive H	AlarmActionActive L
113	30114	BadSteps H	BadSteps L
114	30115	SoftVersion H	SoftVersion L
115	30116	DeviceNo H	DeviceNo L
116	30117	DeviceType H	DeviceType L
117	30118	PrecisedSteps H	PrecisedSteps L
118	30119	MaxTHDU	MaxTHDI
119	30120	MaxCHL	MaxHar3U
120	30121	MaxHar5U	MaxHar7U
121	30122	MaxHar9U	MaxHar11U
122	30123	MaxHar13U	MaxHar15U
123	30124	MaxHar17U	MaxHar19U
124	30125	Res0	Res1
125	30126	MaxT	MinCos
126	30127	MaxAveP H	MaxAveP L
127	30128	MaxAveQ H	MaxAveQ L
128	30129	MaxAveDeltaQ H	MaxAveDeltaQ L
129	30130	AveP0 HH	AveP0 H
130	30131	AveP0 L	AveP0 LL
131	30132	AveP1 HH	AveP1 H
132	30133	AveP1 L	AveP1 LL
133	30134	AveQ0 HH	AveQ0 H
134	30135	AveQ0 L	AveQ0 LL
135	30136	AveQ1 HH	AveQ1 H
136	30137	AveQ1 L	AveQ1 LL
137	30138	AveDeltaQ HH	AveDeltaQ H
138	30139	AveDeltaQ L	AveDeltaQ0 LL
139	30140	AvePQCounter0 HH	AvePQCounter0 H
140	30141	AvePQCounter0 L	AvePQCounter0 LL
141	30142	AvePQCounter1 HH	AvePQCounter1 H

142	30143	AvePQCounter1 L	AvePQCounter1 LL
143	30144	OutputSwitchNo64_0 H	OutputSwitchNo64_0 L
144	30145	OutputSwitchNo64_1 H	OutputSwitchNo64_1 L
145	30146	OutputSwitchNo64_2 H	OutputSwitchNo64_2 L
146	30147	OutputSwitchNo64_3 H	OutputSwitchNo64_3 L
147	30148	OutputSwitchNo64_4 H	OutputSwitchNo64_4 L
148	30149	OutputSwitchNo64_5 H	OutputSwitchNo64_5 L
149	30150	OutputSwitchNo64_6 H	OutputSwitchNo64_6 L
150	30151	OutputSwitchNo64_7 H	OutputSwitchNo64_7 L
151	30152	OutputSwitchNo64_8 H	OutputSwitchNo64_8 L
152	30153	OutputSwitchNo64_9 H	OutputSwitchNo64_9 L
153	30154	OutputSwitchNo64_10 H	OutputSwitchNo64_10 L
154	30155	OutputSwitchNo64_11 H	OutputSwitchNo64_11 L
155	30156	OutputSwitchNo64_12 H	OutputSwitchNo64_12 L
156	30157	OutputSwitchNo64_13 H	OutputSwitchNo64_13 L
157	30158	OutputSwitchOnTime2H_0 H	OutputSwitchOnTime2H_0 L
158	30159	OutputSwitchOnTime2H_1 H	OutputSwitchOnTime2H_1 L
159	30160	OutputSwitchOnTime2H_2 H	OutputSwitchOnTime2H_2 L
160	30161	OutputSwitchOnTime2H_3 H	OutputSwitchOnTime2H_3 L
161	30162	OutputSwitchOnTime2H_4 H	OutputSwitchOnTime2H_4 L
162	30163	OutputSwitchOnTime2H_5 H	OutputSwitchOnTime2H_5 L
163	30164	OutputSwitchOnTime2H_6 H	OutputSwitchOnTime2H_6 L
164	30165	OutputSwitchOnTime2H_7 H	OutputSwitchOnTime2H_7 L
165	30166	OutputSwitchOnTime2H_8 H	OutputSwitchOnTime2H_8 L
166	30167	OutputSwitchOnTime2H_9 H	OutputSwitchOnTime2H_9 L
167	30168	OutputSwitchOnTime2H_10 H	OutputSwitchOnTime2H_10 L
168	30169	OutputSwitchOnTime2H_11 H	OutputSwitchOnTime2H_11 L
169	30170	OutputSwitchOnTime2H_12 H	OutputSwitchOnTime2H_12 L
170	30171	OutputSwitchOnTime2H_13 H	OutputSwitchOnTime2H_13 L
171	30172	ManualStepValue H	ManualStepValue L

1.2.3 Častý problém při použití protokolu Modbus-RTU bez parity

Dle definice protokolu Modbus-RTU se používá zásadně devitibitová délka slova, doplněná jedním startbitem a jedním stopbitem. Devátý datový bit při nastavení liché či sudé parity nese informaci o paritě.

Při nastavení přenosu „bez parity“ je sice tento bit bezvýznamný, ovšem musí v přenášených slovech fyzicky zůstat, jinak přístroj vyhodnotí chybu protokolu a přijatou zprávu nepřijme.

Pokud v komunikačním programu nelze z nějakých důvodů použít devítibitový přenos, dá se tento problém snadno obejít tak, že se v programu nastaví dva stopbity místo jednoho – první stopbit pak přístroj interpretuje jako paritní bit, protokol vyhodnotí jako korektní a přijatou zprávu příslušným způsobem zpracuje a odpoví na ni.

1.2.4 Příklad vyhodnocení načtených dat

Aktuální hodnoty měřených veličin a stav regulátoru je uložen ve struktuře "NovarStatus". Je namapována do Input-registrů od adresy 200, celkem 30 registrů. Načtení vypadá např. takto :

Request: 6.3.2013 16:10:13.64664 (+0.9063 seconds)

```
01 04 00 C8 00 1E F1 FC
```

Answer: 6.3.2013 16:10:13.66164 (+0.0156 seconds)

```
01 04 3C 00 15 FF FF 00 16 80 0A 4E 00 F5 00 8E
00 41 00 7E 00 3F 2E 04 89 06 0C 0E 06 06 00 01
00 00 D4 CF C8 A0 7A 69 65 67 5C 0A 0E 0A 19 AC
FF DA 1A 00 00 16 14 02 08 02 08 06 80 64 00 98
1B
```

1.2.4.1 Účiník (proměnná Kos)

Uložen ve spodním bajtu registru č. 209, hodnota 0x2E = 46(dek), to odpovídá hodnotě účiníku 0,46 L.

1.2.4.2 Činný proud (proměnná Ir) a jalový proud (Ii) základní harmonické složky

Pro zjištění hodnot primárních proudů je třeba znát převod PTP, jelikož přístroj předává sekundární hodnoty proudů. PTP(proměnná *MTP*) je v registru na adrese 203, v našem případě 0x800A; převod PTP je tedy A(hex)=10(dek). Nejvyšší bit je 1, čili sekundár je 5A a jelikož převod je 10, hodnota PTP tedy je 50/5A.

Hodnoty činného a jalového proudu jsou namapovány do registrů na adresách 206 až 208 - horní bajt Ir je ve spodním bajtu registru na adrese 206, tzn. 00, spodní bajt je v horním bajtu registru následujícího, tedy 0x41; hodnota Ir je tedy 0x0041= 65(dek). Jelikož proudy se předávají v jednotkách 0,25mA, proud Ir= 65 * 0,00025 =0,01625A. Toto je hodnota na sekundáru PTP, čili primární činný proud je po vynásobení převodem 0,01625 * 10 = 0,1625A.

Obdobně jalový proud, jehož předaná hodnota je 0x007E, po stejném přepočtu dá 0,315A.

1.2.4.3 Efektivní proud (proměnná I)

Namapován do registrů na adresách 204 a 205 - horní bajt je ve spodním bajtu registru na adrese 204, tzn. 00, spodní bajt je v horním bajtu registru následujícího, tedy 0xF5; hodnota je 0x00F5, po stejném přepočtu jako u *Ir* a *Ii* odpovídá hodnotě 0,6125A.

1.2.4.4 Efektivní napětí (proměnná U) a napětí základní harmonické složky (U50)

Pokud je měřicí napětí k přístroji připojeno přes měřicí transformátor napětí, je nutné nejprve zjistit převod PTN (proměnná *MTN*), jelikož přístroj předává sekundární hodnoty napětí.

Převod je v horním bajtu registru na adrese 225, tedy 0x16=22(dek), čili převod PTN je dle uvedeného kódování (typu lomená funkce) 220. V dolním bajtu je zakódované nominální napětí (sekundáru PTN), hodnota 0x14=20(dek), což odpovídá 100V. Tedy PTN je 22000/100 V.

Napětí základní harmonické složky na sekundáru PTN (proměnná *U50*) je v registru na adrese 221, tedy 0x0A19=2585, a jelikož napětí se předává vždy v jednotkách 0,1V, je napětí na sekundáru 258,5V. Po vynásobení převodem PTN to dá 258,5 * 220 = 56,87 kV.

Obdobně lze vyhodnotit i efektivní napětí (proměnná *U*), která je uloženo na adrese 220 (hodnota 0x0A0E).

1.2.4.5 Trojfázový činný výkon a jalový výkon základní harmonické složky

Hodnoty výkonů se nepředávají přímo, je nutné je spočítat z výše uvedených hodnot proudů a napětí.

Jednofázový činný výkon dostaneme vynásobením činného proudu fázovým napětím základní harmonické složky. Zda je připojené napětí sdružené či fázové, je dané proměnnou *UIMode*, která bude vysvětlena níže.

V daném příkladu předpokládáme, že se jedná o napětí sdružené, takže činný fázový výkon získáme vynásobení činného proudu fázovou hodnotou napětí, kterou získám ze sdružené hodnoty podělením odmocninou ze tří, tedy 56,87 / 1,73 = 32,87 kV. Toto vynásobeno činným proudem 0,1625 dá 5,34 kW; obdobně fázový jalový výkon je 10,35 kvar.

Pokud by se jednalo o napětí fázové, dělení 1,73 se neprovede.

Konečně třífázový činný výkon získáme vynásobením 3, čili 5,34 * 3= 16,05 kW a obdobně Pre=31,06 kvar.

Nyní ještě ke zjištění, zda měřené napětí je fázové, či sdružené. Tuto informaci obsahuje proměnná *UIMode* ve struktuře „Config“. Tu lze načíst jako Holding-registry od adresy 100 :

Request: 6.3.2013 17:40:22.11164 (+0.8750 seconds)

```
01 03 00 64 00 28 04 0B
```

Answer: 6.3.2013 17:40:22.14264 (+0.0312 seconds)

```
01 03 50 43 00 62 09 04 02 00 62 04 03 02 FF 80
0A 03 F5 00 01 0E FF 00 42 00 42 00 85 01 0A 02
15 02 15 02 15 02 15 02 15 02 15 02 15 02 15 02
15 02 15 FD F7 FD F7 7F 00 37 FF 32 FF 05 16 14
28 FB 50 6E 14 28 82 2D 64 01 FE FF FF 01 47 15
AB EE A1 DA 73
```

Proměnná *UIMode* je ve spodním bajtu registru na adrese 107, tedy v našem případě obsahuje hodnotu 0xF5, což je binárně 1111 0101. Bit 3 (číslováno od nuly) je 0, tedy připojené napětí je sdružené.

Hodnotu *UIMode* stačí obvykle načíst pouze jednou při spuštění programu a pak již číst pouze „NovarStatus“, protože tato hodnota se v průběhu činnosti regulátoru nemění. Pro úplnou „blbuvzdornost“ programu lze případně využít proměnné *ConfigChangeCnt* ve struktuře „NovarStatus“ - změna hodnoty této proměnné znamená, že obsluha změnila nastavení regulátoru a mohlo tedy dojít i ke změně hodnoty *UIMode* - pak je potřeba znova jednorázově načíst aktuální hodnotu této proměnné a podle ní provádět výpočet výkonů.

1.2.5 Příklad změny parametru – *požadovaný účiník* (proměnná *ReqCos*)

Předpokládejme, že je potřeba změnit požadovaný účiník pro tarif 1. Je umístěn ve struktuře "Config" pod jménem ReqCos[0].

Popis struktury je uveden v následující kapitole. Nejprve je uvedena "podstruktura" RegParType, kde je uvedeno i zakódování proměnné ReqCos, vlastní Config začíná níže proměnnými RegMode a Res0. Ty jsou přístupné přes holding registr na adrese 100.

V dalším registru na adrese 101 je již požadovaný účiník pro tarif 1. Hodnota zabírá pouze 1 bajt a je umístěna v MSB-bitech holding registru na adrese 101, což lze načíst takto :

Request: 11.8.2014 11:24:22.82164 (+0.9844 seconds)

```
01 03 00 65 00 01 94 15
```

Answer: 11.8.2014 11:24:22.85264 (+0.0313 seconds)

```
01 03 02 62 09 51 22
```

Načtený obsah registru je 6209 hexadecimálně. 62hex odpovídá 98dec, tedy nastavený požadovaný účiník cos1 je 0.98.
Hodnota 09 je proměnná SwitchDelayL, což je doba regulace při nedomompenzování. Hodnota 09 odpovídá, v souladu s popisem podstruktury RegParType, 3 minutám.

Přepis hodnoty lze provést například funkcí Preset Single Register takto :

Request: 11.8.2014 11:32:43.49664 (+9.3906 seconds)

```
01 06 00 65 64 09 73 13
```

Answer: 11.8.2014 11:32:43.51264 (+0.0156 seconds)

```
01 06 00 65 64 09 73 13
```

Zapsána bylo do registru na adrese 101 (=0065hex), zapsaná hodnota 6409hex. 64hex=100dec, tedy nyní je požadovaný účiník 1.00.
Jelikož nejmenší zapisovatelná hodnota je jeden šestnáctibitový registr, musí se při změně účiníku vždy nejprve načíst jeho aktuální stav, pak upravit horní bajt dle potřeby, a teprve pak oba bajty zapsat.
Pokud provedeme znova načtení stavu, je již hodnota upravena :

Request: 11.8.2014 11:33:13.65664 (+29.6406 seconds)

```
01 03 00 65 00 01 94 15
```

Answer: 11.8.2014 11:33:13.68764 (+0.0313 seconds)

```
01 03 02 64 09 52 82
```

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   HWError;            /* 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 */
                                            /* lisi se od aktualniho, pokud se ceka na RelayOffTimeCnt */
                uchar   State;              /* 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...1=alarm od podproudu */
                                            /* bit1...1=alarm od nadproudu */
                                            /* bit2...1=alarm od ztraty napet.signalu */
                                            /* bit3...1=alarm od podpeti */
                                            /* bit4...1=alarm od prepeti */
                                            /* bit5...1=alarm od prekroceni meze THDI */
                                            /* bit6...1=alarm od prekroceni meze THDU */
                                            /* bit7...1=alarm od prekroceni meze CHL */
                                            /* bit8...1=alarm od chyby kompenzace */
                                            /* bit9...1=alarm pri zpetnem nap. */
                                            /* bit10..1=alarm od prekroceni meze poctu sepnuti */
                                            /* bit11..1=alarm pri vypadku reg. kondenzatoru */
                                            /* bit12..1=alarm od prehrati */
                                            /* bit13..1=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;           /* vyrobni cislo */
                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, kodovaní 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 vnitrni potrebu pristroje)
                float AveQ[2];          // klouz. prumer Q  (pro vnitrni potrebu pristroje)
                float AveDeltaQ;        // klouz. prumer DeltaQ  (pro vnitrni potrebu pristroje)
                ulong AvePQCounter[2];  // citac delky klouz. okna  (pro vnitrni 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 ruzne od 00 a FF */
                        /* obsahuje cislo specialniho 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*/
            int       Fi;     /* uhel Ir a Ij ve stupnich */
            char      Kos;    /* účiník 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[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
            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       DeltaIi; // chybejici jalovy proud, rozmer stejny jako I
            char      T;      // teplota ve stupnich C
            uchar     Input;  // bit 0:   ....0, vstup 2 tarifu nesepnut
                        //        ....1, vstup 2 tarifu sepnut
            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 */
                    // STATEIDDLE               0x09    /* nelze regulovat- nejsou merena data */
                    // STATEMANUAL              0x0F    /* nelze regulovat- je v manualu */
                    /* horni nibble-maska 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   ConfigChangeCnt;
        } NSType;   /* NovarStatus-60 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!!!- ceka 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...-individulani nastaveni CLVal */
                                /*          pri teto hodnote zustanou hodnoty CLVal */
                                /*          i pri zavolani funkce SetCLValues */
                                /*          nedotceny(lze je indiv. editovat) */
                                /*          Mozny postup: zadat hodnotu > 0 (predpripravit) */
                                /*          nasledne v editaci CLVal upravit -> pritom */
                                /*          se CSRatio autom. nastavi na 0x00! */
                                /*     -rovnez uspesny prubeh autom. regulace */
                                /* 0xFF-neuspesny prubeh autom. rozpoznani Ck! */
                                /* nastavi CSRatio na 0xFF ! */
                                /* 1...1:1:1:1:1 */
                                /* 2...1:1:2:2:2 */
                                /* 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 az 2A po 0,01A */
                                /* pokud je CSRatio 0, nezobrazuje se */
uchar   Steps;                  /* pocet pouzitych indukt. a kapacitnich stupnu */
                                /* nibble 3-0...CSteps=pocet kap. stupnu */
uchar   QuickSteps;             /* pouze pro Novar-1312: pocet stupnu pro rychlou regul;aci */
                                /* pro ostatni typy bez vyznamu  */
int     CLVal[14];              /* hodnoty stupnu */
                                /* obsahuje hodnotu im. slozky proudu 50Hz */
                                /* v jednotkach 0,25mA, kondik kladny */
                                /* maximalni rozsah je +/-32000, tj. +/-8A */
uint    FixedSteps;             /* bitova mapa vystupu, ktere jsou */
                                /* nastavene jako pevne; 0=pevny vystup */
uint    FixedStepValue; /* bitova mapa hodnot pevnych */
                                /* 0=vystup sepnut, 1=rozepnut */
                                /* pevne nastavene vystupy jsou vyrazeny */
                                /* z regulace, ale rozlozeni CSteps a LSteps */
                                /* zustava-nic se neposouva, vsechny */
                                /* vystupy, ktere 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-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 pevnych stupnu dvou poslednich 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...dtto 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.THD
                                    // 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   Res1;           // rezerva


                        // následujících 20 byte pouze pro verze firmware 1.3 a vyšší :

                uchar   RemoteControl;  // pro provedení NRC, pro přístup přes Modbus nevýznamné
                char    ExtCosValue[5];         // dtto
                uchar   Res6[4];                // rezerva

                int     OffsetCLVal[2];         // hodnota offset. výkonu, pro každý tarif zvlášť
                                                // formát shodný s CLVal
                uchar   OffsetMode;             // bit0 = 1 ... normalni regulace bez offsetu
                                                //        0 ... regulace s offsetem
                uchar   RemoteControlTimeout;   // pro provedení NRC, pro přístup přes Modbus
nevýznamné
                uchar   Res7[4];                // rezerva


                uint    ConfigCRC;      /* CRC Configu. Neni nutno nastavovat, obsah libovolny */
        } CType;                /* Config - 100 bytu (80 bytu pro verzi firmware 1.2 a nižší) */
//=============================================================================
```

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 vyzadovano 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  */
//=========================================================================
```