GE910 Family Ports Arrangements
User Guide

1vv0301049 Rev.3 – 2015-02-22

Making machines talk.

APPLICABILITY TABLE

	SW Versions
GE Family (Embedded)	
GE910-QUAD	**13.00.xx4**
GE910-GNSS	**13.00.xx4**
GE910-QUAD AUTO	**13.00.xx6**

Note: the products equipped with the software versions equal or higher than the versions shown in the table support the features described in the present document. Refer also to “Document History” chapter.

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

LEGAL NOTICE

These Specifications are general guidelines pertaining to product selection and application and may not be appropriate for your particular project. Telit (which hereinafter shall include, its agents, licensors and affiliated companies) makes no representation as to the particular products identified in this document and makes no endorsement of any product. Telit disclaims any warranties, expressed or implied, relating to these specifications, including without limitation, warranties or merchantability, fitness for a particular purpose or satisfactory quality. Without limitation, Telit reserves the right to make changes to any products described herein and to remove any product, without notice.
It is possible that this document may contain references to, or information about Telit products, services and programs, that are not available in your region. Such references or information must not be construed to mean that Telit intends to make available such products, services and programs in your area.

USE AND INTELLECTUAL PROPERTY RIGHTS

These Specifications (and the products and services contained herein) are proprietary to Telit and its licensors and constitute the intellectual property of Telit (and its licensors). All title and intellectual property rights in and to the Specifications (and the products and services contained herein) is owned exclusively by Telit and its licensors. Other than as expressly set forth herein, no license or other rights in or to the Specifications and intellectual property rights related thereto are granted to you. Nothing in these Specifications shall, or shall be deemed to, convey license or any other right under Telit's patents, copyright, mask work or other intellectual property rights or the rights of others.

You may not, without the express written permission of Telit: (i) copy, reproduce, create derivative works of, reverse engineer, disassemble, decompile, distribute, merge or modify in any manner these Specifications or the products and components described herein; (ii) separate any component part of the products described herein, or separately use any component part thereof on any equipment, machinery, hardware or system; (iii) remove or destroy any proprietary marking or legends placed upon or contained within the products or their components or these Specifications; (iv) develop methods to enable unauthorized parties to use the products or their components; and (v) attempt to reconstruct or discover any source code, underlying ideas, algorithms, file formats or programming or interoperability interfaces of the products or their components by any means whatsoever. No part of these Specifications or any products or components described herein may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, without the prior express written permission of Telit.

HIGH RISK MATERIALS

Components, units, or third-party products contained or used with the products described herein are NOT fault-tolerant and are NOT designed, manufactured, or intended for use as on-line control equipment in the following hazardous environments requiring fail-safe controls: the operation of Nuclear Facilities, Aircraft Navigation or Aircraft Communication Systems, Air Traffic Control, Life Support, or Weapons Systems (“High Risk Activities"). Telit, its licensors and its supplier(s) specifically disclaim any expressed or implied warranty of fitness for such High Risk Activities.

TRADEMARKS

You may not and may not allow others to use Telit or its third party licensors’ trademarks. To the extent that any portion of the products, components and any accompanying documents contain proprietary and confidential notices or legends, you will not remove such notices or legends.

THIRD PARTY RIGHTS

The software may include Third Party Right software. In this case you agree to comply with all terms and conditions imposed on you in respect of such separate software. In addition to Third Party Terms, the disclaimer of warranty and limitation of liability provisions in this License shall apply to the Third Party Right software.

TELIT HEREBY DISCLAIMS ANY AND ALL WARRANTIES EXPRESS OR IMPLIED FROM ANY THIRD PARTIES REGARDING ANY SEPARATE FILES, ANY THIRD PARTY MATERIALS INCLUDED IN THE SOFTWARE, ANY THIRD PARTY MATERIALS FROM WHICH THE SOFTWARE IS DERIVED (COLLECTIVELY “OTHER CODE”), AND THE USE OF ANY OR ALL THE OTHER CODE IN CONNECTION WITH THE SOFTWARE, INCLUDING (WITHOUT LIMITATION) ANY WARRANTIES OF SATISFACTORY QUALITY OR FITNESS FOR A PARTICULAR PURPOSE.

NO THIRD PARTY LICENSORS OF OTHER CODE SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND WHETHER MADE UNDER CONTRACT, TORT OR OTHER LEGAL THEORY, ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE OTHER CODE OR THE EXERCISE OF ANY RIGHTS GRANTED UNDER EITHER OR BOTH THIS LICENSE AND THE LEGAL TERMS APPLICABLE TO ANY SEPARATE FILES, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Contents

Figures

Tables

1. Introduction

The present document provides a guideline to connect logically the physical interfaces of the module to the services supported by the module itself. It is up to the user to set the internal logical connections in suitable way to avoid hardware/software resources conflicts.

1.1. Scope

This guide describes the ports/services arrangements provided by the GE910 Family modules. The generic ports/services arrangement is the logical connection of an available serial port to an available 'Access Point' (e.g. AT0, AT1, AT2, TT, PYSER, GPS, etc.).

1.2. Audience

The User Application designers use this document to exploit at best the communication resources offered by the GE910 Family modules, without run up against resources contentions among services.

1.3. Contact Information, Support

For general contact, technical support services, technical questions and report documentation errors contact Telit Technical Support at:

TS-EMEA@telit.com

TS-AMERICAS@telit.com

TS-APAC@telit.com

Alternatively, use:

http://www.telit.com/support

For detailed information about where you can buy the Telit modules or for recommendations on accessories and components visit:

http://www.telit.com

Our aim is to make this guide as helpful as possible. Keep us informed of your comments and suggestions for improvements.

Telit appreciates feedback from the users of our information.

1.4. Related Documents

[1] Telit's CMUX Implementation User Guide, 1vv0300994
[2] AT Commands Reference Guide, 80000ST10025a
[3] Easy Script in Python 2.7, 80378ST10106A
[4] GE910 Hardware User Guide, 1vv0300962

1.5. Document History

Revision	Date	Product/SW Version	Changes
0	2013-05-16		First issue
1	2014-02-18	/	The title of the document has been changed from "GE910 Family Ports Arrangements (Virtual Service Device)" in "GE910 Family Ports Arrangements". New chapters have been added.
		Products added: GE910-QUAD AUTO / 13.00.xx6	/
2	2014-05-06		The title of the chapter 5.1 has been changed from "GPS" to "GNSS". The description of the GNSS Receiver Modes has been added, and the chapters 5.1.1 and 5.1.2 have been changed accordingly.
3	2015-02-22	/	Added AT#PORTCFG=4 and AT#PORTCFG=5 commands. Rearranged some chapters. Removed chapter: 2.1.

1.6. Abbreviation and Acronyms

DTE	Data Terminal Equipment
GNSS	Global Navigation Satellite System
RTD	Real Time Debugger
USIFx	Universal Serial Interface
VSD	Virtual Service Device

2. Ports Arrangements and Virtual Serial Device

Before describing the AT#PORTCFG command used to manage the logical connections between physical ports and services provided by the module, it is useful to introduce the Virtual Serial Device. To have information on physical ports of GE910 modules refer to document [4].

Virtual Serial Device, hereafter called VSD, is a software layer designed to run on GE910 modules. It manages logical connections between the physical serial ports, accessible to the user, and the services supported by the module. VSD supports several Software Access Points used as anchorage points for the logical connections. Tab. 1 shows the physical and logical objects involved in the logical connections management: Physical Serial Ports, Software Access Points, AT Parsers and TT Utilities, Services, and Protocols.

Physical Serial Ports	Software Access Points	AT Parsers and TT Utilities	Services	Protocols
USIF0 USIF1 USB (USB0÷USBx)[2]	AT0 AT1 AT2 TT VHWDTE0 VHWDTE1 PYSER …… GPS/NMEA	Instance #1 Instance #2 Instance #3 RTD	Python GNSS	CMUX (VC1÷VC4)[1]

Tab. 1: Physical and Logical Objects managed by VSD

NOTICE: in document [4] USIF0 and USIF1 ports are called respectively Modem Serial Port 1 (Main) and Modem Serial Port 2 (Auxiliary).

GE910 modules provide three ‘AT Commands Parser Instances’ which are logically independent and connected to three different ‘Access Points’. Each parser recognizes and executes the AT commands received on its ‘Access Point’.

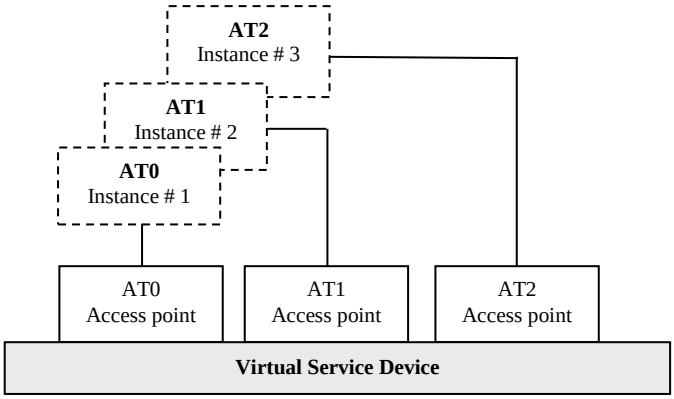

Fig. 1: AT Parser Instances

[1] Four CMUX channels: VC1÷VC4.
[2] USB channels: the number of channels depends on the software version installed on the module.

3. AT#PORTCFG Command

In the following examples, the device connected to the module is a Windows-PC (DTE).

Tab. 2 shows the #PORTCFG parameters, called 'Variant', and USB channels supported by the different types of GE910 modules in accordance with the software version installed on them. To have information on AT command syntax refer to document [2].

Module	Software Versions							
	13.00.xx4		13.00.xx5		13.00.xx6		13.00.xx7	
	variants	USB chan.	variants	USB chan.	variants	USB chan.	variants	USB chan.
GE910-QUAD	0	0,1	0	0,1	0,1,3	0,1,2	0,1,3,4,5	0,1,2
GE910-QUAD AUTO	0	0,1	0	0,1	0,1,3	0,1,2	0,1,3,4,5	0,1,2
GE910-GNSS	/	/	/	/	0,1,3,8,9	0,1,2	0,1,3,4,5,8,9	0,1,2

Tab. 2: #PORTCFG Variants and USB Channels

Here is the sequence to make active the entered AT#PORTCFG command:

- Assume to start from the configuration shown in Fig. 3, it is the factory setting: #PORTCFG is 0;
- Enter, for example, the AT#PORTCFG=1 command through USIF0 port, AT0 parser elaborates the just entered command, but no actions are taken;
- Power down the module;
- Power on the module. The AT#PORTCFG=1 command is executed and the ports arrangement of Tab. 5 is set.

NOTICE: the user shall use only the #PORTCFG variants shown in Tab. 2. All other parameter values returned by the AT#PORTCFG=? command are reserved for Telit internal use only.

The figure below shows an example of USBx channels←→ virtual COMx ports mapping on Windows-PC, it depends on Windows-PC configuration and the number of the USB channels. Telit provides the GE910 USB drivers to be installed on the Windows-PC.

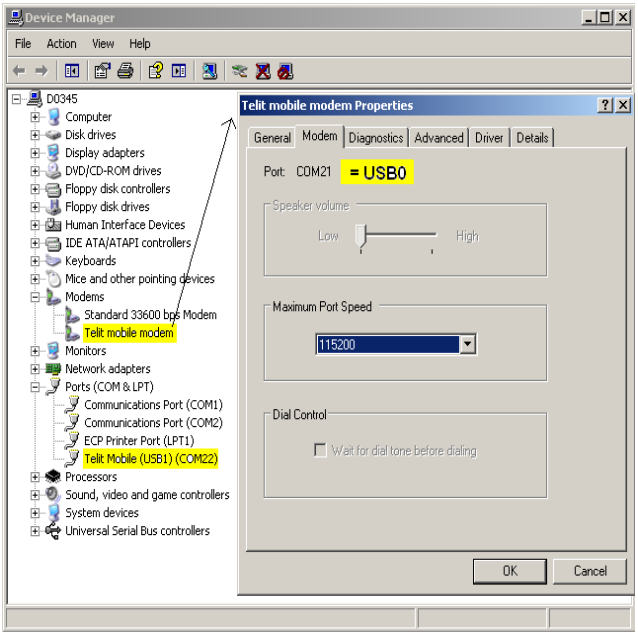

Fig. 2: USBx Mapped into Virtual COMx Ports

In this example, the GE910 module supports two USB channels, see the figure on the left side. The mapping is:

- USB0 channel → COM21
- USB1 channel → COM22

3.1. AT#PORTCFG=0

Assume that the factory PORTCFG setting is '0', and the USB cable is not plugged in. The table below summarizes the logical connections. The GPS/NMEA column is valid only for modules equipped with GNSS receiver.

AT#PORTCFG=0 (Factory setting)					
	AT0	AT1	AT2	TT	GPS/NMEA
No USB cable					
USIF0	X				Ref. chapter 5.1.1
USIF1				RTD	

Tab. 3: #PORTCFG=0, no USB Cable

Assume that the module has the factoring setting configuration shown on the left side. Now, connect the USB cable, the module recognizes the "plug in" event and assumes the factory arrangement summarized in the tables below, in accordance with the software version installed on the module.

AT#PORTCFG=0 (Factory setting)					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			Ref. chapter 5.1.1
USB1			X		Ref. chapter 5.1.1
USIF0	X				Ref. chapter 5.1.1
USIF1				RTD	

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=0 (Factory setting)					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			Ref. chapter 5.1.1
USB1			X		Ref. chapter 5.1.1
USB2	N/A	N/A	N/A	N/A	
USIF0	X				Ref. chapter 5.1.1
USIF1				RTD	

Tab. 4: #PORTCFG=0, with USB Cable

Fig. 3 and Fig. 4 show details concerning the logical connections between external physical ports and internal 'Access points'.

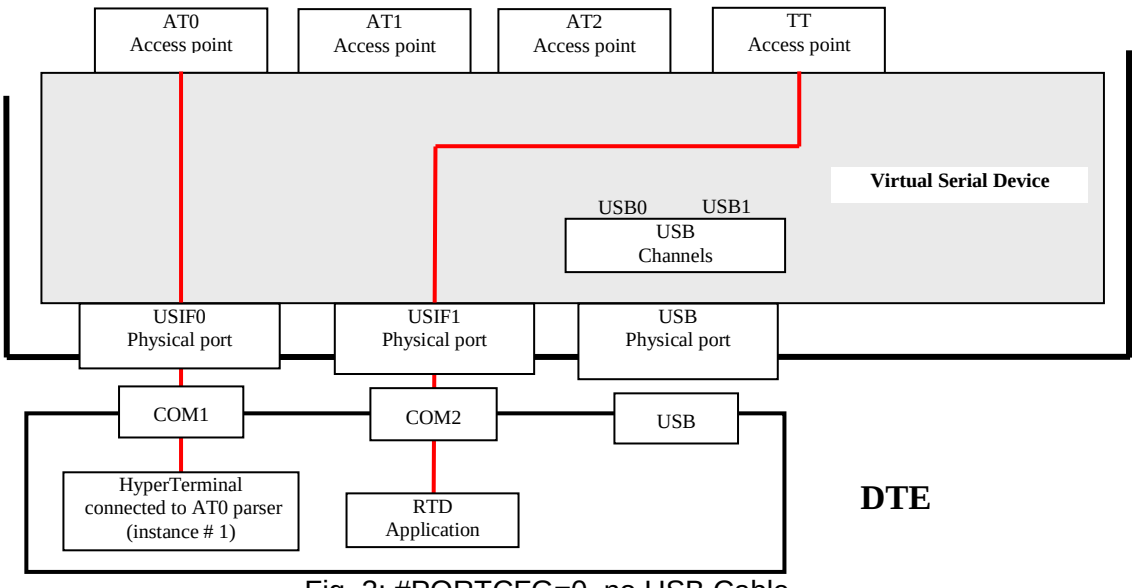

Fig. 3: #PORTCFG=0, no USB Cable

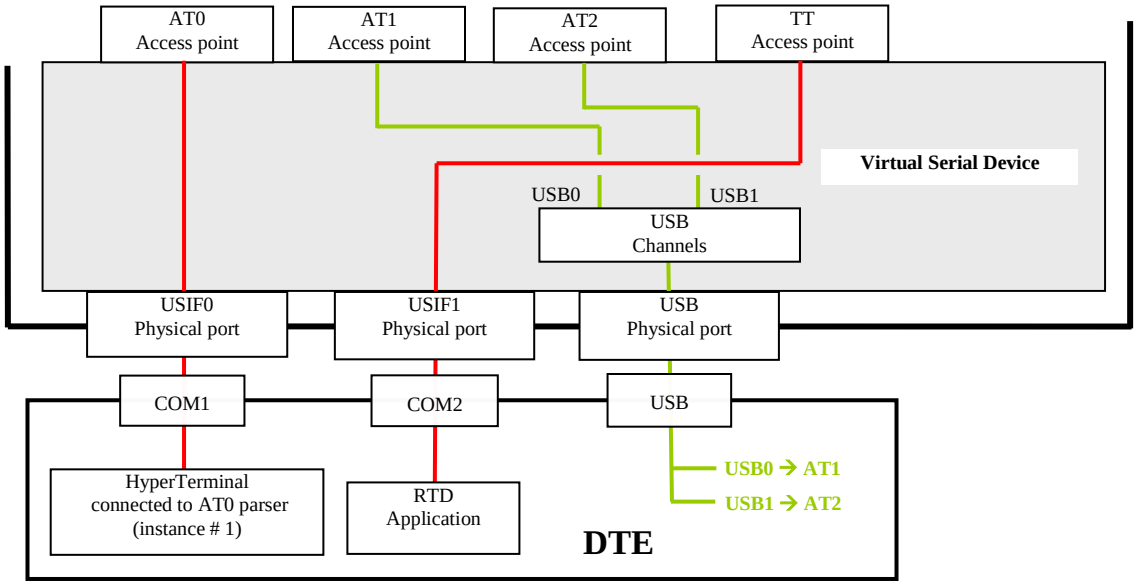

Fig. 4: #PORTCFG=0 + USB Cable

NOTICE: RTD tool is available for the end user.

3.2. AT#PORTCFG=1

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=1					
	AT0	AT1	AT2	TT	GPS/NMEA
No USB cable					
USIF0	X				Ref. chapter 5.1
USIF1	N/A	N/A	N/A	N/A	

Tab. 5: #PORTCFG=1, no USB Cable

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=1					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			Ref. chapter 5.1
USB1			X		Ref. chapter 5.1
USB2				RTD	
USIF0	X				Ref. chapter 5.1
USIF1	N/A	N/A	N/A	N/A	

Tab. 6: #PORTCFG=1, with USB Cable

NOTICE: The GPS/NMEA column is valid only for modules equipped with GNSS receiver.

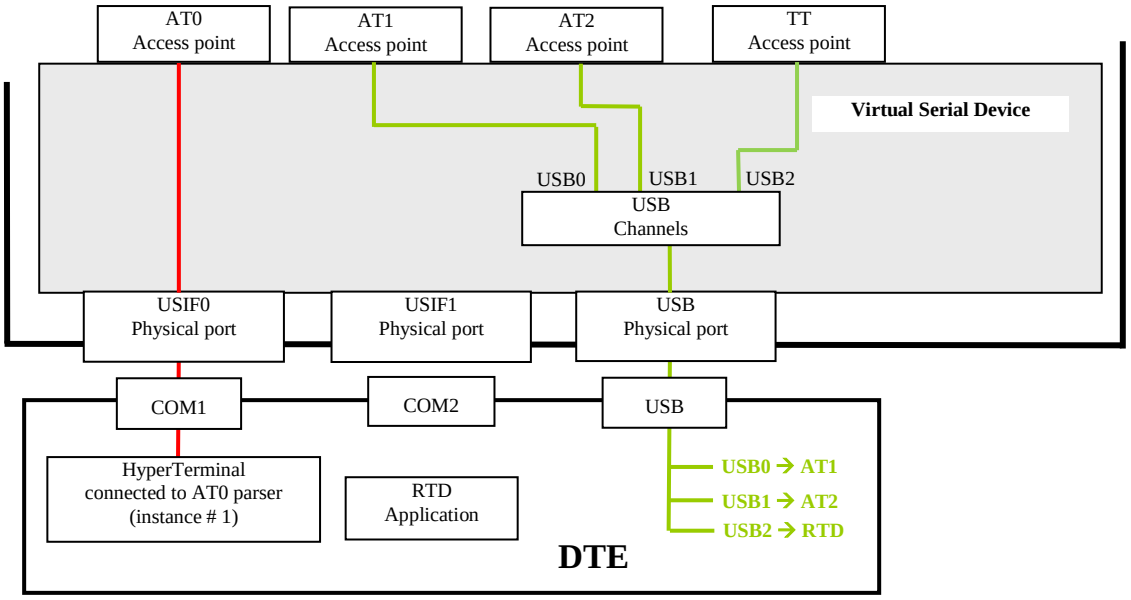

Fig. 5: #PORTCFG=1 + USB Cable

3.3. AT#PORTCFG=3

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=3					
	AT0	AT1	AT2	TT	GPS/NMEA
No USB cable					
USIF0	X				Ref. chapter 5.1
USIF1		X			Ref. chapter 5.1

Tab. 7: #PORTCFG=3, no USB Cable

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=3					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0			X		Ref. chapter 5.1
USB1	N/A	N/A	N/A	N/A	
USB2				RTD	
USIF0	X				Ref. chapter 5.1
USIF1		X			Ref. chapter 5.1

Tab. 8: #PORTCFG=3, with USB Cable

NOTICE: The GPS/NMEA column is valid only for modules equipped with GNSS receiver.

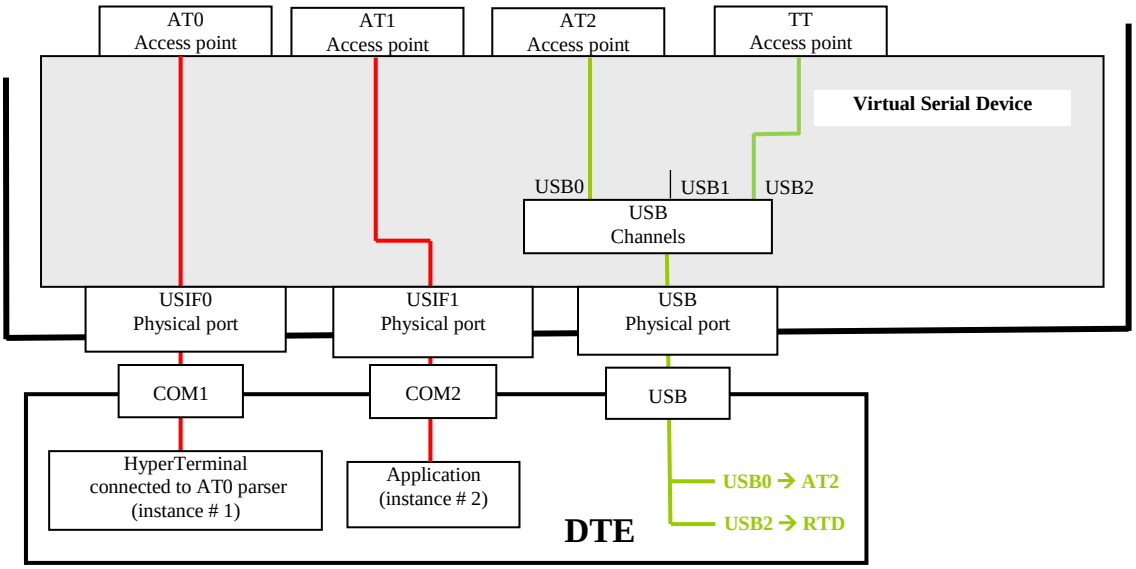

Fig. 6: #PORTCFG=3 + USB Cable

3.4. AT#PORTCFG=4

SW version equal or higher than 13.00.xx7					
AT#PORTCFG=4					
	AT0	AT1	AT2	TT	GPS/NMEA
No USB cable					
USIF0		X			Ref. chapter 5.1
USIF1	N/A	N/A	N/A	N/A	

Tab. 9: #PORTCFG=4, no USB Cable

SW version equal or higher than 13.00.xx7					
AT#PORTCFG=4					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0	X				Ref. chapter 5.1
USB1			X		Ref. chapter 5.1
USB2				RTD	
USIF0		X			Ref. chapter 5.1
USIF1	N/A	N/A	N/A	N/A	

Tab. 10: #PORTCFG=4, with USB Cable

NOTICE: The GPS/NMEA column is valid only for modules equipped with GNSS receiver.

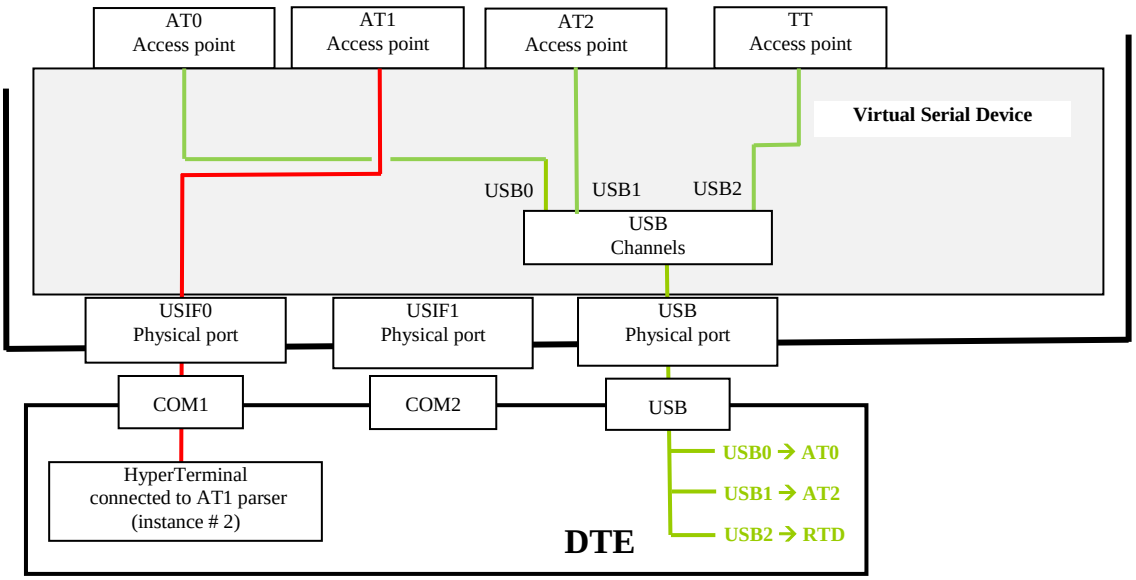

Fig. 7: #PORTCFG=4 + USB Cable

3.5. AT#PORTCFG=5

SW version equal or higher than 13.00.xx7					
AT#PORTCFG=5					
	AT0	AT1	AT2	TT	GPS/NMEA
No USB cable					
USIF0		X			Ref. chapter 5.1
USIF1				RTD	

Tab. 11: #PORTCFG=5, no USB Cable

SW version equal or higher than:13.00.xx7					
AT#PORTCFG=5					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0	X				Ref. chapter 5.1
USB1			X		Ref. chapter 5.1
USB2					
USIF0		X			
USIF1				RTD	

Tab. 12: #PORTCFG=5, with USB Cable

NOTICE: The GPS/NMEA column is valid only for modules equipped with GNSS receiver.

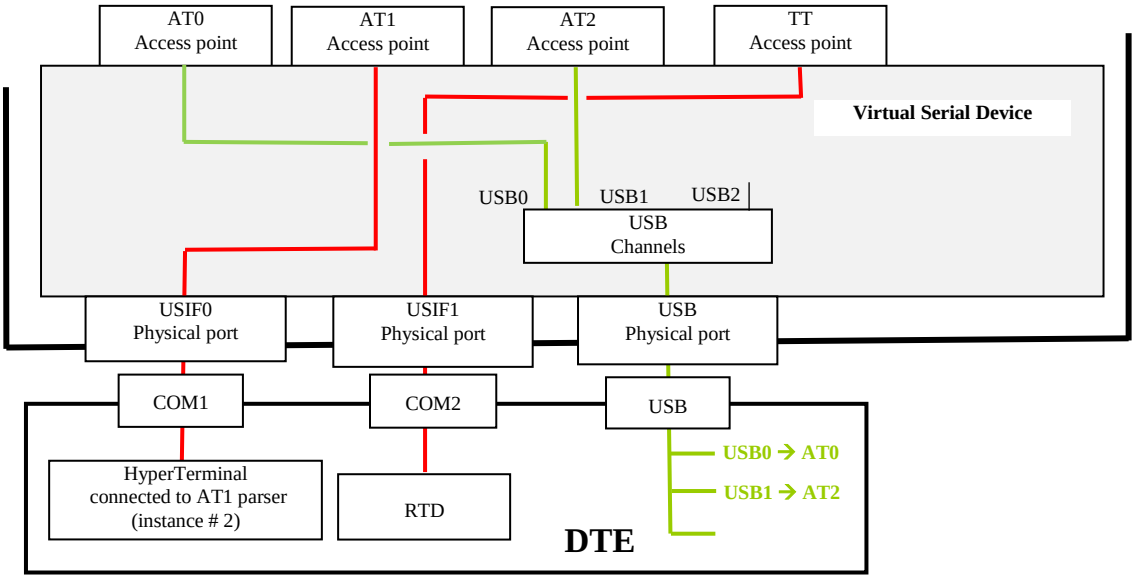

Fig. 8: #PORTCFG=5 + USB Cable

3.6. AT#PORTCFG=8

SW version equal or higher than 13.00.xx6				
AT#PORTCFG=8				
	AT0	AT1	AT2	TT
No USB cable				
USIF0	X			
USIF1				RTD

Tab. 13: #PORTCFG=8, no USB Cable

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=8					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			
USB1			X		
USB2					Ref. chapter 5.1.2.1
USIF0	X				
USIF1				RTD	

Tab. 14: #PORTCFG=8, with USB Cable

NOTICE: The GPS/NMEA column is valid only for modules equipped with GNSS receiver.

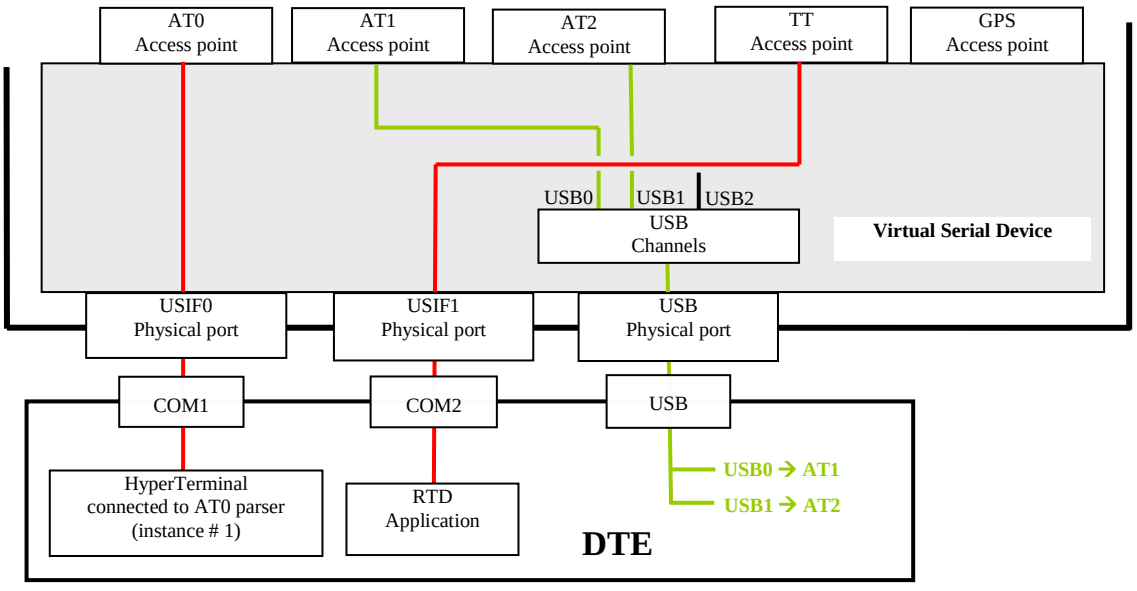

Fig. 9: #PORTCFG=8 + USB Cable

3.7. AT#PORTCFG=9

SW version 13.00.xx5				
AT#PORTCFG=9				
	AT0	AT1	AT2	TT
No USB cable		X		
USIF0	X			
USIF1				RTD

SW version equal or higher than 13.00.xx6				
AT#PORTCFG=9				
	AT0	AT1	AT2	TT
No USB cable		X		
	N/A	N/A	N/A	N/A
USIF0	X			
USIF1				RTD

Tab. 15: #PORTCFG=0, no USB Cable

SW version 13.00.xx5					
AT#PORTCFG=9					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			
USB1					Ref. chapter 5.1.2.2
USIF0	X				
USIF1				RTD	

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=9					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			
USB1					Ref. chapter 5.1.2.2
USB2	N/A	N/A	N/A	N/A	N/A
USIF0	X				
USIF1				RTD	

Tab. 16: #PORTCFG=0, with USB Cable

NOTICE: The GPS/NMEA column is valid only for modules equipped with GNSS receiver.

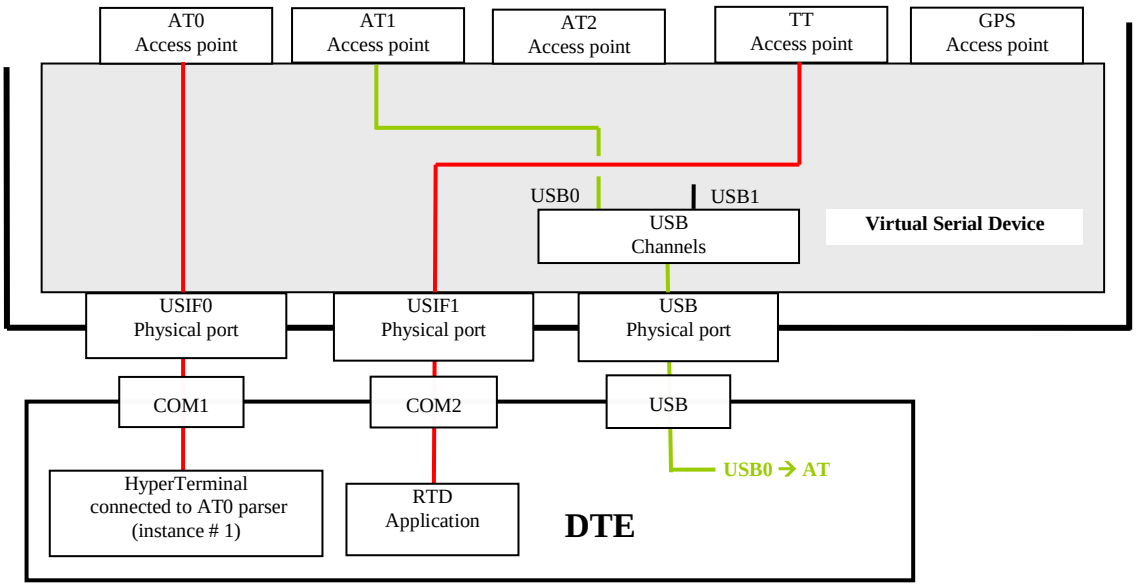

Fig. 10: #PORTCFG=9 + USB Cable

4. CMUX Protocol

This section shows examples of ports arrangement using CMUX protocol. If you need to develop a Multiplexing Protocol running on your application processor (e.g. a user micro-controller), refer to document [1] to get detailed information.

4.1. #PORTCFG=0, no USB + CMUX

The module is configured as indicated in Fig. 3: #PORTCFG=0 (factory setting), no USB cable is plugged in. The used DTE is a Windows-PC, and Fig. 11 shows its serial ports configuration. Now, run on the DTE the TELIT Serial Port MUX application, configure the virtual serial ports of MUX as shown in Fig. 12, and connect logically the virtual serial ports to the physical port COM1, refer to Fig. 13. When the user starts an application (e.g. Hyper Terminal) connected to one of the three virtual ports (the forth one is spare), TELIT Serial Port MUX application sends automatically the AT+CMUX=0 command to the module and the CMUX protocol is activated.

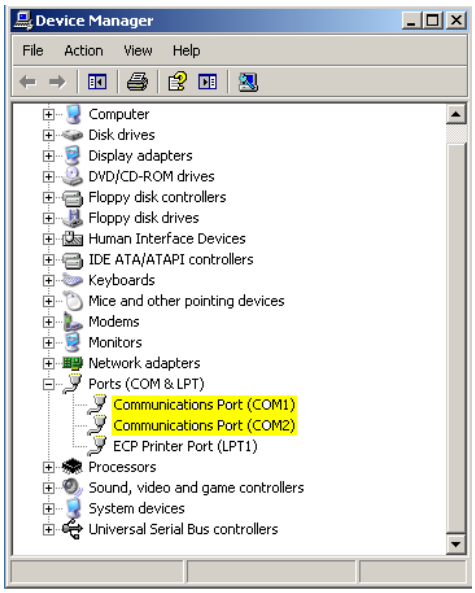

Fig. 11: Physical COMx Ports

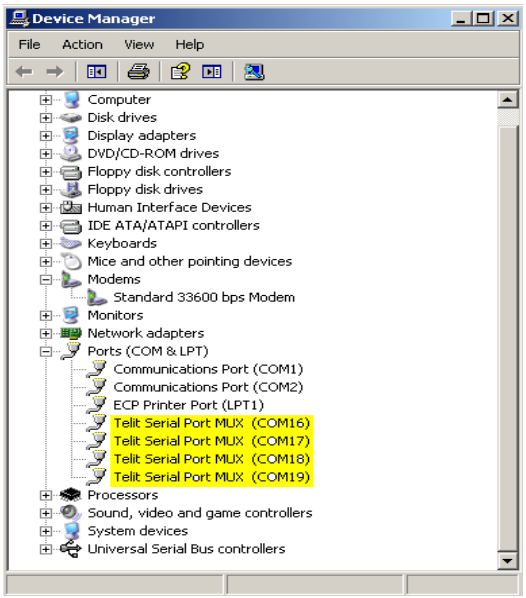

Fig. 12: Virtual Serial Ports of MUX

NOTICE: the configuration of the virtual serial ports of the TELIT Serial Port MUX application must avoid conflict with the physical or virtual serial ports already present on the Windows-PC (DTE).

Tab. 17 summarizes the VCOMx←→COM1/USIF0←→VCx configuration and the RTD application connection via USIF1/COM2 physical ports. The Fig. 13 shows the connections details.

Module ←→ DTE connection	VCOMx → VCx	AT0	AT1	AT2	TT
USB not used					
USIF0 ←→COM1	VCOM16→VC1	X			
	VCOM17→VC2		X		
	VCOM18→VC3			X	
	VCOM19→VC4				
USIF1 ←→ COM2					RTD

Tab. 17: CMUX Connected to USIF0

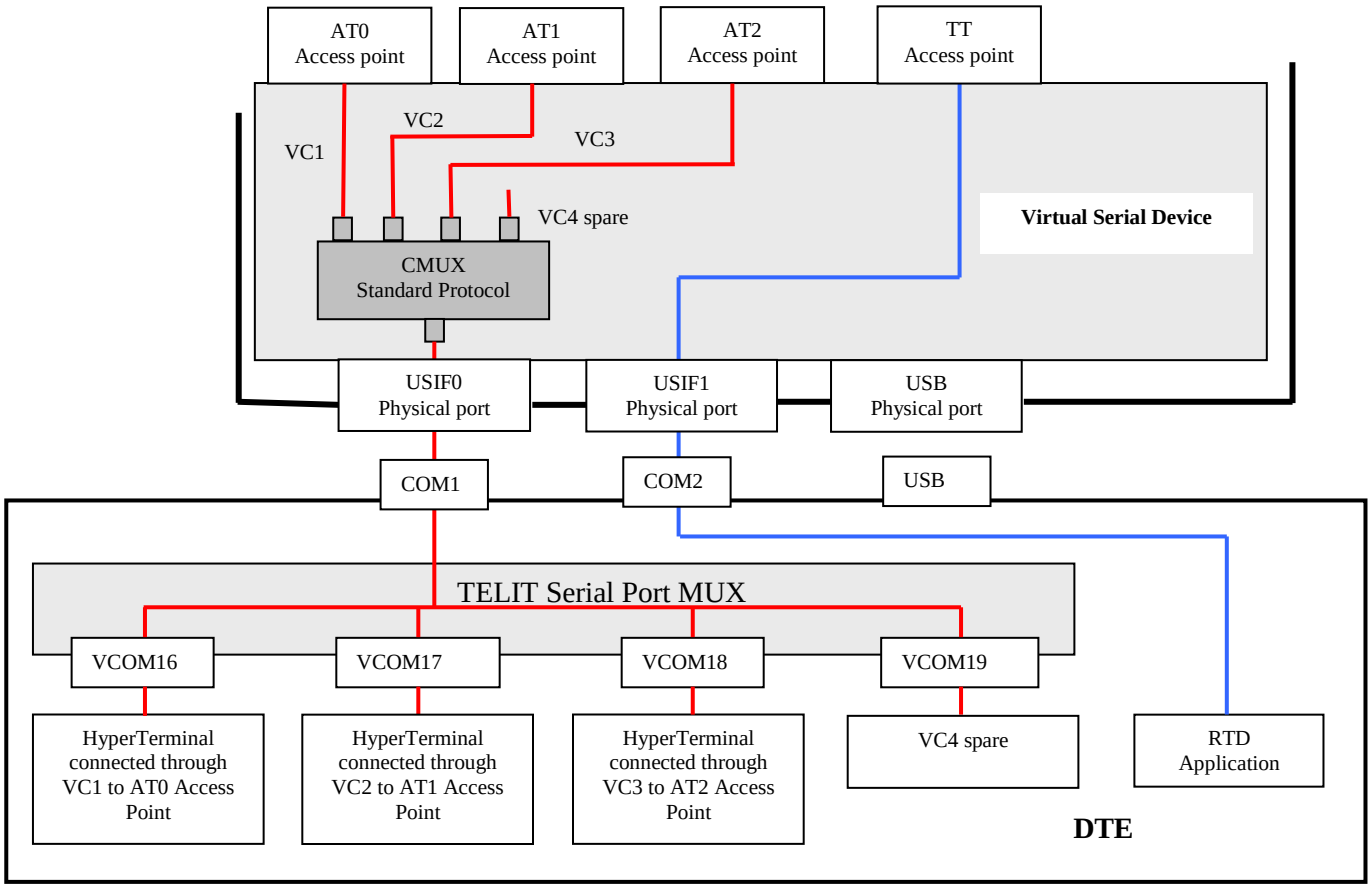

Fig. 13: CMUX Connected to USIF0

4.2. #PORTCFG=0 + USB + CMUX

The module is configured as indicated in Fig. 4: #PORTCFG=0 (factory setting) plus USB cable plugged in. The used DTE is a Windows-PC, and Fig. 14 shows its serial ports configuration (two USB channels are available). Now, run on the DTE the TELIT Serial Port MUX application, configure the virtual serial ports of MUX as shown in Fig. 14, and connect logically the virtual serial ports to the USB1 channel mapped into VCOM22 virtual port as shown on Tab. 18, and Fig. 15. When the user starts an application (e.g. Hyper Terminal) connected to one of the three Virtual Ports (the fourth one is spare), TELIT Serial Port MUX application sends automatically the AT+CMUX=0 command to the module and the CMUX protocol is activated.

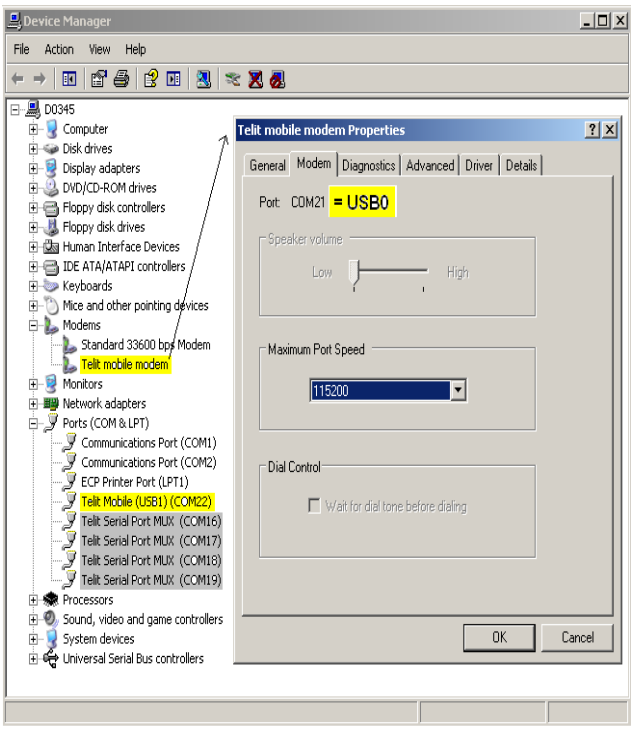

Fig. 14: Virtual Serial Ports of TELIT MUX

As previously described, the used GE910 module provides two USB channels, see the figure on the left side. In this example the mapping is:

- USB0 channel → COM21 (or VCOM21)
- USB1 channel → COM22 (or VCOM22)

In addition, the figure shows the virtual serial ports generated by the TELIT Serial Port MUX application.

NOTICE: the configuration of the virtual serial ports of the TELIT Serial Port MUX application must avoid conflict with the physical or virtual serial ports already present on the Windows-PC (DTE).

The table below summarizes the new ports arrangement.

Module ←→ DTE connection	Channels	USBx → VCOM	VCOMx → VCx	AT0	AT1	AT2	TT
USB ←→ USB	USB0						
	USB1	VCOM22	VCOM16 → VC1	X			
			VCOM17 → VC2		X		
			VCOM18 → VC3			X	
			VCOM19 → VC4				
USIF0 not used							
USIF1 ←→ COM2							RTD

Tab. 18: CMUX Connected to USB1 Channel

Referring to Fig. 15: AT0 (instance # 1) is disconnected from USIF0 and connected to VC1←→USB1 channel←→USB physical port←→VCOM22←→VCOM16←→Hyper Terminal. Instead, the RTD stays on USIF1.

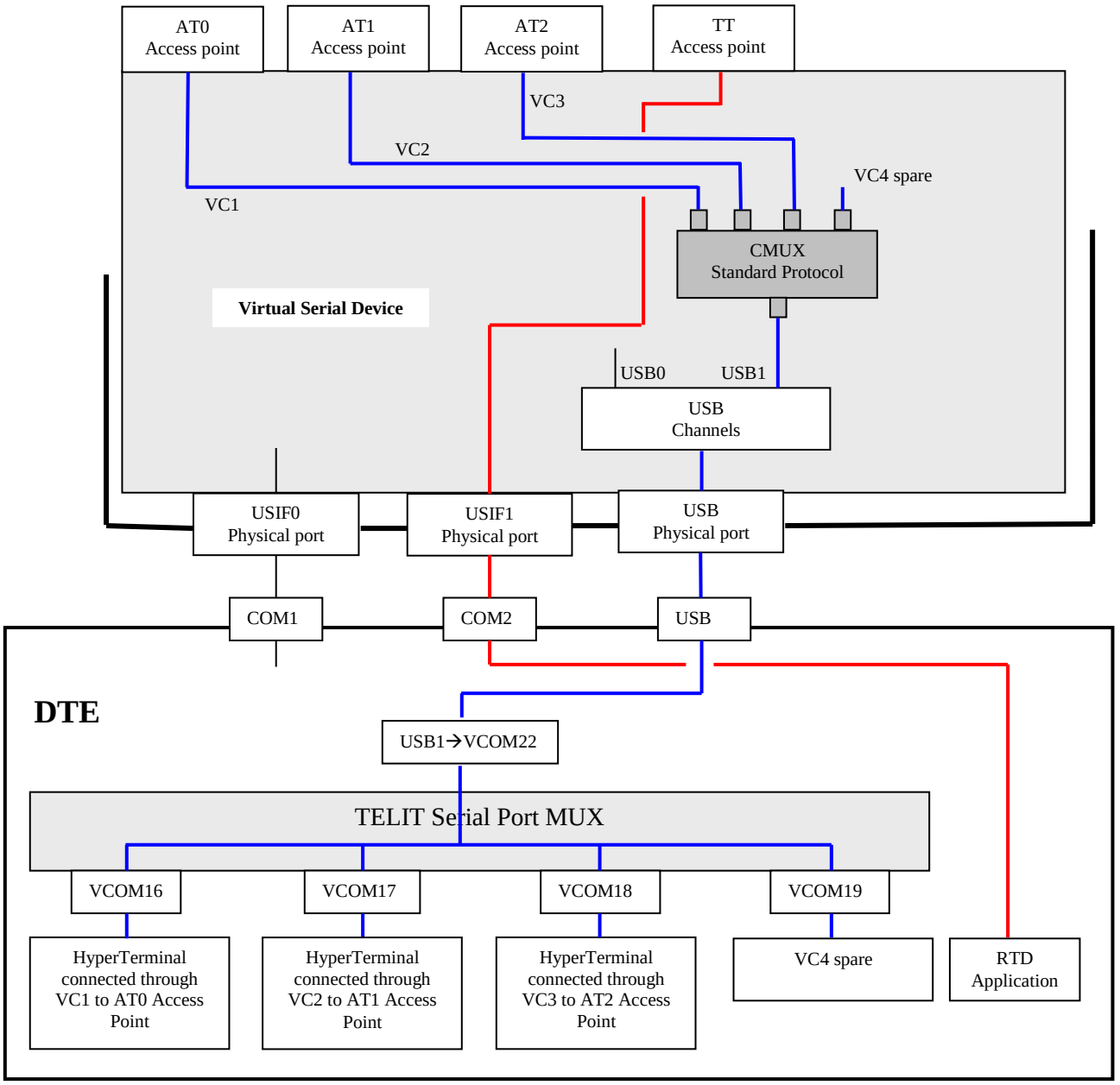

Fig. 15: CMUX Connected to USB1 Channel

5. Services

The GE910 Family provides the following services:

- GNSS
- Python

Different ‘Access Points’ connect services to the Virtual Serial Device layer. The present section describes how the user can access the selected service by means of the external physical serial ports, which, in their turn, connect to the Virtual Serial Device.

5.1. GNSS

GE910-GNSS module is equipped with GNSS receiver. Refer to document [4] to have information on the satellite constellations (GPS, GLONASS, etc.) supported by the GNSS receiver.

The built-in GNSS receiver can send NMEA sentences on different physical ports, in accordance with the current PORTCFG variant. In general, NMEA sentences run on the physical port used by the operator to enter the AT$GPSP and AT$GPSNMUN commands; in this case AT commands and NMEA sentences share the same physical port at the same time. Refer to document [2] to have information on AT commands syntax.

The following sub-chapters show some detailed examples of logical connections settings.

5.1.1. AT#PORTCFG=0 + USB

Assume that the AT#PORTCFG=0 command has been issued and activated, and USB cable is connected to the module. Tab. 4 (in accordance with the software version) shows the corresponding internal logical connections of the module. Now, enter the AT$GPSP=1 and AT$GPSNMUN=1,... commands, for example, through the physical port USIF0, the NMEA sentences are routed on USIF0. After executing the two AT commands, NMEA sentences and AT commands share the USIF0 port at the same time, see figure below.

AT#PORTCFG=0 (Factory setting)					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			
USB1			X		
USIF0	X				X
USIF1				RTD	

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=0 (Factory setting)					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			
USB1			X		
USB2	N/A	N/A	N/A	N/A	
USIF0	X				X
USIF1				RTD	

Tab. 19: #PORTCFG=0, with USB + GPS/NMEA

NOTICE: the user can issue the AT commands through USIF0/AT0, USB0/AT1, or USB1/AT2 channel/parser. The NMEA sentences are routed respectively on USIF0 (as shown by the example), USB0, or USB1 channel.

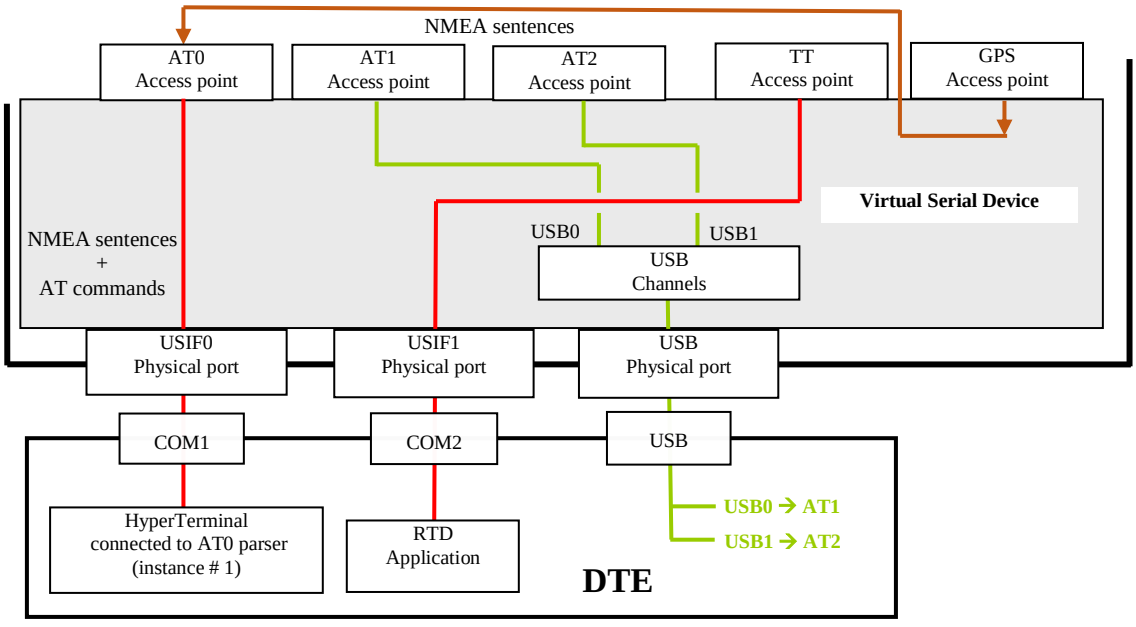

Fig. 16: USIF0 Port Supports NMEA Sentences + AT Commands

5.1.2. GNSS Receiver Control Modes

GNSS Receiver sends NMEA sentences on a physical port in accordance with the selected GNSS Receiver Control Mode and the current PORTCFG variant.

Here are the two GNSS Receiver Control Mode, refer to document [4].

- **External Host Controlled or Standalone Mode**: an external user device (called External/Customer Host) controls the GNSS receiver by means GNSS commands.
- **GSM Controlled Mode**: the GSM side of the module itself controls the GNSS receiver by means of specific AT commands.

The table below describes the procedures to enter one of the two Modes.

GNSS Receiver Control Modes	
External Host Controlled or Standalone Mode	GSM Controlled Mode
• Connect GNSS_NMEA_TX, GNSS_NMEA_RX, and GNSS_EN lines to External/Customer Host. • Drive GNSS_EN line to GND to power ON the GNSS receiver. • When the GNSS receiver is ON, the NMEA sentences run automatically on the GNSS_NMEA_TX line, toward the Host. • Host can send GNSS commands toward GNSS receiver on GNSS_NMEA_RX line. • USIF0 is connected to the Host. If the modem side is powered ON, it is suggested to enter the AT$GPSD=0 command to disable all the AT commands managing GNSS receiver. Power OFF the modem side if AT commands are not used. **NOTICE:** in document [4] USIF0 is called Modem Serial Port 1.	• Do not connect GNSS_EN pin to GND. • Connect USIF0 to External/Customer Host. GNSS_NMEA_TX, GNSS_NMEA_RX lines and USB port are optional. • Power ON the module. Enter the AT$GPSP=1 command to turn ON also the GNSS receiver. • Use AT#PORTCFG=8/9 to route the NMEA sentences on the desired USBx channel. **NOTE:** AT$GPSP=0 is the factory setting; the GNSS receiver is powered OFF.

Tab. 20: GNSS Receiver Control Modes

5.1.2.1. AT#PORTCFG=8, USB2 Channel

The GNSS receiver is in GSM Controlled mode. The AT#PORTCFG=8 connects the GPS Access point to the external USB2 channel. Enable GPS/NMEA sentences via AT$GPSP=1 command entered, for example, through USB0 channel: the NMEA sentences are automatically routed on USB2 channel.

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=8					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			
USB1			X		
USB2					X
USIF0	X				
USIF1				RTD	

Tab. 21: #PORTCFG=8 + NMEA Sentences

NOTICE: the user can issue the AT commands through USIF0/AT0, USB0/AT1, or USB1/AT2 channel/parser. In any case, the NMEA sentences run on USB2 channel.

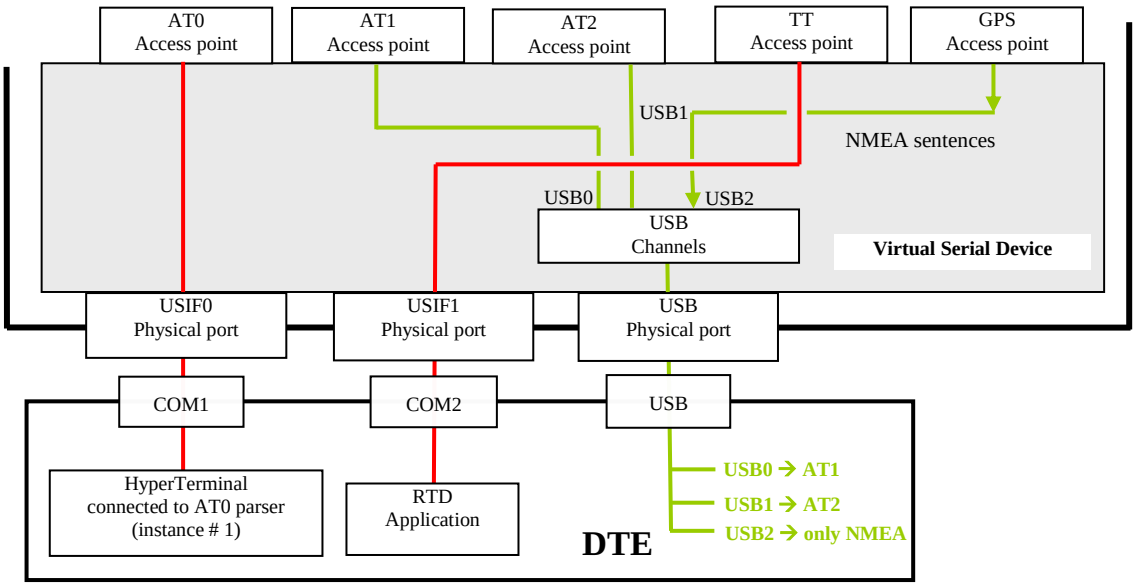

Fig. 17: USB2 Channel Supports only NMEA Sentences

5.1.2.2. AT#PORTCFG=9, USB1 Channel

The GNSS receiver is in GSM Controlled mode. The AT#PORTCFG=9 connects the GPS Access point to the external USB1 channel. Enable GPS/NMEA sentences via AT$GPSP=1 command entered, for example, through USB0 channel: the NMEA sentences are automatically routed on USB1 channel.

SW version 13.00.xx5					
AT#PORTCFG=9					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			
USB1					X
USIF0	X				
USIF1				RTD	

SW version equal or higher than 13.00.xx6					
AT#PORTCFG=9					
	AT0	AT1	AT2	TT	GPS/NMEA
USB0		X			
USB1					X
USB2	N/A	N/A	N/A	N/A	N/A
USIF0	X				
USIF1				RTD	

Tab. 22: #PORTCFG=9 + NMEA Sentences

NOTICE: the user can issue the AT commands through USIF0/AT0, or USB0/AT1 channel/parser. In any case, the NMEA sentences run on USB1 channel.

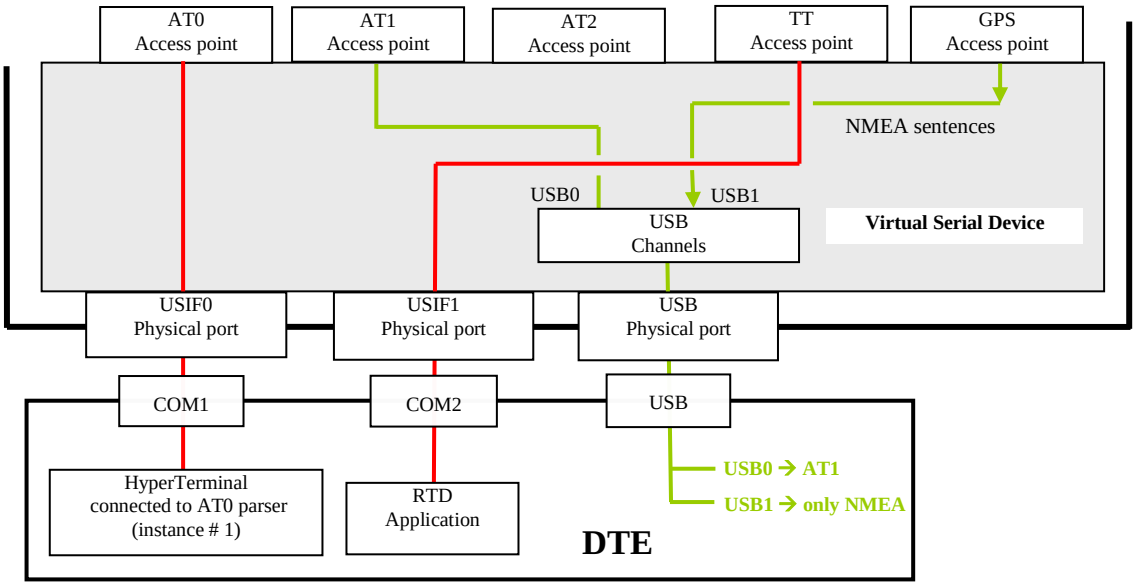

Fig. 18: USB1 Channel Supports only NMEA Sentences

5.2. Python

GE910 modules provide the Python programming language to offer to users a tool to develop scripts in accordance with their application requirements, see document [3]. As shown in Fig. 19, the VSD provides two access points called VHWDTE0 and VHWDTE1. MDM and MDM2 Python modules connect respectively with the two access points.

Assume that the module is using the factory-setting[3] configuration and USB cable is not plugged in; the Fig. 3 depicts the internal logical connections among physical ports and access points, and Tab. 3 summarizes the factory ports arrangement.

When Python script runs the Python instruction ***import MDM***[4], the VSD disconnects the USIF0/AT0 logical connection and establishes the logical connection VHWDTE0/AT0 enabling Python script to access AT0 parser. In the same way, ***import MDM2*** instruction requests to the VSD to establish the logical connection VHWDTE1/AT1. The Fig. 19 shows USIF0 disconnected from AT0 parser, the port is not more usable by an external device.

Python script can run the instruction ***import SER*** to use the USIF0 port. The Fig. 20 shows the USIF0 port connected to Python script through PYSER/SER logical connection.

The Python software modules MDM, MDM2, and SER use the following three independent resources: USIF0 physical port, AT0, and AT1 Access Points. No resources contention can arise among them. As a rule, we can say that the MDM, MDM2, and SER instructions steal the resources regardless their current owner.

As shown in the next pages there are other Python modules to create logical connection between a physical port and an Access point. To have detailed information on the software versions supporting different Python modules refer to document [3]. Here are some figures showing different logical connection configurations.

[3] AT#PORTCFG=0, refer to Chapter 3.
[4] It is assumed that the reader is familiar with Python language.

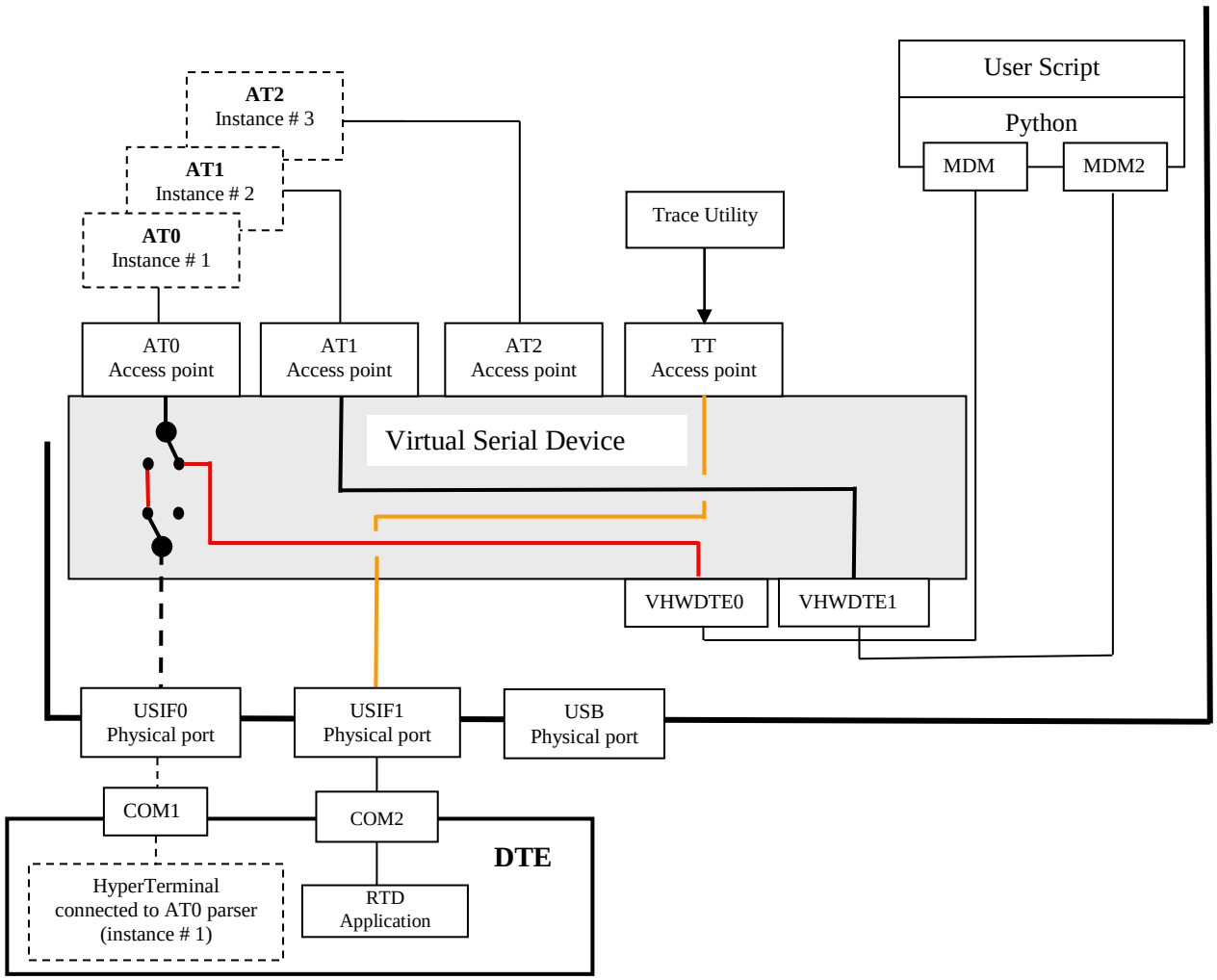

Fig. 19: Python & MDM, MDM2 Modules

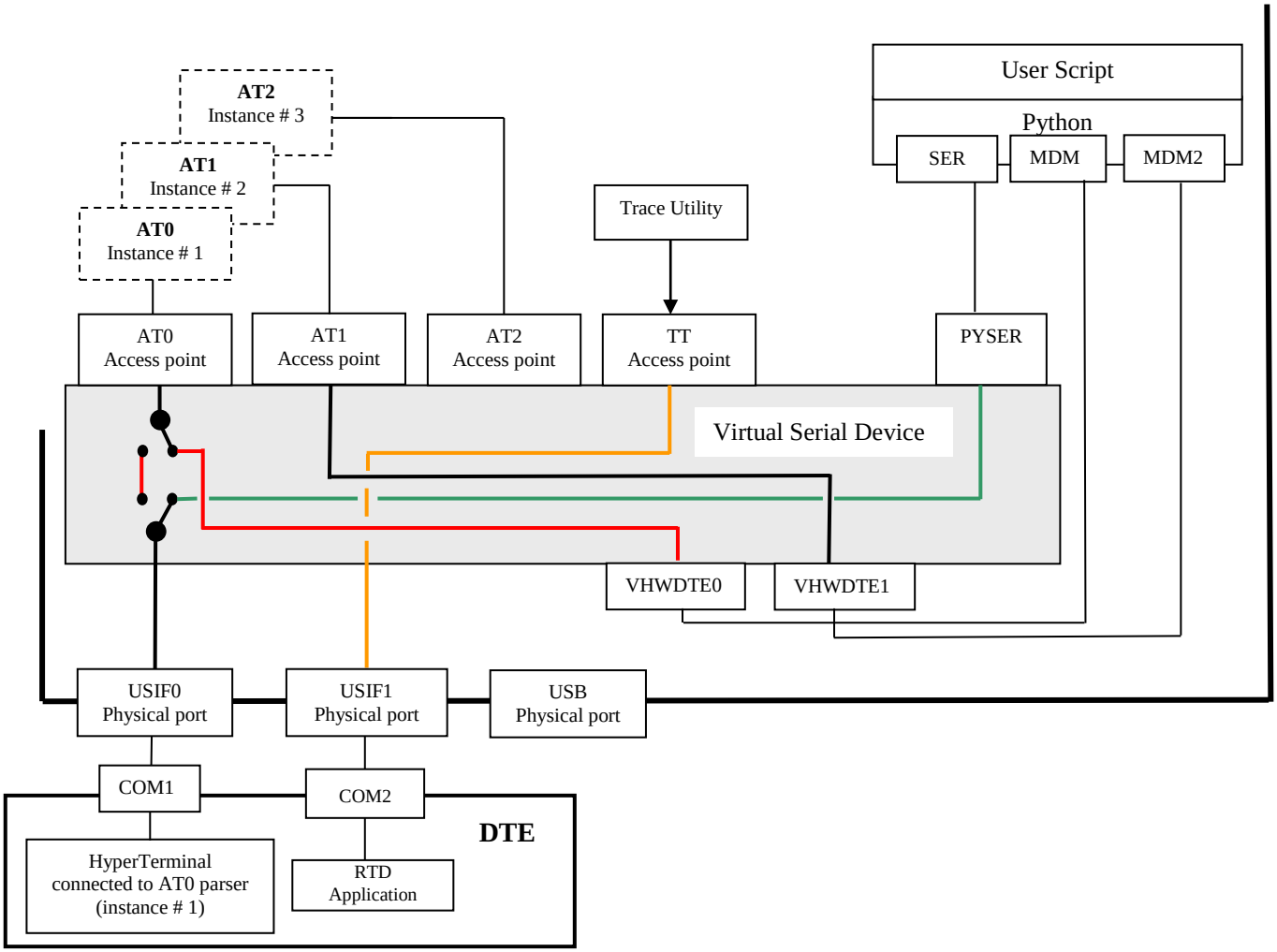

Fig. 20: Python & MDM, MDM2, SER Modules

In accordance with the installed software version, Python script can run the instruction ***import SER2*** to use the USIF1 port. The figure below shows the USIF1port connected to Python script through PYSER2/SER2 logical connection.

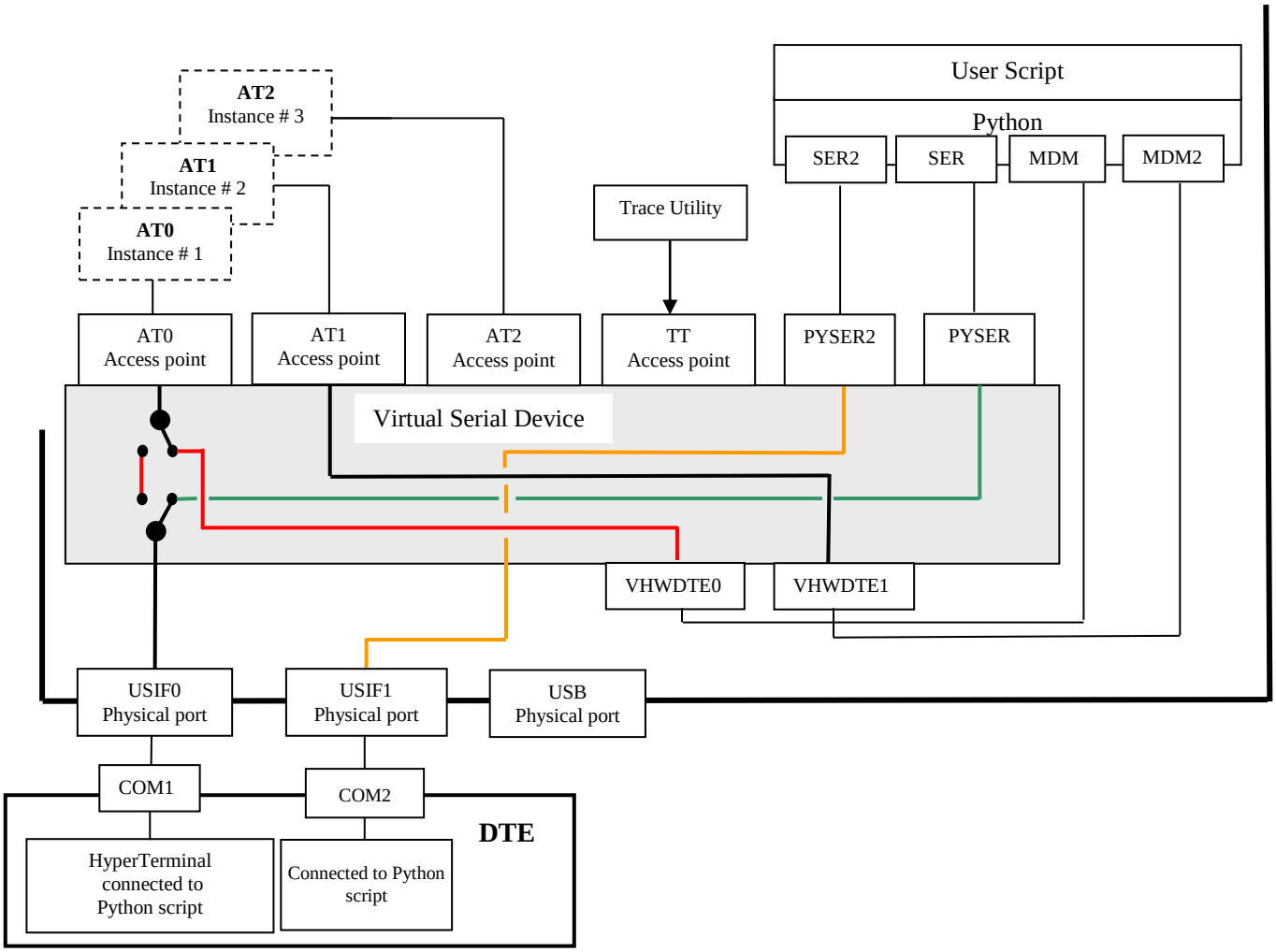

Fig. 21: Python & MDM, MDM2, SER, SER2 Modules

The Python software modules MDM, MDM2, SER, and SER2 use the following four independent resources: USIF0, USIF1 physical ports, and AT0, AT1 Access Points. No resources contention can arise among them. As a rule, we can say that the MDM, MDM2, SER, and SER2 instructions steal the resources regardless their current owner.

In accordance with the installed software version, Python script can run the ***import USB0*** instruction to use the USB0 channel. The figure below shows the USB0 channel connected to Python script through PYUSB0/USB0 logical connection.

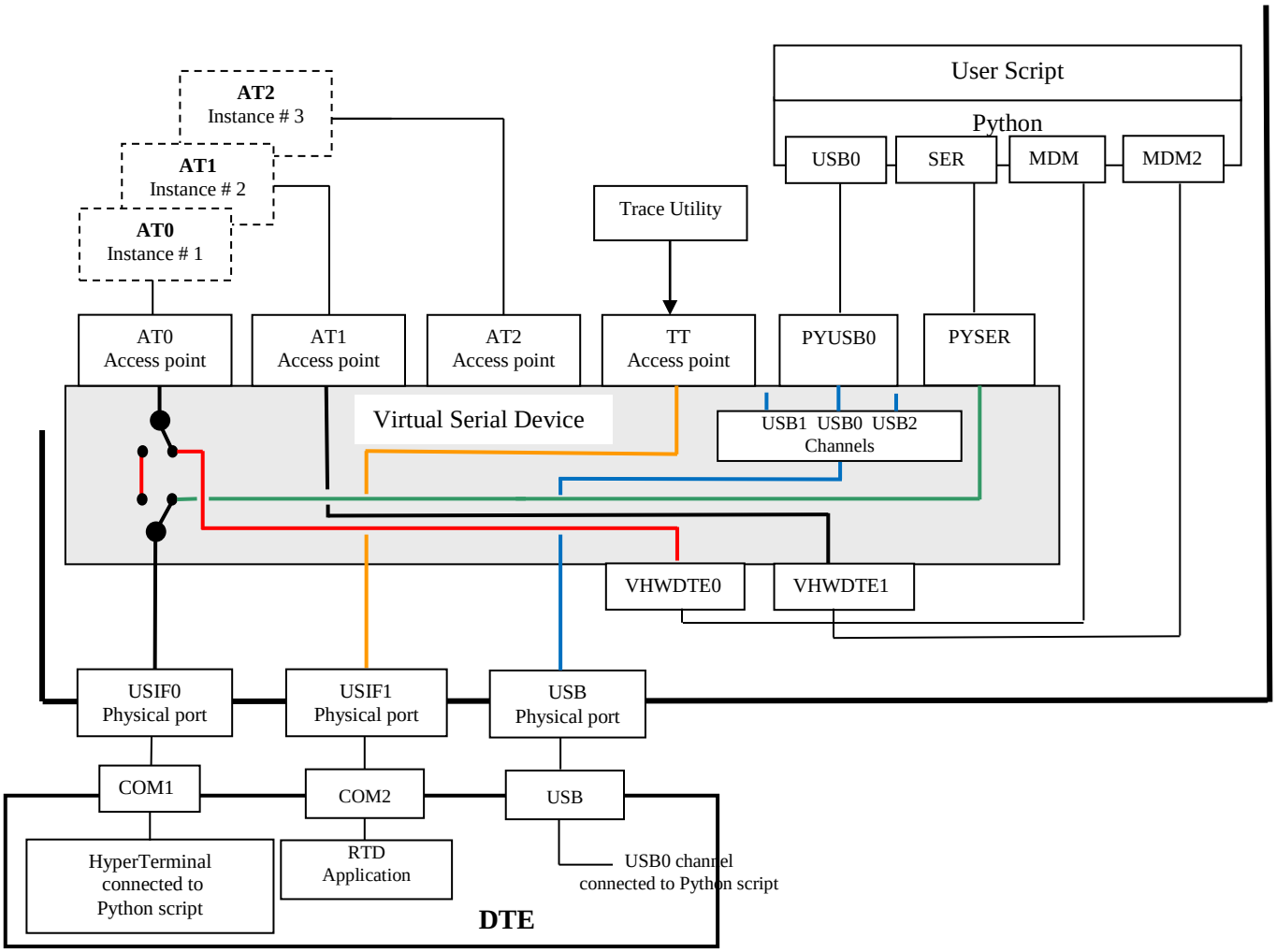

Fig. 22: Python & MDM, MDM2, SER, USB0 Modules

The Python software modules MDM, MDM2, SER, and USB0 use four independent resources: USIF0 physical port, USB0 channel, and AT0, AT1 Access Points. No resources contention can arise among them. As a rule, we can say that the MDM, MDM2, SER, and USB0 instructions steal the resources regardless their current owner.

5.2.1. Python Script Debugging

The user needs to debug a new Python script. The module is using the factory setting #PORTCFG=0 configuration, refer to Tab. 3. Suppose that the Python script runs: ***import MDM, import MDM2, import SER*** and ***print*** instructions. The figure below sketches the results of the execution of the first three instructions:

- AT0 parser connected to Python script through VHWDTE0/MDM logical connection
- AT1 parser connected to Python script through VHWDTE1/MDM2 logical connection
- USIF0 port connected to Python script through PYSER/SER logical connection

Moreover, the figure shows that the ***print*** instruction uses the TT Access point to send print messages to the USIF1 port. The RTD application displays on the DTE, in readable format, the trace data received in binary format from the Trace Utility, and the print Python messages, received from the User Script, in text format.

NOTICE: the Hyper Terminal can only display the print Python messages received in text format.

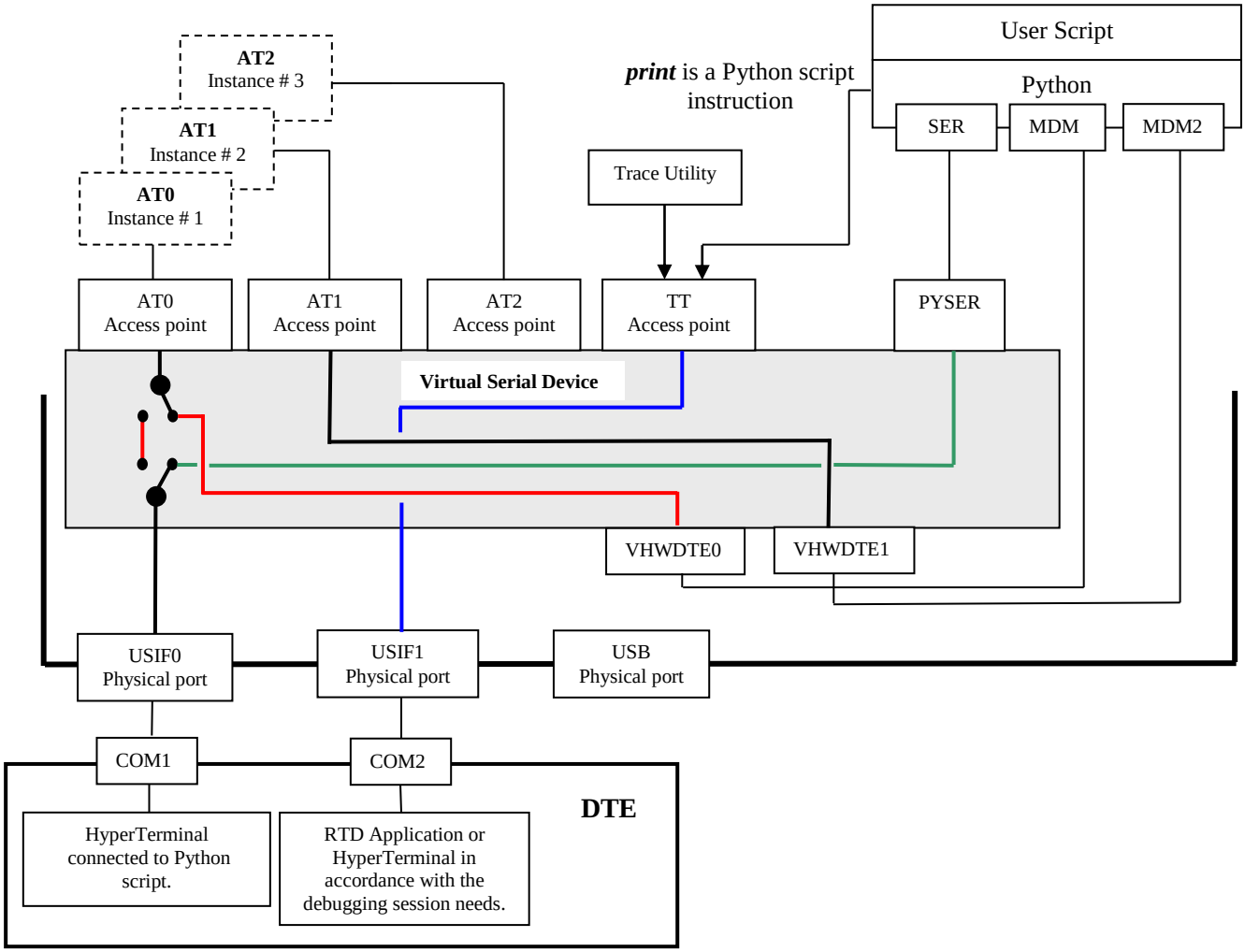

Fig. 23: Python & MDM, MDM2, SER and *print* Modules

6. The Winning Ports Configuration

Here are two examples showing that the last port configuration set by the user overrides the previous one.

There are two ways to change module ports arrangement in addition to use AT#PORTCFG command:

- Plug in/out the USB cable;
- Enter the AT+CMUX=0 command.

NOTICE: TELIT Serial Port MUX application sends automatically the AT+CMUX=0 command to the module, see chapters 4.1, and 4.2.

Example 1

Module: Fig. 3 shows the current ports configuration of the module (factory setting).

User action: the user runs on the Windows-PC the TELIT Serial Port MUX application so configured: Virtual Ports COM16÷COM19 connected logically to COM1, refer to Fig. 12.

PC: it provides the required Virtual Ports. When the user starts an application (e.g. Hyper Terminal) connected to one of the three Virtual Ports (the fourth one is spare), TELIT Serial Port MUX application sends the AT+CMUX=0 command to the module.

Module: in accordance with the received command, the involved AT Parser starts the CMUX protocol. The module enters the configuration shown in Fig. 13.

User action: now, the user plugs in USB cable.

Module: Fig. 4 shows the module configuration.

PC: it provides two new virtual “COM” connected logically to the two USB channels. The CMUX protocol is disabled and the TELIT Serial Port MUX application running on Windows-PC is no more connected to the module, it should be closed. COM1 is ready for new applications.

User action: now, the user disconnects USB cable.

Module: it enters again the configuration shown in Fig. 3.

Example 2

Module: Fig. 3 shows the ports configuration of the module (factory setting).

User action: the user plugs in USB cable.

Module: in accordance with the user action, the module enters the configuration shown in Fig. 4.

PC: it provides two virtual “COM” required by USB port driver to connect logically the two USBx channels.

User action: the user runs on the Windows PC the TELIT Serial Port MUX application so configured:
Virtual Ports COM16÷COM19 logically connected to USB1→VCOM22, refer to Fig. 12.

PC: it provides the required Virtual Ports. When the user starts an application (e.g. Hyper Terminal) on a Virtual Port, TELIT Serial Port MUX sends the AT+CMUX=0 command to the module.

Module: in accordance with the received command, the involved AT Parser starts the CMUX protocol. The module enters the configuration shown in Fig. 15.

User action: now, the user disconnects USB cable.

Module: it enters the configuration shown in Fig. 3.

PC: discards the two virtual “COM” connected logically to the two USBx channels. The CMUX protocol is disabled, TELIT Serial Port MUX application running on Windows PC is no more connected to the module, and it should be closed.

The two examples show that the last required port configuration overrides the previous one.