

Short technical description: INT-RS v1.11 2012-11-06 and ETHM-1 v1.06 2013-02-22

The INT-RS module is an INTEGRA LCD-bus to RS-232 converter. It is dedicated to work with INTEGRA v1.11 2012-11-06 or newer. [The ETHM-1 module converts INTEGRA LCD-bus to Ethernet network - its protocol is described in *Function 2* chapter.](#)

To properly configure INT-RS module with INTEGRA panel, the following steps should be done:

- 1) Set the module address using DIP-switches 3..1 (3-MSB, 1-LSB). Allowed addresses are:
 - 0..3 - for INTEGRA 24 and 32 (i.e. DIP3='OFF')
 - 0..7 - for INTEGRA 64, 128 and 128-WRL
 E.g. to set the 6 address = 110_{bin}, the DIP-switches should be moved to: DIP3='ON', DIP2='ON', DIP1='OFF'.
- 2) Set the module function using DIP-switches 8..4 (8-MSB, 4-LSB). Possible values are 0 to 31 = 00000_{bin} to 11111_{bin}, but only the first few functions are present (see description below).
- 3) Connect INT-RS module to INTEGRA LCD bus using 4-wire cable.
- 4) Enter the service mode, go into the *Structure* menu, enter the *Hardware* submenu, select the *Identification* position and invoke the *LCD keypads id.* function.

For more details refer to INTEGRA manuals.

Function 0 - DIP-switches 8..4 = 00000

The module RS-232 port acts as INT-KLCD keypad serial port. For details refer to INT-KLCD eng.pdf document.

Function 1 - DIP-switches 8..4 = 00001

The module is used by INTEGRA panel for the monitoring purposes. To activate monitoring through INT-RS module, set the *Mon.ETHM-1* option in panel service settings.

If the system contains ETHM-1 modules and INT-RS modules with function 1, setting the *Mon.ETHM-1* option will allow to monitor events only by one of these modules - the one with the lowest address (e.g. the system contains modules: ETHM-1 address 5, INT-RS with function 0 address 1 and INT-RS with function 1 address 3 modules. Monitoring will be processed only through INT-RS with function 1 address 3 module).

RS-232 serial port of INT-RS module is configured as 4800/8/1/N. The DB9-male connector on the PCB makes use of the following lines:

- RX (pin 2) - serial input
- TX (pin 3) - serial output
- DTR (pin 4) - output - active when INT-RS module has communication with INTEGRA
- GND (pin 5) - signal ground
- DSR (pin 6) - input - the module can use this signal only to generate 'No external device DTR signal' event

The GND lines between INT-RS module and external device must be tied together.

The RX and TX lines should be swapped.

The DTR and DSR lines should also be swapped, if they are used.

In INTEGRA service mode it can be set that INT-RS module does or does not generate 'No external device DTR signal' event. It can also be set that INT-RS module does or does not check '?',#13 command (see below). If set, a monitoring trouble arises if external device does not ask INT-RS with '?',#13 question for a time longer than 32 seconds.

Communication between INT-RS module and external device is arranged in such a way that external device should ask INT-RS module to check if a new event is ready to be sent to a monitoring station. All data are ASCII chars ended with CR char (#13 = 0x0D byte). Data exchange is not time dependent.

Commands that INT-RS module understands:

- '?',#13 - a question if a new event is ready (2 bytes: 0x3F, 0x0D)
- '+',m,#13 - confirmation of sending event with marker m (3 bytes: 0x2B, m, 0x0D)
- '-',m,#13 - error sending event with marker m (3 bytes: 0x2D, m, 0x0D)

An answer is returned only on '?',#13 question. Possible answers are listed below:

- 'OK',#13 - no new event to send
- 'EN=m,s,iiii,cc'#13 - 4/2 event to send: m - event marker, s - monitoring station number ('1' or '2'),
iiii - event identifier, cc - event code
- 'EC=m,s,iiii,q,ccc,pp,nnn'#13 - Contact ID event to send: s - monitoring station number ('1' or '2'), m - event
marker, iiiii - event identifier, q and ccc - event code, pp - partition number,
nnn - source number

Events format and what events should be sent (4/2 or Contact ID) are to be set in INTEGRA service mode.

Event marker m is a char between 'a' and 'z'. The current event and its marker remain unchanged upon successive '?',#13 questions, until the event is confirmed by '+',m,#13 command from the external device or if INTEGRA time-out occurs (75 seconds). The next event, if ready, will be submitted by INT-RS module with succeeding value of marker m.

Function 2 - DIP-switches 8..4 = 00010

The module is used by INTEGRA panel for the integration purposes. [The same protocol is used by ETHM-1 module - see it below.](#)

RS-232 serial port of INT-RS module is configured as 19200/8/1/N. The DB9-male connector on the PCB makes use of the same lines as in the case of Function 1.

Communication between INT-RS module and external device is arranged in such a way that external device should ask (send command to) INT-RS module, and the module will answer immediately, if it is not marked otherwise.

Data exchange is no time dependent. The protocol uses the following frame structure (both ways - from and to INT-RS):

0xFE	0xFE	cmd	d1	d2	...	dn	crc.high	crc.low	0xFE	0x0D
------	------	-----	----	----	-----	----	----------	---------	------	------

The 16-bit crc sum is calculated as follows (see Appendix 4):

- 1) Set $crc := 0x147A$
- 2) For all successive bytes $b = cmd, d1, d2, ..., dn$ perform the crc update steps:
 - a) $crc := rl(crc)$ - rotate crc 1 bit left (msb=bit.15 shifts into lsb=bit.0 position)
 - b) $crc := crc \text{ xor } 0xFFFF$
 - c) $crc := crc + crc.high + b$, e.g. if $crc=0xFEDC$ and $b=0xA9$ then: $0xFEDC + 0xFE + 0xA9 = 0x0083$

The 0xFE byte is special value:

- 1) Two (or more) successive 0xFE mean frame synchronization - i.e. if device waits for any data-frame byte and it receives 0xFE, 0xFE - it should interrupt collecting the current frame and start waiting for cmd.
- 2) If device is waiting for the 1st byte of a frame (i.e. waiting for cmd), receiving 0xFE should not change it - device should be still waiting for cmd. So, cmd can not be 0xFE.
- 3) If any byte of the frame (i.e. cmd, d1, d2, ..., dn, crc.high, crc.low) to be sent is equal 0xFE, the following two bytes must be sent instead of single 0xFE byte: 0xFE, 0xF0. In such case only single 0xFE should be used to update crc.
- 4) If 0xFE, 0x0D are received, it means the frame is completed and it can be processed - i.e. check crc and analyze.
- 5) If other value after 0xFE is received - treat it as 0xFE, 0xFE (i.e. treat it as synchronization sequence).

If frame is corrupted (i.e. wrong crc sum or interrupted by 0xFE, 0xFE before completed) or cmd is not known or data length is not suitable for cmd - it is dropped and no answer is given back. **External device should act the same way.**

Exemplary frames: FE FE 09 D7 EB FE 0D FE FE 1C D7 FE F0 FE 0D

Part 1 - Reading INTEGRA state:

cmd	meaning	answer	
0x00	zones violation	0x00 + 16 bytes	(e.g. 06 20 00 00 00 00 00 00 00 00 00 00 00 00 00 80 - violated zones 2, 3, 14 and 128)
0x01	zones tamper	0x01 + 16 bytes	
0x02	zones alarm	0x02 + 16 bytes	
0x03	zones tamper alarm	0x03 + 16 bytes	
0x04	zones alarm memory	0x04 + 16 bytes	
0x05	zones tamper alarm memory	0x05 + 16 bytes	
0x06	zones bypass	0x06 + 16 bytes	
0x07	zones 'no violation' trouble	0x07 + 16 bytes	
0x08	zones 'long violation' trouble	0x08 + 16 bytes	
0x09	armed partitions (suppressed)	0x09 + 4 bytes	
0x0A	armed partitions (really)	0x0A + 4 bytes	
0x0B	partitions armed in mode 2	0x0B + 4 bytes	
0x0C	partitions armed in mode 3	0x0C + 4 bytes	
0x0D	partitions with 1st code entered	0x0D + 4 bytes	
0x0E	partitions entry time	0x0E + 4 bytes	
0x0F	partitions exit time >10s	0x0F + 4 bytes	
0x10	partitions exit time <10s	0x10 + 4 bytes	
0x11	partitions temporary blocked	0x11 + 4 bytes	
0x12	partitions blocked for guard round	0x12 + 4 bytes	
0x13	partitions alarm	0x13 + 4 bytes	
0x14	partitions fire alarm	0x14 + 4 bytes	
0x15	partitions alarm memory	0x15 + 4 bytes	
0x16	partitions fire alarm memory	0x16 + 4 bytes	
0x17	outputs state	0x17 + 16 bytes	
0x18	doors opened	0x18 + 8 bytes	
0x19	doors opened long	0x19 + 8 bytes	
0x1A	RTC and basic status bits	0x1A + 9 bytes (see description below)	

cmd	meaning	answer
0x1B	troubles part 1	0x1B + 47 bytes (see description below)
0x1C	troubles part 2	0x1C + 26 bytes (see description below)
0x1D	troubles part 3	0x1D + 60 bytes (see description below)
0x1E	troubles part 4	0x1E + 29 bytes (see description below)
0x1F	troubles part 5	0x1F + 31 bytes (see description below)
0x20	troubles memory part 1	0x20 + 47 bytes (see description below)
0x21	troubles memory part 2	0x21 + 39 bytes (see description below)
0x22	troubles memory part 3	0x22 + 60 bytes (see description below)
0x23	troubles memory part 4	0x23 + 29 bytes (see description below)
0x24	troubles memory part 5	0x24 + 48 bytes (see description below)
0x25	partitions with violated zones	0x25 + 4 bytes
0x26	zones isolate	0x26 + 16 bytes
0x27	partitions with verified alarms	0x27 + 4 bytes
0x28	zones masked	0x28 + 16 bytes (answer can be delayed up to 5s)
0x29	zones masked memory	0x29 + 16 bytes (answer can be delayed up to 5s)
0x7D	+1 byte - read zone 1..128 temperature	0x7D + 3 bytes (answer can be delayed up to 5s): 1 byte - zone number 1..128 2 bytes - temperature (high,low): 0x0000 = -55.0 °C 0x0001 = -54.5 °C 0x006E = 00.0 °C ... 0xFFFF = undetermined If requested zone is not temperature zone, answer will not be returned.
0x7E	INTEGRA version	0x7E + 14 bytes, e.g. for version 1.23 2012-05-27: 1 byte - INTEGRA type: 0, 1, 2, 3 = INTEGRA 24, 32, 64, 128 4 = INTEGRA 128-WRL SIM300 132 = INTEGRA 128-WRL LEON 66 = INTEGRA 64 PLUS 67 = INTEGRA 128 PLUS 11 bytes - '12320120527' 1 byte - language version (1=english, otherwise other language version) 1 byte - 255=settings stored in FLASH, otherwise not stored

cmd meaning

0x7F list of new states of above data

answer

0x7F + 5 bytes (each bit is set when new data is collected in corresponding command, each bit is cleared after reading the corresponding command):

- 1 byte - .0 - 1 = new data in 0x00 command
- .1 - 1 = new data in 0x01 command
- .2 - 1 = new data in 0x02 command
- .3 - 1 = new data in 0x03 command
- .4 - 1 = new data in 0x04 command
- .5 - 1 = new data in 0x05 command
- .6 - 1 = new data in 0x06 command
- .7 - 1 = new data in 0x07 command
- 1 byte - .0 - 1 = new data in 0x08 command
- .1 - 1 = new data in 0x09 command
- .2 - 1 = new data in 0x0A command
- .3 - 1 = new data in 0x0B command
- .4 - 1 = new data in 0x0C command
- .5 - 1 = new data in 0x0D command
- .6 - 1 = new data in 0x0E command
- .7 - 1 = new data in 0x0F command
- 1 byte - .0 - 1 = new data in 0x10 command
- .1 - 1 = new data in 0x11 command
- .2 - 1 = new data in 0x12 command
- .3 - 1 = new data in 0x13 command
- .4 - 1 = new data in 0x14 command
- .5 - 1 = new data in 0x15 command
- .6 - 1 = new data in 0x16 command
- .7 - 1 = new data in 0x17 command
- 1 byte - .0 - 1 = new data in 0x18 command
- .1 - 1 = new data in 0x19 command
- .2 - 1 = new data in 0x1A command
- .3 - 1 = new data in 0x1B command
- .4 - 1 = new data in 0x1C command
- .5 - 1 = new data in 0x1D command
- .6 - 1 = new data in 0x1E command
- .7 - 1 = new data in 0x1F command
- 1 byte - .0 - 1 = new data in 0x20 command
- .1 - 1 = new data in 0x21 command
- .2 - 1 = new data in 0x22 command
- .3 - 1 = new data in 0x23 command
- .4 - 1 = new data in 0x24 command
- .5 - 1 = new data in 0x25 command
- .6 - 1 = new data in 0x26 command
- .7 - 1 = new data in 0x27 command

Answers description:

RTC and basic status bits	- 7 bytes - time: YYYY-MM-DD hh:mm:ss - 0xYY, 0xYY, 0xMM, 0xDD, 0xhh, 0xmm, 0xss
	1 byte - .210 - day of the week (0=Monday, 1=Tuesday, ..., 6=Sunday)
	.7 - 1 = service mode
	.6 - 1 = troubles in the system (= flashing TROUBLE LED in keypad)
	1 byte - .7 - 1 = ACU-100 are present in the system
	.6 - 1 = INT-RX are present in the system
	.5 - 1 = troubles memory is set in INTEGRA panel
	.3210 - INTEGRA type: 0 = 24, 1 = 32, 2 = 64, 3 = 128, 4 = 128-WRL (to read detailed type use 0x7E command)
troubles part 1	- 16 bytes - troubles - technical zones
	8 bytes - expanders AC trouble
	8 bytes - expanders BATT trouble
	8 bytes - expanders NO BATT trouble
	3 bytes - system troubles (see description below)
	1 byte - CA-64 PTSA modules AC trouble
	1 byte - CA-64 PTSA modules BATT trouble
	1 byte - CA-64 PTSA modules NO BATT trouble
	1 byte - ETHM-1 monitoring trouble
troubles part 2	- 8 bytes - proximity card readers head A trouble
	8 bytes - proximity card readers head B trouble
	8 bytes - expanders supply output overload
	2 bytes - addressable zone expanders short circuit or jammed ACU-100 modules
troubles part 3	- 15 bytes - ACU-100 modules jam level
	15 bytes - radio devices with low battery
	15 bytes - radio devices with no communication
	15 bytes - radio outputs with no communication
troubles part 4	- 8 bytes - expanders with no communication
	8 bytes - switcherooed expanders
	1 byte - LCD keypads with no communication
	1 byte - switcherooed LCD keypads
	1 byte - ETHM-1 modules with no LAN cable / INT-RS modules with no DSR signal
	8 bytes - expanders tamper
	1 byte - LCD keypads tamper
	1 byte - LCD keypad initiation errors
	1 byte - auxiliary STM troubles (in INTEGRA PLUS, in others value of this field is 0)
troubles part 5	- 1 byte - low battery in masters key fobs
	30 bytes - low battery in users key fobs
troubles memory part 1	- 47 bytes - memory of troubles part 1
troubles memory part 2	- 26 bytes - memory of troubles part 2
	1 byte - LCD keypads restart memory
	8 bytes - expanders restart memory
	2 bytes - GSM trouble code (high,low)
	2 bytes - GSM trouble code memory (high,low)
troubles memory part 3	- 60 bytes - memory of troubles part 3
troubles memory part 4	- 29 bytes - memory of troubles part 4
troubles memory part 5	- 16 bytes - long zones violation memory
	16 bytes - no zones violation memory
	16 bytes - zones tamper memory

System troubles:	1st byte	.0	- OUT1 trouble
		.1	- OUT2 trouble
		.2	- OUT3 trouble
		.3	- OUT4 trouble
		.4	- +KPD trouble
		.5	- +EX1 or +EX2 trouble
		.6	- BATT trouble
		.7	- AC trouble
	2nd byte	.0	- DT1 trouble
		.1	- DT2 trouble
		.2	- DTM trouble
		.3	- RTC trouble
		.4	- no DTR signal
		.5	- no BATT present
		.6	- external modem initialization trouble
		.7	- external model command (ATE0V1Q0H0S0=0) trouble
	3rd byte	.0	- no voltage on telephone line (INTEGRA 24, 32, 64 and 128)
		.0	- auxiliary ST processor trouble (INTEGRA 128-WRL)
		.1	- bad signal on telephone line
		.2	- no signal on telephone line
		.3	- monitoring to station 1 trouble
		.4	- monitoring to station 2 trouble
		.5	- EEPROM or access to RTC trouble
		.6	- RAM memory trouble
		.7	- INTEGRA main panel restart memory

Part 2 - INTEGRA control:

0x80	arm in mode 0:	+ 8 bytes - user code (with prefix, if required by INTEGRA), e.g.: if code is '1234', no prefixes: 0x12, 0x34, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF if code is '1234', prefix is '97': 0x97, 0x12, 0x34, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF + 4 bytes - partition list, e.g.: if partition 1, 2, and 29 have to be armed: 0x03, 0x00, 0x00, 0x10 If function is accepted, function result can be checked by observe the system state
0x81	arm in mode 1	data structure as above If function is accepted, function result can be checked by observe the system state
0x82	arm in mode 2	data structure as above If function is accepted, function result can be checked by observe the system state
0x83	arm in mode 3	data structure as above If function is accepted, function result can be checked by observe the system state
0x84	disarm	data structure as above If function is accepted, function result can be checked by observe the system state
0x85	clear alarm	data structure as above If function is accepted, function result can be checked by observe the system state
0x86	zones bypass	+ 8 bytes - user code - see example for 0x80 + 16 bytes - zone list, e.g.: if zone 1, 3, 62 and 120 have to be bypassed: 0x05, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x00 If function is accepted, function result can be checked by observe the system state
0x87	zones unbypass	data structure as above If function is accepted, function result can be checked by observe the system state
0x88	outputs on	+ 8 bytes - user code - see example for 0x80 + 16 bytes - output list - see example for 0x86 If function is accepted, function result can be checked by observe the system state
0x89	outputs off	data structure as above If function is accepted, function result can be checked by observe the system state

- 0x8A open door + 8 bytes - user code - *see example for 0x80*
+ 16 bytes - output list - *see example for 0x86* - outputs of a 101 type can be 'opened'
+ 8 bytes - expander list, *e.g.*:
if expander address 4 and 63 doors have to be opened:
0x10, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80
If function is accepted, function result can be checked by observe the system state
- 0x8B clear trouble mem. + 8 bytes - user code - *see example for 0x80*
If function is accepted, function result can be checked by observe the system state
- 0x8C read event + 3 bytes - last event index. To start reading event log call this function with these 3 bytes equal 0xFF - the last event will be returned. To read previous event, call this function with event index returned by this function and so on.
Function result - 15 bytes in the following format:

- 1 byte - 0x8C
8 bytes - event record - see the table below
3 bytes - event index
3 bytes - event index used to call the function

Bit:	.7	.6	.5	.4	.3	.2	.1	.0
1st byte	Y	Y	Z	E	S2	S2	S1	S1
2nd byte	K	K	K	D	D	D	D	D
3rd byte	M	M	M	M	T	T	T	T
4th byte	t	t	t	t	t	t	t	t
5th byte	P	P	P	P	P	R	C	C
6th byte	c	c	c	c	c	c	c	c
7th byte	n	n	n	n	n	n	n	n
8th byte	S	S	S	u	u	u	u	u

- YY - year marker (i.e. YEAR mod 4, e.g. 2009 mod 4 = 1, 2010 mod 4 = 2)
Z - 1 = record not empty
E - 1 = event present (normally ZE should be both 00 or 11)
S1, S2 - status of monitoring to station 1 and 2, respectively:
00 - new event, not processed by monitoring service
01 - event sent
10 - should not occur
11 - event not monitored
KKK - event class:
000 - zone and tamper alarms
001 - partition and expander alarms
010 - arming, disarming, alarm clearing
011 - zone bypasses and unbypasses
100 - access control
101 - troubles
110 - user functions
111 - system events
DDDDD - day of the month (1..31)
MMMM - month (1..12)
TTTTTTTT - time in minutes (e.g. 17:53 = 17*60+53 = 1073)
PPPPP - partition number
R - 1 = restore
CCCCCCCC - event code - use command 0x8F to convert to text (or see Appendix 1 for event list)
nnnnnnnn - source number (e.g. zone number, user number) (see Appendix 1)
SSS - object number (0..7)
uuuuu - user control number - this number is increased everytime the user is created (ie. it will be changed after erase and create the user). This number is important only in those events which have the user in its description (e.g. arming by user; but e.g. zone alarm - not)

- 0x8D enter 1st code + 8 bytes - user code - *see example for 0x80*
+ 4 bytes - partition list - *see example for 0x80*
+ 2 bytes - 1st code validity period (low,high) in seconds
+ 1 byte - action:
0 - enter 1st code for arm
1 - enter 1st code for disarm
2 - cancel 1st code (validity period inessential)
If function is accepted, function result can be checked by observe the system state
- 0x8E set RTC clock + 8 bytes - user code - *see example for 0x80*
+ 14 bytes - time and date to set (14 ASCII chars: yyyyymmddhhmmss)

0x8F get event text + 2 bytes (high,low) - decode event code to text description:
 .15 - 0=short, 1=long text description
 11 lsb - event code
 Function result - 22 or 52 bytes (depends on selection of short/long format) as follows:
 1 byte - 0x8F
 2 bytes - two bytes used to call this function
 1 byte - kind of long description (see Appendix 2)
 2 bytes - kind of short description (see Appendix 3)
 16 or 46 bytes - event text

0x90 zones isolate *data structure as in 0x86*
 If function is accepted, function result can be checked by observe the system state

0x91 outputs switch *data structure as in 0x88*
 If function is accepted, function result can be checked by observe the system state

Part 3 - users management:

General numbering scheme in INTEGRA is as follow:

- 1..240 - number of user (max. value depends on INTEGRA type)
- 241..248 - number of master (max. value depends on INTEGRA type)
- 255 - number of service

0xE0 read self-info + 4/8 bytes - if 4 bytes - user code only, e.g.:
 if code '1234': 0x12, 0x34, 0xFF, 0xFF
 if 8 bytes - recommended usage - prefix + user code, e.g.:
 if prefix '987', code '1234': 0x98, 0x71, 0x23, 0x4F, 0xFF, 0xFF, 0xFF, 0xFF
 if no prefix, code '1234': 0x12, 0x34, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF

Function result - 30 bytes:

- 1 byte - 0xE0
- 1 byte - user number - see above numbering scheme
- 2 bytes - if user - user telephone code
 if master - 0x00, 0x00
 if service - 1st byte - existing masters, 2nd byte - 0x00
- 4 bytes - user partitions
- 1 byte - XYI0TTTT: X - user did not changed his code yet
 Y - user code is recognized by other user
 I - user right - zones isolating
 TTTT - user type:

- 0 - normal
- 1 - single
- 2 - time renewable
- 3 - time not renewable
- 4 - duress
- 5 - mono outputs
- 6 - bi outputs
- 7 - partitions temporary blocking
- 8 - access to cash machine
- 9 - guard
- 10 - schedule

- 1 byte - user time
- 3 bytes - user rights:
 - 1st byte - .0 - arming
 .1 - disarming
 .2 - alarm clearing in own partitions
 .3 - alarm clearing in own object
 .4 - alarm clearing in whole system
 .5 - arm deferring
 .6 - code changing
 .7 - users editing
 - 2nd byte - .0 - zones bypassing
 .1 - clock setting
 .2 - troubles viewing
 .3 - events viewing
 .4 - zones resetting
 .5 - options changing
 .6 - tests
 .7 - downloading
 - 3rd byte - .0 - can always disarm (i.e. even if armed by other user)
 .1 - voice messaging clearing
 .2 - GuardX using
 .3 - access to temporary blocked partitions
 .4 - entering 1st code
 .5 - entering 2nd code
 .6 - outputs control
 .7 - clearing latched outputs
- 16 bytes - user name
- 1 byte - if user - object number (0..7)
 if master - object number (0..7)
 if service - 0xFF

To use the read self-info command, the user must have the 'GuardX using' right set active, otherwise the error 'requesting user code not found' will be returned.

0xE1	read user	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - user number to read (1..240 - user, 241..248 - master) Function result - 29 bytes: 1 byte - 0xE1 1 byte - user number: 1..240 - user 241..248 - master 255 - service 4 bytes - user partitions 1 byte - XY00TTTT: TTTT - user type: 0 - normal 1 - single 2 - time renewable 3 - time not renewable 4 - duress 5 - mono outputs 6 - bi outputs 7 - partitions temporary blocking 8 - access to cash machine 9 - guard 10 - schedule - X - 1=user did not change own code after it was created - Y - 1=other user tried to change own code to this user code 1 byte - user time 1 byte - user time - temporary value - valid only for schedule user 3 bytes - user rights - <i>see description for 0xE0</i> 16 bytes - user name 1 byte - if user - object number (0..7) - if master - object number (0..7) - if service - 0xFF
0xE2	read users list	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - user number (1..248) which users list is to be read Function result - 62 bytes: 1 byte - 0xE2 1 byte - user number 30 bytes - list of all existing users 30 bytes - list of users that can be edited by this user
0xE3	read user locks	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - user number (1..248) which locks are to be read Function result - 10 bytes: 1 byte - 0xE3 1 byte - user number 8 bytes - list of user locks
0xE4	write user locks	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - user number (1..248) which locks are to be written + 8 bytes - list of user locks
0xE5	remove user	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - user number (1..248) to remove
0xE6	create user	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - user number (1..248) to create, 255 - auto + 4 bytes - user-to-create code + 2 bytes - user-to-create telephone code - 4 x BCD or 0xFFFF + 4 bytes - user-to-create partitions + 1 byte - user-to-create type + 1 byte - user-to-create time + 1 byte - user-to-create temporary time - <i>valid only for schedule user</i> + 1 byte - user-to-create 1st byte of rights + 1 byte - user-to-create 2nd byte of rights + 1 byte - user-to-create 3rd byte of rights + 16 bytes - user-to-create name + 1 byte - user-to-create object - valid only if service is the creator

0xE7 change user

- + 4/8 bytes - user code (see example for 0xE0)
- + 1 byte - user number (1..248) to change
- + 4 bytes - user-to-change code - *will not be changed if equal 0xFFFFFFFF*
- + 2 bytes - user-to-change telephone code - *will not be changed if equal 0xFFFF*
- + 4 bytes - user-to-change partitions
- + 1 byte - user-to-change type
- + 1 byte - user-to-change time
- + 1 byte - user-to-change temporary time - *valid only for schedule user*
- + 1 byte - user-to-change 1st byte of rights
- + 1 byte - user-to-change 2nd byte of rights
- + 1 byte - user-to-change 3rd byte of rights
- + 16 bytes - user-to-change name

In above commands you can set user type as follows:

0. Normal
1. Single
2. Time renewable
3. Time not renewable
4. Duress
5. Mono outputs
6. Bi outputs
7. Partition temporary blocking
8. Access to cash machine
9. Guard
10. Schedule

For users of the 2 and 3 types in the field '+ 1 byte - user-to-create/change time' you should give how many days the user should exist.
 For users of the 10 type in the field '+ 1 byte - user-to-create/change time' you should give user schedule number (1..8), and in the field '+ 1 byte - user-to-create/change temporary time' you give how many days the user should exist (0..254 - 0..254 days, 255 - infinite).
 For users of the 7 type the field '+ 1 byte - user-to-create/change time' stands for the blocking time (1..109 minutes).
 For other user types these two fields are not important (give 0 as filling).

0xE8 user DALLAS/proximity card/key fob managing:

Read card/DALLAS list:	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - '0' (ASCII 48 char) - Function result - 64 bytes: 1 byte - 0xE8 1 byte - '0' 31 bytes - proximity card list 31 bytes - DALLAS list
Read user proximity card:	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - '1' (ASCII 49 char) + 1 byte - user number (1..248) which proximity card to read - Function result - 8 bytes: 1 byte - 0xE8 1 byte - '1' 1 byte - user number 5 bytes - proximity card number
Write user proximity card:	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - '2' (ASCII 50 char) + 1 byte - user number (1..248) which proximity card to write + 5 bytes - proximity card number
Read user DALLAS:	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - '3' (ASCII 51 char) + 1 byte - user number (1..248) which DALLAS to read - Function result - 9 bytes: 1 byte - 0xE8 1 byte - '3' 1 byte - user number 6 bytes - DALLAS number
Write user DALLAS:	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - '4' (ASCII 52 char) + 1 byte - user number (1..248) which DALLAS to write + 6 bytes - DALLAS number
Read user INT-RX key fob:	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - '7' (ASCII 55 char) + 1 byte - user number (1..248) which INT-RX key fob to read - Function result - 14 bytes: 1 byte - 0xE8 1 byte - '7' 1 byte - user number 4 bytes - INT-RX key fob 28-bit serial number (high..low) 6 bytes - settings of key presses (zones number to violate in INTEGRA panel) 1 byte - bit list of keys that generate no events
Write user INT-RX key fob:	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - '8' (ASCII 56 char) + 1 byte - user number (1..248) which INT-RX key fob to write + 4 bytes - INT-RX key fob 28-bit serial number (high..low) + 6 bytes - settings of key presses (zones number to violate in INTEGRA panel) + 1 byte - bit list of keys that generate no events
Read user ABAX key fob:	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - '9' (ASCII 57 char) + 1 byte - user number (1..248) which ABAX key fob to read - Function result - 14 bytes: 1 byte - 0xE8 1 byte - '9' 1 byte - user number 3 bytes - ABAX key fob 20-bit serial number (high..low) 6 bytes - settings of key presses (zones number to violate in INTEGRA panel) 1 byte - bit list of keys that generate no events 1 byte - bit list (max. three '1's) of INTEGRA output status used as acknowledge
Write user ABAX key fob:	+ 4/8 bytes - user code (see example for 0xE0) + 1 byte - 'A' (ASCII 41 char) + 1 byte - user number (1..248) which ABAX key fob to write + 3 bytes - ABAX key fob 20-bit serial number (high..low) + 6 bytes - settings of key presses (zones number to violate in INTEGRA panel) + 1 byte - bit list of keys that generate no events + 1 byte - bit list (max. three '1's) of INTEGRA output status used as acknowledge

Function can give result as below in a case of command that does not return result or in a case of an error:

- 1 byte - 0xE8
- 1 byte - '?'
- 1 byte - repeated command (i.e. '0', '1', '2', '3', '4', '7', '8', '9' or 'A')
- 1 byte - user number (can be inessential in some cases, e.g. in a case of wrong command)
- 1 byte - confirmation or error:
 - 0x00 - ok
 - 0x01 - unknown user code
 - 0x02 - no rights to perform action (on selected user)
 - 0x08 - unknown command
 - 0x8? - other errors

0xEE read device name + 1 byte - device type to read:

- 0 - partition (1..32)
- 1 - zone (1..128)
- 2 - user (1..255)
- 3 - expander/LCD (129..192 - expander, 193..210 - LCD)
- 4 - output (1..128)
- 5 - zone (1..128) with partition assignment (*)

+ 1 byte - device number to read

Function result - 20 bytes (* or 21 bytes):

- 1 byte - 0xEE
- 1 byte - device type - *see above*
- 1 byte - device number - *see above*
- 1 byte - device type/function:
 - if partition - partition type - *see e.g. DloadX for partition types list*
 - if zone - zone reaction - *see e.g. DloadX for zone reactions list*
 - if user - 0
 - if expander - expander type:
 - 1 - CA-64 PP
 - 2 - CA-64 E
 - 3 - CA-64 O
 - 4 - CA-64 EPS
 - 5 - CA-64 OPS
 - 6 - CA-64 ADR
 - 7 - INT-ORS
 - 8 - INT-S/SK
 - 9 - INT-SZ/SZK
 - 10 - CA-64 DR
 - 11 - CA-64 SR
 - 12 - ACU-100
 - 13 - INT-IORS
 - 14 - CA-64 Ei
 - 15 - CA-64 SM
 - 16 - INT-AV
 - 17 - INT-IT
 - 18 - CA-64 EPSi
 - 19 - INT-SCR
 - 20 - INT-ENT
 - 21 - INT-RX
 - 22 - INT-TXM
 - 23 - INT-VG
 - 24 - INT-KNX
 - if LCD - 'LCD' type:
 - 1 - INT-KLCD
 - 2 - INT-KLCDR
 - 3 - CA-64 PTSA
 - 4 - INT-RS
 - 5 - ETHM-1
 - if output - output function - *see e.g. DloadX for output functions list*
- 16 bytes - device name
- 1 byte - partition number (1..32) the zone is assigned to (this 21st byte appears only if it is device type to read number 5 (*))

INT-RS module returns an answer on **every** request - function result or 0xEF result (described below), so after sending any request to the module please wait for answer before sending next request (or give the module e.g. 3 seconds time-out).

0xEF result + 1 byte - result code:

- 0x00 - ok
- 0x01 - requesting user code not found
- 0x02 - no access
- 0x03 - selected user does not exist
- 0x04 - selected user already exists
- 0x05 - wrong code or code already exists
- 0x06 - telephone code already exists
- 0x08 - other error
- 0x8? - other errors
- 0xFF - function accepted (i.e. data length and crc ok), will be processed

Appendix 1 - event list

Full event list that is possible to generate by INTEGRA v1.11 2012-11-06 (older INTEGRA can generate subset of these events):

- the first column is the event code (CCcccccccc)
- the second column is new/restore (R)
- the third column is kind of long description (see Appendix 2)
- the fourth column is event text description

1,0,6,	'Voice messaging aborted	:
2,0,3,	'Change of user access code	:
2,1,3,	'Change of user access code	:
3,0,6,	'Change of user access code	:
4,0,6,	'Zones bypasses	:
5,0,6,	'Zones reset	:
6,0,6,	'Change of options	:
7,0,6,	'Permission for service access	:
7,1,6,	'Permission for service access removed	:
8,0,6,	'Addition of user	:
9,0,6,	'New user	:
10,0,6,	'Edition of user	:
11,0,6,	'User changed	:
12,0,6,	'Removal of user	:
13,0,6,	'User removed	:
14,0,6,	'Recognition of user access code	:
15,0,6,	'User access code recognized	:
16,0,6,	'Addition of master	:
17,0,6,	'Edition of master	:
18,0,6,	'Removal of master	:
19,0,4,	'RS-downloading started	:
19,1,4,	'RS-downloading finished	:
20,0,6,	'TEL-downloading started	:
21,0,6,	'Monitoring station 1A test	:
22,0,6,	'Monitoring station 1B test	:
23,0,6,	'Monitoring station 2A test	:
24,0,6,	'Monitoring station 2B test	:
25,0,4,	'Service mode taken over	:
26,0,2,	'Access to cash machine granted	:
27,0,3,	'Recognition of user access code	:
27,1,3,	'Recognition of user access code	:
28,0,3,	'User access code recognized	:
28,1,3,	'User access code recognized	:
29,0,7,	'Automatically removed temporal user	:
30,0,0,	'Service access automatically blocked	:
31,0,0,	'Main panel software updated	:
32,0,4,	'System settings stored in FLASH memory	:
33,0,0,	'Starter started	:
34,0,0,	'Starter started from RESET jumper	:
35,0,6,	'Zones test function started	:
36,0,7,	'Removal of single user	:
37,0,2,	'First access code entered	:
38,0,3,	'Voice messaging aborted	:
38,1,3,	'Voice messaging aborted	:
39,0,1,	'Vibration sensors test ok	:
40,0,6,	'Change of prefix	:
41,0,0,	'Change of winter time to summer time	:
42,0,0,	'Change of summer time to winter time	:
43,0,6,	'Guard round	:
44,0,5,	'First access code expired	:
45,0,2,	'First access code cancelled	:
46,0,7,	'Remote (telephone) control started	:
46,1,7,	'Remote (telephone) control finished	:
47,0,10,	'Remote switch turned on	:
47,1,10,	'Remote switch turned off	:
48,0,30,	'TCP/IP connection started (Internet)	:
48,1,30,	'TCP/IP connection finished (Internet)	:
49,0,30,	'TCP/IP connection failed (Internet)	:
50,0,31,	'IP address	:
51,0,4,	'Invalidation of system settings in FLASH	:
52,0,6,	'Service note cleared	:
53,0,1,	'Vibration sensors test interrupted	:
54,0,30,	'TCP/IP connection started (DloadX)	:
54,1,30,	'TCP/IP connection finished (DloadX)	:
55,0,30,	'TCP/IP connection failed (DloadX)	:
56,0,30,	'TCP/IP connection started (GuardX)	:
56,1,30,	'TCP/IP connection finished (GuardX)	:
57,0,30,	'TCP/IP connection failed (GuardX)	:
58,0,30,	'TCP/IP connection started (GSM socket)	:
58,1,30,	'TCP/IP connection finished (GSM socket)	:
59,0,30,	'TCP/IP connection failed (GSM socket)	:
60,0,30,	'TCP/IP connection started (GSM http)	:
60,1,30,	'TCP/IP connection finished (GSM http)	:
61,0,30,	'TCP/IP connection failed (GSM http)	:
62,0,6,	'User access	:
63,0,6,	'User exit	:

```

64,0, 4, 'Keypad temporary blocked
65,0, 4, 'Reader temporary blocked
66,0, 1, 'Arming in "Stay" mode
67,0, 1, 'Armin in "Stay, delay=0" mode
68,0, 0, 'System real-time clock set
69,0, 6, 'Troubles memory cleared
70,0, 6, 'User logged in
71,0, 6, 'User logged out
72,0, 6, 'Door opened from LCD keypad
73,0,13, 'Door opened
74,0, 6, 'System restored
75,0, 0, 'ETHM/GPRS key changed
76,0, 6, 'Messaging test started
77,0, 1, 'Alarm monitoring delay
78,0, 1, 'Network cable unplugged
78,1, 1, 'Network cable ok
79,0, 9, 'Messaging trouble
80,0, 9, 'Messaging doubtful
81,0, 9, 'Messaging ok
82,0, 9, 'Messaging confirmed
83,0, 1, '3 wrong access codes
84,0, 1, 'Alarm - proximity card reader tamper
84,1, 1, 'Proximity card reader restore
85,0, 4, 'Unauthorised door opening
86,0, 3, 'User exit
86,1, 3, 'User exit
87,0, 2, 'Partition temporary blocked
88,0, 0, 'GSM module trouble
88,1, 0, 'GSM module ok
89,0, 4, 'Long opened door
89,1, 4, 'Long opened door closed
90,0, 0, 'Download suspended
91,0, 0, 'Download started
92,0, 1, 'Alarm - module tamper (verification error)
92,1, 1, 'Module tamper restore (verification ok)
93,0, 1, 'Alarm - module tamper (lack of presence)
93,1, 1, 'Module tamper restore (presence ok)
94,0, 1, 'Alarm - module tamper (TMP input)
94,1, 1, 'Module tamper restore (TMP input)
95,0,12, 'Output overload
95,1,12, 'Output overload restore
96,0,12, 'No output load
96,1,12, 'Output load present
97,0, 1, 'Long zone violation
97,1, 1, 'Long zone violation restore
98,0, 1, 'No zone violation
98,1, 1, 'No zone violation restore
99,0, 1, 'Zone violation
99,1, 1, 'Zone restore
100,0, 1, 'Medical request (button)
100,1, 1, 'Release of medical request button
101,0, 1, 'Medical request (remote)
101,1, 1, 'Remote medical request restore
110,0, 1, 'Fire alarm
110,1, 1, 'Fire alarm zone restore
111,0, 1, 'Fire alarm (smoke detector)
111,1, 1, 'Smoke detector zone restore
112,0, 1, 'Fire alarm (combustion)
112,1, 1, 'Combustion zone restore
113,0, 1, 'Fire alarm (water flow)
113,1, 1, 'Water flow detection restore
114,0, 1, 'Fire alarm (temperature sensor)
114,1, 1, 'Temperature sensor zone restore
115,0, 1, 'Fire alarm (button)
115,1, 1, 'Release of fire alarm button
116,0, 1, 'Fire alarm (duct)
116,1, 1, 'Duct zone restore
117,0, 1, 'Fire alarm (flames detected)
117,1, 1, 'Flames detection zone restore
120,0, 1, 'PANIC alarm (keypad)
121,0, 2, 'DURESS alarm
122,0, 1, 'Silent PANIC alarm
122,1, 1, 'Silent panic alarm zone restore
123,0, 1, 'Audible PANIC alarm
123,1, 1, 'Audible panic alarm zone restore
126,0, 5, 'Alarm - no guard
130,0, 1, 'Burglary alarm
130,1, 1, 'Zone restore
131,0, 1, 'Alarm (perimeter zone)
131,1, 1, 'Perimeter zone restore
132,0, 1, 'Alarm (interior zone)
132,1, 1, 'Interior zone restore
133,0, 1, 'Alarm (24h burglary zone)
133,1, 1, '24h burglary zone restore
134,0, 1, 'Alarm (entry/exit zone)
134,1, 1, 'Entry/exit zone restore

```

135,0,1,	'Alarm (day/night zone)	:
135,1,1,	'Day/night zone restore	:
136,0,1,	'Alarm (exterior zone)	:
136,1,1,	'Exterior zone restore	:
137,0,1,	'Alarm (tamper perimeter)	:
137,1,1,	'Tamper perimeter zone restore	:
139,0,1,	'Verified alarm	:
143,0,11,	'Alarm - communication bus trouble	:
143,1,11,	'Communication bus ok	:
144,0,1,	'Alarm (zone tamper)	:
144,1,1,	'Zone tamper restore	:
145,0,1,	'Alarm (module tamper)	:
145,1,1,	'Module tamper restore	:
150,0,1,	'Alarm (24h no burglary zone)	:
150,1,1,	'24h no burglary zone restore	:
151,0,1,	'Alarm (gas detector)	:
151,1,1,	'Gas detection zone restore	:
152,0,1,	'Alarm (refrigeration)	:
152,1,1,	'Refrigeration zone restore	:
153,0,1,	'Alarm (heat loss)	:
153,1,1,	'Heat loss zone restore	:
154,0,1,	'Alarm (water leak)	:
154,1,1,	'Water leak zone restore	:
155,0,1,	'Alarm (protection loop break)	:
155,1,1,	'Protection loop break zone restore	:
156,0,1,	'Alarm (day/night zone tamper)	:
156,1,1,	'Day/night zone tamper restore	:
157,0,1,	'Alarm (low gas level)	:
157,1,1,	'Low gas level zone restore	:
158,0,1,	'Alarm (high temperature)	:
158,1,1,	'High temperature zone restore	:
159,0,1,	'Alarm (low temperature)	:
159,1,1,	'Low temperature zone restore	:
161,0,1,	'Alarm (no air flow)	:
161,1,1,	'No air flow zone restore	:
162,0,1,	'Alarm (carbon monoxide detected)	:
162,1,1,	'Restore of carbon monoxide (CO) detection	:
163,0,1,	'Alarm (tank level)	:
163,1,1,	'Restore of tank level	:
200,0,1,	'Alarm (fire protection loop)	:
200,1,1,	'Fire protection loop zone restore	:
201,0,1,	'Alarm (low water pressure)	:
201,1,1,	'Low water pressure zone restore	:
202,0,1,	'Alarm (low CO2 pressure)	:
202,1,1,	'Low CO2 pressure zone restore	:
203,0,1,	'Alarm (valve sensor)	:
203,1,1,	'Valve sensor zone restore	:
204,0,1,	'Alarm (low water level)	:
204,1,1,	'Low water level zone restore	:
205,0,1,	'Alarm (pump activated)	:
205,1,1,	'Pump stopped	:
206,0,1,	'Alarm (pump trouble)	:
206,1,1,	'Pump ok	:
220,0,1,	'Keybox open	:
220,1,1,	'Keybox restore	:
300,0,4,	'System module trouble	:
300,1,4,	'System module ok	:
301,0,4,	'AC supply trouble	:
301,1,4,	'AC supply ok	:
302,0,4,	'Low battery voltage	:
302,1,4,	'Battery ok	:
303,0,0,	'RAM memory error	:
305,0,4,	'Main panel restart	:
306,0,0,	'Main panel settings reset	:
306,1,0,	'System settings restored from FLASH memory	:
312,0,12,	'Supply output overload	:
312,1,12,	'Supply output overload restore	:
330,0,8,	'Proximity card reader trouble	:
330,1,8,	'Proximity card reader ok	:
333,0,11,	'Communication bus trouble	:
333,1,11,	'Communication bus ok	:
339,0,4,	'Module restart	:
344,0,1,	'Receiver jam detected	:
344,1,1,	'Receiver jam ended	:
350,0,0,	'Transmission to monitoring station trouble	:
350,1,0,	'Transmission to monitoring station ok	:
351,0,0,	'Telephone line troubles	:
351,1,0,	'Telephone line ok	:
370,0,1,	'Alarm (auxiliary zone perimeter tamper)	:
370,1,1,	'Auxiliary zone perimeter tamper restore	:
373,0,1,	'Alarm (fire sensor tamper)	:
373,1,1,	'Fire sensor tamper restore	:
380,0,1,	'Zone trouble (masking)	:
380,1,1,	'Zone ok (masking)	:
381,0,32,	'Radio connection troubles	:
381,1,32,	'Radio connection ok	:

```

383,0, 1, 'Alarm (zone tamper)
383,1, 1, 'Zone tamper restore
384,0,32, 'Low voltage on radio zone battery
384,1,32, 'Voltage on radio zone battery ok
388,0, 1, 'Zone trouble (masking)
388,1, 1, 'Zone ok (masking)
400,0, 2, 'Disarm
400,1, 2, 'Arm
401,0, 2, 'Disarm by user
401,1, 2, 'Arm by user
402,0, 2, 'Group disarm
402,1, 2, 'Group arm
403,0,15, 'Auto-disarm
403,1,15, 'Auto-arm
404,0, 2, 'Late disarm by user
404,1, 2, 'Late arm by user
405,0, 2, 'Deferred disarm by user
405,1, 2, 'Deferred arm by user
406,0, 2, 'Alarm cleared
407,0, 2, 'Remote disarm
407,1, 2, 'Remote arm
408,1, 1, 'Quick arm
409,0, 1, 'Disarm by zone
409,1, 1, 'Arm by zone
411,0, 0, 'Callback made
412,0, 0, 'Download successfully finished
413,0, 0, 'Unsuccessful remote download attempt
421,0, 3, 'Access denied
421,1, 3, 'Access denied
422,0, 3, 'User access
422,1, 3, 'User access
423,0, 1, 'Alarm - armed partition door opened
441,1, 2, 'Arm (STAY mode)
442,1, 1, 'Arm by zone (STAY mode)
454,0, 2, 'Arming failed
458,0, 2, 'Delay activation time started
461,0, 1, 'Alarm (3 wrong access codes)
462,0, 3, 'Guard round
462,1, 3, 'Guard round
570,0, 1, 'Zone bypass
570,1, 1, 'Zone unbypass
571,0, 1, 'Fire zone bypass
571,1, 1, 'Fire zone unbypass
572,0, 1, '24h zone bypass
572,1, 1, '24h zone unbypass
573,0, 1, 'Burglary zone bypass
573,1, 1, 'Burglary zone unbypass
574,0, 1, 'Group zone bypass
574,1, 1, 'Group zone unbypass
575,0, 1, 'Zone auto-bypassed (violations)
575,1, 1, 'Zone auto-unbypassed (violations)
601,0, 6, 'Manual transmission test
602,0, 0, 'Transmission test
604,0, 2, 'Fire/technical zones test
604,1, 5, 'End of fire/technical zones test
607,0, 2, 'Burglary zones test
607,1, 5, 'End of burglary zones test
611,0, 1, 'Zone test ok
612,0, 1, 'Zone not tested
613,0, 1, 'Burglary zone test ok
614,0, 1, 'Fire zone test ok
615,0, 1, 'Panic zone test ok
621,0, 0, 'Reset of event log
622,0, 0, 'Event log 50% full
623,0, 0, 'Event log 90% full
625,0, 6, 'Setting system real-time clock
625,1, 0, 'System real-time clock trouble
627,0, 4, 'Service mode started
628,0, 4, 'Service mode finished
985,0,15, 'Exit time started
986,0, 1, 'Warning alarm
987,0, 2, 'Warning alarm cleared
988,0, 1, 'Arming aborted
989,0, 7, 'User logged in (INT-VG)
989,1, 7, 'User logged out (INT-VG)
990,0, 4, 'No connection with KNX system
990,1, 4, 'Connection with KNX system ok
991,0, 1, 'Zone auto-bypassed (tamper violations)
991,1, 1, 'Zone auto-unbypassed (tamper violations)
992,0, 6, 'Confirmed troubles
993,0, 6, 'Confirmed use of RX key fob with low battery
994,0, 6, 'Confirmed use of ABAX key fob with low battery
995,0, 3, 'Remote RX key fob with low battery used
995,1, 3, 'Remote RX key fob with low battery used
996,0, 3, 'Remote ABAX key fob with low battery used
996,1, 3, 'Remote ABAX key fob with low battery used

```

```

997,0, 4, 'Long transmitter busy state
997,1, 4, 'Restore of long transmitter busy state
998,0, 0, 'Transmission test (station 1)
999,0, 0, 'Transmission test (station 2)
1000,0, 1, 'Trouble (zone)
1000,1, 1, 'Trouble restore (zone)
1001,0, 2, 'Forced arming
1002,0, 4, 'No network (PING test)
1002,1, 4, 'Network ok (PING test)
1003,0, 2, 'Arming aborted
1004,0, 0, 'GPRS-downloading started from GSM module
1005,0, 6, 'ETHM-1-downloading started
1006,0, 4, 'Current battery test - absent/low voltage
1006,1, 4, 'Current battery test - ok
1007,0, 1, 'Exit time started
1008,0, 2, 'Exit time started
1009,0,14, 'SMS control - begin
1009,1,14, 'SMS control - end
1010,0,14, 'SMS with no control received
1011,0,14, 'SMS from unauthorized telephone received
1012,0, 6, 'CSD-downloading started
1013,0, 6, 'GPRS-downloading started
1014,0, 4, 'No signal on DSR input
1014,1, 4, 'Signal on DSR input ok
1015,0, 4, 'Time server error
1015,1, 4, 'Time server ok
1016,0, 6, 'Time synchronization started
1017,0, 9, 'SMS messaging ok
1018,0, 9, 'SMS messaging failed
1019,0, 3, 'Remote key fob used
1019,1, 3, 'Remote key fob used
1020,0, 1, 'LCD/PTSA/ETHM-1 initiation error
1021,0, 1, 'LCD/PTSA/ETHM-1 initiation ok
1022,0, 0, 'DOWNLOAD request from ETHM-1 module
1023,0, 6, 'Tamper info cleared

```

The meaning of nnnnnnnn field:

- if users numbering:
 - 1..240 - user
 - 241..248 - master
 - 249 - INT-AV
 - 251 - SMS
 - 252 - timer
 - 253 - function zone
 - 254 - Quick arm
 - 255 - service
- if zone|expander|keypad numbering:
 - 1..128 - zone
 - 129..192 - expander at address 0..63
- INTEGRA 24 and 32:
 - 193..196 - real LCD keypads or INT-RS modules at address 0..3
 - 197..200 - keypad in GuardX connected to LCD keypad at address 0..3, or www keypad in internet browser connected to ETHM-1 at address 0..3
 - 201 - keypad in DloadX connected to INTEGRA via RS cable
 - 202 - keypad in DloadX connected to INTEGRA via TEL link (modem)
- INTEGRA 64, 128 and 128-WRL:
 - 193..200 - real LCD keypads or INT-RS modules at address 0..7
 - 201..208 - keypad in GuardX connected to LCD keypad at address 0..7, or www keypad in internet browser connected to ETHM-1 at address 0..7
 - 209 - keypad in DloadX connected to INTEGRA via RS cable
 - 210 - keypad in DloadX connected to INTEGRA via TEL link (modem)
- if output|expander numbering:
 - 1..128 - output
 - 129..192 - supply output in expander at address 0..63

Appendix 2 - kind of long description

Kind of long description:

- 0 - no additional description
- 1 - partition/zone|expander|keypad
- 2 - partition/user
- 3 - partition keypad/user (partition keypad address in PPPPPR) (not LCD keypad, but LED partition keypad, e.g. INT-S)
- 4 - zone|expander|keypad
- 5 - partition
- 6 - keypad/user
- 7 - user
- 8 - expander reader head
- 9 - telephone
- 10 - output of telephone relay type
- 11 - partition/data bus
- 12 - partition/output|expander (partition not important for main panel outputs)
- 13 - partition/output|expander (partition not important for outputs)
- 14 - telephone in PPPPP/user (telephone: 0 - unknown, 1.. - phone number)
- 15 - partition/timer
- 30 - beginning of TCP/IP address (keypad address in PPPPP)
- 31 - 3rd and 4th bytes of TCP/IP address
- 32 - partition/zone or ABAX output

Appendix 3 - kind of short description

Kind of short description (just another kind of event description) - 2 bytes: MrIRODnT gtwmkues of the following bit meaning:

- s - partition
- e - zone/expander/LCD-keypad
- u - user
- k - expander in RPPPPP
- m - LCD-keypad in PPPPP
- w - output/expander, partition only for expandera
- t - timer
- g - proximity card reader
- T - telephone
- n - number (RAM error)
- D - data bus (0=DTM, 1=DT1, 2=DT2, 129..128+IL_EXPAND=expander)
- o - call back (0='SERV', 1='SERV=', 2='USER', 3='USER=', 4='ETHM-modem', 5='ETHM-RS')
- R - telephone relay
- I - TCP/IP event (2 records !!!)
- r - ABAX input/output, partition only for input
- M - monitoring

Appendix 4 - crc calculation example

Assume that the following data has to be send to INT-RS module: 0xE0, 0x12, 0x34, 0xFF, 0xFF (i.e. read information about user with 1234 code). For this case the following frame should be generated:

0xFE	0xFE	0xE0	0x12	0x34	0xFF	0xFF	0x8A	0x9B	0xFE	0x0D
------	------	------	------	------	------	------	------	------	------	------

The 16-bit crc sum calculation goes as bellow:

- 1) $\text{crc} := 0x147A$
- 2) for byte $b = 0xE0$:
 - $\text{crc} := \text{rl}(\text{crc}) = \text{rl}(0x147A) = 0x28F4$
 - $\text{crc} := \text{crc} \text{ xor } 0xFFFF = 0x28F4 \text{ xor } 0xFFFF = 0xD70B$
 - $\text{crc} := \text{crc} + \text{crc.high} + b = 0xD70B + 0xD7 + 0xE0 = 0xD8C2$
- 3) for byte $b = 0x12$:
 - $\text{crc} := \text{rl}(\text{crc}) = \text{rl}(0xD8C2) = 0xB185$
 - $\text{crc} := \text{crc} \text{ xor } 0xFFFF = 0xB185 \text{ xor } 0xFFFF = 0x4E7A$
 - $\text{crc} := \text{crc} + \text{crc.high} + b = 0x4E7A + 0x4E + 0x12 = 0x4EDA$
- 4) for byte $b = 0x34$:
 - $\text{crc} := \text{rl}(\text{crc}) = \text{rl}(0x4EDA) = 0x9DB4$
 - $\text{crc} := \text{crc} \text{ xor } 0xFFFF = 0x9DB4 \text{ xor } 0xFFFF = 0x624B$
 - $\text{crc} := \text{crc} + \text{crc.high} + b = 0x624B + 0x62 + 0x34 = 0x62E1$
- 5) for byte $b = 0xFF$:
 - $\text{crc} := \text{rl}(\text{crc}) = \text{rl}(0x62E1) = 0xC5C2$
 - $\text{crc} := \text{crc} \text{ xor } 0xFFFF = 0xC5C2 \text{ xor } 0xFFFF = 0x3A3D$
 - $\text{crc} := \text{crc} + \text{crc.high} + b = 0x3A3D + 0x3A + 0xFF = 0x3B76$
- 6) for byte $b = 0xFF$:
 - $\text{crc} := \text{rl}(\text{crc}) = \text{rl}(0x3B76) = 0x76EC$
 - $\text{crc} := \text{crc} \text{ xor } 0xFFFF = 0x76EC \text{ xor } 0xFFFF = 0x8913$
 - $\text{crc} := \text{crc} + \text{crc.high} + b = 0x8913 + 0x89 + 0xFF = 0x8A9B$

And the final crc sum is 0x8A9B.