RACOM
RADIO DATA NETWORKS

MTF protocol for MORSE

version 10.0.62.0
6/29/2015

1. Introduction

MTF protocol (Morse Technology Format) is used to securely transfer data between the points with the technological unit SEP/ADIO. MTF format can also use the MODBUS protocol using function translation by ART table, see the MODBUS description.

2. Data Format

The structure of MTF packet, which is sent from the interface to MORSE network:

```
| header/32 | data_part | chksum/16 |
```

The packet consists of a header, a number of data blocks and checksum. Blocks of `data_part` can be a different number. Format of each item is determined by the `command` and processed one after the other.

Header:

```
|format/8|err/8|reqNo/8|respNo/8|
```

`format/8` beginning of the packet 0x01

`err/8` error message

`reqNo/8` request number

`respNo/8` response number

Data blocks:

```
|type/8|ft/1|cmd/3|size/4|cnt/4|offset/12| data |
```

A short frame is common in devices SEP/ADIO. Long framework is in development, its format is given at the end of this section.

`type/8` Distinguishes between data blocks according to content and defines the format of `| data |` part.

- 00 - reserved

- 01 - digi input - one block of data contains information on up to 16 inputs, each input uses 1-bit from words `mask`, `status` and `value`, while the value is the actual value of input

```
|mask/16|status/16|value/16|
   mask=0  status=0 ... invalid data
   mask=0  status=1 ... undefined data (so-called third state)
   mask=1  status=0 ... bad data, the size does not match
   mask=1  status=1 ... data OK
   value            ... digital input value
```

- 02 - analog input - one block of data contains information about an input (number of blocks is determined by parameter `cnt`

```
|invalid/1|overrun/1|value/14|
   invalid=1  ... Error description
   invalid=0  ... data OK
   overrun=0  ... value is in the interval (0)4-20mA
   overrun=1  ... value/14 hodnota is out of interval  4-20mA
                        (0x4000 <4mA, 0x4FFF > 20mA)
   value/14   ... analog input value
```

- 03 - calib - pro ADIO/SEP

```
|k/16|q/16|
```

- 04 - prodident - identification of the product ADIO/SEP

```
|typ/8|res/8|hw_ver/16|res/16|prod_num/16|res/16|date/32|sign/16|
```

- 05 - holding registers

```
Word, read/write
```

- 06 - digi output

```
|mask/16|status/16|value/16|
   mask=0  status=0 ...invalid data
   mask=0  status=1 ...undefined data (so-called third state)
   mask=1  status=0 ...bad data, the size does not match
   mask=1  status=1 ...data OK
   value            ... digital output value
```

- 07 - analog output

```
|invalid/1|res/1|value/14|
   invalid=1  ...value/14 Error description
   invalid=0  ...value/14 data OK
   value/14   ...analog output value
```

- 08 - counters

```
|invalid/1|res/1|value/30|
   invalid=1  ...value/30 Error description
   invalid=0  ...value/30 data OK
```


- 09 - unused

`ft/1` framework distinguishing, 1 - short, 0 - long

`cmd/3` command
- 0 - write req - write request
- 1 - read req - read request
- 2 - write resp - response to a write request
- 3 - read resp - response to a read request
- 4 - spontaneous data (used in a spontaneous mode SEP/ADIO)
- 5 - spontaneous alarm

`size/4` size of one block of data in a word, example:
- digital input, size = 3 word
- analog input, size = 1 word

`cnt/4` number of data blocks, example for ADIO:
- digital input, cnt = 1 block with 16 inputs, only the inputs 0 and 1 used
- analog input, cnt = 2 blocks, one for each analog input with max 14 bits

`offset/12` adresa první přenášené digitální nebo analogový kanál, příklad pro SEP:
- reading a set of digital inputs 0 to 7 -> offset = 0, cnt = 1
- reading analog inputs 0 a 1 -> offset = 0, cnt = 2
- reading analog inputs 5, 6, 7 -> offset = 5, cnt = 3

`data` data - data length (word) is determined by the product `size×cnt`

Checksum:

`|chksum/16|`

The format of short and long MTF frame

A short frame is used in equipment SEP/ADIO:

```
|typ/8|ft/1|cmd/3|size/4|cnt/4|offset/12| data |
       ft=1 - short frame
```

Long frame:

```
|typ/8|ft/1|res/7|cmd/3|res/1|size/4|cnt/8|offset/16| data |
       ft=0 - long frame
```

Example of short frames generated by ADIO module, broken down into data blocks:

```
0100 C300                format  req No,resp No
01C3 1000 0003 0003 0000  Dinp,spont,3word  1×data,000offset  mask  status  state=00
06C3 1000 0003 0003 0000  Dout,spont,3word  1×data,000offset  mask  status  state=00
02C1 2000 0C82 0001      Ainp,spont,1word  2×data,000offset  OK,stav0xC82  OK,state=0x001
```

```
07C1 2000 0C88 0000       Aout,spont,1word  2×data,000offset  OK,stav0xC88  OK,state=0x000
AFE1                       chksum
```

3. Examples

3.1. Monitoring spontaneous MTF packet on ADIO interface when changing the AI and AO.

The first packet after switching on station. The inputs are not connected and the outputs are not set.

```
17:22:12.017|                 |00000021 00000012|G00I   OUT   42||8A 5user
0100 C202 01C3 1000 0003 0003 0000 06C3 1000 0003 0003 0000 02C1 2000 0008
0001 07C1 2000 0000 0000 C9E1
```

Setting Aout0 using the test menu to 15.67 mA, the output is connected to the input Ain0

```
A0u15670
```

Spontaneous packet informing about the status change

```
17:22:39.617|                 |00000021 00000012|G00I   OUT   42||8A 6user
0100 C300 01C3 1000 0003 0003 0000 06C3 1000 0003 0003 0000 02C1 2000 0C82
0001 07C1 2000 0C88 0000 AFE1

01 00 C3 00 +
01 - format/8
   00 - error/8
      C3 - MTF request number/8
         00 - MTF response number/8

data part 1
+ 01 C3 10 00 00 03 00 03 00 00 +
  01 - typ/8 (01 = digital inputs)
     C3 - 1100 0011
          1xxx xxxx - ft/1 = 1 - short frame
          x100 xxxx - cmd/3 = 4 - spontaneous data
          xxxx 0011 - size/4 = 3 - data block size (words)
        10 00 - 0001 0000 0000 0000
                0001 xxxx xxxx xxxx - cnt/4 = 1 number of data blocks
                xxxx 0000 0000 0000 - offset/12 = 0 starting from 0
               00 03 - mask/16
                    00 03 - status/16
                          00 00 - status of digital inputs/16
data part 2
+ 06 C3 10 00 00 03 00 03 00 00 +
  06 - typ/8 (06 = digital outputs)
     C3 - 1100 0011
          1xxx xxxx - ft/1 = 1 - short frame
          x100 xxxx - cmd/3 = 4 - spontaneous data
          xxxx 0011 - size/4 = 3 - data block size (words)
        10 00 - 0001 0000 0000 0000
                0001 xxxx xxxx xxxx - cnt/4 = 1 number of data blocks
```



```
                xxxx 0000 0000 0000 - offset/12 = 0 starting from 0
               00 03 - mask/16
                    00 03 - status/16
                          00 00 - status of digital outputs/16

data part 3
+ 02 C1 20 00 0C 82 00 01 +
  02 - typ/8 (02 = analog inputs)
     C1 - 1100 0001
          1xxx xxxx - ft/1 = 1 - short frame
          x100 xxxx - cmd/3 = 4 - spontaneous data
          xxxx 0001 - size/4 = 1 - data block size (words)
        20 00 - 0010 0000 0000 0000
                0010 xxxx xxxx xxxx - cnt/4 = 2 number of data blocks
                xxxx 0000 0000 0000 - offset/12 = 0 starting from 0
              0C 82 - 0000 1100 1000 0010
                      0xxx xxxx xxxx xxxx - valid data/1
                      x0xx xxxx xxxx xxxx - value in the range/1
                      xx00 1100 1000 0010 - measured value/14
                    00 01 - 0000 0000 0000 0001
                            0xxx xxxx xxxx xxxx - data OK/1
                            x0xx xxxx xxxx xxxx - value in the range/1
                            xx00 0000 0000 0001 - measured value/14

data part 4 + chksum
+ 07 C1 20 00 0C 88 00 00 + AFE1
  07 - typ/8 (07 = analog outputs)
     C1 - 1100 0001
          1xxx xxxx - ft/1 = 1 - short frame
          x100 xxxx - cmd/3 = 4 - spontaneous data
          xxxx 0001 - size/4 = 1 - data block size (words)
        20 00 - 0010 0000 0000 0000
                0010 xxxx xxxx xxxx - cnt/4 = 2 number of data blocks
                xxxx 0000 0000 0000 - offset/12 = 0 starting from 0
              0C 88 - 0000 1100 1000 1000
                      0xxx xxxx xxxx xxxx - valid data/1
                      x0xx xxxx xxxx xxxx - reserve/1
                      xx00 1100 1000 1000 - set value/14
                    00 00 -  --//--
```

Spontaneous packets with a period set in the menu `GPe0pd t(i)me:60s`

```
17:23:12.017|                |00000021 00000012|G00I   OUT   42||8A 7user
0100 C400 01C3 1000 0003 0003 0000 06C3 1000 0003 0003 0000 02C1 2000 0C82
0001 07C1 2000 0C88 0000 AEE1

17:24:12.017|                |00000021 00000012|G00I   OUT   42||8A 0user
0100 C500 01C3 1000 0003 0003 0000 06C3 1000 0003 0003 0000 02C1 2000 0C82
0001 07C1 2000 0C88 0000 ADE1
```

3.2. Monitoring of serial link and interface with SEP in a MTF format.

Communication MR - SEP on SCC is in the Modbus format:

```
09:19:50.816 tx     10 | S00
011E 000F A00F A000 41C4
09:19:50.877 rx;i   29 | S00
011E 1800 FF00 C72C 7A00 5E00 1C00 6500 7500 0E00 2100 620F 9B0F 9ED4 9D
09:19:52.817 tx      8 | S00
0104 0010 0010 F003
09:19:52.867 rx;i   37 | S00
0104 2000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0093 79
```

CNI interface continues in MTF format:

```
09:19:54.001|                 |69609074 00000075|S00I   OUT   54||8A 1user
0100 0107 01C3 1000 00FF 00FF 00FF 06C3 1000 00FF 00FF 0000 02C1 8000 4000
4000 4000 4000 4000 4000 4000 4000 07C1 2000 0000 0000 25F6
```

A packet with details for each part:

```
| header | data_part 1 | data_part 2 | data_part 3 | data_part 4 | chksum |

0100 0107
01C3 1000 00FF 00FF 00FF
06C3 1000 00FF 00FF 0000
02C1 8000 4000 4000 4000 4000 4000 4000 4000 4000
07C1 2000 0000 0000
25F6

header
01 00 01 07 +
01 - format/8
   00 - error/8
      01 - MTF request number/8
         07 - MTF response number/8

data part 1
+ 01 C3 10 00 00 FF 00 FF 00 FF +
  01 - typ/8 (01 = digital inputs)
     C3 - 1100 0011
          1xxx xxxx - ft/1 = 1 - short frame
          x100 xxxx - cmd/3 = 4 - spontaneous data
          xxxx 0011 - size/4 = 3 - data block size (words)
        10 00 - 0001 0000 0000 0000
                0001 xxxx xxxx xxxx - cnt/4 = 1 number of data blocks
                xxxx 0000 0000 0000 - offset/12 = 0 starting from 0
               00 FF - mask/16
                    00 FF - status/16
                          00 FF - status of digital inputs/16
```



```
data part 2
+ 06 C3 10 00 00 FF 00 FF 00 00 +
  06 - typ/8 (06 = digital outputs)
     C3 - 1100 0011
          1xxx xxxx - ft/1 = 1 - short frame
          x100 xxxx - cmd/3 = 4 - spontaneous data
          xxxx 0011 - size/4 = 3 - data block size (words)
        10 00 - 0001 0000 0000 0000
                0001 xxxx xxxx xxxx - cnt/4 = 1 number of data blocks
                xxxx 0000 0000 0000 - offset/12 = 0 starting from 0
              00 FF - mask/16
                   00 FF - status/16
                         00 00 - status of digital outputs/16

data part 3
+ 02 C1 80 00 40 00 40 00 40 00 40 00 40 00 40 00 40 00 40 00 +
  02 - typ/8 (02 = analog inputs)
     C1 - 1100 0001
          1xxx xxxx - ft/1 = 1 - short frame
          x100 xxxx - cmd/3 = 4 - spontaneous data
          xxxx 0001 - size/4 = 1 - data block size (words)
        80 00 - 1000 0000 0000 0000
                1000 xxxx xxxx xxxx - cnt/4 = 8 number of data blocks
                xxxx 0000 0000 0000 - offset/12 = 0 starting from 0
             40 00 - 0100 0000 0000 0000
                     0xxx xxxx xxxx xxxx - valid data/1
                     x1xx xxxx xxxx xxxx - measured value out of range/1
                     xx00 0000 0000 0000 - measured value/14
                  40 00 -  --//--
                        40 00 -  --//--
                              40 00 -  --//--
                                    40 00 -  --//--
                                          40 00 -  --//--
                                                40 00 -  --//--
                                                      40 00 -  --//--

data part 4 + chksum
+ 07 C1 20 00 00 00 00 00 + 25F6
  07 - typ/8 (07 = analog outputs)
     C1 - 1100 0001
          1xxx xxxx - ft/1 = 1 - short frame
          x100 xxxx - cmd/3 = 4 - spontaneous data
          xxxx 0001 - size/4 = 1 - data block size (words)
        20 00 - 0010 0000 0000 0000
                0010 xxxx xxxx xxxx - cnt/4 = 2 number of data blocks
                xxxx 0000 0000 0000 - offset/12 = 0 starting from 0
             00 00 - 0000 0000 0000 0000
                     0xxx xxxx xxxx xxxx - valid data/1
                     x0xx xxxx xxxx xxxx - reserve/1
                     xx00 0000 0000 0000 - set value/14
                  00 00 -  --//--
```

3.3. Reading digital values with offset 0 and analog values with offsets 0,1,2,7, long frame.

```
| header | data_part 1 | data_part 2 | data_part 3 | chksum |

packet header
01 00 00 00 +
01 - format/8
   00 - error/8
      00 - MTF request number/8
         00 - MTF response number/8

data_part 1
+ 01 00 23 01 00 00 + 000F 000F 0000 +
  01 - typ (01 = digital inputs)
     00 - 0000 0000
          0xxx xxxx - ft/1 = 0 - long frame
          x000 0000 - res/7 - reserve
        23 - 0010 0011
             001x xxxx - cmd/3 = 1 - request for reading
             xxx0 xxxx - res/1 - reserve
             xxxx 0011 - size/4 = 3 - data block size (words)
           01 - cnt/8 = 1 - number of data blocks
              00 00 - offset/16 = 0 starting from 0
                      000F 000F 0000 - mask, status, value

data_part 2
+ 02 00 21 03 00 00 + 00AA 00BB 00CC +
  02 - typ (02 = analog inputs)
     00 - 0000 0000
          0xxx xxxx - ft/1 = 0 - long frame
          x000 0000 - res/7 - reserva
        21 - 0010 0001
             001x xxxx - cmd/3 = 1 - request for reading
             xxx0 xxxx - res/1 - reserva
             xxxx 0001 - size/4 = 1 - data block size (words)
           03 - cnt/8 = 3 - number of data blocks
              00 00 - offset/16 = 0 starting from 0, carries words 0,1,2
                      00AA 00BB 00CC - Ainp 0,1,2

data_part 3 + chksum
+ 02 00 21 01 00 07 + 00DD + chksum
  02 - typ (02 = analog inputs)
     00 - 0000 0000
          0xxx xxxx - ft/1 = 0 - long frame
          x000 0000 - res/7 - reserve
        21 - 0010 0001
             001x xxxx - cmd/3 = 1 - request for reading
             xxx0 xxxx - res/1 - reserve
             xxxx 0001 - size/4 = 1 - data block size (words)
           01 - cnt/8 = 1 - number of data blocks
```


00 07 - offset/16 = 7 starting from 7, carries word 7
00DD - Ainp 7

3.4. Writing binary outputs 2,4,7 with offset 0 and analog outputs with offset 2.3 and reading the analog input offset of 5, a long frame.

```
| header | data_part 1 | data_part 2 | data_part 3 | chksum |

packet header
01 00 00 00 +
01 - format/8
   00 - error/8
      00 - MTF request number/8
         00 - MTF response number/8

data_part 1
+ 06 00 03 01 00 00 00 94 00 94 AA AA +
  06 - typ (06 = digital outputs)
     00 - 0000 0000
          0xxx xxxx - ft/1 = 0 - long frame
          x000 0000 - res/7 - reserve
        03 - 0000 0011
             000x xxxx - cmd/3 = 0 - request for writing
             xxx0 xxxx - res/1 - reserve
             xxxx 0011 - size/4 = 3 - data block size (words)
           01 - cnt/8 = 1 - number of data blocks
              00 00 - offset/16 = 0 starting from 0
                    00 94 - 0000 0000 1001 0100 - mask for Dout 2,4,7/16
                          00 94 - 0000 0000 1001 0100 - status/16
                                AA AA - state of digital outputs/16

data_part 2
+ 07 00 01 02 00 02 + BB BB BB BB +
  07 - typ (07 = analog outputs)
     00 - 0000 0000
          0xxx xxxx - ft/1 = 0 - long frame
          x000 0000 - res/7 - reserve
        01 - 0000 0001
             000x xxxx - cmd/3 = 0 - request for writing
             xxx0 xxxx - res/1 - reserve
             xxxx 0001 - size/4 = 1 - data block size (words)
           02 - cnt/8 = 2 - number of data blocks
              00 02 - offset/16 = 2 starting from 2. output
                      BB BB - data Aout 2
                          BB BB - data Aout 3

data_part 3 + chksum
+ 02 00 21 01 00 05 + CC CC + chksum
  02 - typ (02 = analog inputs)
     00 - 0000 0000
          0xxx xxxx - ft/1 = 0 - lomg frame
          x000 0000 - res/7 - reserve
        21 - 0010 0001
             001x xxxx - cmd/3 = 1 - request for reading
```



```
  xxx0 xxxx - res/1 - reserve
  xxxx 0001 - size/4 = 1 - data block size (words)
01 - cnt/8 = 1 - number of data blocks
   00 05 - offset/16 = 5 starting from 5. input
           CC CC - data Ainp 5
```

4. Configuration

Configuration parameters for ADIO module are described in the document *ADIO protokol for MORSE*[1].

Configuration parameters for input and output unit SEP are described in the document *SEP protokol for MORSE*[2].

[1] http://www.racom.eu/eng/support/prot/adio/index.html
[2] http://www.racom.eu/eng/support/prot/sep/index.html