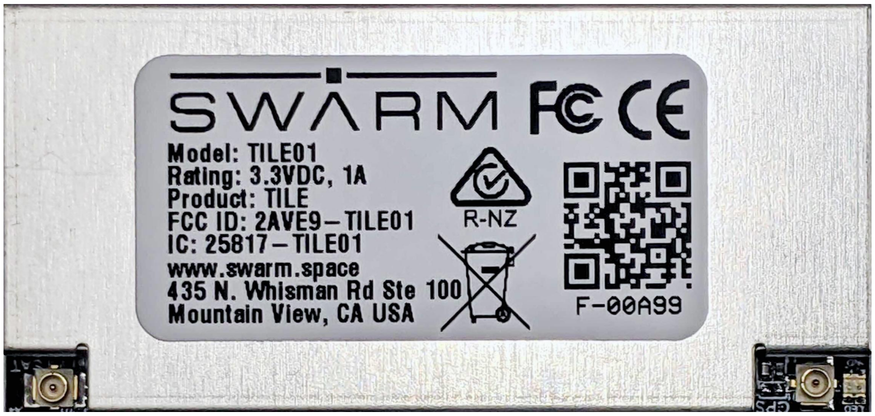
Swarm Tile

Product Manual



Swarm Tile Product Manual
Revision 1.35
November 2021

Legal Notices

This User Manual provides user information and is provided “as is.” Swarm Technologies and its affiliated companies, directors, officers, employees, agents, trustees or consultants (“Swarm”) assume no responsibility for any typographical, technical, content or other inaccuracies in this User Manual. Swarm reserves the right to revise this User Manual or withdraw it at any time without prior notice. You can find the current version of this Swarm Tile User Manual at www.swarm.space.

SWARM MAKES NO GUARANTEES, CONDITIONS, WARRANTIES, OR REPRESENTATIONS, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY IMPLIED REPRESENTATIONS, GUARANTEES, CONDITIONS OR WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, SATISFACTORY QUALITY, NON-INTERFERENCE, ACCURACY OF INFORMATIONAL CONTENT, OR ARISING FROM A COURSE OF DEALING, LAW, USAGE, OR TRADE PRACTICE, USE, OR RELATED TO THE PERFORMANCE OR NONPERFORMANCE OF ANY PRODUCTS, ACCESSORIES, FACILITIES, SERVICES OR USER INFORMATION, EXCEPT AS EXPRESSLY STATED IN THE LIMITED WARRANTY. ANY OTHER STANDARDS OF PERFORMANCE, GUARANTEES, CONDITIONS AND WARRANTIES ARE HEREBY EXPRESSLY EXCLUDED AND DISCLAIMED TO THE FULLEST EXTENT PERMITTED BY LAW. THIS DISCLAIMER AND EXCLUSION SHALL APPLY EVEN IF THE EXPRESS LIMITED WARRANTY PROVIDED IN THE LEGAL INFORMATION BOOKLET FAILS OF ITS ESSENTIAL PURPOSE. FURTHER, THE LIMITED WARRANTY COVERS THE SWARM TILE (INCLUDING HARDWARE, SOFTWARE AND/OR FIRMWARE) AND ACCESSORIES ONLY, AND NO WARRANTY IS MADE AS TO USER INFORMATION AND/ OR COVERAGE, AVAILABILITY OR GRADE OF SERVICE PROVIDED BY SWARM SEPARATELY FOR SWARM SATELLITE SERVICES.

IN NO EVENT SHALL SWARM BE LIABLE, WHETHER IN CONTRACT OR TORT OR ANY OTHER LEGAL THEORY, INCLUDING WITHOUT LIMITATION STRICT LIABILITY, GROSS NEGLIGENCE OR NEGLIGENCE, OR IMPLIED WARRANTY, FOR ANY DAMAGES IN EXCESS OF THE PURCHASE PRICE OF THE Swarm Tile (INCLUDING HARDWARE, SOFTWARE AND/OR FIRMWARE) AND/OR ACCESSORIES AND/OR THE COST OF SWARM SATELLITE SERVICES PROVIDED, NOR SHALL SWARM BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, OR PUNITIVE DAMAGES OF ANY KIND, OR LOSS OF REVENUE OR PROFITS, LOSS OF BUSINESS, LOSS OF PRIVACY, LOSS OF USE, LOSS OF TIME OR INCONVENIENCE, LOSS OF INFORMATION OR DATA, SOFTWARE OR APPLICATIONS OR OTHER FINANCIAL LOSS CAUSED BY THE SWARM TILE (INCLUDING HARDWARE, SOFTWARE AND/ OR FIRMWARE) AND/OR ACCESSORIES AND/OR THE SWARM SATELLITE SERVICES, OR ARISING OUT OF OR IN CONNECTION WITH THE ABILITY OR INABILITY TO USE THE SWARM TILE (INCLUDING HARDWARE, SOFTWARE AND/OR FIRMWARE) AND/OR ACCESSORIES AND/OR THE SWARM SATELLITE SERVICES , TO THE FULLEST EXTENT THESE DAMAGES MAY BE DISCLAIMED BY LAW AND REGARDLESS OF WHETHER SWARM WAS ADVISED OF THE POSSIBILITIES OF SUCH DAMAGES.

Third Party Information

This User Manual might refer to third party sources of information, hardware or software, products or services and/or third party web sites (“third party information”). Swarm does not control, and is not responsible for, any third party information, including without limitation the content, accuracy, copyright compliance, compatibility, performance, trustworthiness, legality, decency, links, or any other aspect of third party information. The inclusion of such third party information does not imply endorsement by Swarm of the third party information. ANY THIRD PARTY INFORMATION THAT IS PROVIDED WITH SWARM’S FACILITIES, SERVICES, PRODUCTS OR USER INFORMATION IS PROVIDED “AS IS.’’ SWARM MAKES NO REPRESENTATIONS, GUARANTEES OR WARRANTIES IN RELATION TO THIRD PARTY INFORMATION AND SWARM SHALL NOT BE LIABLE FOR ANY LOSSES, DAMAGES, LIABILITIES, JUDGMENTS, FINES, AMOUNTS PAID IN SETTLEMENT, EXPENSES OR COSTS OF DEFENSE SUSTAINED IN RELATION TO ANY SUCH THIRD PARTY INFORMATION.

Intellectual Property, Trade Secret, Proprietary or Copyrighted Information

To protect Swarm proprietary and confidential information and/or trade secrets, this User Manual may describe some aspects of Swarm technology in generalized terms. Swarm products may include copyrighted Swarm and third party software. Any such copyrighted software contained in Swarm products may not be modified, reverse engineered, distributed or reproduced in any manner to the extent provided by law. The purchase of any Swarm products shall not be deemed to grant either directly or by implication or otherwise, any license under copyrights, patents, or patent applications of Swarm or any third party software provider, except for the normal, nonexclusive, royalty free license to use that arises by operation of law in the sale of a product.

Content Copyright

You are exclusively responsible for the use of the Swarm Tile, including proper use of third party copyrighted materials. If you violate these terms you agree to defend, indemnify and hold Swarm harmless with respect to any claims or actions by third parties related to your improper use of copyrighted materials and to pay all costs, damages, fines and other amounts incurred by Swarm, or on its behalf, in the defense of any such claims or actions.

Export Compliance

The Swarm Tile is controlled by the export laws and regulations of the United States of America. The U.S. Government may restrict the export or re-export of this product to certain individuals and/or destinations. For further information, contact the U.S. Department of Commerce, Bureau of Industry and Security or visit www.bis.doc.gov

Revision History

Revision	Date	Comment
0.95	03/16/2020	Tile Product Manual - Initial Release
1.00	6/3/2020	Updated Firmware Commands to reflect version 0.7.3. Improved specifications on Tile mechanical and electrical sections
1.10	10/5/2020	Including EU RED Declaration of Conformity Enhanced integration guidance Update to firmware commands to reflect firmware version 0.9.0
1.15	1/12/2021	Enhanced antenna and ground plane specifications Update to firmware commands to reflect firmware version 0.9.4
1.20	3/26/2021	Updated details for Swarm antennas Updated specifications for commands Added firmware version 1.0.0 + specific commands $MM and $RD for 2-way communication
1.25	6/18/2021	Updated Tile MSL Specifications Updated Tile max current consumption (Transmission) to 1 Ampere Updated Figure 4, and Pinout Table (Table 7) to match reference designs and supplied footprint Updated Figure 10 pictures
1.30	7/29/2021	Updated syntax for $CS, $MM, $RD, $TD to reflect firmware version v1.1.0+ Updated information on sleep drift and GPIO timing
1.35		Removed ½ wave antenna information Added information about mPCIE adapter board Updated checksums for $GN, $MT, $PW, $RT and $TD commands Updated Table numbers

Additional Resources:

Please visit our developer tools webpage for quickstart guides and other helpful resources: **https://swarm.space/developertools**

1 Safety Information and Compliance

The Tile is designed to comply with the standards for Radio Emissions Compliance and Electromagnetic Compatibility in the United States, Canada, Australia, New Zealand, United Kingdom, European Union, Brazil, as well as worldwide.

1.1 FCC Compliance

Swarm provides a certification guide on the developer tools webpage of https://swarm.space/ to help with the process: https://swarm.space/swarm-tile-certification-guide/

1.1.1 FCC Interference Statement (Part 15.105 (b))

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

This device complies with part 15 of the FCC Rules. Operation is subject to the condition that this device does not cause harmful interference.

1.1.2 FCC Part 15 Clause 15.21:

Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

1.1.3 FCC Part 15.19(a):

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

1.1.4 FCC ID:

The FCC ID for the Tile is **2AVE9-TILE01**. All manufacturers integrating the Tile into their products are advised to provide a physical or e-label stating “Contains FCC ID: **2AVE9-TILE01**”.

1.1.5 Part 15 Subpart B Disclaimer:

The final host product requires Part 15B compliance testing with the modular transmitter installed.

1.2 ISED Compliance

1.2.1 ISED RSS-Gen Notice CAN ICES-3 (B)NMB-3(B):

This device complies with Industry Canada’s licence-exempt RSSs. Operation is subject to the following two conditions:
(1) This device may not cause interference; and
(2) This device must accept any interference, including interference that may cause undesired operation of the device.

Le présent appareil est conforme aux CNR d’Industrie Canada applicables aux appareils radio exempts de licence. L’exploitation est autorisée aux deux conditions suivantes :
1) l’appareil ne doit pas produire de brouillage;
2) l’appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d’en compromettre le fonctionnement.

1.2.2 IC ID:

The IC ID for the Tile is **25817-TILE01**. All manufacturers integrating the Tile into their products are advised to provide a physical or e-label stating “Contains IC ID: **25817-TILE01**”

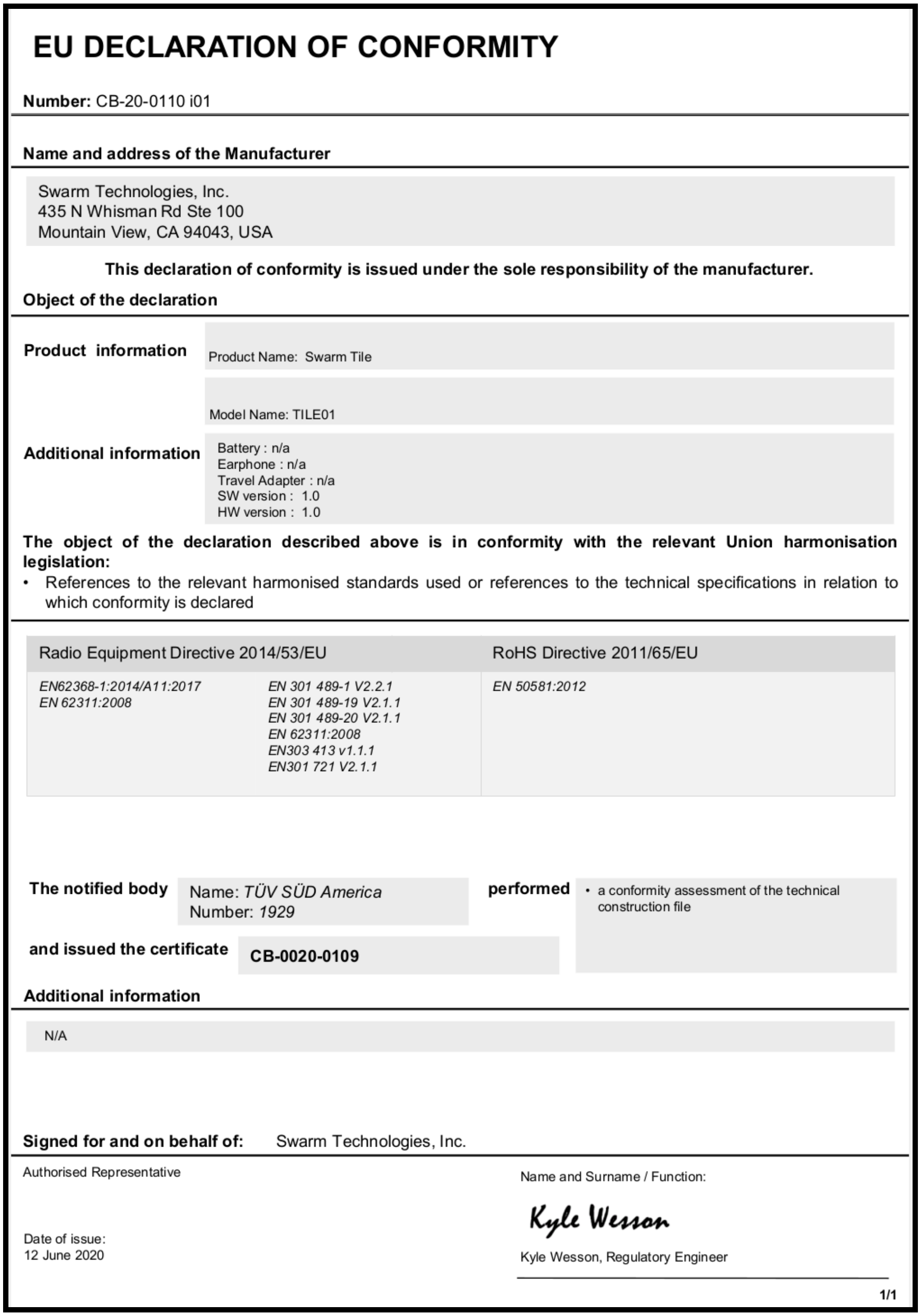
1.3 RF Exposure Guidance

In order to comply with FCC / ISED RF Exposure requirements, this device and antenna must be installed to provide at least 29 cm separation from the human body at all times.

Afin de se conformer aux exigences d'exposition RF FCC / ISED, cet appareil doit être installé pour fournir au moins 29 cm de séparation du corps humain en tout temps.

1.4 EU RED Certification

1.4.1 Declaration of Conformity:



EU DECLARATION OF CONFORMITY

Number: CB-20-0110 i01

Name and address of the Manufacturer

Swarm Technologies, Inc.
435 N Whisman Rd Ste 100
Mountain View, CA 94043, USA

This declaration of conformity is issued under the sole responsibility of the manufacturer.

Object of the declaration

Product information	Product Name: Swarm Tile
	Model Name: TILE01
Additional information	Battery : n/a Earphone : n/a Travel Adapter : n/a SW version : 1.0 HW version : 1.0

The object of the declaration described above is in conformity with the relevant Union harmonisation legislation:

- References to the relevant harmonised standards used or references to the technical specifications in relation to which conformity is declared

Radio Equipment Directive 2014/53/EU		RoHS Directive 2011/65/EU
EN62368-1:2014/A11:2017 *EN 62311:2008*	*EN 301 489-1 V2.2.1* *EN 301 489-19 V2.1.1* *EN 301 489-20 V2.1.1* *EN 62311:2008* *EN303 413 v1.1.1* *EN301 721 V2.1.1*	*EN 50581:2012*

The notified body	Name: *TÜV SÜD America* Number: *1929*	**performed**	• a conformity assessment of the technical construction file
and issued the certificate	**CB-0020-0109**		

Additional information

N/A

Signed for and on behalf of: Swarm Technologies, Inc.

Authorised Representative

Name and Surname / Function:

Kyle Wesson

Kyle Wesson, Regulatory Engineer

Date of issue:
12 June 2020

1/1

1.4.2 Article 10(2)/(10):

This product has been so constructed that the product complies with the requirement of Article 10(2) as it can be operated in at least one Member State as examined and the product is compliant with Article 10(10) as it has no restrictions on putting into service in all EU member states.

1.5 Transceiver Regulatory Certification

The Tile is a regulatory approved modular transmitter that is designed to be integrated into an enclosed host system. With appropriate external connections, the host can be designed to meet full regulatory tests and sold as a regulatory certified product that meets FCC, IC, and CE requirements. **Table 1** below is a partial list of regulatory approvals.

Regulatory Approvals	Radio Tests	EMC Tests	Safety Tests
FCC	FCC CFR47 Parts 2, 15, and 25		
IC	Industry Canada RSS170 Issue 2 - March 2011		
CE	ETSI EN 301 721 V2.1.1 (2016-05) ETSI EG203 367 V1.1.1	CISPR 16-23:2010/A1:2010 EN 55032:2012 EN 6100-4-2/EN55024:2010 EN 6100-4-3/EN55024:2010 EN 6100-4-8/EN55024:2010	EN 62368-1:2014/A11:2017

Table 1: Overview of the Swarm Tile regulatory approvals.

2 Product Overview

The Swarm Tile (Model: TILE01) satellite data modem transmits and receives data to and from Swarm’s space network and is designed to be embedded into a third-party product. Swarm backend systems support the delivery of customer data via a REST API to the cloud service of each user’s choice.

The Tile is a module suitable for a variety of low-bandwidth use cases: from connecting people and tracking vehicles, ships, or packages to relaying sensor data for agriculture, energy, and industrial IoT applications.

The Tile is a Surface Mount Technology (SMT) module that can be easily integrated into any new or existing PCB design. The Tile communicates via a standard 3.3V CMOS serial UART interface or a PC interface with a USB-to-serial converter.

<table>
<tr><th>Category</th><th>Description</th></tr>
<tr><td>Satellite data</td><td>Transmit/receive messages in less than 5 minutes (Q1-2022)</td></tr>
<tr><td>Components</td><td>GPS, VHF radio with integrated T/R switch, U.FL connector for GPS and VHF antenna (SMD interface also available), indicator LEDs, 3.3V serial interface, 3.3V GPIO</td></tr>
<tr><td>Sensors</td><td>Onboard GPS (lat/lon/alt), 1 pulse per second signal, Temperature sensor</td></tr>
<tr><td>Dimensions and Mass</td><td>58.7 mm x 27.4 mm x 6.0 mm, 14g
See detailed description in the Mechanical Specification section.</td></tr>
<tr><td>Power</td><td>
<table>
<tr><th>Mode</th><th>Average</th><th>Peak</th></tr>
<tr><td>Sleep, 3.3V</td><td>N/A</td><td>22 μA</td></tr>
<tr><td>Receiver Active, 3.3V</td><td>30 mA</td><td>35 mA</td></tr>
<tr><td>Transmitter on, 3.3V</td><td>888 mA</td><td>1000 mA</td></tr>
</table>
</td></tr>
<tr><td>Protocol</td><td>Modified NMEA two-letter command set</td></tr>
<tr><td>Bit rate</td><td>1 kbps. Maximum packet size is 192 bytes</td></tr>
</table>

Table 2: Overview of the Swarm Tile.

3 Mechanical Specification

3.1 Tile Dimensions

The overall dimensions of the Tile and its weight are summarized in **Table 3**.

Parameter	Value
Length	58.67±0.3 mm
Width	27.43±0.3 mm
Height	6.00±0.1 mm
Weight	14 g

Table 3: Tile Mechanical Dimensions and Weight.

A diagram of the Tile's mechanical dimensions is shown in **Figure 1**.

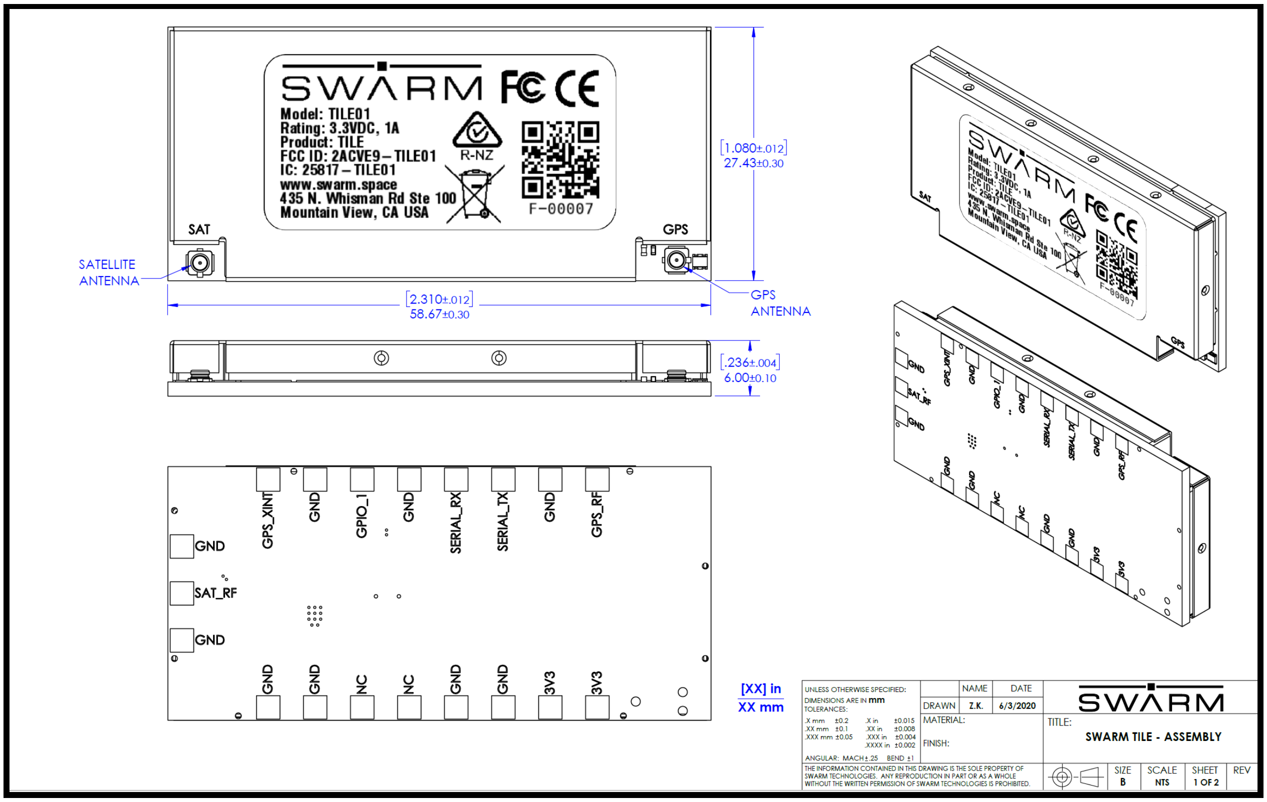


Figure 1: Tile front and back views.

3.2 Environmental

The environmental specifications of the Tile are summarized in **Table 4** below. The Tile is not conformally coated, and as such the user needs to provide any weatherproofing for their application.

Parameter	Value
Operating Temperature Range	-55 °C to +85 °C
Storage Temperature Range	-220 °C +160 °C
Operating Humidity Range	0% to 95%, non-condensable
Storage Humidity Range	0% to 95%, non-condensable

Table 4: Environmental Specifications.

3.3 Mechanical Specification – Placement

The Tile must be fitted within an enclosed host system. The Tile is designed to be soldered directly to a PCB, and is provided with solder pads on its underside.

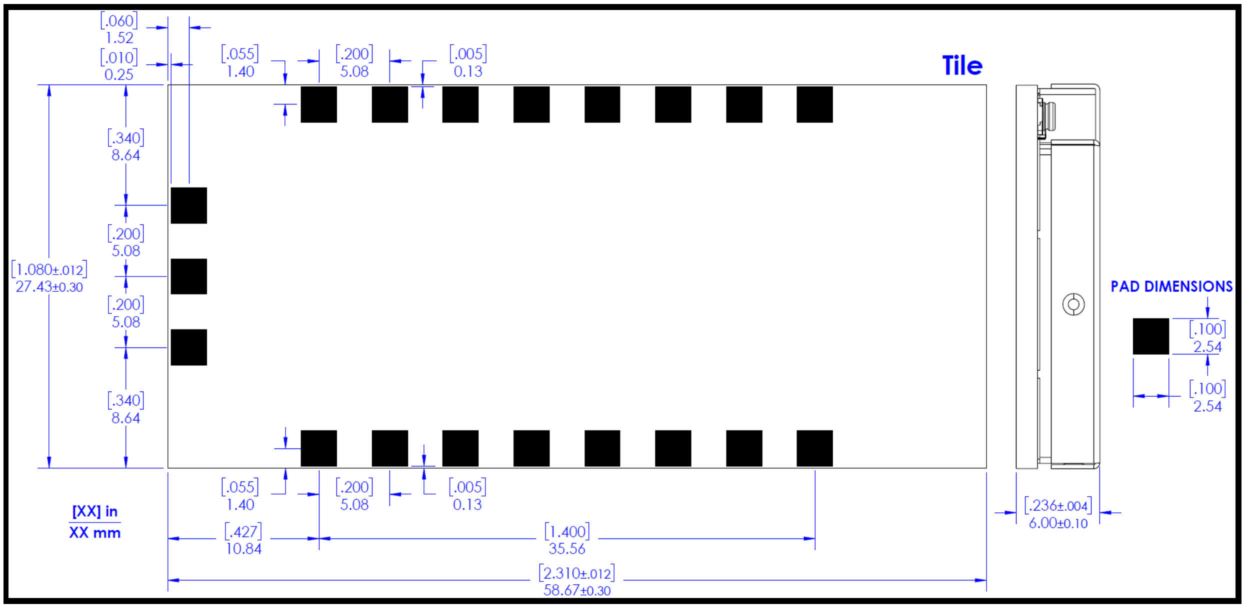


Figure 2: Location of solder pads on the underside of the Tile. Bottom view.

3.4 Moisture Sensitivity Level

The Tile is classified as an MSL 3 device as defined by IPC/JEDEC J-STD-020E standards. Commercial Tile orders will be shipped in a moisture barrier antistatic bag with a desiccant and a moisture indicator card, along with a handling reference label.

Tiles that exceed the MSL3 exposure time of 168 hours are recommended to be baked for 9 hours at 125° C prior to reflow/assembly. Tiles that are used in an Eval Kit or for development purposes do not need to be baked prior to use.

Moisture Sensitivity Level	Floor Life	
	Time	Condition
3	168 Hours	≤30ºC/ 60%RH

Table 5: Tile MSL Specifications per IPC/JEDEC J-STD-020E

3.5 Reflow Soldering

The Tile is designed to be reflow soldered onto a PCB. A recommended reflow profile can be found below in **Figure 3** and **Table 6.**

Use of "No Clean" soldering paste is recommended. An example “No Clean” paste can be found below:

Soldering Paste: M8 SAC305 / PN#89268 (AIM Solder)
Alloy specification: Sn 95.5/ Ag 4/ Cu 0.5 (96.5% Tin/ 3% Silver/ 0.5% Copper)
Melting Temperature: 217° C
Stencil Thickness: 100 to 150 μm for base boards

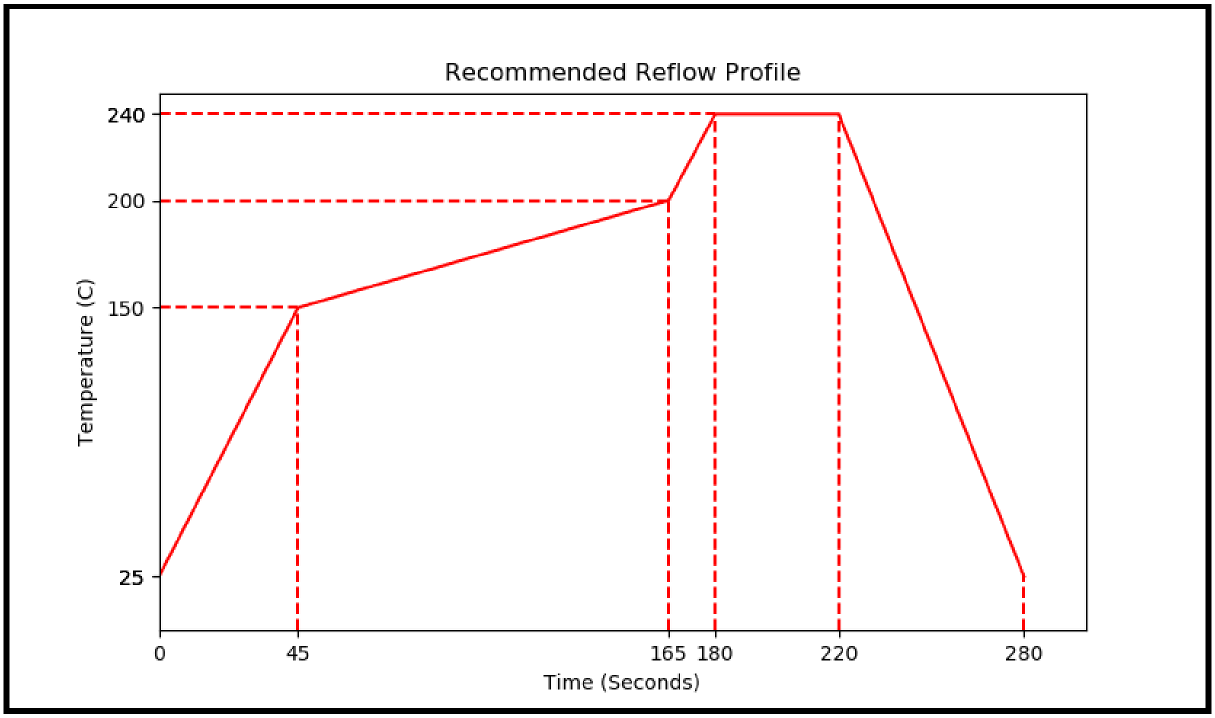


Figure 3: Recommended reflow profile for lead-free solder paste.

Time (Seconds)	Temperature (℃)
0	25
45	150
165	200
180	240
220	240
280	25

Table 6: Data points for reflow soldering.

3.5 Physical Interface Connectors

The Tile incorporates two connectors:

• A GPS Antenna U.FL male connector [TE Connectivity #1909763-1]

• An RF Antenna U.FL male connector [TE Connectivity #1909763-1]

4 Electrical Interfaces

The following subsections contain information for the electrical interfaces of the Tile.

4.1 User Host Device

The user host device provides the following connections to the Tile:

- DC power supply input: 3.3V, 1000 mA peak
  - The DC power supply is ***required*** to be contained within an EMI shielding can
  - Additional filtering of the 3.3V supply is highly recommended
- 3.3 volt serial data interface
- Satellite signal (Use either the provided U.FL connector labeled SAT **or** Pin 10. See section 5.3 - RF Trace Layout Design for more information)
- GPS signal (Use the provided U.FL connector labeled GPS)
- GPIO_1 (optional)

4.2 Tile Pin Allocation

The pin numbering scheme of the Tile is shown in **Figure 4**. The module pins allow soldering onto a printed circuit board (PCB) using standard reflow soldering techniques. The pin function assignment is given in **Tables 7** and **8**. Multiple supply grounds are provided and all power pins / supply grounds are required to be connected to the power supply in order to limit the current on any one pin. Multiple signal grounds are provided to reduce cross-talk. Pins 5 and 6 are intentionally left empty, and must be left unconnected.

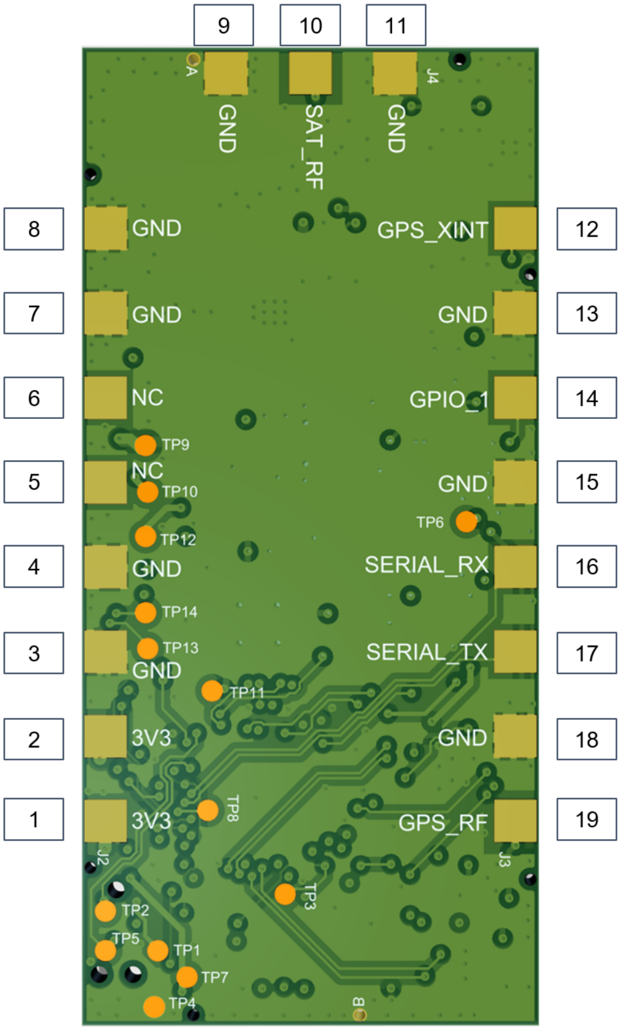


Figure 4: Tile pinout and pin numbers. ***Bottom View***.

Pin Number	Name	Type	Description
1	3V3	VCC	3.3V at up to 1000mA
2	3V3	VCC	3.3V at up to 1000mA
3	GND	Ground	Ground
4	GND	Ground	Ground
5	NC	Not Connected	RESERVED - DO NOT CONNECT
6	NC	Not Connected	RESERVED - DO NOT CONNECT
7	GND	Ground	Ground
8	GND	Ground	Ground
9	GND	Ground	Ground
10	SAT_RF	RF	External satellite radio antenna connection
11	GND	Ground	Ground
12	GPS_XINT	Not Connected	RESERVED - DO NOT CONNECT
13	GND	Ground	Ground
14	GPIO_1	Input/Output	General purpose input/output
15	GND	Ground	Ground
16	SERIAL_RX	Input	3.3V serial receive
17	SERIAL_TX	Output	3.3V serial transmit
18	GND	Ground	Ground
19	GPS_RF	Not Connected	RESERVED - DO NOT CONNECT

Table 7: Tile pin numbers and descriptions.

Additional Notes

Pin Number	Note
14	Connection is unbuffered directly to a GPIO on the Tile processor. Configuration will be provided via Tile firmware. GPIO pin is 3.3V tolerant and open drain, with a sink current limit of 8 mA (20 mA with a relaxed VOL/VOH).
19	GPS_RF uses 50 Ohm input impedance. Pin 19 by default is not connected - by default the GPS U.FL connector is selected.
1, 2	The 3.3V connection points are in parallel with one another and power the Tile. If the designer wants to enable a complete power off mode, a load switch can be provided here.
10	SAT_RF uses 50 Ohm input impedance. Use of SAT_RF instead of the SAT U.FL connector requires the user to use the FCC certified layout as defined in section 5.3 - RF Trace Layout Design

Table 8: Additional notes on pin numbers.

4.3 DC Power Interface

The DC power interface consists of the DC power inputs as summarized in **Table 9**. The power requirements apply to DC power measured at the Tile user connector input and not at the output of the power supply. It is required that users incorporate the required bypass capacitors for the supplied power input as can be seen in **Figure 7.**

Name	Description	Min	Typ	Max	Unit
VCC	Module supply voltage	-	3.30±3%	-	V
VCC Ripple	Module supply voltage ripple	-	-	75	mVpp
VCC Limits	Module supply voltage **absolute** limits	3.15	-	3.60	V
ICC (3.15V)	Current consumption - Sleep Mode	-	-	22.0	µA
	Current consumption - Receiver Active	26.9	27.6	30.4	mA
	Current consumption - GPS Acquisition Mode*	44.7	49.1	55.2	mA
	Current consumption - Transmitter on	904	910	1000	mA
ICC (3.30V)	Current consumption - Sleep Mode	-	-	22.0	µA
	Current consumption - Receiver Active	27.2	27.9	30.5	mA
	Current consumption - GPS Acquisition Mode*	46.0	49.3	54.8	mA
	Current consumption - Transmitter on	880	888	1000	mA
ICC (3.60V)	Current consumption - Sleep Mode	-	-	22.0	µA
	Current consumption - Receiver Active	27.9	29.1	35.0	mA
	Current consumption - GPS Acquisition Mode*	47.6	49.6	55.2	mA
	Current consumption - Transmitter on	845	853	1000	mA

Table 9: Power supply characteristics for 3.15, 3.30, and 3.60V supplies to the Tile.

*Includes satellite receiver active current with GPS in acquisition mode. The Tile enters into GPS acquisition mode for approximately 30 seconds after exiting from sleep mode, on powerup, or when the Tile needs to re-acquire a GPS fix (approximately once every 3 hours) while the Tile is continuously powered on and not in sleep mode.

An example power profile for a Tile powered with 3.3V can be found below in **Figure 5.**

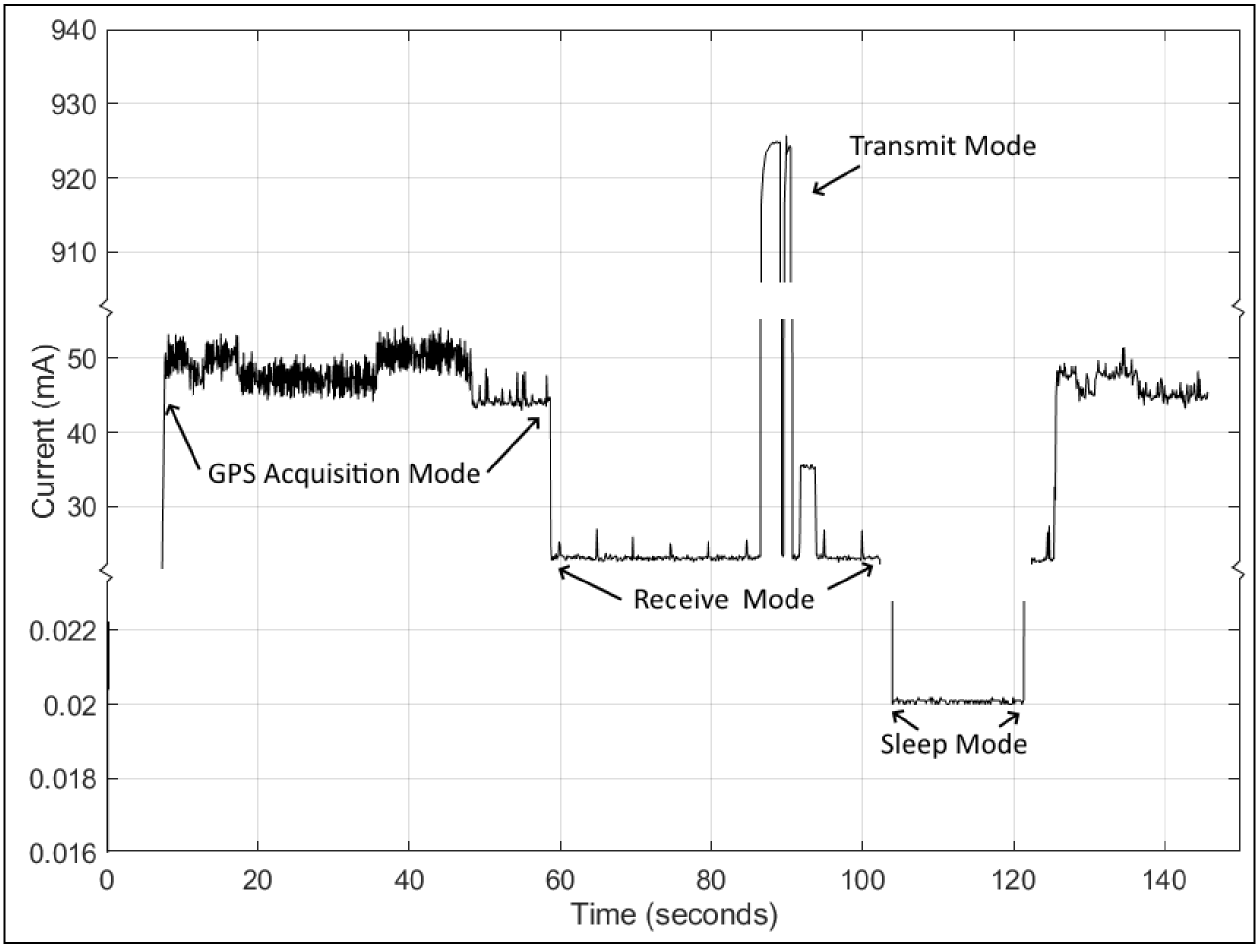


Figure 5: Example current use for a Tile with a 3.3 V input from wake-up, GPS acquisition, Transmit, Receive mode, and then sleep mode. Note the two breaks in the y-axis scale.

Transmissions from the Tile consist of short periods of high current draw conforming to the following characteristics:

Parameter	Value	Note
Transmission Length	3.7 s	Length of transmission for sending a 192 byte packet to space. Actual length may be shorter with a smaller packet size
Transmission Energy	3.1 mW-hrs (11.2 J)	Total amount of energy required for 1 transmission to space (192 bytes)
Recovery Time	0.4s	When sending multiple messages, this is the minimum amount of time that will pass before the Tile will attempt another transmission

Table 10: Sample Tile transmission characteristics for sending a user payload of 192 bytes. Tile provided with 3.3V input.

Other electrical characteristics of the Tile can be found below in **Table 11**:

Parameter	Value
Maximum power at antenna connector	3300 mW
Sum total of all capacitance on Tile	110 µF
Sum total of all inductance on Tile	1983 nH
Largest capacitor on Tile	47 µF
Largest inductor on Tile	1000 nH

Table 11: Other electrical characteristics for the Tile

An example of the interface between a Tile and a third-party host device is shown in **Figure 6** and **Figure 7.** A photo of a Tile integrated onto a third party device can be found in **Figure 8.**

A Serial Debug Header is included in the sample hardware design to upgrade the Tile firmware. It is **critical** that users include this header in their design to provide serial access to the Tile, which can be used to upgrade to future firmware versions and for debugging.

The Serial Debug Header requires the following connections: SERIAL_RX, SERIAL_TX, and GND. It is **highly recommended** that the designer use a 6-pin connector to easily accommodate a COTS TTL (USB-A to UART) cable or Tag-Connect, for example: https://www.ftdichip.com/Products/Cables/USBTTLSerial.htm

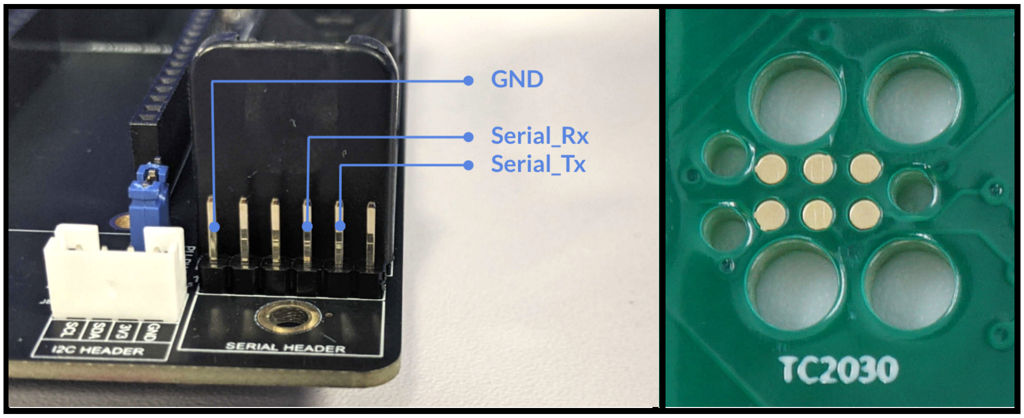


Figure 6: Example of a Serial Debug Header, shown on the Swarm Eval Board, and 6-pin tag-connect pads

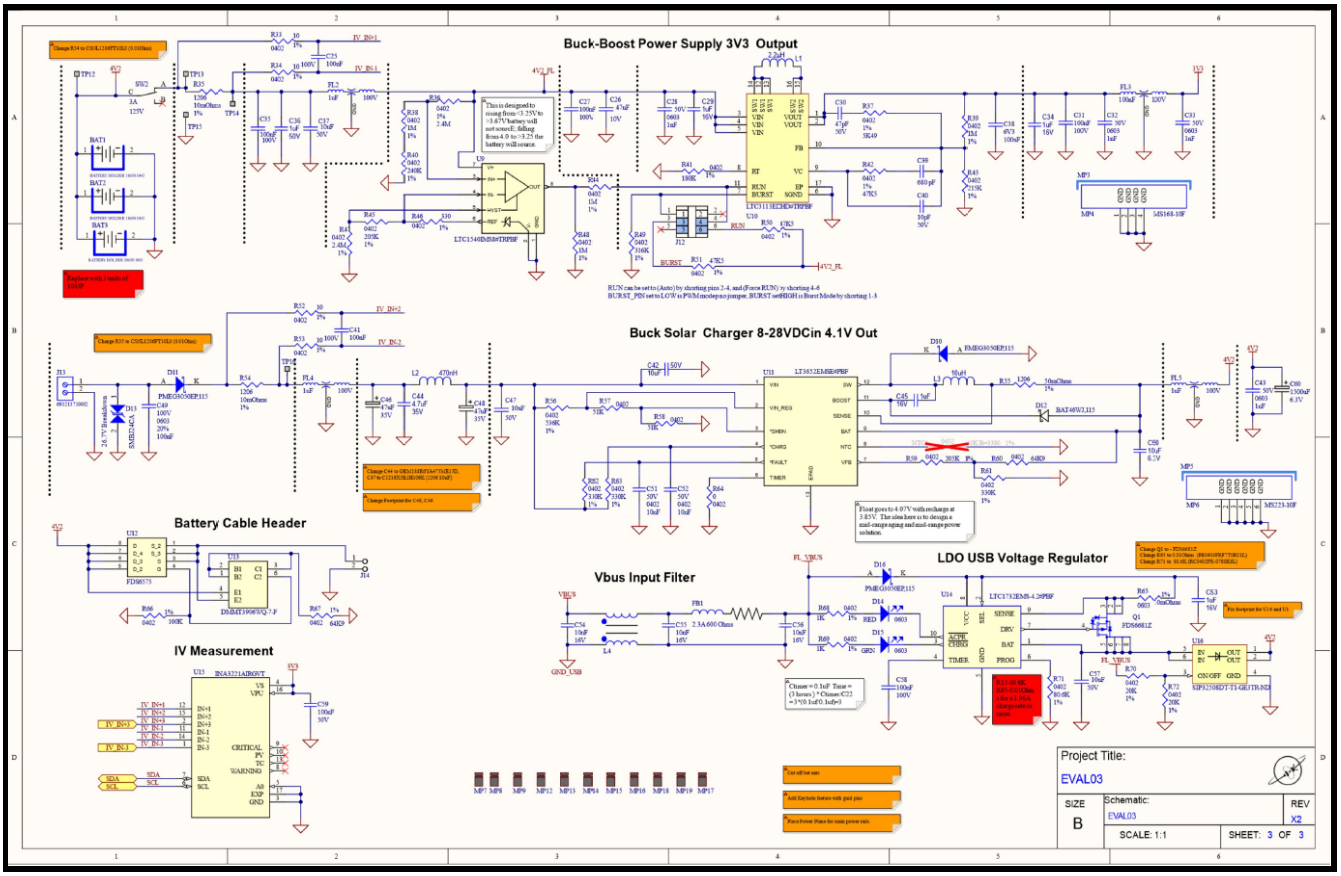

Buck-Boost Power Supply 3V3 Output
LTC3115EDHD#TRPBF
LTC1540IMS8#TRPBF
This is designed to rising from <3.25V to >3.67V battery will not source, falling from 4.0 to >3.25 the battery will source.
RUN can be set to (Auto) by shorting pins 2-4, and (Force RUN) by shorting 4-6
BURST_PIN set to LOW is PWM mode no jumper, BURST set HIGH is Burst Mode by shorting 1-3
Buck Solar Charger 8-28VDCin 4.1V Out
LT3652EMSE#PBF
Float goes to 4.07V with recharge at 3.85V. The idea here is to design a mid-range aging and mid-range power solution.
Battery Cable Header
DMMT3906W-Q-7-F
Vbus Input Filter
LDO USB Voltage Regulator
LTC1732EMS-4.2#PBF
FDS6681Z
SIP32508DT-T1-GE3
IV Measurement
INA3221AIRGVT
Project Title:
EVAL03
SIZE
B
Schematic:
EVAL03
REV
X2
SCALE: 1:1
SHEET: 3 OF 3

Figure 7: Schematic of a buck-boost-solar charger design for a Swarm Tile integration. Full schematic for the Swarm Eval kit available at https://swarm.space/developertools/

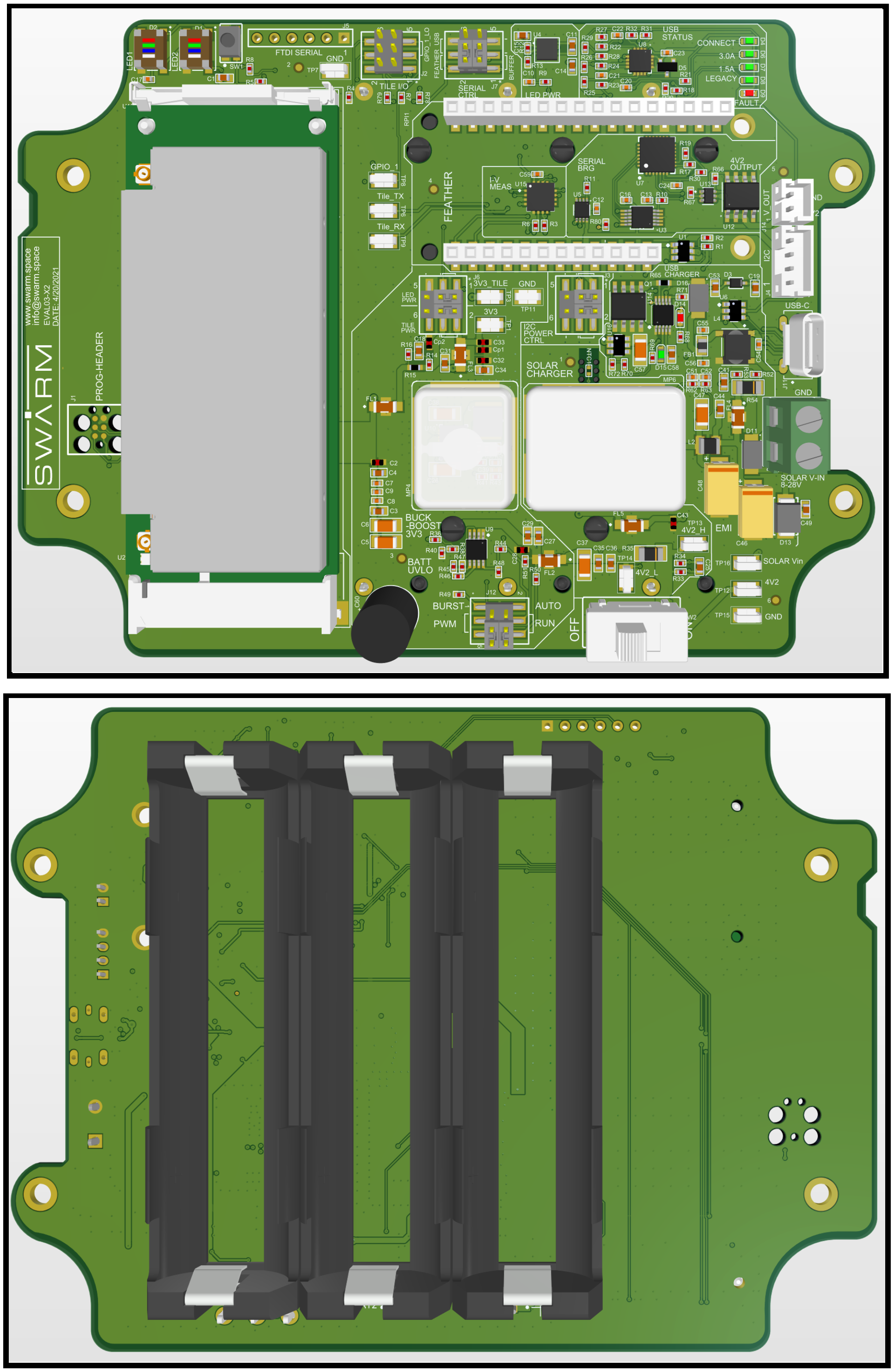


Figure 8: Sample hardware reference design integration for the Swarm Tile. This is a low noise buck-boost-solar charger example with the Tile on mPCIe daughter card (reference design files can be found at www.swarm.space/developertools/). A 6-pin serial programming header (upper-left of image) is highly encouraged so that the Tile firmware can be easily updated in the future. Qty 3 18650 lithium rechargeable batteries in parallel are used in this reference design. The circuit fits into BUD Industries IP68 waterproof case Model # PU-16533-C, with the use of 4 standoffs.

4.4 Power On/Off Control

The Tile can be powered down with the use of the **$PO*1f** command. Once powered down, it can be externally switched on/off by a user-supplied load switch on the 3V3 power rail.
After power to the Tile has been removed, power should not be restored until at least 1 second has elapsed.
If a Tile does not respond to software commands, remove power from the module, wait for at least 1 second, and then power it back on. If the 1 second minimum wait time is not adhered to, the reset circuit may not operate and the Tile could be placed in a non-operational state. The state is not permanent and can be rectified by the above procedure.

4.5 Serial Data Interface

The serial data interface is a CMOS serial UART 3-wire (serial Rx, serial Tx, and ground) interface at 3.3V digital signal levels over which the Tile transfers commands, responses, and message data. The serial communication parameters can be found in **Table 12**.

Parameter	Value
Baud Rate	115200
Data Bits	8 Bits
Parity	None
Stop bits	1 Bit
Flow Control	None

Table 12: Serial communication parameters.

In addition, the electrical characteristics for SERIAL_RX, SERIAL_TX, and GPIO1 can be found in **Table 13.**

Symbol	Parameter	Min	Typ	Max	Unit
VIL	I/O input low level voltage	-0.3	-	0.3*VCC	V
VIH	I/O input high level voltage	0.7*VCC	-	VCC	V
RPU/RPD	Weak pull up/down equivalent resistor (for GPIO1)	25	40	55	kΩ

Table 13: Electrical characteristics for SERIAL_RX, SERIAL_TX, and GPIO1.

All customer messages, data, and settings are stored in non-volatile memory. As such, after a power cycle customer settings as well as any messages that have not yet been transmitted over the Swarm network will be retained on the Tile. The read/write lifetime of the non-volatile memory exceeds 20 years.

4.6 LED Indicators

The Tile contains two indicator LEDs, whose function is described below in **Table 14**. All LEDs are off when the Tile is placed into sleep mode.

LED	Function
Green	During the bootup sequence the green LED will be on solid for 3 seconds. During normal operation after bootup and before shutdown, the green LED will blink 100ms every 5 seconds while the Tile is powered on. This is a “heartbeat” indication that the Tile is working as expected.
Red	After bootup and while the Tile is acquiring a GPS fix, the red LED will flash quickly until a valid GPS fix has been found. Then the red LED will shut off during normal operation. After waking up from sleep, and before a fresh GPS fix has been acquired, there will be a single red LED flash every 5 seconds (following the green LED flash) until a fresh GPS fix has been found. Then the red LED will again shut off during normal operation. During the shutdown sequence, the red LED will be on solid until 3.3V power is removed from the board.

Table 14: LED functions.

5 RF Interface

This section describes the physical characteristics of the RF connectors and specifications of the RF Interface.

5.1 RF Connectors

The Tile's satellite and GPS connectors are both male U.FL connectors [TE Connectivity Part Number 1909763-1]. This is a surface mount connector that is directly attached to the Tile. A Swarm VHF antenna must be used to ensure that the antenna is tuned appropriately to make successful connection to the Swarm satellites. Swarm antennas are tuned for a VSWR of 3.5 or lower at both 137.000 MHz and 150.000 MHz.

5.2 RF Antennas

For illustrative purposes, a picture of our ¼ wave antenna is shown below in **Figure 9.**

Figure 9: Swarm Coiled ¼ Wave Antenna

5.2.1 Antenna Characteristics

The Tile is certified using the antenna described in **Table 15**. No power reduction compensation is required for use with this antenna.

Swarm Coiled ¼ Wave Antenna	
Parameter	**Value**
Length	22 cm
Diameter (Connector)	11.2 mm
Diameter (along major length)	7.6 mm
Weight	31.5 g
Operating Temperature	-55 °C to +130 °C
Operating Humidity	0-100% humidity, condensable
Impedance	50 Ohms nominal
Polarization	Linearly Polarized
VSWR (between Swarm Bands)	Δ < 0.5
Gain	2.0 dBi
Frequency	137.000-138.000 MHz (Rx) 148.000-150.000 MHz (Tx)
Connector	SMA male (U.FL to SMA female cable required)
Antenna Ground Plane	**Required**, see section 5.2.2
Antenna Height Requirement	1m above the ground/solid surfaces
Antenna Classification	Mobile, Fixed
Minimum separation distance from body	29cm

Table 15: Antenna characteristics for Swarm Coiled ¼ Wave Antenna.

Antenna notes continued on the next page…

A GPS antenna is ***required* for operation**. Any passive GPS antenna with an appropriate connection to the Tile's U.FL male connector or via the GPS_OUT pin is acceptable for use. One such example of a passive antenna is: [Molex Part Number 1461860300].

Active GPS antennas are supported at 3.3V and up to 35mA. Any active antennas that require a different voltage or maximum current must be powered by the user's host device.

5.2.2 Ground Plane Requirements

An antenna counterpoise or antenna ground plane is required for the Swarm Coiled ¼ Wave Antenna. Ground planes are electrically conductive surfaces that are connected to the ground conductor of the antenna that serve as a reflecting surface for radio waves.

To date Swarm has reliably communicated with its satellites using a coiled ¼-wave antenna with a 50 cm long, 20 AWG (0.812mm diameter) wire descending from the ground of the SMA connector of the antenna mount. Swarm has also reliably communicated with its satellites using a solar panel off to one side as the ground plane of sufficient size: 30cm x 30cm x 0.3cm

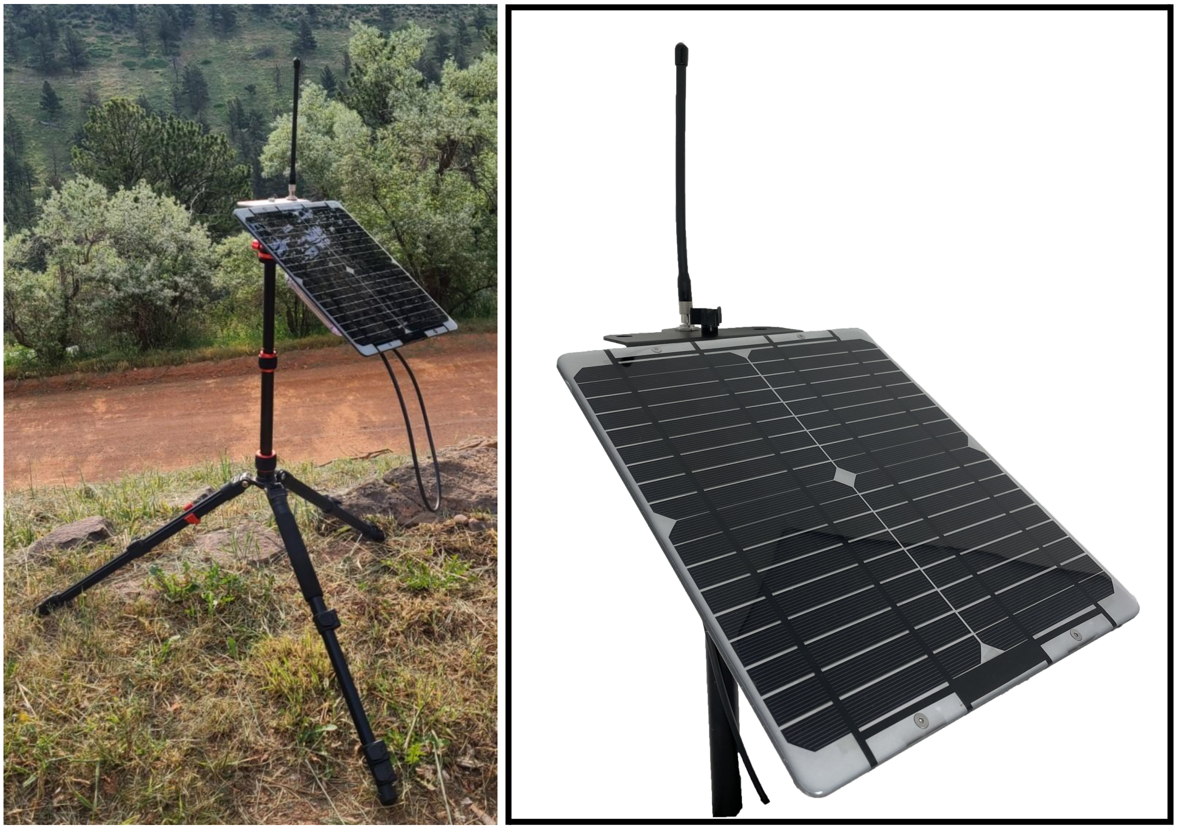


Figure 10: Example ¼-wave antenna with a (Swarm) 9W solar panel serving as the required 30 x 30 cm ground plane off to one side. Tested extensively in the field and works well.

5.3 RF Trace Layout Design

The Tile is certified with a PCB edge SMA connectors [Samtec Part Number SMA-J-P-H-ST-EM1] for the RF and GPS outputs, with a micro-strip trace layout (along with copper keep-out areas) as shown in **Figure 11**.

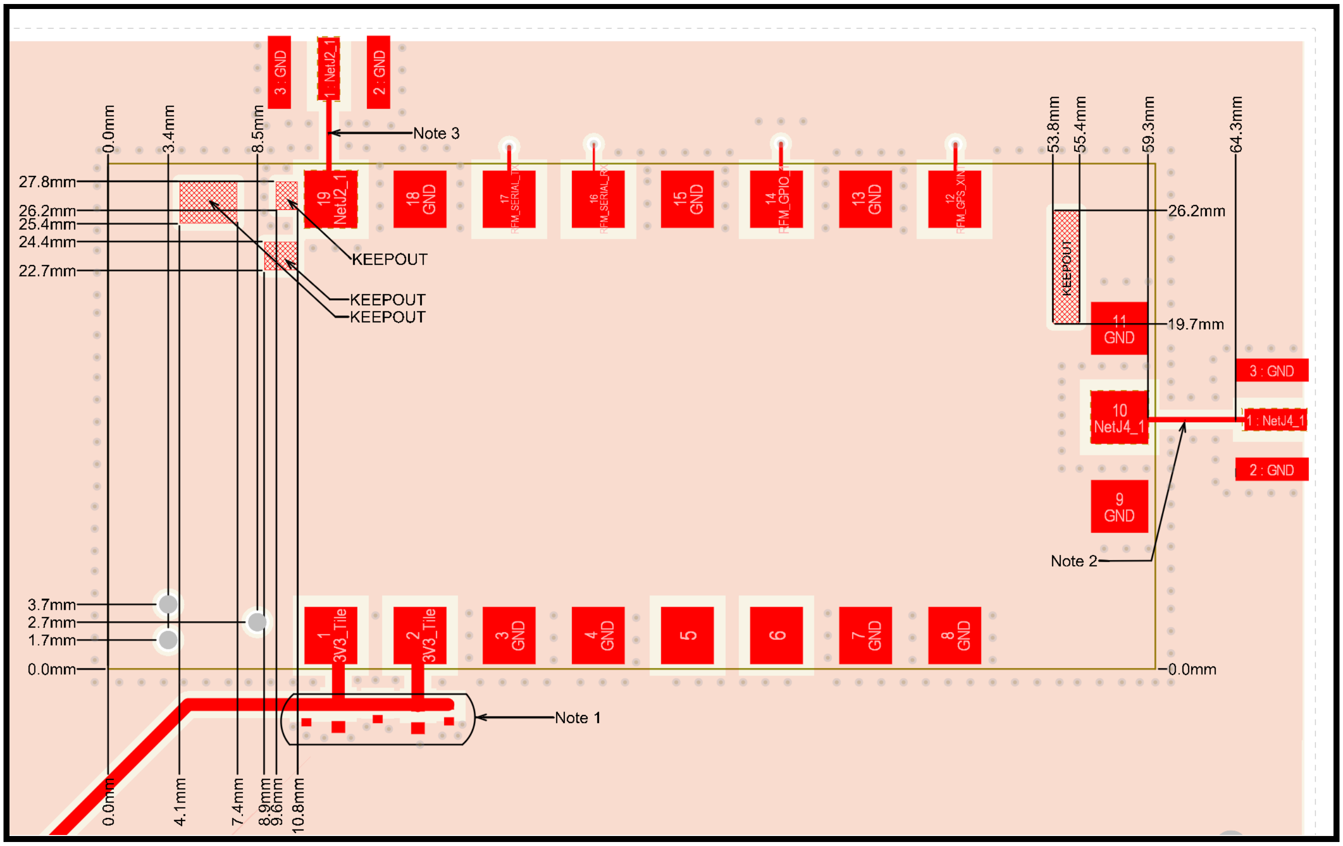


Figure 11: RF trace routing and keep out area for RF output (top layer).

Note 1: The location for the module input and bypass capacitance.
Note 2: SAT_RF track should utilize a 50 Ohm micro-strip specific to the customer board layout, with a ground plane below. A 25 mil copper-to-track clearance for the microstrip is recommended.
Note 3: GPS_RF track should utlize a 50 Ohm micro-strip specific to the customer board layout, with a ground plane below. A 25 mil copper-to-track clearance for the microstrip is recommended.
Note 4: Mounting holes - 1.067mm diameter. Not required in customer design
Note 5: Use a 25 mil copper-to-pad clearance around all the pads that are not ground pads, and a 25 mil copper-to-track clearance around the SMA trace on the board.

5.4 Integration Guidance

The following recommendations are provided for the host device:

- Place the Tile close to the edge of the host PCB, preferably on a corner
- Use a solid GND plane on an inner layer (for best EMC and RF performance).
- Place GND vias close to module GND pads as possible. An example layout can be found in **Figure 11.**
- Route traces to avoid noise being picked up on VCC supply.
- RF Trace keep out areas:
  - Ensure there is no copper in the keep-out areas on any layer of the host PCB as can be seen in **Figure 11**.
  - Locate all mounting hardware and metal as far as possible from the Tile to not impede antenna performance.
  - Host board should not contact test points (TP1, TP2, etc…) if the designer chooses to use vias or traces under the Tile
- Ensure that your PCB includes access to the Serial_Rx and Serial_Tx lines of the Tile via a serial header/connector to enable direct access for firmware updates, and for debugging purposes.
  - See the **Electrical Interfaces** section for more details.

5.5 Antenna Debugging

To quickly verify antenna connection to Tile:

1. Turn on the Tile
2. After bootup, issue **$RT 1*17**. The noise floor the Tile hears (in dBm) will begin displaying at a rate of once per second
3. While the Tile's LED is blinking red rapidly (indicates it has not yet acquired a GPS fix)
   a. Touch the Tile's antenna with your hand or with another object. The noise floor measurement should noticeably change
   b. The Tile will not transmit until it has a GPS fix **and** it hears a Swarm Satellite. There is no risk of the Tile transmitting during this procedure if the red LED is blinking rapidly

6 Software interface

Message types

There are two types of messages:

- **Unsolicited messages** - which include status messages, date/time and GPS information, and notifications that user data has been received. By default all of these are set to not appear, and user settings of these messages persist through restarts and power cycles
- **Command responses** - which include responses to input commands, as well as notifications that user data has been sent or settings have been updated

General command structure

All messages sent and received are NMEA formatted sentences. NMEA sentences are terminated with a single newline character. Each sentence begins with a `$` and ends with `*xx` where `xx` is a two digit hexadecimal checksum of the characters between the `$` and `*xx`. The checksum conforms to the NMEA standard and does not include either the `$` or the `*xx`. Messages with a bad checksum are silently ignored and are not retained. A sample implementation of the NMEA checksum can be found below (written in C)

A `$` will never occur within a command, and may be used to reset the receiving state machine.

An `*` may occur within a command. The receiving state machine will verify the last three characters in the command are `*xx` after the `\n` is received and before the checksum is calculated. Each `x` may be any legal ASCII character in the range `0..9`, `A..F`, or `a..f`.

An example command is provided below to illustrate the command structure and a valid checksum. This command returns the most recent date/time message :

```
$DT @*70
```

```
$DT 20190408195123,V*41
```

Command timing

Once the **$** is received, all subsequent characters must occur within 250 milliseconds of the previous character. If the inter-character delay exceeds 250 ms, the command will be silently discarded, and the receiving state machine will consume and ignore any characters received until the next **$**.

Command responses

An **OK** response confirms that the input parameters have been updated in response to a command. An **ERR** response indicates that additional or invalid characters were included between the two-character command designator and the * of the command.

Boot up sequence

The user application should ignore any characters received during startup until the following NMEA sentence has been received:

```
$TILE BOOT,RUNNING*49
```

The bootloader may output non-NMEA formatted messages during this time. These messages include, but are not limited to: status messages, firmware update progress messages, and error messages. These messages should be ignored and are for Swarm debugging purposes only.

Messages

All messages sent to or from the Hive have an **Application ID** tag associated with them. The **Application ID** can be an arbitrary number from 0 to 64999 (Swarm reserves use of 65000 - 65535), and may be used by the user to organize messages (ex: **Application ID** 1000 could be used for device telemetry, 2000 for commands to the device and 3000 for emergencies).

Implementation of NMEA checksum in C

```
uint8_t nmeaChecksum (const char *sz, size_t len)
{
 size_t i = 0;
 uint8_t cs;

 if (sz [0] == '$')
   i++;

 for (cs = 0; (i < len) && sz [i]; i++)
   cs ^= ((uint8_t) sz [i]);

 return cs;
}
```

Table of Commands

Command	Description
$CS	Configuration Settings
$DT	Date/Time Status
$FV	Retrieve Firmware Version
$GJ	GPS Jamming/Spoofing Indication
$GN	Geospatial Information
$GP	GPIO1 Control
$GS	GPS Fix Quality
$MM	Messages Received Management
$MT	Messages to Transmit Management
$PO	Power Off
$PW	Power Status
$RD	Receive Data Message
$RS	Restart Device
$RT	Receive Test
$SL	Sleep Mode
$TILE	Tile Status
$TD	Transmit Data

Table 16: Table of all Tile commands

$CS - Configuration Settings

Retrieve and display the configuration settings for the Swarm device ID. These settings are determined by Swarm for identifying and communicating with each individual device. Since there are no variable parameters, the correct checksum has been added.

```
$CS*10
```

Returns:

```
$CS DI=<dev_ID>,DN=<dev_name>*xx
```

Value	Description
dev_ID	Device ID that identifies this device on the Swarm network
dev_name	Device type name

Notes:

An **ERR** response will be generated if the command is not entered exactly as shown with no additional spaces or characters.

Example:

```
$CS*10
```

```
$CS DI=0x000e57,DN=TILE*10
```

The Device ID is **0x000e57**, and the device is a **Tile**

$DT - Date/Time

Set or query the rate for **$DT** unsolicited report messages for date and time. Also can retrieve the most current **$DT** message. See unsolicited message description for **$DT** message format.

```
$DT <@|?|<rate>>*xx
```

Parameter	Description
`@`	Repeat most recent **$DT** message
`?`	Query current **$DT** rate
`rate`	Disable or set rate of **$DT** messages

Returns one of:

Value	Description
`$DT <YYYY><MM><DD><hh><mm><ss>,<flag>*xx`	The most recent **$DT** message.
`$DT <rate>*xx`	The current **$DT** rate
`$DT OK*xx`	`rate` updated successfully
`$DT ERR*xx`	Command input error

Parameter	Description
YYYY	Year (1970..2099)
MM	Month (01..12)
DD	Day (01..31)
hh	Hour (00..23)
mm	Minutes (00..59)
ss	Seconds (00..59)
flag	**I** Date/time is invalid **V** Date/time is valid

Notes:

<rate> is a value between 1 and 2147483647 (2^{31}-1) or 0. It will be the number of seconds in between each message. If **<rate>** is 0, no messages will be sent.

If **<rate>** is valid, no **$DT** messages will be sent by the device until the GPS has obtained a valid time reference to set its internal date and time, as indicated by the **$TILE DATETIME*35** message . If the GPS loses its time reference, the message is sent with a flag indicating an invalid state.

Examples:

Calling the most recent date/time message:

```
$DT @*70
```

```
$DT 20190408195123,V*41
```

Returns a date/time of **`April 8th, 2019 7:51:23 PM GMT`**. The date/time is **`valid.`** **`<rate>`** does not have to be valid to call the most recent date/time message.

Setting the rate of date/time messages:

```
$DT 300*03
```

```
$DT OK*34
```

Sets the rate of date/time messages to one message every **`300`** seconds.

Querying the rate of date/time messages:

```
$DT ?*0f
```

```
$DT 60*36
```

Returns a rate of one message every **`60`** seconds.

$FV - Firmware Version

Returns the current device firmware version. Since there are no parameters, the correct checksum has been added.

```
$FV*10
```

Returns one of:

Value	Description
`$FV <version_string>*xx`	The current firmware version
`$FV ERR*xx`	Command input error

Notes:

An **ERR** response will be generated if the command is not entered exactly as shown with no additional spaces or characters.

Example:

```
$FV*10
```

```
$FV 2021-07-16-00:10:21,v1.1.0*74
```

The firmware version on the device is **1.1.0**

$GJ - GPS Jamming/Spoofing Indication

Set or query the rate for **$GJ** unsolicited report messages for jamming and spoofing indicators. Also can retrieve the most current **$GJ** message.

```
$GJ <@|?|<rate>>*xx
```

Parameter	Description
`@`	Repeat most recent **$GJ** message
`?`	Query current **$GJ** rate
`rate`	Disable or set rate of **$GJ** messages

Returns one of:

Parameter	Description
`$GJ <spoof_state>,<jamming_level>*xx`	The most recent **$GJ** message.
`$GJ <rate>*xx`	The current **$GJ** rate
`$GJ OK*xx`	Parameters updated successfully
`$GJ ERR*xx`	Command input error

Parameter	Description
`spoof_state`	**0** Spoofing unknown or deactivated **1** No spoofing indicated **2** Spoofing indicated **3** Multiple spoofing indications
`jamming_level`	Value ranging from 0 to 255 indicating how much carrier wave (CW) jamming is detected. 0 = no CW jamming, 255 = strong CW jamming

Notes:

`<rate>` is a value between 1 and 2147483647 (2^{31}-1). It will be the number of seconds in between each message. If **`<rate>`** is 0, no messages will be sent.

Examples:

Calling the most recent GPS jamming/spoofing message:

```
$GJ @*6d
```

```
$GJ 1,23*31
```

Returns a spoof state of **`No spoofing indicated`**. The carrier wave jamming level is **`23.`**

Setting the rate of GPS jamming/spoofing messages:

```
$GJ 3600*28
```

```
$GJ OK*29
```

Sets the rate of GPS jamming/spoofing messages to one message every **3600** seconds.

Querying the rate ofGPS jamming/spoofing messages:

```
$GJ ?*12
```

```
$GJ 10*2c
```

Returns a rate of one message every **`10`** seconds.

$GN - Geospatial information

Set or query the rate for **$GN** unsolicited report messages for date and time. Also can retrieve the most current **$GN** message.

```
$GN <@|?|<rate>>*xx
```

Parameter	Description
@	Repeat most recent **$GN** message
?	Query current **$GN** rate
rate	Disable or set rate of **$GN** messages

Returns one of:

Value	Description
$GN <latitude>,<longitude>,<altitude>,<course>,<speed>*xx	The most recent **$GN** message
$GN <rate>*xx	The current **$GN** rate
$GN OK*xx	Parameters updated successfully
$GN ERR*xx	Command input error

Parameter	Description
`latitude`	Latitude in d.dddd format (float). The latitude is presented in the N basis (negative latitudes are in the southern hemisphere)
`longitude`	Longitude in d.dddd format (float). The longitude is presented in the E basis (negative longitudes are in the western hemisphere)
`altitude`	Altitude in meters (float)
`course`	Course in degrees (0..359) (float). Course proceeds clockwise, with 0=north, 90=east, 180=south, and 270=west
`speed`	Speed in kilometers per hour (0..999) (float)

Notes:

`<rate>` is a value between 1 and 2147483647 (2^{31}-1). It will be the number of seconds in between each message. If **`<rate>`** is 0, no messages will be sent.

If **`<rate>`** is valid, no **$GN** messages will be sent by the device until the GPS has obtained a valid position reference as indicated by the **$TILE POSITION*2d** message .

Examples:

Calling the most recent GPS message:

```
$GN @*69
```

```
$GN 37.8921,-122.0155,77,89,2*01
```

Returns a location of **`37.8921N, 122.0155W`**. The device's altitude is **`77m`**, its course is **`89 degrees`**, and it is moving at **`2 kilometers per hour`**.

Setting the rate of GPS messages:

```
$GN 30*2a
```

```
$GN OK*2d
```

Sets the rate of GPS messages to one message every **`30`** seconds.

Querying the rate of GPS messages:

```
$GN ?*16
```

```
$GN 15*2d
```

Returns a rate of one message every **`15`** seconds.

$GP - GPIO1 Control

This command allows control of the GPIO1 pin to allow indications or control the operation of the Tile.

```
$GP <?|<mode>>*xx
```

Parameter	Description
?	Display current GPIO1 mode
mode	Set GPIO1 pin mode

Returns one of:

Value	Description
$GP <mode>*xx	The current **$GP** mode
$GP OK*xx	Parameters updated successfully
$GP ERR*xx	Command input error

Notes:

The `?` option allows reading back the current setting. The **`mode`** parameter allows specifying how the GPIO1 pin will operate. The available modes are:

Mode	Description
0	Analog, pin is internally disconnected and not used (default)
1	Input, low-to-high transition exits sleep mode
2	Input, high-to-low transition exits sleep mode
3	Output, set low (1)
4	Output, set high (1)
5	Output, low indicates messages pending for client (2)
6	Output, high indicates messages pending for client (2)
7	Output, low while transmitting. Otherwise output is high (3)
8	Output, high while transmitting. Otherwise output is low (3)
9	Output, low indicates in sleep mode (4). Otherwise output is high
10	Output, high indicates in sleep mode (4). Otherwise output is low

It is the responsibility of the client to provide pull-up resistors to the 3.3V supply rail. See the reference design for resistor sizing. All output modes are open drain.

(1) - These two variations allow the user application to use GPIO1 as a general purpose output.

(2) - If either of these modes are selected, the pin will indicate if the Tile has received one or more unread messages and is holding them for the client. If multiple messages are pending for the client, the pin will maintain the state until all messages have been read.

(3) - GPIO will change states no faster than 500μs prior to the Tile beginning to transmit and 500μs after the Tile has finished transmitting.

(4) - If either of these modes are selected, the pin will be set to the selected state after the client has issued the **$SL** command. The pin will return to the awake state only if the sleep mode is terminated by the **S** or **T** parameter being reached, or activity is detected on the serial RX line.

Examples:

Querying the GPIO1 pin mode:

```
$GP ?*08
```

```
$GP 7*00
```

The GPIO1 pin mode is set to **Output, low while transmitting. Otherwise output is high**.

Setting the GPIO1 pin mode to input and wake on a high-to-low transition:

```
$GP 2*05
```

```
$GP OK*33
```

Sets the GPIO1 pin mode to **input and wake on a high-to-low transition**.

$GS - GPS Fix Quality

Set or query the rate for **$GS** unsolicited report messages for date and time. Also can retrieve the most current **$GS** message..

```
$GS <@|?|<rate>>*xx
```

Parameter	Description
`@`	Repeat most recent **$GS** message
`?`	Query current **$GS** rate
`rate`	Disable or set rate of **$GS** messages

Returns one of:

Parameter	Description
`$GS <hdop>,<vdop>,<gnss_sats>,<unused>,<fix>*xx`	The most recent **$GS** message
`$GS <rate>*xx`	The current **$GS** rate
`$GS OK*xx`	Parameters updated successfully
`$GS ERR*xx`	Command input error

Parameter	Description
hdop	Horizontal dilution of precision (0..9999) (integer = actual hdop * 100)
vdop	Vertical dilution of precision (0..9999) (integer = actual vdop * 100)
gnss_sats	Number of GNSS satellites used in solution (integer)
unused	Always reads as 0, unused
fix_type	**NF** No fix **DR** Dead reckoning only solution **G2** Standalone 2D solution **G3** Standalone 3D solution **D2** Differential 2D solution **D3** Differential 3D solution **RK** Combined GNSS + dead reckoning solution **TT** Time only solution

Notes:

<rate> is a value between 1 and 2147483647 (2^{31}-1). It will be the number of seconds in between each message. If **<rate>** is 0, no messages will be sent.

If **<rate>** is valid, no **$GS** messages will be sent by the device until the GPS has obtained a valid position reference as indicated by the **$TILE POSITION*2d** message .

Examples:

Setting the rate for geospatial information messages to 1 per second:

```
$GS 1*05
```

```
$GS OK*30
```

If **<rate>** is valid, no **$GN** messages will be sent by the device until the GPS has obtained a valid position reference as indicated by the **$TILE POSITION*2d** message .

Calling the most recent geospatial information message:

```
$GS @*74
```

```
$GS 109,214,9,0,G3*46
```

Returns a HDOP of **1.09,** VDOP of **2.14,** the device is using **9** GNSS satellites for this solution**,** and it is a **Standalone 3D solution**.

Querying the rate of geospatial information messages:

```
$GS ?*0b
```

```
$GS 120*07
```

Returns a rate of one message every **120** seconds.

$MM - Messages Received Management

Manage received messages in the device database. Has ability to return count, mark and delete messages. This command is only applicable to Tiles on firmware version 1.0.0+.

```
$MM <C=<U|*>|D=<msgID|R|*>|M=<msgID|*>|N=<D|E|?>|R=<msgID|O|N>>*xx
```

Parameter	Description
`C=<U|*>`	Return count of unread (**U**) or all (*) messages
`D=<msgID|R|*>`	Delete message ID (**msgID**), all read (**R**), or all messages (*)
`M=<msgID|*>`	Mark message ID (**msgID**) or all messages (*) as read
`N=<D|E|?>`	Message notifications disabled (**D**), enabled (**E**), or query (**?**)
`R=<msgID|O|N>`	Read message ID (**msgID**), oldest (**O**), or newest (**N**) message

Returns one of:

Parameter	Description
`$MM <msg_count>*xx`	Number of messages read/all/deleted
`$MM <appID>,<data>,<msg_id>,<es>*xx`	Response to reading a message
`$MM MARKED,<msg_id>*xx`	<**msg_id**> marked as read successfully
`$MM DELETED,<msg_id>*xx`	<**msg_id**> deleted successfully
`$MM OK*xx`	Message command succeeded
`$MM ERR,BADPARAM*xx`	Invalid command or argument to **$MM** command
`$MM ERR,DBXINVMSGID*xx`	Invalid message ID in **D** or **R** command
`$MM ERR,DBXNOMORE*xx`	No messages found when using **R=<O|N>**

Notes:

Messages have three states: Unread, read, and deleted. Once an unread message is read, it's state changes to read. It can subsequently be read again, when called by **msgID**. If a message is deleted, it can no longer be counted, read, marked, or deleted.

If a message is marked read using the **M=<msg_id>** or **M=*** command, marking it as read again is not an error.

<msg_count> is a number indicating the number of messages that are unread in response to the **C=U** command, total number of read and unread messages in response to the **C=*** command, or the number of messages deleted in response to the **D=*** command.

<appID> is the application ID tag of the message. This field will only be returned on Firmwavre version v1.1.0+

<**data>** is in the same format as the original **$RD** response returns.

<**msg_id>** is assigned by the device, and is an unsigned 64-bit value comprised of the device ID, a day of year counter, and a message of day counter. Responses that have a 0 as the message ID indicates the message has not been placed in the queue and therefore has no ID. The value should be treated as a simple arbitrary number.

<es> is the epoch seconds time when the message was received by the Tile.

Examples:

Counting all unread messages:

```
$MM C=U*0b
```

```
$MM 3*13
```

Three messages are unread.

Deleting all read messages:

```
$MM D=R*0b
```

```
$MM 4*14
```

Four read messages were deleted.

Marking a specific message as read:

```
$MM M=21990235111426*56
```

```
$MM MARKED,21990235111426*1e
```

Message with msg_id **21990235111426** has been marked as read.

Reading the newest received message:

```
$MM R=N*01
```

```
$MM AI=10650,6578616d706c65206d657373616765,21990235111426,1584494275*52
```

The message (msg_id = **21990235111426)** is returned. The message was sent with the application ID **10650**. The hexadecimal data returned is **6578616d706c65206d657373616765** (in ascii **= "example message").** The epoch seconds at which the Tile received the message is **1584494275** (Date/Time = **Wednesday, March 18, 2020 1:17:55 PM)**. This message is now marked as read.

Reading the newest received message again:

```
$MM R=N*01
```

```
$MM ERR,DBXNOMORE*03
```

There are no existing unread messages.

Enable message notifications:

```
$MM N=E*16
```

```
$MM OK*24
```

Message notifications are now enabled.

Query message notifications:

```
$MM N=?*6c
```

```
$MM N=E*16
```

Message notifications are enabled.

$MT - Messages to Transmit Management

Manage messages to be transmitted in the device database. Has ability to return count, list, and delete unsent messages.

```
$MT <C=<U>|D=<msgID|U>|<L=msgID|U>>*xx
```

Parameter	Description
`C=<U>`	Return count of unsent (**U**) messages
`D=<msgID|U>`	Delete message ID (**msgID**) or all unsent (**U**) messages
`L=<msgID|U>`	List message ID (**msgID**) or all unsent (**U**) messages

Returns one of:

Parameter	Description
`$MT <msg_count>`	Number of unsent messages
`$MT <data>,<msg_id>,<es>*xx`	Response to listing a message
`$MT DELETED*xx`	<**msg_id**> deleted successfully
`$MT OK*xx`	Message command succeeded
`$MT ERR,BADPARAM*xx`	Invalid command or argument to **$MT** command
`$MT ERR,DBXINVMSGID*xx`	Invalid message ID in **D** or **L** command
`$MT ERR,DBXNOMORE*xx`	No messages found when using **L=<U>**

Notes:

$MT command inputs are case sensitive

`<msg_count>` is a number indicating the number of messages that are unsent in response to the **C=U** command or the number of messages deleted in response to the **D=U** command

<data> is in the same format as the original **$TD** command accepts for message content

<msg_id> is assigned by the device, and is an unsigned 64-bit value comprised of the device ID, a day of year counter, and a message of day counter. Responses that have a 0 as the message ID indicates the message has not been placed in the queue and therefore has no ID. The value should be treated as a simple arbitrary number.

Examples:

Counting all unsent messages:

```
$MT C=U*12
```

```
$MT 12*3a
```

Twelve messages are unsent.

Listing a specific unsent message:

```
$MT L=4428826476689*7c
```

```
$MT 68692066726f6d20737761726d,4428826476689,1605639598*55
```

The unsent message (msg_id = **4428826476689)** is returned. The hexadecimal data returned is **68692066726f6d20737761726d** (in ascii **= "hi from swarm").** The epoch seconds at which the Tile received the message is **1605639598** (Date/Time = **Tuesday, November 17, 2020 6:59:58 PM)**.

Listing all unsent messages:

```
$MT L=U*1d
```

```
$MT 68692066726f6d20737761726d,4428826476689,1605639598*55
$MT 686f6c6120646573646520737761726d,4428826476690,1605639664*53
```

Two messages are unsent and are listed with a line each.

Deleting all unsent messages:

```
$MT D=U*15
```

```
$MT 1*08
```

One message was deleted.

$PO - Power Off

Power off the device. If fully supported, after issuing **$PO*1f** the customer should command any Tile power supplies to disconnect. If power is not disconnected, the Tile enters a low power mode until power is completely removed and restored. Since there are no variable parameters, the correct checksum has been added.

```
$PO*1f
```

Returns one of:

Value	Description
`$PO OK*xx`	Command has been accepted and the Tile will immediately attempt to power off
`$PO ERR*xx`	Command input error

Notes:

An **OK** response confirms that the Tile will shut down. The user should disconnect power from the Tile at this point. If left connected, the Tile will draw approximately 6mA on its 3.3V input. The Tile will not boot again until power has been completely removed and then restored.

Example:

```
$PO*1f

$PO OK*3b

$TILE BOOT,SHUTDOWN*06
```

$PW - Power Status

Set or query the rate for **$PW** unsolicited report messages for device power state. Also can retrieve the most current **$PW** message.

```
$PW <@|?|<rate>>*xx
```

Parameter	Description
`@`	Repeat most recent **$PW** message
`?`	Query current **$PW** rate
`rate`	Disable or set rate of **$PW** messages

Returns one of:

Parameter	Description
$PW <unused>,<unused>,<unused>,<unused>,<temp>*xx	The most recent **$PW** message
`$PW <rate>*xx`	The current **$PW** rate
`$PW OK*xx`	Parameters updated successfully
`$PW ERR*xx`	Command input error

Parameter	Description
`unused`	Will always show as 0.00000
`temp`	CPU Temperature in degrees C to one decimal point (float)

Notes:

<rate> is a value between 1 and 2147483647 (2^{31}-1). It will be the number of seconds in between each message. If **<rate>** is 0, no messages will be sent.

Examples:

Calling the most recent power status message:

```
$PW @*67
```

```
$PW 0.00000,0.00000,0.00000,0.00000,25.0*3e
```

Returns a device temperature of **25.0**℃.

Setting the rate of power status messages:

```
$PW 30*24
```

```
$PW OK*23
```

Sets the rate of power status messages to one message every **30** seconds.

Querying the rate of power status messages:

```
$PW ?*18
```

```
$PW 900*1e
```

Returns a rate of one message every **900** seconds.

$RD - Receive Data Message

This unsolicited message provides an ASCII-encoded hexadecimal string with the user data received from the Swarm network. Some fields also include signal quality information for the received message. Received data unsolicited messages can be enabled/disabled using the **$MM** command with the message notification option.

```
$RD <appID>,<rssi>,<snr>,<fdev>,<data>*xx
```

Parameters	Description
`appID`	Application ID tag of message (shows only with firmware version v1.1.0+)
`rssi`	Received signal strength in dBm for packet (integer)
`snr`	Signal to noise ratio in dB for packet (integer)
`fdev`	Frequency deviation in Hz for packet (integer)
`data`	ASCII encoded data packet

Example:

```
$RD AI=6078,RSSI=-84,SNR=12,FDEV=9,596f7527766520676f74206d61696c21*6e
```

The received packet has an **application ID** of 6078**,** a **rssi** of -84 dBm, a **snr** of -12 dB, and a **fdev** of 9 Hz. The returned encoded hexadecimal data from the packet is **596f7527766520676f74206d61696c21** (in ascii **= “You’ve got mail!”).**

$RS - Restart Device

Perform a software cold restart of the device. Since there are no variable parameters, the correct checksum has been added.

```
$RS*01
```

Returns one of:

Value	Description
`$RS OK*xx`	Command has been accepted and the device will immediately perform a hardware restart
`$RS ERR*xx`	Command input error

Notes:

An **OK** response confirms that the device will successfully restart. No external power cycling is required.

Example:

```
$RS*01
```

```
$RS OK*25
```

```
$TILE BOOT,RESTART*59
```

Notes:

In order to resolve the **DBXTOHIVEFULL** error, ensure you're running firmware version V1.0 or greater and run the following command.

```
$RS dbinit*3d
```

$RT - Receive Test

Set or query the rate for **$RT** unsolicited report messages for device power state. Also can retrieve the most current **$RT** message.

```
$RT <@|?|<rate>>*xx
```

Parameter	Description
`@`	Repeat most recent **$RT** message
`?`	Query current **$RT** rate
`rate`	Disable or set rate of **$RT** messages

Returns one of:

Parameter	Description
$RT RSSI=<rssi_sat>,SNR=<snr>,FDEV=<fdev>,TS=<time>,DI=<sat_id>*xx	The most recent **$RT** message from a satellite. Appears independently of the **rssi_background** noise messages
`$RT RSSI=<rssi_background>*xx`	The most recent **$RT** message. This is a measure of the background noise environment (and is not related to **rssi_sat**)
`$RT <rate>*xx`	The current **$RT** rate
`$RT OK*xx`	Parameters updated successfully
`$RT ERR*xx`	Command input error

Parameter	Description
`rssi_background`	Received background noise signal strength in dBm for open channel (integer). For reliable operation, this value should consistently be less than (more negative) than **-93** dBm: **rssi_background** = -90 dBm Bad (not likely to work) **rssi_background** = -93 dBm Marginal **rssi_background** = -97 dBm OK **rssi_background** = -100 dBm Good **rssi_background** = -105 dBm Great
`rssi_sat`	Received signal strength in dBm for packet from satellite (integer)
`snr`	Signal to noise ratio in dB for packet (integer)
`fdev`	Frequency deviation in Hz for packet (integer)
`time`	Date and time (UTC) of received packet
`sat_id`	Device ID of satellite heard (hexadecimal)

Notes:

`<rate>` is a value between 1 and 2147483647 (2^{31}-1). It will be the number of seconds in between each message. If **`<rate>`** is 0, no messages will be sent, and no packets received from a satellite will be displayed.

For reliable performance, the noise floor can be measured with an antenna connected to the device in the final built up configuration. The **rssi_background** value reported by **$RT** should be less than (a more negative number) than **-93** dBm. If the **rssi_background** noise floor is more than (a less negative number, e.g. -88 dBm) **-93** dBm, then the customer should relocate the device to a different location, reconfigure the device ground plane, and also check that their DC-DC power supplies on their own PCB are not injecting RF background noise into the device.

Examples:

Calling the most recent receive test message. This will return the last packet received from a satellite:

```
$RT @*66
```

```
$RT RSSI=-102,SNR=-1,FDEV=426,TS=2020-10-02 13:56:21,DI=0x000568*04
```

Returns a satellite packet **rssi_sat** of **-102** dBm, SNR of **-1** dB, frequency deviation of **426** hz, a received time of **October 2nd, 2020 at 1:56:21 PM**, and the satellite’s device ID is **0x000568**.

Setting the rate of receive test messages:

```
$RT 1*17
```

```
$RT OK*22
```

Sets the rate of receive test messages to one message every **1** second. The format of these scheduled messages will be the simple noise floor message. Any packets received from a satellite will be displayed as they are received, independently of the **$RT** rate.

Querying the rate of receive test messages:

```
$RT ?*19
```

```
$RT 5*13
```

Returns a rate of one message every **5** seconds.

$SL - Sleep mode

This command puts the device into a low-power sleep mode.

```
$SL [S=<seconds>|U=<[YYYY-MM-DD ]hh:mm:ss>]*xx
```

Parameter	Description
S=<seconds>	Sleep for this many seconds
U=<[YYYY-MM-DD]hh:mm:ss>	Sleep until date (optional) and time

Returns one of:

Value	Description
$SL OK*xx	Sleep period accepted, device is now non-responsive
$SL WAKE,<cause>*xx	Device has woken from selected sleep mode
$SL ERR,TIMENOTSET*xx	Time not yet set from GPS
$SL ERR,BADPARAM*xx	Invalid number of seconds or date/time value
$SL ERR,NOCOMMAND*xx	No **S** or **U** parameter is present
$SL ERR,NOTIME*xx	Attempt to sleep before time is set

The **S** parameter is the number of seconds to sleep. This value may range from 5 to 31536000 (approximately 1 year) seconds. A value not within this range will return **$SL ERR,BADPARAM**. If the command is accepted, the device will emit **$SL OK** and enter sleep mode for the requested duration.

The **U** parameter is a time and optional date the device should sleep until and then wake. If the date is not specified and the time to sleep until is less than the current time, the time is presumed to be in the next day. For example, if the current time is 11:00:00 and **$SL U=09:00:00** is issued, the device will wake 22 hours from now. If a date and time are specified, and that date/time is before the current date/time, **$SL WAKE** will be immediately issued.

The **`$SL WAKE,<cause>`** message is emitted after the Tile wakes from a user commanded sleep mode. The value of **`cause`** will be one of the following:

Cause	Description
`GPIO`	GPIO input changed from inactive to active state
`SERIAL`	Activity was detected on the RX pin of the Tile's UART
`TIME`	The **`S`** or **`U`** parameter time has been reached

If UART activity wakes the Tile, the **`TIMEOUT`** message will not be emitted as the Tile is now awake.

In sleep mode, the real-time clock is not GPS disciplined, and is therefore subject to some degree of drift. The longer the device is asleep, the more the drift will accumulate. The drift is estimated to be 20ppm, so if the Tile is commanded to sleep for 100 hours, the drift may result in the Tile actually sleeping in the range of 99.998 to 100.002 hours. The user should be aware of this when selecting a sleep with a long duration.

If the GPIO1 pin is configured as an input to wake the Tile, the sleep mode will be terminated if activity occurs on GPIO1. If the GPIO1 pin is configured as an output that indicates the Tile's sleep mode, GPIO1 will transition to the appropriate state if the **`$SL OK`** message is emitted.

Examples:

Commanding the Tile to sleep for 3600 seconds (1 hour):

```
$SL S=3600*54
```

```
$SL OK*3b
```

```
$SL WAKE,TIME @ 2019-04-11 18:58:03*77
```

If the Tile receives any serial input before the planned wake time, then the Tile will wake with a message such as:

```
$SL WAKE,SERIAL @ 2019-04-11 18:57:45*6f
```

Similarly, if the GPIO1 pin is configured to wake on a high-to-low (or low-to-high) transition, then on transition on the GPIO1 pin, the Tile will also wake with a message such as:

```
$SL WAKE,GPIO @ 2019-04-11 18:57:55*7f
```

Commanding the Tile to sleep until October 1st, 2021 at 4:30:00 PM:

```
$SL U=2021-10-01 16:30:00*72
```

```
$SL OK*3b
```

$TILE - Tile Status

These unsolicited status messages indicate the readiness of the Tile for normal operation. This includes the conditions at power up, GPS acquisition, and certain error conditions. Tile status messages cannot be disabled.

```
$TILE <msg>,[<data>]*xx
```

Parameter	Description
msg	**BOOT** - Boot process progress with the following **data** reason: ● **ABORT** - A firmware crash occurred that caused a restart ● **POWERON** - Power has been applied ● **RUNNING** - Boot has completed and ready to accept commands ● **UPDATED** - A firmware update was performed ● **VERSION** - Current firmware version information **DATETIME** - The first time GPS has acquired a valid date/time reference **POSITION** - The first time GPS has acquired a valid position 3D fix **DEBUG** - Debug message (**data** - debug text) **ERROR** - Error message (**data** - error text)

Notes:

A **data** message follows the **BOOT** message to indicate the reason for the startup.

The **POSITION** message and **DATETIME** message may occur in any order. Depending on the GPS signal quality, it may take several minutes before the **DATETIME** or **POSITION** message is emitted.

The customer application should wait until the boot process is complete and it has received the **$TILE BOOT,RUNNING*49** message before executing any commands.

$TD - Transmit data

This command transmits data to the Swarm network.

```
$TD [AI=<appID>|HD=<hold_dur>|ET=<expire_time>,]<[string|data]>*xx
```

Parameter	Description
AI=<appID>	Application ID tag for message. Can be specified only with firmware version v1.1.0+
HD=<hold_dur>	Hold duration of message in seconds (optional, default = 172800 seconds, minimum = 60 seconds)
ET=<expire_time>	Time to expire message in epoch seconds
<string|data>	1 to 192 bytes of data (ASCII string) 2 to 384 bytes (hexadecimal written as ascii)

Notes:

In order to resolve the **DBXTOHIVEFULL** error, ensure you're running firmware version V1.0 or greater and run the following command.

```
$RS dbinit*3d
```

In order to send a **$TD** command, you must first wait for a **$TILE DATETIME*35** response after power up. The Tile must wait for a valid time in order to accept a transmit command. All other commands are functional prior to receiving the **$TILE DATETIME*35** response.

Returns one of:

Value	Description
`$TD OK,<msg_id>*xx`	Message accepted for sending
`$TD SENT RSSI=<rssi_sat>,SNR=<snr>,FDEV=<fdev>,<msg_id>*xx`	This is an unsolicited response that occurs at the time when the message is sent to and acknowledged by the satellite. It includes the RF information from the acknowledgment
`$TD ERR,BADAPPID*52`	Invalid application ID
`$TD ERR,BADDATA*0e`	Message has odd number or non-hex characters when sending data as hexadecimal
`$TD ERR,BADEXPIRETIME,0*18`	Invalid hold time
`$TD ERR,ERR*1c`	Unspecified error
`$TD ERR,HOLDTIMEEXPIRED,<msg_id>*xx`	Unable to send within requested hold time
`$TD ERR,NODEVICEID*4d`	The Swarm device ID has not yet been set - contact Swarm Support
`$TD ERR,NOSPACE,0*00`	No space for message
`$TD ERR,TIMENOTSET*5b`	Attempt to send message before time set by GPS
`$TD ERR,DBXTOHIVEFULL*1d`	Queue for queued messages is full. Maximum of 2048 messages may be held in the queue.
`$TD ERR,TOOLONG,0*1b`	Message is too large to send

Notes:

There is a small possibility of duplicate messages being sent over the network depending on the RF environment around the Tile

Messages can be delivered to the Hive out of sequence.

The **HD** and **ET** parameters are optional but must occur before the **<data>** portion of the command. Both affect the time when a message is expired and is no longer considered available to transmit. Expired messages are removed from the outgoing packet database. If neither option is present, the default hold time of **172800** seconds (48 hours) will be used. If both options are specified, the **<hold_dur>** will be ignored. Expired messages will only be removed from the outgoing database after a Swarm Tile communicates with a Satellite.

<appID> is the application ID tag for the message. If this option is not specified the message will be sent with application ID 0.

Application ID Value	Description
0 to **65535**	The application ID tag for the message. Swarm reserves the use of application IDs 65000 to 65535.
Any other value	An error message (**$TD ERR,BADAPPID*52**) is returned.

<hold_dur> is the number of seconds to hold the message if it has not been sent before expiring it from the database.

Hold Duration Value	Description
60 to **31536000** (one year)	The message will be considered expired if not sent within the specified number of seconds. The maximum duration is one year from the current time.
Any other value	An error message (**$TD ERR,BADEXPIRETIME*18**) is returned.

<expire_time> is an epoch second date after which the message will be expired if it has not been sent.

Expiration Time Value	Description
Any other value	An error message (**$TD ERR,BADEXPIRETIME*18**) is returned.

1577836800 to **2147483647** (2020-01-01 00:00:00 to 2038-01-19 03:14:07)	The message will be considered expired if not sent before the specified time. If the specified time is greater than **1577836800** and less than or equal to the current UTC time, the message will not be queued and an expired message **$TD ERR,HOLDTIMEEXPIRED,<msg_id>*xx** will be returned immediately.

<string|data> may be expressed one of two different ways. If all the data to be sent is in the ASCII character range from 0x20 (space) to 0x7e (tilde), then the data may be sent as a string. A string is specified by enclosing the data in double quotes, e.g., "Hello, world". It is permissible for the string to contain double quotes within the string, e.g., "Today is a "new" day". If the data to be sent includes one or more character outside the 0x20 to 0x7e range, then it must be specified as pairs of hex characters ('0'..'9', 'A'..'F', 'a'..'f'), and must be a multiple of 2. Sending 'Hello' as hex would be 48656C6C6F. Illegal characters or an odd number of characters will cause a **BADDATA** message to be returned.

<msg_id> is assigned by the device, and is an unsigned 64-bit value comprised of the device ID, a day of year counter, and a message of day counter. Responses that have a 0 as the message ID indicates the message has not been placed in the queue and therefore has no ID. The value should be treated as a simple arbitrary number.

Typical messaging flow:

1. User queues message to be sent with application ID **40**, with an expiry time of **7200** seconds (2 hours). The use of indexing parameters in the body of the message is recommended to aid in message identification and sequencing:

```
$TD AI=40,HD=7200,"{"d":"Demo message","t":"2021-02-26 14:28:56","seq":"00015"}"*09
```

2. Device acknowledges receipt of message, and message enters into device's internal message database. Message is given the message ID of **5354468575855**. Database has a maximum amount of 2048 messages stored. Database is non-volatile and messages persist through sleep mode and power cycling:

```
$TD OK,5354468575855*2a
```

3. If the message has not expired due to time (as it was set to expire 2 hours from when it was entered), when the device hears a satellite pass overhead it attempts to send any messages in its database. Messages are scheduled to be sent in the order they were entered into the device (e.g. first in database, first transmitted). Each message heard by the satellite is individually acknowledged with a **$TD SENT** response - the acknowledged message is referred to by its message ID (**5354468575855**)

```
$TD SENT,RSSI=-99,SNR=5,FDEV=32,5354468575855*6f
```

4. Acknowledged message in database is marked for deletion by device. Any unsent messages will be attempted to be sent by the device at a later time.

Examples:

Sending a message from the device in ASCII:

```
$TD "{"d":"Demo message","t":"2021-02-26 14:48:56","seq":"00017"}"*08

$TD OK,5354468575916*2c

$TD SENT,RSSI=-104,SNR=-3,FDEV=345,5354468575916*44
```

Sending a message from the device in ASCII that will expire in 1 hour:

```
$TD HD=3600,"{"d":"Demo message","t":"2021-02-26 14:51:21","seq":"00018"}"*17

$TD OK,5354468575917*2d

$TD SENT,RSSI=-103,SNR=2,FDEV=-67,5354468575917*70
```

Sending a message from the device in ASCII that will expire on 2022-01-01 12:34:56:

```
$TD ET=1641040496,"{"d":"Demo","t":"2021-02-26 15:01:22","seq":"00020"}"*45

$TD OK,5354468575919*23

$TD SENT,RSSI=-100,SNR=-3,FDEV=437,5354468575919*4d
```

Sending a message from the device in ASCII with application ID 3000 (requires firmware version v1.1.0+):

```
$TD AI=3000,"{"d":"Demo","t":"2021-02-26 15:33:22","seq":"00021"}"*52

$TD OK,5354468575919*23

$TD SENT,RSSI=-100,SNR=-3,FDEV=437,5354468575919*4d
```

Sending a message from the device in HEXASCII:

```
$TD 49206C6F766520537761726D*4c
```

```
$TD OK,5354468575916*2c
```

```
$TD SENT,RSSI=-107,SNR=3,FDEV=199,5354468575916*69
```