Aplikační list AN OM2006 – 005

Popis protokol Modbus v přístrojích ORBIT MERRET

Ing. Jiří Antoš 21. 1. 2021

Úvod

V tomto dokumentu je popsána Implementace protokolu MODBUS. Na začátku je třeba říci, že MODBUS RTU je třetí protokol instalovaný v přístrojích ORBIT MERRET. Je určen pro komunikace s řídicím nebo nadřazeným systémem. ORBIT MERRET poskytuje software s názvem OM LINK pro snadné ovládání přístroje. Tento software umožňuje nastavovat přístroje bez USB pouze s protokolem ASCII.

Komunikační parametry – low-level protokol

Přenosové médium:	RS485, kroucený pár, maximální délka 1200 m (4000 stop) maximální počet zařízení na lince je 32 pro více zařízení je nutný opakovač
Přenosová rychlost:	600, 1 200, 2 400, 4 800, 9 600, 19 200, 38 400, 57 600, 115 200, 230 400 Baud
Start-bity:	1
Datové bity:	8
Start-bity:	1
Parita:	žádná

Prodleva mezi telegramy: > 3,5 znaky *)

Prodleva mezi symboly: Není vyhodnocována

*) Komunikace je ukončena za předpokladu, že během doby odpovídající 3 1/2 znakům nedorazí žádná data. Tento interval je definován s nejistotou ±250μs. MODBUS má standardní rychlosti do 19 200 Baud. Pro vyšší rychlosti je nutné počítat s touto nejistotou - například 115 200 Baud -> 500±250 μs, 230 400 Baud -> 250 ±250 μs.

Formát telegramu MODBUS RTU

<ADR> <FNC> <DATA> <CRC-L> <CRC-H>

Pole <ADR> je 1 byte adresa v rozsahu od 1 do 247. Broadcast adresa 0 není v přístojích ORBIT MERRET podporována.

Pole <FNC> je 1 byte příkaz, který určuje obsah pole <DATA>. Podrobný popis funkcí a jejich údajů je níže.

Pole <CRC> jsou 2 byty kontrolního součtu CRC, který je popsán v dodatku A.

Podporované formáty dat, podle jejich délky

1 byte	Označení	Rozsah	Umístění v telegramu
index seznamu	List	0 .. 255	<0x00><Bajt>
beznaménkový znak	Char	0 .. 255	<0x00><Bajt>

2 byty	Označení	Rozsah	Umístění v telegramu
beznaménkový celé číslo	Int	0 .. 65535	<Vysoká><Nízká>
2-znakový popis	Desc	' ' .. '', 0x20 .. 0x7F	<První><Druhý>

3 byty	Označení	Rozsah	Umístění v telegramu
3-znakový popis	Desc3	' ' .. '', 0x20 .. 0x7F	<0x00><První><Druhý><Třetí>

4 délky	Označení	Rozsah	Umístění v telegramu
dlouhé beznam. celé číslo	Long	0 .. 4 294 967 295	<MSB><Třetí><Druhý><LSB>
Číslo s plovoucí tečkou	Float	± 6,80564693277E+38	<SE><EM><MM><MM>

Popis čísla s plovoucí tečkou

první byte	<SE>	SEEE EEE	S...znaménko (1(0)/-1(1)); E...Exponent (-127(0x00)...0(0x7F)...128(0xFF))
druhý bajt	<EM>	EMMM MMMM	M...mantisa (1.0...2.0), nejvyšší bit mantisy je vždy 1 a je překryt nejnižším bitem exponentu
3., 4. bajt	<MM><MM>		zbytek bitů mantisy např.: 0x3F80 0000 = Z*2^E*M = 1*2^(0)*1= 1

ORBIT MERRET, spol. s r.o.
Vodňanská 675/30, 198 00 Praha 9, Česky Republika
☎ +420 281 040 200 🖷 +420 281 040 299 orbit@merret.cz

PODROBNÝ POPIS PŘÍKAZŮ

Příkaz 0x01 - Čtení cívek, stavu relé
Požadavek: <ADR><0x01><0x00><0x00><0x00><0x08><CRC Lo><CRC Hi>
Odpověď: <ADR><0x01><0x01><RR><CRC Lo><CRC Hi>
<RR> binárně kódovaný stav relé. Nenainstalované relé vrátí 0
0x80 ... relé 8
0x40 ... relé 7
0x20 ... relé 6
0x10 ... relé 5
0x08 ... relé 4
0x04 ... relé 3
0x02 ... relé 2
0x01 ... relé 1

Příkaz 0x02 - Čtení externích vstupů
Požadavek: <ADR><0x02><0x00><0x00><0x00><0x08><CRC Lo><CRC Hi>
Odpověď: <ADR><0x02><0x01><Vstupy><CRC Lo><CRC Hi>
<Vstupy> binárně kódovaný stav externích vstupů. Nenainstalovaný vstup vrátí 0
0x80 ... vstup 8
0x40 ... vstup 7
0x20 ... vstup 6
0x10 ... vstup 5
0x08 ... vstup 4
0x04 ... vstup 3
0x02 ... vstup 2
0x01 ... vstup 1

Příkaz 0x03 - Čtení položek menu
Formát příkazu závisí na formátu dat položky menu.

Formát 1 nebo 2 byty
Požadavek: <ADR><0x03><AA High><AA Low><0x00><0x01><CRC Lo><CRC Hi>
Odpověď: <ADR><0x03><Data Hi><Data Lo><CRC Lo><CRC Hi>

Formát 3 nebo 4 byty
Požadavek: <ADR><0x03><AA High><AA Low><0x00><0x02><CRC Lo><CRC Hi>
Odpověď: <ADR><0x03><MSB dat><data><data><LSB dat><CRC Lo><CrC Hi>

Upozornění: Čtení více položek nabídky není podporováno.

Příkaz 0x04 - Čtení naměřených nebo vyhodnocených hodnot
Tato data jsou vždy ve formátu s plovoucí čárkou.
Požadavek: <ADR><0x04><AA High><AA Low><0x00><0x02><CRC Lo><CRC Hi>
Odpověď: <ADR><0x04><MSB dat><data><data><LSB dat><CRC Lo><CrC Hi>

Upozornění: Čtení více hodnot není podporováno.

Příkaz 0x05 - Provést akci
Požadavek: <ADR><0x05><AA High><AA Low><0xFF><0x00><CRC Lo><CRC Hi>
Odpověď: <ADR><0x05><AA High><AA Low><0xFF><0x00><CRC Lo><CRC Hi>

Upozornění: Relé jsou ovládány přístrojem. Tento příkaz spustit akci. Více akcí není podporováno.

Příkaz 0x06 - Nastavení položek menu
Tento příkaz je určen pro obsluhu položek menu s délkou dat 1 nebo 2 byty.

Požadavek: <ADR><0x06><AA High><AA Low><Data Hi><Data Lo><CRC Lo><CRC Hi>
Odpověď: <ADR><0x06><AA High><AA Low><Data Hi><Data Lo><CRC Lo><CRC Hi>

Upozornění: Nastavení více položek nabídky není podporováno.

Příkaz 0x10 - Nastavení položek nabídky
Tento příkaz je určen pro obsluhu položek menu s délkou dat 3 nebo 4 byty.

Požadavek: <ADR><0x10><AA High><AA Low><0x00><0x02>
<0x04><MSB dat><data><data><LSB dat><CRC Lo><CRC Hi>
Odpověď: <ADR><0x10><AA High><AA Low><0x00><0x02><CRC Lo><CRC Hi>

Upozornění: Nastavení více položek nabídky není podporováno.

Příkaz 0x11 - Identifikace přístroje na sběrnici
Tento příkaz je určen pro zjištění identifikace připojeného přístroje.

Požadavek: <ADR><0x11><CRC Lo><CRC Hi>
Odpověď: <ADR><0x11><0x1C><0xFF><0xFF>

<data>.. <data>	12 znaků názvu nástroje,	např.	"OM 402UNI "
<'-><data> .. <data>	6 znaků verze firmwaru,	např.	"62-001"
<'-><data> .. <data>	6 znaků režimu měření,	např.	" 40 V"
<CRC Lo><CRC Hi>			

Podporované adresy v poli <AA High><AA Low>
Seznam těchto adresy je možné získat na vyžádání na orbit@merret.cz nebo na našich webových stránkách. Adresa by měla být použita bez první z 5 číslic, například 40012 by měla být použita jako <ADR><0x06>**<0x00><0x0C>**<0x02>...

Upozornění: V některých systémech je potřeba zadávat adresy o 1 vyšší.

Odpověď při chybě
V případě nesprávné adresy nebo CRC není žádná odpověď.
V případě chyby je vrácena chybová odpověď

<ADR> <FNC & 0x80> 01 <CRC Lo> <CRC Hi>	přijat nepodporovaný příkaz
<ADR> <FNC & 0x80> 02 <CRC Lo> <CRC Hi>	nesprávná adresa registru
<ADR> <FNC & 0x80> 03 <CRC Lo> <CRC Hi>	nesprávný počet cívek nebo registrů
<ADR> <FNC & 0x80> 04 <CRC Lo> <CRC Hi>	příkaz nemohl být vykonán. Pravděpodobně jsou chybná data nebo je dynamicky zakázána položka nabídky.

Dodatek A - GENEROVÁNÍ CRC

Pole kontrolního součtu (CRC) jsou dva byty, které obsahují 16-bitovou binární hodnotu. Hodnota CRC je vypočtena odesílajícím zařízením, které ji připojí ke zprávě. Přijímající zařízení přepočítá CRC během příjmu zprávy a porovná vypočtenou hodnotu s hodnotou, kterou obdrželo v poli CRC. Pokud se tyto dvě hodnoty nerovnají, dojdek chybě. Výpočet CRC je spuštěn přednastavením 16bitové hodnoty na všechny 1. Poté se postupně připočtou všechny byty zprávy. Pro generování CRC se použíjí pouze datové bity. Start a stop bity a paritní bit se neuvažují pro výpočet CRC. Při generování CRC je každý 8bitový znak XOR—ován s obsahem registru. Poté se výsledek posune ve směru nejméně významného bitu (LSB) s nulou vyplněnou do nejvýznamnější pozice bitů (MSB). LSB se extrahuje a pokud LSB byl 1, registr je pak XOR-ován s pevnou přednastavenou hodnotou. Pokud LSB byl 0, XOR nebude proveden. Tento proces se opakuje, dokud není provedeno osm posunů. Po posledním (osmém) posunu se pokračuje dalším znakem stejným způsobem. Konečný obsah registru, po použití všech znaků zprávy, je hodnota CRC.

Postup pro generování CRC je:
1, Nastavit 16-bit registr na 0xFFFF (všechny 1 je). Nazvěme jej registrem CRC.
2. XOR na první 8bitový bajt zprávy s nižším bytem registru CRC, výsledek se vloží do registru CRC.
3. Posunout registr CRC o jeden bit doprava (směrem k LSB), vyplnit MSB nulou. Test LSB
4. (Pokud byl LSB 0): Opakujte krok 3 (další posun).
   (Pokud LSB byl 1): XOR registru CRC s hodnotou 0xA001 (1010 0000 0000 0001).
5. Opakujte kroky 3 a 4, dokud nebude provedeno 8 posunů.
6. Opakujte kroky 2 až 5 pro další byte zprávy. Pokračujte v tom, dokud nebudou zpracovány všechny bajty.
7. Konečným obsahem registru CRC je hodnota CRC.
8. Pro připojení CRC do zprávy, je potřeba horní a dolní bajty registru CRC přehodit, jak je popsáno níže.

Připojení CRC ke zprávě
16-bitové CRC (dva byty) je přenášeno v pořadí nižší byte, vyšší byte.

Příklad
Příklad funkce v jazyce C provádějící generování CRC je uveden na následujících stránkách. Všechny možné hodnoty CRC jsou přednastaveny do dvou polí, které lze jednoduše indexovat. Jedno pole obsahuje všechny možné hodnoty CRC 256 pro vyšší byte registru CRC a druhé pole pro nižší byte. Indexování CRC tímto způsobem poskytuje rychlejší provedení než by bylo dosaženo výpočtem nové hodnoty CRC s každým novým znakem ze zprávy.
Poznámka: Tato funkce provádí prohození vysokých nebo nízkých bajtů CRC interně. Bajty jsou již zaměněny v hodnotě CRC, která je vrácena z funkce. Proto hodnota CRC vrácená z funkce může být přímo připojena ke zprávě pro přenos.

Funkce má dva argumenty:
unsigned char *puchMsg ;
Ukazatel na vyrovnávací paměť zprávy obsahující binární data, která mají být použita pro generování CRC
unsigned short usDataLen ;
Počet bytů ve vyrovnávací paměti zprávy. Funkce vrátí CRC jako unsigned short.

Popis protokol Modbus v přístrojích ORBIT MERRET

Ing. Jiří Antoš 21. 1. 2021

Funkce generování CRC

```
unsigned short CRC16(puchMsg, usDataLen)
// unsigned char *puchMsg;                              /* zpráva pro výpočet CRC */
// unsigned short usDataLen;                            /* počet bajtů ve zprávě */
{
       unsigned char uchCRCHi = 0xFF;                   /* vyšší byte CRC s inicializací*/
       unsigned char uchCRCLo = 0xFF;                   /* nižší byte CRC s inicializací */
       unsigned char uIndex;                            /* index vyhledávací tabulky CRC */

       while (usDataLen--)                              /* projdeme vyrovnávací paměť zpráv */
       {
              uIndex =  uchCRCHi ^ *puchMsg++ ;         /* výpočet CRC */
              uchCRCHi = uchCRCLo ^  auchCRCHi[uIndex];
              uchCRCLo = auchCRCLo[uIndex];
       }
       return (uchCRCHi  << 8 | uchCRCLo);
}
```

/* Tabulka hodnot CRC pro vyšší byte */

```
static unsigned char auchCRCHi[] = {
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40
} ;
```

/* Tabulka hodnot CRC pro nižší bajt nízkého pořadí */

```
statický znak auchCRCLo[] = {
0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7, 0x05, 0xC5, 0xC4,
0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,
0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F, 0xDD,
0x1D, 0x1C, 0xDC, 0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32, 0x36, 0xF6, 0xF7,
0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,
0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA, 0xEE,
0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1, 0x63, 0xA3, 0xA2,
0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,
0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8, 0xB9, 0x79, 0xBB,
0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0, 0x50, 0x90, 0x91,
0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58, 0x98, 0x88,
0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83, 0x41, 0x81, 0x80,
0x40
} ;
```

ORBIT MERRET, spol. s r.o.
Vodňanská 675/30, 198 00 Praha 9, Česky Republika
☎ +420 281 040 200 🖷 +420 281 040 299 orbit@merret.cz