Two DeLorme Drive
Yarmouth, ME 04096
(Phone) 800-293-2389
(Fax) 207-846-7054
www.delorme.com

July 2007

DeLORME
GPS2056-10
GPS RECEIVER MODULE SPECIFICATION

General Description

The DeLorme GPS 2056-10 module is a high sensitivity, low power, SMD type 12-channel GPS receiver designed for a broad spectrum of OEM applications. Based on the fast and deep GPS signal search capabilities of the STMicroelectronics STA2056V-Palinuro chip, the DeLorme GPS2056 GPS module enables a complete GPS receiver system by simply adding an active or passive antenna and external power.

The GPS2056-10 module features an integrated RF block combined with the high performance CPU of ARM7TDMI microprocessor. The end result provides excellent navigation performance in the most challenging urban environments. The DeLorme GPS2056-10 offers RTCA-SC159 / WAAS / EGNOS support and provides excellent sensitivity and high configurability in a small package.

Features

- Single-chip highly integrated GPS solution for easy turn-key application
- Fully Integrated RF Section for direct Interface to Active or Passive Antenna Systems
- GPS code library embedded in ROM
- Standard output NMEA sentence structure
- Single 3.3V Supply Voltage with internal regulation
- Evaluation kit and Reference Design Available
- Low Power and standby modes for battery powered applications
- Dedicated SPI Bus for External Non-volitile Memory Improves Time To Fix
- PPS Output Synchronized to UTC for Time Synchronization Applications
- Compact SMD package minimizes footprint

Performance Specifications

12-channel, L1 (1575.42 MHz) GPS receiver

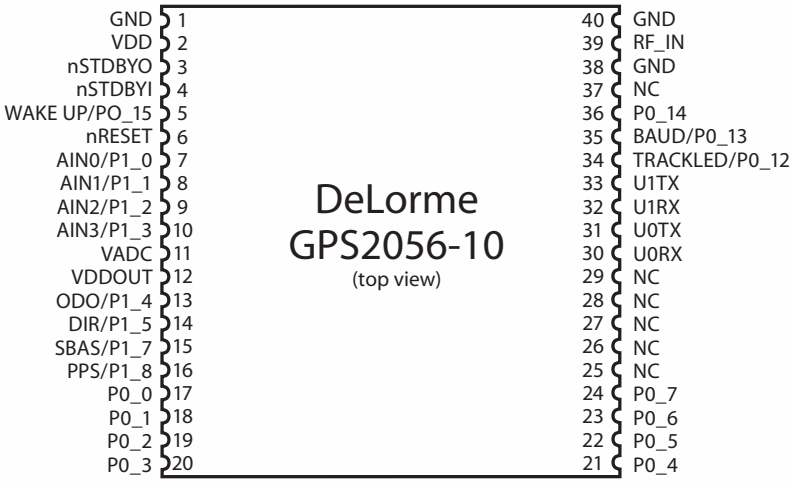
Reacquisition time	0.1 seconds average
Position Accuracy	10 meters
Acquisition Time	
• Hot Start 1 second 95% TTFF	
• Warm Start 38 seconds 95% TTFF	
• Cold Start 114 seconds 95% TTFF	
Reacquisition Time	0.1 seconds average
Position Accuracy	10 meters, 2D RMS
Altitude	<+/35 m vertical in term of 95%
Velocity	0.1 m/s

Time (PPS)	+/-62ns Accuracy synchronized to UTC time
DGPS (Optional)	<0.5m (depending on correction technique)
Acquisition Sensitivity	
• Hot 28dB-Hz	
• Warm 32dB-Hz	
• Cold 35dB-Hz	
Tracking Sensitivity	20dB-Hz
Velocity	515 m/s (1,000 knots) Max
Acceleration	4g Max
Altitude	10,000 m (~32,000 ft) Max
Jerk	20 m/s^3

Functional Description

The DeLorme GPS-2056-10 module simplifies embedded applications of GPS-based information systems. The important components of a basic GPS receiver subsystem are LNA, SAW Filter, RF front end, power conditioning, and the GPS baseband receiver itself. All these subsystems have been built into the DeLorme GPS module to eliminate the cost and time needed for the user to develop them independently. In simple terms, the user must connect an active or passive antenna, provide a power source, and connect the module to a host system via a serial port (or USB) to produce GPS positional data and enable a complete GPS receiver system.

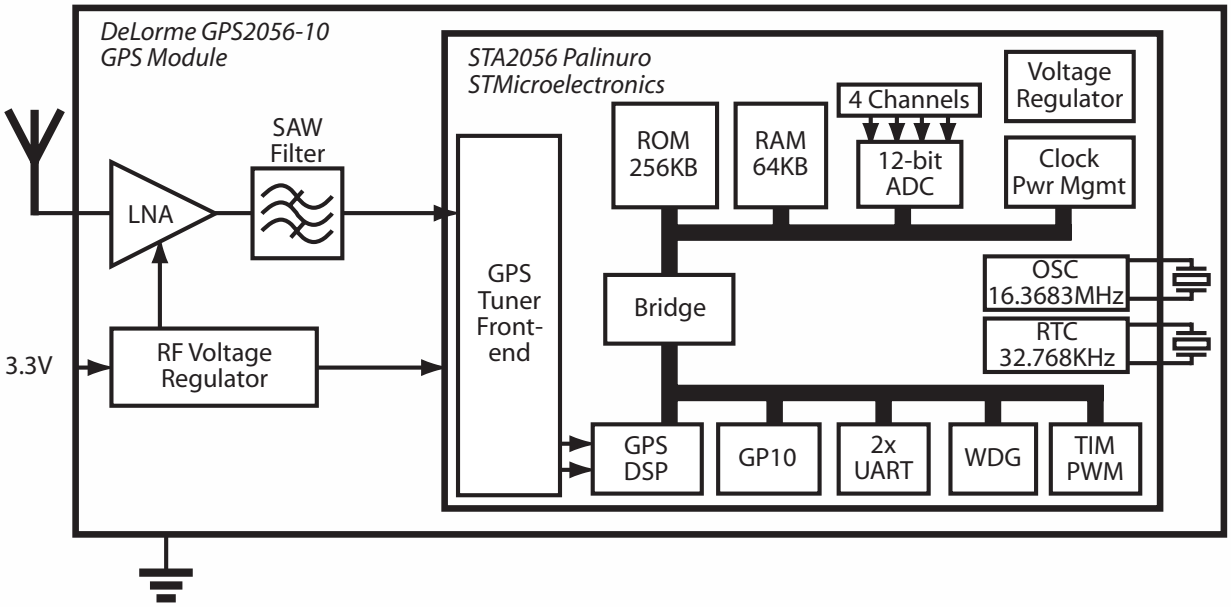
Module Pinout



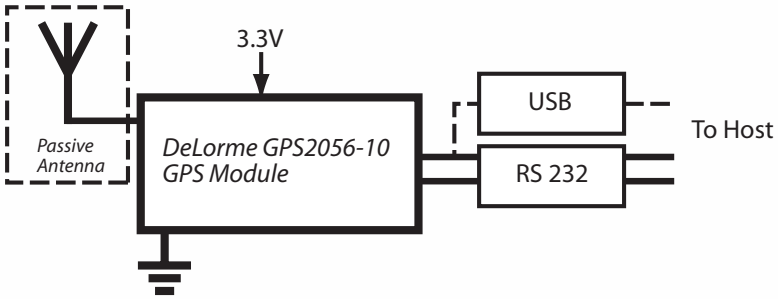
Pin Descriptions

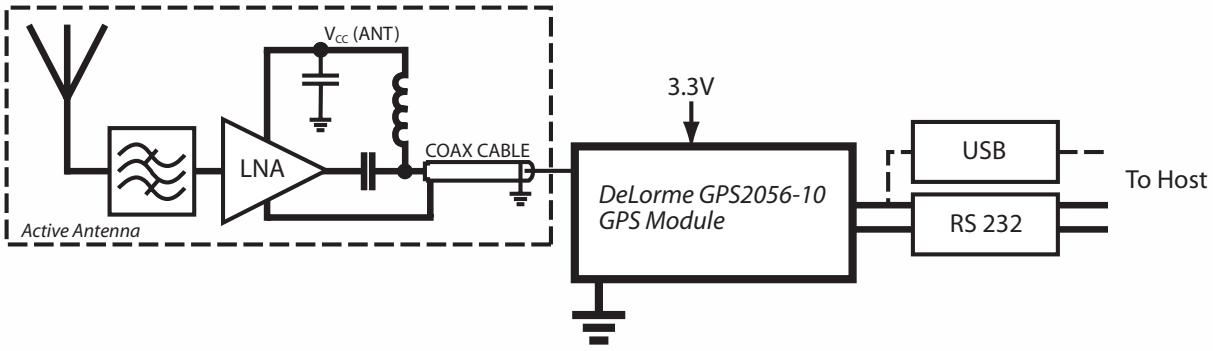
Pin	Name	Function	Description
1	GND	Power	Ground
2	VDD	Power	3.3v +/- 5% supply
3	nSTDBYO	Output	STANDBY output, active low
4	nSTDBYI	Input	STANDBY input, active low
5	WAKEUP / P0_15	Input	Wakeup input or GPIO P0_15
6	nRESET	Input	Module Reset, Active Low
7	AIN0 / P1_0	Tristate	Analog Input 0 or GPIO P1_0
8	AIN1 / P1_1	Tristate	Analog Input 1 or GPIO P1_1
9	AIN2 / P1_2	Tristate	Analog input 2 or GPIO P1_2
10	AIN3 / P1_3	Tristate	Analog Input 3 or GPIO P1_3
11	VADC	Power	External ADC Power 2.5V (float if not used)
12	VDDOUT	Power	3.3V VDD Output (off in standby mode)
13	ODO / P1_4	Tristate	Odometer Wheel Pulse Input or GPIO P1_4
14	DIR / P1_5	Tristate	Direction input or GPIO P1_5 (reverse gear)
15	SBAS / P1_7	Tristate	SBAS ON/OFF or GPIO P1_7 (tie low for off)
16	PPS / P1_8	Tristate	PPS Output or GPIO P1_8
17	P0_0	Tristate	GPIO P0_0
18	P0_1	Tristate	GPIO P0_1
19	P0_2	Tristate	GPIO P0_2
20	P0_3	Tristate	GPIO P0_3
21	P0_4	Tristate	GPIO P0_4
22	P0_5	Tristate	GPIO P0_5
23	P0_6	Tristate	GPIO P0_6
24	P0_7	Tristate	GPIO P0_7
25	NC	-	No Connect
26	NC	-	No Connect
27	NC	-	No Connect
28	NC	-	No Connect
29	NC	-	No Connect
30	U0RX	Input	NMEA Port RXD Input (to module)
31	U0TX	Output	NMEA Port TXD Output (from Module)
32	U1RX	Input	DeBug/RTCA Port RXD Input
33	U1TX	Output	DeBug/RTCA Port TXD Output
34	TRACKLED / P0_12	Tristate	Tracking LED output or GPIO P0_12
35	BAUD / P0_13	Tristate	Baud Rate Select or GPIO P0_13
36	P0_14	Tristate	GPIO P0_14
37	NC	-	No Connect
38	GND	RF Ground	RF Ground, connect only to RF
39	RF_IN	RF Input	GPS Signal In
40	GND	RF Ground	RF Ground, connect only to RF

Block Diagram



Typical Applications





Absolute Maximum Ratings

Symbol	Parameter	Value		Unit
		Min	Max	
V_{DD}	Voltage on VDD with respect to ground (VSS)	-0.3	+3.6	V
VADC	Voltage on VADC pin with respect to ground (VSS)	-0.3	+3.6	V
V_{IN}	Voltage on any pin with respect to ground (VSS)	-0.3	V_{DD} + 0.3	V
I_{OV}	Input Current on any pin during overload condition	-10	+10	mA
I_{TDV}	Absolute sum of all input currents during overload conditions		|200|	mA
T_{ST}	Storage Temperature	-55	+150	°C
ESD	ESD Susceptibility (Human Body Model)	2000		V

Recommended Operating Conditions

Symbol	Parameter	Value		Unit
		Min	Max	
V_{DD}	Digital Supply Voltage for I/O Circuitry	3.0	3.6	V
VADC	Analog Supply Voltage for the A/D Converter	V_{DD}	V_{DD}	V
T_A	Ambient Temperature under bais	-40	+85	°C
T_J	Junction Temperature under bias	-40	+105	°C

DC Electrical Characteristics

Symbol	Parameter	Conditions	Value			Unit
			Min	Typ	Max	
V_{IH}	Input High Level CMOS	With or w/o Hysteresis	2.0			V
V_{IL}	Input Low Level CMOS	With or w/o Hysteresis		0.4		V
V_{HYS}	Input Hysteresis CMOS Schmitt Trigger		0.4	0.8	1.2	V
	Input Hysteresis CMOS Schmitt Trigger	WAKEUP pin only	0.3	0.5		V
V_{OH}	Output High Level High Current Pins (GPIO)	I_{OH} = 8mA	V_{DD} - 0.8			V
	Output High Level Standard Current Pins	I_{OH} = 4mA	V_{DD} - 0.8			V
V_{OL}	Output Low Level High Current Pins (GPIO)	I_{OL} = 8mA		0.4		mA
	Output Low Level Standard Current Pins	I_{OL} = 4mA		0.4		mA
R_{WPU}	Weak Pull-up Resistor	Measured at 0.5 V_{DD}		50		Ω
R_{WPD}	Weal Pull-down Resistor	Measured at 0.5 V_{DD}		50		Ω

AC Electrical Characteristics

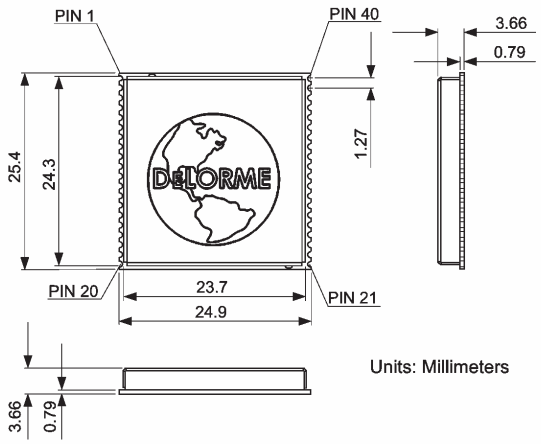
Symbol	Parameter	Conditions	Value			Unit
			Min	Typ	Max	
I_{DDRUN}	RUN mode current	Supply Voltage = 3.3V		55		mA
I_{DDSB1}	STANDBY mode current	LP Vreg and 32kHz OSC on		12	25	uA

Environmental Specifications

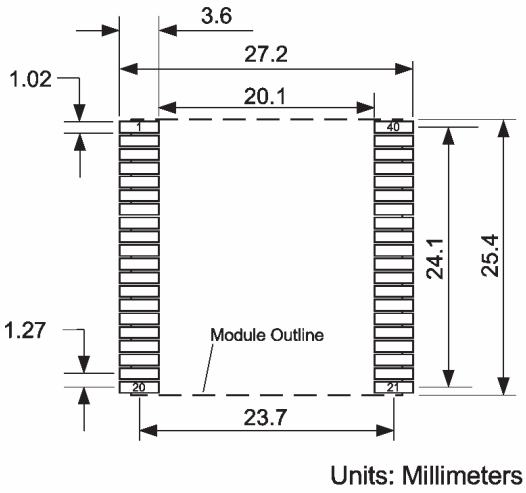
Operating Temperature -40 °C to +85 °C
Storage Temperature -55 °C to +100 °C
Relative Humidity 5% to 95%, non-condensing

Mechanical Specifications

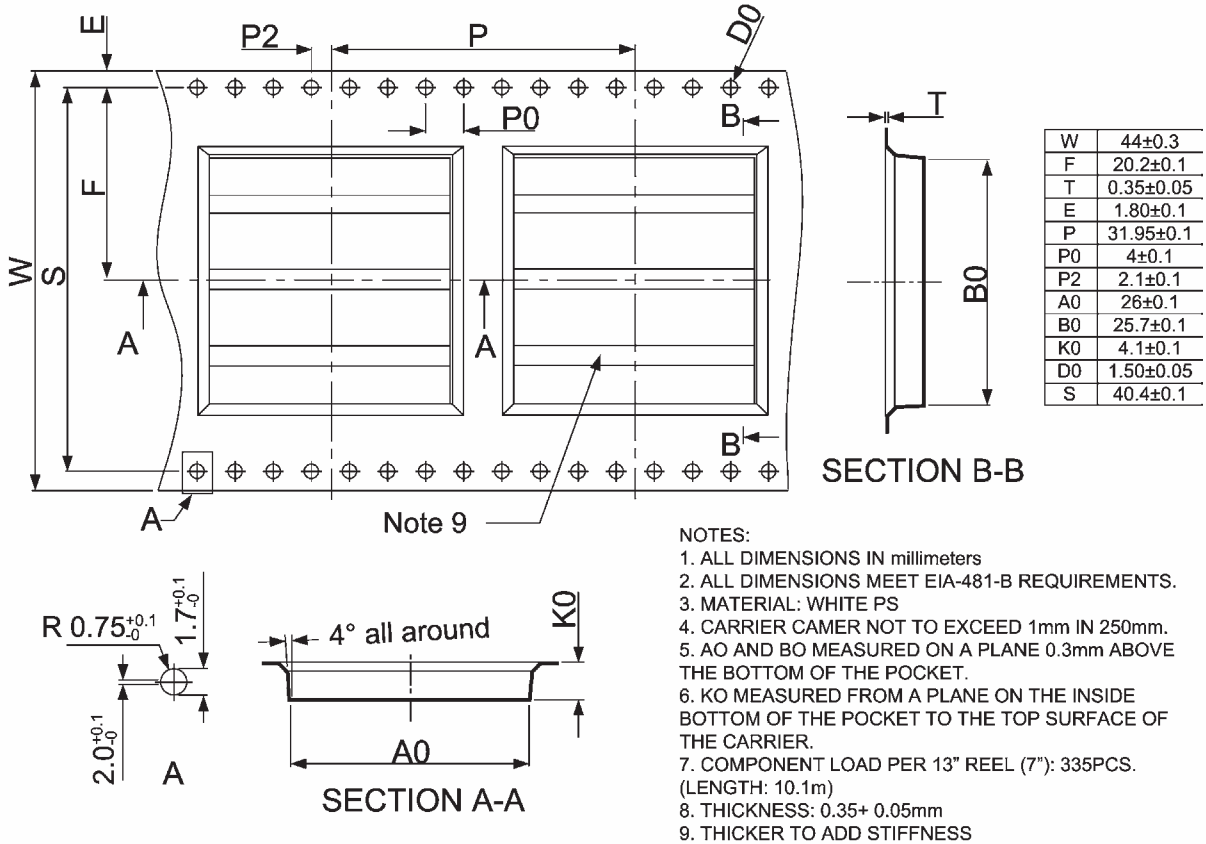
Outline Drawing:



Footprint: (Recommended PCB Layout)



Tape and Reel Drawing:



W	44±0.3
F	20.2±0.1
T	0.35±0.05
E	1.80±0.1
P	31.95±0.1
P0	4±0.1
P2	2.1±0.1
A0	26±0.1
B0	25.7±0.1
K0	4.1±0.1
D0	1.50±0.05
S	40.4±0.1

NOTES:
1. ALL DIMENSIONS IN millimeters
2. ALL DIMENSIONS MEET EIA-481-B REQUIREMENTS.
3. MATERIAL: WHITE PS
4. CARRIER CAMER NOT TO EXCEED 1mm IN 250mm.
5. AO AND BO MEASURED ON A PLANE 0.3mm ABOVE THE BOTTOM OF THE POCKET.
6. KO MEASURED FROM A PLANE ON THE INSIDE BOTTOM OF THE POCKET TO THE TOP SURFACE OF THE CARRIER.
7. COMPONENT LOAD PER 13" REEL (7"): 335PCS. (LENGTH: 10.1m)
8. THICKNESS: 0.35+ 0.05mm
9. THICKER TO ADD STIFFNESS

Ordering info

Packaging	Tape and Reel
Quantity per reel	335
P/N	GPS2056-10
Ordering Code	GM-205610-000

GPS Software

The GPS2056-10 ROM code includes the GPS software release 4.30.3+PAL. The SBAS software (release 1.6) has been also added; it is ON by default and can be deactivated via a software commands (see below).

Software Configuration

GPIO used for software configuration

To increase flexibility, some general purposes pins have been used to configure the software. The configuration values are read only one time at each system restart, just before the GPS startup, so after this period they can assume any other value. The GPIO involved have been configured as IN/OUT pins and an internal weak pull-up/ pull-down has been set on each one to have a default configuration pattern. The GPIO can be left unconnected if the default software configuration does not need to be modified. Care should be exercised on the configuration pin patterns at the startup time if the same pins are used for any other purpose (e.g. connecting the pin to a different device and drive it using a software command). If the configuration needs to be modified, external pull-up/pull-down resistors must be placed on the right pins.

The list below summarizes each pin/port number, associated function and default value.

Pin/Port Number	Description	Default Configuration Value
P0_14	Reserved - Drivers of this pin should be Tri-stated until NMEA Messages Start	Internal Weak Pull-up
SBAS/P1_7	Reserved - Drivers of this pin should be Tri-stated until NMEA Messages Start	Internal Weak Pull-up

NMEA message configuration

Using the BAUD pin two different sets of GPS NMEA out messages, with two different baud rates can be selected.

BAUD/P0_13	NMEA Messages Set	Default
Leave unconnected / external pull-up	Msg_list : GGA5 - VTG - GSA - GSV	X
	Baud Rate: 4800	
	Debug_mode: DEBUG_ON	
	Transmit mode: ON_UTC_SECOND	
External pull-down	Msg_list : GGA5 - GSA - GSV - RMC - TG - TS - PA	
	Baud Rate: 57600	
	Debug_mode: DEBUG_ON	
	Transmit mode: ON_UTC_SECOND	

Pins Configured by Software

One pin is configured by software to drive a 3DFix led. This pin can be also configured by a software command to switch OFF the led functionality and enable the Clock Out, to allow a bus clock measurement.

Port	Description
TRACKLED (GPIO P0_12)	3DFix led: in normal conditions, this pin is able to drive a led to flag the GPS 3DFix availability. Led ON means the 3DFix is available, led OFF means the fix is not available.

External Serial Flash Memory Configuration

Starting with ROM software version 4.30.3+PAL an external serial (SPI) memory is supported to store the GPS backup data (refer to the dedicated section for further details). Three different memories can be used: M95128 (EEPROM), M25P10 (1 Mbit FLASH) and M25P40 (4Mbit FLASH). To configure the software to address the installed memory type, two GPIO pins have been used as listed in the table below:

PO_2	PO_7	Description	Default
Leave unconnected / external pull-up	Leave unconnected / external pull-up	External flash memory is not available	X
Leave unconnected / external pull-up	External pull-down	M25P10 FLASH is available	
External pull-down	Leave unconnected / external pull-up	M95128 EEPROM is available	
External pull-down	External pull-down	M25P40 FLASH is available	

2D Fix Algorithm Configuration
Starting with ROM software version 4.30.3+PAL a 2D fix algorithm has been introduced to reduce the TTFF in critical scenarios (when the number of received satellites is less than 4). In these conditions a less accurate position fix is allowed. If in the final application, the position accuracy is more important than the time to first fix, the 2D algorithm can be disabled. A dedicated GPIO can be configured to disable this feature at the system startup (the same action is available also at system run-time using an NMEA command, see the commands description below).

P1_3	Description	Default
External pull-down	2D Fix algorithm is disabled	
Leave unconnected / external pull-up	2D Fix algorithm is enabled	X

NMEA Output Messages

When the GPS receiver is running a set of messages is presented on the NMEA port. These output strings are compatible with the NMEA 0183 standard and provide all the information needed by a navigation system. At the system startup, using a dedicated configuration pin (see the software configuration section above), it is possible to choose one of the two available set of messages: a basic message list at baudrate of 4800 and the extended message list at baudrate 57600. In the extended message list there are some additional ST proprietary messages (not documented below) that have been included for debug purposes. At system run-time the current message list can be modified using the $PSTMNMEACONFIG or $PSTMRMC commands (see the NMEA software commands section). A description of the available (NMEA 0183) navigation messages is reported in the next pages.

$GPGGA
GPS fix data, message available at startup in the default message list. It can be enabled/disabled with the $PSTMNMEACONFIG
command.
Message rate: 1 Hz
Message: **$GPGGA**,<**PosUTC**>,<**Lat**>,<**LatRef**>,<**Lon**>,<**LonRef**>,<**Qual**>,<**NbSat**>,<**HDOP**>,<**AltMsl**>,M,<**GeoidSep**>,M,<**null**>,<**null**>***checksum<cr><lf>**

Field	Description	Format
PosUTC	Universal time coordinated	Hhmmss.sss
Lat	Latitude	ddmm.mmmmm
LatRef	Latitude direction	'N' or 'S'
Lon	Longitude	dddmm.mmmmm
LonRef	Longitude Direction	'E' or 'W'
Qual	Quality indicator 0 – no fix 1 – GPS fix 2 – Differential GPS fix	X
NbSat	Number of satellites in use	Xx
HDOP	Horizontal dilution of precision	x.x
AltMsl	Antenna altitude above/below main sea level	x.x
M	Meters	'M'
GeoidSep	Geoidal separation	x.x
M	Meters	'M'
Null	–	
Null	–	

$GPGSA
DOP and active satellite list, available at startup in the default message list. It can be enabled/disabled with the $PSTMNMEACONFIG command.
Message rate: 1 Hz
Message: **$GPGSA**,<**Opmode**>,<**FixMode**>,<**Sat**>,<**Sat**>,<**Sat**>,<**Sat**>,<**Sat**>,<**Sat**>,<**Sat**>,<**Sat**>,<**Sat**>,<**Sat**>,<**Sat**>,<**Sat**>,<**PDOP**>,<**HDOP**>,<**VDOP**>***checksum**<**cr**><**lf**>

Field	Description	Format
Opmode	Operating mode (A for automatic switch 2D/3D)	'A'
Fixmode	Fis mode 1 – no fix 2 – 2D fix 3 – 3D fix	X
Sat	ID of the satellite	Xx / null
PDOP	Position dilution of precision (PDOP) in meters	x.x
HDOP	Horizontal dilution of precision (HDOP) in meters	x.x
VDOP	Vertical dilution of precision (VDOP) in meters	x.x

$GPGSV
Satellites in view, available as soon as the first satellite is acquired. It can be enabled/disabled with the $PSTMNMEACONFIG command.
Message rate: 1 Hz
Message: **$GPGSV**,<**MaxMsg**>,<**NumMsg**>,<**NumSats**>,<**SatPrn**>,<**Elev**>,<**Az**>,<**SNR**>,<**SatPrn**>,<**Elev**>,<**Az**>,<**SNR**>,<**SatPrn**>,<**Elev**>,<**Az**>,<**SNR**>,<**SatPrn**>,<**Elev**>,<**Az**>,<**SNR**>***checksum**<**cr**><**lf**>

Field	Description	Format
MaxMsg	Total number of messages	X
NumMsg	Message number	X
NumSats	Total number of staellites in view	Xx
SatPrn	Staellite PRN number	Xx
Elev	Elevation in degrees (90 maximum)	Xx
Az	Azimuth in true degrees (000 to 359)	Xxx
SNR	SNR (C/No) 00 to 99dB, null when tracking	Xx

$GPVTG
Track made good and ground speed, available at startup in the default message list. It can be enabled/disabled with the $PSTMNMEACONFIG command.
Message rate: 1 Hz
Message: **$GPVTG**,<**TrueCourse**>,T, <**MagneticCourse**>,M,<**SpeedKnots**>,N,<**SpeedKmh**>,K***checksum**<**cr**><**lf**>

Field	Description	Format
Course (True)	Measured eading (True)	
T	True	
Course (Magnetic)	Measured heading (magnetic)	
M	Magnetic	
Speed Knots	Measured horizontal speed (knots)	
N	Knots	
SpeedKmh	Measured horizontal speed (km/h)	
K	Kilometers per hour	

$GPRMC
Recommended minimum specific data. Not available at startup in the basic message list at baudrate 4800. Available at startup in the extended message list at baudrate 57600. It can be enabled/disabled with the $PSTMNMEACONFIG command or with a dedicated command $PSTMRMC.
Message rate: 1 Hz
Message: **$GPRMC**,<**PosUTC**>,<**PosStat**>,<**Lat**>,<**LatRef**>,<**Lon**>,<**LonRef**>,<**Spd**>,<**Hdg**>,<**Date**>,<**MagVar**>,<**MagRef**>
***checksum**<**cr**><**lf**>

Field	Description	Format
PosUTC	Universal time coordinated	Hhmmss.sss
PosStat	Position status (A=valid or V=invalid)	'A' or 'V'
Lat	Latitude	Ddmm.mmm
LatRef	Latitude direction	'N' or 'S'
Lon	Longitude	Dddmm.mmm
LonRef	Longitude Direction	'E' or 'W'
Spd	Speed over ground (knots)	x.x
Hdg	Heading track make good (degree true)	x.x
Date	Date	Ddmmyy
MagVar	Magnetic variation (degree)	x.x
MagRef	Magnetic variation	'E' or 'W'

$GPGLL
Geographic Position - Latitude/Longitude, Message available at startup in the default message list. It can be enabled/disabled with the $PSTMNMEACONFIG command.
Message rate: 1 Hz
Message: **$GPGLL**,<**Lat**>,<**LatRef**>,<**Lon**>,<**LonRef**>,<**PosUTC**>,<**Status**>***checksum**<**cr**><**lf**>

Field	Description	Format
Lat	Latitude	Ddmm.mmm
LatRef	Latitude direction	'N' or 'S'
Lon	Longitude	Dddmm.mmm
LonRef	Longitude Direction	'E' or 'W'
PosUTC	Universal time coordinated	Hhmmss.sss
Status	Status A – Data valid, V – Data invalid	'A' or 'V'

NMEA Software Commands

The NMEA software commands are a set of sentences that allow interaction with the module using the NMEA serial port.

The table below summarizes all the software commands implemented in this release:

Syntax	Description	Page
$PSTMINITGPS	Initialize GPS position and time	11
$PSTMCLREPHS	Clear all ephemeris	12
$PSTMCLRALMS	Clear all almanacs	12
$PSTMRMC	Toggle RMC message	12
$PSTMCOLD	Perform COLD start	12
$PSTMWARM	Perform WARM start	12
$PSTMHOT	Perform HOT start	12
$PSTMDUMPEPHEMS	Dump Ephemeris data	12
$PSTMEPHEM	Load Ephemeris data	12
$PSTMNMEAONOFF	Toggle NMEA output	12
$PSTMALMANAC	Load Almanacs data	13
$PSTMDUMPALMANAC	Dump Almanacs data	13
$PSTMGPSRESET	Reset the GPS receiver	13
$PSTMGPSSUSPEND	Suspend GPS	13
$PSTMGPSRESTART	Restart GPS	13
$PSTMTESTRF	Returns satid, frequency offset, phase error, Cn0 of the satellite in channel 0 of the correlator and center frequency offset	13
$PSTMNMEACONFIG	Configures NMEA: baudrate, message list, transmit mode	13
$PSTMTESTROM	Check for ROM data errors	14
$PSTMADCREAD	Read each channel of the ADC	14
$PSTMADCCAL	Execute an ADC Calibration (using two ADC inputs as reference)	15
$PSTMSTBY	Put the system in standby mode	15
$PSTMSTOP	Put the system in stop mode	15
$PSTMSRR	Execute a software reset of the system	15
$PSTMPIOREAD	Read a GPIO port	15
$PSTMPIOWRITE	Write a bit on a GPIO port	16
$PSTMINFOREAD	Send out the software release information	16
$PSTMDEBUGONOFF	Switch ON/OFF the sentences on the debug port	16
$PSTMGETDOP	Get the value of the GPS DOP threshold	16
$PSTMSETDOP	Set the value of the GPS DOP threshold	16
$PSTMGETTRACKTH	Get the value of the tracking threshold	17
$PSTMSETTRACKTH	Set the value of the tracking threshold	17
$PSTMGETMASKANG	Get the mask angle	17
$PSTMSETMASKANG	Set the mask angle	17
$PSTMSENSTART	Start the sensors (ADC and odometer) sampling	17
$PSTMSENEND	Stop the sensors sampling	18
$PSTMGETACQPAR	Get the acquire parameters	18
$PSTMSETACQPAR	Set the acquire parameters	18

$PSTMSBASSTART	Initialize and start all the SBAS routines	19
$PSTMSBASONOFF	Suspend/resume the SBAS activity	19
$PSTMSBASSAT	Change the SBAS satellite	19
$PSTMGETSWVER	Get the GPS library version string	19
$PSTMFDAONOFF	Switch ON/OFF the FDA algorithm	19
$PSTMLPON	Put the CPU speed at 16MHz	19
$PSTMLPOFF	Put the CPU speed at 32MHz	19
$PSTM2DFIXONOFF	Enable/Disable the 2D Fix algorithm	19
$PSTMRFTESTONOFF	Enable/Disable the RF test mode	20
$PSTMCANSAMPON	Start the CAN bus sampling	20
$PSTMCANSAMPOFF	Stop the CAN bus sampling	21
$PSTMGETPPSTC	Get the current value of the RF time correction for PPS	21
$PSTMSETPPSTC	Set the current value of the RF time correction for PPS	21
$PSTMNVMSWAP *	Swap banks in the backup memory	22
$PSTMNVMCREATE *	Create a new directory in the backup memory file system	22
$PSTMNVMWRITE *	Write a data block in a directory	22
$PSTMNVMCOPY *	Read a data block in a directory	22
$PSTMNVMBWRITE *	Write a byte into a data block in a directory	23
$PSTMNVMBREAD *	Read a byte from a data block in a directory	23
$PSTMNVMERASE *	Erase the external serial memory	23
$PSTMNVMITEMERASE *	Erase a directory in the external serial memory	23

* All of the above commands (marked with a star) can be used only if the external serial memory is available.

$PSTMINITGPS

This command initializes the GPS position and time.

Command: **$PSTMINITGPS,<Lat>,<LatRef>,<Lon>,<LonRef>,<Alt>,<Day>,<Month>,<Year>,<Hour>,<Minute>,<Second><cr><lf>**

Parameter	Format	Description
Lat	DDMM.MMM	Latitude
LatRef	'N' or 'S'	Latitude direction
Lon	DDDMM.MMM	Longitude
LonRef	'E' or 'W'	Longitude Direction
Alt	XXXX	Altitude in meters (0000 to 9999)
Day	DD	Day of month (01 to 31)
Month	MM	Month (01 to 12)
Year	YYYY	Year (1996- ...)
Hour	HH	Hour (00 to 23)
Minute	MM	Minute (00 to 59)
Second	SS	Seconds (00 to 59)

Results: The position and time will be initialized, no message will be sent as reply.

$PSTMCLREPHS
This command erases all the ephemeris stored in backup RAM.
Command: **$PSTMCLREPHS<cr><lf>**
Results: All ephemeris, stored in the backup RAM, will be deleted. No message will be sent as reply.

$PSTMCLRALMS
This command erases all the almanacs stored in backup RAM.
Command: **$PSTMCLRALMS<cr><lf>**
Results: All almanacs, stored in the backup RAM, will be deleted. No message will be sent as reply.

$PSTMRMC
Using this command it is possible to add or delete the RMC message from the NMEA output message list.
Command: **$PSTMRMC <cr><lf>**
Results: If the RMC message was present in the NMEA output message list, it will be deleted from the list. If the RMC message wasn't present in the NMEA output message list, it will be added to the list.

$PSTMCOLD
This command erases all the almanacs and ephemeris, stored in the backup RAM and then reboots the system.
Command: **$PSTMCOLD<cr><lf>**
Results: The system reboots.

$PSTMWARM
This command erases all the ephemeris, stored in the backup RAM and then reboots the system.
Command: **$PSTMWARM<cr><lf>**
Results: The system reboots

$PSTMHOT
This command reboots the system without erase any backup data.
Command: **$PSTMHOT<cr><lf>**
Results: The system reboots.

$PSTMDUMPEPHEMS
This command sends out all ephemeris stored in the backup RAM.
Command: **$PSTMDUMPEPHEMS<cr><lf>**
Results: **$PSTMEPHEM,<sat_id>,<N>,<byte1>,.....,<byteN>*<checksum><cr><lf>**

Parameter	Format	Description
sat_id	%j	Satellite number
N	%i	Number of the ephemeris data bytes
byte1	%02x	First byte of the ephemeris data
byteN	%02x	Last byte of the ephemeris data
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMEPHEM
This command allows to the user to load the ephemeris data into backup RAM.
Command: **$PSTMEPHEM,<sat_id>,<N>,<byte1>,.....,<byteN>*<checksum><cr><lf>**

Parameter	Format	Description
sat_id	%j	Satellite number
N	%i	Number of the ephemeris data bytes
byte1	%02x	First byte of the ephemeris data
byteN	%02x	Last byte of the ephemeris data
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

Results: The ephemeris will stored into backup RAM, no message will be sent as reply.

$PSTMNMEAONOFF
This command switches ON or OFF the output NMEA sentences.
Command: **$PSTMNMEAONOFF <cr><lf>**
Results: If the NMEA output message was running, it will be switched OFF. If the NMEA output message was OFF, it will be switched ON.

$PSTMALMANAC
This command allows to the user to load the almanacs data into backup RAM.
Command: **$PSTMALMANAC,<sat_id>,<N>,<byte1>,.....,<byteN>*<checksum><cr><lf>**

Parameter	Format	Description
sat_id	%j	Satellite number
N	%i	Number of the almanac data bytes
byte1	%02x	First byte of the almanac data
byteN	%02x	Last byte of the almanac data
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

Results: The almanac will stored into backup RAM, no message will be sent as reply.

$PSTMDUMPALMANAC
This command sends out all almanacs stored in the backup RAM.
Command: **$PSTMDUMPALMANAC <cr><lf>**
Results: **$PSTMALMANAC,<sat_id>,<N>,<byte1>,.....,<byteN>*<checksum><cr><lf>**

Parameter	Format	Description
sat_id	%j	Satellite number
N	%i	Number of the almanac data bytes
byte1	%02x	First byte of the almanac data
byteN	%02x	Last byte of the almanac data
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMGPSRESET
This command resets the GPS receiver
Command: **$PSTMGPSRESET <cr><lf>**
Results: The GPS receiver will be reset, no message will be sent as reply.

$PSTMGPSSUSPEND
This command suspends the GPS receiver.
Command: **$PSTMGPSSUSPEND <cr><lf>**
Results: The GPS receiver will be suspended, no message will bet sent as reply.

$PSTMGPSRESTART
This command restarts the GPS receiver.
Command: **$PSTMGPSRESTART <cr><lf>**
Results: The GPS receiver will be restarted, no message will bet sent as reply.

$PSTMTESTRF
Returns satid, frequency offset, phase error, Cn0 of the satellite in the channel 0 of the correlator and centre frequency offset
Command: **$PSTMTESTRF<cr><lf>**
Results: **$PSTMTESTRF,<chan_id>,<sat_id>,<frequency_offset>,<phase_noise>,<cn0>,<center_freq>*<checksum><cr><lf>**

Parameter	Format	Description
chan_id	%02d	Correletor channel number
sat_id	%02d	Satellite number
frequency_offset	%05d	Frequency offset
phase_noise	%04d	Phase noise
CN0	%02d	Signal to noise ratio
center_freq	%d	Center frequency offset
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMNMEACONFIG
Configure NMEA: baud rate, message list, transmit mode.
Command: **$PSTMNMEACONFIG,<port>,<baud_rate>,<msg_list>,<transmit_mode><cr><lf>**

Parameter	Format	Description
port	%1d	0 for UART - 1 = Reserved
baud_rate	%07d	UART baudrate
msg_list	%04d	Output message list (see below)
transmit_mode	%1d	0 to transmit on UTC second - 1 to transmit after FIX

Available NMEA output messages

Message	ID
GGA_NMEA_MSG	1
GGA5_NMEA_MSG	2
GSA_NMEA_MSG	4
GSV_NMEA_MSG	8
VTG_NMEA_MSG	16
FIL_NMEA_MSG	32
RMC_NMEA_MSG	64
RF_NMEA_MSG	128
TG_NMEA_MSG	256
TS_NMEA_MSG	512
PA_NMEA_MSG	1024
SAT_NMEA_MSG	2048
RES_NMEA_MSG	4096
TIM_NMEA_MSG	8192

The message list is the sum of the IDs of each message included to the list. Example: to have only GSA and GSV output messages the message list must to be 12.
Results: The selected configuration will be setup, no message will be sent as reply.

$PSTMTESTROM
This command evaluates a CRC of the ROM data and compares it with the one factory stored into ROM.
Command: **$PSTMTESTROM <cr><lf>**
Results: **$PSTMTESTROM,<RomBoot_start_address>,<RomBoot_end_address>,<RomBoot_checksum>,<RomBoot_eval_checksum>,<RomBoot_results>,<GPS_start_address>,<GPS_end_address>,<GPS_checksum>,<GPS_eval_checksum>,<GPS_results>*<checksum><cr><lf>**

Parameter	Format	Description
RomBoot_start_address	%08x	The start address of the ROM Boot Code
RomBoot_end_address	%08x	The end address of the ROM Boot Code
RomBoot_checksum	%04x	The ROM Boot checksum factory stored into ROM
RomBoot_eval_checksum	%04x	The evaluated ROM Boot checksum
RomBoot_results	%d	1 for success - 0 for no success
GPS_start_address	%08x	The start address of the GPS Code
GPS_end_address	%08x	The end address of the GPS Code
GPS_checksum	%04x	The GPS checksum factory stored into ROM
GPS_eval_checksum	%04x	The evaluated GPS checksum
GPS_results	%d	1 for success - 0 for no success
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMADCREAD
Reads each channel of the ADC.
Command: **$PSTMADCREAD<cr><lf>**
Results: **$PSTMADCREAD,<ADC_ch0>,<ADC_ch1>,< ADC_ch2>,< ADC_ch3>*<checksum><cr><lf>**

Parameter	Format	Description
ADC_ch0	%03x	ADC ch0 value (0x0 .. 0x800)
ADC_ch1	%03x	ADC ch0 value (0x0 .. 0x800)
ADC_ch2	%03x	ADC ch0 value (0x0 .. 0x800)
ADC_ch3	%03x	ADC ch0 value (0x0 .. 0x800)
Checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMADCCAL
Execute an ADC Calibration (using two ADC inputs as reference). Sets the sample frequency and evaluates the ADC calibration values. To calibrate the ADC, two reference signals have to be supplied: one on the ADC input 2 and the other on the input 3.
Command: **$PSTMADCCAL,<fs>,<ncycles><cr><lf>**

Parameter	Format	Description
fs	%d	ADC Sample Frequency
ncycles	%d	Number of cycles to perform the calibration.

Results: **$PSTMADCCAL,<ADC_ch2_mean_value>,< ADC_ch3_mean_value >,<fs> ,<ncycles>*<checksum><cr><lf>**

Parameter	Format	Description
ADC_ch2_mean_value	%03x	ADC channel 2 average value (0x0 .. 0x800)
ADC_ch3_mean_value	%03x	ADC channel 3 average value (0x0 .. 0x800)
fs	%d	Sample frequency
ncycles	%d	Number of cycles on which the calibration has been performed.
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMADCCALERROR*<checksum><cr><lf> if no success.

$PSTMSTBY
Put the system in standby mode.
Command: **$PSTMSTBY,<T><cr><lf>**

Parameter	Format	Description
T	%05d	Standby period (seconds). If T is 0 then the system goes in standby mode for undefined time and can be awakened by hardware.

Results: If success the system goes in standby mode for a T time. If T is zero the system stays in standby mode up to the hardware wake-up signal. **$PSTMSTBYERROR*<checksum><cr><lf>** if no success

$PSTMSTOP
Puts the system in stop mode.
Command: **$PSTMSTOP,<T><cr><lf>**

Parameter	Format	Description
T	%05d	Stop period (seconds).

Results: If success the system goes in stop mode for a T time. After T time the system will be reset. **$PSTMSTOPERROR*<checksum><cr><lf>** if no success.

$PSTMSRR
Executes a software reset of the system.
Command: **$PSTMSRR <cr><lf>**
Results: The system will be reset.

$PSTMPIOREAD
Reads a GPIO port
Command: **$PSTMPIOREAD ,<port_number><cr><lf>**

Parameter	Format	Description
port_number	%1d	Port to be read

Results: **$PSTMPIOREAD,<port_number>,<value>*<checksum><cr><lf>**

Parameter	Format	Description
port_number	%1d	Port
value	%08x	Value read on the port (hexadecimal value)
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMPIOWRITE
Writes a bit on a GPIO port.
Command: **$PSTMPIOWRITE,<port_number>,<pin>,<value><cr><lf>**

Parameter	Format	Description
port_number	%1d	Port to be written
pin	%2d	Pin to be written
value	%c	Value to be assigned to the pin : 0 - 1 or 'H' for high impedance

Results: **$PSTMPIOWRITEOK* <checksum><cr><lf>**

$PSTMINFOREAD
Send out the software release information.
Command: **$PSTMINFOREAD<cr><lf>**
Results: **$PSTMVER,<gps_ver>,SW COMMANDS rel. <sw_comm_ver>*<checksum><cr><lf>**

Parameter	Format	Description
gps_ver	%s	GPS software version
sw_comm_ver	%s	Software commands version
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf>characters.

$PSTMSCKONOFF
ON/OFF clock out (the clock value will be: CPU_clock/4 = APB1_clock/2 = 8 MHz with PLL locked)
Command: **$PSTMSCKONOFF <cr><lf>**
Results: If the clock was enabled, it will be disabled. If the clock was disabled, it will be enabled. No message will be sent as reply.

$PSTMDEBUGONOFF
Switch ON/OFF the sentences at the debug port.
Command: **$PSTMDEBUGONOFF <cr><lf>**
Results: If the debug sentences are present on the port, they will be turn OFF. If the debug sentences are not present on the port, they will be turn ON. No message will be sent as reply.

$PSTMGETDOP
Get the current GPS DOP threshold.
Command: **$PSTMGETDOP <cr><lf>**
Results: **$PSTMGETDOP,<3Dpdop>,<3Dhdop>,<3Dvdop>,<2Dpdop>,<2Dhdop>,<2Dvdop>*<checksum><cr><lf>**

Parameter	Format	Description
3Dpdop	%f	3D fix position DOP threshold
3Dhdop	%f	3D fix horizontal DOP threshold
3Dvdop	%f	3D fix vertical DOP threshold
2Dpdop	%f	2D fix position DOP threshold
2Dhdop	%f	2D fix horizontal DOP threshold
2Dvdop	%f	2D fix vertical DOP threshold
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf>characters.

$PSTMSETDOP
Set the value of the GPS DOP threshold
Command: **$PSTMSETDOP,<3Dpdop>,<3Dhdop>,<3Dvdop>,<2Dpdop>,<2Dhdop>,<2Dvdop><cr><lf>**

Parameter	Format	Description
3Dpdop	%f	3D fix position DOP threshold
3Dhdop	%f	3D fix horizontal DOP threshold
3Dvdop	%f	3D fix vertical DOP threshold
2Dpdop	%f	2D fix position DOP threshold
2Dhdop	%f	2D fix horizontal DOP threshold
2Dvdop	%f	2D fix vertical DOP threshold

Results: If success the **$PSTMSETDOPOK* <checksum><cr><lf>** will be sent as reply.
In case of error the **$PSTMSETDOPERROR* <checksum><cr><lf>** message will be sent.

$PSTMGETTRACKTH
Get the current GPS tracking threshold.
Command: **$PSTMGETTRACKTH <cr><lf>**
Results: **$PSTMGETTRACKTH,<track_th>,<th_min>,<th_max>*<checksum><cr><lf>**

Parameter	Format	Description
track_th	%d	Current tracking threshold value.
th_min	%d	Minimum allowed value for the tracking threshold.
th_max	%d	Maximum allowed value for the tracking threshold.
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMSETTRACKTH
Set the value of the tracking threshold.
Command: **$PSTMSETTRACKTH< track_th ><cr><lf>**

Parameter	Format	Description
track_th	%d	Tracking threshold value.

Results: If success the **$PSTMSETTRACKTHOK* <checksum><cr><lf>** will be sent as reply.
In case of error the **$PSTMSETTRACKTHERROR* <checksum><cr><lf>** message will be sent

$PSTMGETMASKANG
Get the current GPS mask angle.
Command: **$PSTMGETMASKANG <cr><lf>**
Results: **$PSTMGETMASKANG,<mask_angle >* <checksum><cr><lf>**

Parameter	Format	Description
mask_angle	%d	Current Mask Angle
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMSETMASKANG
Set the GPS mask angle.
Command: **$PSTMSETMASKANG<mask_angle ><cr><lf>**

Parameter	Format	Description
mask_angle	%d	Mask angle value

Results: If success the **$PSTMSETMASKANGOK* <checksum><cr><lf>** will be sent as reply.
In case of error the **$PSTMSETMASKANGERROR* <checksum><cr><lf>** message will be sent

$PSTMSENSTART
Sets the sampling frequency and starts the periodic sensors sampling. A new NMEA message has added to the current NMEA message list.
Command: **$PSTMSENSTART,<msg_format >,<samp_freq><cr><lf>**

Parameter	Format	Description
msg_format	%x	Specifies which data will be present in the output message. To define the msg format it has to sum the codes of the desired data to include: 0x01 - CPU Time 0x02 - ADC channel 0 0x04 - ADC channel 1 0x08 - ADC channel 2 0x10 - ADC channel 3 0x20 - Odometer count Example1: to include only CPU Time and ADC channel 2, msg_format = 0x9 Example 2: to include all data, msg_format = 0x3F
samp_freq	%d	Sampling frequency (Hz)

Results: If success, a message with sampling data will be add to the NMEA message list. The sensor's data message will have the following pattern (general case with only one sample per row and with all data included):

$PSTMSENSOR,<msg_format>,<samp_numb>,<GPS_fix_cpu_t>, <Sen_cpu_t>,<ADC_ch0>,<ADC_ch1>,<ADC_ch2>, <ADC_ch3>,<Odo_count >,<overrun>*<checksum><cr><lf>

Parameter	Format	Description
msg_format	%02x	Message format (useful for decoding purpose)
Samp_numb	%02x	Number of the samples in the message.
GPS_fix_cpu_t	%06x	Time that the GPS obtained a fix. The relative fix data will be sent in the next (GGA or RMC) NMEA sentence.
Sen_cpu_t	%06x	Time at which the sensors have been sampled. (Same measurement scale of the GPS_fix_cpu_t)
ADC_ch0	%03x	ADC channel 0 data
ADC_ch1	%03x	ADC channel 1 data
ADC_ch2	%03x	ADC channel 2 data
ADC_ch3	%03x	ADC channel 3 data
Odo_count	%04x	Odometer counter
Overrun	%1d	Flags if a sampling overrun occurs.
		1 : overrun occurred, some data has been lost between the current and the previous sensor message. 0 : no overrun.
Checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

In case of error the **$PSTMSENSTARTERROR* <checksum><cr><lf>** message will be sent

$PSTMSENEND
Stops the periodic sensor's sampling and removes the sensor's data message from the NMEA message list.
Command: **$PSTMSENEND<cr><lf>**
Results: No message will be sent as reply.

$PSTMGETACQPAR
Get the current GPS acquiring parameters.
Command: **$PSTMGETACQPAR<cr><lf>**
*Results:***$PSTMGETACQPAR,<acquire_threshold>,<acquire_wide_frequency_range>,<acquire_prediction_integration_time>,<acquire_prediction_wide_integration_time>,<acquire_all_sats_prediction_integration_time>,<acquire_all_sats_integration_time>,<acquire_full_search_integration_time>*<checksum><cr><lf>**

Parameter	Format	Description
acquire_threshold	%d	Acquire threshold
acquire_wide_frequency_range	%d	Acquire wide frequency range
acquire_prediction_integration_time	%d	Acquire prediction integration time
acquire_prediction_wide_integration_time	%d	Acquire prediction wide integration time
acquire_all_sats_prediction_integration_time	%d	Acquire all sats prediction integration time
acquire_all_sats_integration_time	%d	Acquire all sats integration time
acquire_full_search_integration_time	%d	Acquire full search integration time
Checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMSETACQPAR
Set the GPS acquiring parameters.
Command: **$PSTMSETACQPAR,<acquire_threshold>,<acquire_wide_frequency_range>,<acquire_prediction_integration_time>,<acquire_prediction_wide_integration_time>,<acquire_all_sats_prediction_integration_time>,<acquire_all_sats_integration_time>,<acquire_full_search_integration_time><cr><lf>**

Parameter	Format	Description
acquire_threshold	%d	Acquire threshold
acquire_wide_frequency_range	%d	Acquire wide frequency range
acquire_prediction_integration_time	%d	Acquire prediction integration time
acquire_prediction_wide_integration_time	%d	Acquire prediction wide integration time
acquire_all_sats_prediction_integration_time	%d	Acquire all sats prediction integration time
acquire_all_sats_integration_time	%d	Acquire all sats integration time
acquire_full_search_integration_time	%d	Acquire full search integration time

$PSTMSBASSTART
Initialize and start all SBAS routines. The SBAS software has been included in this ROM image but has been disabled by default. For this reason at each system reset the SBAS activity will be OFF and needs the $PSTMSBASSTART command to start.

Command: **$PSTMSBASSTART<cr><lf>**
Results: The SBAS version string will be sent as reply.

$PSTMSBASONOFF
Suspend / resume the SBAS software execution. When the SBAS is running (after a $PSTMSBASSTART command) it can be suspended and resumed using the $PSTMSBASONOFF command.
Command: **$PSTMSBASONOFF<cr><lf>**
Results: If SBAS was running it will be suspended, if it was suspended it will start to run.

$PSTMSBASSAT
Change the SBAS satellite.
*Command:***$PSTMSBASSAT,<prn><cr><lf>**

Parameter	Format	Description
prn	%d	Satellite PRN (Range: from 120 to 138 and 0)

Results: If the SBAS satellite is available in the above range, the software starts tracking. If the parameter is zero, the system automatically searches for the SBAS satellite available in the user region.

$PSTMGETSWVER
Get the version string of the GPS library embedded in the ROM code.
Command: **$PSTMGETSWVER <cr><lf>**
Results: The version string will be sent out. As example: $PSTMVER,GPSLIB_04.30.03+PAL ARM-Jul 28 2006 12:23:26

$PSTMFDAONOFF
Toggles the Fault Detection Algorithm ON/OFF. The FDA can be used to overcome drift due to interference.
Command: **$PSTMFDAONOFF<cr><lf>**
Results: If the FDA was enabled it will be disabled. If the FDA was disabled it will be enabled. On power-up the default mode is OFF. After each power cycle the FDA must be toggled back to the ON state.

$PSTMLPON
Put the CPU speed at 16MHz. This command is available starting with ROM version 4.30.3.
Command: **$PSTMLPON<cr><lf>**
Results: The CPU speed will change to 16MHz. NOTE: this command could be useful to reduce dynamically the power consumption, when the CPU load is not critical.

$PSTMLPOFF
Put the CPU speed at 32MHz. This command is available starting with ROM version 4.30.3.
Command: **$PSTMLPOFF<cr><lf>**
Results: The CPU speed will change to 32MHz. NOTE: this command could be useful to restore the standard CPU frequency at the end of a low power period.

$PSTM2DFIXONOFF
Enable/Disable the 2D Fix algorithm. This command is available starting with ROM version 4.30.3.
Command: **$PSTM2DFIXONOFF<status><cr><lf>**

Parameter	Format	Desccription
status	%d	0: Disable 2D Fix algorithm 1: Enable 2D Fix algorithm

Results: According with the input parameter the 2D fix algorithm will be enabled or disabled.
NOTE: to see the effects on the TTFF the GPS engine should be reset using the $PSTMGPSRESET command.

$PSTMRFTESTONOFF
Enable/Disable the RF test mode. This command is available starting with ROM version 4.30.3.
Command: **$PSTMRFTESTONOFF,<sat_id>,<status><cr><lf>**

Parameter	Format	Description
sat_id	%d	Satellite's ID that will be searched during testing
status	%d	0: Disable RF test 1: Enable RF test

Results: According with the input parameter the RF test will be started or stopped. When the test is running, the GPS will try to acquire only the provided satellite's ID. As soon as it has been acquired, it will be tracked on the channel zero. To get the satellite's parameters during tracking, the $PSTMTESTRF can be used. NOTE: the GPS engine must be reset ($PSTMGPSRESET) as soon as the $PSTMRFTESTONOFF command has been sent.

$PSTMCANSAMPON
Configures the CAN bus parameters (message ID, bit rate etc,) and starts to catch the message data. A new NMEA message will be added to the current NMEA message list. This command is available starting with ROM version 4.30.3.
Command: **PSTMCANSAMPON,<msg_format>,<can_msg_id>,<mask>,<length>,<type>,<tseg_1>,<tesg_2>,<sjw>,<brp><cr><lf>**

Parameter	Format	Description
msg_format	%x	Specifies which data will be present in the output message To define the msg format it has to sum the codes of the desired data to include: 0x01 – CPU Time 0x02 – CAN_ID_MSG 0x04 – CAN_LEN_MSG 0x08 – CAN_B0_MSG 0x10 – CAN_B1_MSG 0x20 – CAN_B2_MSG 0x40 – CAN_B3_MSG 0x80 – CAN_B4_MSG 0x100 – CAN_B5_MSG 0x200 – CAN_B6_MSG 0x400 – CAN_B7_MSG Example 1: to include only CPU Time, CAN_ID_MSG and CAN_B0_MSG msg_format = 0xB Example 2: to include all data, msg_format = 0x7FF
can_msg_id	%d	ID of the message to be received
Mask	%d	CAN RX message mask
length	%d	CAN RX message data length
type	%d	CAN RX message type 0: standard message type 1: extended message type
tseg_1	%d	Time segment before the sample point. Valid values for TSeg1 are [1..15]
tseg_2	%d	Time segment after sample point. Valid values for TSeg2 are [0..7]
sjw	%d	Synchronization Jump Width. Valid programmed values are [0..3]
brp	%d	Baud Rate Prescaler. The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Baud Rate Prescaler are [0..63]

Results: If success, a message with CAN data will be added to the NMEA message list; the syntax is described below:
$PSTMCANMSG,<msg_format>,<samp_numb>,<GPS_fix_cpu_time>, <msg_cpu_time>,<can_msg_id>,<msg_data_length>, <msg_data_b0>,<msg_data_b1>,<msg_data_b2>,<msg_data_b3>,<msg_data_b4>,<msg_data_b5>,<msg_data_b6>,<msg_data_b7>,<overrun>*<checksum><cr><lf>

Parameter	Format	Description
msg_format	%02x	Mesage format (useful for decoding purpose) as described in the previous table
samp_numb	%02x	Number of teh can data samples available in the NMEA message string
GPS_fix_cpu_t	%06x	Time at which the GPS obtained a fix. The relative fix data will be sent in the next (GGA or RMC) NMEA sentence. This parameter can be used to synchronize the data coming from the can bus and the GPS fix time (e.g. deadreckoning applications)
msg_cpu_time	%06x	Time at which the can message has been received. (Same measurement scale and time reference of the GPS_fix_cpu_t)
can_msg_id	%08x	Received CAN message ID
msg_data_length	%1x	Received CAN message data length
msg_data-B0	%02x	Received Can message data byte 0.
msg_data-B1	%02x	Received Can message data byte 1.
msg_data-B2	%02x	Received Can message data byte 2.
msg_data-B3	%02x	Received Can message data byte 3.
msg_data-B4	%02x	Received Can message data byte 4.
msg_data-B5	%02x	Received Can message data byte 5.
msg_data-B6	%02x	Received Can message data byte 6.
msg_data-B7	%02x	Received Can message data byte 7.
overrun	%1d	Flags if a buffer overrun occurs. 1: overrun occurred; some data has been lost between the current and the previous NMEA CAN message. 0: no overrun.
cheksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

In case of error the **$PSTMCANSAMPONERROR* <checksum><cr><lf>** message will be sent.

$PSTMCANSAMPOFF
Stops the CAN data receiving and removes the corresponding NMEA output message from the message list. This command is available starting with ROM version 4.30.3.
Command: **$PSTMCANSAMPOFF<cr><lf>**
Results: The CAN message ID receiving will be stopped and the $PSTMCANMSG NMEA message will be removed from the message list.

$PSTMGETPPSTC
Get the current value of the RF time correction for PPS. This parameter has been included to compensate for any additional error on the pulse per second signal introduced by the antenna cable length and by the RF signal conditioning circuit. This command is available starting with ROM version 4.30.3.
Command: **$PSTMGETPPSTC<cr><lf>**
Results: **$PSTMGETPPSTC, <time_correction>*<checksum><cr><lf>**

Parameter	Format	Description
time_correction	%e	Time Correction in seconds (Default value is 500E-9 sec).
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

$PSTMSETPPSTC
Set the current value of the RF time correction for PPS. This parameter has been included to compensate for any additional error on the pulse per second signal introduced by the antenna cable length and by the RF signal conditioning circuit. This command is available starting with ROM version 4.30.3.
Command: **$PSTMSETPPSTC, <time_correction><cr><lf>**

Parameter	Format	Description
time_correction	%e	Time correction in seconds.

Results: If success: **$PSTMSETPPSTCOK,<time_correction>*<checksum><cr><lf>**

Parameter	Format	Description
time_correction	%e	Time correction in seconds (Default value is 500E–9 sec).
checksum	%02x	Checksum of the message bytes without *<checksum><cr><lf> characters.

If no success the message:
$PSTMSETPPSTCERROR*<checksum><cr><lf> will be sent back.

$PSTMNVMSWAP
Swap banks in the backup memory. This command is available only if the external serial non-volatile memory is present. This command is available starting with ROM version 4.30.3.
Command: **$PSTMNVMSWAP<cr><lf>**
Results: The non-volatile backup memory banks will be swapped, no message will be sent as reply.

$PSTMNVMCREATE
Create a new directory in the backup memory file system. This command is available only if the external serial non-volatile memory is present. This command is available starting with ROM version 4.30.3.
Command: **$PSTMNVMCREATE,<nvm_id>,<nvm_copies>,<nvm_size><cr><lf>**

Parameter	Format	Description
nvm_id	%d	NVM directory ID. It identifies the folder for a specific type of data. Allowed values are: [128...255]. The IDs between 0 and 127 are reserved for the GPS.
nvm_copies	%d	Number of copies of the same data allowed in the specified directory. The number of copies must be chosen according with the data block size and the frequency at which this data will be updated, in order to guarantee a flash erase period (at least) of 2 hours (Every data update generates a copy; the flash will be erased as soon as the directory is full).
nvm_size	%d	Size of the data block to be stored in the specified directory.

Results: If success the **$PSTMNVMCREATEOK*<checksum><cr><lf>** message will be sent back. If no success (e.g. the memory is full or the available space is not enough to allocate the new directory) the **$PSTMNVMCREATEERROR*<checksum><cr><lf>** message will be sent back.

$PSTMNVMWRITE
Write a data block in a directory. This command is available only if the external serial non-volatile memory is present. This command is available starting with ROM version 4.30.3.
Command: **$PSTMNVMWRITE,<nvm_id>,<type>,<data_array><cr><lf>**

Parameter	Format	Description
nvm_id	%d	NVM directory ID. It identifies the folder for a specific type of data. Allowed values are: [128...255]. The IDs between 0 and 127 are reserved for the GPS.
type	%d	Specifies the format of the data_array parameter: 0: Binary – the bytes in the data_array are the bytes to be stored. 1: Hexadecimal – the bytes in the data_array are the hexadecimal values of the data to be stored.
data_array	%d	Is the array of bytes to be stored (data block)

Results: If success the **$PSTMNVMWRITEOK*<checksum><cr><lf>** message will be sent back.
If no success the **$PSTMNVMWRITEERROR*<checksum><cr> <lf>** message will be sent back.

$PSTMNVMCOPY
Read a data block in a directory. This command is available only if the external serial non-volatile memory is present. This command is available starting with ROM version 4.30.3.
Command: **$PSTMNVMCOPY,<nvm_id>,<type>cr><lf>**

Parameter	Format	Description
nvm_id	%d	NVM directory ID. It identifies the folder for a specific type of data. Allowed values are: [128...255]. The IDs between 0 and 127 are reserved for the GPS.
type	%d	Specifies the format of the returned data: 0: Binary – the bytes in the returned messages are the read bytes. 1: Hexadecimal – the bytes in the returned message are the hexadecimal values of the read data.

Results: If success the **$PSTMNVMCOPY,<data_array>*<checksum><cr><lf>** message will be sent back.

Parameter	Format	Description
data_array	%d	It is the array of bytes, read in the specified directory, according with the specified format type.

If no success the **$PSTMNVMCOPYERROR*<checksum><cr><lf>** message will be returned.

$PSTMNVMBWRITE
Write a byte into a data block in a directory. This command can be used to store additional data in a directory without replaceing what has been written in the past (e.g. log periodically positions and data). In this case, during the directory creation, the data block size is the maximum space available and it will be filled accessing to the internal bytes. The user is responsible for the internal bytes address management. This command is available only if the external serial non-volatile memory is present. This command is available starting with ROM version 4.30.3.
Command: **$PSTMNVMBWRITE,<nvm_id>,<add_offset>,<n_bytes>,<type>,<data_array><cr><lf>**

Parameter	Format	Description
nvm_id	%d	NVM directory ID. It identifies the folder for a specific type of data. Allowed values are: [128...255]. The IDs between 0 and 127 are reserved for the GPS.
add_offset	%d	It is the offset in bytes from the base address of data block in the directory.
n_bytes	%d	Number of bytes to be stored.
type	%d	Specifies the format of the data_array parameter: 0: Binary – the bytes in the data_array are the bytes to be stored. 1: Hexadecimal – the bytes in the data_array are the hexadecimal values of the data to be stored.
data_array	%d	The array of bytes to be stored (data block).

Results: If success the **$PSTMNVMBWRITEOK*<checksum><cr><lf>** message will be sent back.
If no success the **$PSTMNVMBWRITEERROR*<checksum><cr><lf>** message will be sent back.

$PSTMNVMBREAD
Read a byte from a data block in a directory. This command can be used to read each byte (or a set of bytes) of the data block (e.g. read positions and data logged in the past). The user is responsible of the internal bytes address management. This command is available only if the external serial non-volatile memory is present. This command is available starting with ROM version 4.30.3.
Command: **$PSTMNVMBREAD,<nvm_id>,<add_offset>,<n_bytes>,<type>cr><lf>**

Parameter	Format	Description
nvm_id	%d	NVM directory ID. It identifies the folder for a specific type of data. Allowed values are: [128...255]. The IDs between 0 and 127 are reserved for the GPS.
add_offset	%d	It is the offset in bytes from the base address of data block in the directory.
n_bytes	%d	Number of bytes to be read.
type	%d	Specifies the format of the returned data: 0: Binary – the bytes in the returned messages are the read bytes. 1: Hexadecimal – the bytes in the returned message are the hexadecimal values of the read data.

Results: If success the **$PSTMNVMBREAD,<data_array>*<checksum><cr><lf>** message will be sent back.

Parameter	Format	Description
data_array	%d	It is the data array of bytes, read in the specified directory, from the specified offset, according with the specified format type.

If no success the **$PSTMNVMBREADERROR*<checksum><cr><lf>** message will be returned.

$PSTMNVMERASE
Erase the external serial memory. This command is available only if the external serial non-volatile memory is present. This command is available starting with ROM version 4.30.3.
Command: **$PSTMNVMERASE<cr><lf>**
Results: If success the **$PSTMNVMERASEOK*<checksum><cr><lf>** message will be sent back.
If no success the **$PSTMNVMERASEERROR*<checksum><cr><lf>** message will be sent back.

$PSTMNVMITEMERASE
Erase only a directory in the external serial memory. This command can be used to erase the data stored in the specified directory in order to store new data in the same directory. This command is available only if the external serial non-volatile memory is present. This command is available starting with ROM version 4.30.3.
Command: **$PSTMNVMITEMERASE<cr><lf>**
Results: If success the **$PSTMNVMITEMERASEOK*<checksum><cr><lf>** message will be sent back. If no success the **$PSTMNVMITEMERASEERROR*<checksum><cr> <lf>** message will be sent back.

External Serial Memory

GPS Data Backup in the External Memory

In order to have a short time to fix (TTF) at every system startup, the GPS software can benefit from some vital data (like almanac, ephemeris, nco frequency offset, etc.) stored in a backup memory and updated up to the last software execution. For these reasons the consistency on the GPS backup data must be preserved even if the system is turned OFF or it is in a low power mode. To accomplish this feature, the STA2056 (Palinuro) chip has a small backup SRAM inside, 4k bytes in size and powered by the backup power supply. This means that the final system needs to also have a battery onboard to supply the backup power rails.

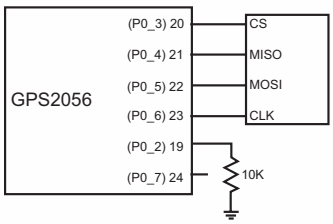
In addition to the internal backed-up SRAM memory, ROM software version 4.30.03+PAL has added the ability to store the GPS data (and also some other user data) into an external serial (SPI) non volatile memory. The internal backed-up SRAM management is always available, so the user is free to decide if they want to use additional external memory to the final system layout. Due to the serial connection the access to the non volatile memory (read/write) is slower than accessing the internal SRAM. To overcome any problems when the GPS software must access to read and/or store data into backup memory, the software has been structured to keep the same interface between the GPS software and the internal SRAM backup data and a dedicated task (low priority task) to update in the "background" external memory, while reading the new values from the internal one. At every system startup, if the internal SRAM data are lost, the backup data will be restored from the serial memory (it takes less than 2 seconds). In this way the software is exactly the same (with the external memory or without it) and the user needs only to connect the external memory with the right GPIO configurations.

Adding the external non volatile memory to the final system layout adds an extra feature: the ability to store any other kind of data together with the GPS data backup (e.g. user application data). The management of memory area is based on a file system that provides all the features to create new directories and to read/store data; it is also responsible to erase a sector as soon as it becomes full to keep safe all the last updates in each directory. A set of NMEA commands are available to access, with a host controller, the external non-volatile memory (refer to the "NMEA Software Commands" section above for further details).

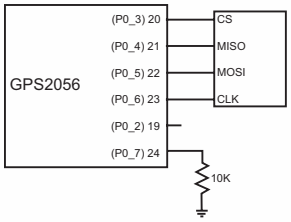
Supported Memories and Schematics

Three different types of memory are supported by the ROM software version 4.30.03+PAL. Below are the hardware connections and configuration schematics for the supported type of memory. If no external memory is available on board, the two configuration pins must be unconnected or connected to a pull-up resistor.

1) EEPROM memory (M95128) to address a low cost solution, with 16 K bytes of memory that is available for the GPS data backup and few additional user data.



2) 1 Mbit FLASH memory (M25P10) to have a long lifetime and to have more memory space to use as additional backup memory by the user application.



3) 4 Mbit FLASH memory (M25P40) to have much more space for the user needs.