Reference Guide

PulseOn OHR Module P-OHR1F

Version 0.95

15 Apr 2016

Table of Contents

Changelog

Version	Change
0.7	Initial release
0.8	Expanded hardware sections, updated to latest protocol and API
0.82	Titles updated, Raspberry Pi Pin mapping changed, protocols expanded
0.85	Changes to protocol and message length
0.9	Minor modifications to the message protocol
0.91	Additions and modifications to the message protocol
0.92	Small change regarding API
0.93	Minor changes to API
0.94	Minor changes to API
0.95	Small style edits and references to I²C removed

Main Changes In Version 0.93

Gender values changed in User Settings message to male = 0 and female = 1, Enable Sleep Analysis added to Inbound Message IDs.

Changes In Version 0.94

Artifact warning message content extended to include connect/disconnect charger (See Inbound Message IDs).

1 Basic Operation

1.1 Introduction

The PulseOn Optical Heart Rate Module P-OHR1F is a low-power, fully integrated sensor module that includes heart rate detection algorithms. It is a multi-wavelength (green & IR) solution with optimally matched asymmetric geometry providing large dynamic measurement range and low power consumption.

The PulseOn module operates on the principle of simple usage modes. These modes are optimized to provide accurate results with minimum power usage. The host system can select between the predefined modes for different use cases. The modes are:

- On demand - Continuous activity tracking, HR on demand.
- Sports - Continuous HR- and activity tracking during sports.
- Sampled - Sampled HR and continuous activity tracking during daily life.
- Sleep - HR tracking during sleep for the purpose of stress- and recovery analysis (Currently not yet available).
- Firmware update - Special mode used for firmware update.
- Idle - Idle mode (no tracking, minimum power consumption).

The module communicates with the host system by using SPI interface. All the communication is frame based. Each frame contains varying number of messages. The host can control the module with the following messages:

- Getters and setters - Include mode selection, mode parameters, speed information, artifact warnings, demanding heart rate.
- Settings message - Personal settings.

The module can send messages of the following types:

- User settings message - Module communicates back user settings that it has received from the host.
- Activity messages, instantaneous, accumulated, and swim – Accelerometer based tracking results.
- Heart rate message - HR based tracking results.
- Interbeat Intervals message - Interbeat intervals for analysis of Heart Rate Variability.
- Device status message - Contains information on the module operation. Operating mode, error code, operational.
- Command status message - Informs the host the status of particular command.

2 Applications and Usage modes

- Optical Heart rate monitoring
- Heart beat inter-beat-intervals for the purpose of calculating heart rate variability (necessary for stress- and recovery analysis)
- Continuous activity tracking (activity classification, steps count, and inertia based speed, distance, and energy expenditure)
- (Optional by integrated Firstbeat analytics; Energy expenditure (kCal), Training intensity (%VO2Max), Training effect (EPOC), and Fitness level (VO2Max)

2.1 Overview

The usage mode is set by the host, and it determines which sensor signals are acquired and which values are calculated from them. It also allows the host to flash a new version of the module firmware. The message format is described in Inbound Message IDs. After power-on, the module starts in Idle mode.

3 Usage modes

Mode name	Description	Available data	Sensor settings	Power consumption
Idle	No activity	N/A	Sensors OFF	N/A
On demand	Continuous activity tracking, HR on demand (intermittent value)	Steps (running, walking) Activity class (running, walking, other, rest, sleep) Speed and distance (inertia based) Heart rate (intermittent) Wrist detection	OHRM off, switched on for ~1m when demanded to read intermittent HR, 3D accelerometer on	<1-15mW
Sports	Continuous HR- and activity tracking during exercise	Steps Activity class Heart rate Speed and distance (inertia based) Wrist detection Energy consumption (kcal)* Oxygen consumption (%VO_2max)* Training effect* VO_2max (if speed available)*	Continuous OHRM, automatic optical channel selection, 3D accelerometer on	15mW
Sampled	Periodic HR measurements and continuous activity tracking during daily life	Steps Activity class Heart rate Speed and distance (inertia based) Wrist detection Energy consumption (kcal)[1] Oxygen consumption (%VO_2max)[2]	OHRM sampled with determined sampling ratio, automatic optical channel selection, 3D accelerometer on	3-15mW

Mode name	Description	Available data	Sensor settings	Power consumption
Sleep*	Continuous HR- and activity tracking during sleep	Steps Activity class Heart rate Wrist detection Energy consumption (kcal)* Oxygen consumption (%VO_2max)[3]	Continuous OHRM, automatic optical channel selection, 3D accelerometer on	5-9mW
Firmware update	Special mode used for firmware update	Update status	Sensors OFF	N/A

* Not yet available

3.1 Mode descriptions

Only a single usage mode at a time is active and the module output depends on the usage mode. The module does not keep the absolute time and it supplies instead free-running timestamps. Accumulated values, such as energy expenditure and distance, can be reset by the host sending the message RESET_ACCU_DATA (see Section Inbound Message IDs).

3.1.1 On demand mode

In on demand mode device is tracking only 3D accelerometer signal and OHRM is turned off by default. When requested by the host, OHRM is turned on for as long as necessary in order to measure one reliable HR value or as long as requested for multible values.

This mode is targeted to be enabled 24/7. Key targets:

- Continuous activity tracking
  - Low-power accelerometer tracking and processing
  - Activity data send every 1s by default (the host can set the interval or switch off activity messages if necessary)
- HR tracking on demand
  - HR is demanded by host and the result is sent as soon as a reliable HR is found
  - HR max, min, and mean value stored with timestamp
- Wrist detection

3.1.2 Sports mode

In sports mode[4] device is tracking HR during sports for maximum accuracy and reliability. Firstbeat ETE library[5] is used to calculate energy consumption, oxygen consumption, and

training effect. Key targets:

- Maximum HR reliability during sports
- Continuous HR tracking
    - HR data is sent every 3 seconds by default (the host can set the interval or switch off HR messages if necessary)
- Continuous Activity tracking
    - Activity data is sent every 1 second by default (the host can set the interval or switch off activity messages if necessary)
- Dynamic optical channel selection
- Wrist detection

3.1.3 Sampled mode

In sampled mode, HR is measured periodically (once every 60 seconds by default). Accelerometer related parameters are tracked continuously. Key targets:

- HR is provided to the host every 60 seconds by default (the interval between consecutive measurement can be set by the host)
- Continuous activity tracking as in on demand mode and sports mode.
- Sampled mode

3.1.4 Sleep mode

In sleep mode, HR- and activity are tracked continuously . Key targets:

- Continuous HR tracking during sleep for the purpose of calculating HR Variability
- Dynamic optical channel selection
- Wrist detection
- Currently not yet available

3.1.5 Firmware update mode

Special mode used for firmware update. Firmware update is initialized when mode is entered. Mode is exited automatically when the firmware update is finished. Firmware update is canceled if the update mode is exited before the update process has finished. Please see Firmware Update section for details.

3.1.6 Idle mode

In idle mode the module doesn't actively measure anything. This is more or less transitional state and host is expected to use this mode only temporarily during transitions. For example after startup the device is in idle mode and host is expected to set mode to what is applicable at that time.

3.2 Device startup

This section briefly describes how the module is started up.

3.2.1 Quick guide for booting the device

The device can be started by powering both VDIG and Vbat and letting the device from reset state by raising NRESET pin. The interrupt line (INF_INT) should be pulled up, but this is not critical as the device does not finish start-up until the reset line is pulled up.

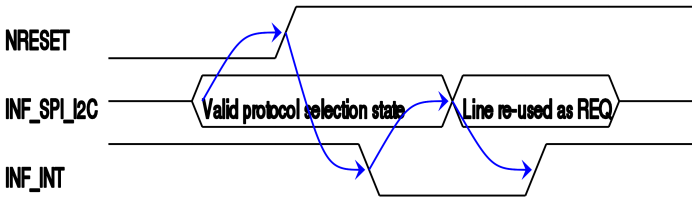
After the device is finished powering up, it pushes to the low (INF_INT) line to indicate the device has data to read (in this case, the message indicating that the device was successfully powered on). After the device has initialized it is in idle mode.

In the case the device has pending SW update ongoing, it takes longer to get into operational state, as it has to finish the pending SW (firmware) installation.

3.2.2 Protocol selection

Note-Even though I²C is not supported the protocol selection needs to be impleneted as described here.

When the device is released from reset, host should maintain INF_I2C/_SPI line in right position for protocol selection, (low for SPI) until the first interrupt after which the line can be reused. The time for the interrupt line should be relatively short during normal operation, but might be long - for example during the SW update. Therefore, no timing-based assumptions should be made when to no longer maintain valid protocol selection state on the line.



Reset, selection and interrupt lines during start-up.

The INF_I2C/_SPI line is reused as the REQ line after start-up. In the case SPI is used as a communications bus, the line should be pulled up initially after the first interrupt (there is no hard real-time requirements for this - pulling the line up can be delayed). Please refer to SPI timing section for more information on REQ line usage.

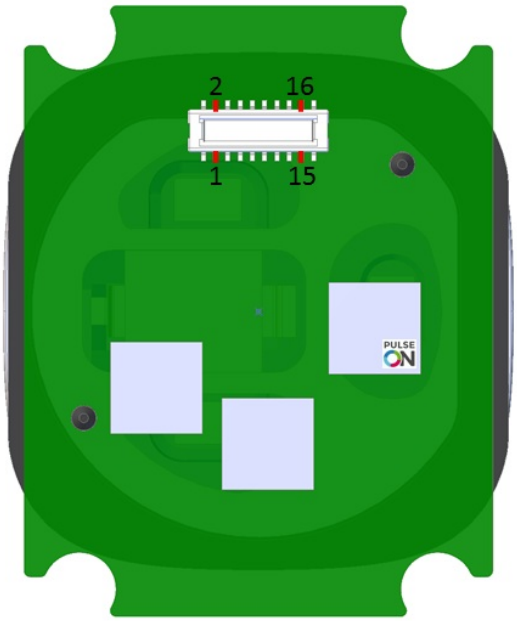
Please note that at the moment, I²C is not supported and SPI is always used, and the line is ignored during start-up. However, for compatibility with future versions, it is recommeded to keep the line in the proper state.

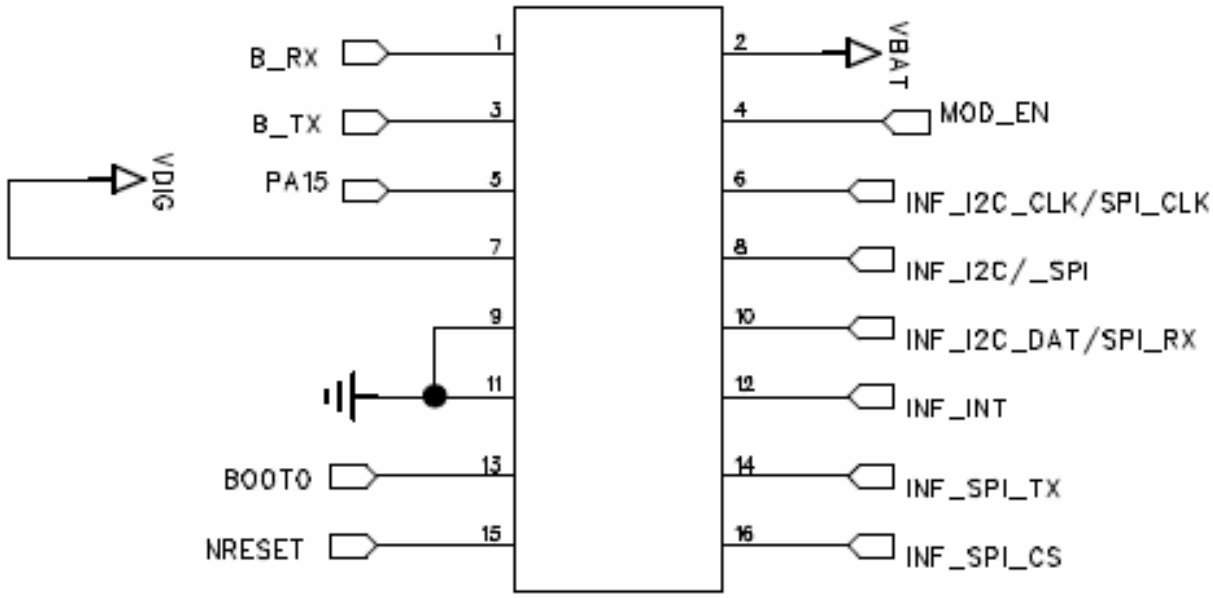
3.3 Firmware Update

Firmware updates for PulseOn OHRM work by putting the device into firmware update mode, transmitting the firmware image in small segments that are stored in internal flash and finally sending a CRC32 checksum for the whole image to begin the actual update, which overwrites the previous firmware when the checksum matches. This is done internally during a restart, after which the device should start up normally. Please note that the booting takes significantly longer when updating the software.

The protocol messages used to perform a software update are described in section SW Update messages.

4 Pin Configuration and Functions





Please notice-the compatible host connector type made by SMK CPB0316-0150F

PIN#	Pin Name	Description
1	B_RX	DNC
2	Vbat	Battery voltage in (2,7V - 5,5V)
3	B_TX	DNC
4	MOD_EN	DNC
5	PA15	RES
6	INF_I2C_CLK/SPI_CLK	Clock signal from host for selected interface (I2C or SPI)
7	VDIG	1,8V input from host (1,65V - 2,0). Vdd.
8	INF_I2C/_SPI	Interface selection input - (1=I2C / 0=SPI). Reused as SPI request signal after startup.
9	GND	Ground
10	INF_I2C_DAT/SPI_RX	I2C Data or SPI_RX (MOSI) for module
11	GND	Ground
12	INF_INT	Interrupt signal from OHRM module for host.
13	BOOT0	DNC
14	INF_SPI_TX	SPI MISO from module
15	NRESET	Reset signal for module
16	INF_SPI_CS	SPI bus Chip select for the module.

INF signals are the main interface signals for the host. They are having the following parameters:

Vol max: 0,45V (Iio = 4mA).
Voh min: Vdd-0,45V (Iio = 4mA).
Vil max: 0,3*Vdd
Vih min: 0,39*Vdd+0,59V.
Vih max: 5.5VInputs

NRESET: Vih max = Vdig + 0,3V having internal pull-up to Vdig. Preferred reset driving method: Activate reset by pulling down reset line with HOST's open-drain output.

5 Module Protocol

This section describes how the communication works between host and the module with different protocols. The protocol is selected during module start-up, for details please see Device startup section.

5.1 SPI

The SPI interface has two lines besides the standard SPI lines: **IRQ** (referred as **INF_INT** in pin mapping section) line and **REQ** (**INF_I2C/_SPI** line reused after module startup). **IRQ** line is used for the module to indicate the host that there is data available to read or that the module is ready to start SPI transaction when host initiates the transaction by pulling **REQ** line down to indicate it wants to send/receive data. The purpose of using the **REQ** line with an interrupt reply is to act as a flow control, which allows the module to optimize power usage in ways that might compromise fast response to changes on the slave/chip select line. See the Host Initiated SPI transfer section for more details.

Otherwise, the SPI lines are mapped to the interface in the following way:

- **INF_SPI_CS**: SPI slave/chip select
- **INF_I2C_CLK/SPI_CLK**: SPI clock
- **INF_I2C_DAT/SPI_RX**: SPI MOSI
- **INF_SPI_TX**: SPI MISO

As in standard SPI communication, the host should keep the slave/chip select line high when it's not transmitting and keep it low during transmission. The other 3 lines behave in the standard manner as well: host clocks the clock line and uses MOSI to transmit to module and MISO for reading data from the module.

The length of transmission should be 32 bytes or the behavior is undefined.

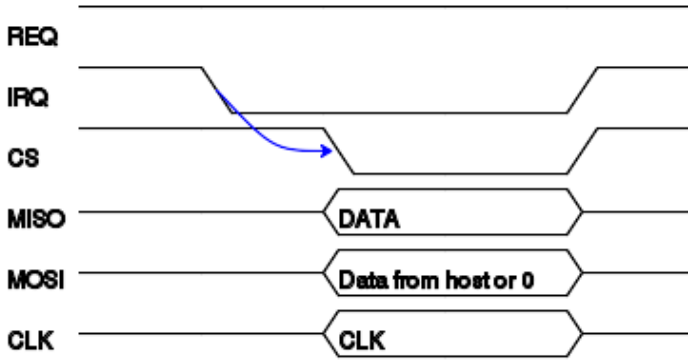
Note: There is no control for not reading data over SPI, which means that the host should check the received transmission for valid data whenever it has sent something to the module.

As with SPI both sides are transmitting, the host needs to make sure that the data transmitted by the host is zero in the case the host doesn't want to send commands to the module.

5.1.1 Module initiated SPI transfer

When the module indicates data availability by pulling the **IRQ** line down, a standard SPI transaction can be used immediately to retrieve the data from the module. The only constraint is the length of the transmission that should equal the constant length defined in section above.

The image below illustrates timings related to the SPI transaction in this case:



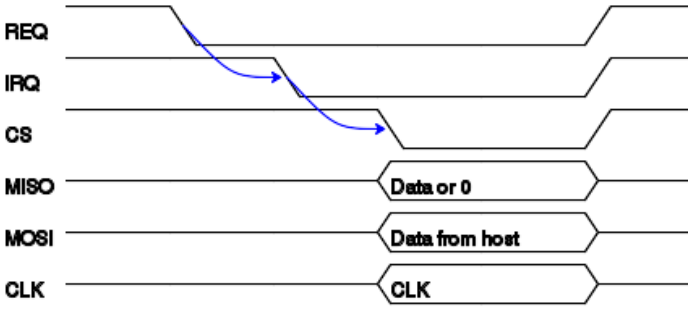
Module initiated SPI Transfer

5.1.2 Host Initiated SPI transfer

The REQ line exists to allow the host to initiate SPI transactions. When the line is pulled down, the module initializes the data to be transmitted to the host and pulls the IRQ line down. From this point onwards, the transaction behaves like in the first scenario. After the host has asserted the CS line, it should release the REQ line. If the host would like to transfer another frame, it should assert the REQ Line again after the CS is released.

Note: in this case, the module might transmit only zeros (0x00), indicating that there is no new data available from the module.

The image below illustrates timings for a SPI transaction in this case:



Host initiated SPI Transfer

6 Protocol Data

This section describes in detail the communication protocol between the host and the module. The communication protocol is based on zero-padded 32-byte fixed-length frames. Each frame can contain zero or a varying number of messages.

6.0.1 Message Format

Each message starts with a message ID followed by n bytes of message content. Multiple messages can be concatenated into one frame so that the next message ID follows directly after the content of the first message.

Name	Byte	Description
ID	0	Message ID
Message content	1..n	Message content
ID or 0x00	n+1	Next message ID or 0x00 indicating the end of messages in the frame

When the first received byte is 0x00, the message is interpreted as "do nothing" and the rest of the message is ignored. This is particularly useful when using the SPI interface, which always requires writing some data to the module in which case zero data should be written when the user doesn't want to send commands to the module while reading the data.

6.1 Inbound Communication

Whenever the host transmits a full sequence to the module, it interprets it as a message. The first byte indicates the message ID and the rest are the message-specific payload.

6.1.1 Inbound Message IDs

These messages are getters and setters sent from the host to the module.

ID	Parameter	Type	Default	Description
0x00	Do nothing			Do nothing with following data.
0x40	Set Usage Mode	uint8_t	0	Set Operating mode (, Idle = 0, OnDemand = 1, Sports = 2, Sampled = 3, Sleep = 4, Firmware update = 5). See the section on Usage Modes for details.
0x41	Set Sampled Mode Interval	uint16_t	60	Set interval in seconds between periodic heart rate measurements in Sampled Mode
0x42	Set HR Interval	uint16_t	3000	Set heart rate message reporting interval in milliseconds. A value of 65535 disables heart rate messages

ID	Parameter	Type	Default	Description
0x43	Set Activity Interval	uint16_t	1000	Set activity message reporting interval in milliseconds. A value of 65535 disables activity messages
0x44	Set User settings	struct		Set user settings defined in the User Settings Message
0x45	Set Speed	uint16_t	0	Current speed in 10 * km / hour (optional). [6]
0x46	Set Altitude	uint16_t	0	Current altitude (meters). 0-10000m is a valid range. Datapoints having altitude out of range are ignored (optional).[7]
0x47	Artifact warning	uint16_t	0	Warns the module that the following optical data can contain artifacts. Available warnings are connect charger = 1, disconnect charger = 2.
0x48	Request HR	N/A		Requests heart rate measurement from the module in the on demand mode.
0x49	Reset accumulated data	uint8_t		Reset HR_DATA = 1, reset ACCU_ACTIVITY_DATA and SWIM_ACTIVITY_DATA = 2, reset both HR and ACTIVITY data = 3
0x4A	Set Activity Class	uint8_t		If set to Other, Walk, Run, or Bike, it overrides ActivityClass and also sets WorkoutClass to 1 in the Instantaneous Activity Message. If set to Rest (default) ActivityClass and WorkoutClass are determined by the module
0x4B	Enable Swim Analysis	struct		Enables and disables swim analysis, and sets pool size (see Enable Swim Analysis)
0x4C	Enable Sleep Analysis	uint8_t	1	Enables and disables sleep analysis (see SleepClass in Instantaneous Activity Message)
0x4E	Request HR measurement	uint16_t		Will request HR measurement over period specified in seconds in the parameter. If parameter is 0, the module will cancel current periodical HR measurement.
0xC0	Get Usage mode	N/A		Module responds with 0x10 Usage mode message
0xC1	Get Sampled Mode Interval	N/A		Module responds with 0x11 Sampled Mode Interval message
0xC2	Get HR Interval	N/A		Module responds with 0x12 HR Interval message
0xC3	Get Activity Interval	N/A		Module responds with 0x13 Activity Interval
0xC4	Get User settings	N/A		Module responds with 0x14 User Settings message

ID	Parameter	Type	Default	Description
0xCA	Get Version information	N/A		Module responds with 0x0A Version information message
0x81	Update payload	N/A		Send firmware update payload.
0x82	Update CRC	N/A		Send firmware update CRC.

6.1.2 Enable Swim Analysis Message

By default Swim Analysis is disabled. It can be enabled and disabled by the host, and the host can also set the swimming pool size if known.

Name	Byte	Bits	Description
ID	0	7..0	Inbound message ID=0x4B
Enable Swim Analysis	1	7..0	A value > 0 enables swim analysis whereas a value of 0 disables it
Swim Pool Size	2	7..0	AUTO_DETECTION = 0, 25_METER = 1, 50_METER = 2

6.1.3 User Settings Message

This message is intended for both inbound and outbound communication. The ID determines whether the message is sent by the host to be written to the module (**ID=0x44**) or by the module to be replied to the host (**ID=0x14**).

Name	Byte	Bits	Description
ID	0	7..0	Inbound message ID=0x44, Outbound message ID=0x14 as reply to read user settings ID=0xC4
Timestamp	1 2 3 4	31..24 23..16 15..8 7..0	Relative timestamp in ms (must be 0x0000 in inbound frame, in read user settings this is free running timestamp)
Activity Level	5	7..0	User's average level of physical exercise. Possible values are 0,10,20,30,40,50,60,70,75,80,85,90,95 and 100. (optional)[8]
Age	6	7..0	Age in years.
Height	7 8	15..8 7..0	Height in millimeters.
Weight	9	7..0	Weight in kilograms.
Gender	10	7..0	0 for men and 1 for women.
Min HR	11	7..0	Minimal heart rate as beats per minute. [30,80] (optional, 0 if not known)[9]

Name	Byte	Bits	Description
Max HR	12	7..0	Maximal heart rate as beats per minute. [100,240] (optional, 0 if not known)[10]
VO2max	13 14	15..8 7..0	10 * VO_2max in mL/(kg·min)[11]
FirstIteration	15	7..0	Reserved for future use

6.1.4 HR measurement requests

There are two different HR measurement request messages: REQUEST_HR (0x48) and REQUEST_HR_MEASUREMENT (0x4E). Both are intended to be used to measure HR when the module is in either on demand or in sampled mode. The difference between requests is that first starts measurement until HR is obtained (in similar ways as sampled mode) while the second requests measurement for a period defined in seconds in the 16 bit parameter of the message. Requests to measure the HR for period of time can be cancelled by sending REQUEST_HR_MEASUREMENT message with 0 as parameter.

6.1.5 SW Update messages

The SW update messages work only in SW update mode. Use set usage mode (0x40) message to enter update mode and to abort transmission of SW update image by leaving update mode.

6.1.5.1 Update payload

The transmission of SW image is split into 16 byte chunks. Update payload message contains single chunk along its CRC32. Once a chunk is received the CRC32 will be checked, if it matches the payload will be written to flash and module will send command status message (0x15) with successful status, otherwise the module will reply with non-successful (non-0) status.

When the module enters SW update mode, the update is initialized and next payload will be written in the beginning of SW image section. After each successful payload receiving next payload will be written into next section.

Name	Byte	Description
ID	0	0x81
Payload	1	Payload byte 0
Payload	2	Payload byte 1
Payload	3	Payload byte 3
...	...	...
Payload	15	Payload byte 14
Payload	16	Payload byte 15

Name	Byte	Description
CRC32	17	CRC32 MSB
CRC32	18	CRC32
CRC32	19	CRC32
CRC32	20	CRC32 LSB

The table below shows most common status replies to this command.

Status	Fancy name	Description
0x00	NO_ERROR	Chunk successfully accepted. Next write will go to next chunk.
0x76	ERROR_STATE	Command failed because the device is not in update mode
0x78	ERROR_CRC	Command fialed because CRC supplied didn't match CRC calculated
0x77	ERROR_PAYLOAD	Command failed because of internal module failure

6.1.5.2 Update CRC

After sending all of the update payloads, update can be started by sending CRC32 calculated for whole image. This command should generate always command status message reply. When the CRC32 has been received successfully the module will restart itself which means it should be expected to restart very soon after sending OK status for SW update CRC/begin command.

Because of this it is recommended to leave REQ line in a desired state for protocol selection at least until failed status has been received (or a device status reporting the device to be in the idle mode to indicate it has started up successfully) to ensure that the INF_I2C/_SPI line is in the right state during the device startup.

The description of the message below.

Name	Byte	Description
ID	0	0x82
CRC32	1	CRC32 MSB
CRC32	2	CRC32
CRC32	3	CRC32
CRC32	4	CRC32 LSB

The table below shows most common status replies to this command.

Status	Fancy name	Description
0x00	NO_ERROR	CRC accepted. The device should be rebooted immediately after this.
0x73	ERROR_GENERIC	Command failed, specific status message not specified

6.2 Outbound Communication

These messages are sent from module to the host.

6.2.1 Outbound Message IDs

ID	Message	Type	Description
0x00	Do nothing	None	Empty message
0x01	Instantaneous Activity	struct	Instantaneous results based on accelerometer data
0x0B	Accumulated Activity	struct	Accumulated results based on accelerometer data
0x02	HR	struct	Results based on optical heart rate
0x03	Interbeat intervals	struct	Results for analysis of Heart Rate Variability
0x09	Device Status	struct	Device status
0x0A	Version information	struct	Version information
0x11	Sampled Mode Interval	uint16_t	Interval in seconds between periodic heart rate measurements
0x12	HR Interval	uint16_t	Heart rate message reporting interval in milliseconds
0x13	Activity Interval	uint16_t	Activity message reporting interval in milliseconds
0x14	User Settings	struct	User settings
0x15	Command Status	struct	Command status

6.2.2 Instantaneous Activity Message

Instantaneous results based on accelerometer data. The results reflects what the user is doing at the current moment.

Name	Byte	Bits	Description
ID	0	7..0	Message ID=0x01
Timestamp	1 2 3 4	31..24 23..16 15..8 7..0	Relative timestamp in ms
WornState	5	7..0	0 = Not Worn, 1 = Worn, 2 = Badly Placed, 3 = Not Worn And Not Usable
ActivityClass	6	7..0	Rest = 0, Other = 1, Walk = 2, Run = 3, Bike = 4, Other Rhythmic = 5, Swimming = 6
SleepClass	7	7..0	Wake = 0, Light Sleep = 1, Deep Sleep = 2
WorkoutClass	8	7..0	No Workout = 0, Workout = 1

Name	Byte	Bits	Description
Speed	9 10	15..8 7..0	Forward speed during walking and running in 10 * km/h

6.2.3 Accumulated Activity Message

Accumulated results based on accelerometer data. The results summarise what the user has been doing since the last reset.

Name	Byte	Bits	Description
ID	0	7..0	Message ID=0x0B
Timestamp	1 2 3 4	31..24 23..16 15..8 7..0	Relative timestamp in ms
WalkSteps	5 6 7	23..16 15..8 7..0	Number of walking steps since the last reset of the counter
RunSteps	8 9 10	23..16 15..8 7..0	Number of running steps since the last reset of the counter
BikeSteps	11 12 13	23..16 15..8 7..0	Number of bike steps since the last reset of the counter
OtherSteps	14 15 16	23..16 15..8 7..0	Number of other steps since the last reset of the counter
WalkDistance	17 18 19 20	31..24 23..16 15..8 7..0	Forward distance during walking in meters since the last reset of the counter
RunDistance	21 22 23 24	31..24 23..16 15..8 7..0	Forward distance during running in meters since the last reset of the counter
KCalInertia	25 26 27 28	31..24 23..16 15..8 7..0	Energy expenditure in kcal since last reset of the counter

6.2.4 Swim Activity Message

Swimming statistics based on accelerometer data. Note that Swim Analysis is disabled by default. It can be enabled by sending the module an ENABLE_SWIM_ANALYSIS message.

Name	Byte	Bits	Description
ID	0	7..0	Message ID=0x0C
Timestamp	1 2 3 4	31..24 23..16 15..8 7..0	Relative timestamp in ms
SwimLapCount	5 6	15..8 7..0	Number of laps since the last reset of the counter
SwimDistance	7 8	15..8 7..0	Distance in meters since the last reset of the counter
SwimTime	9 10 11 12	31..24 23..16 15..8 7..0	Time in seconds since the last reset of the counter
SwimPace	13 14	15..8 7..0	Pace in min/km (x10) of the last lap
SwimSpeed	15 16	15..8 7..0	Swimming speed in meters/hour of the last lap

6.2.5 Heart Rate Message

Results based on optical heart rate

Name	Byte	Bits	Description
ID	0	7..0	Message ID=0x02
Timestamp	1 2 3 4	31..24 23..16 15..8 7..0	Free running relative timestamp in ms
HR	5	7..0	Heart rate in beats per minute
HRQI	6	7..0	Hear rate quality index [0,100]
HROperatingStatus	7	7..0	0=Not_Valid, 1=Valid
MinHR	8	7..0	Minimal HR in beats per minute since last reset of the counter
MaxHR	9	7..0	Maximal HR in beats per minute since last reset of the counter
Average HR	10	7..0	Average HR in beats per minute since last reset of the counter
Training Effect	11	7..0	10 * Training Effect of the exercise [10,50] (1.0,5.0) [12]

Name	Byte	Bits	Description
RelativeOC	12	7..0	Instantaneous oxygen consumption as a percentage of personal VO_2max[13]
KCal	13 14	15..8 7..0	Cumulative Energy Expenditure in kcal[14]
VO2max	15 16	15..8 7..0	10 * estimated VO_2max in mL/(kg·min). >0 if new value is found[15]

6.2.6 Interbeat Intervals Message

Results based on optical measurement. Available in Sampled Mode and Sleep Mode, not available in Sports Mode and On Demand Mode.

Name	Byte	Bits	Description
ID	0	7..0	Message ID=0x03
Timestamp	1 2 3 4	31..24 23..16 15..8 7..0	Free running relative timestamp in ms
IBI	5 6	15..8 7..0	IBI in milliseconds
IBIQI	7	7..0	IBI quality index [0,100]

6.2.7 Device Status Message

Message telling brief status of the device.

Name	Byte	Bits	Description
ID	0	7..0	Message ID=0x09
Timestamp	1 2 3 4	31..24 23..16 15..8 7..0	Free running relative timestamp in ms
Mode	5	7..0	Operating mode the device is currently in.
Reserved	6		Do not interpret at the moment.
	7		
	8		
	9		
	10		

6.2.8 Version information message

Message telling information about the device firmware.

Name	Byte	Description
ID	0	0x0A
reserved	1 2	Reserved for future use
prod id	3	Single product id.
HW reserved	4 5	Reserved for HW information
SW major	6	Major SW version. 0x80 bit indicates unreleased.
SW minor	7	Minor SW version
Protocol major	8	Protocol major version number. 0x80 bit indicates unreleased.
Protocol minor	9	Protocol minor version number.
SW stamp	10	SW build stamp MSB
SW stamp	11	SW build stamp
SW stamp	12	SW build stamp
SW stamp	13	SW build stamp LSB
SW VCS ID	14	SW VCS identifier MSB
SW VCS ID	15	SW VCS identifier
SW VCS ID	16	SW VCS identifier
SW VCS ID	17	SW VCS identifier LSB
reserved	18	Reserved for future use
reserved	19	Reserved for future use
reserved	20	Reserved for future use
reserved	21	Reserved for future use

In practice **SW stamp** is an UNIX timestamp that is taken during the firmware build. **SW VCS ID** is a git commit shorthash represented as unsigned 32 bit integer.

6.2.9 Command Status Message

Message telling brief status of command that was sent to module. Some commands as software update commands generate the status message as reply to indicate the success of the command.

Name	Byte	Bits	Description
ID	0	7..0	Message ID (0x15 in this case)

Name	Byte	Bits	Description
Timestamp	1 2 3 4	31..24 23..16 15..8 7..0	Free running relative timestamp in ms
Command	5	7..0	Command for which this status applies
Status	6	7..0	Status, 0 indicates success
Data	7 8		Command specific extra data

6.2.9.1 Common error codes

Below is a table of some common error codes used in **Status** field that are sent when a particular command fails. For example, an incoming message whose ID is not recognized causes a command status message to be sent for that particular (invalid) command with ERROR_UNKNOWN_MSG_ID as **Status**.

It should be noted that most of the commands don't send a status message when they succeed but rather generate the message explicitly on failure. More information on command specific error codes such as whether status message is sent on successful command is described in the command documentation.

ID	Message	Type	Description
0x00	NO_ERROR	uint8_t	The operation was successful
0x73	ERROR_GENERIC	uint8_t	ID of incoming message or internal operation
0x74	ERROR_UNKNOWN_MSG_ID	uint8_t	ID of incoming message
0x75	ERROR_INVALID_DATA	uint8_t	ID of incoming message
0x76	ERROR_STATE	uint8_t	ID of incoming message or internal operation
0x77	ERROR_PAYLOAD	uint8_t	ID of incoming message or internal operation
0x78	ERROR_CRC	uint8_t	ID of incoming message or internal operation

Glossary and List of Acronyms

ACC

Accelerometer

DNC

Do not connect

HR

Heart rate

Heart rate quality indicator (HRQI)

An indicator of the source and reliability at which the heart rate estimation was made (HRQI = 0 for estimated samples).

Heart rate variability (HRV)

The physiological phenomenon of variation in the time interval between heartbeats. It is measured by the variation in the interbeat interval.

Interbeat interval (IBI)

The time interval between individual heartbeats. Measured in units of milliseconds. In normal heart function, each IBI value varies from beat to beat. This natural variation is known as heart rate variability (HRV).

I²C

Inter-Integrated Circuit, pronounced I-squared-C, is a multi-master, multi-slave, single-ended, serial computer bus. It is used for attaching lower-speed peripherals to processors on embedded systems.

Operating mode

Usage mode and operating mode are used interchangeably.

Optical Heart Rate (OHR)

A heart rate that was measured with the analysis of a photoplethysmogram.

P-OHR1F

A PulseOn optical heart rate sensor module.

R-R interval (RRI)

The interval between successive Rs, where R is a point corresponding to the peak of the QRS complex of the ECG wave.

RRIQI

RRI quality indicator.

SPI

Serial Peripheral Interface (SPI) is a synchronous serial data protocol used by microcontrollers for communicating with one or more peripheral devices.

VO_2Max

The maximum or optimum rate at which the heart, lungs, and muscles can effectively use oxygen during exercise, used as a way of measuring a person's individual aerobic capacity. VO_2max is expressed as a relative rate in millilitres of oxygen per kilogram of body mass per minute (e.g., mL/(kg·min)).

---

1. Requires Firstbeat ETE Library license. ↩
2. Requires Firstbeat ETE Library license. ↩
3. Requires Firstbeat ETE Library license. ↩
4. Algorithm is closely identical to PulseOn Sports & Fitness product algorithm ↩
5. Requires Firstbeat ETE Library license. ↩
6. Estimate available only when Firstbeat ETE library is used ↩
7. Estimate available only when Firstbeat ETE library is used ↩
8. Estimate available only when Firstbeat ETE library is used ↩
9. Estimate available only when Firstbeat ETE library is used ↩
10. Estimate available only when Firstbeat ETE library is used ↩
11. Estimate available only when Firstbeat ETE library is used ↩
12. Estimate available only when Firstbeat ETE library is used ↩
13. Estimate available only when Firstbeat ETE library is used ↩
14. Estimate available only when Firstbeat ETE library is used ↩
15. Estimate available only when Firstbeat ETE library is used ↩