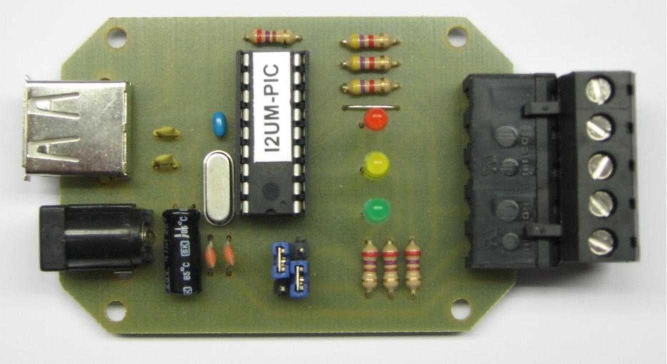
Inhalt



1. General

1.1. Communication parameters

The I2C USB modem receives and sends the commands via a virtual COM port with the following settings:

- Baud rate: 115200
- Parity: none
- Character bits: 8
- Stop bit: 1

After the modem has fully received a command, it is applied to the I2C bus. It then generates a response byte which is sent back to the PC

1.2. protocol

The modem is controlled via a protocol. A complete protocol (hereinafter referred to as a frame) includes:

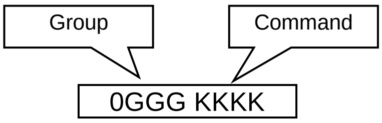
- the command
- the number of bytes in the data block (maximum 128)
- the data block itself and
- the end character (EOT= END OF TRANSMISSION).

1.2.1. Command frame

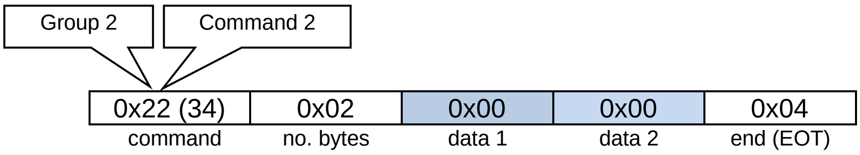
The commands for the I2C USB modem have been grouped into logical groups.

- • (1) Info all general requests to the modem are summarized in this group.
- • (2) Configure all commands related to the modem itself
- • (3) I2C all commands related to the I2C bus
- • (4) Analyze the commands for the analysis and interrupt functions

The group number is in the upper nibble of the command, the command is in the lower nibble.



Beispiel für einen Commands-Frame mit 2 Byte Data:

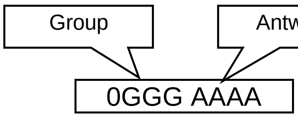


Info:
For better understanding, the data block is always shown with a light blue background in this description.
The numbers given are hexadecimal values in the format 0x_ _.
For larger hex numbers, the decimal value was written in brackets after them.

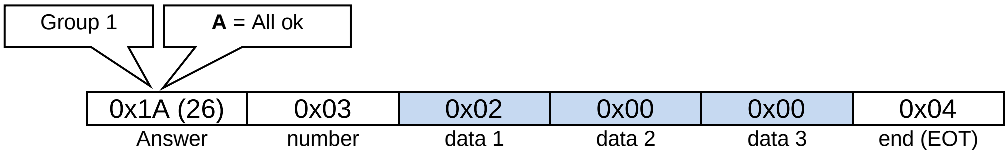
1.2.2. Response frame

If a sent command has been processed by the modem, a response is returned to the PC via the virtual COM port.

The group number of the transmitted command is again stated in the upper nibble of the response byte. If the modem understood the command and was able to execute it successfully, there is a hexadecimal “A” in the lower nibble of the response byte for “All is ok”.



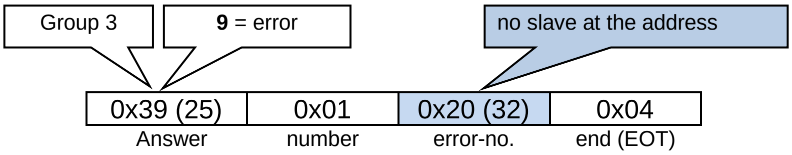
Example of a response frame with three bytes of data:



0x1A (26)	0x03	0x02	0x00	0x00	0x04
Answer	number	data 1	data 2	data 3	end (EOT)

In the event of an error, the group number is also in the upper nibble - in the lower nibble of the However, the response bytes are a “9”.

Example of a response frame with ERROR number:



0x39 (25)	0x01	0x20 (32)	0x04
Answer	number	error-no.	end (EOT)

1.3. Error numbers

In this list you will find a list of possible error numbers:

Error number		description
0x01	(1 dez)	All is OK, no error
0x02	(2 dez)	The group address (upper nibble) is unknown
0x03	(3 dez)	The command (lower nibble) is unknown
0x04	(4 dez)	The data block length was not sent or is incorrect
0x05	(5 dez)	The length of the data block is too long
0x06	(6 dez)	END OF TRANSMISSION (04) is missing at the end of the frame
0x07	(7 dez)	04 was not sent as EOT
0x08	(8 dez)	Timeout sending data block
0x10	(16 dez)	Version command is constructed incorrectly. The frame head is correct but the rest is not correct!
0x11	(17 dez)	Too much data on the call modem command
0x20	(32 dez)	No slave responds to the sent address
0x21	(33 dez)	The slave has the Acknolw. not reacted
0x22	(34 dez)	If the slave triggers a clock stretch, the modem waits maximum 1.5 seconds. A timeout then occurs.
0x42	(66 dez)	The listen timeout value is invalid
0x43	(67 dez)	The list timeout value has been exceeded. The address you were looking for did not appear during monitoring.
0x44	(68 dez)	While waiting for the next SCL imp. there was a timeout. A slave can block the SCL line if it needs time to read data from internal registers (clock stretch). This time is limited to a maximum of 1.5 seconds.
0x45	(69 dez)	IS table is full, no further entry possible. Although there are already 16 entries in the table, an attempt was made to make another entry.
0x46	(70 dez)	Too few bytes were sent when sending the IS table data. 3 bytes are always required per entry, address low byte, address high byte and the number of read operations.
0x47	(71 dez)	No more than 4 read operations can be entered into the IS table.
0x48	(72 dez)	Error deleting IS table
0x49	(73 dez)	A write address should be entered into the IS table. However, the table only accepts read addresses.
0x4A	(74 dez)	Error starting INT monitoring
0x4B	(75 dez)	INT monitoring should be started with an empty table
0x4C	(76 dez)	The table is to be changed even though INT monitoring is running.
0x4D	(77 dez)	All addresses in IS table have been read and the INT signal is still LOW
0xFF	(255)	Unknown command!

2. Commands

2.1. Group 1 commands = INFO

This group contains all commands that provide general information.

2.1.1. VERSION 11 hex = 17 dec.

This command can be used to determine the current firmware version. To do this, the modem sends a frame without data (number = 0).

Command:

0x11 (17)	0x00	0x04
Command	Number	End (EOT)

Data with the version number

Answer:

0x1A (26)	0x03	0x02	0x30	0x00	0x04
Answer	Number	Version	NK1	NK2	End (EOT)

The answer frame contains 3 bytes of user data, which contains the firmware version. The first byte in the data block represents the version number and the following two represent the decimal place. In our example this is 2 3 0 and therefore version 2.30

2.1.2. MODEM-CALL 12 hex =18 dez.

This command has the task of “triggering” the modem. This makes it easy to determine whether the modem is still connected to the PC or whether there are problems with the USB connection.

Command:

0x12 (18)	0x00	0x04
Command	Number	End (EOT)

# = Modem ok

Answer:

0x1A (26)	0x01	0x23 (35)	0x04
Answer	Number	#	End (EOT)

The modem sends a prompt. In this case it is a “#”.

2.2. Commands of group 2 = KONFIG

The group contains the commands for configuring the I2C USB modem.

2.2.1. PULLUP 21 hex = 33 dez.

The I2C bus needs +5V termination at one point in the network. This is usually done with resistors. During communication, the bus participants pull these high levels to ground. There are applications where termination should not be made at the master. Then the pull-up resistors must be able to be switched off.

2.2.1.1. Pullup´s read

This command can be used to read the current status of the pullup resistors.

Command:

0x21 (33)	0x00	0x04
Command	Number	End (EOT)

128 = PullUp resistors ON
0 = PullUp resistors OFF

Answer:

0x2A (42)	0x01	0x80 (128)	0x04
Answer	Number	EIN	End (EOT)

The modem sends back a value that represents the state of the pullups. Is the value 0x80 (128) the pullups are switched on. At 0 the pull-ups are de-energized.

2.2.1.2. Pullup´s switch on

If a “1” is sent in the data block, the modem switches the pull-up resistors ON.

1 = PullUp resistors ON

Command:

0x21 (33)	0x01	0x01	0x04
Command	Number	EIN	End (EOT)

Answer:

0x2A (42)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

2.2.1.3. Pullup´s switch off

If a “0” is sent in the data block, the modem switches its pull-up resistors OFF.

0 = PullUp resistors OFF

Command:

0x21 (33)	0x01	0x00	0x04
Command	Number	AUS	End (EOT)

Answer:

0x2A (42)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

2.2.2. I2C-SPEED 22 hex = 34 dez.

With this command the clock speed on the I2C bus can be set between 350 kHz and 40 Hz. This is a two-byte value that must be stored with the LSB first in the data block.

ACHTUNG:
When selecting the bus speed, please note that with a clock rate of 40 Hz and 128 bytes that are to be transferred or read to the slave, the transmission
can take approx. 30 seconds.

The value to be transferred can be calculated using the following formulas:

$$value = \frac{1}{bus\ cycle[Hz] * 0{,}4 * \ 10^{-6}\ s}$$

$$bus\ cycle[Hz] = \frac{1}{value\ * 0{,}4 * \ 10^{-6}\ s}$$

The division into the two bytes is then carried out using the following formula, where FIX is the integer part of a number.

$$HBy = \ FIX\ (value : 256)$$

$$LBy = \ value\ \ - \ HBy * 256$$

<u>Example:</u>
You want to set a bus clock of 2500 Hz. The formula results in a value of 1000
1000 / 256 = 3 remainder 232
The 3 must be written in the high byte and the 232 in the low byte

In the table below you will find the calculated data for some common bus speeds:

Takt [kHz]	Takt [Hz]	value	high-Byte	low-Byte
350 kHz	350000 Hz	7	0	7
250 kHz	250000 Hz	10	0	10
125 kHz	125000 Hz	20	0	20
100 kHz	100000 Hz	25	0	25
50 kHz	50000 Hz	50	0	50
25 kHz	25000 Hz	100	0	100
10 kHz	10000 Hz	250	0	250
5 kHz	5000 Hz	500	1	244
2.50 kHz	2500 Hz	1000	3	232
1.25 kHz	1250 Hz	2000	7	208
1 kHz	1000 Hz	2500	9	196
0.5 kHz	500 Hz	5000	19	136
0.1 kHz	100 Hz	25000	97	168

2.2.2.1. Speed set

2.2.2.2. In order to set the bus speed, the calculated 2-byte value must be transferred.

Current bus cycle
Value 2000 = 1250 Hz

Command:

0x22 (34)	0x02	0XD0 (208)	0x07 (7)	0x04
Command	Number	Low Byte	High Byte	End (EOT)

Answer:

0x2A (42)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

If no errors have occurred, the modem responds with an OK frame

2.2.2.3. Speed check

If the speed command is called with an empty data block (length 0), the modem returns the currently set bus clock as a 2-byte value. The current bus clock can be calculated from the two byte values using the specified formula.

Command:

0x22 (34)	0x00	0x04
Command	Number	End (EOT)

Current bus cycle
Value 1000 = 2500 Hz

Answer:

0x2A (42)	0x02	0xE8 (232)	0x03 (3)	0x04
Answer	Number	Low Byte	High Byte	End (EOT)

$$\boxed{bus\ cycle[Hz] = \frac{1}{(HBy * 256 + LBy)\ * 0{,}4 *\ 10^{-6}\ s}}$$

2.3. Commands of group 3 = I2C

This group contains all the commands that are needed to send or receive data on the I2C bus.

2.3.1. I2C-SET 31 hex = 49 dez.

This command can be used for testing purposes **INT**, **SCL** and **SDA** can be set individually to high or low.

Attention:
After the test, all signals must be set to high again, otherwise the bus will not recognize the start signal.

Command:

all Bits to „High“

0x31 (49)	0x01	0x07	0x04
Command	Number	0 0 0 0 0 **INT** **SCL** **SDA**	End (EOT)

Bits 7 – 3 are ignored. Bit two is called **INT**, Bit one is interpreted as **SCL** and bit zero as **SDA**
The number in the data block must be calculated using the value of the bit position. (INT=4, SCL=2, SDA=1)

- 0x01 set SDA=1, SCL=0 and INT=0
- 0x02 set SDA=0, SCL=1 and INT=0
- 0x04 set SDA=0, SCL=0 and INT=1
- 0x07 set SDA=1, SCL=1 and INT=1

Answer:

Target state
All bits on “high”

Actual state
INT=0 SCL=1 SDA=1

0x3A (58)	0x02	0x07	0x03	0x04
Command	Number	0 0 0 0 0 **INT** **SCL** **SDA**	0 0 0 0 **INT** **SCL** **SDA**	End (EOT)
		Target state	actual state	

If the command was recognized correctly, the modem answered with the target state and the current actual state of the I2C signals:

2.3.2. I2C-GET 32 hex = 50 dez.

I2C-Get queries the current status of the **INT**, **SCL** and **SDA** signals.

Command:

0x32 (50)	0x00	0x04
Command	Number	End (EOT)

Actual state
INT=1 SCL=1 SDA=1

0x3A (58))	0x01	0x07	0x04
Answer	Number	0 0 0 0 0 **INT** **SCL** **SDA**	End (EOT)

Info:
On the I2C USB modem, the **SDA** and **SCL** LEDs light up when the bus signal is high. The **INT** LED lights up when the signal is low, i.e. when a signal change has been detected on an input card.

2.3.3. I2C-DATA 33 hex = 51 dez.

With this command you can receive data from an I2C slave or send data to an I2C slave.
The modem recognizes whether data should be sent or received by the R/W bit in the address. If the R/W bit (bit 0) is “Low”, data is written. If the R/W bit is “High”, data is read.

> ***Info:***
> „Data is written to even slave addresses and data is read from odd addresses.

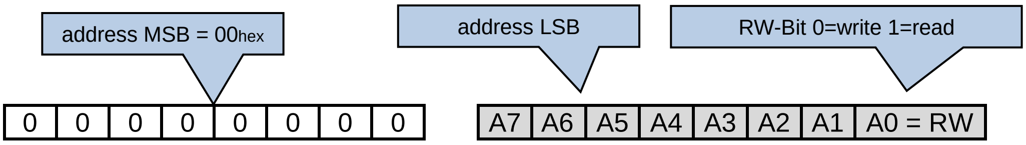
> **Attention:**
> The I2C USB modem can read or write a maximum of 128 bytes with one data operation. Please check the data sheet of the slave to see what amounts of data can be read or written. When writing to EEproms, for example, there must be a short pause after eight bytes.

2.3.3.1. Structure of the 8-bit and 10-bit slave address

In order to read data from an I2C slave, the address is first stored in the data block and then the desired number of bytes to be read is specified.

Normally the slave address of an I2C slave only has 7 (or 8) bits. But since more and more ICs are being used
10 bit wide address comes onto the market, the address was divided into bytes. If the MSB is zero, the modem interprets the transmitted address as a 7-bit address. If there is data in the MSB, a 10-bit wide address access is generated.

Structure 8-bit address:

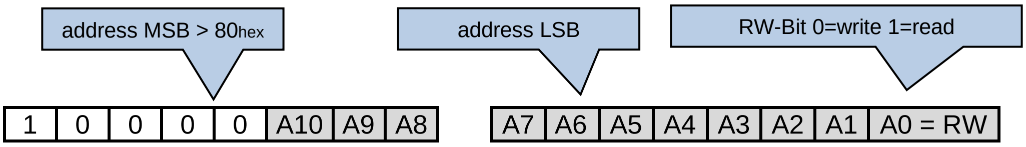


0	0	0	0	0	0	0	0

A7	A6	A5	A4	A3	A2	A1	A0 = RW

> ***Info:***
> If the slave is to be addressed with an 8-bit address, all of them must be in the MSB byte
> Bits must be set to LOW.

Structure 10-bit address



1	0	0	0	0	A10	A9	A8

A7	A6	A5	A4	A3	A2	A1	A0 = RW

> ***Info:***
> If the slave is to be addressed with a 10-bit address, the highest bit in the MSB byte must be set to HIGH.

2.3.3.2. Data lesen

In the example below, 5 bytes are to be read from an I2C slave at address 161.

The command has 3 Bytes

from 8-Bit-Adress 161

5 Bytes read

Command:

0x33 (51)	0x03	0xA1 (161)	0x00	0x05	0x04
Command	Number	Adr. LSB	Adr. MSB	no. Bytes	End (EOT)

Answer:

0x3A (58))	0x05	0x0A	0x0B	0x0C	0x0D	0x0E	0x04
Answer	Number	Data 1	Data 2	Data 3	Data 4	Data 5	End (EOT)

Here two bytes are to be read from an I2C slave with a 10-bit address.

The command has 3 Bytes

From 10-Bit adress 161

5 Bytes read

Command:

0x33 (51)	0x03	0xA1 (161)	0x80	0x02	0x04
Command	Number	Adr. LSB	Adr. MSB	no. Bytes	End (EOT)

Answer:

0x3A (58))	0x05	0x11	0x22	0x04
Answer	Number	Data 1	Data 2	End (EOT)

2.3.3.3. Data schreiben

In order to send I2C data to a slave, the address is first stored in the data block and then the data to be sent is added afterwards.

In the example below, 3 bytes are to be written to an I2C slave at address 172.

The command has 5 Bytes

From adress 172

3 Bytes write

Command:

0x33 (51)	0x05	0xAC (172)	0x00	0xAA	0xBB	0xCC	0x04
Command	Number	Adr. LSB	Adr. MSB	Data 1	Data 2	Data 3	End (EOT)

Answer:

0x3A (58)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

If no errors have occurred, the modem responds with an OK frame

2.4. Commands of group 4 = ANALYSE

This group contains commands that can be used to analyze the data traffic on the I2C bus in order to find errors in I2C communication.

2.4.1. SET FILTER 0x41 hex = 65 dez.

Since general monitoring of the I2C bus would provide too much data, a filter can be set to an I2C slave address with this command. If the USB modem “eavesdrops” on the data traffic using the LISTEN command, the data is only recorded when the address set in the filter is recognized in the I2C protocol.

The command has 2 Bytes

Listen address 112

Command:

0x41 (65)	0x02	0x70 (112)	0x00	0x04
Command	Number	Adr. LSB	Adr. MSB	End (EOT)

Answer:

0x4A (74)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

If the address and the frame are valid, the address is entered into the filter and the modem answers with an OK frame.

The command has 2 Bytes

Listen without a filter

Command:

0x41 (65)	0x02	0x00	0x00	0x04
Command	Number	Adr. LSB	Adr. MSB	End (EOT)

Answer:

0x4A (74)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

If the address 0x0000 is entered in the filter, the modem records all data on the I2C bus and sends it to the PC. (→ Kapitel 2.4.2.2 „LISTEN “)

2.4.2. LISTEN 0x42 hex = 66 dez.

This command starts recording the data traffic on the I2C bus. Four different recording modes are available:

- Time-limited with address filter
- Unlimited time with address filter
- Time-limited without address filter
- Unlimited time without address filter

> ***Info:***
> **!** The filter address must first be transmitted to the modem using the SET FILTER command.

Example of a data stream =I2C-🟩tart =I2C-🟥top

🟩	A0	00	🟥	🟩	A1	01	02	03	04	05	🟥	🟩	70	FD	🟥	🟩	A0	05	🟥	🟩	A0	06	07	08	09	0A	🟥

In the first block, the address pointer is set to 0 on an I2C EEprom (address 160).
In the next block, the master reads 5 bytes of data from the EEprom (address 161).
In the third block, the bit pattern “11111101” is output at a port block (address 112).
In the fourth block, the address pointer is set to 5 on the I2C EEprom (address 160).
In the fifth block, 5 bytes of data are written into the EEprom (address 160).

2.4.2.1. LISTEN with a time limit

The “listening time” can be set between 0.052 and 13 seconds using the command. The calculation is done using the following formula.

$$\boxed{listening\ time[sec] = Wert * 0{,}052\ sec.}$$

The following example starts the list function for approx. 6 seconds with the previously set filter.

The command has 1 Byte

value 120 = 6,24 seconds

Command:

0x42 (66)	0x01	0x78 (120)	0x04
Command	Number	time value	End (EOT)

Answer:
If the filter address is found in the data stream, the modem sends an answer frame.

For example, if the filter is set to 0x70 and a master sends 0xDF to the address 0x70, the following frame is generated by the I2C USB modem:

🟩	A0	00	🟥	🟩	A1	01	02	03	04	05	🟥	🟩	70	DF	🟥	🟩	A0	05	🟥	🟩	A0	06	07	08	09	0A	🟥

0x4A (74)	0x02	0x70 (112)	0xFD (223)	0x04
Answer	Number	Filter	Data	End (EOT)

In the data area you will first find the address that was overheard, then the data that was seen on the bus.

2.4.2.2. LISTEN without time limit

If time 0 is passed with the LISTEN command, the modem starts unlimited monitoring. This is terminated by the host when it sends EOT 0x04 (4 dec.) to the modem.

The command has 1 Byte

0 = no time limit

Command:

0x42 (66)	0x01	0x0 (0)	0x04
Command	Number	time value	End (EOT)

Answer:
At the start of monitoring, the modem sends “SOT” (***S***tart ***O***f ***T***ransmision) in the following frame:

0x4A (74)	0x03	0x53	0x4F	0x54
Answer	Number	„S“	„O“	„T“

It is noticeable that the frame end 0x04 (4 dec.) is missing.
Now comes the data that the modem finds on the I2C bus. The modem starts transmitting data to the PC as soon as an I2C start signal is detected in the data stream. With the I2C stop, the data transmission also stops. An “open frame” is created that must be closed by the PC.

🟩	A0	00	🟥	🟩	A1	01	02	03	04	05	🟥	🟩	70	DF	🟥	🟩	A0	05	🟥	🟩	A0	06	07	08	09	0A	🟥

If the filter is set to 0xA1, the following bytes are sent from the modem to the PC

0xA1 (161)	0x01 (1)	0x02 (2)	0x03 (3)	0x04 (4)	0x05 (5)

If the filter is switched off, all bytes that are recognized on the bus are sent to the modem

0xA0 (160)	0x00 (0)	0xA1 (161)	0x01 (1)	0x02 (2)	0x03 (3)	0x04 (4)	0x05 (5)

0x70 (112)	0x05 (223)	0xA0 (160)	0x05 (5)

0xA0 (160)	0x06 (6)	0x07 (7)	0x08 (8)	0x09 (9)	0x0A (10)

If the modem no longer requires data, the PC must send an EOT = 0x04 (4 dec.).

Command:

0x04
End (EOT)

Answer:

0x4A (74)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

2.4.2.3. LISTEN-ERROR 0x4B hex = 75 dez.

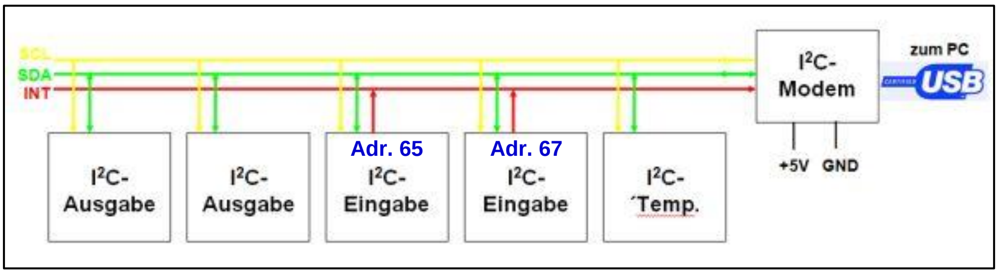
When eavesdropping on the I2C bus, data may be sent incompletely. For example, if the address and the first date are still sent correctly and the second date is incomplete, the I2C USB modem transmits the data that has been overheard up to this point to the PC. To indicate that something went wrong during data transmission on the I2C bus, the modem enters a 0x4B (75 dec) in the frame header.

2.4.3. LOAD TABLE 0x43 hex = 67 dez.

We have integrated an interesting feature into our I2C USB modem for interrupt processing on the I2C bus. All interrupt-capable slaves can be connected in parallel at the INT input. On our input cards with the PCF8574 or PCF8574A module, the INT is pulled to “LOW” as soon as a signal change is detected at the inputs. As of version 2.2, the USB modem is able to independently query a number of slaves when the INT edge falls and report changed input signals to the PC without having to carry out a read operation.

For this purpose, a table is created in the modem in which the addresses of all connected input cards and the number of read operations are stored. Interrupt processing is enabled

(CHECK-INT = 1), this table is processed from top to bottom in the event of an interrupt. If the modem reads data from the slave that triggered the interrupt, the interrupt disappears. The I2C USB modem recognizes that this slave triggered the interrupt and informs the PC of the slave number and the content of the data read out.



To make an entry in the IS table of the I2C USB modem, the following frame must be sent

The command has 3 Bytes

Von Adresse 65

Slave 1x auslesen

Command:

0x43 (67)	0x03	0x41 (65)	0x00	0x01	0x04
Command	Number	Adr. LSB	Adr. MSB	Number. Leseoperationen	End (EOT)

Answer:

0x4A (74)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

If the data has been validly and successfully entered into the table, the modem responds with an OK. The next address can then be stored

The command has 3 Bytes

Von Adresse 67

Slave 3x auslesen

Command:

0x43 (67)	0x03	0x43 (67)	0x00	0x02	0x04
Command	Number	Adr. LSB	Adr. MSB	Number. Leseoperationen	End (EOT)

Answer:

0x4A (74)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

!

Info:
The IS table can hold a maximum of 16 addresses + number of read operations. It is stored in the EEprom of the PIC processor and is retained in the event of a power failure.

Attention:
When INT monitoring is active, the IS table cannot be edited.

2.4.4. CLEAR TABLE 0x44 hex = 68 dez.

If this command is sent to the modem, all 16 entries in the IS TABLE will be deleted.

Command:

0x44 (68)	0x00	0x04
Command	Number	End (EOT)

Answer:

0x4A (74)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

⚠ **Attention:**
When INT monitoring is active, the table cannot be deleted.

2.4.5. CHECK-INT 0x45 hex = 69 dez.

This command starts (value=1) or stops (value=0) the INT monitoring.

The command has 1 Byte

1 = Monitoring ON
0= Monitoring OFF

Command:

0x42 (66)	0x01	0x1 (1)	0x04
Command	Number	value	End (EOT)

Answer:

0x4A (74)	0x01	0x01	0x04
Answer	Number	OK	End (EOT)

If CHECK-INT is active, only the two commands for manipulating the IS table are blocked. Otherwise, the modem continues to work normally and monitors the status in the background
INT management.

If an INT signal occurs (red LED lights up), the following happens:

- The modem sends a BUSY frame to the PC.
- The modem processes the IS table one after the other and reads the stored number of bytes from the slave. If the INT signal disappears, the slave that triggered the interrupt has been found.
- The modem sends the data read from this slave to the PC.
- The modem returns to normal operation.

Structure of BUSY frame:

The command has 1 Byte

IS-Tabelle hat 2 Einträge

0x4F (79)	0x01	0x02	0x04
Answer	Number	values in der IS-Table	End (EOT)

The BUSY frame indicates that the modem will not be accessible for a certain period of time because the IS table must now be processed. The time it takes is determined by the size of the table (number of entries and read operations). If the INT signal is deleted during one of these read operations, the modem responds with a data frame:

Structure of DATA frame:

example 1:

The command has 2 Byte

Slave-adress

data

0x4A (74)	0x02	0x41 (65)	0xDF	0x04
Answer	Number	Slave 65	Data	End (EOT)

During the read operation at address 0x41, the INT signal was cleared. The modem received the date 0xDF

example 2:

The command has 3 Byte

Slave- adress

data

0x4A (74)	0x03	0x43 (67)	0x01	0x00	0x04
Answer	Number	Slave 65	Data 1	Data 2	End (EOT)

During the read operation at address 0x43, the INT signal was cleared. The modem received the date 0x01 and then 0x00. As a result, there was a signal change at bit 0 from “HIGH” to “LIOW” on the buffered I2C input card.

⚠ **Attention**
The INT signal only consists of one line to which several blocks are connected. If several blocks trigger an INT signal at the same time, they overlap.
The modem is not able to recognize an overlay!

Example:
The addresses 0x71, 0x75 and 0x79 are entered in the table as INT sources. Now 0x71 and 0x79 trigger an INT at the same time.
According to the table, the modem reads the data from address 0x71, but the INT signal remains because it is kept low by slave 0x79.
Only when the modem reads 0x79 is the INT signal released again. (INT level becomes high – red LED goes out) The data from address 0x79 is transferred to the PC. The data of the block at address 0x71 is lost due to the overlay