Swift Navigation Binary Protocol

Protocol Specification 0.50

Contents

1	**Overview**	**1**
2	**Message Framing Structure**	**2**
3	**Basic Formats and Payload Structure**	**3**
4	**Message Types**	**4**
5	**Stable Message Definitions**	**6**
5.1	Logging	6
5.2	Navigation	7
5.3	Observation	15
5.4	Settings	19
5.5	System	26
6	**Draft Message Definitions**	**28**
6.1	Acquisition	28
6.2	Bootload	29
6.3	Ext Events	34
6.4	File IO	35
6.5	Flash	42
6.6	Piksi	52
6.7	Tracking	63

1 Overview

The Swift Navigation Binary Protocol (SBP) is a fast, simple, and minimal binary protocol for communicating with Swift devices. It is the native binary protocol used by the Piksi GPS receiver to transmit solutions, observations, status, and debugging messages, as well as receive messages from the host operating system, such as differential corrections and the almanac. As such, it is an important interface with your Piksi receiver and the primary integration method with other systems.

This document provides a specification of SBP framing and the payload structures of the messages currently used with Swift devices. SBP client libraries in a variety of programming languages are available at http://docs.swiftnav.com/wiki/SwiftNav_Binary_Protocol.

2 Message Framing Structure

SBP consists of two pieces:

- an over-the-wire message framing format
- structured payload definitions

As of Version 0.50 , the frame consists of a 6-byte binary header section, a variable-sized payload field, and a 16-bit CRC value. All multibyte values are ordered in **little-endian** format. SBP uses the CCITT CRC16 (XMODEM implementation) for error detection[1].

Offset (bytes)	Size (bytes)	Name	Description
0	1	Preamble	Denotes the start of frame transmission. Always 0x55.
1	2	Message Type	Identifies the payload contents.
3	2	Sender	A unique identifier of the sender. On the Piksi, this is set to the 2 least significant bytes of the device serial number. A stream of SBP messages may also included sender IDs for forwarded messages.
5	1	Length	Length (bytes) of the Payload field.
6	N	Payload	Binary message contents.
$N+6$	2	CRC	Cyclic Redundancy Check of the frame's binary data from the Message Type up to the end of Payload (does not include the Preamble).
	$N+8$		Total Frame Length

Table 2.0.1: Swift Binary Protocol message structure. N denotes a variable-length size.

Swift devices, such as the Piksi, also support the standard NMEA-0183 protocol for single-point position solutions. Observations transmitted via SBP can also be converted into RINEX. Note that NMEA doesn't define a standardized message string for RTK solutions. To make it possible to achieve RTK accuracy with legacy host hardware or software that can only read NMEA, recent firmware versions implement a "pseudo-absolute" mode.

[1]CCITT 16-bit CRC Implementation uses parameters used by XMODEM, i.e. the polynomial: $x^{16}+x^{12}+x^{5}+1$. For more details, please see the implementation at https://github.com/swift-nav/libsbp/blob/master/c/src/edc.c#L59. See also *A Painless Guide to CRC Error Detection Algorithms* at http://www.ross.net/crc/download/crc_v3.txt

3 Basic Formats and Payload Structure

The binary payload of an SBP message decodes into structured data based on the message type defined in the header. SBP uses several primitive numerical and collection types for defining payload contents.

Name	Size (bytes)	Description
s8	1	Signed 8-bit integer
s16	2	Signed 16-bit integer
s32	4	Signed 32-bit integer
s64	8	Signed 64-bit integer
u8	1	Unsigned 8-bit integer
u16	2	Unsigned 16-bit integer
u32	4	Unsigned 32-bit integer
u64	8	Unsigned 64-bit integer
float	4	Single-precision float (IEEE-754)
double	8	Double-precision float (IEEE-754)
array	—	Fixed or variable length array of any fill type
string	—	Fixed or variable length string (NULL padded/terminated)
bitfield	—	A primitive type, typically a u8, can encode boolean and enumerated status flags.

Table 3.0.2: SBP primitive types

Example Message

As an example, consider this framed series of bytes read from a serial port:

`55 02 02 cc 04 14 70 3d d0 18 cf ef ff ff ef e8 ff ff f0 18 00 00 00 00 05 00 43 94`

This byte array decodes into a `MSG_BASELINE_ECEF` (see pg. 9), that reports the baseline position solution of the rover receiver relative to the base station receiver in Earth Centered Earth Fixed (ECEF) coordinates. The segments of this byte array and its contents breakdown as follows:

Field Name	Type	Value	Bytestring Segment
Preamble	u8	0x55	55
Message Type	u16	MSG_BASELINE_ECEF	02 02
Sender	u16	1228	cc 04
Length	u8	20	14
Payload		—	70 3d d0 18 cf ef ff ff ef e8 ff ff f0 18 00 00 00 00 05 00
MSG_BASELINE_ECEF			
.tow	u32	416300400 msec	70 3d d0 18
.x	s32	−4145 mm	cf ef ff ff
.y	s32	−5905 mm	ef e8 ff ff
.z	s32	6384 mm	f0 18 00 00
.accuracy	u16	0	00 00
.nsats	u8	5	05
.flags	u8	0	00
CRC	u16	0x9443	43 94

Table 3.0.3: SBP breakdown for `MSG_BASELINE_ECEF`

4 Message Types

Packages define a logical collection of SBP messages. The contents and layout of messages in packages marked **stable** are unlikely to change in the future. **Draft** messages *will change with future development* and are detailed purely for *informational purposes only*. Many draft messages are implementation-defined, and some collections, such as the acquisition package, are used for internal development.

Package	Msg ID	Name	Size (bytes)	Description
Stable				
Logging	0x0401	MSG_LOG	$N + 1$	Plaintext logging messages with levels
Navigation	0x0100	MSG_GPS_TIME	11	GPS Time
	0x0206	MSG_DOPS	14	Dilution of Precision
	0x0200	MSG_POS_ECEF	32	Single-point position in ECEF
	0x0201	MSG_POS_LLH	34	Geodetic Position
	0x0202	MSG_BASELINE_ECEF	20	Baseline Position in ECEF
	0x0203	MSG_BASELINE_NED	22	Baseline in NED
	0x0204	MSG_VEL_ECEF	20	Velocity in ECEF
	0x0205	MSG_VEL_NED	22	Velocity in NED
Observation	0x0043	MSG_OBS	$16N + 7$	GPS satellite observations
	0x0044	MSG_BASE_POS	24	Base station position
	0x0047	MSG_EPHEMERIS	185	Satellite broadcast ephemeris
Settings	0x00A1	MSG_SETTINGS_SAVE	0	Save settings to flash
	0x00A0	MSG_SETTINGS_WRITE	N	Write device configuration settings
	0x00A4	MSG_SETTINGS_READ_REQ	N	Read device configuration settings
	0x00A5	MSG_SETTINGS_READ_RESP	N	Read device configuration settings
	0x00A2	MSG_SETTINGS_READ_BY_INDEX_REQ	2	Read setting by direct index
	0x00A7	MSG_SETTINGS_READ_BY_INDEX_RESP	$N + 2$	Read setting by direct index
	0x00A6	MSG_SETTINGS_READ_BY_INDEX_DONE	0	Finished reading settings
System	0xFF00	MSG_STARTUP	4	System start-up message
	0xFFFF	MSG_HEARTBEAT	4	System heartbeat message
Draft				
Acquisition	0x0014	MSG_ACQ_RESULT	16	Satellite acquisition result
Bootload	0x00B3	MSG_BOOTLOADER_HANDSHAKE_REQ	0	Bootloading handshake request
	0x00B4	MSG_BOOTLOADER_HANDSHAKE_RESP	$N + 4$	Bootloading handshake response
	0x00B1	MSG_BOOTLOADER_JUMP_TO_APP	1	Bootloader jump to application
	0x00DE	MSG_NAP_DEVICE_DNA_REQ	0	Read FPGA device ID over UART request
	0x00DD	MSG_NAP_DEVICE_DNA_RESP	8	Read FPGA device ID over UART response
Ext Events	0x0101	MSG_EXT_EVENT	12	Reports timestamped external pin event
File IO	0x00A8	MSG_FILEIO_READ_REQ	$N + 9$	Read file from the file system
	0x00A3	MSG_FILEIO_READ_RESP	$N + 4$	File read from the file system
	0x00A9	MSG_FILEIO_READ_DIR_REQ	$N + 8$	List files in a directory
	0x00AA	MSG_FILEIO_READ_DIR_RESP	$N + 4$	Files listed in a directory
	0x00AC	MSG_FILEIO_REMOVE	N	Delete a file from the file system
	0x00AD	MSG_FILEIO_WRITE_REQ	$N + 9$	Write to file
	0x00AB	MSG_FILEIO_WRITE_RESP	4	File written to
Flash	0x00E6	MSG_FLASH_PROGRAM	$N + 5$	Program flash addresses
	0x00E0	MSG_FLASH_DONE	1	Flash response message
	0x00E7	MSG_FLASH_READ_REQ	5	Read STM or M25 flash address request
	0x00E1	MSG_FLASH_READ_RESP	5	Read STM or M25 flash address response
	0x00E2	MSG_FLASH_ERASE	5	Erase sector of device flash memory

	0x00E3	MSG_STM_FLASH_LOCK_SECTOR	4	Lock sector of STM flash memory
	0x00E4	MSG_STM_FLASH_UNLOCK_SECTOR	4	Unlock sector of STM flash memory
	0x00E8	MSG_STM_UNIQUE_ID_REQ	0	Read device's hardcoded unique ID request
	0x00E5	MSG_STM_UNIQUE_ID_RESP	12	Read device's hardcoded unique ID response
	0x00F3	MSG_M25_FLASH_WRITE_STATUS	1	Write M25 flash status register
Piksi	0x0069	MSG_ALMANAC	0	Legacy message to load satellite almanac
	0x0068	MSG_SET_TIME	0	Send GPS time from host
	0x00B2	MSG_RESET	0	Reset the device
	0x00C0	MSG_CW_RESULTS	0	Legacy message for CW interference channel (Piksi =¿ host)
	0x00C1	MSG_CW_START	0	Legacy message for CW interference channel
	0x0022	MSG_RESET_FILTERS	1	Reset IAR filters
	0x0023	MSG_INIT_BASE	0	Initialize IAR from known baseline
	0x0017	MSG_THREAD_STATE	26	State of an RTOS thread
	0x0018	MSG_UART_STATE	58	State of the UART channels
	0x0019	MSG_IAR_STATE	4	State of the Integer Ambiguity Resolution (IAR) process
	0x001B	MSG_MASK_SATELLITE	5	Mask a satellite from use in Piksi subsystems
Tracking	0x0013	MSG_TRACKING_STATE	$9N$	Satellite tracking channel states
	0x001C	MSG_TRACKING_IQ	$8N+5$	Tracking channel correlations

Table 4.0.5: SBP message types

5 Stable Message Definitions

5.1 Logging

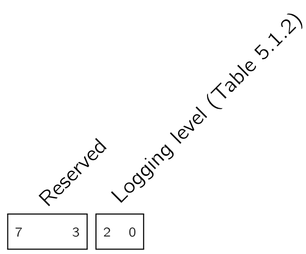
Logging and debugging messages from the device. These are in the implementation-defined range (0x0000-0x00FF).

MSG_LOG — 0x0401

This message contains a human-readable payload string from the device containing errors, warnings and informational messages at ERROR, WARNING, DEBUG, INFO logging levels.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`level`	Logging level
1	N	string		`text`	Human-readable string
	$N+1$				Total Payload Length

Table 5.1.1: MSG_LOG 0x0401 message structure



Field 5.1.1: Logging level (`level`)

Value	Description
0	DEBUG
1	INFO
2	WARN
3	ERROR

Table 5.1.2: Logging level values (`level[0:2]`)

5.2 Navigation

Geodetic navigation messages reporting GPS time, position, velocity, and baseline position solutions. For position solutions, these messages define several different position solutions: single-point (SPP), RTK, and pseudo-absolute position solutions.

The SPP is the standalone, absolute GPS position solution using only a single receiver. The RTK solution is the differential GPS solution, which can use either a fixed/integer or floating carrier phase ambiguity. The pseudo-absolute position solution uses a user-provided, well-surveyed base station position (if available) and the RTK solution in tandem.

MSG_GPS_TIME — 0x0100

This message reports the GPS time, representing the time since the GPS epoch began on midnight January 6, 1980 UTC. GPS time counts the weeks and seconds of the week. The weeks begin at the Saturday/Sunday transition. GPS week 0 began at the beginning of the GPS time scale.

Within each week number, the GPS time of the week is between between 0 and 604800 seconds (=60*60*24*7). Note that GPS time does not accumulate leap seconds, and as of now, has a small offset from UTC. In a message stream, this message precedes a set of other navigation messages referenced to the same time (but lacking the ns field) and indicates a more precise time of these messages.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	2	u16	weeks	`wn`	GPS week number
2	4	u32	ms	`tow`	GPS time of week rounded to the nearest millisecond
6	4	s32	ns	`ns`	Nanosecond residual of millisecond-rounded TOW (ranges from -500000 to 500000)
10	1	u8		`flags`	Status flags (reserved)
	11				Total Payload Length

Table 5.2.1: MSG_GPS_TIME 0x0100 message structure

MSG_DOPS — 0x0206

This dilution of precision (DOP) message describes the effect of navigation satellite geometry on positional measurement precision.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32	ms	`tow`	GPS Time of Week
4	2	u16		`gdop`	Geometric Dilution of Precision
6	2	u16		`pdop`	Position Dilution of Precision
8	2	u16		`tdop`	Time Dilution of Precision
10	2	u16		`hdop`	Horizontal Dilution of Precision
12	2	u16		`vdop`	Vertical Dilution of Precision
	14				Total Payload Length

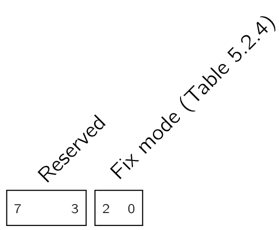
Table 5.2.2: MSG_DOPS 0x0206 message structure

MSG_POS_ECEF — 0x0200

The position solution message reports absolute Earth Centered Earth Fixed (ECEF) coordinates and the status (single point vs pseudo-absolute RTK) of the position solution. If the rover receiver knows the surveyed position of the base station and has an RTK solution, this reports a pseudo-absolute position solution using the base station position and the rover's RTK baseline vector. The full GPS time is given by the preceding MSG_GPS_TIME with the matching time-of-week (tow).

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32	ms	`tow`	GPS Time of Week
4	8	double	m	`x`	ECEF X coordinate
12	8	double	m	`y`	ECEF Y coordinate
20	8	double	m	`z`	ECEF Z coordinate
28	2	u16	mm	`accuracy`	Position accuracy estimate (not implemented). Defaults to 0.
30	1	u8		`n_sats`	Number of satellites used in solution
31	1	u8		`flags`	Status flags
	32				Total Payload Length

Table 5.2.3: MSG_POS_ECEF 0x0200 message structure



Field 5.2.1: Status flags (`flags`)

Value	Description
0	Single Point Positioning (SPP)
1	Float RTK
2	Fixed RTK

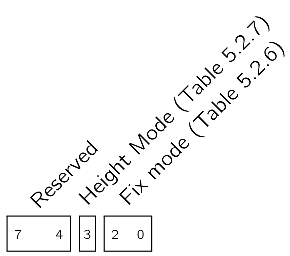
Table 5.2.4: Fix mode values (`flags[0:2]`)

MSG_POS_LLH — 0x0201

This position solution message reports the absolute geodetic coordinates and the status (single point vs pseudo-absolute RTK) of the position solution. If the rover receiver knows the surveyed position of the base station and has an RTK solution, this reports a pseudo-absolute position solution using the base station position and the rover's RTK baseline vector. The full GPS time is given by the preceding MSG_GPS_TIME with the matching time-of-week (tow).

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32	ms	`tow`	GPS Time of Week
4	8	double	deg	`lat`	Latitude
12	8	double	deg	`lon`	Longitude
20	8	double	m	`height`	Height
28	2	u16	mm	`h_accuracy`	Horizontal position accuracy estimate (not implemented). Defaults to 0.
30	2	u16	mm	`v_accuracy`	Vertical position accuracy estimate (not implemented). Defaults to 0.
32	1	u8		`n_sats`	Number of satellites used in solution.
33	1	u8		`flags`	Status flags
	34				Total Payload Length

Table 5.2.5: MSG_POS_LLH 0x0201 message structure



Field 5.2.2: Status flags (`flags`)

Value	Description
0	Single Point Positioning (SPP)
1	Fixed RTK
2	Float RTK

Table 5.2.6: Fix mode values (`flags[0:2]`)

Value	Description
0	Height above Ellipsoid
1	Height above mean sea level

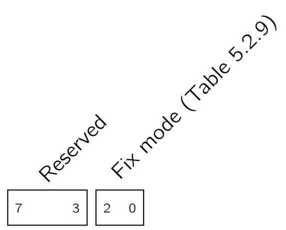
Table 5.2.7: Height Mode values (`flags[3]`)

MSG_BASELINE_ECEF — 0x0202

This message reports the baseline solution in Earth Centered Earth Fixed (ECEF) coordinates. This baseline is the relative vector distance from the base station to the rover receiver. The full GPS time is given by the preceding MSG_GPS_TIME with the matching time-of-week (tow).

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32	ms	`tow`	GPS Time of Week
4	4	s32	mm	`x`	Baseline ECEF X coordinate
8	4	s32	mm	`y`	Baseline ECEF Y coordinate
12	4	s32	mm	`z`	Baseline ECEF Z coordinate
16	2	u16	mm	`accuracy`	Position accuracy estimate (not implemented). Defaults to 0.
18	1	u8		`n_sats`	Number of satellites used in solution
19	1	u8		`flags`	Status flags
	20				Total Payload Length

Table 5.2.8: MSG_BASELINE_ECEF 0x0202 message structure



Field 5.2.3: Status flags (`flags`)

Value	Description
0	Float RTK
1	Fixed RTK

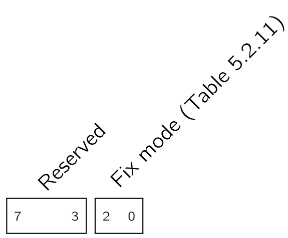
Table 5.2.9: Fix mode values (`flags[0:2]`)

MSG_BASELINE_NED — 0x0203

This message reports the baseline solution in North East Down (NED) coordinates. This baseline is the relative vector distance from the base station to the rover receiver, and NED coordinate system is defined at the local tangent plane centered at the base station position. The full GPS time is given by the preceding MSG_GPS_TIME with the matching time-of-week (tow).

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32	ms	`tow`	GPS Time of Week
4	4	s32	mm	`n`	Baseline North coordinate
8	4	s32	mm	`e`	Baseline East coordinate
12	4	s32	mm	`d`	Baseline Down coordinate
16	2	u16	mm	`h_accuracy`	Horizontal position accuracy estimate (not implemented). Defaults to 0.
18	2	u16	mm	`v_accuracy`	Vertical position accuracy estimate (not implemented). Defaults to 0.
20	1	u8		`n_sats`	Number of satellites used in solution
21	1	u8		`flags`	Status flags
	22				Total Payload Length

Table 5.2.10: MSG_BASELINE_NED 0x0203 message structure



Field 5.2.4: Status flags (`flags`)

Value	Description
0	Float RTK
1	Fixed RTK

Table 5.2.11: Fix mode values (`flags[0:2]`)

MSG_VEL_ECEF — 0x0204

This message reports the velocity in Earth Centered Earth Fixed (ECEF) coordinates. The full GPS time is given by the preceding MSG_GPS_TIME with the matching time-of-week (tow).

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32	ms	`tow`	GPS Time of Week
4	4	s32	mm/s	`x`	Velocity ECEF X coordinate
8	4	s32	mm/s	`y`	Velocity ECEF Y coordinate
12	4	s32	mm/s	`z`	Velocity ECEF Z coordinate
16	2	u16	mm/s	`accuracy`	Velocity accuracy estimate (not implemented). Defaults to 0.
18	1	u8		`n_sats`	Number of satellites used in solution
19	1	u8		`flags`	Status flags (reserved)
	20				Total Payload Length

Table 5.2.12: MSG_VEL_ECEF 0x0204 message structure

MSG_VEL_NED — 0x0205

This message reports the velocity in local North East Down (NED) coordinates. The full GPS time is given by the preceding MSG_GPS_TIME with the matching time-of-week (tow).

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32	ms	`tow`	GPS Time of Week
4	4	s32	mm/s	`n`	Velocity North coordinate
8	4	s32	mm/s	`e`	Velocity East coordinate
12	4	s32	mm/s	`d`	Velocity Down coordinate
16	2	u16	mm/s	`h_accuracy`	Horizontal velocity accuracy estimate (not implemented). Defaults to 0.
18	2	u16	mm/s	`v_accuracy`	Vertical velocity accuracy estimate (not implemented). Defaults to 0.
20	1	u8		`n_sats`	Number of satellites used in solution
21	1	u8		`flags`	Status flags (reserved)
	22				Total Payload Length

Table 5.2.13: MSG_VEL_NED 0x0205 message structure

5.3 Observation

Satellite observation messages from the device.

MSG_OBS — 0x0043

The GPS observations message reports all the raw pseudorange and carrier phase observations for the satellites being tracked by the device. Carrier phase observation here is represented as a 40-bit fixed point number with Q32.8 layout (i.e. 32-bits of whole cycles and 8-bits of fractional cycles).

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32	ms	`header.t.tow`	Milliseconds since start of GPS week
4	2	u16	week	`header.t.wn`	GPS week number
6	1	u8		`header.n_obs`	Total number of observations. First nibble is the size of the sequence (n), second nibble is the zero-indexed counter (ith packet of n)
$16N+7$	4	u32	cm	`obs[N].P`	Pseudorange observation
$16N+11$	4	s32	cycles	`obs[N].L.i`	Carrier phase whole cycles
$16N+15$	1	u8	cycles	`obs[N].L.f`	Carrier phase fractional part
$16N+16$	1	u8	dB Hz	`obs[N].cn0`	Carrier-to-Noise density
$16N+17$	2	u16		`obs[N].lock`	Lock indicator. This value changes whenever a satellite signal has lost and regained lock, indicating that the carrier phase ambiguity may have changed.
$16N+19$	4	u32		`obs[N].sid`	Signal identifier of the satellite signal - values 0x00 through 0x1F represent GPS PRNs 1 through 32 respectively (PRN-1 notation); other values reserved for future use.
	$16N+7$				Total Payload Length

Table 5.3.1: MSG_OBS 0x0043 message structure

MSG_BASE_POS — 0x0044

The base station position message is the position reported by the base station itself. It is used for pseudo-absolute RTK positioning, and is required to be a high-accuracy surveyed location of the base station. Any error here will result in an error in the pseudo-absolute position output.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	8	double	deg	`lat`	Latitude
8	8	double	deg	`lon`	Longitude
16	8	double	m	`height`	Height
	24				Total Payload Length

Table 5.3.2: MSG_BASE_POS 0x0044 message structure

MSG_EPHEMERIS — 0x0047

The ephemeris message returns a set of satellite orbit parameters that is used to calculate GPS satellite position, velocity, and clock offset. Please see the Navstar GPS Space Segment/Navigation user interfaces (ICD-GPS-200, Table 20-III) for more details.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	8	double	s	tgd	Group delay differential between L1 and L2
8	8	double	m	c_rs	Amplitude of the sine harmonic correction term to the orbit radius
16	8	double	m	c_rc	Amplitude of the cosine harmonic correction term to the orbit radius
24	8	double	rad	c_uc	Amplitude of the cosine harmonic correction term to the argument of latitude
32	8	double	rad	c_us	Amplitude of the sine harmonic correction term to the argument of latitude
40	8	double	rad	c_ic	Amplitude of the cosine harmonic correction term to the angle of inclination
48	8	double	rad	c_is	Amplitude of the sine harmonic correction term to the angle of inclination
56	8	double	rad/s	dn	Mean motion difference
64	8	double	radians	m0	Mean anomaly at reference time
72	8	double		ecc	Eccentricity of satellite orbit
80	8	double	m^(1/2)	sqrta	Square root of the semi-major axis of orbit
88	8	double	rad	omega0	Longitude of ascending node of orbit plane at weekly epoch
96	8	double	rad/s	omegadot	Rate of right ascension
104	8	double	rad	w	Argument of perigee
112	8	double	rad	inc	Inclination
120	8	double	rad/s	inc_dot	Inclination first derivative
128	8	double	s	af0	Polynomial clock correction coefficient (clock bias)
136	8	double	s/s	af1	Polynomial clock correction coefficient (clock drift)
144	8	double	s/s^2	af2	Polynomial clock correction coefficient (rate of clock drift)
152	8	double	s	toe_tow	Time of week
160	2	u16	week	toe_wn	Week number
162	8	double	s	toc_tow	Clock reference time of week
170	2	u16	week	toc_wn	Clock reference week number
172	1	u8		valid	Is valid?
173	1	u8		healthy	Satellite is healthy?
174	4	u32		sid	Signal identifier being tracked - values 0x00 through 0x1F represent GPS PRNs 1 through 32 respectively (PRN-1 notation); other values reserved for future use
178	1	u8		iode	Issue of ephemeris data
179	2	u16		iodc	Issue of clock data
181	4	u32		reserved	Reserved field
	185				Total Payload Length

Table 5.3.3: MSG_EPHEMERIS 0x0047 message structure

5.4 Settings

Messages for reading and writing the device's device settings.

These are in the implementation-defined range (0x0000-0x00FF). Note that some of these messages share the same message type ID for both the host request and the device response. See the accompanying document for descriptions of settings configurations and examples: https://github.com/swift-nav/piksi\firmware/blob/master/docs/settings.pdf

MSG_SETTINGS_SAVE — 0x00A1

The save settings message persists the device's current settings configuration to its onboard flash memory file system.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 5.4.1: MSG_SETTINGS_SAVE 0x00A1 message structure

MSG_SETTINGS_WRITE — 0x00A0

The setting message writes the device's configuration.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	*N*	string		`setting`	A NULL-terminated and delimited string with contents [SECTION_SETTING, SETTING, VALUE].
	N				Total Payload Length

Table 5.4.2: MSG_SETTINGS_WRITE 0x00A0 message structure

MSG_SETTINGS_READ_REQ — 0x00A4

The setting message reads the device's configuration.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	N	string		`setting`	A NULL-terminated and delimited string with contents [SECTION_SETTING, SETTING].
	N				Total Payload Length

Table 5.4.3: MSG_SETTINGS_READ_REQ 0x00A4 message structure

MSG_SETTINGS_READ_RESP — 0x00A5

The setting message reads the device's configuration.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	*N*	string		`setting`	A NULL-terminated and delimited string with contents [SECTION_SETTING, SETTING, VALUE].
	N				Total Payload Length

Table 5.4.4: MSG_SETTINGS_READ_RESP 0x00A5 message structure

MSG_SETTINGS_READ_BY_INDEX_REQ — 0x00A2

The settings message for iterating through the settings values. It will read the setting at an index, returning a NULL-terminated and delimited string with contents [SECTION_SETTING, SETTING, VALUE].

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	2	u16		`index`	An index into the device settings, with values ranging from 0 to length(settings)
	2				Total Payload Length

Table 5.4.5: MSG_SETTINGS_READ_BY_INDEX_REQ 0x00A2 message structure

MSG_SETTINGS_READ_BY_INDEX_RESP — 0x00A7

The settings message for iterating through the settings values. It will read the setting at an index, returning a NULL-terminated and delimited string with contents [SECTION_SETTING, SETTING, VALUE].

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	2	u16		`index`	An index into the device settings, with values ranging from 0 to length(settings)
2	N	string		`setting`	A NULL-terminated and delimited string with contents [SECTION_SETTING, SETTING, VALUE].
	$N + 2$				Total Payload Length

Table 5.4.6: MSG_SETTINGS_READ_BY_INDEX_RESP 0x00A7 message structure

MSG_SETTINGS_READ_BY_INDEX_DONE — 0x00A6

The settings message for indicating end of the settings values.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 5.4.7: MSG_SETTINGS_READ_BY_INDEX_DONE 0x00A6 message structure

5.5 System

Standardized system messages from Swift Navigation devices.

MSG_STARTUP — 0xFF00

The system start-up message is sent once on system start-up. It notifies the host or other attached devices that the system has started and is now ready to respond to commands or configuration requests.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`reserved`	Reserved
	4				Total Payload Length

Table 5.5.1: MSG_STARTUP 0xFF00 message structure

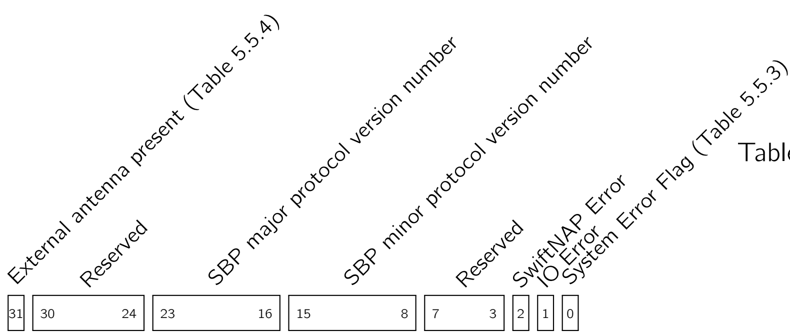
MSG_HEARTBEAT — 0xFFFF

The heartbeat message is sent periodically to inform the host or other attached devices that the system is running. It is used to monitor system malfunctions. It also contains status flags that indicate to the host the status of the system and whether it is operating correctly. Currently, the expected heartbeat interval is 1 sec.

The system error flag is used to indicate that an error has occurred in the system. To determine the source of the error, the remaining error flags should be inspected.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`flags`	Status flags
	4				Total Payload Length

Table 5.5.2: MSG_HEARTBEAT 0xFFFF message structure



Field 5.5.1: Status flags (`flags`)

Value	Description
0	System Healthy
1	An error has occurred

Table 5.5.3: System Error Flag values (`flags[0]`)

Value	Description
0	No external antenna detected
1	External antenna is present

Table 5.5.4: External antenna present values (`flags[31]`)

6 Draft Message Definitions

6.1 Acquisition

Satellite acquisition messages from the device.

MSG_ACQ_RESULT — 0x0014

This message describes the results from an attempted GPS signal acquisition search for a satellite PRN over a code phase/carrier frequency range. It contains the parameters of the point in the acquisition search space with the best signal-to-noise (SNR) ratio.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	float		`snr`	SNR of best point. Currently dimensonless, but will have units of dB Hz in the revision of this message.
4	4	float	chips	`cp`	Code phase of best point
8	4	float	hz	`cf`	Carrier frequency of best point
12	4	u32		`sid`	Signal identifier of the satellite signal for which acquisition was attempted - values 0x00 through 0x1F represent GPS PRNs 1 through 32 respectively (PRN-1 notation); other values reserved for future use.
	16				Total Payload Length

Table 6.1.1: MSG_ACQ_RESULT 0x0014 message structure

6.2 Bootload

Messages for the bootloading configuration on the device.

These are in the implementation-defined range (0x0000-0x00FF), and are intended for internal use only. Note that some of these messages share the same message type ID for both the host request and the device response.

MSG_BOOTLOADER_HANDSHAKE_REQ — 0x00B3

The handshake message request from the host establishes a handshake between the device bootloader and the host. The response from the device is MSG_BOOTLOADER_HANDSHAKE_RESP.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 6.2.1: MSG_BOOTLOADER_HANDSHAKE_REQ 0x00B3 message structure

MSG_BOOTLOADER_HANDSHAKE_RESP — 0x00B4

The handshake message response from the device establishes a handshake between the device bootloader and the host. The request from the host is MSG_BOOTLOADER_HANDSHAKE_REQ. The payload contains the bootloader version number and the SBP protocol version number.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`flags`	Bootloader flags
4	N	string		`version`	Bootloader version number
	$N + 4$				Total Payload Length

Table 6.2.2: MSG_BOOTLOADER_HANDSHAKE_RESP 0x00B4 message structure

MSG_BOOTLOADER_JUMP_TO_APP — 0x00B1

The host initiates the bootloader to jump to the application.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`jump`	Ignored by the device
	1				Total Payload Length

Table 6.2.3: MSG_BOOTLOADER_JUMP_TO_APP 0x00B1 message structure

MSG_NAP_DEVICE_DNA_REQ — 0x00DE

The device message from the host reads a unique device identifier from the SwiftNAP, an FPGA. The host requests the ID by sending a MSG_NAP_DEVICE_DNA_REQ message. The device responds with a MSG_NAP_DEVICE_DNA_RESP message with the device ID in the payload. Note that this ID is tied to the FPGA, and not related to the Piksi's serial number.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 6.2.4: MSG_NAP_DEVICE_DNA_REQ 0x00DE message structure

MSG_NAP_DEVICE_DNA_RESP — 0x00DD

The device message from the host reads a unique device identifier from the SwiftNAP, an FPGA. The host requests the ID by sending a MSG_NAP_DEVICE_DNA_REQ message. The device responds with a MSG_NAP_DEVICE_DNA_RESP messagage with the device ID in the payload. Note that this ID is tied to the FPGA, and not related to the Piksi's serial number.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	8	u8[8]		`dna`	57-bit SwiftNAP FPGA Device ID. Remaining bits are padded on the right.
	8				Total Payload Length

Table 6.2.5: MSG_NAP_DEVICE_DNA_RESP 0x00DD message structure

6.3 Ext Events

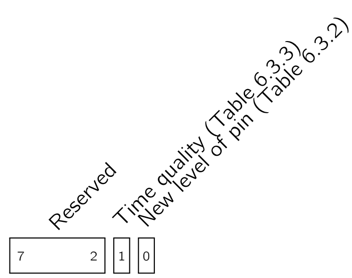
Messages reporting accurately-timestamped external events, e.g. camera shutter time.

MSG_EXT_EVENT — 0x0101

Reports detection of an external event, the GPS time it occurred, which pin it was and whether it was rising or falling.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	2	u16	weeks	`wn`	GPS week number
2	4	u32	ms	`tow`	GPS time of week rounded to the nearest millisecond
6	4	s32	ns	`ns`	Nanosecond residual of millisecond-rounded TOW (ranges from -500000 to 500000)
10	1	u8		`flags`	Flags
11	1	u8		`pin`	Pin number. 0..9 = DEBUG0..9.
	12				Total Payload Length

Table 6.3.1: MSG_EXT_EVENT 0x0101 message structure



Field 6.3.1: Flags (`flags`)

Value	Description
0	Low (falling edge)
1	High (rising edge)

Table 6.3.2: New level of pin values (`flags[0]`)

Value	Description
0	Unknown - don't have nav solution
1	Good (¡ 1 microsecond)

Table 6.3.3: Time quality values (`flags[1]`)

6.4 File IO

Messages for using device's onboard flash filesystem functionality. This allows data to be stored persistently in the device's program flash with wear-levelling using a simple filesystem interface. The file system interface (CFS) defines an abstract API for reading directories and for reading and writing files.

These are in the implementation-defined range (0x0000-0x00FF), and intended for internal-use only. Note that some of these messages share the same message type ID for both the host request and the device response.

MSG_FILEIO_READ_REQ — 0x00A8

The file read message reads a certain length (up to 255 bytes) from a given offset into a file, and returns the data in a MSG_FILEIO_READ_RESP message where the message length field indicates how many bytes were succesfully read.The sequence number in the request will be returned in the response.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`sequence`	Read sequence number
4	4	u32	bytes	`offset`	File offset
8	1	u8	bytes	`chunk_size`	Chunk size to read
9	N	string		`filename`	Name of the file to read from
	$N + 9$				Total Payload Length

Table 6.4.1: MSG_FILEIO_READ_REQ 0x00A8 message structure

MSG_FILEIO_READ_RESP — 0x00A3

The file read message reads a certain length (up to 255 bytes) from a given offset into a file, and returns the data in a message where the message length field indicates how many bytes were succesfully read. The sequence number in the response is preserved from the request.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`sequence`	Read sequence number
4	N	u8[N]		`contents`	Contents of read file
	$N + 4$				Total Payload Length

Table 6.4.2: MSG_FILEIO_READ_RESP 0x00A3 message structure

MSG_FILEIO_READ_DIR_REQ — 0x00A9

The read directory message lists the files in a directory on the device's onboard flash file system. The offset parameter can be used to skip the first n elements of the file list. Returns a MSG_FILEIO_READ_DIR_RESP message containing the directory listings as a NULL delimited list. The listing is chunked over multiple SBP packets. The sequence number in the request will be returned in the response.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`sequence`	Read sequence number
4	4	u32		`offset`	The offset to skip the first n elements of the file list
8	N	string		`dirname`	Name of the directory to list
	$N+8$				Total Payload Length

Table 6.4.3: MSG_FILEIO_READ_DIR_REQ 0x00A9 message structure

MSG_FILEIO_READ_DIR_RESP — 0x00AA

The read directory message lists the files in a directory on the device's onboard flash file system. Message contains the directory listings as a NULL delimited list. The listing is chunked over multiple SBP packets and the end of the list is identified by an entry containing just the character 0xFF. The sequence number in the response is preserved from the request.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`sequence`	Read sequence number
4	N	u8[N]		`contents`	Contents of read directory
	$N+4$				Total Payload Length

Table 6.4.4: MSG_FILEIO_READ_DIR_RESP 0x00AA message structure

MSG_FILEIO_REMOVE — 0x00AC

The file remove message deletes a file from the file system.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	*N*	string		`filename`	Name of the file to delete
	N				Total Payload Length

Table 6.4.5: MSG_FILEIO_REMOVE 0x00AC message structure

MSG_FILEIO_WRITE_REQ — 0x00AD

The file write message writes a certain length (up to 255 bytes) of data to a file at a given offset. Returns a copy of the original MSG_FILEIO_WRITE_RESP message to check integrity of the write. The sequence number in the request will be returned in the response.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`sequence`	Write sequence number
4	4	u32	bytes	`offset`	Offset into the file at which to start writing in bytes
8	N	string		`filename`	Name of the file to write to
9	N	u8[N]		`data`	Variable-length array of data to write
	$N+9$				Total Payload Length

Table 6.4.6: MSG_FILEIO_WRITE_REQ 0x00AD message structure

MSG_FILEIO_WRITE_RESP — 0x00AB

The file write message writes a certain length (up to 255 bytes) of data to a file at a given offset. The message is a copy of the original MSG_FILEIO_WRITE_REQ message to check integrity of the write. The sequence number in the response is preserved from the request.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`sequence`	Write sequence number
	4				Total Payload Length

Table 6.4.7: MSG_FILEIO_WRITE_RESP 0x00AB message structure

6.5 Flash

Messages for reading/writing the device's onboard flash memory. Many of these messages target specific flash memory peripherals used in Swift Navigation devices: the STM32 flash and the M25Pxx FPGA configuration flash.

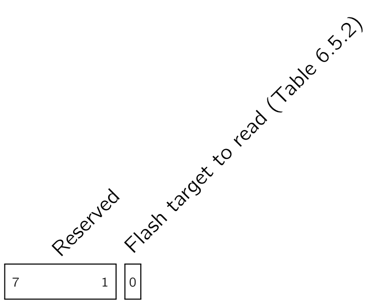
These are in the implementation-defined range (0x0000-0x00FF), and are intended for internal-use only.

MSG_FLASH_PROGRAM — 0x00E6

The flash program message programs a set of addresses of either the STM or M25 flash. The device replies with either a MSG_FLASH_DONE message containing the return code FLASH_OK (0) on success, or FLASH_INVALID_LEN (2) if the maximum write size is exceeded. Note that the sector-containing addresses must be erased before addresses can be programmed.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`target`	Target flags
1	3	u8[3]	bytes	`addr_start`	Starting address offset to program
4	1	u8	bytes	`addr_len`	Length of set of addresses to program, counting up from starting address
5	N	u8[N]		`data`	Data to program addresses with, with length N=addr_len
	$N+5$				Total Payload Length

Table 6.5.1: MSG_FLASH_PROGRAM 0x00E6 message structure



Field 6.5.1: Target flags (`target`)

Value	Description
0	FLASH_STM
1	FLASH_M25

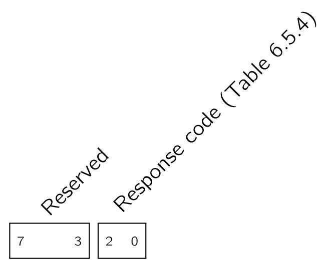
Table 6.5.2: Flash target to read values (`target[0]`)

MSG_FLASH_DONE — 0x00E0

This message defines success or failure codes for a variety of flash memory requests from the host to the device. Flash read and write messages, such as MSG_FLASH_READ_REQ, or MSG_FLASH_PROGRAM, may return this message on failure.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`response`	Response flags
	1				Total Payload Length

Table 6.5.3: MSG_FLASH_DONE 0x00E0 message structure



Field 6.5.2: Response flags (`response`)

Value	Description
0	FLASH_OK
1	FLASH_INVALID_FLASH
2	FLASH_INVALID_LEN
3	FLASH_INVALID_ADDR
4	FLASH_INVALID_RANGE
5	FLASH_INVALID_SECTOR

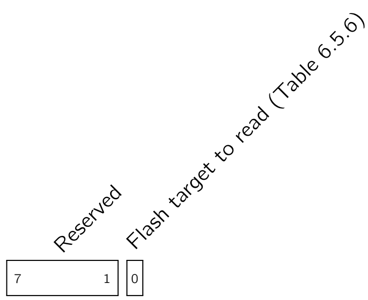
Table 6.5.4: Response code values (`response[0:2]`)

MSG_FLASH_READ_REQ — 0x00E7

The flash read message reads a set of addresses of either the STM or M25 onboard flash. The device replies with a MSG_FLASH_READ_RESP message containing either the read data on success or a MSG_FLASH_DONE message containing the return code FLASH_INVALID_LEN (2) if the maximum read size is exceeded or FLASH_INVALID_ADDR (3) if the address is outside of the allowed range.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`target`	Target flags
1	3	u8[3]	bytes	`addr_start`	Starting address offset to read from
4	1	u8	bytes	`addr_len`	Length of set of addresses to read, counting up from starting address
	5				Total Payload Length

Table 6.5.5: MSG_FLASH_READ_REQ 0x00E7 message structure



Field 6.5.3: Target flags (`target`)

Value	Description
0	FLASH_STM
1	FLASH_M25

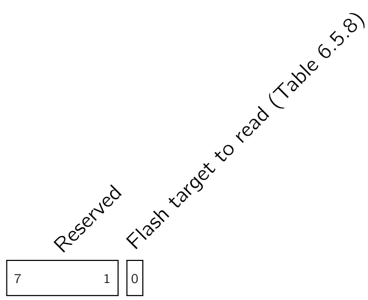
Table 6.5.6: Flash target to read values (`target[0]`)

MSG_FLASH_READ_RESP — 0x00E1

The flash read message reads a set of addresses of either the STM or M25 onboard flash. The device replies with a MSG_FLASH_READ_RESP message containing either the read data on success or a MSG_FLASH_DONE message containing the return code FLASH_INVALID_LEN (2) if the maximum read size is exceeded or FLASH_INVALID_ADDR (3) if the address is outside of the allowed range.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`target`	Target flags
1	3	u8[3]	bytes	`addr_start`	Starting address offset to read from
4	1	u8	bytes	`addr_len`	Length of set of addresses to read, counting up from starting address
	5				Total Payload Length

Table 6.5.7: MSG_FLASH_READ_RESP 0x00E1 message structure



Field 6.5.4: Target flags (`target`)

Value	Description
0	FLASH_STM
1	FLASH_M25

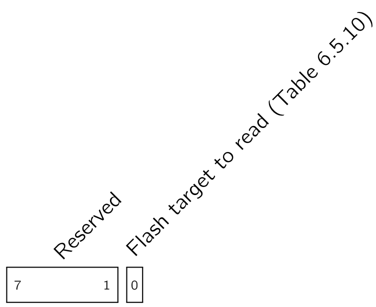
Table 6.5.8: Flash target to read values (`target[0]`)

MSG_FLASH_ERASE — 0x00E2

The flash erase message from the host erases a sector of either the STM or M25 onboard flash memory. The device will reply with a MSG_FLASH_DONE message containing the return code - FLASH_OK (0) on success or FLASH_INVALID_FLASH (1) if the flash specified is invalid.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`target`	Target flags
1	4	u32		`sector_num`	Flash sector number to erase (0-11 for the STM, 0-15 for the M25)
	5				Total Payload Length

Table 6.5.9: MSG_FLASH_ERASE 0x00E2 message structure



Field 6.5.5: Target flags (`target`)

Value	Description
0	FLASH_STM
1	FLASH_M25

Table 6.5.10: Flash target to read values (`target[0]`)

MSG_STM_FLASH_LOCK_SECTOR — 0x00E3

The flash lock message locks a sector of the STM flash memory. The device replies with a MSG_FLASH_DONE message.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`sector`	Flash sector number to lock
	4				Total Payload Length

Table 6.5.11: MSG_STM_FLASH_LOCK_SECTOR 0x00E3 message structure

MSG_STM_FLASH_UNLOCK_SECTOR — 0x00E4

The flash unlock message unlocks a sector of the STM flash memory. The device replies with a MSG_FLASH_DONE message.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`sector`	Flash sector number to unlock
	4				Total Payload Length

Table 6.5.12: MSG_STM_FLASH_UNLOCK_SECTOR 0x00E4 message structure

MSG_STM_UNIQUE_ID_REQ — 0x00E8

This message reads the device's hardcoded unique ID. The host requests the ID by sending a MSG_STM_UNIQUE_ID_REQ. The device responds with a MSG_STM_UNIQUE_ID_RESP with the 12-byte unique ID in the payload.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 6.5.13: MSG_STM_UNIQUE_ID_REQ 0x00E8 message structure

MSG_STM_UNIQUE_ID_RESP — 0x00E5

This message reads the device's hardcoded unique ID. The host requests the ID by sending a MSG_STM_UNIQUE_ID_REQ. The device responds with a MSG_STM_UNIQUE_ID_RESP with the 12-byte unique ID in the payload..

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	12	u8[12]		`stm_id`	Device unique ID
	12				Total Payload Length

Table 6.5.14: MSG_STM_UNIQUE_ID_RESP 0x00E5 message structure

MSG_M25_FLASH_WRITE_STATUS — 0x00F3

The flash status message writes to the 8-bit M25 flash status register. The device replies with a MSG_FLASH_DONE message.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8[1]		`status`	Byte to write to the M25 flash status register
	1				Total Payload Length

Table 6.5.15: MSG_M25_FLASH_WRITE_STATUS 0x00F3 message structure

6.6 Piksi

System health, configuration, and diagnostic messages specific to the Piksi L1 receiver, including a variety of legacy messages that may no longer be used.

These messages are in the implementation-defined range (0x0000-0x00FF), and largely intended for internal-use only.

MSG_ALMANAC — 0x0069

This is a legacy message for sending and loading a satellite alamanac onto the Piksi's flash memory from the host.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 6.6.1: MSG_ALMANAC 0x0069 message structure

MSG_SET_TIME — 0x0068

This message sets up timing functionality using a coarse GPS time estimate sent by the host.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 6.6.2: MSG_SET_TIME 0x0068 message structure

MSG_RESET — 0x00B2

This message from the host resets the Piksi back into the bootloader.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 6.6.3: MSG_RESET 0x00B2 message structure

MSG_CW_RESULTS — 0x00C0

This is an unused legacy message for result reporting from the CW interference channel on the SwiftNAP. This message will be removed in a future release.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 6.6.4: MSG_CW_RESULTS 0x00C0 message structure

MSG_CW_START — 0x00C1

This is an unused legacy message from those host for starting the CW interference channel on the SwiftNAP. This message will be removed in a future release.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

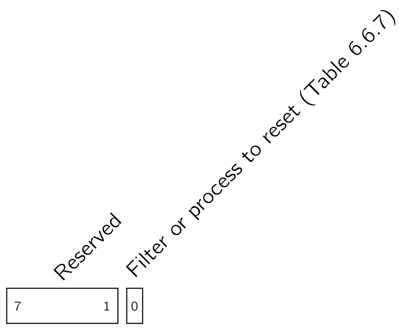
Table 6.6.5: MSG_CW_START 0x00C1 message structure

MSG_RESET_FILTERS — 0x0022

This message resets either the DGNSS Kalman filters or Integer Ambiguity Resolution (IAR) process.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`filter`	Filter flags
	1				Total Payload Length

Table 6.6.6: MSG_RESET_FILTERS 0x0022 message structure



Field 6.6.1: Filter flags (`filter`)

Value	Description
0	DGNSS filter
1	IAR process

Table 6.6.7: Filter or process to reset values (`filter[0]`)

MSG_INIT_BASE — 0x0023

This message initializes the integer ambiguity resolution (IAR) process on the Piksi to use an assumed baseline position between the base station and rover receivers. Warns via MSG_PRINT if there aren't a shared minimum number (4) of satellite observations between the two.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
	0				Total Payload Length

Table 6.6.8: MSG_INIT_BASE 0x0023 message structure

MSG_THREAD_STATE — 0x0017

The thread usage message from the device reports real-time operating system (RTOS) thread usage statistics for the named thread. The reported percentage values require to be normalized.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	20	string		`name`	Thread name (NULL terminated)
20	2	u16		`cpu`	Percentage cpu use for this thread. Values range from 0 - 1000 and needs to be renormalized to 100
22	4	u32	bytes	`stack_free`	Free stack space for this thread
	26				Total Payload Length

Table 6.6.9: MSG_THREAD_STATE 0x0017 message structure

MSG_UART_STATE — 0x0018

The UART message reports data latency and throughput of the UART channels providing SBP I/O. On the default Piksi configuration, UARTs A and B are used for telemetry radios, but can also be host access ports for embedded hosts, or other interfaces in future. The reported percentage values require to be normalized.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	float	kB/s	`uart_a.tx_throughput`	UART transmit throughput
4	4	float	kB/s	`uart_a.rx_throughput`	UART receive throughput
8	2	u16		`uart_a.crc_error_count`	UART CRC error count
10	2	u16		`uart_a.io_error_count`	UART IO error count
12	1	u8		`uart_a.tx_buffer_level`	UART transmit buffer percentage utilization (ranges from 0 - 255)
13	1	u8		`uart_a.rx_buffer_level`	UART receive buffer percentage utilization (ranges from 0 to 255)
14	4	float	kB/s	`uart_b.tx_throughput`	UART transmit throughput
18	4	float	kB/s	`uart_b.rx_throughput`	UART receive throughput
22	2	u16		`uart_b.crc_error_count`	UART CRC error count
24	2	u16		`uart_b.io_error_count`	UART IO error count
26	1	u8		`uart_b.tx_buffer_level`	UART transmit buffer percentage utilization (ranges from 0 - 255)
27	1	u8		`uart_b.rx_buffer_level`	UART receive buffer percentage utilization (ranges from 0 to 255)
28	4	float	kB/s	`uart_ftdi.tx_throughput`	UART transmit throughput
32	4	float	kB/s	`uart_ftdi.rx_throughput`	UART receive throughput
36	2	u16		`uart_ftdi.crc_error_count`	UART CRC error count
38	2	u16		`uart_ftdi.io_error_count`	UART IO error count
40	1	u8		`uart_ftdi.tx_buffer_level`	UART transmit buffer percentage utilization (ranges from 0 - 255)
41	1	u8		`uart_ftdi.rx_buffer_level`	UART receive buffer percentage utilization (ranges from 0 to 255)
42	4	s32	ms	`latency.avg`	Average latency
46	4	s32	ms	`latency.lmin`	Minimum latency
50	4	s32	ms	`latency.lmax`	Maximum latency
54	4	s32	ms	`latency.current`	Smoothed estimate of the current latency
	58				Total Payload Length

Table 6.6.10: MSG_UART_STATE 0x0018 message structure

MSG_IAR_STATE — 0x0019

This message reports the state of the Integer Ambiguity Resolution (IAR) process, which resolves unknown integer ambiguities from double-differenced carrier-phase measurements from satellite observations.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	4	u32		`num_hyps`	Number of integer ambiguity hypotheses remaining
	4				Total Payload Length

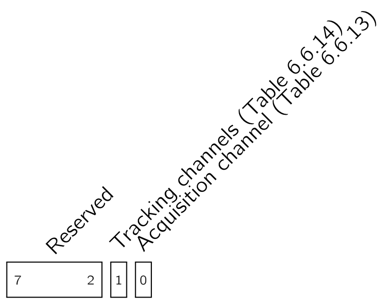
Table 6.6.11: MSG_IAR_STATE 0x0019 message structure

MSG_MASK_SATELLITE — 0x001B

This message allows setting a mask to prevent a particular satellite from being used in various Piksi subsystems.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`mask`	Mask of systems that should ignore this satellite.
1	4	u32		`sid`	Signal identifier for which the mask is applied
	5				Total Payload Length

Table 6.6.12: MSG_MASK_SATELLITE 0x001B message structure



Field 6.6.2: Mask of systems that should ignore this satellite. (`mask`)

Value	Description
0	Enabled
1	Skip this satellite on future acquisitions

Table 6.6.13: Acquisition channel values (`mask[0]`)

Value	Description
0	Enabled
1	Drop this PRN if currently tracking

Table 6.6.14: Tracking channels values (`mask[1]`)

6.7 Tracking

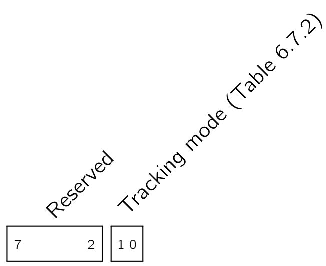
Satellite code and carrier-phase tracking messages from the device.

MSG_TRACKING_STATE — 0x0013

The tracking message returns a variable-length array of tracking channel states. It reports status and snr power measurements for all tracked satellites.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
$9N + 0$	1	u8		states[N].state	Status of tracking channel
$9N + 1$	4	u32		states[N].sid	Signal identifier being tracked - values 0x00 through 0x1F represent GPS PRNs 1 through 32 respectively (PRN-1 notation); other values reserved for future use
$9N + 5$	4	float	dB Hz	states[N].cn0	Carrier-to-noise density
	$9N$				Total Payload Length

Table 6.7.1: MSG_TRACKING_STATE 0x0013 message structure



Field 6.7.1: Status of tracking channel (state)

Value	Description
0	Disabled
1	Running

Table 6.7.2: Tracking mode values (state[0:1])

MSG_TRACKING_IQ — 0x001C

When enabled, a tracking channel can output the correlations at each update interval.

Offset (bytes)	Size (bytes)	Format	Units	Name	Description
0	1	u8		`channel`	Tracking channel of origin
1	4	u32		`sid`	Signal identifier being tracked - values 0x00 through 0x1F represent GPS PRNs 1 through 32 respectively (PRN-1 notation); other values reserved for future use
$8N+5$	4	s32		`corrs[N].I`	In-phase correlation
$8N+9$	4	s32		`corrs[N].Q`	Quadrature correlation
	$8N+5$				Total Payload Length

Table 6.7.3: MSG_TRACKING_IQ 0x001C message structure