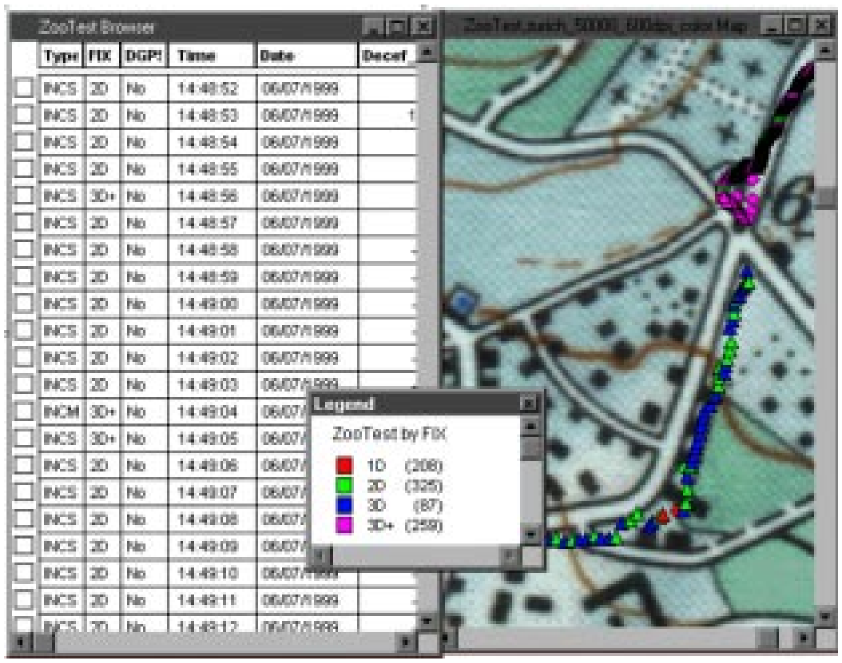
Logging Option on µ-blox GPS receivers Application Note

µ-blox ag
Gloriastrasse 35
CH-8092 Zürich
Switzerland
http://www.u-blox.ch

2nd September 1999

Abstract

The µ-blox GPS receivers are available with an optional software function that enables user defined position- and time-logging in the on-board flash memory. This application note describes the concept of the logging implementation, the protocol extensions and some real world examples.
It also describes the u-logger utility. This utility demonstrates the possibilities of this extension. It may be used to configure the filters or to download or erase the data stored in the flash memory.



Type	FIX	DGPS	Time	Date	Decel
INCS	2D	No	14:48:52	06/07/1999	
INCS	2D	No	14:48:53	06/07/1999	1
INCS	2D	No	14:48:54	06/07/1999	
INCS	2D	No	14:48:55	06/07/1999	
INCS	3D+	No	14:48:56	06/07/1999	
INCS	2D	No	14:48:57	06/07/1999	
INCS	2D	No	14:48:58	06/07/1999	-
INCS	2D	No	14:48:59	06/07/1999	-
INCS	2D	No	14:49:00	06/07/1999	-
INCS	2D	No	14:49:01	06/07/1999	-
INCS	2D	No	14:49:02	06/07/1999	-
INCS	2D	No	14:49:03	06/07/1999	-
INCM	3D+	No	14:49:04	06/07/	
INCS	3D+	No	14:49:05	06/07/	
INCS	2D	No	14:49:06	06/07/	
INCS	2D	No	14:49:07	06/07/	
INCS	2D	No	14:49:08	06/07/	
INCS	2D	No	14:49:09	06/07/	
INCS	2D	No	14:49:10	06/07/	
INCS	2D	No	14:49:11	06/07/1999	-
INCS	2D	No	14:49:12	06/07/1999	

ZooTest by FIX

🟥	1D	(208)
🟩	2D	(325)
🟦	3D	(87)
🟪	3D+	(259)

Contents

1 Introduction

The Logging Option is a software extension to the standard firmware. The basic idea is to store data in the on-board flash memory in addition to the transmission over the serial port.

This powerful software extension opens up a wide range of applications.

- Vehicle tracking
- Road pricing systems
- Automatic project accounting for field personnel
- Behavioural studies of animals

The Logging Option is designed for maximum stored data and maximum flexibility. A differential storage technology is used to store the data in the flash memory.
There are two main logging functionalities: position fix and GPIO[1] logging. Both can be configured separately and are totally independed from each other. Beside the traditional position fix logging, it can also be configured to store changes on the GPIO Pins. For example, a temperature sensor or a on/off event could be logged.

The information stored includes:

- GPS Timestamp (WNO, TOW), not UTC corrected, resolution 1 [s]
- Position (ECEF), resolution 1 [m]
- Velocity, range 0 - 1023 [km/h], resolution 1 [km/h]
- Number of satellites used for navigation
- DGPS used
- GPIO signal levels, GPIO pins 0 to 11

The user may also configure the filter parameters to suit his application. The Logging Option defines a new set of SiRF binary protocol messages. The messages can be used for configuring the filter parameters. This enables user defined position-, time- and velocity-logging in then on-board memory. Download and erasing of the flash is also supported by this protocol extension. Although the logging option is designed as an extension to the SiRF binary protocol, data is also stored while in NMEA mode.

2 Storage Format

Data is stored in different storage records depending on the type and values. The three most significant bits (bits 15 to 13) determine the type of the storage record. In addition to the basic types, a flexible storage record, the so called escape type storage record, is defined for future logging applications.

[1] GPIO: General Purpose I/Os

Bits[15:13]	Type	Size[a]	Description
111	NONE	1	No or unwritten data
100	FIX_FULL	9	Position fix data, full storage format
010	FIX_INCL	5	Position fix data, large incremental storage format
000	FIX_INCM	4	Position fix data, medium incremental storage format
110	FIX_INCS	3	Position fix data, small incremental storage format
101	GPIO_FULL	3	GPIO data, full storage format
011	GPIO_INC	2	GPIO data, incremental storage format
001	ESCAPE	var[b]	used for future logging applications

Table 1: Storage Types

[a] Size in words (1 word = 2 bytes)
[b] Size is detemined by the second byte of the storage record

2.1 Empty Storage Record

If the first three bits are all '1', then the word is considered as unwritten data and is skipped therefore.

The following table lists the layout of the data in memory.

Word \Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	x[a]												

Table 2: Storage Type NONE

[a] reserved

2.2 Position Fix Storage Records

The logging option has different kinds of position fix storage records. The following parameters are stored:

WNO	10 Bit	Week Number (in GPS notation), $[week]$
TOW	20 Bit	Time of Week (in GPS notation), $[s]$
DTOW	16 Bit	Difference between last and current TOW, $[s]$
ECEF_X/Y/Z	32 Bit[a]	ECEF X/Y/Z Coordinate, $[m]$
DECEF_X/Y/Z	5/10/16[b] Bit[a]	Difference between last and current ECEF X/Y/Z Coordinate $[m]$
V	10 Bit	Velocity[c] $[km/h]$
SV	2 Bit	Number of satellites. See table 4 for meaning.
DGPS	1 Bit	Differential GPS (1 = used, 0 = not used).

Table 3: Position fix logging parameters

[a] signed integer
[b] size depends on storage format
[c] absolute speed or speed over ground, depending on the flags in the configuration.

SV[1:0]	Symbol	Description
0 0	1D	less than 3 satellites used or Dead Reckoning
0 1	2D	3 satellites used
1 0	3D	4 or more satellites used
1 1	3D+	5 or more satellites used and fix is validated

Table 4: SV bits meaning

The following tables list the layout of the data in memory.

Word \Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	SV[1:0]		DGPS	V[9:0]									
1	WNO[9:0]										x[a]		TOW[19:16]			
2	TOW[15:0]															
3	ECEF_X[31:16]															
4	ECEF_X[15:0]															
5	ECEF_Y[31:16]															
6	ECEF_Y[15:0]															
7	ECEF_Z[31:16]															
8	ECEF_Z[15:0]															

Table 5: Storage Type FIX_FULL

[a] reserved

Word \Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	SV[1:0]		DGPS	V[9:0]									
1	DTOW[15:0]															
2	DECEF_X[15:0]															
3	DECEF_Y[15:0]															
4	DECEF_Z[15:0]															

Table 6: Storage Type FIX_INCL

Word \Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	SV[1:0]		DGPS	V[9:0]									
1	DTOW[15:0]															
2	DECEF_Z[5:0]						DECEF_X[9:0]									
3	x[a]		DECEF_Z[9:6]				DECEF_Y[9:0]									

Table 7: Storage Type FIX_INCM

[a] reserved

Word \Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	SV[1:0]		DGPS	V[9:0]									
1	DTOW[15:0]															
2	x[a]	DECEF_Z[4:0]					DECEF_Y[4:0]					DECEF_X[4:0]				

Table 8: Storage Type FIX_INCS

[a] reserved

2.3 GPIO Storage Records

The logging option has different kinds of GPIO storage records. The following parameters are stored:

WNO	10 Bit	Week Number (in GPS notation), $[week]$
TOW	20 Bit	Time of Week (in GPS notation), $[s]$
DTOW	16 Bit	Difference between last and current TOW, $[s]$
GPIO	12 Bit	Values of the GPIO Pins 11 to 0

Table 9: GPIO logging parameters

The following tables list the layout of the data in memory.

Word \Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	x[a]	GPIO[11:0]											
1	WNO[9:0]										x[a]		TOW[19:16]			
2	TOW[15:0]															

Table 10: Storage Type GPIO_FULL

[a] reserved

Word \Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	x[a]	GPIO[11:0]											
1	DTOW[15:0]															

Table 11: Storage Type GPIO_INC

[a] reserved

2.4 ESCAPE Type Storage Records

ESCAPE type storage records are defined for future use of the logging firmware. They have a flexible format. Its size can be determined by the second byte (SIZE field).

For example diagnostics strings may be written to the flash memory as ESCAPE_TYPE `0x1F` with a zero terminated string as the payload.

The following table lists the layout of the data in memory.

Word \Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	ESCAPE_TYPE					SIZE							
1	…															
…	…															
…	…															
SIZE	…															

Table 12: Storage Type ESCAPE

2.5 Decompressing a downloaded memory block

The algorithm decompresses a previously downloaded memory block.

```
 -- Download the data from the GPS receiver flash.
 -- Use the logging protocol extension.
 -- Get the first storage record into Data.

 -- Decompress all storage records while we have Data.
WHILE (Data)

    -- Now decode the storage record
    -- Get the Type bits of the storage record.

   IF (Type = EMPTY) THEN

       -- No data, just skip this word.

   ELSE IF (Type = FIX_FULL) THEN

       -- Save WNO and DTOW fields as the position fix logging time stamp
       -- Save the position, speed and the other mode flags

   ELSE IF ((Type = FIX_INCL) OR (Type = FIX_INCM) OR (Type = FIX_INCS)) THEN

       -- Add the DTOW field to the last position fix logging time stamp.
       -- Add the DECEF to the last position.
       -- Save the speed and the other mode flags.

   ELSE IF (Type = GPIO_FULL) THEN

       -- Save WNO and DTOW fields as the GPIO logging time stamp.
       -- Save the GPIO values.

   ELSE IF (Type = GPIO_INC) THEN

       -- Add the DTOW field to the last GPIO logging time stamp.
       -- Save the gpio values.

   ELSE IF (Type = ESCAPE) THEN
```

```
        -- Handle the additional ESCAPE type storage records.

  END IF

  -- The size of each storage record can be determined from the type
  -- and if it is an escape type from the additional size field.
  -- Get the next storage record into Data.

END WHILE
```

3 Logging Algorithms

The following section describes when the Logging otpion is logging data to the flash memory and how the filter parameters influences it.

3.1 Position Fix Logging Algorithm

The position fix logging algorithm stores the following information:

- Timestamp of stored position, Resolution 1 $[s]$.
- Velocity. Range 0 …1023 $[km/h]$, Resolution 1 $[km/h]$.
- Position. Full ECEF Position. Resolution 1 $[m]$.
- Number of SVs (<3, 3, 4 or >4 SVs).
- DGPS used.

The algorithm stores according to the following pseudo code, which is called whenever a position fix is done:

```
-- Calculate the difference between now and the last storage time.
Time_Diff = Current.Time - Last.FixTime

IF (Time_Diff > T_Min) THEN

  -- Calculate the difference between here and the last stored position.
  Distance_Diff = ABS(Current.Position - Last.Position)

  -- Only store if it passes the time and distance filter.
  IF ((Distance_Diff > D_delta) OR (Time_Diff > T_Max)) THEN

    IF ((Distance_Diff > 32767) OR (Time_Diff > 65535)) THEN

      -- Store a FIX_FULL record to the flash.

    ELSE IF (Distance_Diff > 511) THEN

      -- Store a FIX_INCL record to the flash.
    ELSE IF (Distance_Diff > 15) THEN

        -- Store a FIX_INCM record to the flash.

    ELSE

        -- Store a FIX_INCS record to the flash.

    END IF

    -- Backup storage time and position.
    Last.FixTime = Current.Time
    Last.Position = Current.Position

  END IF

END IF
```

3.2 GPIO Logging Algorithm

The GPIO logging algorithm stores the following information:

- Timestamp of stored position, Resolution 1 $[s]$.
- Values of all GPIO pins 0 to 11.

The algorithm stores according to the following pseudo code, which is called once every second:

```
-- Calculate the difference between now and the last storage time.
T_Diff = Current.Time - Last.GPIOTime

IF (T_Diff > T_Min) THEN

  -- Only store if the GPIOs have changed (& is the bitwise and operator).
  IF ((Current.GPIOValue & Mask) ≠ (Last.GPIOValue & Mask)) THEN

    IF (T_Diff > 65535) THEN
        -- Store a GPIO_FULL record  to the flash.
    ELSE
        -- Store a GPIO_INC record  to the flash.
    END IF

    -- Backup the storage time and GPIO values.
    Last.GPIOTime = Current.Time
    Last.GPIOValue = Current.GPIOValue

  END IF

END IF
```

4 Protocol Extension

The logging protocol extension can be used with SiRF Binary protocol only. Please refer to the µ-blox protocol documentation for a specification of the transport- and verification layer. This document describes the payload portion of the extended SiRF binary protocol, only.

The following Input[2] - and Output messages are supported:

MID	Message
`0xB6`	LogSectorErase (responses with LogSectorEraseEnd)
`0xB8`	LogRead (responses with LogData)
`0xBA`	LogPollSectorInfo (responses with LogSectorInfo)
`0xBB`	LogPollInfo (responses with LogInfo)
`0xBC`	LogSetConfig
`0xBD`	LogPollConfig (responses with LogConfig)
`0xBE`	LogFixSetConfig
`0xBF`	LogFixPollConfig (responses with LogFixConfig)
`0xC0`	LogGPIOSetConfig
`0xC1`	LogGPIOPollConfig (responses with LogGPIOConfig)

Table 13: Input Messages

MID	Message
`0x79`	LogData (response to LogRead)
`0x7A`	LogSectorInfo (response to LogPollSectorInfo)
`0x7B`	LogSectorEraseEnd (response to LogSectorErase)
`0x7C`	LogInfo (response to LogPollInfo)
`0x7D`	LogConfig (response to LogPollConfig)
`0x7E`	LogFixConfig (response to LogFixPollConfig)
`0x7F`	LogGPIOConfig (response to LogGPIOPollConfig)

Table 14: Output Messages

4.1 Input Messages

4.1.1 LogSectorErase

This message causes the receiver to erase a specific flash sector. The receiver disables flash writing. After erasing the receiver returns a message of type LogSectorEraseEnd (0x7B). After erasing all sectors you must reset the receiver. Send the Navigation Initialisation Message (MID = 0x80).

Parameter	Type	Description
MID	U8	`0xB6`
Sector	U8	Sector number
		Payload: 2 Byte

Table 15: LogSectorErase Message

[2] Input as seen from the receiver, i.e. from the Host PC to the µ-blox receiver

4.1.2 LogRead

This message requests 512 bytes of stored and compressed log data. The module returns a message of type LogData (0x79)

Parameter	Type	Description
MID	U8	`0xB8`
Address	U32	Address from which data should be returned.
		Payload: 5 Bytes

Table 16: LogRead Message

4.1.3 LogPollSectorInfo

This message requests information on a specific sector of the flash memory. The receiver returns a message of type LogSectorInfo (0x7A).

Parameter	Type	Description
MID	U8	`0xBA`
Sector	U8	Sector number
		Payload: 2 Bytes

Table 17: LogPollSectorInfo Message

4.1.4 LogPollInfo

This message requests information on flash memory and logging space. The receiver returns a message of type LogInfo (0x7C).

Parameter	Type	Description
MID	U8	`0xBB`
		Payload: 1 Byte

Table 18: LogPollInfo Message

4.1.5 LogSetConfig

This message sets the general logging configuration.

Parameter	Type	Description
MID	U8	`0xBC`
Flags	U16	Logging Flags. See table 20 for meaning
		Payload: 3 Bytes

Table 19: LogSetConfig Message

Bit #	Meaning
Bit 1	Logging Debug Messages (1=Enabled, 0=Disabled)
Bit 8	Flash 1PPS LED when logging (1=Enabled, 0=Disabled)

Table 20: LogSetConfig.Flags Bitmap

4.1.6 LogPollConfig

This message requests the general logging configuration. The receiver returns a message of type LogConfig (0x7D).

Parameter	Type	Description
MID	U8	`0xBD`
		Payload: 1 Byte

Table 21: LogPollConfig Message

4.1.7 LogFixSetConfig

This message sets the position fix logging configuration.

Parameter	Type	Description
MID	U8	`0xBE`
Flags	U16	Fix Logging Flags. See table 23 for meaning
T_min $[s]$	U16	Logging Period. This gives the minimum period with which a position is stored, if moved more than D_delta meters during that period. 0=always
T_max $[s]$	U16	Logging Period Max. This gives the time with which a position record is stored, regardless of the D_Delta parameter. 0=disabled
D_delta $[m]$	U16	Distance Filter. The Position is stored if moved D_delta or more and the time elapsed is greater than T_min. 0=disabled.
		Payload: 9 Bytes

Table 22: LogFixSetConfig Message

Bit #	Meaning
Bit 0	Position Fix Logging Control (0=Disabled, 1=Enabled)
Bit 2	Output Measured Navigation on Serial Port (SiRF Binary Message 2) while Logging (1=Don't Output, 0=Output)
Bit 3	Log Filter for 4SV Solution (1=Log only if 4 or more SV used)
Bit 6	Speed Format (0=3D Speed, 1=2D Speed, Speed over ground)

Table 23: LogFixSetConfig.Flags Bitmap

4.1.8 LogFixPollConfig

This message requests the position fix logging configuration. The receiver returns a message of type LogFixConfig (0x7E).

Parameter	Type	Description
MID	U8	`0xBF`
		Payload: 1 Byte

Table 24: LogFixPollConfig Message

4.1.9 LogGPIOSetConfig

This message sets the GPIO logging configuration.

Parameter	Type	Description
MID	U8	`0xCA`
Flags	U16	GPIO Logging Flags. See table 26 for meaning
T_min $[s]$	U16	Logging Period. This gives the maximum frequency with which GPIO changes are stored.
Mask	U16[a]	Pin Mask, Any modification applies to the here masked pins only. (1 = Change, 0 = Leave)
Direction	U16[a]	Direction Bitmask (1=Input, 0=Output)
Value	U16[a]	Value Bitmask (1 = High, 0 = Low)
Check	U16[a]	Check Bitmask (1=Log if Pin changes)
		Payload: 11 Bytes

Table 25: LogGPIOSetConfig Message

[a]Bitmask: the bit X represents GPIO X, bits 12 to 15 are not used

Bit #	Meaning
Bit 0	GPIO Logging Control (0=Disabled, 1=Enabled)

Table 26: LogGPIOSetConfig.Flags Bitmap

4.1.10 LogGPIOPollConfig

This message requests the GPIO logging configuration. The receiver returns a message of type LogGPIOConfig (0x7F).

Parameter	Type	Description
MID	U8	`0xCB`
		Payload: 1 Byte

Table 27: LogGPIOPollConfig Message

4.2 Output Messages

4.2.1 LogData

This message is sent as a response to a LogRead message.

Parameter	Type	Description
MID	U8	`0x79`
Start	U32	Start address of this 512 Byte Block.
Data[256]	256 x U16	Compressed Data[a]
		Payload: 517 Bytes

Table 28: LogData Message

[a]See section 2 for a description of the compressed data structures

4.2.2 LogSectorInfo

This message is sent as a response to a LogPollSectorInfo message.

Parameter	Type	Description
MID	U8	`0x7A`
Sector	U8	sector number
Flags	U16	(reserved)
Size	U32	Size of this sector in bytes.
Base	U32	Start address of this sector. To be used with `LogRead`.
Free	U32	Number of bytes available in this sector.
		Payload: 16 Bytes

Table 29: LogSectorInfo Message

4.2.3 LogSectorEraseEnd

This message is sent as a response to a LogSectorErase message.

Parameter	Type	Description
MID	U8	`0x7B`
Sector	U8	Sector number
		Payload: 2 Byte

Table 30: LogSectorEraseEnd Message

4.2.4 LogInfo

This message is sent as a response to a LogPollInfo message.

Parameter	Type	Description
MID	U8	`0x7C`
S_First	U8	Index of first free sector (zero-based)
S_Last	U8	Index of last free sector (zero-based)
A_First	U32	First address in the logging space.
A_Last	U32	Last address in the logging space.
A_Start	U32	Start address of the used logging space.
Size	U32	Size of the used logging space.
		Payload: 19 Byte

Table 31: LogInfo Message

4.2.5 LogConfig

This message is sent as a response to a LogPollConfig message.

Parameter	Type	Description
MID	U8	`0x7D`
Flags	U16	See LogSetConfig message.
		Payload: 3 Byte

Table 32: LogConfig Message

4.2.6 LogFixConfig

This message is sent as a response to a LogFixPollConfig message.

Parameter	Type	Description
MID	U8	`0x7E`
Flags	U16	See LogFixSetConfig message.
T_min $[s]$	U16	See LogFixSetConfig message.
T_max $[s]$	U16	See LogFixSetConfig message.
D_delta $[m]$	U16	See LogFixSetConfig message.
		Payload: 9 Byte

Table 33: LogFixConfig Message

4.2.7 LogGPIOConfig

This message is sent as a response to a LogGPIOPollConfig message.

Parameter	Type	Description
MID	U8	`0x7F`
Flags	U16	See LogGPIOSetConfig message.
T_min $[s]$	U16	See LogGPIOSetConfig message.
Mask	U16	See LogGPIOSetConfig message.
Direction	U16	See LogGPIOSetConfig message.
Value	U16	See LogGPIOSetConfig message.
Check	U16	See LogGPIOSetConfig message.
		Payload: 13 Bytes

Table 34: LogGPIOConfig Message

4.3 Transfering logged data using the extended protocol

```
 -- get information on the flash structure.

 -- send the message LogPollInfo and receive the message LogInfo.

 -- allocate the required memory to store the data.
Data = MEMALLOC(A_Current - A_First)

 -- now download all the data.

Address = A_First
WHILE (Address < A_Current)

    -- now download the block from address Address.

    -- send the message LogRead and receive the message LogData.

    -- copy the received block to its position in Data.

    -- calculate the starting address of the next block to download.
  Address = Address + 512

END WHILE

 -- decompress Data.
 -- use the algorithm given in section 2.5.
```

5 Transfering logged data using u-logger.exe

The µ-logger is a simple program to demonstrate and evaluate the logging capabilities of the µ-blox GPS logging firmware and the protocol extension. It allows you to configure the module and to download or erase the logged data.

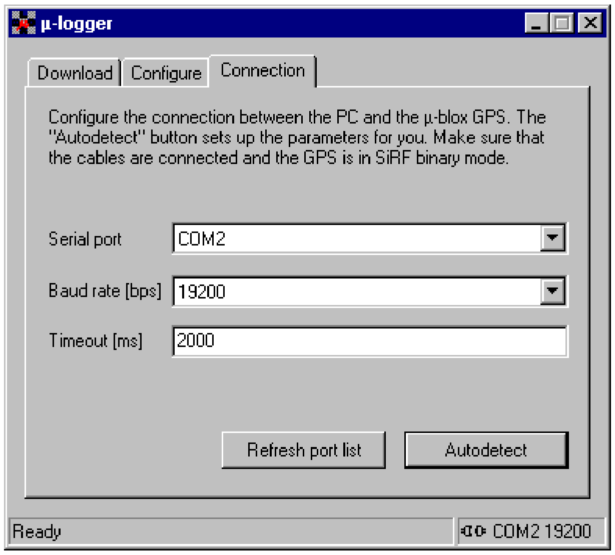
The logged data may be stored in various formats that can be postprocessed using third party programs.

The program runs on IBM-compatible PCs running Microsoft Windows 95/98 or Microsoft Windows NT 4. It needs an unused serial port where the µ-blox GPS receiver is connected to.

The status bar shows the actual connection and its current status. It also indicates the step and progress of the current operation. The user may abort any operation by pressing the *Cancel* button.

Note – *All programm settings are stored in the File* `u-blox.ini` *under the section µ-logger in your windows directory.*

5.1 Setup the communication



Configure the connection between the PC and the µ-blox GPS. The "Autodetect" button sets up the parameters for you. Make sure that the cables are connected and the GPS is in SiRF binary mode.

Serial port: COM2
Baud rate [bps]: 19200
Timeout [ms]: 2000

Ready — COM2 19200

1. Select the communications port from the *serial port* pulldown menu to which the serial cable has been connected on your PC. If the port is not in the port list, an other program is still connected with your GPS. Close the connection and then press the *Refresh port list* button. The port should now appear in the port list.

2. Select the appropriate *baud rate* from the baud rate pull down menu (default baud rate is 19200).
3. You may want to change the *Timeout* for the serial communication (default is 2000 ms).

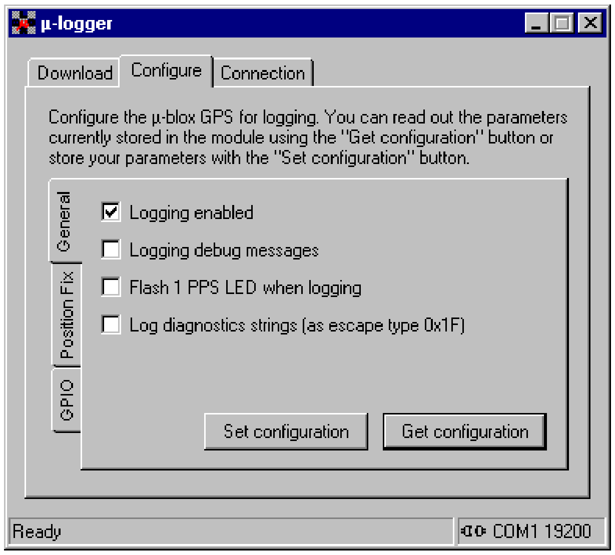
Note – *The* Auto detect *button checks all serial ports and baud rates for a a connected GPS. If this does not detect the GPS, make sure that it uses the SiRF binary mode protocol and the cables are properly connected.*

5.2 Configure the GPS logging parameters

The *Set configuration* stores the parameters selected in the dialog box to the module. The *Get configuration* button reads out the configuration parameters from the module and fills them into the dialogbox.

Note – *You must set or get the configuration for each subtab separately.*

5.2.1 General logging configuration



Configure the µ-blox GPS for logging. You can read out the parameters currently stored in the module using the "Get configuration" button or store your parameters with the "Set configuration" button.

- [✓] Logging enabled
- [ ] Logging debug messages
- [ ] Flash 1 PPS LED when logging
- [ ] Log diagnostics strings (as escape type 0x1F)

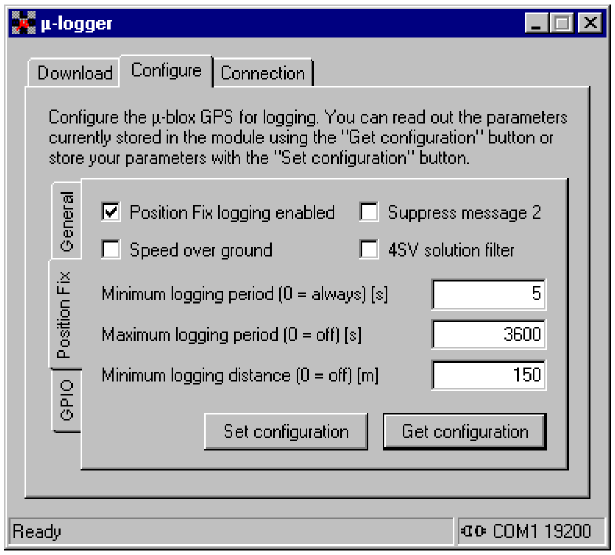
Ready
COM1 19200

- Check the *Logging enabled* checkbox to enable logging. This flag allows you to enable or disable all the logging algorythms quickly. (default is on)

Note – *If the* Logging enabled *flag is not set, the GPS receiver will not store any record regardless of other flags.*

- Check *Logging debug messages* if the logging debug messages should be transmitted. (default is off)
- Check the *Flash 1 PPS LED when logging*, if you want the LED indicating a logging cycle. (default is off)
- Check *Log diagnostics strings* if you want the module to store important events like "Reset" into the flash. (default is off)

5.2.2 Position fix logging configuration



Configure the µ-blox GPS for logging. You can read out the parameters currently stored in the module using the "Get configuration" button or store your parameters with the "Set configuration" button.

[✓] Position Fix logging enabled
[ ] Suppress message 2
[ ] Speed over ground
[ ] 4SV solution filter

Minimum logging period (0 = always) [s]	5
Maximum logging period (0 = off) [s]	3600
Minimum logging distance (0 = off) [m]	150

Ready
COM1 19200

- Check the *Position fix logging enabled* checkbox to enable logging. (default is on) **Make sure the *Logging enabled* flag in the *General* subtab is also enabled.**
- Check *Suppress message 2* if the Measured Navigation message (MID 2) of the SiRF binary protocol is not transmitted. (default sends Msg)
- Check the *Speed over ground*, if you only want to log the speed over ground as velocity instead of the absolute speed of all three dimensions. (default is absolute speed)

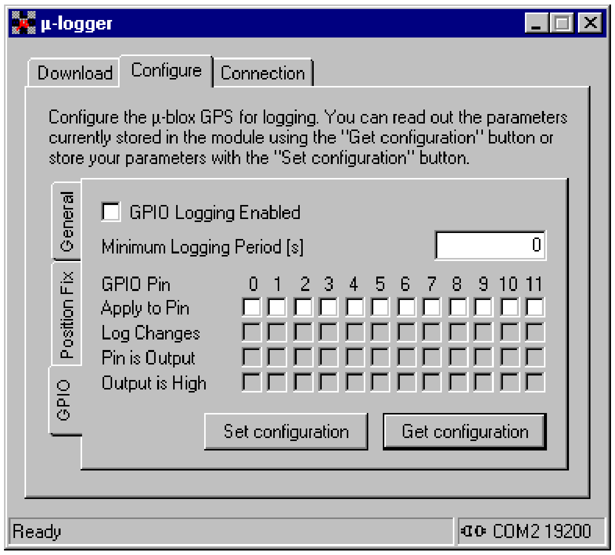
- Check the *4SV solution filter*, if you only want to log when 3D position fixes are calculated. (default is off)
- Set the *Minimum logging period* (T_min). This value gives the maximum frequency with which position information should be stored, if moved more than the *Minimum Distance between logging* value during that period. Set this value to 0 to log always. (default is 5$[s]$)
- Set the *Maximum logging period* (T_max). This gives the time with which a position information is stored, regardless of the *Minimum Distance between logging* value. Set this value to 0 to disable it. (default is 3600$[s]$)
- Set the *Minimum Distance between logging* (D_delta). This is the distance filter. Current position information is stored if moved more than this value since last stored value. Set this value to 0 to disable it. (default is 150$[m]$)

5.2.3 GPIO logging configuratuion

GPIO logging is more difficult to configure since it interacts with the hardware connected to the gpio pins. The GPS has no knowledge of what components are connected, and therefore has no way to find out which signals are driven by external components and which can be driven by the GPS itself.

Note – *You have to make sure that the GPS module and its counterpart don't drive the signal line at the same time. Improper use can lead to permanent damage to the system!*

The GPS is configured by bitmasks which are represented by the checkboxes in the dialog box.



Configure the µ-blox GPS for logging. You can read out the parameters currently stored in the module using the "Get configuration" button or store your parameters with the "Set configuration" button.

☐ GPIO Logging Enabled

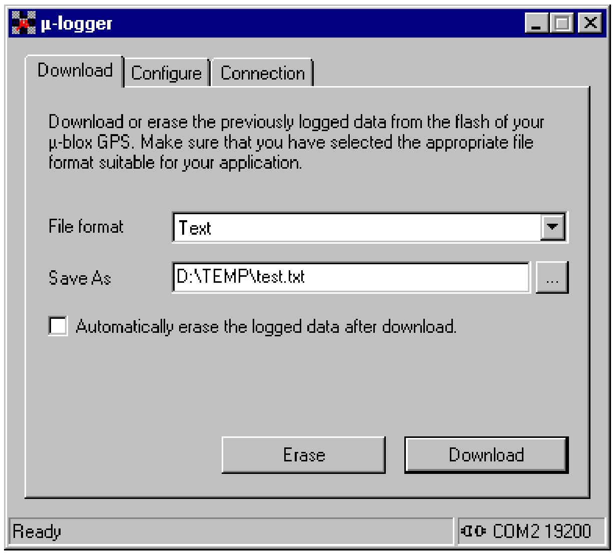
Minimum Logging Period [s] 0

GPIO Pin	0	1	2	3	4	5	6	7	8	9	10	11
Apply to Pin	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Log Changes	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Pin is Output	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Output is High	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Ready COM2 19200

- Check the *GPIO logging enabled* checkbox to enable logging. (default is off) **Make sure the *Logging enabled* flag in the *General* subtab is also enabled.**
- Set the *Minimum logging period* (T_min). This value gives the maximum frequency with which GPIO information should be stored. (default is $0[s]$)
- The *Apply to pin* (Change) check boxes. The configuration parameters of the other bitmasks apply only to the GPIOs that are checked here.
- The *Log changes* (Check) check boxes. All GPIOs that are checked are observed once every second, changes in the signal are logged to the flash. (default checks nothing)
- The *Pin is output* (Direction) check boxes. GPIOs that are checked here are configured as output instead of input. (default is all inputs)
- The *Output is high* (Value) check boxes. The signal level of GPIOs that are checked here is set to high instead of low. (default is all outputs low)

5.3 Download or erase the logged data



Download or erase the previously logged data from the flash of your µ-blox GPS. Make sure that you have selected the appropriate file format suitable for your application.

File format: Text

Save As: D:\TEMP\test.txt

☐ Automatically erase the logged data after download.

1. Select the appropriate file format. Please note, that the NMEA formats contain only information of the Position Fix storage records.

Type	Description
Binary	Compressed data as in the flash, saved binary, big-endian byte order
NMEA	Decompressed data, saved as NMEA GLL, RMC, GGA and VTG messages
NMEA GLL	Decompressed data, saved as NMEA GLL messages
NMEA RMC	Decompressed data, saved as NMEA RMC messages
NMEA GGA	Decompressed data, saved as NMEA GGA messages
NMEA VTG	Decompressed data, saved as NMEA VTG messages
Text	Decompressed data, saved as ASCII text, All data
Text FIX	Decompressed data, saved as tabular ASCII text, Fix data only
Text GPIO	Decompressed data, saved as tabular ASCII text, GPIO data only

Table 35: Download file formats

2. Select the path and filename, where you want to store the File.
   You can press the button next to the edit box to select the path and filename.
3. If you want to *Automatically erase the logged data after download* check the check box.
4. Click the *Download* button to download and store the data on the disk.

***Note** – The* Erase *button deletes the flash area reserved for data logging.*

6 Fix Logging Performance Example

Assume that the Fix logging parameters are T_min = 5$[s]$, T_max = 3600$[s]$ and D_Delta = 150$[m]$. The logging time depends on the memory available for logging and on how the receiver is moved.

The µ-blox GPS receivers have 1'024 kBytes (8 MBits) of flash memory of which 704 kBytes may be used for logging.[3] The following equation calculates how long data can be logged.

$$\text{logging time} = \text{time between storage}\frac{\text{free flash memory}}{\text{size per storage record}}$$

- The worst case is that we have to store a logging record in FIX_FULL format every second. The logging time will around 25 hours.
- Let's assume that we are constantly traveling with 50 km/h (14 m/s). The time between storage will be 11 seconds. Since we moved about 150 meters the data is most probably stored in the FIX_INCM storage record format. The calculated logging time is about 22 days.

[3]Old generation receivers have 512 kBytes (4 MBytes) of flash memory of which 192 kBytes are free.

7 Real Example

This example shows a short ride (for about 15 minutes) with a car through downtown Zurich. Table 36 lists the configuration used in this example. No differential GPS was connected.

Parameter	Value
Flags	0x01
T_min	2 $[s]$
T_max	3600 $[s]$
D_delta	50 $[m]$

Table 36: Module fix logging configuration

After the ride the memory used logging in the flash was 946 bytes and 117 positions are stored. The file size of the downloaded data depends on the format selected.

The following lines show an extraxt of a hexadecimal dump of the file saved in binary format.

```
000001a0: .... 1831 0003 73e3 03fe 1836 0003 7be1
000001b0: 03fc 1835 0003 83df 03ff 1836 0003 87e2
000001c0: 03fc 1833 0003 83e0 03fd 1838 0003 7be2
000001d0: 03fe 182e 0003 7fe4 0001 1828 0003 73e6
000001e0: 03fe 1819 0005 77e4 0003 182c 0004 73dd
000001f0: 03fd 1816 0004 53e1 0000 1824 0006 bc10
00000200: 3c22 1821 0003 b412 3c12 1813 0004 b80e
00000210: 3c17 1816 0006 c00c 3c21 181e 0005 c80c
00000220: 3c1f 1827 0003 c40c 3c1a 1827 0003 b80b
00000230: 3c1d 1824 0003 c00a 3c1d 1818 0004 c40b
00000240: 3c1d ....
```

The following lines show an extraxt of the data saved as a tabular text.

```
FIX_Type  FIX DGPS WNO  TOW    DTOW  Time     Date       Decef_X Decef_Y Decef_Z Ecef_X   Ecef_Y   Ecef_Z   Speed Longitude   Latitude   Altitude
....
FIX_INCM  3D+ No    999 120495     3 09:28:15 03/01/1999     -29      -2      28  4278899   643178  4670897    49    8.548354  47.380770      418
FIX_INCM  3D+ No    999 120498     3 09:28:18 03/01/1999     -31      -4      30  4278868   643174  4670927    54    8.548363  47.381159      419
FIX_INCM  3D+ No    999 120501     3 09:28:21 03/01/1999     -33      -1      32  4278835   643173  4670959    53    8.548415  47.381571      420
FIX_INCM  3D+ No    999 120504     3 09:28:24 03/01/1999     -30      -4      33  4278805   643169  4670992    54    8.548421  47.381972      424
FIX_INCM  3D+ No    999 120507     3 09:28:27 03/01/1999     -32      -3      32  4278773   643166  4671024    51    8.548445  47.382380      426
FIX_INCM  3D+ No    999 120510     3 09:28:30 03/01/1999     -30      -2      30  4278743   643164  4671054    56    8.548478  47.382761      427
FIX_INCM  3D+ No    999 120513     3 09:28:33 03/01/1999     -28       1      31  4278715   643165  4671085    46    8.548546  47.383132      432
FIX_INCM  3D+ No    999 120516     3 09:28:36 03/01/1999     -26      -2      28  4278689   643163  4671113    40    8.548571  47.383474      435
FIX_INCM  3D+ No    999 120521     5 09:28:41 03/01/1999     -28       3      29  4278661   643166  4671142    25    8.548666  47.383831      437
FIX_INCM  3D+ No    999 120525     4 09:28:45 03/01/1999     -35      -3      28  4278626   643163  4671170    44    8.548695  47.384234      434
FIX_INCM  3D+ No    999 120529     4 09:28:49 03/01/1999     -31       0      20  4278595   643163  4671190    22    8.548756  47.384558      428
FIX_INCM  3D+ No    999 120535     6 09:28:55 03/01/1999      16      34     -17  4278611   643197  4671173    36    8.549170  47.384317      430
FIX_INCM  3D+ No    999 120538     3 09:28:58 03/01/1999      18      18     -19  4278629   643215  4671154    33    8.549370  47.384065      430
FIX_INCM  3D+ No    999 120542     4 09:29:02 03/01/1999      14      23     -18  4278643   643238  4671136    19    8.549644  47.383842      428
FIX_INCM  3D+ No    999 120548     6 09:29:08 03/01/1999      12      33     -16  4278655   643271  4671120    22    8.550052  47.383633      428
FIX_INCM  3D+ No    999 120553     5 09:29:13 03/01/1999      12      31     -14  4278667   643302  4671106    30    8.550435  47.383439      429
FIX_INCM  3D+ No    999 120556     3 09:29:16 03/01/1999      12      26     -15  4278679   643328  4671091    39    8.550752  47.383243      428
FIX_INCM  3D+ No    999 120559     3 09:29:19 03/01/1999      11      29     -18  4278690   643357  4671073    39    8.551110  47.383033      425
FIX_INCM  3D+ No    999 120562     3 09:29:22 03/01/1999      10      29     -16  4278700   643386  4671057    36    8.551470  47.382842      423
FIX_INCM  3D+ No    999 120566     4 09:29:26 03/01/1999      11      29     -15  4278711   643415  4671042    24    8.551828  47.382650      422
....
```

The following lines show an extraxt of the data saved as NMEA GLL messages.

```
....
$GPGLL,4722.8462,N,00832.9013,E,092815.000,A,*16
$GPGLL,4722.8696,N,00832.9018,E,092818.000,A,*19
$GPGLL,4722.8943,N,00832.9049,E,092821.000,A,*10
$GPGLL,4722.9183,N,00832.9053,E,092824.000,A,*1B
$GPGLL,4722.9428,N,00832.9067,E,092827.000,A,*1B
```

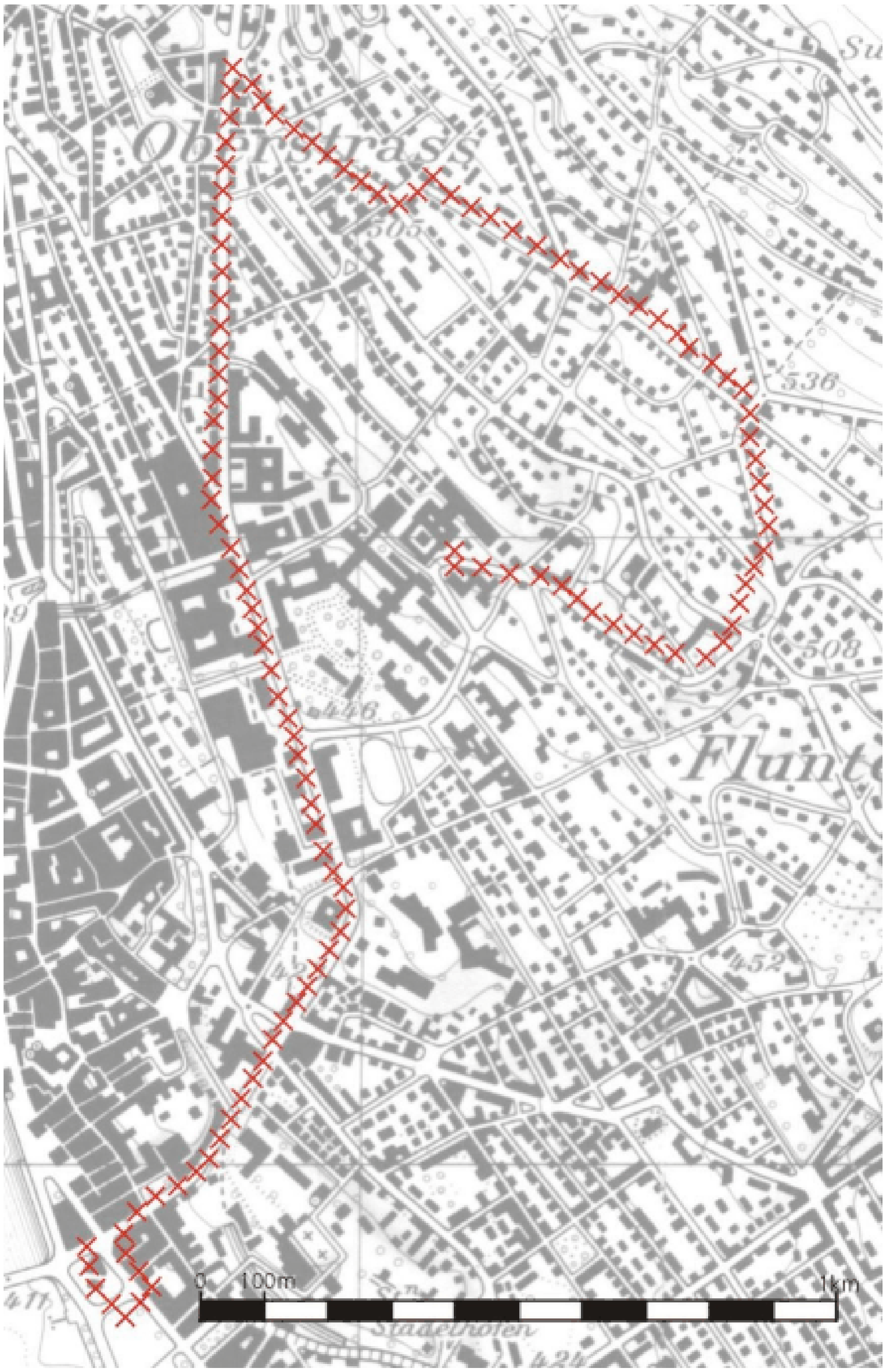


Figure 1: Short ride through downtown Zurich

```
$GPGLL,4722.9656,N,00832.9087,E,092830.000,A,*18
$GPGLL,4722.9879,N,00832.9128,E,092833.000,A,*1C
$GPGLL,4723.0085,N,00832.9143,E,092836.000,A,*17
$GPGLL,4723.0299,N,00832.9199,E,092841.000,A,*1F
$GPGLL,4723.0540,N,00832.9217,E,092845.000,A,*1D
$GPGLL,4723.0735,N,00832.9254,E,092849.000,A,*16
$GPGLL,4723.0590,N,00832.9502,E,092855.000,A,*12
$GPGLL,4723.0439,N,00832.9622,E,092858.000,A,*1C
$GPGLL,4723.0305,N,00832.9786,E,092902.000,A,*15
$GPGLL,4723.0180,N,00833.0031,E,092908.000,A,*13
$GPGLL,4723.0063,N,00833.0261,E,092913.000,A,*12
$GPGLL,4722.9946,N,00833.0451,E,092916.000,A,*14
$GPGLL,4722.9820,N,00833.0666,E,092919.000,A,*1C
$GPGLL,4722.9705,N,00833.0882,E,092922.000,A,*18
$GPGLL,4722.9590,N,00833.1097,E,092926.000,A,*1F
....
```

The following lines show an extraxt of the data saved as NMEA RMC messages.

```
....
$GPRMC,092815.000,A,4722.8462,N,00832.9013,E,0.00,,010399,,*11
$GPRMC,092818.000,A,4722.8696,N,00832.9018,E,0.00,,010399,,*1E
$GPRMC,092821.000,A,4722.8943,N,00832.9049,E,0.00,,010399,,*17
$GPRMC,092824.000,A,4722.9183,N,00832.9053,E,0.00,,010399,,*1C
$GPRMC,092827.000,A,4722.9428,N,00832.9067,E,0.00,,010399,,*1C
$GPRMC,092830.000,A,4722.9656,N,00832.9087,E,0.00,,010399,,*1F
$GPRMC,092833.000,A,4722.9879,N,00832.9128,E,0.00,,010399,,*1B
$GPRMC,092836.000,A,4723.0085,N,00832.9143,E,0.00,,010399,,*10
$GPRMC,092841.000,A,4723.0299,N,00832.9199,E,0.00,,010399,,*18
$GPRMC,092845.000,A,4723.0540,N,00832.9217,E,0.00,,010399,,*1A
$GPRMC,092849.000,A,4723.0735,N,00832.9254,E,0.00,,010399,,*11
$GPRMC,092855.000,A,4723.0590,N,00832.9502,E,0.00,,010399,,*15
$GPRMC,092858.000,A,4723.0439,N,00832.9622,E,0.00,,010399,,*1B
$GPRMC,092902.000,A,4723.0305,N,00832.9786,E,0.00,,010399,,*12
$GPRMC,092908.000,A,4723.0180,N,00833.0031,E,0.00,,010399,,*14
$GPRMC,092913.000,A,4723.0063,N,00833.0261,E,0.00,,010399,,*15
$GPRMC,092916.000,A,4722.9946,N,00833.0451,E,0.00,,010399,,*13
$GPRMC,092919.000,A,4722.9820,N,00833.0666,E,0.00,,010399,,*1B
$GPRMC,092922.000,A,4722.9705,N,00833.0882,E,0.00,,010399,,*1F
$GPRMC,092926.000,A,4722.9590,N,00833.1097,E,0.00,,010399,,*18
....
```

8 Related Documents

Related documents can be found at `http://www.u-blox.ch/restricted`.

Revision History		
Date	**Revision**	**Changes**
Jul. 30, 1999	1.00	GPIO, Escape logging, u-logger v1.00
Feb. 26, 1999	0.93	New, extended logging format, u-logger v0.20
Feb. 12, 1999	0.92	Added u-logger Section, Pseudocodes and Examples
Jan. 5, 1999	0.91	Added Storage Format tables
Dec. 24, 1998	0.90	Initial Version