Technical Data Sheet

RFID reader

CTU-M series

CTU-Mxx-man-eng-v2.pdf

Fig. 1 CTU-M2R

1. Introduction

CTU-M device series is OEM miniature RFID card reader operating at frequency of 13,56 MHz

Main features:

- Support of Mifare S50, S70, Ultralight, Desfire
- built-in antenna
- card memory with build-in lock driver,
- lots of communication interfaces type, depend on version (see table below)
- Built-in relay and buzzer
- Built-in push-button for reset to default settings
- 2 configurable inputs/outputs
- Two-state outputs control
- Read-out of two-state input
- changeable format of sending ID
- MAD functionality
- Data password protected
- Software update via serial interface using *NEFIR* program

CTU-M reader series												
Module type	GPIO	Card memory	Event memory	Relay	Power supply	INTERFACES						
						RS-232	RS-485	RS-232TTL	SPI	I2C	WIEGAND	1WIRE
CTU-M2R*	②	**40**	×	✓	**7-16**	✓						
CTU-M4R	②	40	×	✓	7-16		✓					
CTU-M5N*	②	**40**	×	×	**5**			✓	✓	✓	✓	✓
CTU-M5R	②	40	×	✓	5			✓	✓	✓	✓	✓
CTU-M2RM	②	1000	4000	✓	7-16	✓						

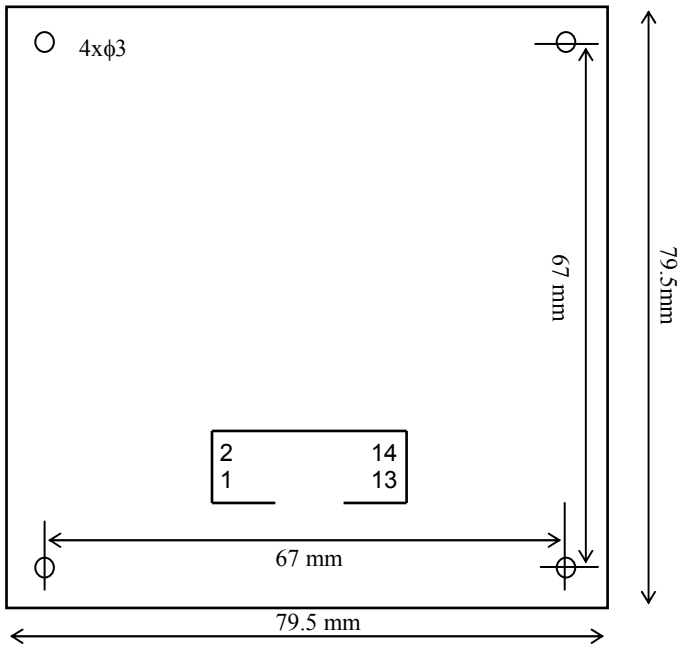
*** - standard version, rest of version for special order**

2. General specification

Supported functionality depending on transponder / card type:		
Transponder type	**ID number read-out**	**Full write and read-out of memory blocks**
S50	YES	YES
S70	YES	YES
Ultralight	YES	YES
Desfire	YES	NO

CTU-Mxx module parameters	
Supply voltage (M2R, M4R model)	7-16 V
Supply voltage (M5R model)	4,5 - 5,5 V
Max. supply current	120 mA
Rated operation radio frequency of module	13,56MHz
Working temperature	-20°c - +65°c
Max. relay current	2A
Appr. read distance for S50	7 cm
Max. output current for GPIO	20mA
Transmission parameters for RS232/RS485/RSTTL	2400, 4800, 9600, 19200, 38400, 57600, 115200 bps, 8 data bits, 1 stop bit, no parity compliant with „Netronix Protocol”
Address on I^2C bus	0xC0
1WIRE family code,address (configurable)	0x01,0x01
WIEGAND number of bits	37

3. Dimension, terminal description



Rys.2 top side view

Pin no.	Description
1	RS232RX, RS485B, RSTTL_RX, 1WIRE, MOSI, SDA
2	RS232TX, RS485A, RSTTL_TX, MISO
3	SCK, SCL
4	CS
5	MCLR
6	GND
7	VCC
8	GPIO 1
9	GPIO 2
10	GND
11	NC
12	NC
13	RELAY 1
14	RELAY 2

4. Module settings by on-board switch

Two function of build-in switch:

- configure to factory settings – press button for 8 seconds
- change interface and RFID transponder type – press button in schematic:

STEP	Number of press	1	2	3	4	5	6
1	MENU1 – interface selection*	-	RS232/485	SPI	WIEGAND	1WIRE	I2C
2	Triple beep						

* - type of interface depends on CTU-M model

5. Transmission protocols

5.1. RS-232/485 transmission protocol

In this data sheet RS-232/485 protocol has been confined to descriptions of commands, responses and their parameters. Header and CRC control sum exist always and are compliant with full “Netronix Protocol” document.

Command frame:

Header	C_CommandName	Response_parameters1…n	CRC

Response frame:

Header	C_CommandName+1	Response_parametrers…m	OperationCode	CRC

RS protocol operation can be tested by means of development tools including free of charge “FRAMER” software”.

5.2. Protocol for I^2C transmission

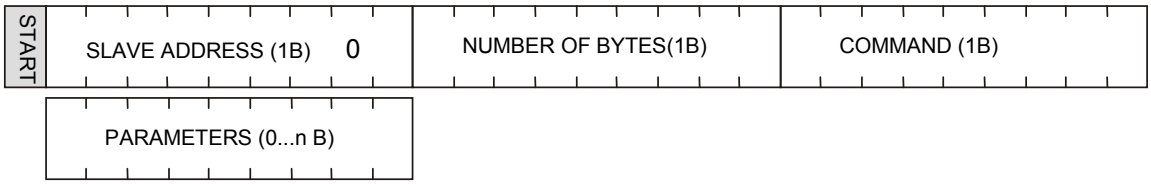
5.2.1. Data exchange algorithm

A module configured depending on table showed on point 4. operates in I^2C interface mode in following sequences:

1. Master (external device) writes command with parameters if necessary into slave device (CTU module)
2 The command is performed (immediately after receiving byte sent quantity declared in frame)
3. Master device reads response, its parameters and operation code. In case of receiving busy byte 0xCB, repeat attempt to read the response after ca. 1 ms

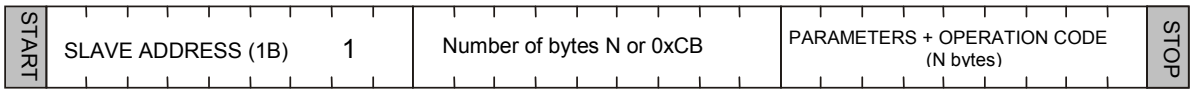
(commands connected with write to/and read from transponders can last up to 100 ms).

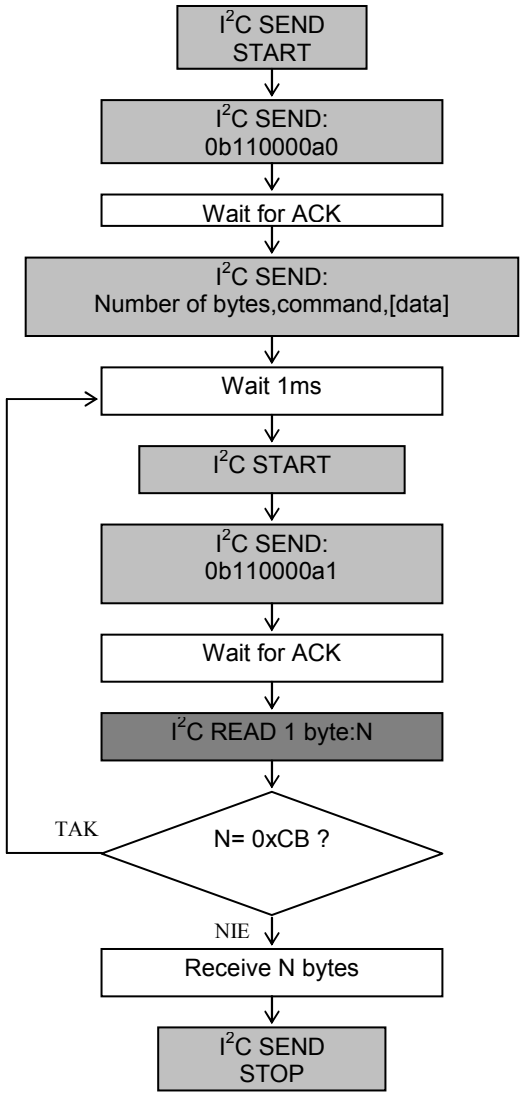
We write inquiry-command to CTU module:



The „number of bytes" field must contain information on byte quantity sent directly "command" fields and "parameters".

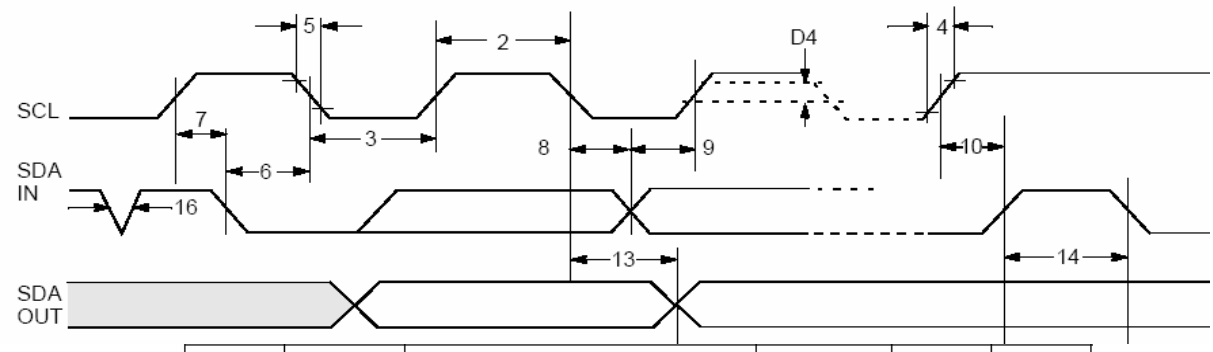
We have then:





5.2.2. Timings

Module sends and receives data at 400 kHz clock frequency considering timings showed below.



Param. No.	Sym.	Characteristic	Min.	Max.	Units
1	FCLK	Clock Frequency	— —	400 100	kHz
2	THIGH	Clock High Time	600 4000	— —	ns
3	TLOW	Clock Low Time	1300 4700	— —	ns
4	TR	SDA and SCL Rise Time **(Note 1)**	— —	300 1000	ns
5	TF	SDA and SCL Fall Time	—	300	ns
6	THD:STA	Start Condition Hold Time	600 4000	— —	ns
7	TSU:STA	Start Condition Setup Time	600 4700	— —	ns
8	THD:DAT	Data Input Hold Time	0	—	ns
9	TSU:DAT	Data Input Setup Time	100 250	— —	ns
10	TSU:STO	Stop Condition Setup Time	600 4000	— —	ns
11	TSU:WP	WP Setup Time	600 4000	— —	ns
12	THD:WP	WP Hold Time	1300 4700	— —	ns
13	TAA	Output Valid from Clock **(Note 2)**	— —	900 3500	ns
14	TBUF	Bus free time: Time the bus must be free before a new transmission can start	1300 4700	— —	ns
15	TOF	Output Fall Time from VIH Minimum to VIL Maximum	20+0.1CB —	250 250	ns

Note 2: Reader keeps in low state first clock pulse of each byte sent until proper state is placed on SDA line.

5.3. Protocol for 1WIRE (Dallas) bus.

Family code	ID1…ID5	Address	CRC
1 byte	5 bytes	1 byte	1 bytet

ID1...5 – unique ID number of transponder
CRC_DAL- check sum of data send

The format conforms 1-WIRE Dallas (e.g.. DS1990A). It means, that described module could be used as a replacement of DS1990A drop.
During operation, a module tries to read-out transponder periodically. If it fails (no successful read-out), module does not response for pulses sent from 1-WIRE master unit. Bus does not "see" the module, which corresponds with lack of reader applying, it means applying the DS1990A drop to drop reader. If module reads out the transponder, the module starts to send data via 1-WIRE bus.

Calculate the CRC value

According to DS1990A specification C value is calculated from equation x^8+x^5+x^4+1 with initial value equal to 0x00. The CRC is calculated on basis of all frame bytes excluding the last one.

An example of CRC value calculation procedure written in C language

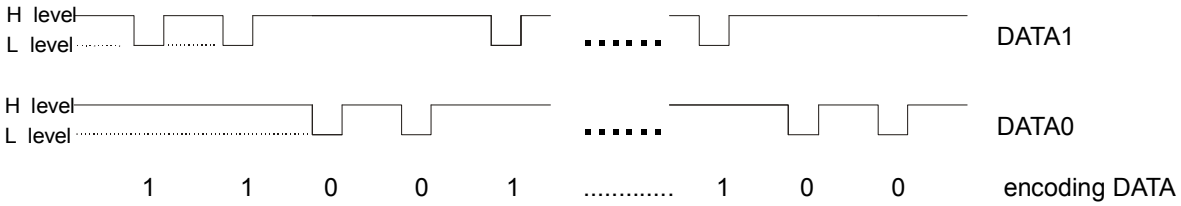
```
unsigned char CalcCRCDallas(unsigned char *SourceAdr)
{
unsigned char i,k,In,CRC=0;
        for(i=0;i<7;i++)
        {
                In=*SourceAdr;
                for(k=0;k<8;k++)
                {
                        if((In^CRC)&1) CRC=((CRC^0x18)>>1)|0x80;
                        else CRC=CRC>>1;
                        In>>=1;
                }
                SourceAdr++;
        }
        return(CRC);
}
```

where *SourceAdr is beginning flag of data buffer

5.4. Wiegand protocol

The format conforms WIEGAND protocol specification for N bits. During operation, a module tries to read-out transponder periodically. If it fails (no successful read-out), module does not send data (bus does not "see" the module). If module reads out the transponder, the module starts to send data via Wiegand bus.

Pulse sequence from left to right.



Total number of pulses (level L) is equal to N. The first being bit sent complements up to parity the bits from first half of total bits. The last bit N complements up to non-parity the bits from second half of bits being sent.
It means, that two bits out of N bits assure the transmission correctness. Information is being sent is written by means bits 2 to N-1, it gives N information bits.

Check sums for bit sequence:

for even N:

EXXXXXXXXXXXXYYYYYYYYYYYYO

or for odd N:

EXXXXXXXXXXXXXXXXXX.....
.....YYYYYYYYYYYYYYYYYO

Where:
E = bit complementing up to parity
O = bit complementing up to non-parity

X = mask for parity calculation
Y = mask for non-parity calculation

5.5. Key management

Key management feature includes key loading to internal key memory. For security reasons, these keys cannot be red-out.

To maintain the highest level of data security, employed a particular philosophy of working with these keys.

It allows unit or person who possesses the highest level of confidence to load a key. Such loading operation can be made one time only, or very rarely.

Reader operation in given application is based on using a key not directly, but on recalling key number, to login to sector.

The result is that, in substance, key does not appear in data bus in given application.

Additionally, a user is advised to make sure key should have proper access rights to sectors. This is accomplished by card initialization process, where new confidential keys are loaded to cards with proper access rights, which are assigned to these keys.

Keys A and B are assigned to each sector.

Commands C_LoadKeyToSKB and C_LoadKeyToDKB load these keys to reader memory without information on key type (A or B).

During logging to sector, user has to input as a parameter value of 0xAA or 0xBB, if he wants, the key which is being recalled would be treated as an A or B.

5.5.1. Key loading into dynamic key memory

Dynamic memory features of automatic content delete in case of supply decay. The memory can be overwritten many times.

Command frame:

Header	C_LoadKeyToDKB	Key1…6	CRC

Where:

Parameter name	Parameter description	Value range
C_LoadKeyToDKB	Key loading to key dynamic memory	0x14
Key1…6	6-byte code	whichever

Response frame:

Header	C_LoadKeyToDKB +1		OperationCode	CRC

5.5.2. Key loading to key static memory

Important feature of static memory is that in case of supply decay, data stored in it will not be lost. The memory can be overwritten many times.

Command frame:

Header	C_LoadKeyToSKB	Key1…6, KeyNo	CRC

Where:

Parameter name	Parameter description	Value range
C_LoadKeyToSKB	Key loading to key static memory	0x16
Key1…6	6-byte key	whichever
KeyNo	Key number. It possible to load 32 different keys to a reader.	0x00…0x1f

Response frame:

Header	C_LoadKeyToSKB +1		OperationCode	CRC

5.6. Commands for communication with transponder

5.6.1. On/off switching of reader field

Command frame:

Header	C_TurnOnAntennaPower	State	CRC

Where:

Parameter name	Parameter description	Value range
C_TurnOnAntennaPower	On/off switching of reader field	0x10
State	On state	0x00 – switching the field off 0x01 – switching the field on

Response frame:

Header	C_TurnOnAntennaPower +1		OperationCode	CRC

5.6.2. Selecting one of many transponders

Command frame:

Header	C_Select	RequestType	CRC

Where:

Parameter name	Parameter description	Values
C_Select	Selecting one of many transponders	0x12
RequestType	Type of transponder selection	0x00 - Standard selecting from group of transponders, which are not in stand-by mode 0x01 - Selecting from group of transponders, which are in reader field.

Response frame:

Header	C_Select +1	ColNo, CardType, ID1…….IDn	OperationCode	CRC

Where:

Parameter name	Parameter description	Meaning
ColNo	Number of collisions during one transponder selecting. This figure can be equal to the transponder quantities, which are in the field simultaneously, and which are not in stand-by state.	
CardType	Type of selected transponder	0x50 – S50 0x70 – S70 0x10 – Ultra Light 0xdf – Des Fire
ID1…IDn	Unique number of transponder	ID1 – LSB, IDn – MSB

5.6.3. Logging by means of Dynamic Key Buffer to selected sector of transponder

To complete logging successfully, it is important after any input of the reader, to reload the Dynamic Key Buffer.

Command frame:

Header	C_LoginWithDKB	SectorNo, KeyType, DKNo	CRC

Where:

Parameter name	Parameter description	Value range
C_LoginWithDKB	Logging to sector	0x18
SectorNo	Transponder sector number, to which user wants to login.	0x00 – 0x0f (s50) 0x00 – 0x27 (s70)
KeyType	Key type, which is inside internal Dynamic Key Buffer.	0xAA – key of A type 0xBB – key of B type
DKNo	Dynamic key number	0x00

Response frame:

Header	C_LoginWithDKB +1		OperationCode	CRC

5.6.4. Logging by means of Static Key Buffer to selected sector of transponder

To complete logging successfully, it is important to load Static Key Buffer first.

Command frame:

Header	C_LoginWithSKB	SectorNo, KeyType, SKNo	CRC

Where:

Parmeter name	Parameter description	Value range
C_LoginWithSKB	Logging to sector	0x1a
SectorNo	Transponder sector number, to which user wants to login.	0x00 – 0x0f (s50) 0x00 – 0x27 (s70)
KeyType	Key type, which is inside internal Static Key Buffer.	0xAA – key of A type 0xBB – key of B type
SKNo	Static Key number	0x00…0x1F

Response frame:

Header	C_LoginWithSKB +1		OperationCode	CRC

5.6.5. Reading-out the content of transponder block

Command frame:

Header	C_ReadBlock	BlockNo	CRC

Where:

Parameter name	Parameter description	Value range
C_ReadBlock	Read-out of transponder block content	0x1e
BlockNo	Block number within given sector	**Sector and block numeration

Response frame:

Header	C_ReadBlock +1	Data1….. Data16	OperationCode	CRC

Where:

Parameter name	Parameter description	Value range
Data1…. Data16	Red-out of data from transponder block	

5.6.6. Writing the content of transponder block

Command frame:

Header	C_WriteBlock	BlockNo, Data1….. Data116	CRC

Where:

Parameter name	Parameter description	Value range
C_WriteBlock	Write of transponder block content	0x1c
BlockNo	Block number within given sector	**Sector and block numeration
Data1…. Data16	Data, which are to be written into transponder block.	whichever

Response frame:

Header	C_WriteBlock +1		OperationCode	CRC

5.6.7. Copying the content of transponder block into other block

Command frame:

Header	C_CopyBlock	SourceBlockNo, TargetBlockNo	CRC

Where:

Parameter name	Parameter description	Value range
C_CopyBlock	Copying the content of transponder block into other block	0x60
SourceBlockNo	Source block	**Sector and block numeration
TargetBlockNo	Target block for data	

Response frame:

Header	C_CopyBlock +1		OperationCode	CRC

5.6.8. Writing the page content into Mifare UL

Command frame:

Header	C_WritePage4B	PageAdr, Data1...4	CRC

Where:

Parameter name	Parameter description	Value range
C_WritePage4B	Writing the page content into Mifare UL	0x26
PageAdr	Page number in transponder	0x00…0x0f
Data1...4	Data, which are to be written	whichever

Response frame:

Header	C_WritePage4B +1		OperationCode	CRC

5.6.9. Reading the page content in Mifare UL

Command frame:

Header	C_ReadPage16B	PageAdr	CRC

Where:

Parameter name	Parameter description	Value range
C_ReadPage16B	Read-out of page content in Mifare UL	0x28
PageAdr	Page address, from which read-out of following four pages should start. If PageAdr>0x????, starts read-out process of pages, which are present at memory beginning.	0x00…0x0f

Response frame:

Header	C_ReadPage16B +1	Data1…16	OperationCode	CRC

Where:

Parameter name	Parameter description	Value range
Data1…16	Red-out of data from four subsequent pages.	whichever

5.6.10. Writing values to transponder block

Command frame:

Header	C_WriteValue	BlockNo, BackupBlockNo,Value1...4,	CRC

Where:

Parameter name	Parameter description	Value range
C_WriteValue	Write of values to transponder block.	0x34
BlockNo	Block number within given sector, into which the Value will be written.	**Sector and block numeration
BackupBlockNo	Declared block number including the Value copy. BackupBlockNo has no influence for system operation, but user can/should make the Value copy by himself.	**Sector and block numeration
Value1...4	The Value, which is written to	whichever

	transponder block.	

Response frame:

Header	C_WriteValue +1		OperationCode	CRC

5.6.11. Reading-out the values from transponder block

Command frame:

Header	C_ReadValue	BlockNo	CRC

Where:

Parameter name	Parameter description	Value range
C_ReadValue	Read-out of the Value from transponder block.	0x36
BlockNo	Block number within given sector, from which the Value will be red-out.	**Sector and block numeration

Response frame:

Header	C_ReadValue+1	Value1...4, BackupBlockNo	OperationCode	CRC

Where:

Parameter name	Parameter description	Value range
Value1...4	Red-out Value from transponder block.	
BackupBlockNo	Block number, which can include the Value copy.	**Sector and block numeration

5.6.12. Increasing the value included in transponder block

To execute a command successfully, format of data included in declared block should be “Value” format.

Command frame:

Header	C_IncrementValue	BlockNo, Value1...4	CRC

Where:

Parameter name	Parameter description	Value range
C_IncrementValue	Increasing the value included in transponder block.	0x30
BlockNo	Block number within given sector, in which the Value will be modified.	**Sector and block numeration

Value1...4	Value, which is being added to existed real value of block transponder.	

Response frame:

Header	C_IncrementValue +1		OperationCode	CRC

5.6.13. Decreasing the value included in block transponder

To execute a command successfully, format of data included in declared block should be “Value” format.

Command frame:

Header	C_DecrementValue	BlockNo, Value1...4	CRC

Where:

Parameter name	Parameter description	Value range
C_DecrementValue	Decreasing the Value included in transponder block.	0x32
BlockNo	Block number within given sector, in which the Value will be modified	**Sector and block numeration
Value1...4	The Value, which is being subtracted from existed real value of block transponder.	whichever

Response frame:

Header	C_DecrementValue+1		OperationCode	CRC

5.6.14. Setting the transponder in field into sleep mode

To set transponder to sleep mode, select it first.

Command frame:

Header	C_Halt		CRC

Parameter name	Parameter description	Value range
C_Halt	Setting the transponder in field into sleep mode.	0x40

Response frame:

Header	C_Halt+1		OperationCode	CRC

5.7. Reader inputs and outputs

Reader has inputs and outputs which are configurable. Inputs are controlled directly from microcontroller outputs. Output load current is up to 20 mA.

5.7.1. Writing the output state

Command frame:

C_WriteOutputs	IONo, State

Where:

Parameter name	Parameter description	Value range
C_WriteOutputs	Output state write	0x70
IONo	I/O port number. The port should be configured as an output	0x1..0x7 for UW-U4R 0x1..0xC for UW-U4G
State	Requested output state	0x00 or 0x01

Response frame:

C_WriteOutputs +1		Operation Code

5.7.2. Reading the input state

Command frame:

C_ReadInputs	IONo

Where:

Parameter name	Parameter description	Value range
C_ReadInputs	Input state reed-out	0x72
IONo	I/O port number. Should be configured as an input.	0x0..0x7 dla UW-U4R 0x0..0xC dla UW-U4G

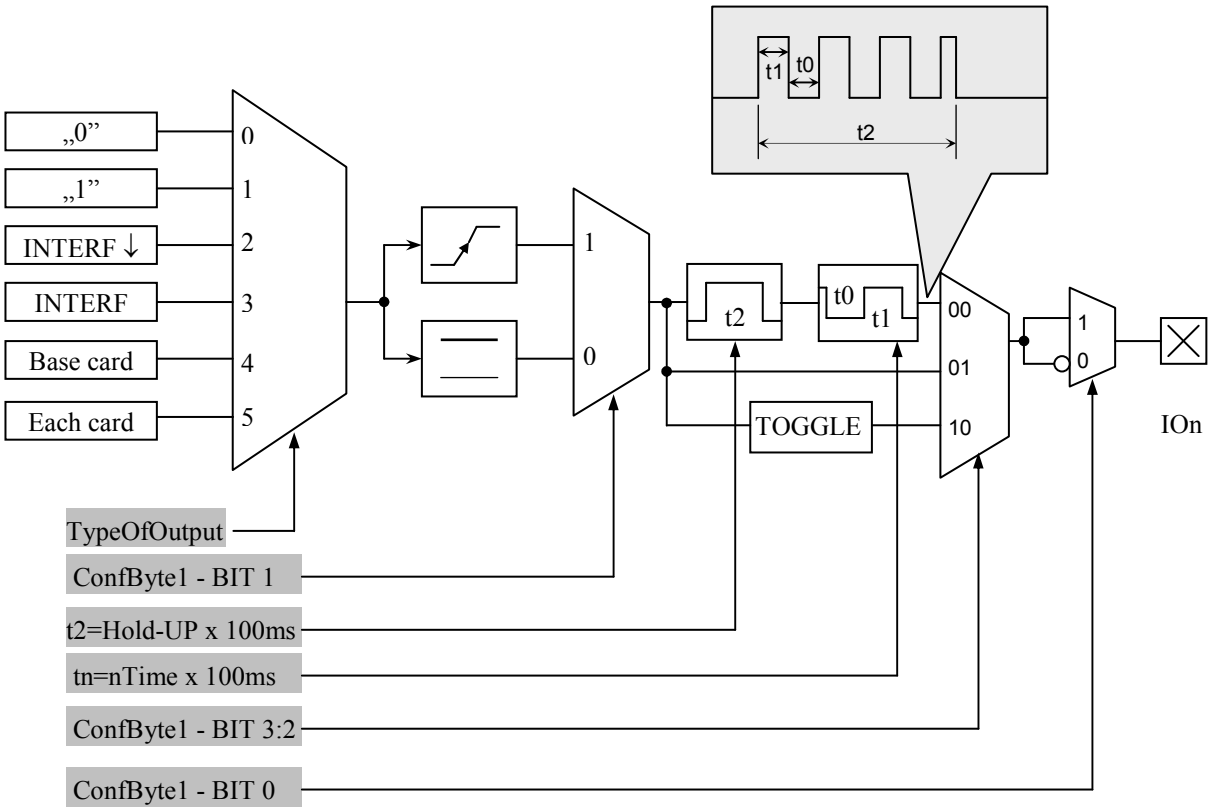
Response frame:

C_ReadInputs +1	State,[COUNTER]	Operation Code

Where:

Parameter name	Parameter description	Value range
State	Input state which has been red	
Counter	Counter state for counter type input.	

5.7.3. Writing the settings to any port



Command frame:

Header	C_SetIOConfig	IONo, IOConfigData1…n	CRC

If we set a port as output, IOConfigData1…n parameters are as below:

Dir, ConfByte1, TypeOfOutput, Hold-up, 0Time, 1Time

Where:

Parameter name	Parameter description	Value range
C_SetIOConfig	Writing the configuration of every port	0x50
IONo	I/O port number, which is to be configured	0x0..0x4
Dir	Port direction	0x00 – output
ConfByte1	One byte in which: BIT0 assigns output type as normally open or normally closed. BIT 1 determines reaction method of each output as sensitive for simulation changing (slope sensitive)	**ConfByte1 Bit 0** 0-Normally closed 1-Normally open **ConfByte1 Bit 1** 0-level sensitive 1-slope sensitive

	or as sensitive for simulation state (state sensitive). BIT3:2 determines operation method of output referring to trigger signal state.	**ConfByte1 Bit 3:2** 00 – rectangular wave generator 01- directly 10 – output state change
TypeOfOutput	Source of driving signal	0x00 – permanently off 0x01 – permanently on 0x02 – driven via serial interface 0x03 – driven via serial with automatic reset(edge emulation) 0x04 – driven by internal access control mechanism ACM. This output is driven in case of applying the card to reader, which is written into internal card base. 0x05 – set in case of applying freely selected card to reader.
Hold-up	Time of maintaining the on state after actuation stopped. This time is specified as: Hold-up x 100 ms During “hold-up” time, it is possible to configure the output, whish is able to generate rectangular wave. By means of following parameters are configured “Logic 1” time and “Logic 0” time:	
0Time	Logic 0 time	
1Time	Logic 1 time	

<u>If we set a port as a input, IOConfigData1…n parameters would be as below:</u>
Dir, Triger, TypeOfInput, Delay,

Where:

Parameter name	Parameter description	Value range
C_SetIOConfig	Writing the configuration of freely selected port.	0x50
IONo	I/O port number, which is to be configured.	0x00,0x01,0x07
Dir	Port direction	0x01 – input
TypeOfInput	Input type	0x03
Delay	Delay	0x00

5.7.4. Reading-out the configuration of freely selected port

Command frame:

Header	C_GetIOConfig	IONo	CRC

Where:

Parameter name	Parameter description	Value range
C_GetIOConfig	Reading-out the configuration of freely selected port.	0x52
IONo	I/O port number, which configuration is to be red-out.	0x00…0x05

Response frame:

Header	C_GetIOConfig +1	IOConfigData1…n	OperationCode	CRC

Where:

Parameter name	Parameter description	Value range
IOConfigData1…n	This is the same, as in case of configuration write.	

Some I/O of CTU-M reader has no possibility to toggle port direction.
To accomplish proper configuration, input proper direction option to given port.

LIST OF EXISTING PORTS, WHICH CAN BE DRIVEN IN UW-M4R		
Port number	Direction	Description
0	input/output	GPIO1
1	input/output	GPIO 2
2	output	RELAY
3	output	BUZZER

Response frame:

Header	C_SetIOConfig +1		OperationCode	CRC

5.8. Access password

5.8.1. Logging to reader

Command frame:

Header	C_LoginUser	Data1…n, 0x0	CRC

Where:

Parameter name	Parameter description	Value range
C_LoginUser	Logging to reader	0xb2

Data1…n	This is any byte string	Any from range: 0x01…0xff. String length, which can be 0 to 8 bytes
0x00	Logic Zero, which terminates a string.	0x00

Response frame:

Header	C_LoginUser+1		OperationCode	CRC

5.8.2. Changing the password

Command frame:

Header	C_ChangeLoginUser	Data1…n, 0x0	CRC

Where:

Parameter name	Parameter description	Value range
C_ChangeLoginUser	Password change	0xb4
Data1…n	This is any byte string, which will form valid access password.	Any from range: 0x01…0xff. String length, which can be 0 to 8 bytes
0x00	Logic Zero, which terminates a string.	0x00

If =0x00, a reader will not be protected by password. At any moment, there is possible to set new password later on, to protect the reader by it.

Response frame:

Header	C_ChangeLoginUser+1		OperationCode	CRC

5.8.3. Logging out of the reader

This command sets latest password as an invalid.

Command frame:

Header	C_LogoutUser		CRC

Parameter name	Parameter description	Value range
C_LogoutUser	Logging out of the reader.	0xd6

Response frame:

Header	C_LogoutUser +1		OperationCode	CRC

5.9. Operating the transponder internal memory

5.9.1. Reading-out the transponder number from memory

Command frame:

Header	C_CardMemoryRead	AdrL, AdrH	CRC

Where:

Parameter name	Parameter description	Value range
C_CardMemoryRead	Read-out of transponder number from memory.	0x20
AdrL, AdrH	Younger and older byte respectively.	0x0000…0x01fd

Response frame:

Header	C_CardMemoryRead+1	ID1(L)….ID5(H), Right	OperationCode	CRC

Where:

Parameter name	Parameter description	Value range
ID1(L)….ID5(H)	Five bytes of transponder number	
Right	Access rights to given transponder	0x01

5.9.2. Writing the transponder name to memory

Command frame:

Header	C_CardMemoryWrite	AdrL, AdrH, ID1(L)….ID5(H), Right	CRC

Where:

Parameter name	Parameter description	Value range
C_CardMemoryWrite	Write of transponder number into memory.	0x22
AdrL, AdrH	Younger and older byte respectively	0x00…0x01fd
ID1(L)….ID5(H)	Five bytes of transponder number	Any of five bytes
Right	Access rights or function performed by transponder.	0x01

Response frame:

Header	C_CardMemoryWrite+1		OperationCode	CRC

Where:

5.10. Operating the built-in access control

5.10.1. Writing the configuration of access control

Command frame:

Header	C_AccesControlConfigWrite	Mode	CRC

Where:

Parameter name	Parameter description	Value range
C_AccesControlConfigWrite	Write of access control configuration.	0x74
Mode	Operation mode of control access module.	0x00 – module disabled 0x01 – module enabled

Response frame:

Header	C_AccesControlConfigWrite+1		OperationCode	CRC

5.10.2. Reading-out the configuration of access control

Command frame:

Header	C_AccesControlConfigRead		CRC

Where:

Parameter name	Parameter description	Value range
C_ AccesControlConfigRead	Read-out of access control configuration.	0x76

Response frame:

Header	C_AccesControlConfigRead+1	Mode	OperationCode	CRC

Where:

Parameter name	Parameter description	Value range
Mode	Operation mode of access control module.	0x00 – module disabled 0x01 – module enabled

5.10.3. Writing the “automatic read” configuration

This command sets operation method of automatic device, reading the unique transponder number UID.

The reader described below makes possible to hold-on operation of automatic device for a while, in case of suitable transmission via serial interface.

If the reader will operate in mixed mode i.e.:
- automatic reading device UID is enabled and:
- master device (computer, controller) communicates with reader or with transponders via reader,
it is required, to configure the reader correctly, so in case of communication with a reader or transponder, automatic reading device would hold-on its operation.

Command frame:

Header	C_SetAutoReaderConfig	ATrig, AMode, AOfflineTime, ASerial, Abuzz	CRC

Where:

<table>
<tr><th>Parameter name</th><th>Parameter description</th><th colspan="2">Value range</th></tr>
<tr><td>C_SetAutoReader Config</td><td>Writing the automatic device configuration.</td><td colspan="2">0x58</td></tr>
<tr><td>ATrig</td><td>Defines, when automatic reading device UID will operate.</td><td colspan="2">0-automatic device disabled permanently
1-automatic device enabled permanently
2=enabled automatically in case of transmission lack on interface for a time longer than AOfflineTime
3=enabled automatically, in case of no recall of communication commands with transponder for a time longer than AOfflineTime</td></tr>
<tr><td>AOfflineTime</td><td>Lack of transmission time on interface bus
T= AofflineTime * [100ms]
Lack of transmission can concern to any commands (Atrig=2), or commands for communication with transponder (Atrig=3).

Commands for communication with transponder:
C_TurnOnAntennaPower
C_Select</td><td colspan="2">0x00…0xff</td></tr>
<tr><td>ASerial</td><td>Automatic sending the UID transponder number, after reading it automatically from transponder.</td><td colspan="2">0-never
1-for the first applying the transponder only
2-sends all</td></tr>
<tr><td rowspan="6">AMode</td><td rowspan="6">Selection the format of sending number

8 bits:

MSB LSB
R R H CR R E I A</td><td>R</td><td>Reserved, always 0</td></tr>
<tr><td>CR=1</td><td>Number which is ended with line end mark CR+LF</td></tr>
<tr><td>E=1</td><td>information extended with cards umber in filed and card type</td></tr>
<tr><td>I=1</td><td>Number in reversed order</td></tr>
<tr><td>A=1
H=0</td><td>Number sent in ASCII format</td></tr>
<tr><td>A=0
H=0</td><td>Number sent in Nertonix format</td></tr>
<tr><td></td><td></td><td>A=0
H=1</td><td>Number sent in binary format</td></tr>
<tr><td>ABuzz</td><td>Automatic indication of reading by means of buzzer, after automatic UID read-out from transponder.</td><td colspan="2">0-never
1-for the first applying the transponder only
2-indicates all</td></tr>
</table>

Response frame:

Header	C_ SetAutoReaderConfig +1		OperationCode	CRC

5.10.4. Reading-out the configuration of automatic device

Command frame:

Header	C_ GetAutoReaderConfig		CRC

Where:

Parameter name	Parameter description	Value range
C_GetAutoReaderConfig	Read-out of automatic device configuration.	0x5a

Response frame:

Header	C_ GetAutoReader Config +1	ATrig, AOfflineTime, ASerial, Abuzz,Amulti	OperationCode	CRC

Where:

The meaning of response parameters is the same as described before.

5.10.5. Setting the date and time

Following setting has no influence for reader operation today.

Command frame:

Header	C_SetRtc	Year, Month, Day, Hour, Minute, Second	CRC

Where:

Parameter name	Parameter description	Value range
C_SetRtc	Date and time set-up	0xb8
Year	year	0…99
Month	month	1…12
Day	day	1…31
Hour	hour	0…23
Minute	minute	0…59
Second	second	0…59

Response frame:

Header	C_SetRtc +1		OperationCode	CRC

5.10.6. Reading-out the date and time

Command frame:

Header	C_GetRtc		CRC

Where:

Parameter name	Parameter description	Value range
C_GetRtc	Read-out of date and time	0xb6

Response frame:

Header	C_GetRtc+1	Year, Month, Day, Hour, Minute, Second	OperationCode	CRC

Where:
The meaning of response parameters is the same as described before.

5.11. Configuring the UART serial interface

5.11.1. Writing the configuration of serial port

Command:

	C_SetInterfaceConfig	Mode, Adr, Baudrate	

Where:

Parameter name	Parameter description	Value range
C_SetInterfaceConfig	Serial interface configuration write	0x54
Mode		0x01
Adr	Address on RS-485 bus	0x01…0xfe
Baudrate	Data baud rate on RS-485 bus	0x01=2400 bps 0x02=4800 bps 0x03=9600 bps 0x04=19200 bps 0x05=38400 bps 0x06=57600 bps 0x07=115200 bps

Response:

	C_SetInterfaceConfig +1		OperationCode	

5.11.2. Reading the configuration of serial interface

Command:

	C_ GetInterfaceConfig		

Where:

Parameter name	Parameter description	Value range
C_GetInterfaceConfig	Serial interface configuration read-out	0x56

Odpowiedź:

	C_ GetInterfaceConfig +1	Mode, Adr, Baudrate	OperationCode	

Where:
The meaning of response parameters is the same as described before.

5.12. Managing the events

The CTU-MxxM series readers has equipped with event memory of capacity 4400 records. Reason of event can be operation related to card or state changing on reader outputs. The readers does not have RTC clock with battery back-up. After supply failure, clock is reset to defaults: date: 1 January 2000, time: 00:00:00. Event counter is not reset.

5.12.1. Setting the event recorder

Command frame:

Header	C_SetEventTrig	CardTrig, IO1Trig	CRC

Wherein:

Parameter name	Parameter description	Value range
C_SetEventTrig **0x7C**	Setting the event masking	0x7C
CardTrig	Masking the events related to card (see below)	0x00 - 0xFF
In1Trig-In4Trig	Masking the events related to inputs (see below)	0x00-0xFF

Response frame:

Header	C_SetEventTrig+1		OperationCode	CRC

Masking byte of events related to card

Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Reserve	Out of memory	Card removal	Card adding	Reserve	Master card	Outside base card	Inside base card

E.g. byte 0x25(00100101) means that events will be written in case of:

- *inside base card has been red-out,*
- *card written as master has been written,*
- *inside base card has been removed*

Masking bytes related to inputs state change

Byte	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
IO1Trig	IO[3]F	IO[3]R	IO[2]F	IO[2]R	IO[1]F	IO[1]R	IO[0]F	IO[0]R

Bits IO[n]R denote reaction to the input **n** positive slope,
Bits IO[n]F denote reaction to the input **n** negative slope

E.g. In4Trig-In1Trig configuration byte sequence: ***0x00,0x31,0x40,0x08****, causes, that events will be written in case of:*

- *Any state change of input with index 10 occurs*
- *Positive slope appears on input with index 8*
- *Positive slope appears on input with index 7*
- *Negative slope appears on input with index 1*

During configuring the event triggers, decide which port is configured as an input. Do not configure events for those I/O's, which are outputs.

To guarantee correctness of event write process, time between two subsequent triggers must be longer than 20 ms.

5.12.2. Reading the event recorder

Command frame:

Header	C_GetEventTrig		CRC

Wherein:

Parameter name	Parameter description	Value range
C_GetEventTrig **0x7E**	Reading the configuration of event recorder	0x7E

Response frame:

Header	C_GetEventTrig+1	CardTrig, In1Trig	OperationCode	CRC

5.12.3. Reading the counters related to event memory.

Command frame:

Header	C_GetEventParam		CRC

Wherein:

Parameter name	Parameter description	Value range
C_GetEventParam **0x78**	Reading the configuration of event recorder	0x78

Response frame:

Header	C_GetEvent Param+1	CapL, CapH, PointerL, PointerH,TotB3,TotB2,TotB1,TotB0	OperationCode	CRC

CapH:CapL – two-byte value, which defines event memory capacity.
PointerH:PointerL – two-byte value, which marks from first free event.
TotB3:TotB2:TotB1:TotB0 – four-byte value, which defines number of events recorded from the moment of counter reset.
Events are recorded in sequence from index 0 up to Cap-1. In the moment memory gets full, the counter is being "overturned", and older inputs are overwritten.

Example:
If using C_GetEventParam command, we have red that event memory capacity is 4400 inputs; the total value of input events is 5678. For instance, if we want to read the event with no. 5600, event index event of interest will be 5678-4400=1278.
If we want to read the last event, we can use Pointer value. The last event index will be Pointer-1.

5.12.4. Reading the events

Command frame:

Header	C_GetEvent	EvNoL, EvNoH	CRC

Wherein:

Parameter name	Parameter description	Value range
C_GetEvent **0x7a**	Reading the event	0x7a
EvNoL,EvNoH	LSB and MSB of event index	

Response frame:

Header	C_GetEvent+1	YY,MM,DD,hh,mm,ss,type, B1,B2,B3,B4,B5	OperationCode	CRC

YY,MM,DD – year, month, day of event occurrence
hh,mm,ss - hour, minute, second of event occurrence
type - event type

Depending on value “type” the 8-th bit of byte, there are distinguished two assignments:

Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1
0 – card	Out of memory	Removed	Added	reserved	Master	Outside base	Inside base
1 -inputs	reserved	reserved	reserved	N4	N2	N1	N0

N4:N0 –number of input, on which event trigger signal appeared.

- If given event was related to card, the B1-B5 bytes comprise card ID number.

B1	B2	B3	B4	B5
UID1	UID2	UID3	UID4	UID5 (Unique)

- If event is related to input change, B1-B5 bytes comprise information regarding input state, and have format:

B1				B2	B3	B4	B5
IO0	IO1	IO2	IO3	Reserved	Reserved	Reserved	Reserved

5.13. MAD – Mifare Application Directory

5.13.1. Card MAD formatting

Command frame:

Header	C_FormatMad	Type, Infobyte	CRC

Wherein:

Parameter name	Parameter description	Value range
C_FormatMad **0xA8**	Formatting to MAD	0xA8
Type	1 - MAD1 (15 sectors) 2 – MAD2 (30 sectors)	0x01,0x02
Infobyte	Mark in emitent sector (default 0x00)	0x00-0x1F

Response frame:

Header	C_FormatMad+1		OperationCode	CRC

Notes:
Before you run C_FormatMad command:

- switch AutoReader mode off (using C_SetAutoReaderConfig command)
- load the keys (default 0xff,0xff,0xff,0xff,0xff,0xff)
- turn antenna supply on (using C_TurnOnAntennaPower)
- select the cart (using C_Select command)
- login to sector with number 0, using key of AA type

5.13.2. Adding the application to MAD directory

Command frame:

Header	C_AddApplication	LSB, MSB, Sector	CRC

Wherein:

Parameter name	Parameter description	Value range
C_AddApplication	Adding application	0xAA
LSB	LSB of application number	0x00 - 0xFF
MSB	MSB of application number	0x00 - 0xFF
Sector	Number of sector, in which the application is to be present	0x01-0x0F :MAD1 0x01-0x1F :MAD2

Response frame:

Header	C_AddApplication+1		OperationCode	CRC

Notes:
Application number should be other than 0x0000
Before you run C_AddApplication command:

- switch AutoReader mode off (using command C_SetAutoReaderConfig)
- load the keys (default 0xff,0xff,0xff,0xff,0xff,0xff)
- turn antenna supply on (using C_TurnOnAntennaPower command)
- select the card (using C_Select command)
- login to sector with number 0, using key of AA type

5.13.3. Pursuing the sector for given application

Command frame:

Header	C_GetSectorMad	LSB, MSB	CRC

Wherein:

Parameter name	Parameter description	Value range
C_GetSectorMad **0xAC**	Pursuing the sector	0xAC
LSB	LSB of application number	0x00 - 0xFF
MSB	MSB of application number	0x00 - 0xFF

Response frame:

Header	C_GetSectorMad+1	Sector	OperationCode	CRC

Notes:

Before you run C_GetSectorMad command:

- switch AutoReader mode off (using C_SetAutoReaderConfig command)
- load the keys (using 0xff,0xff,0xff,0xff,0xff,0xff)
- turn antenna supply on (using C_TurnOnAntennaPower command)
- select the card (using C_Select command)
- login to sector with number 0, using key of AA type

If response byte is 0x00, it will mean, that given application is not present in MAD catalogue.

5.13.4. Pursuing the next sector of application

Command frame:

Header	C_GetSectorMadNext	LSB, MSB	CRC

Wherein:

Parameter name	Parameter description	Value range
C_GetSectorMad	Pursuing the next sector	0xAE

Response frame:

Header	C_GetSectorMadNext+1	Sector	OperationCode	CRC

Notes:
Before you run C_GetSectorMadNext command, perform sector searching operation using C_GetSectorMad, command, of which pursuing result was other than 0.

If response byte is 0x00, it will mean, than no more sectors have been found for given application.

5.14. Other commands

5.14.1. Remote reset of reader

Command frame:

Header	C_Reset		CRC

Where:

Parameter name	Parameter description	Value range
C_Reset	Remote reader reset	0xd0

Response frame:

Header	C_Reset +1		OperationCode	CRC

5.14.2. Reading-out the reader software

Command frame:

Header	C_FirmwareVersion		CRC

Where:

Parameter name	Parameter description	Value range
C_FirmwareVersion	Read-out of reader software version	0xfe

Response frame:

Header	C_FirmwareVersion+1	Data1…..n	OperationCode	CRC

Where:
Data1…n is sequence of dots, which are written as an ASCII codes.

5.14.3. Change buzzer volume

Use this command to set and store setting in EEPROM memory.

Command frame:

Header	C_BuzzerConfig	volume	CRC

Gdzie:

Parameter name	Parameter description	Value range
C_BuzzerConfig		0xD8
volume	Buzzer volume value	0x00-0x0a

Response frame:

Header	C_BuzzerConfig+1		OperationCode	CRC

5.15. Code meanings in response frames

Operation code name	Description	Value
OC_Error	Error	0x00
OC_ParityError	Parity error	0x01
OC_RangeError	Parameter range error	0x02
OC_LengthError	Data quantity error	0x03
OC_ParameterError	Parameter error	0x04
OC_Busy	Internal modules are busy at the moment.	0x05
OC_NoACKFromSlave	No internal communication	0x22
OC_CommandUnknown	Unknown command	0x07
OC_WrongPassword	Wrong password or last password terminated i.e. automatic LogOut occurred.	0x09
OC_NoCard	No transponder	0x0a
OC_BadFormat	Wrong data format.	0x18
OC_FrameError	Transmission error. Noise occurrence possible.	0x19
OC_NoAnswer	No response from transponder.	0x1E
OC_TimeOut	Operation time out. No transponder in reader field possible.	0x16
OC_Successful	Operation completed successfully.	0xff

6. Meaning of symbols used in the specification

**Sectors and block numeration

For S50 cards:
SectorNo=0x00…0x0f
BlockNo=0x00…0x03

For S70 cards:
SectorNo=0x00…0x20 BlockNo=0x00…0x03
SectorNo=0x21…0x27 BlockNo=0x00…0x0f

7. Mechanism of Master ID

Master ID mechanism is based on principle the quick adding/removing of user card to/out of reader memory by means of „master card”.

If you want to register a card as a „master card”, it is required to clear card memory first by means of reset function to factory defaults.

After clearing the memory, apply selected card to module, whenever you like. This moment, the card becomes “master card”. It is impossible to remove or add the master card by means of other card.

If you want to register a card as a “user card”, apply “master card” to reader first, and next during five seconds, apply registered card.

If you want to remove “user card” from memory, apply “master card” to reader first, and next during five seconds apply card which is being removed.

After applying to a reader the “user card”, the reader enables electric output, which has been programmed as a controlled by internal access control mechanism.

8. Reset to default settings

To restore default settings, connect reset terminal with ground for 2 s or longer. During restoring the defaults following reader parameters are fixed:

Parameter name or functionality	Value or setting
Address on serial bus	0x01
Baud rate on serial bus	9600 bps
Access password	0x0 - no password
Port 0	Common purpose input
Port 1	Common purpose input
Port 2	lock “on” indication
Port 3	lock “on” indication
“Autoreader” configuration	0x2,0x14,0x1,0x1,0x01,0xff
Transponder type	Unique
Whole internal memory of transponders with Master card	0xff 0xff 0xff 0xff means “memory cleared”

9. Operation example of transponder

After correct connection of reader and achieving the bi-directional communication between the reader and master computer, it is possible to perform read-out and write operation of transponder memory.

Following operation assumes, that reader is in default condition, and applied S50 card is in default condition too. It means this card has full access rights and both 0xff ff ff ff ff ff keys.

Logging to the reader is to make changes in its factory configuration.
C_LoginUser, 0x31, 0x32, 0x33, 0x34, 0x00

Because during manual experiments, time between subsequent commands sent via serial interface is large and reaches values from some second to some minutes, it is required to disable internal UID automatic read-out device.
It should be done by means of command:

SetAutoReaderConfig with parameters: 0x00, 0x00, 0x00, 0x00.

To read-out the transponder, first load key to key memory.
So load the key to SKB, by means of:
C_LoadKeyToSKB, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00

Enable the field.
TurnOnAntennaPower, 0x01
Apply transponder to reader.

Select transponder
C_Select, 0x00

Login to e.g. sector 3.
C_LoginWithSKB, 0x03, 0xAA, 0x00

Read-out 2nd block content in 3rd sector.
C_ReadBlock, 0x02

If all Operation Codes in response frames were marked as OC_Successful, so obtained values are the values which have been red-out from the block.

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for RF Modules ***category:***

Click to view products by Netronix ***manufacturer:***

Other Similar products are found below :

HMC-C009 HMC-C011 nRF24L01P-MODULE-PCB HMC-C021 HMC-C024 XB9XT-DPRS-721 XBP9B-DMUTB022 nRF24L01P-MODULE-SMA CMD-KEY2-418-CRE XM-C92-2P-UA XB9XT-DPUS-721 V640-A90 HMC-C583 MAAM-008818-TR3000 MTSMC-H5-U SIMSA868-PRO SIMSA915C-PRO SIMSA868C-PRO SIMSA433C-PRO SIMSA915-PRO XBP9B-DMUT-042 HMC-C582 HMC-C022 XBP9B-DPST-041 XBP9B-DMWT-042 SM-MN-00-HF-RC HMC-C031 MT-02 M1002GB 702-W SIMSA868C-N-PRO SIMSA433C-N-PRO SIMSA915C-N-PRO ADP-R202-00B PEPPER WIRELESS C1 USB S2-10732-Z1T61 S2-107XB-Z2356-Z2352 S2-10672-Z1L85 S2-10686-Z1L1D S2-10688-Z1L1T S2-106BA-Z1P20 S2-1060C-Z1F0A S2-106R4-Z1Q6F-Z1Q6Q S2-106R4-Z1Q6J-Z1Q6Q S2-106RB-Z1Q6V-Z1Q6Q S2-107DR-Z1Y5B SU60-2230C-PU RC-TFSK3-868 NANO RFID POE 650201424G