Nano Core Interface Description v2 (v2.0.0.1678)

IDD

1002 - Nano Core

Author(s)			
Function:	Software Engineer	Function:	Software Engineer
Name:	Jos Vos	Name:	Dawid Zalewski
Date (yyyy-mm-dd)	Signature	Date (yyyy-mm-dd)	Signature

Document Approval			
Function:	Mechatronic System Engineer	Function:	
Name:	Mark-Willem Jansen	Name:	
Date (yyyy-mm-dd)	Signature	Date (yyyy-mm-dd)	Signature
Function:		Function:	
Name:		Name:	
Date (yyyy-mm-dd)	Signature	Date (yyyy-mm-dd)	Signature

Document Release	
Function:	Mechatronic System Engineer
Name:	Mark-Willem Jansen
Date (yyyy-mm-dd)	Signature

Document Identification and Filing		
Product:	Version:	Status:
1002	**1.0**	**Released**
Project:	Dossier:	Doc. ID:
522.08	**DHF**	**PRD1002-4-123**

Change log			
Version	Date (YYYY-MM-DD)	Author	Description of change(s)
~~1.0~~	~~2017-04-25~~	~~SLI~~	~~Duplicate of D1129953_IDD Nano Core Communication Protocol_00_11~~
~~1.1~~	~~2017-05-18~~	~~DZA~~	~~First draft Nano Core OEM~~
~~1.2~~	~~2017-05-23~~	~~DZA~~	~~Minor updates to draft~~
~~1.3~~	~~2017-05-24~~	~~DZA~~	~~Changed BeatToBeat OEM messages format~~
~~1.4~~	~~2017-05-29~~	~~DZA~~	~~Minor updates to draft~~
~~1.5~~	~~2017-06-12~~	~~DZA~~	~~Changed Status message, changed BeatToBeat OEM messages~~
~~1.6~~	~~2017-06-20~~	~~DZA~~	~~Draft ready for review~~
~~1.7~~	~~2017-06-22~~	~~DZA~~	~~Processed review comments~~
~~1.8~~	~~2017-06-29~~	~~DZA~~	~~Processed review comments~~
~~1.9~~	~~2017-07-07~~	~~MJN~~	~~Document prepared for review~~
~~1.10~~	~~2017-07-07~~	~~DZA~~	~~Moved bootloader command to separate IDD~~
~~1.11~~	~~2017-07-20~~	~~BZW~~	~~Changed BraCal abort message Error to Warning~~
~~1.12~~	~~2017-07-27~~	~~BZW~~	~~Added extra status in ModelFlow which indicated if a BraCal can be performed.~~
~~1.13~~	~~2017-08-03~~	~~DZA~~	~~Moved the information about whether BraCal is allowed from the status field to a separate bit.~~
~~1.14~~	~~2017-08-03~~	~~DZA~~	~~Changed the Zero HCU command specs to match the implementation~~
~~1.15~~	~~2017-09-21~~	~~BZW~~	~~Added error for driver failure (introduced in software version v1.2.4)~~
~~1.16~~		~~BZW~~	
~~1.17~~	~~2017-11-02~~	~~BZW~~	~~Restored to version 1.15~~
~~2.1~~	~~2017-11-15~~	~~MJN~~	~~Removed 0xFF response of 'z' command.~~
0.1	2017-11-16	BZW	Created new document for v2.0.0.1678 of the firmware.
0.2	2017-11-16	SLI	Review comments.
1.0	2017-11-17	BZW	Process review commets.

Review Status			
Version before review	Review date (YYYY-MM-DD)	Reviewers	Company
0.1	2017-11-16	SLI	DEMCON
Result of review	Accepted after processing of comments		

Former Profile History

Rev_ver	Date	Author	Description
00_01	10-06-2014	Raymond Otte	Initial version
00_01	2-9-2014	WBO	Change setmode to execute command
00_02	5-11-2014	JVS	Reworked a lot of commands to simplify behavior. Added bandwidth calculations.
00_03	24-02-2015	JVS	Added error and warnings. Reworked versioning
00_04	18-02-2016	RJE	Service-tests: add units of results-values
00_05	11-03-2016	JVS	Added received ID to the bootloader nack.
00_06	11-04-2016	RJE	Fill in review data, replace 'NanoCore' by 'Nano Core', correct some typos
00_07	11-05-2016	RJE	Add physical status to DataTransmission command
00_08	20-05-2016	JVS	Added production tests
00_09	07-06-2016	JVS	Added accelerometer test and some small updates to

			other tests.
00_10	28-11-2016	JVS	Updated behavior of beat to beat packet during specific conditions.
00_11	23-01-2017	JVS	Updated beat to beat packet with additional information

Former review status

Rev_ver before review	Review date	Reviewers	Company
00_05	11-04-2016	RJE	DEMCON
Result of review:	~~Rejected~~ / Accepted after processing of comments / ~~Accepted~~		
00_11	23-01-2017	MJN	DEMCON
Result of review:	~~Rejected~~ / ~~Accepted after processing of comments~~ ~~/ Accepted~~		

References

Number	Author(s)	Description
[1]		

Abbreviations

Abbreviation	Description
IDD	Interface Design Description

Definitions

Definition	Description
DEMCON	DEMCON Advanced Mechatronics B.V.

TABLE OF CONTENTS

1. INTRODUCTION

This document describes the protocol that is implemented on the Finapres Nano Core. This is used to interface display devices to the Nano Core and control the Nano Core operations.

2. COMMUNICATION PROTOCOL

2.1 Hardware configuration

The serial port of the Nano Core is configured as follows:

- 115200 Baud rate
- 1 stop byte
- No flow control

2.2 Global message layout

STX 1 byte	LEN 1 byte	LEN 1 byte	STX 1 byte	cmd 1 byte	cmd-data … bytes	CRC 1 byte

2.2.1 STX, character

Detecting and differentiating separate messages is done using the STX, LEN, LEN, STX. Note that LEN is a single byte which is sent 2 times. Maximum length is therefore limited to 255 bytes. Implementation can use the following easy rules:

- STX = 0xD4
  LEN = the total number of bytes of cmd + cmd-data
  receive STX, LEN, LEN, STX → possible start of message.
- Receive LEN byte and check if the CRC over these bytes matches the CRC at the end of the message.

2.2.2 Cmd and cmd-data

Each message has a cmd, followed by cmd-data. The cmd identifies the type of message. The amount and type of cmd-data differs per cmd, and can also be 0. The cmds are defined separately.

All messages sent from the PC (except the get status message) will be acknowledged by the Nano. This is done by sending a message back. With the same cmd to ACK the message, or with (cmd OR 0x80) to NACK the message, right after the NACK-ed cmd there will be an error code. See 2.5.2

Messages sent by the µC are not ACK by the PC.

2.2.3 Error-checking

Every message has a CRC-check in order to detect transmission failures. The STX and LEN are excluded from the CRC-calculations (if STX or LEN are not received correctly, the message is lost anyway, since we are not able to detect it anyway). We choose an 8-bit CRC with the following polynomial (CRC-8 Dallas/Maxim [1]):

$CRC = x^8 + x^5 + x^4 + x^0$

If a CRC-check fails, the message is simply discarded. If a message is important, it is indicated in the protocol that the receiver should send a response. If such a response times out at the sender-side, it can be resent. After a multiple or time-outs, the receiver can be indicated as 'connection lost'.

[1] See http://www.ross.net/crc/download/crc_v3.txt and http://en.wikipedia.org/wiki/Cyclic_redundancy_check#Commonly_used_and_standardized_CRCs

2.3 Byte ordering

Multi-byte values are sent in 'little endian' ordering (standard Intel format), i.e. 'least significant byte' first. So a 4-byte value 0x12345678 is sent as {0x78, 0x56, 0x34, 0x12}.

2.4 Operating Modes

The Finapres Nano protocol can be in the following modes:

Boot self-test
Nano is starting up, doing selftest, etc. After starting up the Nano will automatically go to Idle Mode, if self test found a un recoverable error the Nano will go to Error state.
All commands are readonly in this mode.

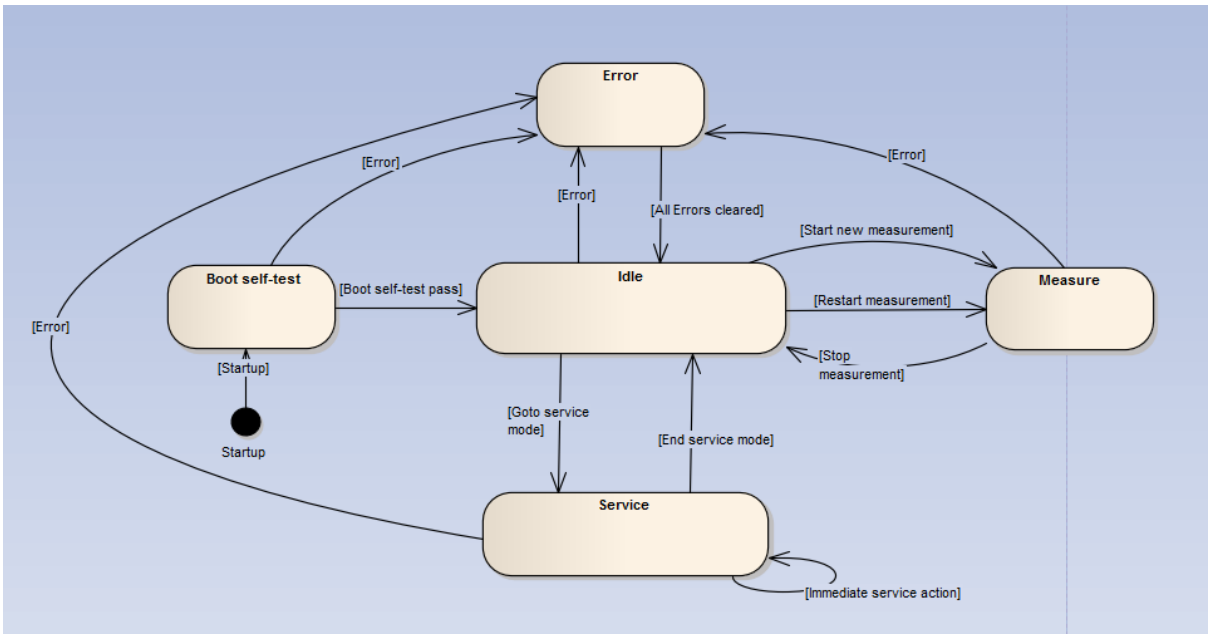
Idle
No active measurement or service, measurements can be started and services can be started.

Measure
Active measurement is running.

Error
An error has ocured, this needs to be cleared in order to get back to idle.

Maintainance
Maintainance can be started, starting and stopping diagnostics.

BootLoader
Mode is not shown in the figure below, but is reachable when an enter bootloader command is sent. This command can be sent from Idle and Error



Not all commands are available in all modes, the table below shows which commands are available in each mode.

	Boot self-test	Idle	Measure	Service	Boot loader	Error
Version Info	✓	✓	✓	✓	✓	✓
Alive			✓	✓		
Status	✓	✓	✓	✓		✓
StatusUpdate	✓	✓	✓	✓		
DataTransmission			✓	✓		
B2BDataTransmission			✓			
Mode	readonly	✓	✓	✓	read only	✓
PatientData		✓	readonly	readonly		readonly
CuffUsage		✓	readonly	readonly		readonly
ZeroHCU		✓	✓	✓		
PhysioCal			✓			
MaintenanceTest				✓		
FirmwareFlashStart					✓	
FirmwareData					✓	
Hardware Info Erase					✓	
Hardware Info Data					✓	

2.5 Reused data types

2.5.1 Mode [1 byte]

Mode							
Main-mode				*Sub-mode*			*Transition*
7	6	5	4	3	2	1	0

Bit 7-4 = Main-mode,
- 0b0000 = Starting up
- 0b 0001 = Idle
- 0b 0011 = Measure
- 0b 0100 = Service
- 0b 0111 = Boot loader
- 0b 1111 = Error

Bit 1-3 = reserved for Sub-mode
- Depending on which mode the device is in, the sub-mode is changed.
- If mode is Bootloader
- 0b000 = Bootloader Idle
- 0b001 = Bootloader Flash Erased
- 0b010 = Bootloader Board Erased

Bit 0 = indicates if the device is in a mode transition,
- 0 = No transition going on and mode/sub-mode is current mode,
- 1 = Transition to mode/sub mode is going on.

2.5.2 Nack error code [1 byte]

Nack error code							
7	6	5	4	3	2	1	0

Error codes:

0x01	Bootloader: out of order packet
0x02	Bootloader: flash not started for some reason
0x07	Current message is not allowed
0x08	Parameter-value is out of range
0xFC	Message data length incorrect
0xFD	Not implemented
0xFE	Not supported message ID
0xFF	Unknown message ID

2.5.3 Error code [1 byte]

Internal	*Error code*						
7	6	5	4	3	2	1	0

- Error code (bits 0 – 6). Error codes need to be cleared individually, after the first error is cleared, more errors can be present, and will also have to be cleared.
- Internal, flag indicating the responsibility ownership for the error:
  0= the error must be cleared by the host
  1= internal error; it needs not to be cleared. Nano Core is not in the Error Mode and is performing some other tasks. An attempt to clear the error will be made by Nano Core.

```
NoError                                 =  0,
GeneralError                            =  1,
LedContr_erro_LowControlVolt            =  2,
LedContr_erro_HighControlVolt           =  3,
LedContr_erro_HighLedCurrent            =  4,
LedContr_erro_TooManyIteratio           =  5,
LedContr_erro_CurrBelowDrift            =  6,
LedContr_erro_CurrAboveDrift            =  7,
LedContr_erro_VoltBelowRange            =  8,
LedContr_erro_VoltAboveRange            =  9,
Plethysm_erro_TooMuchLight              = 10,
PhysScan_erro_SyncBeatTimeOut           = 11,
PhysScan_erro_ScanFailed                = 12,
PhysScan_erro_BeatDownTimeOut           = 13,
PresMoni_erro_IncorrectPress            = 14,
PresMoni_erro_UnstablePress             = 15,
ManoBeDe_erro_PressTooLow               = 16,
SignInMo_erro_MeanPressLow              = 17,
SignInMo_erro_UnacceptableP             = 18,
SignInMo_erro_MeanPressHigh             = 19,
PreContr_erro_CuffPreSensRang           = 20,
PreContr_erro_VolPreSensRang            = 21,
PreContr_erro_CuffPreExceed             = 22,
PreContr_erro_CuffPreExceLong           = 23,
PreContr_erro_VolPreExceed              = 24,
PreContr_erro_CurrExceedLong            = 25,
SysIntgr_erro_VoltageSenseFailure       = 26,
SysIntgr_erro_HcuRefSenseFailure        = 27,
SysIntgr_erro_PletRefSenseFailure       = 28,
SysIntgr_erro_HouseTempSenseFailure     = 29,
SysIntgr_erro_cuffPressureSenseFailure  = 30,
SysIntgr_erro_volumePressureSenseFailure= 31,
SysIntgr_erro_VoltageOutLimits          = 32,
SysIntgr_erro_HcuRefOutLimits           = 33,
SysIntgr_erro_PletRefOutLimits          = 34,
SysIntgr_erro_cuffPressureSignalOffset  = 35,
SysIntgr_erro_volumePressureSignalOffset= 36,
SysIntgr_erro_pressureSensorTimeout     = 37,
AppContr_erro_PressureToHigh            = 39,
AppContr_erro_caseTemperatureOutLimits  = 40,
AppContr_erro_pcbTemperatureOutLimits   = 41,
AppContr_erro_MeasurementToLong         = 42,
HcuContr_erro_hcuOffsetToBig            = 43,
HcuContr_erro_NotAllowed                = 44,
AppContr_erro_KeepAliveNotReceived      = 45,
Driver_erro_SensorFailed                = 46
```

2.5.4 Warning code [4 bytes]

Warning code																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Warning codes: Copied directly from ErrorStatus.h

```
NoWarning                          = 0x00      ,
GeneralWarning                     = 0x01      ,
PhysScan_warn_NewScanWithAHB       = 0x01 << 1 ,
PhysAdju_warn_BeatUpTimeOut        = 0x01 << 2 ,
PhysAdju_warn_BeatDownTimeOut      = 0x01 << 3 ,
ManoBeDe_warn_PulseVeryLow         = 0x01 << 4 ,
ManoBeDe_warn_NoPulse              = 0x01 << 5 ,
SignInMo_warn_DecreasePletSp       = 0x01 << 6 ,
OsciCont_warn_DecreasePletSp       = 0x01 << 7 ,
PreContr_warn_BadStart             = 0x01 << 8 ,
PreContr_warn_I2T_Protection       = 0x01 << 9 ,
PreContr_warn_HighCurrent          = 0x01 << 10,
PreContr_warn_PlungerPosEnd        = 0x01 << 11,
PreContr_warn_TrackingError        = 0x01 << 12,
PreContr_warn_PowerLimit           = 0x01 << 13,
SysIntgr_warn_PressureSensorTrend = 0x01 << 14,
AppContr_warn_MeasurementLong      = 0x01 << 15,
ModFlow_warn_BraCalLong            = 0x01 << 16,
ModFlow_warn_ BraCalLongAborted     = 0x01 << 17
```

2.5.5 Status [13 bytes]

Status Nano Core OEM [13]									
Mode 1 Byte	Error 1 Byte	Warning 4 Bytes	Misc 1 Byte	Cuff 1 Byte	Physiocal 1 Byte	BeatsTill Physiocal 1 Byte	Physiocal Interval 1 Byte	Cuff Control 1 Byte	Model Flow 1 Byte

- Mode, see 2.5.1
- Error, see 2.5.2 for error codes. If bit 7 of the error code is set, the error needs not to be cleared by host.
- Warning code, see 2.5.4
- Status misc,

Misc							
HCU			*HCU Settings*		*Spare status bits*		
7	6	5	4	3	2	1	0

  - HCU
    000 = HCU not connected
    001 = HCU not zeroed
    010 = HCU zeroed
    011 = HCU zeroed (uncertain), could happen if HCU is disconnected and connected again, height could be incorrect because we don't know if same HCU is reconnected
    100 = HCU is zeroing at the moment

- Status Cuff,

Cuff							
MinutesTillSwitch						*Current Cuff*	
7	6	5	4	3	2	1	0

  - MinutesTillSwitch,
    0x00 = Automatic cuff control is disabled
    0x3e = Switching (or enabling) cuff now
    otherwise = Minutes until next automatic cuff switch (or retry)
  - CurrentCuff
    01 = Cuff 1 in use,
    10 = Cuff 2 in use
- Status Physiocal,

Physiocal							
PhysiocalState		*Spare status bits*		*Quality*			
7	6	5	4	3	2	1	0

  - Quality
    0000 = 0, Off
    0001 = 1, Uncertain
    0010 = 2, Very inadequate
    0011 = 3, Inadequate
    0100 = 4, Adequate
    0101 = 5, Sufficient
    0110 = 6, Useful
    0111 = 7, Good
    1000 = 8, Great
    1001 = 9, Excellent
  - PhysiocalState
    00 = Turned off,
    01 = Idle,
    10 = Physiocal scan active
    11 = Physiocal adjust active
- BeatsTillPhysiocal, Number of beats till physiocal
- PhysiocalInterval, Number of beats between physiocal
- Automatic Cuff Control,

CuffControl							
Retry number					*Status*		
7	6	5	4	3	2	1	0

  - Status
    000 = 0, Automatic cuff control disabled
    001 = 1, Automatic cuff control enabled, normal operation
    010 = 2, Automatic cuff control is temporarily suspended by other Nano Core functions
    011 = 3, Automatic cuff control is temporarily suspended by other Nano Core functions. A switch will happen as soon as it's resumed

    100 = 4, Busy switching cuffs or enabling the cuff at the start of measurement
    101 = 5, Switching (or enabling) failed due to internal logic error, time to next retry is transmitted in MinutesTillSwitch of Cuff byte,
    110 = 6, Switching (or enabling) failed due to error, time to next retry is transmitted in MinutesTillSwitch of Cuff byte. Error that prevented cuff switching is transmitted in the Error field (it needs not to be cleared), see 2.5.3 for possible error

  - Retry number: counter holding the current retry attempt number
    0x1e = at least 30 retries already performed
    0x1f = final retry attempt
    otherwise = retry attempt number

– Status ModelFlow,

ModelFlow							
Calibration allowed	*Patient*	*Reserved*	*Calibration status*		*ModelFlow status*		
7	6	5	4	3	2	1	0

- ModelFlow status
  000 = 0, Not active, missing input data
  001 = 1, Not active, all required input data provided
  010 = 2, Active, measuring
  011 = 3, Active, performing BraCal
- Calibration status
  00 = 0, Calibration not applied
  01 = 1, Calibration applied, using calibration values obtained through BraCal procedure
  10 = 2, Calibration applied, using stored calibration values
- Reserved
- Patient
  0 = Patient data was not set, or received data was out of range,
  1 = Patient data was set
- Calibration allowed
  0 = Performing BraCal procedure is not possible
  1 = Performing BraCal procedure is allowed

2.5.6 TimeStamp / SampleNumber [2 byte]

Time stamps are specified as counter of samples from the high frequency data.
This way every high frequency sample is consecutively numbered, and missing samples are easy to detect.
Once 65.535 is reached the next sample number will be 0. This will happen about every 327 sec at 200Hz (65 seconds on 1kHz)

2.6 Commands

2.6.1 Nack message

	cmd-ID	*Nack-code*
Nano → PC	Nack-cmd	Nack-code

Nack-cmd:
The command requested OR-ed with 0x80, for example the 'v' command, is nacked with 0xf6.
Nack-code
The reason for not acknowledge. See 2.5.2.

2.6.2 Version Info ('v') [130 bytes]

Structs are returned on request. These structs are direct copies from the firmware information:

	cmd-ID	*cmd-data*	
PC → Nano	'v'=0x76	info ID	
Nano → PC	'v'=0x76	0x00	Hardware info struct [128 bytes]
Nano → PC	'v'=0x76	0x0A	Application version struct [128 bytes]
Nano → PC	'v'=0x76	0x0B	Bootloader version struct [128 bytes]
Nano → PC	'v'=0x76	0x0C	Unified identification strings [varying length]
Nano → PC	'v'=0x76	0x0D	Unique device ID [12 bytes]

The following info can be acquired by info ID:
– 0x00 = Hardware info struct
– 0x0A = Application version struct
– 0x0B = Bootloader version struct

- 0x0C = Unified identification string
- 0x0D = Unique device ID – a 96-bit unique processor identifier, set by the processor manufacturer

The hardware info struct:

Register	Size	Offset	Value
Magic ID	0x04	0x00	0x49 0x6E 0x66 0x6F (= “Info”)
Struct Length	0x02	0x04	0x00 0x80 (= 128)
Struct Version	0x01	0x06	0x01 (= 1)
Struct Type	0x01	0x07	0x48 = ‘H’ = Hardware
Hw Version	0x02	0x08	
Hw Model	0x02	0x0A	
Hw Config	0x04	0x0C	(32 bits, t.b.d.)
Reserved	0x0C	0x10	(reserved for future extension, 14 bytes)
Serial Nr	0x64	0x1C	(free definable data, 100 bytes)

Description of the struct-elements:

- Magic ID:
  This is used to provide a check for a firmware version. If this at the specified location it does not show 0x49 0x6E 0x66 0x6F or “Info” in ASCII, this is not a correctly built, or incompatible firmware image.
- Struct Length:
  In this version of the software, the struct is 128 bytes in total length.
- Struct Version:
  This is version 1 of the struct.
- Struct Type:
  This defines the information contained in the struct, fixed as ‘H’. This way, it has the same layout as the version info struct.
- Hw Version:
  This describes the version of the complete Nano Core hardware from a software compatibility point of view. It is used by the programming-sw to determine the proper firmware version that can be flashed into this Nano Core hardware (hardware-firmware mapping logic). The firmware itself can also check whether it is allowed to run on this hardware.
  The HwVersion must match the ‘hardware’-nr of the firmware.
- Hw Model:
  The HwModel can be used for different product/application-types. The following products types are defined:
  0x00 = Undefined (shall be handled as Nano Core Nova)
  0x01 = Nano Core Nova
  0x02 = Nano Core OEM
- Hw Config:
  Each of the 32 bits indicate that a hardware-option is available. The available hardware-options are t.b.d. Hardware options can refer to both electrical and mechanical options (or versions of options). With these bits, the application-software can properly act upon certain options, or use a specific parameter-set. It exists apart from the HwVersion in order to isolate ‘options’ from ‘compatibility’.
- Reserved:
  These bytes are reserved for future extensions.
- Serial Nr:
  A set of 100 bytes to store the serial number of the Nano Core in a free manner. This will most likely be a (null-terminated) string.

Application and Bootloader version struct:

Register	Size	Offset	Value
Magic ID	0x04	0x00	0x49 0x6E 0x66 0x6F (= “Info”)
Struct Length	0x02	0x04	0x00 0x80 (= 128)
Struct Version	0x01	0x06	0x01 (= 1)
Struct Type	0x01	0x07	0x42 = ‘B’ = Bootloader 0x4E = ‘N’ = Nano Core 0x49 = ‘I’ = Integration
Hardware	0x02	0x08	
Major	0x01	0x0A	
Minor	0x01	0x0B	
Patch	0x02	0x0C	
Revision (SVN)	0x02	0x0E	
Protocol Version	0x01	0x10	**2**
Build Information	0x6F	0x11	111 bytes

Description of the struct-elements:

- MagicID:
  This is used to provide a check for a firmware version. If it does not show 0x6f66 6e49 or “Info” in ascii at the specified location, this is not a correctly built, or incompatible firmware image.
- StructLength:
  In this version of the software, the struct is 128 bytes in total length.
- StructVersion:
  This is version 1 of the struct.
- Major, Minor, Patch, Revision, Hardware:
  Version information is stored in these values. The hardware-nr must match the HwVersion of the board-info struct. In order to keep the bootloader as simple and unaware as possible, it will not check hardware compatibility. But the firmware itself must perform the check at startup. The hardware-nr is essentially part of the full firmware version numbering-scheme:

- Protocol Version:
  Describes which communication protocol is used by the software.
- Build Information:
  0x70 bytes build information which can be filled with whatever value. For example:
  Bootloader_Debug.V0.0.00906.00000.0.bin

Unified identification strings, comma-separated, ASCII-encoded, null-terminated unified identifier, formatted as:
«model_id»,«hardware»,«serial_number»,«APPLICATION_INFO»,«BOOTLOADER_INFO»
where:

- model-id: model id, as defined in hardware info struct→ hw model
- APPLICATION_INFO, BOOTLOADER_INFO: application/bootloader identifiers in format:
  «app-string»_«app-char»_«major».«minor».«patch»_«revision»_«protocol», with:
  - app-string: human readable application type (“Bootloader”, “Nano Core”, “Integration”)
  - app-char: machine readable application type (‘B’, ‘N’, ‘I’)
  - major, minor, patch, revision, hardware: as defined in application/bootloader version struct
  - protocol: communication protocol version

Valid example:
3,2,0123456789ABCDEF,Nano Core_N_1.2.3_4567_2,Bootloader_B_3.2.1_0123_1

2.6.3 Status (‘s’) [15 bytes]

	cmd-ID	*cmd-data*	
PC → Nano	‘s’=0x73	–	
Nano → PC	‘s’=0x73	TimeStamp[2]	Status[13]

- TimeStamp, 2 bytes, see 2.5.6
- Status, 13 bytes, see 2.5.5

2.6.4 Alive (‘a’) [1 byte]

	cmd-ID	*cmd-data*
PC → Nano	‘a’=0x61	–
Nano → PC	‘a’=0x61	--

During measurement alive has to be sent at 1 Hz. This will ensure that communication between the Nano Core and the host is still active. If the Alive command has not been received for some time, the Nano Core will stop measuring and enter idle mode.

2.6.5 StatusUpdate (‘u’) [2/4 bytes]

	cmd-ID	*cmd-data*	
PC → Nano	‘u’=0x75	Update mode	[TimeMs[2]]
Nano → PC	‘u’=0x75	Update mode	[TimeMs[2]]

- Update mode, bit flags
  - Bit 0, 0 = No time based updates.
    1 = Time based update, update every ‘TimeMs’
- TimeMs(Optional), 2 bytes, Update time in ms, only needed if time base update mode flag is set

2.6.6 DataTransmission (‘d’ and ‘D’) [5\9\10 bytes]

	cmd-ID	*cmd-data*				
Nano → PC	‘d’=0x64	TimeStamp[2]	BP[2]	HGT[2]	PLET[2]	PHYSIOCAL[1]
Nano → PC	‘D’=0x44	‘p’=0x62	TimeStamp_hcFAP[2]		hcFAP[2]	
Nano → PC	‘D’=0x44	‘b’=0x62	TimeStamp_reBAP[2]		reBAP[2]	

DataTransmission commands don’t need confirmation. Capitalized cmd-ID (‘D’ instead of ‘d’) indicates that there is a subcommand field (‘b’ or ‘a’) present.

- TimeStamp, 2 bytes, see 2.5.6.
- BP, 2 bytes, Finger blood pressure in 1/10 mmHg, signed 16 bit value
- HGT, 2 bytes, Height correction in 1/10 mmHg, signed 16 bit value
- PLET, 2 bytes, Plethysmograph signal, unsigned 16 bit value
- PHYSIOCAL, 1 byte, physiocal state, see 2.5.5.
- TimeStamp_hcFAP, 2 bytes, see 2.5.6.
- hcFAP, 2 bytes, Height-corrected finger arterial pressure in 1/10 mmHg, signed 16 bit value (only Nano Core OEM)
- TimeStamp_reBAP, 2 bytes, see 2.5.6.
- reBAP, 2 bytes, Brachial arterial pressure in 1/10 mmHg, signed 16 bit value (only Nano Core OEM)

2.6.7 BeatToBeatDataTransmission ('b' and 'B') [10\14\24\28 bytes]

	cmd-ID	*cmd-data*			
Nano → PC	'b'=0x62	TimeStamp 2 Bytes		B2Bnr 1 Byte	Finger pressure data 11 Bytes
Nano → PC	'B'=0x42	'd'=0x64	TimeStamp Derived 2 Bytes	B2Bnr Derived 1 Byte	OEM Derived data 10
Nano → PC	'B'=0x42	'r'=0x72	TimeStamp Reconstructed 2 Bytes	B2Bnr Reconstructed 1 Byte	OEM Reconstructed data 6 bytes

BeatToBeatDataTransmission commands don't need confirmation. Capitalized cmd-ID ('B' instead of 'b') indicates that there is a subcommand field ('d' or 'r') present.

- TimeStamp, TimeStampDerived and TimeStampReconstructed, 2 bytes, see 2.5.6
- B2Bnr, B2BnrDerived and B2bnr Reconstructed 1 byte, B2B sample counters, unsigned 8 bit value

Finger pressure data [11]					
Sys[2]	Dia[2]	Map[2]	HR[2]	IBI[2]	Artefact[1]

- Finger pressure data, 11 bytes
  - Sys, 2 bytes, Systolic blood pressure in 1/10 mmHg, unsigned 16 bit value
  - Map, 2 bytes, Mean arterial in 1/10 mmHg, unsigned 16 bit value
  - Dia, 2 bytes, Diastolic blood pressure in 1/10 mmHg, unsigned 16 bit value
  - HR, 2 bytes, Heart rate in 1/10 beats per minute, unsigned 16 bit value
  - IBI, 2 bytes, Inter beat interval in milliseconds, unsigned 16 bit value
  - Artefact, 1 byte, If and which artefact has occurred during the beat.

Artefact bits, copied from source code:

```
timeOut             1 << 0
physiocalBeat       1 << 1
spiked              1 << 2
imperfect           1 << 3
oscill              1 << 4
damped              1 << 5
sampleMissing       1 << 6
pressureControl     1 << 7
```

Note: This packet can be sent where all fields contain zeroes. This is done when no pulsation is found.

OEM Derived Data [10]
DerivedBasic 10 bytes

- OEM Derived Data, 10 bytes

DerivedBasic [10]				
fiSys[2]	fiDia[2]	fiMap[2]	HR[2]	IBI[2]

- DerivedBasic, 10 bytes, derived pressure variables
  - fiSys, 2 bytes, Systolic finger blood pressure in 1/10 mmHg, unsigned 16 bit value
  - fiMap, 2 bytes, Mean finger arterial pressure in 1/10 mmHg, unsigned 16 bit value
  - fiDia, 2 bytes, Diastolic finger blood pressure in 1/10 mmHg, unsigned 16 bit value
  - HR, 2 bytes, Heart rate in 1/10 beats per minute, unsigned 16 bit value
  - IBI, 2 bytes, Inter beat interval in milliseconds, unsigned 16 bit value

Reconstructed [6]		
reSys[2]	reDia[2]	reMap[2]

– OEM Reconstructed Data, 6 bytes, reconstructed pressure variables
  - reSys, 2 bytes, Systolic brachial pressure in 1/10 mmHg, unsigned 16 bit value
  - reDia, 2 bytes, Diastolic brachial pressure in 1/10 mmHg, unsigned 16 bit value
  - reMap, 2 bytes, Mean arterial pressure of the brachial pressure in 1/10 mmHg, unsigned 16 bit value

2.6.8 Mode (‘m’) [1/2 bytes]

	cmd-ID	*cmd-data*
PC → Nano	‘m’=0x6D	-
Nano → PC	‘m’=0x6D	Mode

- No data is given, the current mode is returned.
- PC can check the returned mode.
- Mode , 1 byte see, 2.5.1, The current mode

2.6.9 Execute command

	cmd-ID	*cmd-data*
PC → Nano	‘e’	Command[1]

- Command
  0x01 = Start measurement (only allowed from idle, also clears all errors)
  0x02 = Stop measurement (only allowed from measure)
  0x03 = Enter service mode (only allowed from idle)
  0x04 = Exit service (only allowed from service)
  0x05 = Enter bootloader (only allowed from idle)
  0x06 = Clear first error, can be used to clear and clear all the errors individually.

2.6.10 PatientData (‘p’) [1/8 bytes]

	cmd-ID	*cmd-data*			
PC → Nano	‘p’=0x70	-			
PC → Nano	‘p’=0x70	Age[2]	Weight[2]	Lenght[2]	Gender
Nano → PC	‘p’=0x70	Age[2]	Weight[2]	Lenght[2]	Gender

If on request no data is specified the Nano responses with the current settings.

- Age, 2 bytes, Age in month, unsigned 16 bit value
- Weight, 2 bytes, Weight in kg, unsigned 16 bit value
- Length, 2 bytes, Length in cm, unsigned 16 bit value
- Gender, 1 byte, 0x01 = Male, 0x02 = Female

2.6.11 CuffUsage (‘c’) [1/2 bytes]

	cmd-ID	*cmd-data*
PC → Nano	‘c’=0x63	
PC → Nano	‘c’=0x63	CuffInterval[1]
Nano → PC	‘c’=0x63	CuffInterval[1]

If on request no data is specified the Nano responses with the current settings.

CuffInterval							
Interval						Command	
7	6	5	4	3	2	1	0

- CuffInterval, 1 byte:
  - Command:
    0x00 = set interval (uses data transmitted in bits 2-7),
    0x01 = use cuff 1,
    0x02 = use cuff 2,
    0x03 = do cuff switch now (only if automatic cuff control is enabled)
  - Interval, automatic cuff switching interval, unsigned 6 bit value:
    0x00 = disable automatic cuff control
    0x3f = restart cuff control scheduler timer
    0x3e = reserved
    0x3d = reserved
    otherwise = cuff switching interval in minutes

Note: In Nano → PC data transmission Command is always either 0x01 or 0x02, other values are only used in PC → Nano messages.

2.6.12 HCU Commands ('z') [1/4 bytes]

	cmd-ID	*cmd-data*
PC → Nano	'z'=0x7A	-
Nano → PC	'z'=0x7A	Result

Tries to zero the HCU if no command data is given, then it returns the result shown below

- Result, Zeroing result,
  0x00 = HCU not connected, Correction is NaN (0xFFFF)
  0x04 = HCU connected, started zeroing procedure

2.6.13 PhysioCal ('h') [1/2 bytes]

	cmd-ID	*cmd-data*
PC → Nano	'h'=0x68	[On/Off]
Nano → PC	'h'=0x68	On/Off

- If on request no data is specified the Nano responses with the current settings.
- On/Off, 1 byte, 0x00 = PhysioCall off, 0x01 = PhysioCall on, (0xFF in response indicates that setting PhysioCal is not allowed)

2.6.14 ModelFlowCalibration ('f')

	cmd-ID	*cmd-data*		
PC → Nano	'f'=0x66	'r'=0x72 (Results)		
PC → Nano	'f'=0x66	's'=0x73 (Start) / 'a'=0x61 (Abort)		
PC → Nano	'f'=0x66	'c'=0x63	calSys[2]	calDia[2]
Nano → PC	'f'=0x66	'c'=0x63	calSys[2]	calDia[2]
Nano → PC	'f'=0x66	'r'=0x72	calStatus[1]	dSYSreBAP[2]

- Results, 1 byte, 'r'=0x72, request calibration results for the current cuff
- Start/Abort, 1 byte, 's'=0x73 = BraCal procedure started, 'f'=0x66 = BraCal procedure aborted.
- calSys and calDia, 2 bytes each, systolic and diastolic brachial pressure for calibration in 1/10 mmHg, signed 16 bit integer. Sending this message finishes the BraCal procedure. If data is accepted by Nano Core it responds with the same values, otherwise an error is communicated.
- BraCal procedure results for the current cuff:
  - calStatus, 1 byte, status of the BraCal calibration:
    - 0x00 = calibration has not been performed yet,
    - 0x01 = calibration performed, using calibration values obtained during the last BraCal run,

- 0x02 = calibration performed, using saved calibration values,
- 0x03 = BraCal procedure is running

- dSysreBAP, 2 bytes, difference between the systolic pressure values before and after calibration in 1/10 mmHg, signed 16 bit value. This value is meaningless if calStat indicates no calibration has been perofmed or BraCal is running

2.7 Service commands

These commands are only accepted if the system is in service mode.

2.7.1 ServiceTest ('t') [1/X bytes]

	cmd-ID	*Sub-cmd*	*cmd-data*	
PC → Nano	't'=0x74	0x00	-	
Nano → PC	't'=0x74	0x00	Status	
	cmd-ID	*Sub-cmd*	*cmd-data*	
PC → Nano	't'=0x74	0x01	TestId	Settings
Nano → PC	't'=0x74	0x01	TestId	Status
	cmd-ID	*Sub-cmd*	*cmd-data*	
PC → Nano	't'=0x74	0x02	-	
Nano → PC	't'=0x74	0x02	-	
	cmd-ID	*Sub-cmd*	*cmd-data*	
PC → Nano	't'=0x74	0x03	TestId	
Nano → PC	't'=0x74	0x03	Test Id	Results

- Subcmd, 1 byte, manipulate state of the system
  0x00 = get status
  0x01 = start test / update test
  0x02 = stop test
  0x03 = get results of test
- TestId, 1 byte, the test to perform. Described in the following sections.
- Settings, 12 bytes test specific parameters, divided in 3x uint32, described in the following sections
- Status, 1 byte, Status of the test
  0x00 = Test idle
  0x01 = Test running
  0xFF = Test not allowed
- Results 12 bytes test specific results divided in 3x int32, described in the following sections.

Test commands influence the statemachine of the test system directly. Tests can only be started if the status is idle and the system is in service mode.

2.7.2 SteadyPressureTest (Testid = 0x01)

Steady pressure test is used to apply a steady pressure to the specified output. Cuffs can be selected by setting the cuff setting, and the pressure is selected by writing the pressure. This test returns no results, however, errors can be thrown by the pressure controller and then the test will be stopped.

Settings1	*Settings2*	*Settings3*
Cuff	Pressure	0x0000 0000

- Cuff (Uint32)
  If bit 0 is 1, cuff 1 valve is actuated
  If bit 1 is 1, cuff 2 valve is actuated
- Pressure (Uint32)
  Pressure is specified in mmHg with a maximum of 350 mmHg.

Results of the test

Results1	*Results2*	*Results3*
0x0000 0000	0x0000 0000	0x0000 0000

2.7.3 SquareWaveTest (Testid = 0x02)

Pressure is generated in a squarewave pattern, the amplitude is fixed to 100mmHg, the mean pressure and frequency can be selected by the user. Results are not returned in this test. However, the pressure controller can throw errors, which will stop the test and set the Nano Core in Error mode.

Settings1	*Settings2*	*Settings3*
Cuff	Mean Pressure	Period

- Cuff (Uint32)
  If bit 0 is 1, cuff 1 valve is actuated
  If bit 1 is 1, cuff 2 valve is actuated
- Pressure (Uint32)
  Pressure is specified in mmHg with a maximum of 350 mmHg.
- Period (Uint32)
  Period is specified in ms, with a minimum of 10 ms and a maximum of 1000 ms.

Results of the test

Results1	*Results2*	*Results3*
0x0000 0000	0x0000 0000	0x0000 0000

2.7.4 ExhaustValveTest (Testid = 0x03)

Test to determine if the exhaust valve is functional.
The settings are described below, settings are all 0

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
Time [ms]	0x0000 0000	0x0000 0000

- Time (ms)
  The time it takes to deflate the volume.

2.7.5 CuffValveTest (Testid = 0x04)

Test to determine if the cuff valves are functional.
The settings are described below, settings are all 0

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1		*Results2*	*Results3*
Cuff1 deflate [ms] (16 upper bits)	Cuff2 deflate [ms] (16 lower bits)	Deflate time [ms] from 40kPa to 20kPa	Deflate time [ms] from 13.3kPa to 6.6kPa

- All time in ms

2.7.6 HBridgeLinearityTest (Testid = 0x05)

Test to find the performance of the H-Bridge, including linearity and coil impedance.
The settings are described below, settings are all 0

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
Impedance [mOhm]	Remaining Current [mA]	0x0000 0000

2.7.7 VolumeRatioTest (Testid = 0x06)

Test to determine the ratio between the both volumes and the current pressure ratio
The settings are described below, settings are all 0

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
Pressure / Current [Pa/A]	1000 * BellowPressure / VolumePressure [-]	0x0000 0000

2.7.8 LeakageTest (Testid = 0x07)

Determines the leakage of the system
The settings are described below, settings are all 0

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
1000 * Ratio [-]	0x0000 0000	0x0000 0000

2.7.9 PumpPerformanceTest (Testid = 0x08)

Measures important characteristics of the pump
The settings are described below, settings are all 0

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
Flow in first pulse [ml/min]	Flow at 40 kPa [ml/min]	Restart pressure [Pa]

2.7.10 PressurePulseTest (Test = 0x09)

Generates a pressure pulse, can be used to check if pressure generation is sufficient.
The settings are described below, settings are all 0

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
First Pressure Pulse [Pa]	Maximum Pulse [Pa]	0x0000 0000

2.7.11 Production Pressure Sensor test (Test = 0x0A)

Opens a cuff valve, and determines the mean and std of both sensors.
Settings are all 0.

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
Bellows Sensor Mean [Pa * 1000]	Volume Sensor Mean [Pa * 1000]	Bellows std [Pa << 16] + volume std [Pa]

2.7.12 Production led controller test (Test = 0x0B)

Sets the ledcontroller voltage to a certain value, and measures the current through the led. The mean and std are returned.
Settings are all 0.

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
Led Current mean [A * 1000000]	Led current std [A * 1000000]	0x0000 0000

2.7.13 Production plethysmograph test (Test = 0x0C)

Checks if the frontend gain switch is functional.
Settings are all 0.

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
Plet mean with gain off [nA]	Plet mean with gain on [nA]	Plet std with gain on [nA * 1000]

2.7.14 Production HCU amplifier test (Test = 0x0D)

Output mean and std of the raw HCU signal, of which half the range is subtracted.
Settings are all 0.

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
HCU mean value [mmHg * 1000 + 1000000]	HCU std value [mmHg * 1000]	0x0000 0000

2.7.15 Production temperature Sensor test (Test = 0x0E)

Gives mean and std of the temperature sensor.
Settings are all 0.

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
Temperature mean [C * 1000]	Temperature std [C * 1000]	0x0000 0000

2.7.16 Production accelerometer test (Test = 0x0F)

Gives mean and std of the accelerometer sensor on the X axis which runs in the same direction as the plunger actuator.
Settings are all 0.

Settings1	*Settings2*	*Settings3*
0x0000 0000	0x0000 0000	0x0000 0000

Results of the test

Results1	*Results2*	*Results3*
Accelerometer x mean [m/s^2 * 1000 + 1000000]	Accelerometer x std [m/s^2 * 1000]	0x0000 0000