Easy GPRS User Guide

80000ST10028 Rev. 0 - 02/01/07

Making machines talk.

This document is relating to the following products:

GM862-GPS	3 990 250 657

GE863-GPS	3 990 250 660

GT863-PY	3 990 250 466

GM862-QUAD	3 990 250 655
GM862-QUAD-PY	3 990 250 656
GM862-QUAD	3 990 250 659
GM862-QUAD-PY	3 990 250 658

GE863-OUAD Pb balls	3 990 250 664
GE863-PY Pb balls	3 990 250 665
GE863-QUAD Pb free	3 990 250 653
GE863-PY Pb free	3 990 250 654
GE863-QUAD Pb free	3 990 250 662
GE863-PY Pb free	3 990 250 661

GE864-QUAD	3 990 250 675
GE864-PY	3 990 250 676

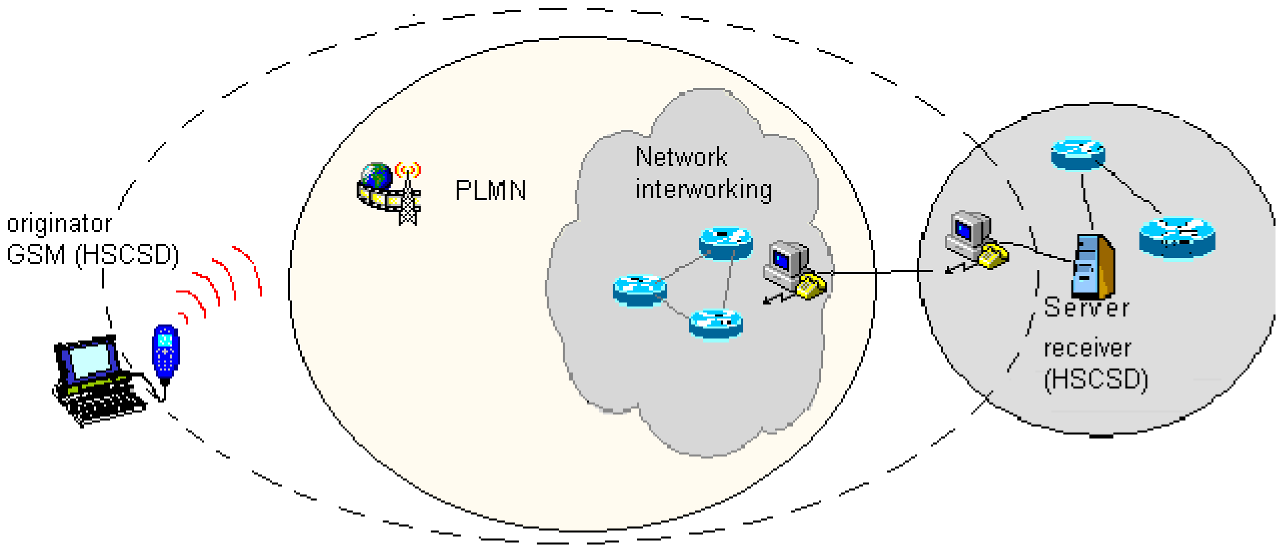
GE863-QUAD	3 990 250 651
GE863-PY	3 990 250 650

Contents

1 GPRS Operations

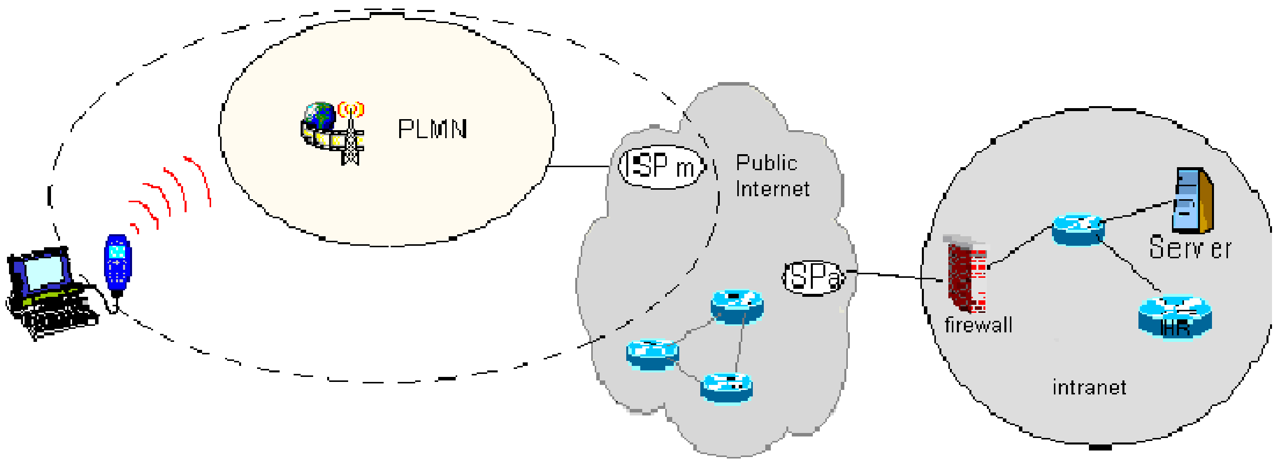
1.1 Introduction

The General Packet Radio Services (GPRS) standard permits DATA transfers in a completely different way with respect to previous point to point communications made with Circuit Switch Data (CSD) GSM modems.
In CSD operations the modem establishes a connection with the other party (another modem) in such a way that all the Network devices in between are transparent to the data exchanged, simulating a real point to point connection, just as if the other party is directly connected with the controlling application of the modem. The other party can be either an Internet Service Provider (ISP) or a private server, but in any case, the arrival point must have a modem to connect to (Landline, ISDN or GSM CSD). The connection establishment procedure defines a particular path where all the information exchanged between the two peers flows and this path is reserved for exclusive use of these 2 peers for all the time the connection is active.
This approach has the drawbacks of a long time to set-up the link between the two peers (up to a minute) and a time counting bill which proceeds even if no data is exchanged because the path resources are reserved anyway; furthermore the speed of the data transfer is limited to 14400 bps.
An example of this kind of operation is shown in the following picture, where the point to point connection is between the two peers as if all the devices inside the dashed line are not present:



CSD interconnectivity

In GPRS operations instead, the connection is made directly towards internet as if the GPRS modem was a network IP socket interface. There's no data path reserved for the data exchange between the two peers, instead the resources are allocated dynamically on demand and the data exchanged is organized into packets typically TCP/IP, furthermore the maximum transfer speed can be much faster than GSM CSD.
An example of GPRS connection is shown in the following picture, where the GPRS connection is between the GPRS modem and the internet as if all the devices inside the dashed line are not present:



GPRS interconnectivity

Due to this kind of connection, when activating the GPRS connection you must provide the network parameters to enter through the internet point of the GPRS network ISP (Internet Service Provider) and not the phone number to be dialed; therefore it is not possible to establish a direct point to point GPRS connection between two modems as in CSD case, instead an internet tunneling must be done to achieve a point to point connection between two peers.
This approach as the immediate advantage of projecting the controlling application of the GPRS modem directly on the internet, ready to be accessed virtually from anywhere in the world at the same cost on the GPRS; actually the billing of the GPRS connection is based on the amount of data exchanged (number of packets transferred) independently from the time the connection is active or where these packet must be delivered. Therefore, it is possible to leave the controlling application always connected and ready to receive/send data on demand, while paying only for the data really exchanged.
The drawback of the GPRS connection is that the controlling application must have its own TCP/IP protocol stack embedded to decode the packets that arrive from GPRS and encode the ones to be sent through the internet.
There are few considerations than must be done on the GPRS connections:

- the GPRS connection speed with a GPRS class 10 multislot device is asymmetrical, 3 time slots in reception (43200 bps max) and 2 time slot in sending (28800 bps max) or 4 time slots in reception (57600 bps max) and 1 time slot in sending (14400 bps max).
- The controlling application of the module must have a TCP/IP - PPP software stack to interface with the GPRS modems.

- The controlling application must relay on some ISP that may be the Network Operator of the SIM to gain access to the internet through the GPRS connection.
- Because of the point before, the receiving application must have internet access either.
- Since the communication is based upon TCP/IP packets, then it is possible to talk contemporarily with more than one peer.
- When required, the data security in internet shall be guaranteed by security protocols over the TCP/IP that must be managed by the controlling application.

A modem can be in 4 different states:

- GPRS DETACHED, which corresponds to the "not reachable" condition for the GPRS service;
- GPRS ATTACHED, which corresponds roughly to the "registered" condition for the GPRS service;
- GPRS context activated, which corresponds to the “reachable on the network” condition with IP address assigned (this is possible with AT command: AT#GPRS=1 and also some other AT commands)
- CONNECTED, which roughly corresponds to the connected status;

A thing that must be noted on the GPRS connect, is the fact that, if the mobile IP address (the internet address) is assigned by the ISP dynamically, then when the GPRS context of the device is not activated it has no address and therefore it cannot be reached by internet requests. The same thing occurs in the case the GPRS device has a static IP address assigned to it by the ISP, but it is DETACHED.
In these cases there's no possibility for the internet peer to "call" the GPRS device through internet, the only way to alert it is to call it in GSM mode (either a Data or a Voice call are suited) and the GPRS module application must recognize the caller, eventually abort the GSM call and connect to the internet in GPRS to receive the packets from the internet peer.

NOTE: Mobile device can be reachable from internet network only if the IP assigned by the operator is public IP; not all operators offer this service.

To explain further the differences between CSD and GPRS an example application made in both ways will be shown.

1.1.1 CSD application example

Let’s suppose you have several remote meteorological measurement units spread around the territory, and you want to access them wirelessly through a GSM module in CSD operation.
For each remote unit, there's a modem to connect with the server application, with its own SIM card and unique phone number.
Now there are two possibilities:

- the server application calls on demand the remote units, provided it has stored their phone numbers in a private database.

- the remote units call the server application modem when needed and eventually retry in the case they found it busy; this time the phone number to be stored is only one, the server number which must be stored on the remote units.

In both cases, once connected, the remote unit sends the meteorological data to the server, which places it in a central database for further reading by anyone who accesses the meteorological internet site for example.
The drawback of this approach is that the CSD modem needs about 30s to establish the connection and, depending on the amount of data to be transferred (usually few hundreds bytes), some seconds to transfer them. So let’s say we pay a 40s call while we need only 10s to transfer data.

1.1.2 GPRS application example

The same application can be preformed with all the Telit modules using the GPRS feature.
The remote unit is always connected to the internet taking advance of the features of the GPRS system, when it needs to send data to the server application it simply fills the TCP/IP packets for the server with the meteorological data and gives them to the Telit module to deliver. The central server has a single modem to connect to the internet, receives the TCP/IP packets from all the remote units and places the contained data in the central database.
The advantage of using GPRS is that the remote unit is always connected and reachable and it pays only for the amount of data (small) transferred and not for the connection time as in CSD operations; in addiction the call billing is equal for devices placed anywhere in the Network Operator State and the server can be anywhere in the World.
Furthermore, in the CSD operation the server shall have a set of modems and multiple phone lines to ensure that the calling units will not find it busy, while a single modem is enough for GPRS operation.
The speed at which the packets can be downloaded is up to 57600 bps (class 10 device working at 4+1), 4 times faster than CSD.

In the following paragraphs more detailed information will be given on how to establish GPRS connection.

1.2 Preliminary GPRS context parameters setting

1.2.1 Context parameter setting

The context parameters are all the set of information to identify the internet entry point interface provided by the ISP. With these parameters the GPRS network identifies the ISP to be used to gain access to the internet and defines the value of the IP address of the GPRS device once connected.

- send command

AT+CGDCONT[=[<cid>[,<PDP_type>[,<APN>[,<PDP_addr>[,<d_comp>[,<h_comp>[,<pd1>[,…[,pdN]]]]]]]]]]<cr>

where:

<cid> - (PDP Context Identifier) numeric parameter which specifies a particular PDP context definition.
1..*max* - where the value of *max* is returned by the Test command
<PDP_type> - (Packet Data Protocol type) a string parameter which specifies the type of packet data protocol
"IP" - Internet Protocol
"PPP" - Point to Point Protocol
<APN> - (Access Point Name) a string parameter that represents logical name used to select GGSN or external packet data network. If the value is null or omitted, then the subscription value will be requested.
<PDP_addr> - a string parameter that identifies the terminal in the address space applicable to the PDP. The allocated address may be read using the **+CGPADDR** command.
<d_comp> - numeric parameter that controls PDP data compression
0 - off (default if value is omitted)
1 - on
<h_comp> - numeric parameter that controls PDP header compression
0 - off (default if value is omitted)
1 - on
<pd1>, …, **<pdN>** - zero to N string parameters whose meanings are specific to the **<PDP_type>**

NOTE: a special form of the Set command, **+CGDCONT=<cid>**, causes the values for context number **<cid>** to become undefined.

NOTE: issuing **AT+CGDCONT<CR>** is the same as issuing the Read command.

NOTE: issuing **AT+CGDCONT=<CR>** returns the **OK** result code.

- wait for response:

Response	Reason	Action
OK	context parameters have been successfully stored	proceed ahead
ERROR	some error occurred	check parameters and retry

For example:

1- Let's assume you want to set-up the GPRS context number 1(cid) with your GPRS connection parameters
APN: ibox.tim.it
IP address: dynamically assigned by the ISP
Packet Data Protocol type: Internet Protocol (IP)
Data compression: OFF
Header compression: OFF

command:
AT+CGDCONT= 1,"IP","ibox.tim.it","0.0.0.0",0,0 <cr>
response
OK

1.2.2 Minimum Quality of the Service Requested

The minimum quality of service requested parameters represent the boundary under which the connection quality is not anymore acceptable and will be terminated.

- send command

AT+CGQMIN=<cid>,<precedence>,<delay>,<reliability>,<peak>,<mean><cr>

where:

<cid> - is the index number of the desired context to be written (up to 5 different context).
<precedence> - is the precedence class**.** It is applied when the network has a heavy duty and user precedence must be followed to ensure operations, the higher the priority the better the service.
0 - subscribed (default)
1 - High priority
2 - Normal priority
3 - Low priority

<delay> - is the delay class. It represents the maximum allowable time delay class between the sending and the reception of a packet.

0 - subscribed (default)
1 - delay class 1
2 - delay class 2
3 - delay class 3
4 - delay class 4 (best effort)

<reliability> - is the connection reliability class. It represents the connection reliability requested, the higher is the number the less reliable is the data exchanged.

0 - subscribed (default)
1 - reliability class 1 (acknowledged GTP,LLC and RLC; protected data)
2 - reliability class 2 (unacknowledged GTP, acknowledged LLC and RLC; protected data)
3 - reliability class 3 (unacknowledged GTP and LLC, acknowledged RLC; protected data)
4 - reliability class 4 (unacknowledged GTP,LLC and RLC; protected data)
5 - reliability class 5 (unacknowledged GTP,LLC and RLC; unprotected data)

<peak> - is the peak data transfer throughput

0 - subscribed (default)
1 - up to 7,8 kbps
2 - up to 15,6 kbps
3 - up to 31,3 kbps
4 - up to 62,5 kbps
5 - up to 125 kbps
6 - up to 250 kbps
7 - up to 500 kbps
8 - up to 1000 kbps
9 - up to 2000 kbps

<mean> - is the mean data transfer throughput

0 - subscribed (default)
1 - up to 0,8 kbps
2 - up to 1,6 kbps
3 - up to 3,9 kbps
4 - up to 7,8 kbps
5 - up to 15,6 kbps
6 - up to 39 kbps
7 - up to 78 kbps
8 - up to 156 kbps
9 - up to 390 kbps
10 - up to 7,6 Mbps
11 - up to 15.2 Mbps
12 - up to 38.2 Mbps
13 - up to 76.3 Mbps
14 - up to 152 Mbps
15 - up to 381 Mbps
16 - up to 762 Mbps
17 - up to 1525 Mbps
18 - up to 3815 Mbps

31 - Best Effort

- wait for response:

Response	Reason	Action
OK	context parameters have been successfully stored	proceed ahead
ERROR	some error occurred	check parameters and retry.

NOTE: If your minimum requirements are too high, then it can happen that it is impossible to establish a GPRS connection, because the network has not enough resources to guarantee that quality of service. If does this happen, then you shall try reducing your minimum quality requirements.

For example:

1- Let's assume you want to set-up the GPRS context number 1(cid) written before with your GPRS min QoS parameters:
Precedence class: Normal priority
Delay class: subscribed
Reliability class: subscribed
Peak throughput: not less than 15,6 kbps
Mean throughput: not less than 7,8 kbps

command:
AT+CGQMIN= 1,2,0,0,5,4 <cr>
response
OK

NOTE: Telit suggests to setup AT+CGQMIN=1,0,0,0,0,0

1.2.3 Requested Quality of the Service

The requested quality of service parameters represents the connection quality that is requested to the network on GPRS context activation.

- send command

AT+CGQREQ=<cid>,<precedence>,<delay>,<reliability>,<peak>,<mean><cr>

where:

<cid> - is the index number of the desired context to be written (up to 5 different context).

<precedence> - is the precedence class
<delay> - is the delay class
<reliability> - is the connection reliability class
<peak> - is the peak data transfer throughput
<mean> - is the mean data transfer throughput

Parameters assume the same values as in the previous section.

- wait for response:

Response	Reason	Action
OK	context parameters have been successfully stored	proceed ahead
ERROR	some error occurred	check parameters and retry

For example:

1- Let's assume you want to set-up the GPRS context number 1(cid) written before with your GPRS requested QoS parameters:
Precedence class: High priority
Delay class: subscribed
Reliability class: subscribed
Peak throughput: subscribed
Mean throughput: best effort

command:
AT+CGQREQ= 1,1,0,0,0,31 <cr>
response
OK

NOTE: Telit suggests to setup AT+CGQMIN=1,0,0,3,0,0

1.3 GPRS context activation and data state entering

This operation corresponds to the dial and connect of a CSD GSM data call issued to an internet service provider.

- send command

ATD*99*<cid>#<cr>**

where:

<cid> - is the index number of the desired context to be used (up to 5 different context)

- wait for response:

Response	Reason	Action
CONNECT	GPRS connection is being processed	proceed ahead with the authentication & Packed data protocol
ERROR	some error occurred	check context parameters and retry. See par. 1.2.1, 1.2.2, 1.2.3 check also Network registration status.
+CME ERROR: <error code>	some error occurred	check context parameters and retry. See par. 1.2.1, 1.2.2, 1.2.3 check also Network registration status.

For example:

1- Let's assume you want to activate and enter the GPRS state with context number 1(cid) written before with your GPRS requested QoS parameters:

command:
ATD*99***1# <cr>
response
CONNECT

At this point, your application should start the PPP protocol with the LCP Exchange phase:

➔ LCP Configure Request

🡐 LCP Configure Acknowledge

  🡒 PAP Authentication
  🡐 PAP-Ack

🡒 NCP (IP) Configure Request
🡐 NCP (IP) Configure Acknowledge

At this point the TCP/IP - PPP protocol stack is up and data packets can be exchanged.

NOTE: Explanation of TCP/IP and PPP protocol stack is beyond the scope of this document. Further information on the LCP protocol and PPP protocol definition can be found in the RFC1661. Further information on the PAP protocol definition can be found in the RFC1334. Further information on the IPCP protocol definition can be found in the RFC1332.

NOTE: The CONNECT result code is raised before complete GPRS connection establishment.

1.4 GPRS data state exit

→ LCP Terminate Request
← LCP Terminate Acknowledge

- Wait for **NO CARRIER** response.

or in alternative:

- send escape sequence:

+++

- wait for 2s (default silence time)
- wait for response:

Response	Reason	Action
OK	Telit module is in command mode now	proceed ahead
ERROR	some error occurred	check command syntax and timing and retry
NO CARRIER	connection has been closed	proceed ahead

- send command

ATH<cr>

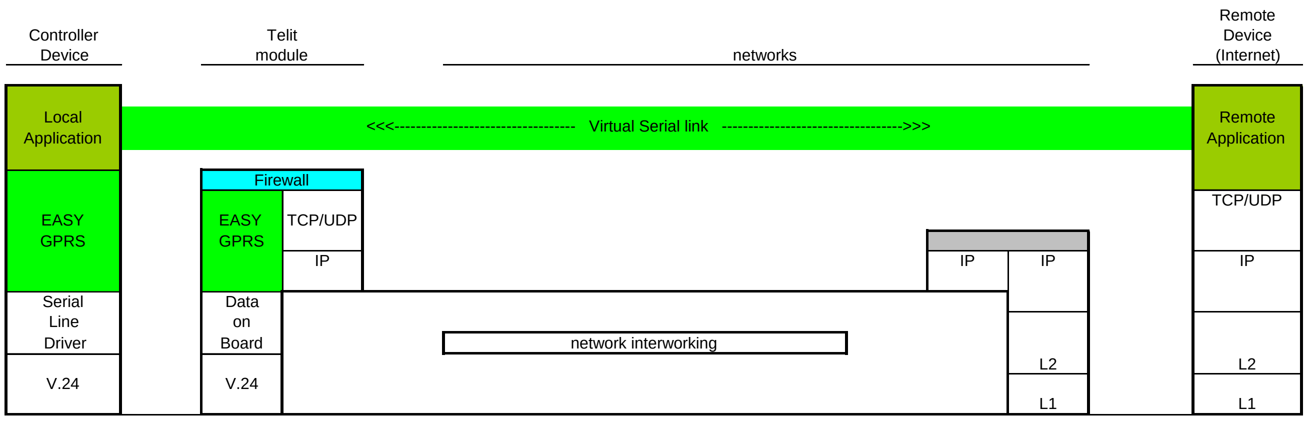
- wait for response:

Response	Reason	Action
OK	GPRS connection has been closed	
ERROR	some error occurred	check command syntax and retry

2 Enhanced Easy GPRS Extension

2.1 Overview

The Easy GPRS feature allows the Telit module users to contact a device in internet and establish with it a raw data flow over the GPRS and Internet networks.
This feature can be seen as a way to obtain a "virtual" serial connection between the Application Software on the Internet machine involved and the controller of the **Telit module**, regardless of all the software stacks underlying.
An example of the protocol stack involved in the devices is reported:



This particular implementation allows to the devices interfacing to the Telit module the use of the GPRS and Internet packet service without the need to have an internal TCP/IP stack since this function is embedded inside the module. The Easy GPRS overcomes some of the known limitations of the previous implementation and implements some new features such as:

- Keep the GPRS context active even after the closing of a socket, allowing the application to keep the same IP address;
- Also Mobile terminated (incoming) connections can be made, now it is possible to receive incoming TCP connection requests;
- A new internal firewall has been implemented in order to guarantee a certain level of security on internet applications.

2.1.1 Easy GPRS Outgoing connection

The Easy GPRS feature provides a way to place outgoing TCP/UDP connections and keep the same IP address after a connection, leaving the GPRS context active.
The steps required to open a socket and close it without closing the GRPS context are:

- configuring the GPRS Access
- configuring the embedded TCP/IP stack behaviour
- defining the Internet Peer to be contacted
- request the GPRS context to be activated
- request the socket connection to be opened
- exchange data
- close the TCP connection while keeping the GPRS active

All these steps are achieved through AT commands. As for common modem interface, two logical statuses are involved: command mode and data traffic mode.

- In Command Mode (CM), some AT commands are provided to configure the Data Module Internet stack and to start up the data traffic.
- In data traffic mode (Socket Mode, SKTM), the client can send/receive a raw data stream which will be encapsulated in the previously configured TCP / IP packets which will be sent to the other side of the network and vice versa. Control plane of ongoing socket connection is deployed internally to the module.

2.1.1.1 Configuring the GPRS access

The GPRS access configuration is done by setting:

- the GPRS context number 1 parameters (see +CGDCONT command)
- the Authentication parameters: User Name and Password (see commands #USERID, #PASSW)

2.1.1.2 Configuring the embedded TCP/IP stack

The TCP/IP stack behaviour must be configured by setting:

- the packet default size (see command #PKTSZ)
- the data sending timeout (see command #DSTO)
- the socket inactivity timeout (see command #SKTTO)

2.1.1.3 Saving settings for the Internet peer to be contacted

In order to maintain all settings for TCP/IP stack refer to the AT command #SKTSAV that saves the actual socket parameters in the NVM of the device.

2.1.1.4 Request the GPRS context to be activated

With the command #GPRS you can activate or deactivate a GPRS context INDEPENDENTLY from the TCP socket opening:

AT#GPRS=1 activates the context,
AT#GPRS=0 deactivates the context

Therefore with the AT#GPRS=1 command the module:

- Telit module activates the context previously defined with AT+CGDCONT
- Telit module proceeds to the authentication with the parameters specified

Note that activating a context implies getting an IP address from the network and this will be maintained throughout the session.
The response code to the AT#GPRS=1 command reports the IP address obtained from the network, allowing the user to report it to his server or application.
Deactivating the context implies freeing the network resources previously allocated to the device.

2.1.1.5 Open the connection with the internet host

With the AT command #SKTD (socket Dial) the TCP/UDP request to connect with the internet host starts:

- DNS query is done to resolve the IP address of the host name internet peer
- Telit module establishes a TCP/UDP (depending on the parameter request) connection with the given internet host
- Once the connection is up the module reports the code: CONNECT

NOTE: all peer specifications of this socket Dial are within the command and not the one stored with #SKTSET command.
From this moment the data incoming in the serial port is packet and sent to the Internet host, while the data received from the host is serialised and flushed to the Terminal Equipment.

NOTE: when disconnecting the #SKTD command does not close the GPRS context, leaving it active for next connections until an AT#GPRS=0 command is issued or the network requests a context closing.

2.1.1.6 Close the Socket without deactivating the context

The connection can be closed for the following reasons:

- remote host TCP connection close
- socket inactivity timeout
- Terminal Equipment by issuing the escape sequence "+++"
- Network deactivation

NOTE: if there is an escape sequence in the raw data to be sent, then the TE must work it out and sent it in a different fashion to guarantee that the connection is not closed.
The pause time is defined in the parameter S12.
On the reception of an escape sequence if the socket was opened with the AT#SKTD command, the Telit module closes the connection, does not deactivate the GPRS context and returns to command mode issuing the NO CARRIER code.

2.1.2 Easy GPRS Incoming Connection

The Easy GPRS feature provides a way to accept incoming TCP/UDP connections and keep the same IP address after a connection, leaving the GPRS context active.
The steps that will be required to open a socket in listen, waiting for connection requests from remote hosts and accept these request connections only from a selected set of hosts, then close it without closing the GRPS context are:

- configuring the GPRS Access
- configuring the embedded TCP/IP stack behaviour (see par. 2.1.1.2)
- defining the Internet Peer that can contact this device (firewall settings) (see par. 2.1.2.1)
- request the GPRS context to be activated (see par. 2.1.1.4)
- request the socket connection to be opened in listen (see par. 2.1.2.2)
- receive connection requests (see par. 2.1.2.3)
- exchange data
- close the TCP connection while keeping the GPRS active (see par. 2.1.1.6)

All these steps are achieved through AT commands.
As for common modem interface, two logical statuses are involved: command mode and data traffic mode.

- In Command Mode (CM), some AT commands are provided to configure the Data Module Internet stack and to start up the data traffic.
- In data traffic mode (Socket Mode, SKTM), the client can send/receive a raw data stream which will be encapsulated in the previously configured TCP / IP packets which will be sent to the other side of the network and vice versa. Control plane of ongoing socket connection is deployed internally to the module.

2.1.2.1 Defining the Internet Peer that can contact this device (firewall settings)

The Telit module has an internal Firewall that controls the behaviour of the incoming connections to the module.
The firewall applies for INCOMING (listening) connections, OUTGOING connections will be always done regardless of the firewall settings.
Firewall General policy is DROP, therefore all packets that are not included into an ACCEPT chain rule will be silently discarded.

When packet incomes from the IP address <incoming IP>, the firewall chain rules will be scanned for matching with the following criteria:

<incoming IP> & <net mask> = <ip_address> ?

if the result is yes, then the packet is accepted and the rule scan is finished, otherwise the next chain is taken into account until the end of the rules when the packet is silently dropped if no matching was found.

For example, let's assume we want to accept connections only from our devices which are on the IP addresses ranging from 197.158.1.1 to 197.158.255.255

We need to add the following chain to the firewall:
AT#FRWL=1,"197.158.1.1","255.255.0.0"

2.1.2.2 Request the socket connection to be opened in listen

With the AT command #SKTL (socket Listen) the TCP request to start listening for connection requests is executed. The Telit module opens a listening socket on the port specified, waiting for incoming TCP connections (depending on the parameter request) with the internet hosts

The parameters that shall be specified are the local port where packets shall be received, the type of socket and the closing behaviour.

2.1.2.3 Receiving connection requests

Once the connection request is received, the module reports an indication of connection with an unsolicited code

+CONN FROM: <remote address>

- then connection is accepted and once it is up the module reports the code:
  CONNECT

From this moment the data incoming in the serial port is packet and sent to the Internet host, while the data received from the host is serialised and flushed to the Terminal Equipment.
Note that the connections request are FIRST screened in the firewall, then if they are accepted they pass to the listening socket; therefore only hosts that are in the ACCEPT chain rules of the firewall can induce a connection request, the other host requests will be silently discarded without any indication to the remote host (for security reasons).
Once the connection is received and closed, the socket is not anymore in listen. If the application needs again to be in listen, then it shall send again the socket listen #SKTL command.

NOTE: before issuing this command the GPRS context should be activated with AT#GPRS=1. When disconnecting the #SKTL command does not close the GPRS context, and leaves it active for the

future connections until an AT#GPRS=0 command is issued or the network requests a context closing.

2.1.3 FTP Client

On top of the embedded TCP/IP stack, an FTP client is available. Such FTP is a versatile protocol suite, designed to be powerful, compact and simple to use for file transfer over the TCP/IP and therefore over the GPRS network.
As far as the AT commands list is concerned, the customer shall refer to the AT Commands Reference Guide.

2.1.4 Email Client

SMTP and POP Application Layer protocol are available on the top of the TCP/IP protocol. This permits sending and receiving Emails on your embedded device.
To get more information on system settings and AT commands available for email client please consult the examples in the section 2.2 and AT Commands Reference Guide.

2.1.5 Known limitations

The implementation of the EASY GPRS feature has the following known limitations:

- Only one socket can be opened at a time, no multiple socket connections can be made;
- Only one connection request can be accepted at a time, subsequent requests will be silently discarded.
- Only the first GPRS context is associated with this feature;

It is taken for granted that external processor will be able to handle at least a limited v.24 implementation: RTS, CTS and, highly recommended, DCD lines; this because software flow control is not applicable to the feature.
Due to the particularity of this feature, the flow control of both the directions uplink and downlink is interlocked.

2.2 FTP OPERATIONS

A set of AT commands is available to support the FTP activities. The fist command is called #FTPTO (FTP Time-Out) which defines the time-out for FTP operations. The module has already a factory default time defined that is 10 s.

If it is needed to be modified, the syntax is:

AT#FTPTO[=<tout>]

Parameter:
<tout> - time-out in 100 ms units

100..5000 - hundreds of ms (factory default is 100)

NOTE: The parameter is not saved in NVM.
NOTE: if parameter **<tout>** is omitted the behaviour of Set command is the same as Read command.

Example:

```
AT#FTPTO=1000<cr>      (set the timeout to 100sec)
OK
```

2.2.1 Opening and Closing an FTP Connection

With the command **AT#FTPOPEN=<server:port>,<username>,<password>,<mode>** is possible to open the FTP connection.

The parameters are:
<server:port> - string type, address and port of FTP server (factory default port 21).
<username> - string type, authentication user identification string for FTP.
<password> - string type, authentication password for FTP.
<mode>
0 - active mode (default)
1 - passive mode

In order to close the FTP connection the AT command **AT#FTPCLOSE** should be used.

2.2.2 Setting the FTP Transfer Type

With the command **AT#FTPTYPE[=<type>]** is possible to configure the file transfer type. The command must be provided during an FTP connection.

Parameter:
<type> - file transfer type:
 0 - binary
 1 - ASCII

NOTE: The command causes an **ERROR** result code to be returned if no FTP connection has been opened yet.

NOTE: If the parameter is omitted then the behaviour of Set command is the same of Read command.

2.2.3 FTP File transfer to the server

With the command **AT#FTPPUT=<filename> ,** to issued during an FTP connection, is possible to open a data connection and starts sending **<filename>** file to the FTP server.

If the data connection succeeds, a **CONNECT** indication is sent, otherwise a **NO CARRIER** indication is sent.

Parameter:
<filename> - string type, name under which you choose to save the file on the server (must have the right extension: es. if the file you're sending is .txt then the **<filename>** can be test.txt)

NOTE: use the escape sequence **+++** to close the data connection.

NOTE: The command causes an **ERROR** result code to be returned if no FTP connection has been opened yet.

Example of an FTP file transfer to the server:

Define PDP contest:
```
AT+CGDCONT=1,"IP", "internet.wind.biz"<cr>
OK
```

GPRS Context Activation, as response gives IP of the module:
```
AT#GPRS=1<cr>
+IP: 193.199.234.255
OK
```

```
Opening of FTP connection:
AT#FTPTO=1000<cr>                (FTP settings of time-out)
OK

AT#FTPOPEN=”199.188.25.77”,”user”,”pass”,0<cr>
OK
In this case port of FTP server is not specified, which means that it has the default value: 21

AT#FTPTYPE=0<cr>                 (FTP settings of file type)
OK

FTP file transfer to the server in the file named “file.txt”:
AT#FTPPUT="file.txt"<cr>
CONNECT

(send the file)

+++                 (escape sequence +++ to close the data connection)
NOCARRIER

AT#FTPCLOSE<cr>        (closing FTP connection)
OK

Deactivation of GPRS context if required:
AT#GPRS=0<cr>
OK
```

2.2.4 FTP File download from the server

With the command **AT#FTPGET=<filename> ,** to issued during an FTP connection, opens a data connection and starts getting a file **<filename>** from the FTP server.

If the data connection succeeds, a **CONNECT** indication is sent, otherwise a **NO CARRIER** indication is sent. The file is received on the serial port.

Parameter:
<filename> - file name, string type.

NOTE: The command causes an **ERROR** result code to be returned if no FTP connection has been opened yet.

Example of an FTP file download from the server:

Define PDP contest:

```
AT+CGDCONT=1,”IP”, "internet.wind.biz"<cr>
OK
```

GPRS Context Activation, as response gives IP of the module:

```
AT#GPRS=1<cr>
+IP: 193.199.234.255
OK
```

Opening of FTP connection:

```
AT#FTPTO=1000<cr>                (FTP settings of time-out)
OK

AT#FTPOPEN=”199.188.25.77”,”user”,”pass”,0<cr>
OK
```

In this case port of FTP server is not specified, which means that it has the default value: 21

```
AT#FTPTYPE=0<cr>                 (FTP settings of file type)
OK

AT#FTPCWD="incoming"             (change working directory if requiered)
OK
```

In order to get the list of files on the working directory from the server AT command AT#FTPLIST should be used.

Downloading FTP file “file.txt” from the server:

```
AT#FTPGET="file.txt"<cr>
CONNECT
```

(receive the file)

Data connection will be closed automatically when the file sending is terminated:

```
NO CARRIER

AT#FTPCLOSE<cr>        (closing FTP connection)
OK
```

Deactivation of GPRS context if required:

```
AT#GPRS=0<cr>
OK
```

TIP: The #GPRS command activates the context and it is necessary to start the FTP connection.

To get more information for other available commands on the FTP functionality please refer to the AT Commands Reference Guide.

2.3 Examples[1]

2.3.1 Easy GPRS - HTTP client application

Let’s suppose we want to connect our embedded device to an HTTP server and retrieve an HTML page using the EASY GPRS feature.

Initial data:

Server to be contacted	www.telit.com
Application Layer Protocol	HTTP1.0 (RFC1945); HTTP1.1 (RFC2068)
Page to be retrieved	homepage of server
GPRS settings	
APN	internet.gprs
IP of GPRS device	dynamically assigned by the network
DNS	assigned by the network
USERID	EASY GPRS
PASSWORD	EASY GPRS

Checking on the RFC990 the HTTP service we can found that the port 80 is dedicated for HTTP service, therefore our HTTP server will be waiting for incoming connections on that port and we will fix the EASY GPRS port to be contacted on the remote server exactly to 80.
Second thing we have to discover is whether the transport protocol has to be TCP or UDP; on the RFC1945 we can read that the HTTP Application layer protocol is meant to be on top of TCP/IP protocol, therefore the transport protocol choice will fall on TCP.
Now we have all the information needed to configure our system.

With our microcontroller we issue to the Telit module the following AT commands:

```
AT+CGDCONT = 1,"IP","internet.gprs","0.0.0.0",0,0<cr>         (1-GPRS context setting)
AT#USERID = "EASY GPRS"<cr>                                   (2-Authentication setting)
AT#PASSW = "EASY GPRS"<cr>                                    (2-Authentication setting)
AT#PKTSZ=300                                                  (3-Socket setting: sets the
default packet size to be used by the TCP/UDP/IP stack for data sending.)
AT#DSTO=50                                                    (3-Socket setting)
```

[1] **NOTE:** For the detailed information about AT commands reported in examples please consult AT Commands Reference Guide

AT#SKTTO=90 (3-Socket setting: sets the maximum time with no data exchanging on the socket that the module awaits before closing the socket and deactivating the GPRS context)
AT#SKTCT=600 (3-Socket setting)

For our convenience we will store all these parameters with the command:

```
AT#SKTSAV<cr>
```

Next step is activation of the GPRS context:

```
AT#GPRS=1<cr>
+IP: 193.199.234.255
OK
```

This command replies with the IP address assigned by the network.

Now we can proceed with contacting the server with AT command for socket dial:

```
AT#SKTD=0,80,"www.telit.com",0,0
```

When we receive the CONNECT indication, then we are exchanging data with the HTTP server program on the remote host machine.

Now following the HTTP protocol we ask for the homepage by sending the following lines on the serial line:

```
GET / HTTP/1.1<cr><lf>
Host: www.telit.com<cr><lf>
Connection: keep-alive<cr><lf>
<cr><lf>
```

TIP: Remember that the strings, which are sent to the HTTP server, have to be ended by line feed character. To see the issued commands enable the local echo.

As a response to our query the HTTP server will reply with the HTML code of the homepage and some debugging responses that we will see directly on the serial line:

```
HTTP/1.1 200 OK
Date: Thu, 06 2003 10:21:58 GMT
Server: Apache/1.3.27 (Unix)
Last-Modified: Thu, 06 2003 10:21:58 GMT
Content-Type: text/html
Connection: close

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 FINAL//EN">
<HTML>
… here is all the HTML code of the page..
</HTML>

<pause>+++<pause>
```

NO CARRIER

The Telit module is now back to command mode.

2.3.2 Easy GPRS - EMAIL sending application

Let's suppose we want to send with our embedded device an EMAIL by using a SMTP server.

Initial data:

Server to be contacted	smtp.domain.com
SMTP service	port #25
Application Layer Protocol	SMTP (RFC821)
Sender	"module"<module@domain.com>
Email account username	module@domain.com
Email account password	telit
Receiver	"Receiver"<receiver@server.net>
Subject	Email Test
Message body	This message is sent in order to test Easy GPRS feature. Hello World!
GPRS settings	
APN	internet.gprs
IP of GPRS device	dynamically assigned by the network
DNS	assigned by the network
USERID	EASY GPRS
PASSWORD	EASY GPRS

Checking on the RFC990 the SMTP service we can found that the port 25 is dedicated for SMTP service, therefore our SMTP server will be waiting for incoming connections on that port and we will fix the EASY GPRS port to be contacted on the remote server exactly to 25.
Second thing we have to discover is whether the transport protocol has to be TCP or UDP; on the RFC821 we can read that the SMTP Application layer protocol is meant to be on top of TCP/IP protocol, therefore the transport protocol choice will fall on TCP.
Now we have all the information needed to configure our system.

With our microcontroller we issue to the Telit module the following AT commands:

```
AT+CGDCONT = 1,"IP","internet.gprs","0.0.0.0",0,0<cr>      (1-GPRS context setting)
AT#USERID = "EASY GPRS"<cr>                                 (2-Authentication setting)
AT#PASSW = "EASY GPRS"<cr>                                  (2-Authentication setting)
AT#PKTSZ=300                                                (3-Socket setting)
AT#DSTO=50                                                  (3-Socket setting)
AT#SKTTO=90                                                 (3-Socket setting)
AT#SKTCT=600                                                (3-Socket setting)
```

For our convenience we will store all these parameters with the command:

```
AT#SKTSAV<cr>
```

Next step is activation of the GPRS context:

```
AT#GPRS=1<cr>
+IP: 193.199.234.255
OK
```

The command gives as response the IP address assigned by the network.

Now we can proceed with contacting the server with AT command for socket dial:

```
AT#SKTD= 0,25,"smtp.domain.com",0,0<cr>
```

When we receive the CONNECT indication, then we are exchanging data with the SMTP server program on the remote host machine.

Following the SMTP protocol we proceed with the HELLO presentation and mail delivery directly over the serial line (in blu you can find the data sent by us, in violet the one received from host):

```
          220 smtp.domain.com ESMTP Service (7.0.027-DD01) ready

HELO pcprova<cr><lf>

          250 smtp.domain.com


AUTH LOGIN<cr><lf>                                   (authentication method)

          334 VXRlcm8gkXU6

Z204NjJAZG9tYWluLmNvbQ==<cr><lf>             (module@domain.com base64 encoding)

          334 UHFzc6dcvmQ6

dGVsaXQ= <cr><lf>                                    (telit base64 encoding)

          235 2.0.0 OK Authenticated

MAIL FROM: module@domain.com <cr><lf>                (Sender)

          250 2.1.0 module@domain.com... Sender ok

RCPT TO: receiver@server.net <cr><lf>                (Receiver)

          250 2.1.5 receiver@server.net... Recipient ok
```

```
DATA<cr><lf>

        354 Enter mail, end with "." on a line by itself

Return-Receipt-To: < module@domain.com ><cr><lf>
Reply-To: < module@domain.com ><cr><lf>
From: < module@domain.com ><cr><lf>
To: < receiver@server.net ><cr><lf>
Subject: Email test<cr><lf>
Date: Fri, 19 Sep 2003 11:41:32 +0200<cr><lf>
MIME-Version: 1.0<cr><lf>
X-Priority: 3 (Normal) <cr><lf>
X-MSMail-Priority: Normal<cr><lf>
X-Mailer: GM862 TELIT SW, Build 1.0.1000 (1.0.1111.0) <cr><lf>
Importance: Normal<cr><lf>
X-MimeOLE: Produced By GM862 TEST SW<cr><lf>
<cr><lf>
Content-Type: text/plain; <cr><lf>
        charset="iso-8859-1"<cr><lf>
Content-Transfer-Encoding: 7bit<cr><lf>
<cr><lf>
This message is sent in order to test Easy GPRS feature.  Hello World!<cr><lf>
<cr><lf>
. <cr><lf>

        250 2.0.0 h8J9QNH3008461 Message accepted for delivery

QUIT<cr><lf>

        221 2.0.0 smtp.domain.com closing connection

+++
NO CARRIER
```

The Telit module is now back in the command mode.

2.3.3 Easy GPRS -EMAIL receiving application

Let's suppose we want to receive with our embedded device an EMAIL by using a POP3 server.

Initial data:

Server to be contacted	POP.mail.server
POP service	port #110
Application Layer Protocol	POP3 (RFC1785)
Receiver	"module"<module@domain.com>

Email account username	module@domain.com
Email account password	telit
GPRS settings	
APN	internet.gprs
IP of GPRS device	dynamically assigned by the network
DNS	assigned by the network
USERID	EASY GPRS
PASSWORD	EASY GPRS

Checking on the RFC1785, we can found that the port 110 is dedicated for POP3 service, therefore our POP server will be waiting for incoming connections on that port and we will fix the EASY GPRS port to be contacted on the remote server exactly to 110.
Second thing we have to discover is whether the transport protocol has to be TCP or UDP; on the RFC1785 we can read that the POP3 Application layer protocol is meant to be on top of TCP/IP protocol, therefore the transport protocol choice will fall on TCP.
Now we have all the information needed to configure our system.
With our microcontroller we can now issue to the Telit module the following AT commands:

```
AT+CGDCONT = 1,"IP","internet.gprs","0.0.0.0",0,0<cr>          (1-GPRS context setting)
AT#USERID = "EASY GPRS"<cr>                                     (2-Authentication setting)
AT#PASSWD = "EASY GPRS"<cr>                                     (2-Authentication setting)
AT#PKTSZ=300                                                    (3-Socket setting)
AT#DSTO=50                                                      (3-Socket setting)
AT#SKTTO=90                                                     (3-Socket setting)
AT#SKTCT=600                                                    (3-Socket setting)
```

For our convenience we store all these parameters with the command:

```
AT#SKTSAV<cr>
```

Now we can activate the GPRS context:

```
AT#GPRS=1<cr>
+IP: 193.199.234.255
OK
```

The commands gives as response the IP address assigned to the module by the network.

```
AT#SKTD=0,110,"POP.mail.server",0,0<cr>
```

When we receive the CONNECT indication, then we are exchanging data with the POP3 server program on the remote host machine.
Following the POP3 protocol we can proceed with the authentication directly over the serial line (in blue you can find the data sent by us, in violet the one received from host):

```
   +OK POP3 PROXY server ready (7.0.027) <A6B4DDEA93433C73A01@pop4.libero.it>

USER module@domain.com<cr><lf>
```

```
    +OK Password required

PASS telit<cr><lf>
    +OK 1 messages

LIST\r\n
    +OK
    1 19550
    .

RETR 1<cr><lf>
    +OK 19550 bytes
    Return-Path: <module@domain.com>
    Received: from smtp5.libero.it (193.70.192.55) by ims2d.libero.it (7.0.028)
    id 40DFC49A010E5708 for test@libero.it; Tue, 17 Aug 2004 12:24:02+0200
    Received: from smtp.telital.com (194.185.15.65) by smtp5.libero.it (7.0.027-DD01)
    .

QUIT<cr><lf>
    +OK POP3 server closing connection
+++
NO CARRIER
```

2.3.4 Remote connection between two modules

Configuration for the module that receives data (server):

Define PDP Context	AT+CGDCONT=1,"IP","ibox.tim.it","0.0.0.0"
GPRS Context Activation	AT#GPRS=1
Firewall Setup(if requiered)	AT#FRWL=1,"198.158.1.1","0.0.0.0"
Socket Listen	AT#SKTL=1,0,1024,255

First you have to define PDP context filling in the information of APN in this example: ibox.tim.it.
Next step is activation of GPRS context which gives as reply the IP of the module assigned by network:

```
AT#GPRS=1
+IP: 217.201.142.223
OK
```

Before opening socket in listen it is possible to define an accept firewall chain in order to filter IP of the senders.
At the end with AT commad AT#SKTL=1,0,1024,255 the socket will be set in listen on the port #1024.

Configuration for the module that opens connection (client):

Define PDP Context	AT+CGDCONT=1,”IP”,”ibox.tim.it”,”0.0.0.0”
GPRS Context Activation	AT#GPRS=1
Socket Dial	AT#SKTD=0,1024,”217.201.142.223”

First you have to define PDP context filling in the information of APN in this example: ibox.tim.it. Next step is activation of GPRS context which gives as reply the IP of the module assigned by network. Now you can open the connection with the remote host with IP address 217.201.142.223 on the port 1024 (as in example).

NOTE: IP of the modules can be verified with the following AT command line: AT#GPRS=1

3 List of acronyms

Abbreviation	Description
Ack	Acknowledge
APN	Access Point Name
AT	Attention commands
CM	Command mode
CR	Carriage Return
CSD	Circuit Switched Data
CTS	Clear To Send
DCD	Data Carrier Detected
FTP	File Transfer Protocol
GGSN	Gateway GPRS Serving/Support Node
GPRS	General Radio Packet Service
GSM	Global System for Mobile communication
GTP	GPRS Tunneling Protocol
HTML	Hyper Text Markup Language
HTTP	Hypertext Transfer Protocol
HSCSD	High-Speed Circuit-Switched Data
IP	Internet Protocol
ISDN	Integrated Services Digital Network
ISP	Internet Service Provider
LCP	Link Control Protocol
LLC	Logical Link Control
MS	Mobile Station
MT	Mobile Terminated
NCP	Network Control Protocol
OEM	Other Equipment Manufacturer
PAP	Password Authentication Protocol
PDP	Packet Data Protocol
PDU	Protocol Data Unit
PLMN	Public Land Mobile Network
PPP	Point to Point Protocol
QoS	Quality Of Service
RLC	Radio Link Control
RoHS	Reduction of Hazardous Substances
RTS	Ready To Send
SIM	Subscriber Identity Module
SKTM	Socket Mode
SMTP	Simple Mail Transfer Protocol
TCP	Trasmission Control Protocol

4 Document Change Log

Revision	Date	Changes
ISSUE #0	02/01/2007	Initial release