Hasomed GmbH
Paul-Ecke-Straße 1,
39114 Magdeburg, Germany
www.hasomed.de

HASOMED®
Hard- und Software für die Medizin

ScienceMode2

RehaStim2 Stimulation Device

Description and Protocol

Version	1.24
Date	2012-12-12
Author	Bjoern Kuberski
Email	bjoern.kuberski@hasomed.de

Table Of Contents

1. ScienceMode2 General Description....3
1.1 Introduction....3
1.2 Stimulation Device....3
1.3 Stimulation Commands....4
1.4 Motomed Commands....6
1.5 RehaStim2 Graphical User Interface....7
1.6 ScienceMode2 PC Test Software....7
1.7 Limitation....7
2. Protocol General....8
2.1 Serial Connection....8
2.2 Packet Structure....8
2.3 Timeouts / Time Intervals....9
3. Protocol Commands Overview....10
3.1 Connection commands....10
3.2 RehaStim2 Modes....10
3.3 Stimulation Commands....10
3.4 Motomed Commands....10
3.5 Communication Direction of Commands....11
3.6 Modes and Commands Overview....13
4. Protocol Commands Details....14
4.1 Initialization of Connection Commands....14
4.2 RehaStim2 Mode Commands....15
4.3 Stimulation Commands....16
4.4 Motomed Commands....19
5. Error Handling....29
5.1 Error Codes for the Response Commands....29
5.2 Error Detection....29
5.3 Communication errors....30
6. Disclaimer....31

1. ScienceMode2 General Description

1.1 Introduction

ScienceMode2 is a communication protocol between a PC (or external device) and the 8-channel functional electrical stimulator RehaStim2 by the company HASOMED GmbH (see http://www.hasomed.com). Using the ScienceMode2 commands, the PC can control the stimulation and the movement therapy devices of the MOTOmed series by the company RECK (see http://www.motomed.com). The bidirectional communication of ScienceMode2 allows to create complex stimulation patterns and training scenarios for a wide range of of research applications.

Figure 1: Portable 8 channel stimulator RehaStim2

1.2 Stimulation Device

The current-controlled 8-channel stimulator RehaStim2 is a certified medical product and possesses two independent current sources which are multiplexed to 4 outputs each. Together with the motion therapy devices of the MOTOmed series by RECK the RehaStim2 combines movement therapy with electrical stimulation, whereupon paralysed muscles are actively engaged to generate muscle strength. The portable RehaStim2 is operated by a keypad and the colored LCD-Display The technical specification of the stimulator is listed in Table 1.

Current	[0, 130] mA
Pulse width	[0, 500] μs
Frequency	[1, 50] Hz for 8 channels
Pulse form	Biphasic rectangular impulses with balanced electric charge
Channels	8 (2 current sources)
Serial Port	USB with galvanic isolation

Table 1: Technical details of the RehaStim2 stimulation device

Figure 2 shows the form of a biphasic pulse on an ideal resistive load. Notice that

there is a fixed pause of 100 μs between the two phases of the pulse. At the end of the pulse the remaining charge on the electrodes and skin is removed by an active short-cut (change of electrode polarity for 1 μs).

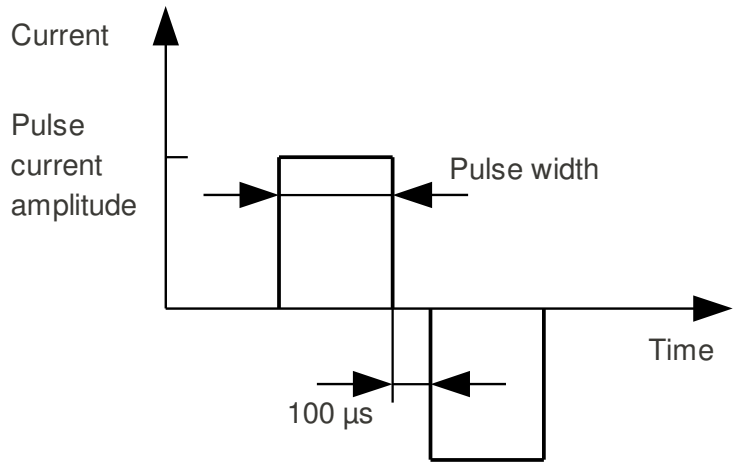

Figure 2: Pulse width and current amplitude for a biphasic pulse.

The stimulator provides a skin resistance check for safety reasons. The resistance is determined by analyzing the effect of a small test impulse which is send before each stimulation pulse. If the resistance is not inside normal ranges then a stimulation pulse will not be generated.

1.3 Stimulation Commands

The stimulation commands control the pulse generation of the stimulator. There are three modes of pulse generation available: the continuous channel list mode, the one shot channel list mode and the single pulse generation.

Continuous Channel List Mode (CCLM)

Using this mode, the generation of complex patterns is greatly simplified. A list of stimulation channels has to be specified, on which pulses or even pulse groups (doublets or triplets) will repeatedly be generated. The channel list is repeatedly processed with the main stimulation period t_1. According to pulse group mode (single, doublets or triplets), the channel list is processed one, two or three times using the inter pulse interval t_2. Figure 3 shows the concept of the CCLM with the help of an example. Pulse generation takes place on the selected channels, ordered by the channel numbers. For each selected channel a time slot of 1.5ms is reserved, even if current or pulse width are zero for the channel. Both stimulation modules (current sources) are processed sequentially. Stimulation module A generates pulses on the channel 1 to 4 if applicable and stimulation module B generates pulses on the channels 5 to 8 if applicable.

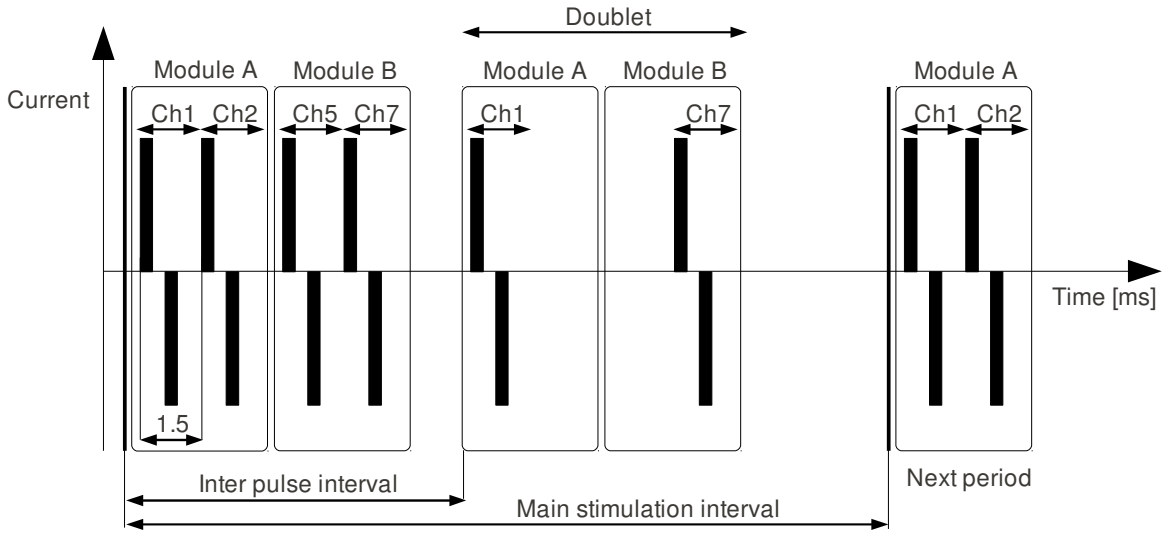

Figure 3: Example for the CCLM of the stimulator. Stimulation pulses are represented by black bars where the same width and amplitude have been assumed for simplicity. The channel list encloses the channels 1, 2, 7 and 7. The main stimulation frequency is 1 / main stimulation interval. Doublets are generated on the channel 1 and 7.

The CCLM is initialized by command using the parameters in Table 2:

Parameter	**Description**
Inter pulse interval (t_2)	Defines the time between two pulse groups, if doublets or triplet are selected. The minimum inter pulse interval is 8 ms per stimulation module (4 x 1.5 ms for each channel + 2 ms communication buffer).
Main stimulation interval (t_1)	Defines the main time, where the channel list is repeatedly processed. We have $t_1 \geq n_{Pgr} \cdot t_2$, where n_{Pgr} is the maximum of the selected pulse group modes used ($n_{Pgr} = 1$ for single pulses, $n_{Pgr} = 2$ for doublets, $n_{Pgr} = 3$ for triplets).
Active channels	Defines the active channels.
Low frequency channels	Defines the channels, which have a lower frequency than the normal channels by the low frequency factor.
Low frequency factor	Defines how many times the stimulation is skipped for the channels specified in low frequency channels. (0 = no skip, 1 = skip once etc.)
Channel execution	Defines the execution of the pulses in a fixed 1.5 ms interval or as fast as possible (not yet implemented).

Table 2: Parameter description of channel list mode initialization

After initialization the CCLM is started using a command with current, pulse width and pulse group mode parameters for each activated channel. The same command is used to update those parameters. The channel list is repeatedly processed according to the specified intervals.

The CCLM can be stopped with the appropriate command. There is also a command for getting the current stimulation mode. Note that if the start/update command is send faster than the main stimulation interval (or interpulse interval), some start/update commands might be skipped, because the parameters are overwritten before the stimulation is executed.

One Shot Channel List Mode (OSCLM)

Just as in the Continuous Channel List Mode a channel list in OSCLM is defined by the initialization step. However, processing of the channel list is not automatically repeated so that the time period t_1 ceases to exist. Instead, the channel list will be processed once and pulses and pulse groups are generated, only if the start/update command is issued. The OSCL mode offers the possibility to control the main stimulation frequency by the PC while the inter-pulse time of the doublets and triplets is realized by the stimulator. The OSCLM is initialized by setting the main stimulation time to the lowest possible value (main stimulation interval = 1). Note that like in the CCLM the parameters could be overwritten, if the start/update command is send faster than the interpulse interval.

Single Pulse

Using this command the stimulator generates a single pulse on a specified channel with the desired current amplitude and pulse width. The stimulator will generate the pulse immediately after processing the command. Complex stimulation patterns may be generated by repeatedly sending this command. The PC controls the stimulation frequency by sending the command accordingly.

Stimulation errors

There are three types of errors that can happen during stimulation: emergency switch (kill switch) is activated (or not connected), electrode error and stimulation module error. If one of these occur, RehaStim2 stops the current stimulation, resets its mode to start mode and sends an error command to the PC. When the problem is fixed, the stimulation can be restarted using the appropriate commands.

Channel numbers

Table 3 shows the mapping of the channel numbers to the channel color.

Stimulation module	Channel number	Side color	Channel color
A	1	Yellow	Red
	2	Yellow	Blue
	3	Yellow	Black
	4	Yellow	White
B	5	Green	Red
	6	Green	Blue
	7	Green	Black
	8	Green	White

Table 3: Channel number to channel color mapping

1.4 Motomed Commands

The Motomed commands control and receive data from the movement therapy devices of the MOTOmed series by the company RECK.

Training types

There are two training types available to control the MOTOmed: the basic and the phase training. Each mode allows to set speed, gear and rotation direction during the training. Additionally the actual MOTOmed values angle, speed and torque are transmitted to the PC. There is also a command to lock the MOTOmed display keyboard.

Basic training

The basic training allows to control the training using only one parameter (arm/leg trainer). There are commands to start, pause, continue and to stop the training. A disadvantage is that no training data is send to the PC.

Phase training

The phase training allows detailed configuration of the MOTOmed settings and parameters. Before training the phase training needs to be initialized. Thereafter a command starts the first phase. A second phase with different parameters could be started later. The training can be paused using the appropriate command, while another phase restarts it again. This phase concept allows to define many custom training scenarios. After the end of each phase the MOTOmed training data is send to the PC.

1.5 RehaStim2 Graphical User Interface

ScienceMode2 is activated in the start screen of the RehaStim2 software.

The graphical user interface displays

- the initialization parameters of the stimulation.
- the current modes of the Stimulation and Motomed modules.
- the target values (speed, gear, rotation direction) and the actual values (angle, speed, torque)

from Motomed module.
- the status code of the last command.
- the received and the sent commands as well as the error messages in a debug window.
- a yellow title bar during stimulation, a red title bar when an error occurs and a gray title bar when there is no stimulation.

1.6 ScienceMode2 PC Test Software

The PC Test Software can be used for testing and debugging. It runs under Windows XP, Vista, and Windows 7 and has the following features:

- Send protocol commands with parameters and display response commands.
- Show the RehaStim2 information commands.
- Display the received and sent commands as well as the error messages in a debug window.

1.7 Limitation

Note that the interaction with the MOTOmed is defined only if it is fully controlled by the ScienceMode2. Any interaction with the MOTOmed-Cockpit during ScienceMode2 usage can lead to undefined behavior. To avoid any problems, make sure that the keyboard lock is always on (default setting during phase training).

2. Protocol General

2.1 Serial Connection

Use the USB connection cable to connect PC and RehaStim2 (using the PC plug). RehaStim2 has a integrated FTDI chip to transform the USB to a RS-232 signal. Therefore the FTDI driver needs to be installed. The Windows driver is included in the FDTI folder. Using the driver the PC can communicate with RehaStim2 via serial COM port.

After connecting the USB wire to PC and RehaStim2 and installing the FDTI driver, go to the Device Manager and look up the COM-Port for the USB serial device. Use this port for the PC Test software.

It is important to adjust the standard driver settings. In the Device Manager open the USB serial device window and continue to open the Properties window. Navigate to the port settings and then open the advanced window. Set the latency timer to 1 ms.

The serial settings of the ScienceMode2 protocol are listed in Table 4.

Parameter	**Value**
Baud rate	460 800
Data bits	8
Stop bits	1
Parity	Even
Flow Control RTS/CTS	None

Table 4: Serial settings for ScienceMode2 protocol

2.2 Packet Structure

The packet structure is shown in Table 5. Every packet starts with the start byte and ends with the stop byte (see Table 6). The ScienceMode2 protocol uses byte stuffing. Every constant from Table 6, except the stuffing key, is escaped with the stuffing byte from the same table. The value is then XORed with the stuffing key. For example: if the command data has a byte which is identical to the start byte value, this start byte (0xF0) is stuffed to 0x81, 0xA5.

Start byte 1 Byte	Checksum 2 Byte	Data length 2 Byte	Packet number 1 Byte	Command 1 Byte	Command data < 61 Byte	Stop byte 1 Byte
			<— Checksum —>			
			<— Data length —>			

Table 5: Packet structure of the ScienceMode2 commands

The checksum and data length are both 1 byte, but always stuffed, therefore 2 bytes long. They are computed from the stuffed packet number, the command and the command data. The checksum is a CRC-8-CCITT ($x^8 + x^2 + x + 1$; 0x07).

To count the packets there are two packet counter, one in the RehaStim2 and one on the PC side. The PC increases the packet number by one for every new command and the same number is echoed from the RehaStim2 in the response. When the RehaStim2 sends the information packets, it uses its own packet counter.

Constants	Hex	Binary
Start byte	0xF0	11110000
Stop byte	0x0F	00001111
Stuffing byte	0x81	10000001
Stuffing key	0x55	01010101

Table 6: Byte constants

The command is represented by a number for the different commands, responses and information packets, whereas the command data is reserved for the parameter of the command. The ScienceMode2 protocol uses little endian byte order.

2.3 Timeouts / Time Intervals

Name	Time	Description
InitRepetitionTime	500 ms	Defines the interval, in which the RehaStim2 sends the Init command to the PC.
WatchdogTimeout	1200 ms	Defines the time, in which the PC needs to send a valid command packet to the RehaStim2, otherwise the connection. A valid packet is a packet without transfer error. If no command is send, the watchdog command can be sent to RehaStim2.
ActualValuesInterval	20 ms	Defines how often the actual values packets are sent to the PC.
MaxResponseTime	100 ms	Defines the maximum time the RehaStim2 needs to respond to a command packet. (Average response time < 20 ms)

Table 7: Timeout intervals

3. Protocol Commands Overview

3.1 Connection commands

Value	Command	Description
1	Init	Initialize connection
2	InitAck	Response to initialize connection
3	UnknownCommand	Unknown command received (send to PC)
4	Watchdog	Watchdog command

3.2 RehaStim2 Modes

Value	Command	Description
10	GetStimulationMode	Get current Stimulation mode
11	GetStimulationModeAck	Response to get current stimulation mode
12	GetMotomedMode	Get current Motomed mode
13	GetMotomedModeAck	Response to get current Motomed mode

3.3 Stimulation Commands

Value	Command	Description
30	InitChannelListMode	Initialize stimulation (Channel list mode)
31	InitChannelListModeAck	Response to initialize stimulation
32	StartChannelListMode	Start / update stimulation (Channel list mode)
33	StartChannelListModeAck	Response to start / update stimulation
34	StopChannelListMode	Stop stimulation (Channel list mode)
35	StopChannelListModeAck	Response to stop stimulation
36	SinglePulse	Send single pulse
37	SinglePulseAck	Response to send single pulse
38	StimulationError	Send stimulation error (send to PC)

3.4 Motomed Commands

Value	Command	Description
50	InitPhaseTraining	Initialize phase training
51	InitPhaseTrainingAck	Response to initialize phase training
52	StartPhase	Start phase training / change phase
53	StartPhaseAck	Response to start phase training / change phase
54	PausePhase	Start pause phase

55	PausePhaseAck	Response to start pause phase
56	StopPhaseTraining	Stop phase and phase training
57	StopPhaseTrainingAck	Response to stop phase and phase training
58	PhaseResult	Send the results of the last phase (send to PC)
60	ActualValues	Send actual values (send to PC)
70	SetRotationDirection	Set rotation direction during the training
71	SetRotationDirectionAck	Response to set rotation direction during the training
72	SetSpeed	Set passive speed during the training
73	SetSpeedAck	Response to set passive speed during the training
74	SetGear	Set gear during the training
75	SetGearAck	Response to set gear during the training
76	SetKeyboardLock	Set / unset MOTOmed keyboard lock
77	SetKeyboardLockAck	Response to set / unset MOTOmed keyboard lock
80	StartBasicTraining	Start basic training
81	StartBasicTrainingAck	Response to start basic training
82	PauseBasicTraining	Pause basic training
83	PauseBasicTrainingAck	Response to pause basic training
84	ContinueBasicTraining	Continue basic training after break
85	ContinueBasicTrainingAck	Response to continue basic training after break
86	StopBasicTraining	Stop basic training
87	StopBasicTrainingAck	Response to stop basic training
89	MotomedCommandDone	Last Motomed command processed (new Motomed command can be send) (send to PC)
90	MotomedError	Send Motomed error (send to PC)

3.5 Communication Direction of Commands

Initialization of Connection

Communication	Description
RehaStim2 —> PC PC —> RehaStim2	• RehaStim2 sends Init command • PC sends InitAck response

Information commands

Communication	Description
RehaStim2 —> PC	• RehaStim2 sends ActualValues, TrainingData StimulationError, MotomedCommandDone

Command / Response Communication (all other commands)

Communication	Description
PC —> RehaStim2 RehaStim2 —> PC	• PC sends Command • RehaStim2 sends CommandAck

3.6 Modes and Commands Overview

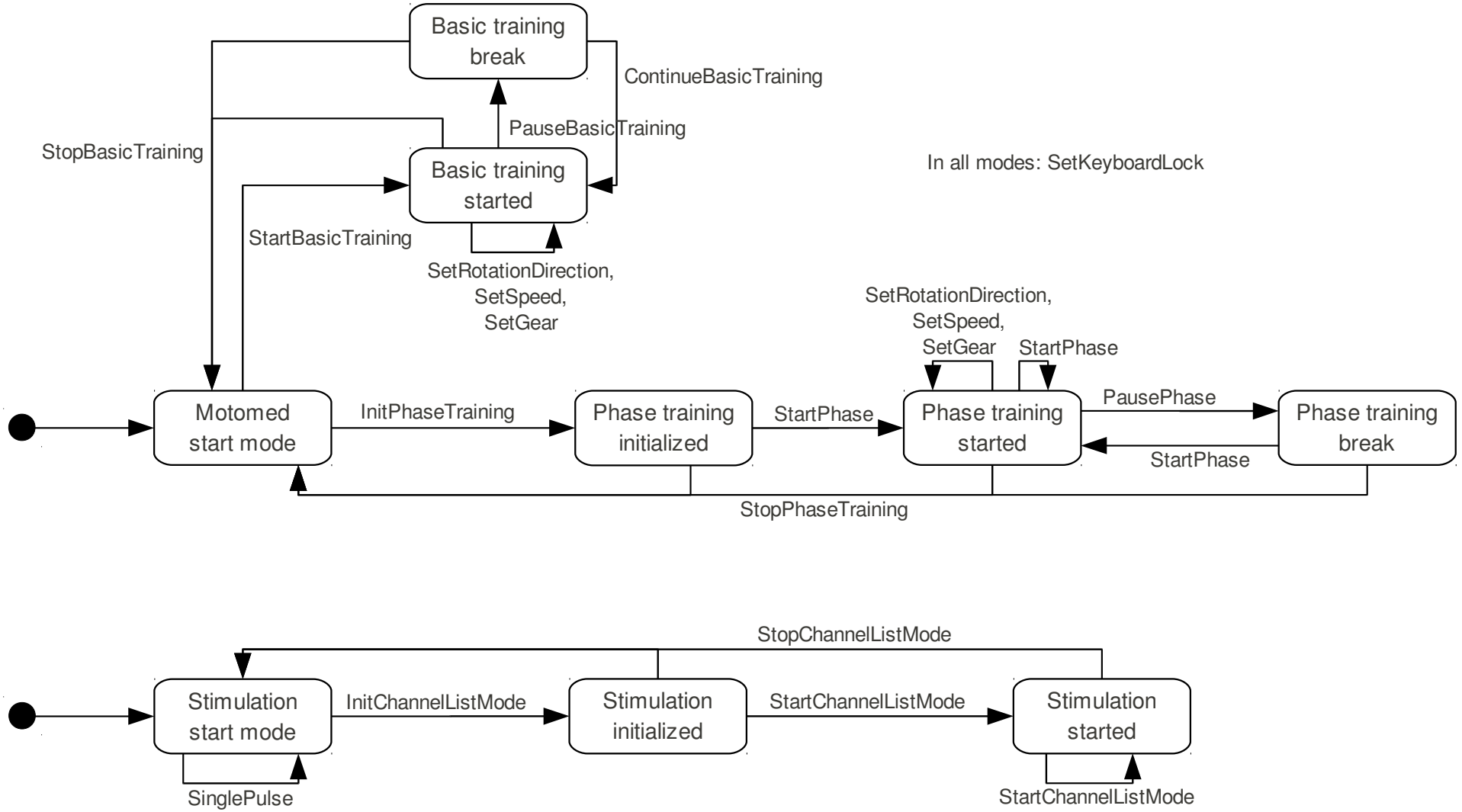

Figure 4: Modes and commands overview

4. Protocol Commands Details

The First Bytes

(for every command)

0	Start byte
1	Stuffing byte
2	Checksum (stuffed)
3	Stuffing byte
4	Data length (stuffed)

4.1 Initialization of Connection Commands

Init

Initialize connection

5	Packet number	Range: [0, 255]
6	Command	Init
7	Version number	Current version number of the protocol
8	Stop byte	

InitAck

Response to initialize connection

5	Echo packet number	Range: [0, 255]
6	Command	InitAck
7	Result	0: Version number accepted, connection established -5: Incompatible version number
8	Stop byte	

UnknownCommand

Unknown command received (send to PC)

5	Echo packet number	Range: [0, 255]
6	Command	UnknownCommand
7	Echo command byte	Range: [0, 255] Echos the received command value
8	Stop byte	

Watchdog

Watchdog command

5	Packet number	Range: [0, 255]
6	Command	Watchdog
7	Stop byte	

4.2 RehaStim2 Mode Commands

GetStimulationMode

Get current Stimulation mode

5	Packet number	Range: [0, 255]
6	Command	GetStimulationMode
7	Stop byte	

GetStimulationModeAck

Response to get current Stimulation mode

5	Echo packet number	Range: [0, 255]
6	Command	GetStimulationModeAck
7	Result • if Result ≠ 0, then byte 8 = stop byte	0 Successful -1 Transfer error -8 Busy error
8	Stimulation mode	0 Start mode 1 Stimulation initialized 2 Stimulation started
9	Stop byte	

GetMotomedMode

Get current Motomed mode

5	Packet number	Range: [0, 255]
6	Command	GetMotomedMode
7	Stop byte	

GetMotomedModeAck

Response to get current Motomed mode

5	Echo packet number	Range: [0, 255]
6	Command	GetMotomedModeAck

<table>
<tr><td>7</td><td>Result
• if Result ≠ 0, then byte 8 = stop byte</td><td><table><tr><td>0</td><td>Successful</td></tr><tr><td>-1</td><td>Transfer error</td></tr><tr><td>-8</td><td>Busy error</td></tr></table></td></tr>
<tr><td>8</td><td>Motomed mode</td><td><table><tr><td>0</td><td>Start mode</td></tr><tr><td>1</td><td>Phase training initialized</td></tr><tr><td>2</td><td>Phase training started</td></tr><tr><td>3</td><td>Phase training break</td></tr><tr><td>4</td><td>Basic training started</td></tr><tr><td>5</td><td>Basic training pause</td></tr><tr><td>6</td><td>Motomed busy</td></tr><tr><td>-1</td><td>Motomed connection error</td></tr></table></td></tr>
<tr><td>9</td><td>Stop byte</td><td></td></tr>
</table>

4.3 Stimulation Commands

InitChannelListMode

Initialize stimulation (Channel list mode)

5	Packet number	Range: [0, 255]
6	Command	InitChannelListMode
7	Low frequency factor	Range: [0, 7] 0 = no skip .. 7 = skip seven times
8	Active channels	Range: [0, 255] Bit 0 corresponds to channel 1. .. Bit 7 corresponds to channel8.
9	Active low frequency channels	Range: [0, 255] Bit 0 corresponds to channel 1. .. Bit 7 corresponds to channel8.
10	Inter pulse interval coded	Range: [0, 255] Inter pulse interval = [0, 255] · 0.5 ms + 1.5 = [1.5, 129] ms Note that in the current software version the minimum inter pulse interval is 8 ms.
11	Main stimulation interval coded MSB	Range: [0, 2048] = 0 One shot channel list mode > 0 Continuous channel list mode

		Main stimulation interval = [0, 2048] · 0.5 ms + 1 ms = [1, 1025] ms Note that in the current software version the minimum main stimulation interval is 8 ms.
12	Main stimulation interval coded LSB	
13	Channel execution	0 – Fixed interval 1 – As fast as possible
14	Stop byte	

InitChannelListModeAck

Response to Initialize stimulation

5	Echo packet number	Range: [0, 255]
6	Command	InitChannelListModeAck
7	Result	0 – Stimulation initialized -1 – Transfer error -2 – Parameter error -3 – Wrong mode error -8 – Busy error
8	Stop byte	

StartChannelListMode

Start / update stimulation (Channel list mode)

5	Packet number	Range: [0, 255]
6	Command	StartChannelListMode
For each active channel, 4 bytes should be send in increasing order.		
7	Mode	0 – Single pulse 1 – Doublet 2 – Triplet
8	Pulse width MSB	Range: [0, 500] μs (in current version [20, 500] µs, if (pw < 20) then pw = 20)
9	Pulse width LSB	
10	Current	Range: [0, 130] mA
...		
< 40	Stop byte	

StartChannelListModeAck

Response to start / update stimulation

5	Echo packet number	Range: [0, 255]
6	Command	StartChannelListMode
7	Result	0: Sent start stimulation / new parameters to stimulation units -1: Transfer error -2: Parameter error -3: Wrong mode error -8: Busy error
8	Stop byte	

StopChannelListMode

Stop stimulation (Channel list mode)

5	Packet number	Range: [0, 255]
6	Command	StopChannelListMode
7	Stop byte	

StopChannelListModeAck

Response to stop stimulation

5	Echo packet number	Range: [0, 255]
6	Command	StopChannelListModeAck
7	Result	0: Stimulation stopped -1: Transfer error
8	Stop byte	

SinglePulse

Send single pulse

5	Packet number	Range: [0, 255]
6	Command	SinglePulse
7	Channel	Range: [0, 7]
8	Pulsweite MSB	Range: [0, 500] in μs (in current version [20, 500] μs, if (pw < 20) then pw = 20)
9	Pulsweite LSB	
10	Current	Range: [0-130] in mA
11	Stop byte	

SinglePulseAck

Response to send single pulse

5	Echo packet number	Range: [0, 255]
6	Command	SinglePulseAck
7	Result	0: Single pulse sent to stimulation module -1: Transfer error -2: Parameter error -3: Wrong mode error -8: Busy error
8	Stop byte	

StimulationError

Stimulation error (send to PC)

5	RehaStim2 packet number	Range: [0, 255]
6	Command	StartChannelListModeAck
7	Error	-1: Emergency switch activated / not connected -2: Electrode error -3: Stimulation module error
8	Stop byte	

4.4 Motomed Commands

InitPhaseTraining

Initialize phase training

5	Packet number	Range: [0, 255]
6	Command	InitPhaseTraining
7	Trainer	0: Leg training 1: Arm training
8	Stop byte	

InitPhaseTrainingAck

Response to initialize phase training

5	Echo packet number	Range: [0, 255]
6	Command	InitPhaseTrainingAck

7	Result	0 Phase training initialized -1 Transfer error -2 Parameter error -3 Wrong mode error -4 Motomed connection error -7 Motomed busy error -8 Busy error
8	Stop byte	

StartPhase

Start phase training / change phase

5	Packet number	Range: [0, 255]
6	Command	StartPhase
7	Phase variant	0 Standard active 1 Standard passive 2 Symmetry training 3 MOTOmax game
8	Passive speed U/min (Cadence)	Range: • [0, 60] Standard devices • [0, 90] Parkinson devices
9	Gear	Range: [0, 20]
10	Rotation Direction	0 Backward 1 Forward
11	Fly wheel	Range: [0, 100]
12	Spasm detection	0 off 1 on, without direction restoration 2 on, with direction restoration
13	Training side	0 Both side 1 Left side 2 Right side
14	Crank orientation	0 diagonal 1 equal
15	Stop byte	

StartPhaseAck

Response to start phase training / change phase

5	Echo packet number	Range: [0,255]
6	Command	StartPhaseAck
7	Result	0: Start phase training / change phase sent to MOTOmed -1: Transfer error -2: Parameter error -3: Wrong mode error -4: Motomed connection error -7: Motomed busy error -8: Busy error
8	Stop byte	

PausePhase

Start pause phase

5	Packet number	Range: [0, 255]
6	Command	PausePhase
7	Stop byte	

PausePhaseAck

Response to start pause phase

5	Echo packet number	Range: [0-255]
6	Command	PausePhaseAck
7	Result	0: Start pause sent to MOTOmed -1: Transfer error -2: Parameter error -3: Wrong mode error -4: Motomed Connection error -7: Motomed busy error -8: Busy error
8	Stop byte	

StopPhaseTraining

Stop phase and phase training

5	Packet number	Range: [0-255]
6	Command	StopPhaseTraining
7	Stop byte	

StopPhaseTrainingAck

Response to stop phase and phase training

5	Echo packet number	Range: [0, 255]
6	Command	StopPhaseTrainingAck
7	Result	0 – Stop phase training sent to MOTOmed -1 – Transfer error -2 – Parameter error -3 – Wrong mode error -4 – Motomed Connection error -7 – Motomed busy error -8 – Busy error
8	Stop byte	

PhaseResult

Send the results of the last phase (send to PC)

5	RehaStim2 packet number	Range: [0, 255]
6	Command	PhaseResult
7	Phase counter	Range: [0, 255]
8	Passive distance MSB	in 10 m
9	Passive distance LSB	
10	Active distance MSB	in 10 m
11	Active distance LSB	
12	Average power	in W
13	Maximum power	in W
14	Phase duration MSB	in s
15	Phase duration LSB	
16	Active phase duration MSB	in s
17	Active phase duration LSB	
18	Phase work MSB	in J
19	Phase work LSB	
20	Success value	in %

21	Symmetry (left)	in %
22	Average muscle tone	in dNm
23	Stop byte	

ActualValues

Send actual values (send to PC)

5	RehaStim2 packet number	Range: [0, 255]
6	Command	ActualValues
7	Angle MSB	Represents the current angle in ° starting from the 12 o'clock position of the left crank from user's perspective.
8	Angle LSB	
9	Speed MSB	Represents the current speed in rpm. The sign defines the direction. • + Forward • - Backward
10	Speed LSB	
11	Torque MSB	Represents the current torque in dNm. The sign defines the activity. • + Active • - Passive
12	Torque LSB	
13	Stop byte	

SetRotationDirection

Set rotation direction during the training

5	Packet number	Range: [0, 255]
6	Command	SetRotationDirection
7	Rotation direction	0 Backward 1 Forward
8	Stop byte	

SetRotationDirectionAck

Response to set rotation direction during the training

5	Echo packet number	Range: [0, 255]
6	Command	SetRotationDirectionAck

7	Result	0 Sent rotation direction to MOTOmed -1 Transfer error -2 Parameter error -3 Wrong mode error -4 Motomed connection error -7 Motomed busy error -8 Busy error
8	Stop byte	

SetSpeed

Set passive speed during the training

5	Packet number	Range: [0, 255]
6	Command	SetSpeed
7	Passive speed U/min (Cadence)	Range: • [0, 60] Standard devices • [0, 90] Parkinson devices
8	Stop byte	

SetSpeedAck

Response to set passive speed during the training

5	Echo packet number	Range: [0, 255]
6	Command	SetSpeedAck
7	Result	0 Sent speed to MOTOmed -1 Transfer error -2 Parameter error -3 Wrong mode error -4 Motomed connection error -7 Motomed busy error -8 Busy error
8	Stop byte	

SetGear

Set gear during training

5	Packet number	Range: [0, 255]
6	Command	SetSpeed
7	Gang	Range: [0, 20]

8	Stop byte	

SetGearAck

Response to set gear during training

5	Echo packet number	Range: [0, 255]
6	Command	SetGearAck
7	Result	0 Set Gear to MOTOmed -1 Transfer error -2 Parameter error -3 Wrong mode error -4 Motomed connection error -7 Motomed busy error -8 Busy error
8	Stop byte	

SetKeyboardLock

Set / unset MOTOmed keyboard lock

5	Packet number	Range: [0, 255]
6	Command	SetKeyboardLock
7	Keyboard lock	0 off 1 on
8	Stop byte	

SetKeyboardLockAck

Response to set / unset MOTOmed keyboard lock

5	Echo packet number	Range: [0, 255]
6	Command	SetKeyboardLockAck
7	Result	0 Sent Keyboard lock to MOTOmed -1 Transfer error -2 Parameter error -3 Wrong mode error -4 Motomed connection error -7 Motomed busy error -8 Busy error
8	Stop byte	

StartBasicTraining

Start basic training

5	Packet number	Range: [0, 255]
6	Command	StartBasicTraining
7	Trainer	0 Leg training 1 Arm training
8	Stop byte	

StartBasicTrainingAck

Response to start basic training

5	Echo packet number	Range: [0, 255]
6	Command	StartBasicTrainingAck
7	Result	0 Sent start basic training to MOTOmed -1 Transfer error -2 Parameter error -3 Wrong mode error -4 Motomed connection error -7 Motomed busy error -8 Busy error
8	Stop byte	

PauseBasicTraining

Pause basic training

5	Packet number	Range: [0, 255]
6	Command	PauseBasicTraining
7	Stop byte	

PauseBasicTrainingAck

Response to pause basic training

5	Echo packet number	Range: [0, 255]
6	Command	PauseBasicTrainingAck
7	Result	0 Sent basic pause to MOTOmed -1 Transfer error -3 Wrong mode error -4 Motomed connection error

		-7 Motomed busy error -8 Busy error
8	Stop byte	

ContinueBasicTraining

Continue basic training

5	Packet number	Range: [0, 255]
6	Command	ContinueBasicTraining
7	Stop byte	

ContinueBasicTrainingAck

Response to continue basic training

5	Echo Packet number	Range: [0, 255]
6	Command	ContinueBasicTrainingAck
7	Result	0 Sent continue basic training to MOTOmed -1 Transfer error -3 Wrong mode error -4 Motomed Connection error -7 Motomed busy error -8 Busy error
8	Stop byte	

StopBasicTraining

Stop basic training

5	Packet number	Range: [0, 255]
6	Command	StopBasicTraining
7	Stop byte	

StopBasicTrainingAck

Response to stop basic training

5	Echo packet number	Range: [0, 255]
6	Command	StopBasicTrainingAck
7	Result	0 Sent stop basic training to MOTOmed -1 Transfer error -3 Wrong mode error

		-4 Motomed connection error -7 Motomed busy error -8 Busy error
8	Stop byte	

MotomedCommandDone

Last Motomed command processed (new Motomed command can be send)

5	Packet number	Range: [0, 255]
6	Command	MotomedCommandDone
7	Result	0 MOTOmed has successfully processed the last Motomed command.
8	Stop byte	

MotomedError

Motomed error (send to PC)

5	RehaStim2 packet number	Range: [0, 255]
6	Command	MotomedError
7	Error	-4 Motomed connection error -6 Invalid Motomed trainer
8	Stop byte	

5. Error Handling

5.1 Error Codes for the Response Commands

Error code	Error	Description
0	No error	No errors have been occurred.
-1	Transfer error	The transferred checksum or data length does not match with the received packet.
-2	Parameter error	The command parameters are not in the allowed range. This error is also sent, if there are too many or too few parameters in the packet.
-3	Wrong mode error	The command can not be processed in the current mode.
-4	Motomed connection error	RehaStim2 has no connection to the MOTOmed device.

-5	Incompatible protocol version	The PC sends this to RehaStim2, if it does not accept the protocol version number.
-6	Invalid Motomed trainer	The selected trainer is not available on the MOTOmed device.
-7	Motomed busy error	MOTOmed is currently processing the last Motomed command. Wait for the MotomedCommandDone packet.
-8	Busy error	RehaStim2 is currently processing the last command. Wait for the response packet.
separate Packet	Unknown command	The received command number is invalid.

5.2 Error Detection

Error	Error detection	Reaction
Frame error	Hardware	The received bits are rejected.
Parity error	Hardware	The received bits are rejected.
Overflow	Hardware	The additional bits are rejected.
Packet frame error	A packet is framed by a start and a stop byte. The error is detected if after a start byte a second start byte is found.	The received bytes are rejected up to the second start byte.
Data length error	The length written in the packet does not match with transferred length,	The response (Ack) of the command is sent with the transfer error code.
Checksum error	The checksum written in the packet does not match with the computed checksum of the transferred packet.	The response (Ack) of the command is sent with the transfer error code.
Unknown command error	The packet contains an invalid command number.	The command UnknownCommand is sent.
Init connection error	There is no response to the Init command.	The Init command is sent again after InitRepetitionTime.
Wrong mode error	The received command can not be processed in the current mode.	The response (Ack) of the command is sent with the wrong mode error.
Too many / few parameters	The received packet contains too many or too few parameters.	The response (Ack) of the command is sent with the parameter error code.
Invalid parameter values	The received command contains invalid parameter values.	The response (Ack) of the command is sent with the parameter error code.
Data transfer physically broken / disturbed	RehaStim2 does not receive a valid packet within the WatchdogTimeout.	• Stimulation and Motomed training are stopped.

		• All modes are reset. • RehaStim2 sends the Init command.

5.3 Communication errors

Initialization of Connection

Communication	Error description	Solution
RehaStim2 –!–> PC	• RehaStim2 sends Init. • Init is not received by PC or has a transfer error.	• RehaStim2 repeats Init after InitRepetitionTime.
RehaStim2 —> PC PC –!–> RehaStim2	• RehaStim2 sends Init. • PC sends InitAck. • InitAck is not received by RehaStim2 or has a transfer error.	• RehaStim2 repeats Init after InitRepetitionTime.

Information commands

Communication	Error description	Solution
RehaStim2 –!–> PC	• RehaStim2 sends Actual-Values, TrainingData, StimulationError, MotomedCommandDone • Packets are not received by PC or have transfer errors.	• The PC give up on the packets and if necessary interpolate the actual values. • The PC checks current modes from RehaStim2 to determine its next behavior.

Command / Response Communication

Communication	Error description	Solution
PC –!–> RehaStim2	• PC sends Command. • Command packet is not received by PC or has transfer error.	• PC waits for the response up to the MaxResponseTime. • PC checks the current mode and if necessary sends the command again.
PC –!–> RehaStim2 RehaStim2 –!–> PC	• PC sends Command. • RehaStim2 sends CommandAck. • Response is not received by PC or has transfer error.	• PC waits for the response up to the MaxResponseTime. • PC checks the current mode and if necessary sends the command again.

6. Disclaimer

The software included is provided by the copyright holders and contributors "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for particular purpose are disclaimed. In no event shall the foundation or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.