WAVEON 606

Application Programming Guide

Revision V1.2

Disclaimer

June 2005

The information in this specification is subject to change without notice.

INFOWAVE PTE LTD SHALL NOT BE LIABLE FOR TECHNICAL OR EDITORIAL ERRORS OR OMISSIONS CONTAINED HEREIN, NOR FOR INCIDENTAL OR CONSQUENTIAL DAMAGES RESULTING FROM THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL. THIS INFORMATION IS PROVIDED "AS IS" AND INFOWAVE DISCLAIMS ANY WARRANTIES, EXPRESS, IMPILED OR STATUTORY, AND EXPRESSLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR PARTICULAR PURPOSE, GOOD TITLE AND AGAINST INFRINGEMENT.

600 Sin Ming Avenue 4th Floor Singapore 575733

Revision Record

Revision	Date	Author	Description
1.0	27/02/06	XinBin Shu	Updates: ○ Initial version and its cumulative updates;
1.1	08/05/06	XinBin Shu	Updates: ○ Modifications in $PSVL, $RPIT, $GPOU, $DSCG, $PSWD; ○ Added Quick Install Guide and Appendix E.
1.2	11/07/06	XinBin Shu	Updates: ○ Added new AT commands $SPIN and $ADRP. ○ Moved some setting from $RPIT and $DSCG to $ADRP. ○ Added more error code into $STER. ○ Added Appendix F. ○ Added EVENT_OVER_SPEED into $EVMN.

Content

Overview

Waveon606 supports a set of proprietary Wavelet User Commands (WUC), which is used to manage device operation. This document is a guide on all WUC commands. It is a reference for application programmer.

This document is divided into sections as shown in table below. If it is found conflicted with device output, please follow device output.

NO.	Section	Contents
1	"Introduction"	A general image of Wavelet device.
2	"Command General Format"	Format in general
3	"Command List"	Describe commands in detail.

1. Introduction

1.1 Scope of this document

1.2 Definitions and Abbreviations

Definitions	Meaning
U8_TYPE	8-bit unsigned char
BOOLEAN	0 or 1
U16_TYPE	16-bit unsigned integer
U32_TYPE	32-bit unsigned integer
INT	32-bit integer
FLOAT	32-bit float
DOUBLE	64-bit float
STRING	String end with NULL character.
<VALUE>	VALUE is compulsory.
[VALUE]	VALUE is optional.
Device	Waveon606
Backend	The remote facility used to manage Device, E.g., Remote Server, Mobile Phone etc.
WUC	Wavelet User Command
EMPTY_STR	An empty string.
<CR>	Carry Return
<LF>	Line Feed
MIN_RPT_TIME	10 seconds. Minimum GPS report time.
WORD	32-bit value.
TX / RX	Transmit / Receive

1.3 Typical Deployment

A typical deployment scenario is given in Fig1.1.

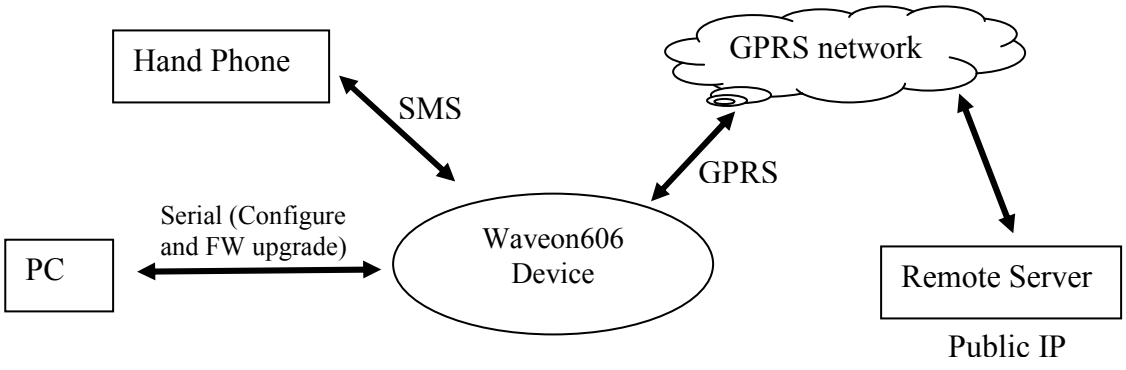
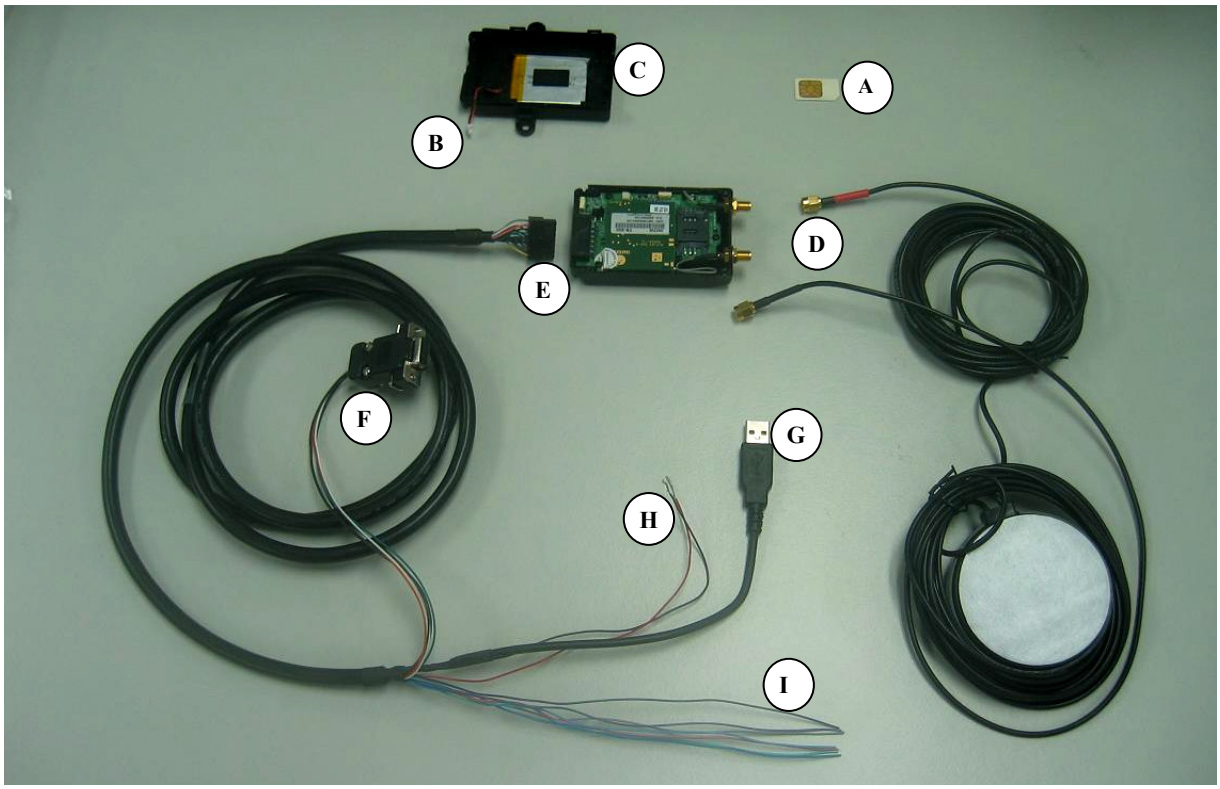


Fig1.1. Typical scenario of Wavelet deployment

1.4 Quick Installation Guide

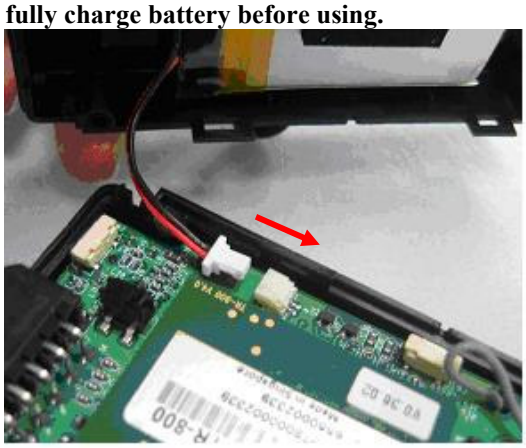
Blow pictures illustrate how to assemble waveon606e in steps from A to I. The steps will be similar when assembling other wavone606 product.



(A). SIM card
(B). Battery connector. Please note **battery is the ONLY power source for waveon606**
(C). Upper housing (with screw holes)
(D). GSM antenna connector and GPS antenna (red) connector.
(E). 14-pin connector (See Appendix E for detail PIN description)
(F). RS232 connector which is connected to PC COM port
(G). 5-volt battery charging port which is normally connected to PC USB port.
(H). 12-volt battery charging cable which is normally connected to vehicle battery
(I). GPIN and GPOU pins and other cables.

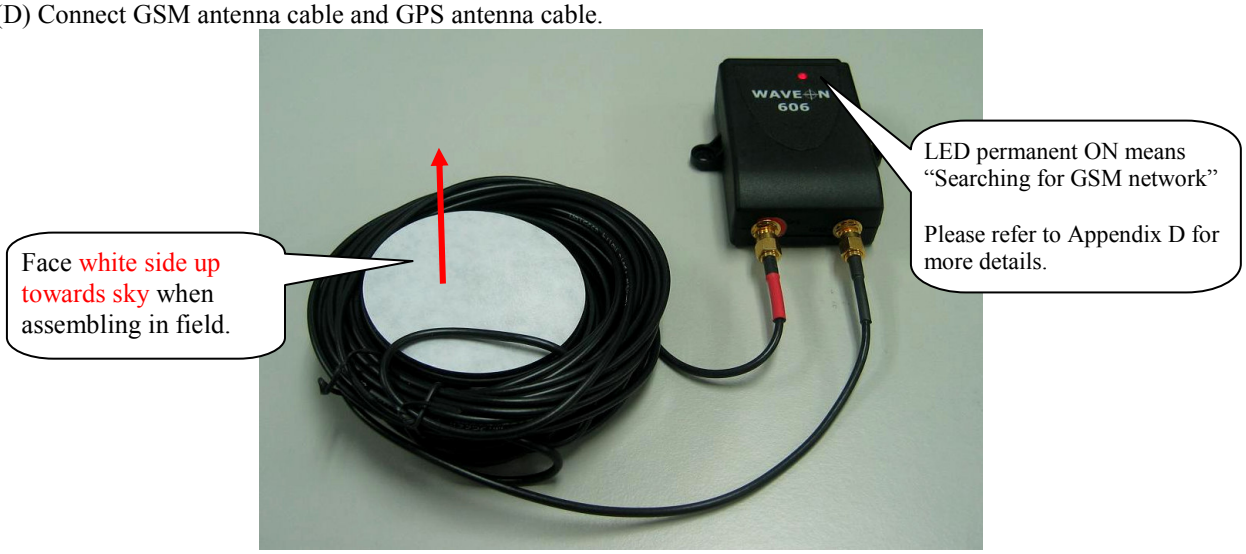
(A) Insert SIM card similar as normal hand phone.

(B) Plug battery into main board connector. **Battery from factory may be flat, please refer to (G), (H) on how to charge battery. Recommend to fully charge battery before using.**



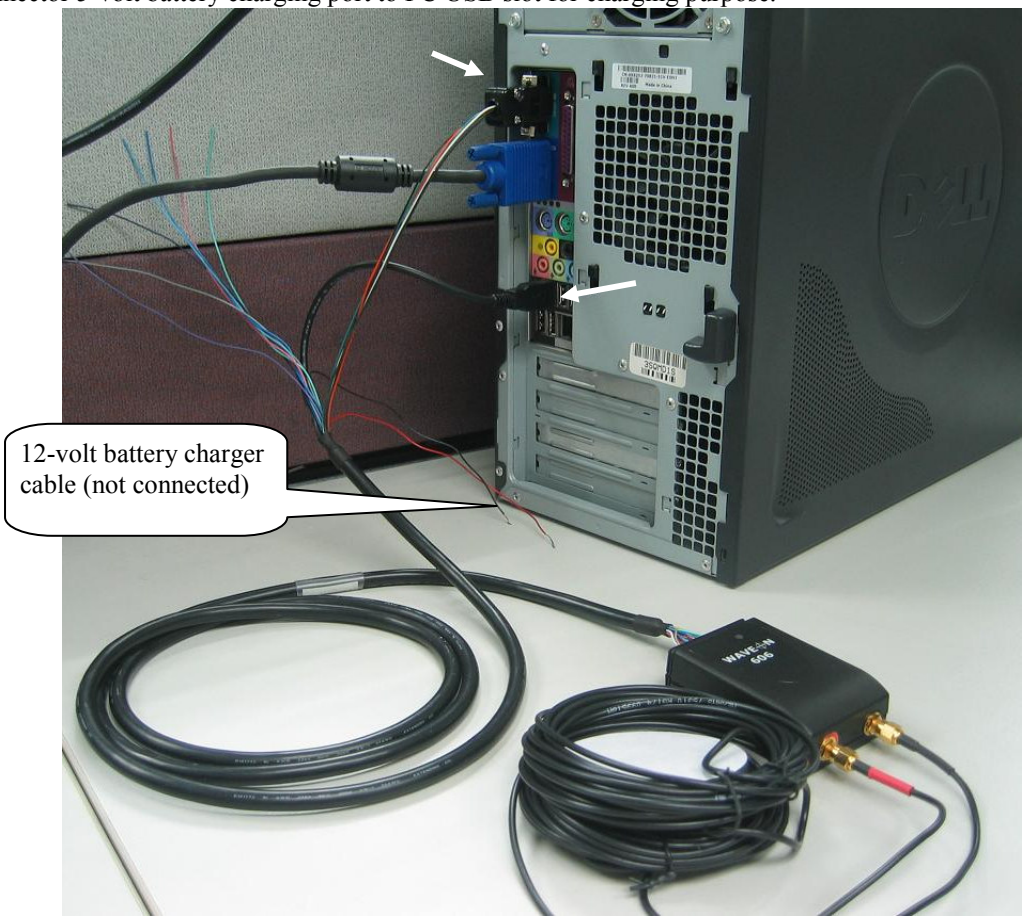
(C) Assemble upper housing after make sure all cables are connected correctly. **Fix it with main board by using four screws (not shown in picture) which can be found in package.**

(D) Connect GSM antenna cable and GPS antenna cable.



(E) Connect 14-pin connector to Waveon606 (for configuration and firmware upgrade)

(F) Connector RS232 connector to PC COM port.
(G) Connector 5-volt battery charging port to PC USB slot for charging purpose.



(H) Alternatively, connect 12-volt battery charging cable to 12 DC power source for charging purpose. (No picture)

1.5 Start Device Configuration

After installation (see "Quick Installation Guide") is finished, it is time to configure the Device by using WUC commands. There are three ways available for configuration. That is, via UART, via SMS or via GPRS. UART must be used for the first configuration. SMS can be used after configuring SMS recipient by using $SMSN. GPRS can be used after setting up backend server and configuring GPRS network properly by using $OPMD, $GPRS, $RPIT and $DSCG etc.

- When configuring via UART port, encapsulate WUC command into $UART as shown below.
  AT$UART="<actual WUC command>"
- When configuring via SMS, encapsulate WUC command into $CSMS as shown below.
  AT$CSMS=<password>,"<actual WUC command>"
- When configuring via GPRS, use WUC command directly without encapsulation.
  <actual WUC command>

Below introduced how to start UART configuration. Suppose your PC is installed with Windows XP and HyperTerminal application, please follow below steps (1.5.1 ~ 1.5.6) to configure your HyperTerminal and then use WUC commands to do the configuration. The steps will be similar in other Windows OS.

1.5.1 Open HyperTerminal from:
"WindowsXP" ➔ "Start" ➔ "All Programs" ➔ "Accessories" ➔ "Communications" ➔"HyperTerminal"

1.5.2 Enter W606 in **Name** field. Click **OK.** (Fig1.2)

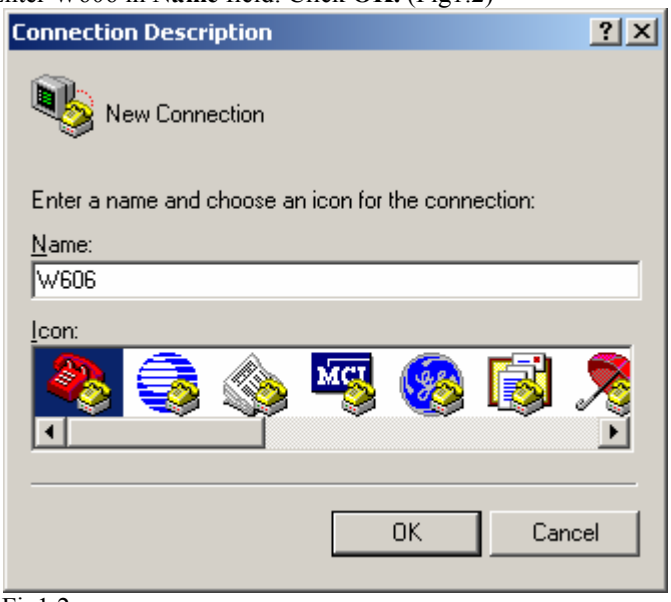


Fig1.2

1.5.3 Select COM1 for **Connect using**. It must correspond to your physical connection.

1.5.4 Click **OK**. (Fig1.3)

Fig1.3

1.5.5 Configure **Port Settings** as shown below. Click **OK**. (Fig1.4)

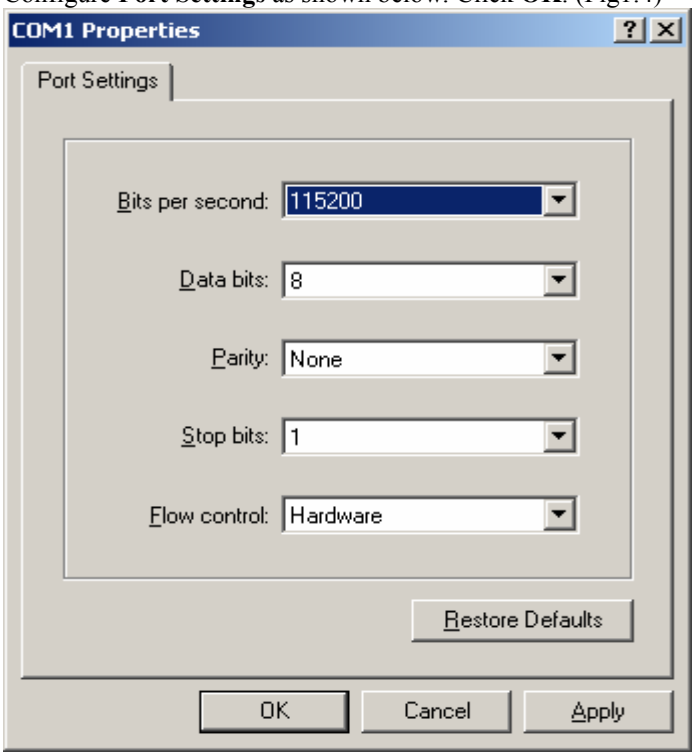


Fig1.4

1.5.6 In HyperTerminal window, type WUC command as described in $UART.(Fig1.5)

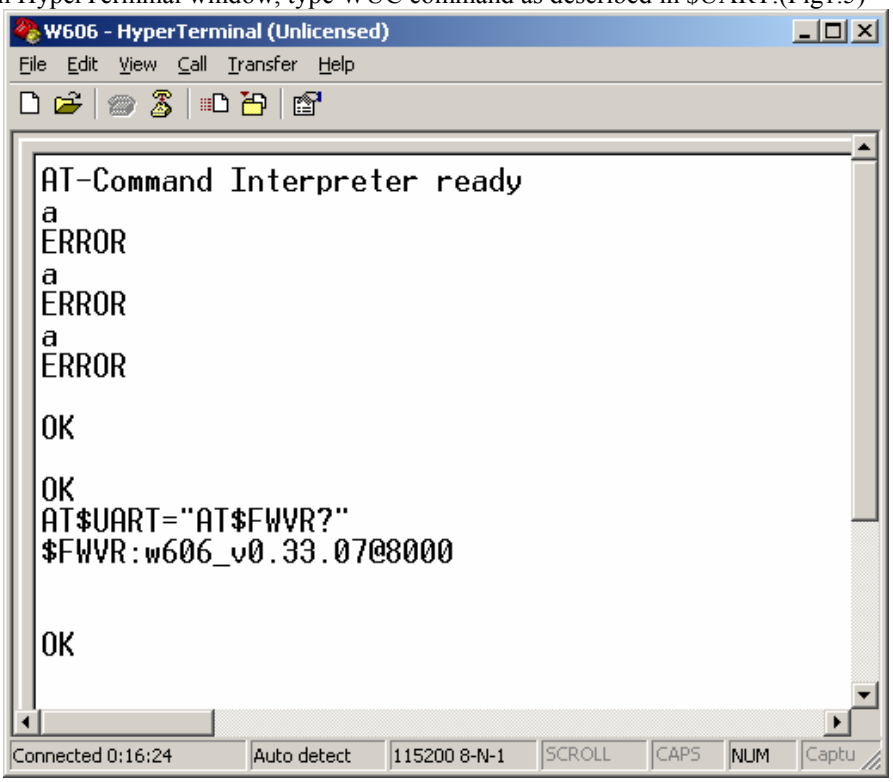


Fig1.5

Tips:

In case that no typing is allowed in HyperTerminal, please click **'a'** (or **'A'**) and **Enter** button repeatedly until device responses. There will be some ERROR message when entering 'a' or 'A'. Kindly ignore them since it just tells that 'a' or 'A' is not valid command.

Warning:

Upon typing any AT command into HyperTerminal, Device will enter a so-called "AT command" mode. It is a temporary mode dedicated for AT commands execution. After entering "AT command" mode, the normal operational mode defined in $OPMD is totally stopped. To exit from "AT command" mode, a RESET (E.g., $RSET via HyperTerminal or SMS) is mandatory. After RESET, no further AT typing is allowed in HyperTerminal. Otherwise Device will enter "AT command" mode again. (Fig1.6)

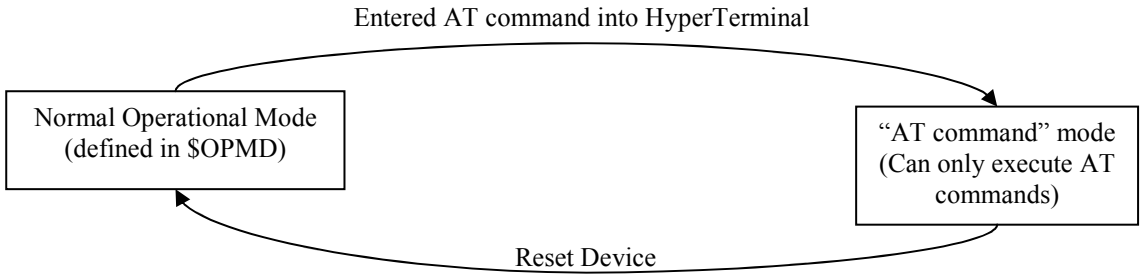


Fig1.6 Relation between Normal Operational mode and "AT command" mode

2. Command General Information

WUC commands may be able to support up to four types: SET, GET, REPORT and ACK. All WUC commands can support ACK, while some can only support one or two, if not all, among SET, GET, or REPORT. The details of each command are described in the Section "Command list".

2.1 Command Types

Operation	Send Direction	Usage
('=') SET	Backend to Device	o Update Device configurations and features; o Erase flash space;
('?') GET	Backend to Device	o Query current configurations or status from Device.
(':') REPORT	Device to Backend	o Report GPS position at intervals; o Answer GET command from Backend; o Report system status; o Report alert and error message if any;
('!') ACK	Either direction	o Acknowledgement after handling SET/GET/REPORT command. It can be enabled/disabled by $AKTO command.

2.2 Command Format

All WUC commands follow a format shown below. Do conform exactly to it in the application development process and pay additional attention to the RED part.

2.2.1 General Format

[AT]<Name><Type>[<Value1>,<Value2>…][@VEID][*CHKSUM]<CR><LF>

Where

Field	Definition
[AT]	Start mark for all commands sent from Backend to Device. It must be in CAPITAL character.
<Name>	WUC name in format of $XXXX. It must be in CAPITAL character.
<Type>	Type of WUC <Name>. Its value can be one of the following four characters: o '=' stands for SET o '?' stands for GET o ':' stands for REPORT o '!' stands for ACK
[<Value1>,<Value2>…]	Parameters of WUC <Name>. Please refer to section "Command List" for more detail.
[@VEID]	VEID is a 16-bit decimal value with a range from 0001 to FFFF. It is a unique Device identifier and may be appended to the commands sent by Device.
[*CHKSUM]	CHKSUM provides limited error-checking capability. It is an 8-bit HEX value which is the XOR of all characters between '$' and '*'. Please note '$' and '*' are not included in checksum calculation.
<CR><LF>	Terminal mark.

2.2.2 The detailed format for each type

- Format of SET type.
  AT$XXXX=<value1>[,<value2>,<value3>…<valueN>][*CHKSUM]<CR><LF>

  Note: [@VEID] is NOT required since it is always sent by Backend to Device.

- Format of GET type
  AT$XXXX?[*CHKSUM]<CR><LF>
  AT$XXXX?<value>[*CHKSUM]<CR><LF>

  Note: [@VEID] is NOT required since it is always sent by Backend to Device.

- Format of REPORT type
  $XXXX:<value1>[,<value2>,<value3>…<valueN>][@VEID][*CHKSUM]<CR><LF>

- Format of ACK type
  When sent from Backend:
  AT$XXXX! [*CHKSUM]<CR><LF>
  AT$XXXX![value][*CHKSUM]<CR><LF>

  When sent from Device:
  $XXXX! [@VEID][*CHKSUM]<CR><LF>
  $XXXX![value][@VEID][*CHKSUM]<CR><LF>

2.2.3 Notes

- All characters, unless specifically mentioned, are in printable ASCII form.
- No SPACE is allowed inside command;
- All commands from Backend to Device have prefix “AT”;
- All commands from Device to Backend DONT have prefix “AT”;
- All values are in decimal except that [*CHKSUM] is in hexadecimal;
- [VALUE] is an optional value while <VALUE> is compulsory.
- All commands must be appended with <CR><LF>;
- **(Optional)** indicates that certain parameter can be ignored during configuration.

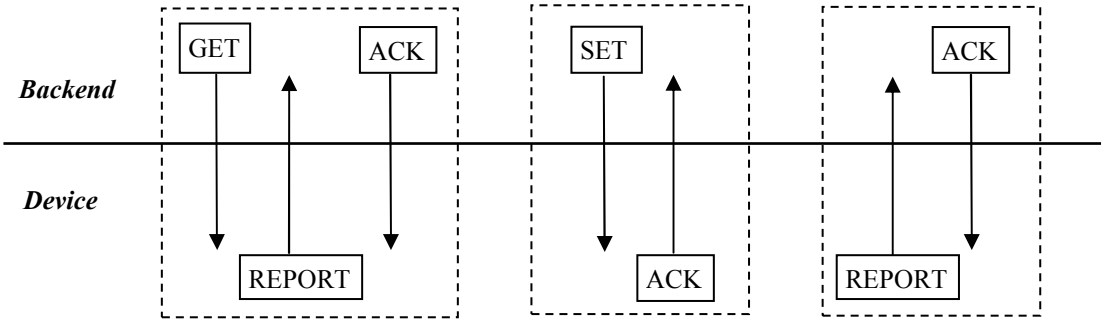
2.3 Communicate Procedure

All WUC commands, unless specially mentioned, conform to the procedures listed below. They can be divided into two categories: *ASCII-based commands* and *BINARY-based commands*. Please note [@VEID] and [*CHKSUM] should be considered accordingly in actual implementation.

In Fig2 and Fig3, "A ➔ B" means that packet is sent from A to B.

2.3.1 ASCII-based commands

All WUC commands, except $ELDA and $GLDA, are based on ASCII string. These commands will follow a procedure as shown in Fig2.1.

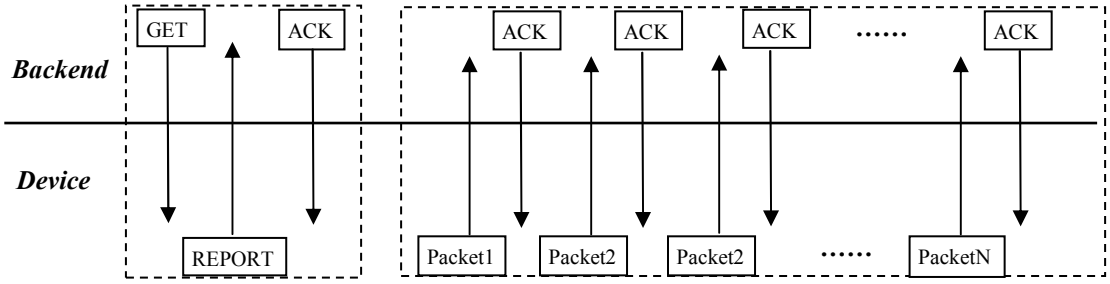


a) Backend initiates GET b) Backend initiates SET c) Device initiates REPORT

Fig2.1 Communicate Procedure of ASCII-based commands

2.3.2 BINARY-based commands

So far only $ELDA and $GLDA are BINARY-based commands (Please refer to $ELDA and $GLDA for more details). They follow a TFTP-like, Infowave Proprietary Binary Protocol (IPBP) to send out data from Device. Their communicate procedure is given in Fig2.2.



Step 1: Backend requests to rend Step 2: Device sends data in binary packets

Fig2.2 Communicate Procedure of BINARY-based commands

- IPBP Packet structure

Field Name	Type	Length (byte)	Meaning
Header	STRING	6	Constant string "$ELDA" or "$GLDA"
Sequence No	U16_TYPE	2	Packet sequence number starting from one.
Vehicle ID	U16_TYPE	2	A unique Device identifier.
Items	U16_TYPE	2	Number of items included in Data. • Each packet has **25** items. Except that, • The last packet has a smaller number of items or 0 items.
Data	U8_TYPE	The remaining	Actual data. E.g., in case of "$ELDA", Data will be the event logs.

- More details on IPBP
  IPBP is a TFTP-like protocol. As shown in Fig3, it has two steps.

  Step1: Backend requests to read.
  - Backend sends request with GET command, mainly tell the number of items to be retrieved.
  - Device responses with REPORT command, mainly tell the number of items allowed to be retrieved.
  - Backend acknowledges the response with ACK command.
  - Only when step1 finished successfully, step2 will start.

  Step 2: Device sends data in binary packets
  - After proper ACK is received in Step1, Device sends out the first packet with appropriate items.
  - If the number of its items is smaller than 25, it is the last packet and the process finished.
  - Otherwise, upon the consequential ACK is received, Device sends out the second packet.
  - This process continues until the last packet is sent.

- Example for retrieving event log data.

No	Backend	Direction	Device	Meaning
Query number of available items.				
1.	AT$ELRQ?	→	-	Query current available items.
2.	-	←	$ELRQ:0,120,1000	Report 120 items are available.
3.	AT$ELRQ!	→	-	Acknowledgement.
Step 1: Backend requests to read.				
4.	AT$ELRQ?55	→	-	Request to retrieve the latest 55 items.
5.	-	←	$ELRQ:55,120,1000	Response 55 items are allowed to retrieve.
6.	AT$ELRQ!55	→	-	Acknowledge 55 items.
Step 2: Device sends data in binary packets				
7.		←	$ELDA packet 1	The 1st packet with 25 items.
8.	AT$ELDA!1	→	-	Acknowledge the 1st packet.
9.	-	←	$ELDA packet 2	The 2nd packet with 25 items.
10.	AT$ELDA!2	→	-	Acknowledge the 2nd packet.
11.	-	←	$ELDA packet 3	The last packet with 5 items
12.	AT$ELDA!3	→	-	Finished.

2.4 Command Error Handle

As shown in Fig2.3, if SET or GET command is invalid, Device will response with $STER command which tells the error code. Meanwhile the corresponding process will be stopped. Please refer to $STER for details.

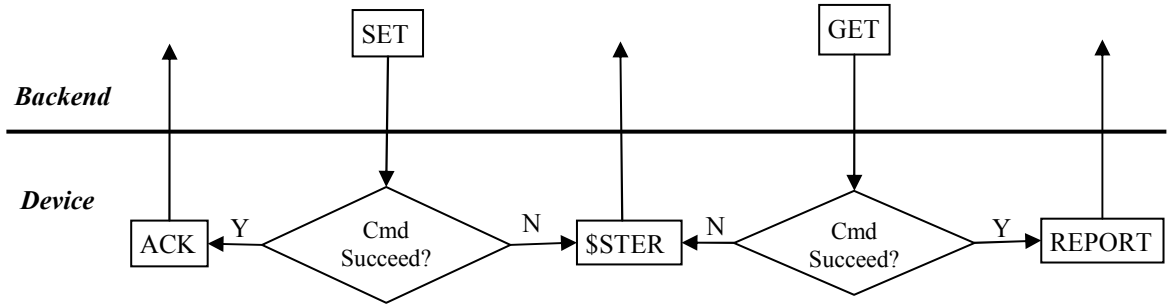


Fig2.3. Command Error handle

3. Command List

This section describes WUC commands in details. The ACK type for all WUC commands is described separately in subsection 3.1. Other types are explained in subsections onwards.

Before going further, please note that,

- In **Type** subsection, the sting after "ACK:", "GET:", "SET:" and "REPORT:" is the actual command.
- Examples for all commands should append <CR><LF> to the end if used in real trial.
- **[A, B]** indicates a range which are integer values from A to B, inclusively.
- **{x, y, …, w}** indicates a set of discrete values x, y,… and w.
- The string length given below includes NULL terminal. E.g., APN password is a string of up to 16 characters, the actual password can contain 15 characters in maximum.
- Call "AT$SCFG=1" to save all configuration updates.

3.0 "$XXXX" (acknowledgement)

- **Definition**
  Each WUC command, starting from subsection 3.1, acknowledges received packet with its own name. The "XXXX" here is just a wildcard. In real implementation, it should be replaced with the actual command name which is to be acknowledged. In sum, it will be used in three conditions.
  - When Backend receives a command, it acknowledges with AT$XXXX![<value>]
  - When Backend receives a binary packet, it acknowledges with AT$XXXX!<value>
  - When Device receives a SET command, it acknowledges with $XXXX![<value>]
- **Type**
  ACK: AT$XXXX![<Parameters>]
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	value	U16_TYPE		Legal Values : [0, 65535] Default Value: N/A --- 0~5000: ACK when WUC succeeds (0 can be omitted) Others : undefined. When WUC failed, its error information will be returned by $STER command.
2	Human string	STRING (Optional)		A string help user to identify the error code.
E.g. AT$PSVL! //Backend acknowledges a successful $PSVL report. $GPRS! //Device acknowledges a successful $GPRS SET command. $STER:xx,… //Device acknowledges a failed WUC command. See $STER for more details.				

3.1 "$AKTO" (AcKnowledge and TimeOut)

- **Definition**

  Enable/Disable acknowledgement for $PSVL and adjust UDP timeout value.
  - If Acknowledgement is disabled, GPS report ($PSVL) does not require acknowledgement but other commands still need it. As a cost, the $PSVL report will become less stable.
  - If Acknowledge is enabled, all commands from Device have to be acknowledged. On one hand, when Device sends out a command, it will wait for ACK from Backend for maximum **Timeout** milliseconds. If no ACK after **Timeout** milliseconds, Device will try to resend the same packet for up to 2 times. If still no ACK after resends, further actions will be taken. For $PSVL command, current GPS message will be temporarily buffered and retried as soon as the communication between Device and Backend is recovered. For other commands, their message will be discarded. On the other hand, when Device receives a command, it will send out ACK indicating the execution result of received command.

- **Type**

  GET: AT$AKTO?
  SET: AT$AKTO=<Parameters>
  REPORT: $AKTO:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Acknowledgement	U8_TYPE		Legal Values : {0, 1} Default Value: 1 0: disable 1: enable When Acknowledgement is disabled, $PSVL does not require ACK.
2	Timeout	U16_TYPE	milisecond	Legal Values: [2000, 8000] Default Value: 3000 The maximum time waiting for ACK. It is valid only when Acknowledgement is enabled.

E.g.

(SET)
AT$AKTO=1,4000

(GET)
AT$AKTO?

(REPORT)
$AKTO:1,4000

Note: the content inside parentheses is comment. The same in subsections.

3.2 "$PSVL" (PoSition and VeLocity)

- **Definition**
  Device reports GPS message to Backend at intervals. The GPS message is packed in a proprietary format as shown in the table "Parameters".
- **Type**
  GET: AT$PSVL?
  REPORT: $PSVL:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Date	U32_TYPE	ddmmyy	Current date
2	UTC Time	U32_TYPE	hhmmss	**When "Date" is not ZERO**, it indicates Current UTC time. **When "Date" is ZERO**, that is, before getting the first valid GPS position, it indicates device uptime.
3	Position Fix	U8_TYPE	-	Legal Values: {0, 1, 2, 9} Default Value: N/A 0: No fix / Invalid 1: Standard GPS (2D/3D) 2: Differential GPS 9: Last known STANDARD GPS (ON/OFF via 5th parameter of $RPIT) Position Fix Indicator. When position is NOT fixed (==0), all parameters (Id >3) are invalid and should be ignored in your application.
4	Number of satellites	U8_TYPE	-	Legal Values: [1, 12] Default Value: N/A Dummy when "Position Fix" == 0.
5	Latitude	DOUBLE	degree	North:+, South: - "Precision" in 1/1000000 degree Dummy when "Position Fix" == 0.
6	Longitude	DOUBLE	degree	East:+, West: - "Precision" in 1/1000000 degree Dummy when "Position Fix" == 0.
7	Altitude	FLOAT	meter	"Precision" in 0.1m Dummy when "Position Fix" == 0.
8	Speed	FLOAT	kilometer per hour	"Precision" in 0.01kph When "Position Fix" == 0, this field MUST be ignored.

				For debugging purpose, this value indicates whether GPS module is alive or not. Value is changing means GPS module is alive. Otherwise not.
9	True course	FLOAT	degree	Valid Values : [0, 359] Default Value: N/A "Precision" in 0.01degree Course Over Ground When "Position Fix" == 0, this field MUST be ignored. For debugging purpose, this value indicates current battery level in unit of mV.
10	HDOP	FLOAT	-	"Precision" in 0.01 Horizontal Dilution of Precision When "Position Fix" == 0, this field MUST be ignored. For debugging purpose, this value indicates current GSM signal strength in unit of dBm.
E.g. (REPORT) $PSVL:030805,013553,1,5,1.387785,103.849695,31.1,0.015,341.79,1.92				

- **Notes**
  "Precision" does not indicate the precision of position but the least significant bit in the parameter. The actual position precision is determined by the GPS system as well as GPS module.

3.3 "$OPMD" (Operational MoDe when device boots up)

- **Definition**

  Manage Device operational mode. The mode tells Device in which way it communicates with outside. It can be one of the following six modes.

Mode	Meaning
MODE_MODEM_ONLY	Device works as a standard GPRS or GSMDATA modem. When connected to PC via HyperTerminal, Device accepts all WUC commands and normal AT commands.
MODE_GPRS_ONLY (Default mode)	Device reports GPS messages at certain interval which is determined by "GPRS Report Interval" in $RPIT and parameters in $DSCG. Normally Device sends out GPS messages and all other messages via UDP-based GPRS network. **When GPRS is disconnected, Device stops active TX of all messages except alert message.** Device buffers oncoming GPS message into GPS queue and sends out alert message via SMS. If SMS TX fails, Device retries for up to **3** times. If retry fails, Device discards the message. During the period after GPRS is disconnected, Device keeps reconnecting GPRS in every 30 seconds. After GPRS is recovered, Device continues to work as normal.
MODE_GPRS_SMS	Device reports GPS messages at certain interval which is determined by "GPRS Report Interval" in $RPIT and parameters in $DSCG. Normally Device sends out GPS messages and all other messages via UDP-based GPRS network. **When GPRS is disconnected, Device redirects all messages to SMS.** If SMS TX fails, Device retries for up to **3** times. If retry fails, Device buffers GPS message into GPS queue and discards all other message. During the period after GPRS is disconnected, Device keeps reconnecting GPRS in every 30 seconds. After GPRS is recovered, Device automatically directs all messages back to GPRS and continues to work as normal.
MODE_SMS_ONLY	Device reports GPS messages at certain interval which is determined by "SMS Report Interval" in $RPIT and parameters in $DSCG. **Normally Device sends out all messages via SMS.** If SMS TX fails, Device retries for up to **3** times. If retry fails, Device discards all messages.
MODE_GSM_DATA_ONLY	It is the same as MODE_GPRS_ONLY except that Device use GSM-DATA network instead of GPRS. This mode is useful in some region where GPRS is not available while GSM-Data is available.

- **Method**

  GET: AT$OPMD?
  SET: AT$OPMD=<Parameters>
  REPORT: $OPMD=<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Operational Mode	U8_TYPE		Legal Values : {0, 1, 2, 3, 4} Default Value: 1 0: MODE_MODEM_ONLY 1: MODE_GPRS_ONLY 2: MODE_GPRS_SMS 3: MODE_SMS_ONLY 4: MODE_GSM_DATA_ONLY
E.g. (SET) AT$OPMD=1 (GET) AT$OPMD? (REPORT) $OPMD:1*1D				

3.4 "$RPIT" (RePort InTerval for $PSVL messages)

- **Definition**

  Manage time intervals at which Device reports GPS message via GPRS or SMS. It is called "Time Interval" in short. Parameter 2~3 are Time Interval for different report medium. When certain interval expired, Device will report current GPS position. Moreover, there is also advanced options (see $ADRP) available to further optimize device's report behavior.

  Besides "Time Interval", Device also supports "Distance/Course Interval" as explained in command $DSCG. Please note, only either of $RPIT and $DSCG can be enabled at anytime. When "Time Interval" is to be enabled, "Distance/Course Interval" must be disabled (***AT$DSCG=0,0,0,0***) in advance.

- **Method**

  GET: AT$RPIT?
  SET: AT$RPIT=<Parameters>
  REPORT: $RPIT:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Dummy			
2	GPRS Report Interval	U16_TYPE	second	Legal Values : {0, [MIN_RPT_TIME, 65535]} Default Value: 30 --- 0: disable others: time in second Valid in **MODE_GPRS_ONLY**, **MODE_GPRS_SMS** and **MODE_GSM_DATA_ONLY** mode.
3	SMS Report Interval	U16_TYPE	second	Legal Values : {0, [MIN_RPT_TIME, 65535]} Default Value: 30 0: disable others: time in second Valid in **MODE_SMS_ONLY** mode.
4	Dummy			

E.g.
(SET)
AT$RPIT=0,30,40,0

(GET)
AT$RPIT?

(REPORT)
$RPIT:0,30,40,0

3.5 "$DSCG" (DiStance and Course over Ground)

- **Definition**

  Manage distance/course intervals at which Device reports GPS message via GPRS or SMS. It is called "distance/course intervals" in short. Parameter 1 and 2 specify distance interval and course interval accordingly. When certain interval expired, Device will report current GPS position. Moreover, there is also advanced options (see $ADRP) available to further optimize device's report behavior.

  Please note, only either of $RPIT and $DSCG can be enabled at anytime. When "Distance/Course Interval" is to be enabled, "Time Interval" must be disabled (***AT$RPIT=0,0,0,0***) in advance.

- **Method**

  GET: AT$DSCG?
  SET: AT$DSCG=<Parameters>
  REPORT: $DSCG:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Distance Interval	U16_TYPE	meter	Legal Values : {0, [50, 65535]} Default Value: 0 ---------- 0: disable others: distance in meters GPS message will be sent based on distance. The minimum valid distance should be calculated based on your average speed and 10-second time: Min_dist = Speed * 1000 * 10 / 3600 E.g., Speed (Km per hour) Min_dist (m) 25 69 50 138 75 208 Its performance is heavily depending on the quality of GPS signal. The higher quality of GPS signal, the more accurate interval. It will stop reporting when no GPS signal.
2	Interval of COG (Course over ground)	U16_TYPE	degree	Legal Values : {0, [2, 359]} Default Value: 0 ---------- 0: disabled Others: enabled. When COG difference is greater than this value, current GPS message is reported. Its performance is heavily depending on the quality of GPS signal. It will stop reporting when no GPS signal.
3	Distance Threshold	float	meter	Legal Values : [0.0, 100.0] Default Value: 0.0 ---------- 0: disable A threshold for reducing ERROR of distance

				interval. When Device is stationary, its GPS position will still keep floating slightly and thus Device is keeping “moving” slightly. A distance delta is calculated in every 2 seconds. When distance delta is smaller than threshold, it is treated as ZERO. Otherwise, it is counted into “Distance Interval”. Please use WUC **“$TRSD”** to get a suitable threshold value.
4	COG Threshold	float	degree	Legal Values : [0.0, 100.0] Default Value: 0.0 --- 0: disable A threshold for reducing ERROR of COG interval. When Device is stationary, its GPS position will still keep floating slightly and thus Device is keeping “moving” slightly. A true course delta is calculated in every 2 seconds. When course delta is smaller than threshold, it is treated as ZERO. Otherwise, it is counted into “Interval of COG”. Please use WUC **“$TRSD”** to get a suitable threshold value.
5	Dummy			
E.g. (SET) AT$DSCG=1000,1000,1,1,0 (GET) AT$DSCG? (REPORT) $DSCG:1000,1000,1,1,0				

3.6 "$ADRP" (ADvanced RePort)

- **Definition**
  Provide advanced options to $RPIT and $DSCG. For example, report last known GPS position, or stop reporting upon ignition off etc.
- **Method**
  GET: AT$ADRP?
  SET: AT$ADRP=<Parameters>
  REPORT: $ADRP:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Report zero position	U8_TYPE		Legal Values : {0, 1} Default Value: 0 --- 0: report last known GPS position when no GPS; 1: report real-time zero position when no GPS.
2	Report when ignition ON (Optional)	U8_TYPE		Legal Values: {0, 1, 2} Default Value: 2 --- 0: Report when ignition ON. Ignition ON is active LOW. That is, Device keeps reporting when ignition pin is LOW, stops reporting when ignition pin is HIGH. 1: Report when ignition ON. Ignition ON is active HIGH. That is, Device keeps reporting when ignition pin is HIGH, stops reporting when ignition pin is LOW. 2: Report regardless of ignition HIGH/LOW status. Please note it is valid only for "Time Interval" but not for "distance/course intervals".
3	Additional Reports after ignition OFF (Optional)	U16_TYPE		Legal Values: [0, 65535] Default Value: 0 --- When ignition is detected OFF, device has ability to further send out X additional reports at the same intervals. X is configurable via this parameter. Please note that this feature is valid only when "Ignition" (see above) is set to 0 or 1. Please note it is valid only when parameter 2 "Report when ignition ON" is set as 0 or 1.
4	Disconnect GPRS after ignition OFF (Optional)	U8_TYPE		Legal Values: {0, 1} Default Value: 0 --- 0: keep GPRS connection. **(Recommended)** 1: disconnect GPRS connection after ignition OFF and reconnect GPRS connection after ignition ON. **Unless necessary,** it is not recommended to use 1 since it will prevent Device from sending out alert during ignition OFF. Please note it is valid only when parameter 2 "Report when ignition ON" is set as 0 or 1.

5	Buffer non-fix GPS (Optional)	U8_TYPE		Legal Values: {0, 1] Default Value: 0 --- 0: discard non-fix GPS when TX failed. **(Recommended)** 1: buffer non-fix GPS when transmission failed. When failed talking with Backend during a short period, Device is able to temporarily buffer limited number of latest reports for future retrying. This option controls if non-fix GPS needs to be buffered in addition to valid GPS. **Recommended value is 0 to leave more space for valid GPS.**
(SET) AT$ADRP=0,2,0,0,0 //Last known position, report regardless of ignition pin status. AT$ADRP=0,1,5,0,0 //Last known position, report only when ignition pin is HIGH, additional reports is 5.				

3.7 "$SHTM" (SHift TiMe)

- **Definition**

  It defines a valid time window during which Device will report GPS message, while outside which Device stops reporting. For example, Device can be set to report GPS messages from 7:00:00 (hh:mm:ss) to 12:00:00.

- **Type**

  GET: AT$SHTM?
  SET: AT$SHTM=<Parameters>
  REPORT: $SHTM:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Enable/Disable	U8_TYPE		Legal Values : {0, 1, 2} Default Value: 0 ---------- 0: disable 1: enable ONE time window
2	TW1_start	U32_TYPE	hhmmss	Legal Values : [000000, 235959] Default Value: 0 Specify the time windows.
3	TW1_end	U32_TYPE	hhmmss	
4	Dummy			
5	Dummy			

E.g.
(SET)
AT$SHTM=0 //Disable shift time feature

(SET)
AT$SHTM=1,100533,230000 //Start from 10:05:33 and end at 23:00:00

(GET)
AT$SHTM?

(REPORT)
$SHTM:0*08
$SHTM:1,100533,230000*0C

3.8 "$GPRS" (GPRS network parameters)

- **Definition**
  Configure GPRS network parameters, Backend IP address and port number.
- **Type**
  GET: AT$GPRS?
  SET: AT$GPRS=<Parameters>
  REPORT: $GPRS:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	APN server	STRING		Legal Values : string of up to 24 characters Default Value: internet
2	APN username	STRING		Legal Values : string of up to 24 characters Default Value: EMPTY_STR
3	APN password	STRING		Legal Values : string of up to 16 characters Default Value: EMPTY_STR
4	Server IP address	STRING		Legal Values : string of up to 16 characters Default Value: EMPTY_STR IP is in format xxx.xxx.xxx.xxx
5	Port num	U16_TYPE		Legal Values : [1024, 65535] Default Value: 12000 The port numbers are divided into three ranges: the Well Known Ports, the Registered Ports, and the Dynamic and/or Private Ports. The Well Known Ports are those from 0 through 1023. The Registered Ports are those from 1024 through 49151. The Dynamic and/or Private Ports are those from 49152 through 65535
E.g. (SET) AT$GPRS=internet,uname,passwd,127.0.0.1,5556 (GET) AT$GPRS? (REPORT) $GPRS:internet,uname,passwd,127.0.0.1,5556				

3.9 "$SMSN" (SMS Network parameters)

- **Definition**
  Configure SMS parameters.
- **Type**
  GET: AT$SMSN?
  SET: AT$SMSN=<Parameters>
  REPORT: $SMSN:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	(DUMMY)	U16_type		Legal Values : [0, 65535] Default Value: 0 ---------- 0: disable >0: enable. **This feature is not implemented so far.**
2	SMS recipient number	STRING		Legal Values : string of up to 32 characters Default Value: "+6512345678" Device will send all SMS to this number only.
3	SMS center number (DUMMY)	STRING		Legal Values : string of up to 32 characters Default Value: EMPTY_STR In most cases, SMS center number is already stored in SIM card. So this parameter is disabled for timing.

E.g.
(SET)
AT$SMSN=0,+6512345678,0

(GET)
AT$SMSN?

(REPORT)
$SMSN:0,+6512345678,0*19

3.10 "$VNUM" (Voice NUMbers.)

- **Definition**
  Configure EMERGENCY call number and normal call number.
- **Type**
  GET: AT$VNUM?
  SET: AT$VNUM=<Parameters>
  REPORT: $VNUM:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Emergency call	STRING		Legal Values : string of up to 32 characters Default Value: EMPTY_STR When this button is clicked, Device will send an EMERG_CALL alert to Backend.
2	Voice Number 2	STRING		Legal Values : string of up to 32 characters Default Value: EMPTY_STR When this button is clicked, Device will send a VOICE_CALL alert to Backend.
E.g. (SET) AT$VNUM=+6511111111,+6522222222 (GET) AT$VNUM? (REPORT) $VNUM:+6511111111,+6522222222*16				

3.11 "$EVMN " (EVent MaNagement)

- **Definition**

  It manages the way for Device to handles all available events which are listed in subsection "**Predefined Events**" (see below) and will be triggered upon their "**Trigger Condition**". Device may handle events in two ways. One is to send an alert to Backend as shown in command "$ALER". It is configurable via the 1st parameter "Alert ON/OFF". The other is to log it into Device internal log database which can be retrieved by using commands "$ELRQ" and "$ELDA". It is configurable via the 2nd parameter "Logging ON/OFF". Please note that both parameters are bit-wised decimal values with each bit corresponding to one event.

- **Type**

  GET: AT$EVMN?
  SET: AT$EVMN=<Parameters>
  REPORT: $EVMN:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Alert ON/OFF	U32_TYPE	-	Legal Values : [0, 4294967295] or [0, 0xFFFFFFFF] Default Value: 371 (=0x173) ---------- 0: disable 1: enable. It is a bit-wised decimal value. Each bit corresponds to on type of event.
2	Logging ON/OFF	U32_TYPE	-	Legal Values : [0, 4294967295] or [0, 0xFFFFFFFF] Default Value: 307 (=0x133) ---------- 0: disable 1: enable. It is a bit-wised decimal value. Each bit corresponds to on type of event.

Here is an example to turn ON **alert** functionality for events EVENT_EMERG_CALL, EVENT_TAMPER, EVENT_POWER_LOW and EVENT_POWER_LOST, and turn ON **logging** for event EVENT_TAMPER alone, and turn OFF alert/logging for all other events.

- First calculate the value for both parameters in terms of "Predefined Event" (see below).
  1st parameter = sum(0x00000001+0x00000002+0x00000010+0x00000020) = 0x33(HEX) = **51** (Decimal).
  2nd parameter= sum(0x00000002) = 0x2(HEX) = **2** (Decimal)
- Then call **AT$EVMN=51,2** to enable alert and logging functionalities as required. And make sure $EVMN command returns successful acknowledgement **$EVMN!**.
- Finally save the configuration via $SCFG command. Otherwise your setting will be lost after reset.

(SET)
AT$EVMN=51,2

(GET)
AT$EVMN?

(REPORT)
$EVMN:51,2

- **Predefined Events**

Events Name	Value	Trigger Condition
EVENT_EMERG_CALL	0x00000001	Emergency call button is clicked.
EVENT_TAMPER	0x00000002	Tamper sensor is triggered.
EVENT_POWER_LOW	0x00000010	Buttery or DC power is below than predefined value.
EVENT_POWER_LOST	0x00000020	Battery or DC power is too low to support Device.
EVENT_NO_GPS_NMEA	0x00000040	GPS module is not sending enough position messages.
EVENT_IGNITION	0x00000100	Change of Ignition status. E.g. ON to OFF, OFF to ON.
EVENT_CNTX_GPS_LOSS	0x00000200	Device receives X continuous invalid GPS data since it gets its first valid GPS data. Value X is adjustable via "$MSCL".
Reserved	0x00000400	
Reserved	0x00000800	
EVENT_DINPUT_1	0x00001000	Status change of digital input pin 1.
EVENT_DINPUT_2	0x00002000	Status change of digital input pin 2.
EVENT_OVER_SPEED	0x00004000	Device's speed exceeds a predefined value. The speed limitation is adjustable via "$MSCL".
EVENT_ACTIVE_DINPUT_1	0x00008000	Optional. See "$VSDM" for details.
Reserved	0x00010000	
Reserved	0x00020000	
EVENT_MOVE_FM_STATIC	0x00040000	A stationary Device starts moving. User can customize the move-judge-speed via "$MSCL".
EVENT_NO_INTERVAL	0x10000000	Time interval is disabled by mistake.
EVENT_TX_TIMEOUT	0x20000000	UDP transmit retry fail.
EVENT_NETWORK_FAILURE	0x40000000	GPRS or GSM-Data is disconnected.

- **Notes**

  The reliability of some events, such EVENT_OVER_SPEED and EVENT_MOVE_FM_STATIC, are heavily depending on the quality of GPS positioning and may encounter heavy ERROR when GPS position is in very low quality.

3.12 "$ALER" (ALERt when certain event happen.)

- **Definition**
  Send out alert message when an event happens.
- **Type**
  REPORT: $ALER:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Event type	U32_TYPE	-	Legal Values : refer to "Predefined Events" Default Value: N/A
2	Event value	U32_TYPE	-	Legal Values: [0,2^{32}-1] Default Value: 0
3	date	U32_TYPE	ddmmyy	Current date
4	UTC Time	U32_TYPE	hhmmss	Current UTC time. Before the first GPS position fix, it is device's system time. During GPS position fix, it is the real UTC time from satellite. During the time when GPS position gets lost again, it is an estimated UTC time based on the history data.
5	Position Fix	U8_TYPE	-	Position Fix indicator
6	Latitude	DOUBLE	degree	
7	Longitude	DOUBLE	degree	
8	Altitude	FLOAT	degree	
9	Speed	FLOAT	Km per hour	
10	Total distance	U32_TYPE	meter	Distance is calculated based on GPS position. Its accuracy is heavily depending on quality of GPS signal. Please use "$TRSD" to further reduce its ERROR.

- **Event Value**

Event Type	Event Value	Value Meaning
EVENT_EMERG_CALL	-	-
EVENT_TAMPER	-	-
EVENT_POWER_LOW EVENT_POWER_LOST	{1, 2, 4}	1 (Battery), 2 (12VDC), 4 (5VDC)
EVENT_IGNITION	{0, 1}	0 (LOW), 1 (HIGH)
EVENT_CNTX_GPS_LOSS	{X}	X (Number of GPS loss)
EVENT_DINPUT_1	{0, 1}	0 (LOW), 1 (HIGH)
EVENT_DINPUT_2	{0, 1}	0 (LOW), 1 (HIGH)
EVENT_OVER_SPEED	{X}	X (Current Speed)
EVENT_ACTIVE_DINPUT_1	-	-
EVENT_INCOME_CALL	-	-
EVENT_MOVE_FM_STATIC	{X}	X (Current Speed)
EVENT_NO_INTERVAL	-	-
EVENT_TX_TIMEOUT	{1, 2, 4}	1(GPS report), 2 (AT cmd), 4 (Log data)
EVENT_NETWORK_FAILURE	-	-

- **Notes**
  - Alert for EVENT_EMERG_CALL has the highest priority. If it fails to be sent out or its ACK fails to be received, Device will keep resending it at intervals of 30 seconds. The alert from all other events will be blocked except that from new EVENT_EMERG_CALL.
  - Alert for other events has the same priority. They are sent in sequence.

3.13 "$SCFG" (SaVe ConFiGuration)

- **Definition**
  Save user configurations into non-volatile memory. Any configuration updates will be permanently saved only after calling this command.
- **Type**
  SET: AT$SCFG=<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Operation	U8_TYPE		Legal Values : {1, 177} Default Value: N/A ---------- 1: save configuration. 177: restore factory default. After restoring factory default, Device will **automatically** reset itself. Please note that if restore to default from remote control, the control will be lost because factory default will not work with your Backend.
E.g. (SET) AT$SCFG=1				

3.14 "$VEID" (Vehicle ID)

- **Definition**

  Assign a unique vehicle ID to Device and manage the appendant VID and checksum. Appending VID helps Backend to identify the source of received message. Appending checksum helps Backend to verify the integrity of received message.

- **Type**

  GET: AT$VEID?
  SET: AT$VEID=<Parameters>
  REPORT: $VEID:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Vehicle ID number	U16_TYPE		Legal Values : [0, 65535] Default Value: 0 ---------- 0: default vehicle ID. 1 ~ 65535: valid vehicle ID.
2	Append VID	U8_type		Legal Values : {0, 1} Default Value: 0 ---------- 0: don't append VID 1: append VID Appending VID help user to identify the sender. Recommend to turn it **ON**.
3	Append Checksum	U8_type		Legal Values : {0, 1} Default Value: 0 ---------- 0: don't append Checksum 1: append Checksum Appending checksum provides an additional way to validate received data. But It may not be necessary in actual application. Because WUC is based on UDP and UDP has much stronger checksum compared to this one. Recommend to turn it **OFF**.
E.g. (SET) AT$VEID=8000,1,1 (GET) AT$VEID? (REPORT) $VEID:8000,0,1*2D				

3.15 "$ELRQ" (Event Log data ReQuest for retrieve or erase.)

- **Definition**

  When an event happens, Device may log it into event log database depending on the settings in "$EVMN".

  This command manages the event log database. In sum, it has the following three functionalities.
  - ° Query current available number of items saved in Device with command "AT$ELRQ?"
  - ° Request to retrieve the last X number of items with command "AT$ELRQ?X"
  - ° **Erase** current event log data with command "AT$ELRQ=2"

- **Method**

  GET: AT$ELRQ? Or AT$ELRQ?<GETvalue>
  SET: AT$ELRQ=<SETvalue>
  REPORT: $ELRQ:<Parameters>

- **GETvalue**

Legal Values	Comments
[1, MAX_LOG]	It is the number of log items to be retrieved.

- **SETvalue**

Legal Values	Comments
{2}	2: erase event log database.

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Number of items allowed to be retrieved	U16_TYPE		Legal Values : {0} Default Value: N/A
2	Current available number of items saved in Device	U16_TYPE		Legal Values : [0, MAX_LOG] Default Value: 0
3	Maximum number of items can be saved in Device	U16_TYPE		Legal Values : { MAX_LOG } Default Value: MAX_LOG --- MAX_LOG=1000

3.16 "$ELDA" (retrieve Event Log DAta.)

- **Definition**

  It is a BINARY-based command when transmitting data **via UDP**. Device packs its event log data into "$ELDA" binary packet and send it out. Backend is responsible to parse received "$ELDA" packet as per its packet structure and extract the event log data. For details on how to use this command, please refer to section "Communicate Procedure". For details on the packet structure, please refer to "Packet Structure for UDP" below.

  It is an ASCII-based command when transmitting data **via UART port**. For details on the packet structure, please refer to "Packet Structure for SERIAL port" below.

- **Method**

  REPORT: <Packet Structure>

- **Packet Field**

Field Name	Type	Length (byte)	Meaning
Header	STRING	6	Constant string "$ELDA"
Sequence No	U16_TYPE	2	Packet sequence number starting from **one**.
Vehicle ID	U16_TYPE	2	A unique Device identifier.
Items	U16_TYPE	2	Number of items included in Data. • Each UDP packet has **25** items in maximum. • Each packet for SERIAL port has **3** items in maximum. • The last packet has a smaller number of items or 0 items.
Data	U8_TYPE	The remaining	Actual event log data.

- **Packet Structure for UDP**

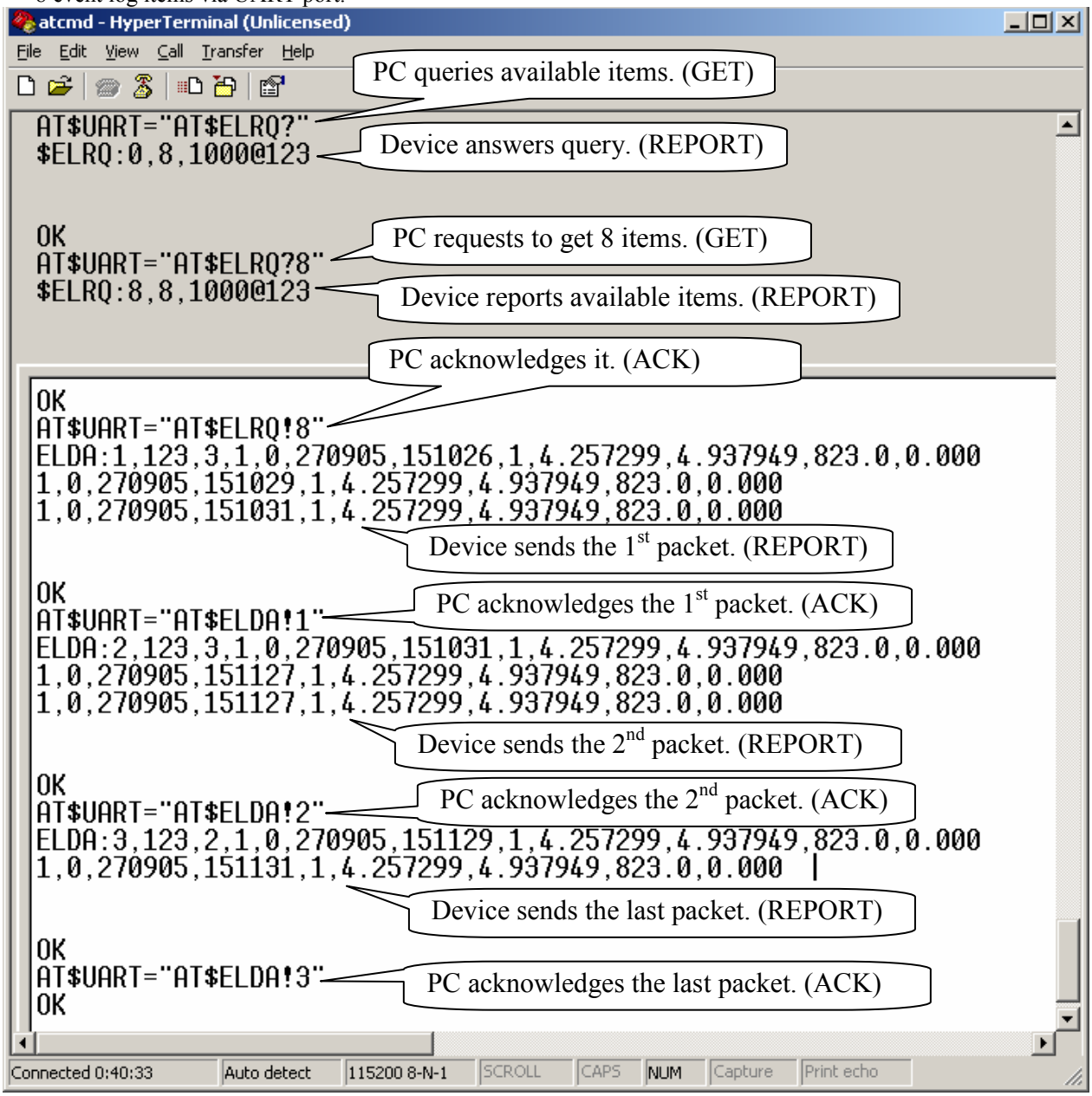
bits	0 7	8 15	16 23	24 31
Package Header	Header (0~3)			
	Header (4~5)		Sequence No	
	Vehicle ID		Items	
Data (Item One)	Event Type			
	Value			
	Date			
	UTC Time			
	Position Fix	Reserved		
	Latitude (high WORD)			
	Latitude (low WORD)			
	Longitude (high WORD)			
	Longitude (low WORD)			
	Altitude			
	Speed (Kilometers per hour)			
Data (Item Two)	(Repeat in format of "Data (Item One)") ……			

° Notes: Latitude and Longitude, and any other **double** type variables, are saved in two WORDs. Each WORD is 32-bit long in length. The WORD at the lowest address contains the sign bit, the exponent, and the most significant part of the mantissa. The WORD at the higher address contains the least significant part of the mantissa. This is true regardless of the endianness of the target. However, the ordering of bytes within each WORD depends on the endianness of the target.

° Sample: please refer to section "2.3.2 BINARY-based commands"

- **Packet Structure for UART port**
  $ELDA:<Sequence No>,<Vehicle ID>,<Items>,<Event 1><\r\n>,<Event 2><\r\n>,...,<Event n><\r\n>

  Each <Event> ends with <\r\n>. Please refer to $ALERT for <Event> format. Below is an example for retrieving 8 event log items via UART port.



3.17 "$FWVR" (FirmWare VeRsion)

- **Definition**
  Get current firmware version number. The version number is in format: **DEVICE_VERSION**.
- **Type**
  GET: AT$FWVR?
  REPORT: $FWVR: <Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Firmware Version	STRING		DEVICENAME_VERSION E.g. w606_v0.22.02 DEVICENAME: w606 VERSION: v0.22.02, 0 is major version number, 22 is minor version number, and 02 is the third number.
E.g. (GET) AT$FWVR? (REPORT) $FWVR:w606_v0.22.02				

3.18 "$RSET" (RESET Device)

- **Definition**
  Trigger a software reset. It is necessary to make new configuration effective. A normal scenario will be:
  - Update device configurations with WUC commands listed in this document;
  - Save new configurations with "$SCFG" command;
  - Reset Device with "$RSET" command and new configurations will take effect.
- **Type**
  SET: AT$RSET=<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Operation	U8_TYPE		Legal Values : {1} Default Value: N/A ---------- 1: reset device
E.g. (SET) AT$RSET=1				

3.19 "$PSWD" (PAssWorD configure)

- **Definition**
  Configure Device password. Password consists of up to 16 characters from {0,1…,9}, {a,b…,z} and {A,B…Z}. For security reason, the password is writable only. When password lost, please reset a new password via GPRS or UART port. It is assumed that GPRS/UART access is protected by users.
- **Type**
  SET: AT$PSWD=<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	#password#	STRING		Legal Values : up to 16 alphanumeric characters Default Value: default It is **case-sensitive** and must be encapsulated with '#'
E.g. (SET) AT$PSWD=#default#				

3.20 "$GSMD" (GSM Data network)

- **Definition**
  Configure parameters for GSM-Data network. When set to MODE_GSM_DATA_ONLY, Device will use these settings to dial up.
- **Type**
  GET: AT$GSMD?
  SET: AT$GSMD=<Parameters>
  REPORT: $GSMD:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Dial number	STRING		Legal Values : string of up to 32 characters Default Value: EMPTY_STR
2	ISP username	STRING		Legal Values : string of up to 24 characters Default Value: EMPTY_STR
3	ISP password	STRING		Legal Values : string of up to 16 characters Default Value: EMPTY_STR
E.g. (SET) AT$GSMD=+6596763332,a,b (GET) AT$GSMD? (REPORT) $GSMD:+6596763332,a,b*03				

3.21 "$CSMS" (Configure via SMS)

- **Definition**
  It is dedicated for configuring Device via SMS. Together with a password ($PSWD), the actual WUC command is encapsulated as one of its parameters. When device receives this command, it will first verify the password. If password is valid, it will extract and execute the actual WUC command. And then return the execution result of the actual WUC command.
- **Type**
  SET: AT$CSMS=<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Password	STRING		Legal Values : string of up to 16 characters Default Value: default Password is case-sensitive.
2	"User command"	STRING		Legal Values : string of up to 96 characters. Default Value: N/A It MUST be put into quotation mark.
E.g. (SET) AT$CSMS=default,"AT$GPRS=shwapint,,,127.0.0.1,12000"				

3.22 "$UART" (configure via UART port)

- **Definition**
  It is dedicated for configuring Device via UART port. The actual WUC command is encapsulated as its parameter. When device receives this command, it will extract and execute the actual WUC command. And then return the execution result of the actual WUC command.
- **Type**
  SET: AT$UART=<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
2	"User command"	STRING		Legal Values : string of up to 96 characters. Default Value: N/A It MUST be put into quotation mark.
E.g. (SET) AT$UART="AT$GPRS=shwapint,,,127.0.0.1,12000"				

3.23 "$GPOU" (General Purpose OUtput)

- **Definition**
  Set or clear general purpose digital output pin. The last known status will be remembered by device and restored after resetting.
- **Type**
  SET: AT$GPOU=<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Pin	U8_TYPE		Legal Values : {12, 1} Default Value: N/A --- 12: digital output pin 1 : buzzer (For testing purpose).
2	value	BOOLEAN		Legal Values : {0, 1} Default Value: N/A --- 0: set PIN to LOW 1: set PIN to HIGH
E.g. (SET) AT$GPOU=12,1 //digital output pin is set to HIGH (10~12V)				

3.24 "$GPIN" (General Purpose INput)

- **Definition**
  Read general purpose digital input pins.
- **Type**
  GET: AT$GPIN?<Pin>
  REPORT: $GPIN:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Pin	U8_TYPE		Legal Values : {10, 2, 3} Default Value: N/A ---------- 10: digital_input_1 2: digital_input_2 3: Ignition pin (For testing purpose.)
2	value	BOOLEAN		Legal Values : {0, 1, Unknown} Default Value: N/A ---------- 0: PIN is LOW 1: PIN is HIGH Unknown: invalid pin number.
E.g. (GET) AT$GPIN?2 (REPORT) $GPIN:2,0 //Digital_input_2 pin is in LOW status.				

3.25 "$MGPS " (Manage GPS)

- **Definition**

  Turn ON or OFF GPS module on the fly. By default and in normal operational mode, GPS module will always be turned ON. This configuration is just for testing purpose and will not be saved.

- **Type**

  GET: AT$MGPS?
  SET: AT$MGPS=<Parameters>
  REPORT: $MGPS:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	ON / OFF	U8_TYPE		Legal Values : {0, 1} Default Value: N/A ---------- 0: turn OFF GPS module; 1: turn ON GPS module.

E.g.
(SET)
AT$MGPS=1 //Turn ON GPS module.
AT$MGPS=0 //Turn OFF GPS module.

(GET)
AT$MGPS?

(REPORT)
$MGPS:0

3.26 "$MSCL" (Maximum Speed and Continuous Loss of GPS message)

- **Definition**

  It defines the maximum allowed speed and maximum continuous GPS loss. When vehicle exceeds the maximum, an EVENT_OVER_SPEED event is triggered. When GPS continuous loss reaches the maximum, an EVENT_CNTX_GPS_LOSS event is triggered.

- **Type**

  GET: AT$MSCL?
  SET: AT$MSCL=<Parameters>
  REPORT: $MSCL:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Maximum Speed	U16_TYPE	Kilometer per hour	Legal Values : [50, 65535] Default Value: 255 When actual speed is **GREATER than** this value, event EVENT_OVER_SPEED will be triggered. Please refer to $EVMN for event management.
2	Maximum continuous GPS loss	U8_TYPE		Legal Values : [4, 100] Default Value: 4 The maximum continuous GPS loss.
3	Speed to judge movement	U8_TYPE	Kilometer per hour	Legal Values : {0, [1, 20]} Default Value: 0 ---------- 0 : disabled Event EVENT_MOVE_FM_STATIC is judged based on this value. If actual speed exceeds this value, Device is treated as moving, otherwise stationary.

E.g.
(SET)
AT$MSCL=200, 10

(GET)
AT$MXSP?

(REPORT)
$MXSP: 200, 10

3.27 "$VSDM" (Vehicle Signal Delay Monitor)

- **Definition**

  It is an optional feature. If it is enabled, device will raise an event in the following condition:
  - Device detected an active signal in ignition pin;
  - After delay_time seconds, Device detected an active signal in digital_input_one pin.

- **Type**

  GET: AT$VSDM?
  SET: AT$VSDM=<Parameters>
  REPORT: $VSDM:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Enable/Disable	U8_TYPE		Legal Values : {0, 1} Default Value: 0 ---------- 0: disable; 1: enable
2	Ignition Active	U8_TYPE		Legal Values : {0, 1} Default Value: 0 ---------- 0: active LOW 1: active HIGH
3	Digital_input_1 Active	U8_TYPE		Legal Values : {0, 1} Default Value: 0 ---------- 0: active LOW 1: active HIGH Totally there are two digital input pin. This parameter refers to the 1st digital input pin.
4	Delay time	U8_TYPE	minutes	Legal Values : [1, 255] Default Value: 5
E.g. (GET) AT$VSDM? (REPORT) $MXSP:0,,,*30				

3.28 "$GSPW " (Gsm Signal and DC/battery PoWer.)

- **Definition**
  Return GSM signal strength and the power status of DC/Battery.
- **Type**
  GET: AT$GSPW?
  SET: AT$GSPW=<Parameters5>,< Parameters6>,< Parameters7>
  REPORT: $GSPW:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	GSM Signal strength	U32_TYPE	dBm	Strength of GSM signal.
2	Batter Level	U8_TYPE	level	Valid Values : {0,1,2,3,4} Default Value: N/A ---------- 4: Full battery. … 1: Low battery. 0: Running out of battery.
3	Voltage of 12V DC power	U16_TYPE	mili-volt	Indicate voltage of 12V DC power in mili-volt.
4	Voltage of 5V DC power	U16_TYPE	mili-volt	Indicate voltage of 5V DC power in mili-volt.
5	Threshold for 12V DC power low	U16_TYPE	mili-volt	When voltage of 12V DC is lower than this value, raise EVENT_POWER_LOW It must be bigger than “Threshold for 12V DC power loss”.
6	Threshold for 12V DC power loss	U16_TYPE	mili-volt	When voltage of 12V DC is lower than this value, raise EVENT_POWER_LOSS
7	Threshold for 5V DC power low	U16_TYPE	mili-volt	When voltage of 5V DC is lower than this value, raise EVENT_POWER_LOW

3.29 "$TRSD" (ThReSholDs for distance and true course)

- **Definition**
  It is used to get suitable "Distance Threshold" and "COG Threshold" for WUC "$DSCG".
- **Type**
  GET: AT$TRSD?
  REPORT: $TRSD:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Current Distance Delta	float	meter	Legal Values : [0, 255] Default Value: N/A Current distance delta based on 2-second calculation. To get a suitable "Distance Threshold": o Keep Device stationary in open air; o Boot up Device and wait for around 2 minute for GPS position fix; o Call "$TRSD" in every 10 seconds for around 10 times. o Do arithmetical average on the 10 readings of "Current Distance Delta"; o Set a "Distance Threshold" value suitable enough to filter the average distance delta.
2	Current COG Delta	float	degree	Legal Values : [0, 255] Default Value: N/A Current true course delta based on 2-second calculation. To get a suitable "COG Threshold": o Keep Device moving along a geographic straight line in open air; o Boot up Device and wait for around 2 minute for GPS position fix; o Call "$TRSD" in every 10 seconds for around 10 times. o Do arithmetical average on the 10 readings of "Current COG Delta"; Set a "COG Threshold" value suitable enough to filter the average COG delta.

3.30 "$VODO" (Virtual ODOmeter)

- **Definition**
  Provide part of odometer features.
- **Method**
  GET: AT$VODO?
  SET: AT$VODO=<SETvalue>
  REPORT: $VODO:<Parameters>
- **SETvalue**

Legal Values	Comments
{1}	1: reset total distance.

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Device Uptime	U32_TYPE	second	Valid Values : [0, 4294967296] Default Value : N/A Time passed by since device boots up.
2	Total Distance	U32_TYPE	meter	Valid Values : [0, 4294967296] Default Value : N/A Total distance traveled since device gets its first valid GPS message. It can be reset by SET command.
3	Current Speed	U16_type	Kilometers per hour	Current speed parsed from GPS message.

3.31 "$SPIN" (Save SIM PIN)

- **Definition**

  Save SIM PIN into flash. This command is valid only for UART-based configuration. When issuing this command via UDP or SMS, it will simply return command fail.

- **Method**

  GET: AT$SPIN?
  SET: AT$SPIN =<Parameters>
  REPORT: $SPIN:READY

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	#SIM PIN#	STRING		Legal Values : string with 4 to 8 characters Default Value: 0000 Please make sure this PIN is CORRECT PIN for your SIM card.

E.g.
(GET)
AT$SPIN?

(REPORT)
$SPIN:READY //When SIM card is ready. Otherwise, error is reported by $STER.

(SET)
AT$SPIN=#1234# //Save 4-digit SIM PIN 1234 into flash.

- **Example for SIM PIN**

  Suppose there is a SIM card with SIM PIN deactivated by default. Its default SIM PIN is "1234". Below is an example for

  (1) How to check SIM card status?
  (2) How to check if SIM PIN is activated or not?
  (3) How to activate SIM PIN?
  (4) How to change SIM PIN?
  (5) How to manually unlock SIM card?
  (6) How to save SIM PIN into Device? ($SPIN)
  (7) How to verify if saved SIM PIN works properly?

  Please note AT commands AT+CPIN, AT+CLCK, AT+CPWD are standard GSM AT command. Refer to Appendix F for more details.

  == Sample ==========
  AT-Command Interpreter ready

  // (1) How to check SIM card status?
  AT+CPIN?
  +CPIN: READY

  OK

  // (2) How to check if SIM PIN is activated or not?
  AT+CLCK="sc",2 // 2==query
  +CLCK: 0 // 0==deactivated, 1==activated.

  OK

```
// (3) How to activate SIM PIN?
AT+CLCK="sc",1,"1234"                  // 1==activate, 0==deactivate
OK

AT+CLCK="sc",2
+CLCK: 1

OK

// (4) How to change SIM PIN?
// Change SIM PIN from default “1234” to “56789”
AT+CPWD="sc","1234","56789"          // “1234” is current SIM PIN, “56789” is new SIM PIN
OK

//Reset device
AT$UART="AT$RSET=1"
$RSET!

AT-Command Interpreter ready

// Check SIM PIN status again after reset. It is waiting SIM PIN to be given
AT+CPIN?
+CPIN: SIM PIN

OK

// (5) How to manually unlock SIM card?
AT+CPIN="56789"                        // Enter “56789” to unlock SIM card.
OK

// Check if unlock successfully.
AT+CPIN?
+CPIN: READY

OK

// (6) How to save SIM PIN into Device? Please make sure the SIM PIN saved is absolutely correct.
AT$UART="AT$SPIN=#56789#"          // Save SIM PIN “56789” into Device for future auto-unlocking.
$SPIN!

OK

//Reset device
AT$UART="AT$RSET=1"
$RSET!
AT-Command Interpreter ready

// (7) How to verify if saved SIM PIN works properly?
// Device will unlock SIM card automatically with saved SIM PIN. Check if unlock successfully.
AT+CPIN?
+CPIN: READY

OK
```

3.32 "$TEST" (TEST device)

- **Definition**
  Provide information and functionalities for debug purpose.
- **Method**
  GET: AT$TEST?
  SET: AT$TEST=<SETvalue>
  REPORT: $TEST:<Parameters>
- **SETvalue**

Legal Values	Comments
{12}	12: send an UDP packet.

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	GPS RX counter	U32_TYPE		Valid Values : [0, 2^{32}-1] Default Value : N/A The count of received GPS NMEA messages. [Typical usage] check if GPS module works. When it keeps increasing, GPS module works. Otherwise if it keeps constant, GPS module does not work.
2	Dummy			
3	Dummy			
4	Dummy			
5	Dummy			
6	Dummy			
7	Dummy			

3.33 "$RTCF" (RunTime ConFiguration)

- **Definition**
  It configures some of device parameters at runtime. These parameters can not be saved.
- **Method**
  GET: AT$RTCF?
  SET: AT$RTCF=<Parameters>
  REPORT: $RTCF:<Parameters>

- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Reserved			

3.34 "$STER" (STATus or ERror message reporting)

- **Definition**
  It handles system status and error message.
- **Type**
  GET: AT$STER?
  REPORT: $STER:<Parameters>
- **Parameters**

Id	Parameters	Type	Unit	Comments
1	Status / Error	U8_TYPE		Legal Values : [0, 99] Default Value: N/A See table “System Status” and “Error Message” for details
2	Human string	STRING		A string helps to quickly understand the meaning of error code. 50: “Unknown cmd”, 51: “GET unsupported” 52: “SET unsupported” 53: “Access log fail” 54: “Device busy” 55: “Func not started” 56: "Invalid parameter(s)" 57: "Cmd failed" 58: "SIM PIN locked!" 59: "PUK PIN locked!!!" 60: "SIM card undetected"

System Status (0~49)		
Numeric	**Aliases**	**Comments**
0	SYS_START_IDLE	Device is just started or kept in idle status.
1	SYS_NO_NETWORK	No SIM card.
2	SYS_NETWORK_ATTACHED	Device is attached to GSM network.
4	SYS_NETWORK_CONNECTING	Device is connecting to GPRS or GSM-Data network.
5	SYS_NETWORK_CONNECT_FAIL	Fail in connecting to GPRS or GSM-Data network.
6	SYS_NETWORK_CONNECTED	Successful in connecting to GPRS or GSM-Data network.
8	SYS_NETWORK_DISCONNECTED	GPRS or GSM-Data network gets disconnected after a successful connection.
10	SYS_ON_INCOMING_CALL	Device is handling an incoming GSM call.
12	SYS_ON_OUTGOING_CALL	Device is handling an outgoing GSM call.
14	SYS_SMS_MODE_STARTED	Device runs into MODE_SMS_ONLY mode.
15	SYS_SERIAL_MODE_STARTED	Device runs into MODE_SERIAL_ONLY mode.
16	SYS_MODEM_MODE_STARTED	Device runs into MODE_MODEM_ONLY mode.
49	SYS_UNKNOWN_STATUS	Device is not initialized yet.

Error Message (50~99)		
Numeric	**Aliases**	**Comments**
50	ERR_UNKNOWN_COMMAND	Unknown command or wrong format.
51	ERR_CMD_READ_UNSUPPORTED	Command does not support GET operation.
52	ERR_CMD_WRITE_UNSUPPORTED	Command does not support SET operation.
53	ERR_ACCESS_LOG_FAIL	Invalid access or fail to access to log data.
54	ERR_DEVICE_BUSY	Device is busy in handling another command.
55	ERR_FUNC_NOT_STARTED	Fail to initialize Device. No operation is allowed.

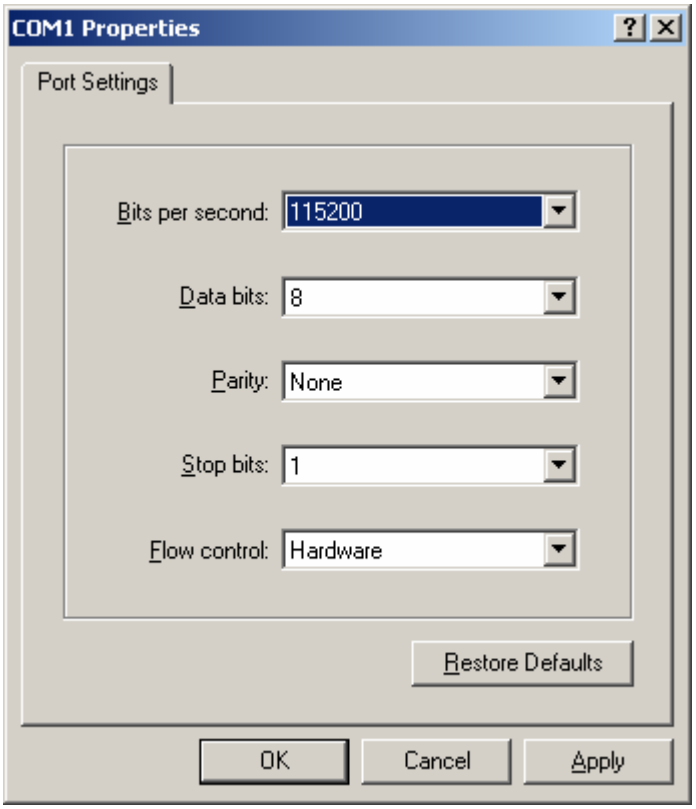
56	ERR_INVALID_PARAMETER	Command has invalid parameters.
57	ERR_CMD_FAILED	Command execution failed.
58	ERR_SIM_PIN_LOCKED	Device is currently locked by SIM PIN.
59	ERR_PUK_PIN_LOCKED	Device is currently locked by PUK PIN.
60	ERR_SIM_CARD_UNDETECTED	No SIM card is installed or SIM card is not detected.

Appendix A: Device Basic Flowchart

Device has different behavior in different operational mode (See $OPMD). This appendix will give you a general idea on how Device works in each mode.

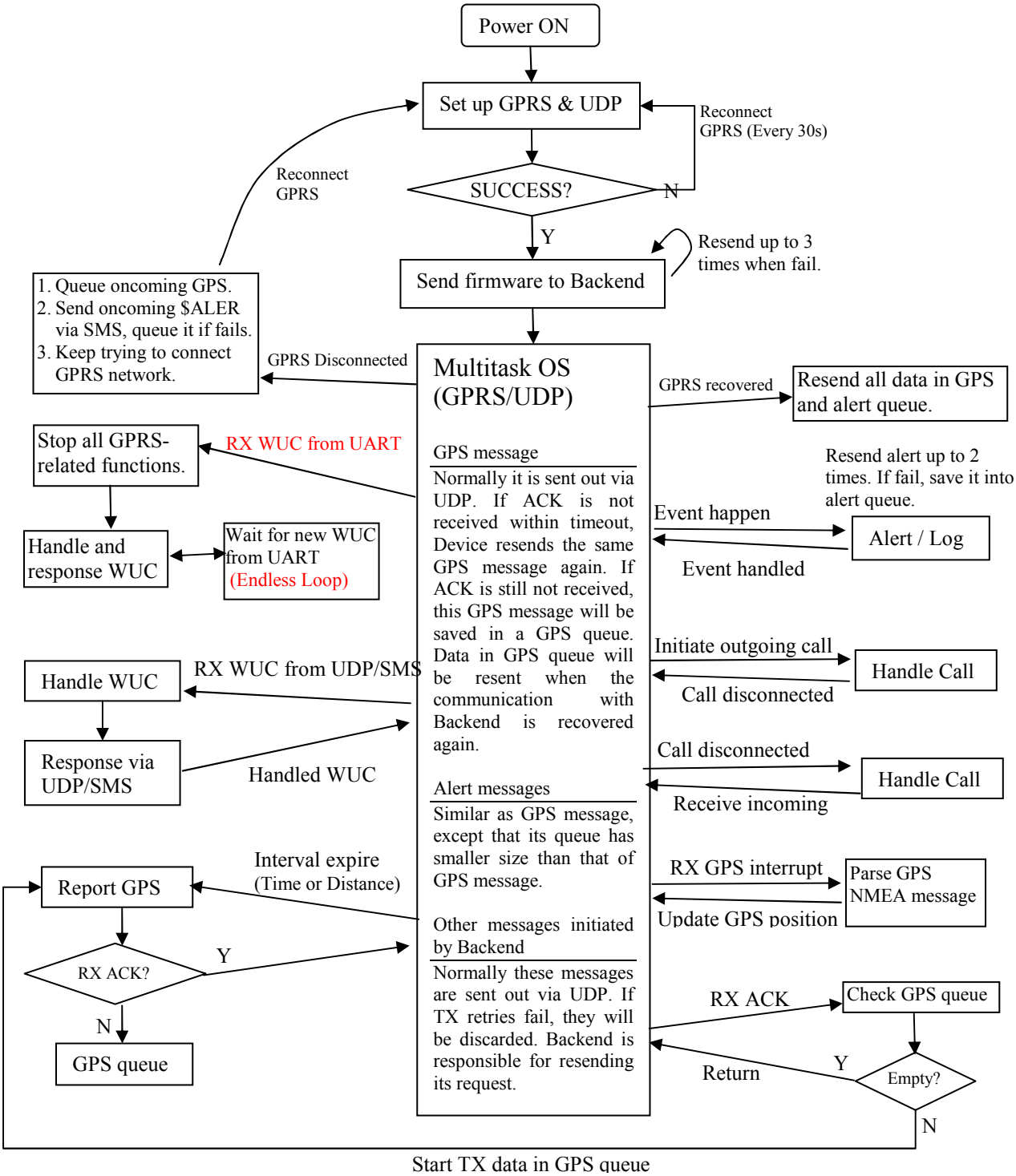
A.1. MODE_MODEM_ONLY

In this mode, Device works as a standard GPRS modem. Its behavior is the same as that of standard GPRS modem. Except it supports not only normal AT commands, but also WUC commands listed in this document. PC's COM port configuration is as shown in FigA.1.



FigA.1 PC's COM port configuration

A.2. MODE_GPRS_ONLY



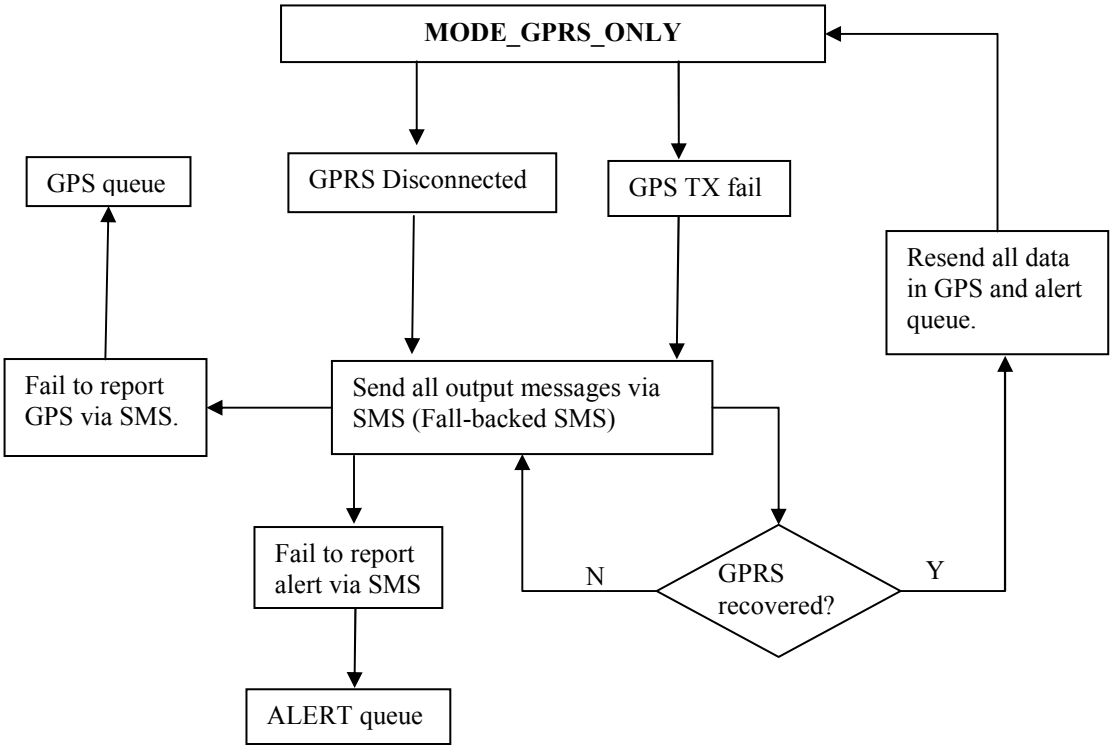
Notes:

- Emergency alert has highest priority. They can not be overwritten by others. While others can overwrite each other.
- After "RX WUC from UART", device enters **AT command** mode. All GPRS-related functionalities will be stopped. To return back the normal status, a **Software Reset** is required.

FigA.2 Flowchart of MODE_GPRS_ONLY

A.3. MODE_GPRS_SMS

It adds SMS fall back feature based on MODE_GPRS_ONLY. Its flowchart is shown in Fig 7.



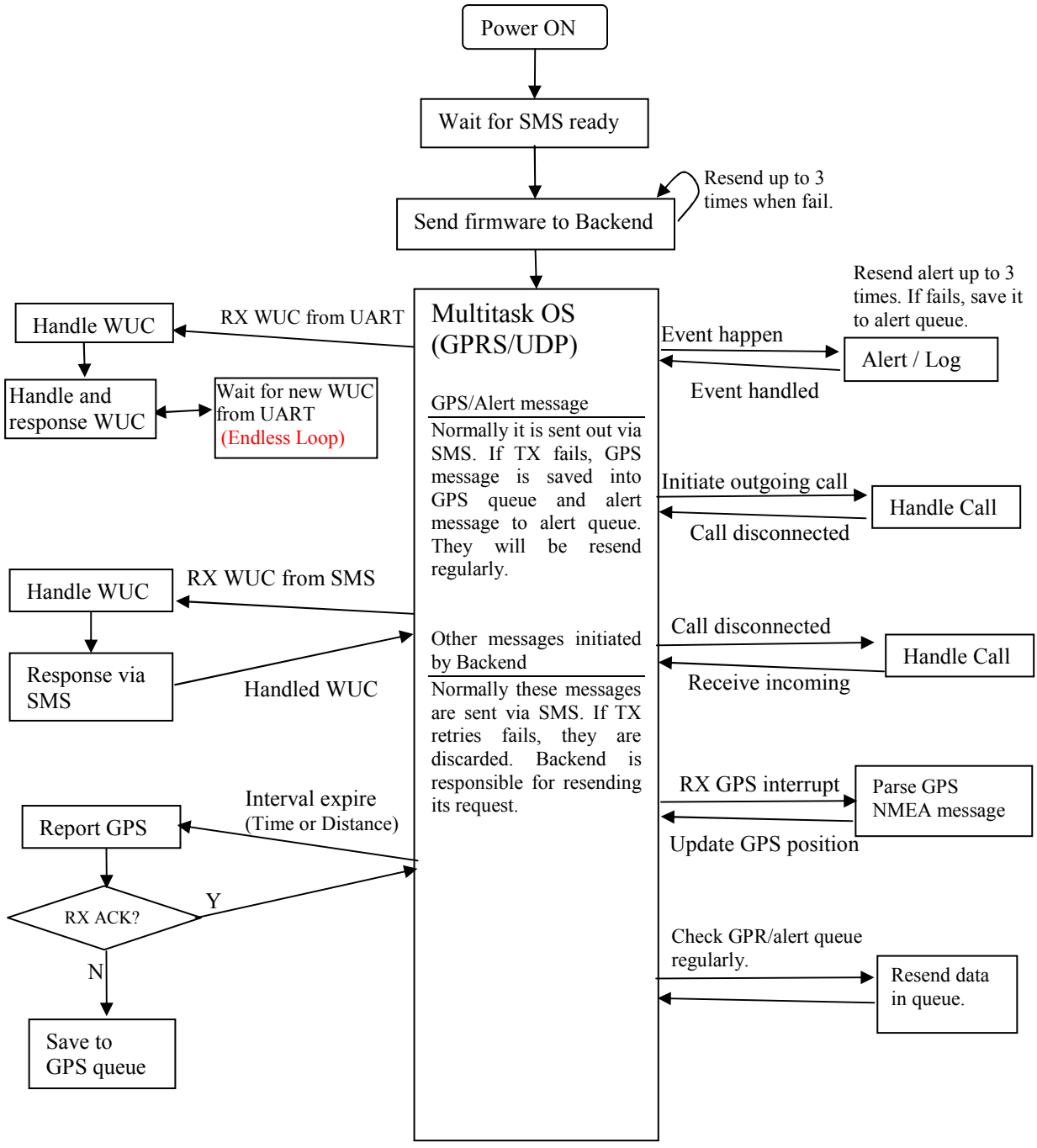
Notes:

- Except GPS and alert message, other message will be discarded when SMS retry fails.

FigA.3 Flowchart of MODE_GPRS_SMS

A.4. MODE_SMS_ONLY

In this mode, Device's flowchart is shown in Fig8.



Notes:

- It only has the basic functionalities and is not as powerful as GPRS-based mode.

FigA.4 Flowchart of MODE_SMS_ONLY

A.5. GSM_DATA_ONLY_MODE

This mode has exactly the same functionalities as MODE_GPRS_ONLY. It is useful in certain region where GPRS network is not available but GSM-Data network does. You may treat it as backup mode.

Its flowchart is also the same as MODE_GPRS_ONLY, except that it uses GSM-Data to connect internet. Please refer to “MODE_GPRS_ONLY” for details.

Appendix B: GPS/Alert Queue

Sometimes GPS/Alert messages may not be able to reach Backend because of various reasons. For example, Device got disconnected from GPRS network, GPRS signal is temporarily missing, Backend Server is rebooting, and GPRS network is in low quality etc. The feature of GPS/Alert queue is an effective way to protect each GPS position and alert message from being lost.

Current GPS queue can buffer up to **150** GPS positions. Whenever GPS TX fail or GPRS is disconnected, the current and oncoming **valid** GPS positions will be added into GPS queue. As soon as the communication between Device and Backend gets recovered, all GPS positions in queue will be resent.

Current Alert queue can buffer up to **40** alert messages. Its procedure is similar to GPS's.

Appendix C: A Simple Server Example in C language

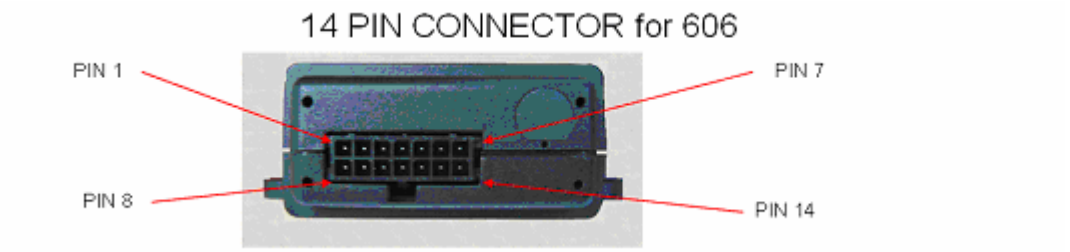
The source code below is part of a simple server for Infowave Waveon606 serial product. It gives you an idea on how simple its server can be. Also, it may help you quickly implement a server to evaluate message from Device. Please note this sample server does not support $ELDA.

```
/* Sample written in C language */

/* START >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*/
char *atcmd = NULL;
char buffer[512];
char response[48];
int respLength = 0;

while(1) {
    memset((void *)buffer, 0, sizeof(buffer));
    memset((void *)response, 0, sizeof(response));

    /* Receive UDP packet by polling */
    if(recvfrom(udpSocket, buffer, sizeof(buffer), 0, (struct sockaddr*)&from, &fromlen) != SOCKET_ERROR)
    {
        printf("%s", buffer);  //Print the string received from the Device.

        /* Handle received Message */
        //E.g. convert $PSVL to NMEA $GPGGA.

        /* Send back acknowledgement if required. */
        atcmd = strchr (buffer, '$');          //Search for string started with '$'
        if ( NULL != atcmd ) {                 //If found, pack the ACK packet.
            strcpy(response,"AT");             //Add "AT" prefix.
            strncat(response, atcmd, 5);       //Append WUC command "$XXXX" read from RX buffer.
            strcat(response, "!\r\n");         //Append <CR><LF> required by WUC format.
            respLength = strlen(response);     //Get length of ACK packet.

            /* Acknowledgement */
            sendto(udpSocket, response, respLength, 0, (struct sockaddr*)&from, fromlen);
        }
    }

    Sleep(100);         //Sleep 100ms.
}

/* <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< END */
```

Appendix D: Wavelet LED Characteristics

	Module Status	LED Activity (Unit: millisecond)
1.	Switching OFF Device (click POWER button onboard)	**Two short ON + one long ON + one SPARK** ON(200)/OFF(200)/ON(200)/OFF(200) + ON(800)/OFF(3000) + SPARK
2.	No GSM network	**"Permanently" OFF** The "Slow Flashing" (see 6) in background will be turned OFF.
3.	Searching for GSM network	**Permanently ON**
4.	Successful TX via GPRS or SMS	**One or two SPARK with constant total life-span.** The actual behavior depends on GPS status. • When GPS position is valid Two SPARK: ON(20)/OFF(80)/ON(20)/OFF • When GPS position is not valid One SPARK: ON(120)/OFF
5.	Buffered into GPS queue	**The same as item "Successful TX via GPRS or SMS"**
6.	GSM network OK (Background)	**Slow Flashing**. On for 20, OFF for 9000
7.	Received incoming call	**Fast Flashing**. On for 100, OFF for 900
8.	Fatal Error (irrecoverable in field)	**Isochronous Fast Flashing**. ON for 500, OFF for 500

Appendix E: Description of 14-PIN Connector and Cable

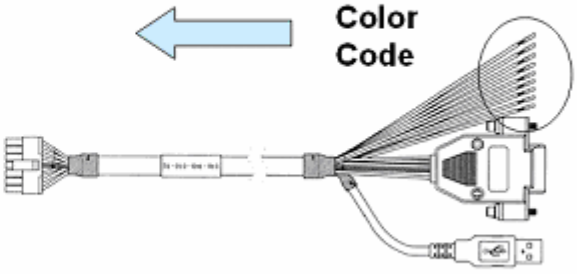


14 PIN	SIGNAL NAME	DIR	POLARITY	DESCRIPTION
1	EMERGENCY	I	Active HIGH	EMERGENCY CALL 1
2	DIGITAL IN 1	I	PROG	DIGITAL IN 1
3	DIGITAL OUT	O	PROG	DIGITAL OUT
4	CALL 2	I	Active HIGH	CALL 2
5	UART-RTS	I	-	UART
6	UART-TX	I	-	UART
7	DIGITAL IN 2	I	PROG	DIGITAL IN 2
8	+5V	-	-	USB Charger
9	+12V	-	-	POWER SUPPLY INPUT
10	RINGER	O	PROG	RINGER or DIGITAL OUT 2
11	GND	-	-	GROUND
12	UART-CTS	O	-	UART
13	UART-RX	O	-	UART
14	IGNITION	I	PROG	IGNITION INPUT

NOTE: HIGH: 10V~12V; LOW: 0V~0.8V
Output pin (3, 10) only function when +12V presented on Pin9

14 PIN Cable for 606

Color	Pin No.	SIGNAL NAME
Dark Blue	1	EMERGENCY
Light Green	2	DIGITAL IN 1
Light Blue	3	DIGITAL OUT
Light Purple	4	CALL 2
Gray	7	DIGITAL IN 2
Red	9	+12V
Purple	10	RINGER
Black	11	GND
Pink	14	IGNITION



Appendix F: $SPIN related GSM AT Commands

ME	Mobile Equipment	Refers to the GSM engines
MS	Mobile Station	Refers to the GSM engines
DTE	Data Terminal Equipment	Refers to the host terminal/application in control
DCE	Data Communication Equipment	Refers to the device controlled by the host

F.1. Enter PIN +CPIN

Description: This command is used to enter ME a password that is necessary before it can be operated (SIM PIN, SIM PUK, PH-SIM PIN, etc.). If the PIN is to be entered twice, the TA shall automatically repeat the PIN. If no PIN request is pending, no action is taken towards ME and an error message is returned to TE.

If the PIN required is SIM PUK or SIM PUK2, the second pin is required. This second pin, <newpin>, is used to replace the old pin in the SIM.

Read command returns an alphanumeric string indicating whether some password is required or not. It is up to application to validate the PIN status every time ME is reset or power on.

Syntax: AT+CPIN=<pin>

Command	Possible response(s)
AT+CPIN=<pin>[,<newpin>]	*+CME ERROR: <err>*
AT+CPIN?	+CPIN: <code> *+CME ERROR: <err>*
AT+CPIN=?	
AT+CPIN?	+CPIN: SIM PIN
AT+CPIN="1234" Note: enter SIM PIN	OK Note: SIM PIN is correct

Defined values:
<pin>, <newpin>: string type values

<code> values reserved by the present document:

READY	ME is not pending for any password
SIM PIN	ME is waiting SIM PIN to be given
SIM PUK	ME is waiting SIM PUK to be given
PH-SIM PIN	ME is waiting phone-to-SIM card password to be given
PH-FSIM PIN	ME is waiting phone-to-very first SIM card password to be given
PH-FSIM PUK	ME is waiting phone-to-very first SIM card unblocking password to be given
SIM PIN2	ME is waiting SIM PIN2 to be given (this <code> is recommended to be returned only when the last executed command resulted in PIN2 authentication failure (i.e. +CME ERROR: 17); if PIN2 is not entered right after the failure, it is recommended that ME does not block its operation)
SIM PUK2	ME is waiting SIM PUK2 to be given (this <code> is recommended to be returned only when the last executed command resulted in PUK2 authentication failure (i.e. +CME ERROR: 18); if PUK2 and new PIN2 are not entered right after the failure, it is recommended that ME does not block its operation)
PH-NET PIN	ME is waiting network personalization password to be given
PH-NET PUK	ME is waiting network personalization unblocking password to be given
PH-NETSUB PIN	ME is waiting network subset personalization password to be given
PH-NETSUB PUK	ME is waiting network subset personalization unblocking password to be given
PH-SP PIN	ME is waiting service provider personalization password to be given
PH-SP PUK	ME is waiting service provider personalization unblocking password to be given
PH-CORP PIN	ME is waiting corporate personalization password to be given
PH-CORP PUK	ME is waiting corporate personalization unblocking password to be given

F.2. Facility lock +CLCK

Description: This is used to lock, unlock or interrogate a ME or a network facility <fac>. Password is normally needed to do such actions.

Syntax: AT+CLCK=<fac>,<mode>[,<"passwd">[,<class>]]

Command	Possible response(s)
AT+CLCK=<fac>,<mode>[,<"passwd">[,<class>]]	*+CME ERROR: <err>* when <mode>=2 and command successful: +CLCK: <status>[,<class1> [<CR><LF>+CLCK: <status>,<class2> [...]]
AT+CLCK=?	+CLCK: (list of supported <fac>s) *+CME ERROR: <err>*
AT+CLCK="SC",1,"1234"	OK Note: SIM lock enabled
AT+CPIN="1234"	OK Note: Correct PIN entered
AT+CLCK="SC",0,"1234"	OK Note: SIM lock disabled
AT+CLCK="AO",2 Note: Query BAOC status	+CLCK: 1,1 OK Note: BAOC is active
AT+CLCK=?	+CLCK: ("SC","AO","OI","OX","AI","IR","AB","AG","AC","FD","PS","PN","PU","PP","PC","PF")

Defined values:
<fac> values reserved by the present document:
"SC" SIM (lock SIM card) (SIM asks password in ME power-up and when this lock command issued)
"AO" BAOC (Barr All Outgoing Calls)
"OI" BOIC (Barr Outgoing International Calls)
"OX" BOIC-ex HC (Barr Outgoing International Calls except to Home Country)
"AI" BAIC (Barr All Incoming Calls)
"IR" BIC-Roam (Barr Incoming Calls when Roaming outside the home country)
"AB" All Barring services (refer GSM 02.30 [19]) (applicable only for <mode>=0)
"AG" All outgoing barring services (applicable only for <mode>=0)
"AC" All incoming barring services (applicable only for <mode>=0)
"FD" SIM fixed dialing memory feature (if PIN2 authentication has not been done during the current session, PIN2 is required as <passwd>)
"P2" SIM PIN 2

<mode>:
0 unlock
1 lock
2 query status

<status>:
0 not active
1 active

<passwd>: string type; shall be the same as password specified for the facility from the ME user interface or with command Change Password +CPWD

<class*x*> is a sum of integers each representing a class of information (default 7):
1 voice (telephony)
2 data (refers to all bearer services; with <mode>=2 this may refer only to some bearer service if TA does not support values 16, 32, 64 and 128)
4 fax (facsimile services)
8 short message service
16 data circuit sync
32 data circuit async
64 dedicated packet access
128 dedicated PAD access

F.3. Change password +CPWD

Description: This command sets a new password for the facility lock function defined by command Facility Lock +CLCK.

Syntax: AT+CPWD=<fac>,<"oldpwd">,<"newpwd">

Command	Possible response(s)
AT+CPWD=<fac>,<oldpwd>,<newpwd>	*+CME ERROR: <err>*
AT+CPWD=?	+CPWD: list of supported (<fac>,<pwdlength>)s *+CME ERROR: <err>*
AT+CPWD="SC","1234","6789"	OK Note: SIM PIN changed from 1234 to 6789
AT+CPWD=?	+CPWD: ("SC",4),("AO",4),("OI",4),("OX",4),("AI",4),("IR",4),("AB",4),("AG",4),("AC",4),("P2",4)

Defined values:
<fac>:
"P2": SIM PIN2 refer Facility Lock +CLCK for other values

<oldpwd>, <newpwd>: string type; <oldpwd> shall be the same as password specified for the facility from the ME user interface or with command Change Password +CPWD and <newpwd> is the new password

<pwdlength>: integer type maximum length of the password for the facility