QTP 16Big

Quick Terminal Panel  16 keys  Big display

USER MANUAL

Via dell' Artigiano, 8/6
40016 San Giorgio di Piano
(Bologna) ITALY
E-mail: grifo@grifo.it
http://www.grifo.it http://www.grifo.com
Tel. +39 051 892.052 (a.r.) FAX: +39 051 893.661

QTP 16Big
Rel. 5.00 Edition 20 November 2006

GPC®, grifo®, are trade marks of grifo®
Operator interface provided with a large dimension display and very low price. Dimension: **96x192 mm**: 9 mm frontal; 34 mm rear. Aluminium Container shape with front plastics frame, provided of mounting Brackets. Front panel with keyboard, protected by anti scratch polyester cover, with **IP 54** protection. Front panel mounting. Equipped with 2 alphanumeric displays by **20x4 Big characters**, either **LCD** or **Fluorescent** type; **LED Backlite** and **Contrast** of **LCD** display regolated by user through software; adjustable **VFD** display **Brightness**. Membrane keypad with 16 keys provided of double serigraphy: numeric and function. Functions of debounce, autorepeat, keycklick and disable for the keys pressed. **5 LEDs** and **Buzzer** managed by software in different modes. Panel name, and/or **LED** indicator, **Personalization Label** slot. 3 relays outputs, completely driven by software. **EEPROMs** for setup, messages, user characters, keys codes, etc. Up to 3371 different messages can be saved and displayed, even with auto scrolling mode. **Real Time Clock** (RTC) backed by dedicated **Lithium** battery; complete **Alarm Clock** capable to drive one relay output. **CAN** communication line provided of proper line driver. Asynchronous serial line configurable in **RS 232, RS 422, RS 485** or **Current Loop**. Synchronous line in **I2C BUS**. Network connection through proper protocols. Local setup for operating modalities selection. Up to 256 **Different Characters** visible; 8 user characters provided of selectable **Pattern**. Wide range **DC** or **AC** power supply from **5 Vdc** to **24 Vac**; total power consumption change according with used configuration, from 0,3 to 6,2 W. On board electronic protection against voltage peaks, by **TransZorb**. Comfortable connectors for a fast cablation, with standard pin outs. Possibility to require customized keyboard and program packages.
IMPORANT

Although all the information contained herein have been carefully verified, grifo® assumes no responsibility for errors that might appear in this document, or for damage to things or persons resulting from technical errors, omission and improper use of this manual and of the related software and hardware.

grifo® reserves the right to change the contents and form of this document, as well as the features and specification of its products at any time, without prior notice, to obtain always the best product.

For specific informations on the components mounted on the card, please refer to the Data Book of the builder or second sources.

SYMBOLS DESCRIPTION

In the manual could appear the following symbols:

- Attention: Generic danger
- Attention: High voltage
- Attention: ESD sensitive device

Trade Marks

abcde® , GPC®, grifo® : are trade marks of grifo®.

Other Product and Company names listed, are trade marks of their respective companies.
GENERAL INDEX

INTRODUCTION ................................................................................................................... 1

VERSION ................................................................................................................................. 3

GENERAL INFORMATION ...................................................................................................... 4
BUZZER .................................................................................................................................... 7
EEPROM ................................................................................................................................. 7
KEYBOARD ............................................................................................................................. 7
DISPLAY ................................................................................................................................... 8
ON BOARD POWER SUPPLY ................................................................................................. 8
I2C BUS LINE ......................................................................................................................... 8
CAN INTERFACE .................................................................................................................... 9
ASYNCRONOUS SERIAL LINE ............................................................................................... 9
REAL TIME CLOCK .............................................................................................................. 10
RELAYS OUTPUTS .................................................................................................................. 10
QTP 16BIG LIBRARY .............................................................................................................. 10

TECHNICAL FEATURES ......................................................................................................... 11
GENERAL FEATURES ........................................................................................................... 11
PHYSICAL FEATURES ............................................................................................................ 12
ELECTRIC FEATURES ........................................................................................................... 14

INSTALLATION ........................................................................................................................ 15
CONNECTIONS ...................................................................................................................... 16
CN4 - POWER SUPPLY CONNECTOR .................................................................................. 16
CN6 - CAN INTERFACE CONNECTOR .................................................................................. 18
CN3 - I2C BUS LINE CONNECTOR ...................................................................................... 20
CN7 - RELAYS OUTPUTS CONNECTOR ................................................................................ 22
CN5 - SERIAL LINE CONNECTOR .......................................................................................... 23
J1- BOOT LOADER ACTIVAT ION CONNECTOR ................................................................ 28
JUMPERS ................................................................................................................................ 30
OPTIONAL EEPROM PROTECTION ..................................................................................... 31
BACK UP ................................................................................................................................ 31
SERIAL LINE CONFIGURATION ............................................................................................ 32
RL2 RELAY ENABLE ............................................................................................................... 34
POWER SUPPLY .................................................................................................................... 36
CAN INTERFACE CONNECTION .......................................................................................... 37

SOFTWARE DESCRIPTION ................................................................................................... 38
LOCAL SET UP ....................................................................................................................... 38
COMMUNICATION BUFFERS ................................................................................................. 39
DATA STORED ON EEPROM ............................................................................................... 40
KEYBOARD ACQUISITION ..................................................................................................... 40
KEYS CODES .......................................................................................................................... 41
CHARACTERS VISUALIZATION ON DISPLAY ...................................................................... 42
CONTRAST REGULATION ...................................................................................................... 43
COMMUNICATION MODALITIES ................................................................. 44
MASTER-SLAVE 9 BITS COMMUNICATION ................................................. 44
I2C BUS COMMUNICATION ........................................................................ 48
NORMAL COMMUNICATION ........................................................................... 51
HOW TO START .............................................................................................. 52
DEMO PROGRAMS .......................................................................................... 53

COMMANDS ............................................................................................... 54
COMMANDS FOR CURSOR POSITION ......................................................... 54
CURSOR LEFT .............................................................................................. 54
CURSOR RIGHT .......................................................................................... 54
CURSOR DOWN .......................................................................................... 54
CURSOR UP ................................................................................................. 55
HOME .......................................................................................................... 55
CARRIAGE RETURN ................................................................................. 55
CARRIAGE RETURN+LINE FEED ................................................................. 55
ABSOLUTE PLACEMENT OF ALPHANUMERIC CURSOR ....................... 55
COMMANDS FOR CHARACTERS ERASURE ........................................... 56
BACKSPACE .............................................................................................. 56
CLEAR PAGE ............................................................................................. 56
CLEAR LINE ............................................................................................... 56
CLEAR END OF LINE ................................................................................ 56
CLEAR END OF PAGE .............................................................................. 56
COMMANDS FOR CURSOR ATTRIBUTES MANAGEMENT .................... 57
CURSOR OFF ............................................................................................. 57
STEADY STATIC CURSOR ON ................................................................. 57
BLINKING BLOCK CURSOR ON ............................................................... 57
COMMANDS FOR GENERAL FUNCTIONS ............................................ 58
READ Firmware VERSION ........................................................................ 58
READ CARD CODE .................................................................................. 58
FLUORESCENT DISPLAY BRIGHTNESS SETTING .................................. 58
OPERATING MODE SELECTION ................................................................. 59
COMMUNICATION RESET ....................................................................... 59
GENERAL RESET .................................................................................... 59
BEEP .......................................................................................................... 60
BUZZER, LEDS, BACKLIGHT ACTIVATION ........................................... 60
COMMANDS FOR EEPROM ................................................................. 62
REQUEST FOR EEPROM AVAILABILITY .............................................. 62
WRITE OF PRESENCE BYTE ................................................................. 62
READ PRESENCE BYTE ........................................................................ 62
WRITE BYTE ON EEPROM ................................................................. 63
READ BYTE FROM EEPROM ............................................................... 63
COMMANDS FOR KEYBOARD MANAGEMENT ................................... 64
KEY RECONFIGURATION ........................................................................ 64
KEYCLICK ON WITHOUT MEMORIZATION ........................................ 64
KEYCLICK OFF WITHOUT MEMORIZATION ........................................ 64
KEYCLICK ON WITH MEMORIZATION .................................................. 65
KEYCLICK OFF WITH MEMORIZATION .................................................. 65
COMMANDS FOR USER CHARACTERS ................................................................. 66
DEFINITION OF USER CHARACTER ............................................................. 67
DEFINITION AND MEMORIZATION OF USER CHARACTER ......................... 67
COMMANDS FOR MESSAGE MANAGEMENT ................................................... 68
READING OF MAX MESSAGE NUMBER ......................................................... 68
READING OF LAST GROUP AND MESSAGE MANAGED ................................. 69
SELECT CURRENT MESSAGE GROUP .......................................................... 69
MESSAGE STORAGE ..................................................................................... 69
MESSAGE READING ................................................................................... 70
VISUALIZATION OF MESSAGES ................................................................. 70
SCROLLING MESSAGES VISUALIZATION ..................................................... 71
SET AUTOMATIC VISUALIZATION ............................................................. 72
COMMANDS FOR I2C BUS COMMUNICATION AS MASTER ....................... 74
START I2C BUS .......................................................................................... 74
STOP I2C BUS ............................................................................................ 74
TRANSMIT BYTE ON I2C BUS .................................................................... 75
RECEIVE BYTE FROM I2C BUS ................................................................. 75
COMMANDS FOR SRAM AND CLOCK ......................................................... 76
WRITE BYTE ON BACKED SRAM ............................................................... 76
READ BYTE FROM BACKED SRAM ........................................................... 76
SET CLOCK ............................................................................................... 77
ACQUIRE CLOCK ...................................................................................... 77
SHOW TIME ON DISPLAY ............................................................................ 78
SHOW DATE ON DISPLAY .......................................................................... 79
SET CLOCK ALARM .................................................................................. 80
ACQUIRE CLOCK ALARM ........................................................................... 81
COMMANDS FOR RELAYS OUTPUTS MANAGEMENT .................................... 82
WRITE ALL DIGITAL OUTPUTS ................................................................. 82
ENABLE SINGLE DIGITAL OUTPUT ............................................................ 82
DISABLE SINGLE DIGITAL OUTPUT ............................................................ 84

APPENDIX A: COMMANDS SUMMARY TABLES .......................................... A-1

APPENDIX B: DISPLAY CHARACTERS ......................................................... B-1

APPENDIX C: MOUNTING NOTES ............................................................... C-1
TERMINAL DIMENSIONS ........................................................................... C-1
FRONT PANEL MOUNTING ......................................................................... C-3
PERSONALIZATION LABEL INSERTION ..................................................... C-4
FIXING FRONT PANEL TO CONTAINER ..................................................... C-5

APPENDIX D: VIEW AREA AND CHARACTERS DIMENSIONS ....................... D-1

APPENDIX E: DEFAULT CONFIG., OPTIONS, ACCESSORIES ...................... E-1

APPENDIX F: ALPHABETICAL INDEX ........................................................... F-1
FIGURES INDEX

FIGURE 1: LOCATION OF HARDWARE AND FIRMWARE VERSION ................................................................. 3
FIGURE 2: AVAILABLE MODELS .............................................................................................................. 5
FIGURE 3: QTP 16Big COMPLETE VIEW ................................................................................................. 9
FIGURE 4: REAR VIEW .......................................................................................................................... 13
FIGURE 5: CONSUMPTIONS TABLE ........................................................................................................ 14
FIGURE 6: LOCATION OF JUMPERS, CONNECTORS, BUZZER, BATTERY, ETC. ................................. 15
FIGURE 7: CN4 - POWER SUPPLY CONNECTOR .................................................................................. 16
FIGURE 8: AC POWER SUPPLY 8+24 Vac ........................................................................................... 17
FIGURE 9: DC POWER SUPPLY +10+38 Vdc ..................................................................................... 17
FIGURE 10: STABILIZED POWER SUPPLY +5 Vdc (OPTION) ............................................................. 17
FIGURE 11: CN6 - CAN INTERFACE CONNECTOR ............................................................................. 18
FIGURE 12: CAN LINE CONNECTION ................................................................................................... 18
FIGURE 13: CAN NETWORK CONNECTION EXAMPLE ......................................................................... 19
FIGURE 14: CN3 - I2C BUS LINE CONNECTOR .................................................................................... 20
FIGURE 15: CONNECTION EXAMPLE FOR I2C BUS POINT TO POINT COMMUNICATION ............... 20
FIGURE 16: CONNECTION EXAMPLE FOR I2C BUS NETWORK COMMUNICATION ............................ 21
FIGURE 17: CN7 - RELAYS OUTPUTS CONNECTOR ........................................................................... 22
FIGURE 18: RELAYS OUTPUTS CONNECTION ...................................................................................... 22
FIGURE 19: CN5 - SERIAL LINE CONNECTOR ....................................................................................... 23
FIGURE 20: RS 232 POINT TO POINT CONNECTION EXAMPLE ..................................................... 24
FIGURE 21: RS 422 POINT TO POINT CONNECTION EXAMPLE ..................................................... 24
FIGURE 22: RS 485 POINT TO POINT CONNECTION EXAMPLE ..................................................... 24
FIGURE 23: RS 485 NETWORK CONNECTION EXAMPLE ..................................................................... 25
FIGURE 24: CURRENT LOOP 4 WIRES POINT TO POINT CONNECTION EXAMPLE ........................ 26
FIGURE 25: CURRENT LOOP 2 WIRES POINT TO POINT CONNECTION EXAMPLE ........................ 26
FIGURE 26: CURRENT LOOP NETWORK CONNECTION EXAMPLE .................................................. 27
FIGURE 27: J1 - BOOT LOADER ACTIVATION CONNECTOR ............................................................... 28
FIGURE 28: COMPONENTS MAP SOLDER SIDE .................................................................................... 29
FIGURE 29: COMPONENTS MAP COMPONENTS SIDE ......................................................................... 29
FIGURE 30: JUMPERS TABLE ................................................................................................................ 30
FIGURE 31: LOCATIONS OF DRIVERS FOR SERIAL COMMUNICATION ................................................ 33
FIGURE 32: QTP 16Big-C4 ...................................................................................................................... 35
FIGURE 33: QTP 16Big-F4 ...................................................................................................................... 35
FIGURE 34: POWER SUPPLY EXP5-1 ................................................................................................. 37
FIGURE 35: KEYS NUMBERS AND LOCATION .................................................................................... 41
FIGURE 36: DEFAULT KEYS CODES ...................................................................................................... 42
FIGURE 37: FLOW CHART FOR MASTER-SLAVE 9 BITS COMMUNICATION ....................................... 46
FIGURE 38: EXAMPLE OF MASTER-SLAVE 9 BITS COMMUNICATION .............................................. 47
FIGURE 39: FLOW CHART FOR MASTER -> QTP 16Big COMMUNICATION IN I2C BUS .................. 48
FIGURE 40: FLOW CHART FOR QTP 16Big -> MASTER COMMUNICATION IN I2C BUS ............... 49
FIGURE 41: I2C BUS NETWORK CONNECTION .................................................................................. 50
FIGURE 42: FLOW CHART FOR NORMAL COMMUNICATION ............................................................. 51
FIGURE 43: RS 232 CONNECTION WITH PC ........................................................................................ 52
FIGURE 44: LEDs NUMERATION, POSITION AND COLOUR .............................................................. 61
FIGURE 45: FRONT PANEL WITH KEYBOARD ..................................................................................... 65
FIGURE 46: USER CHARACTERS PATTERN ................................................................. 66
FIGURE 47: NUMBER OF MESSAGES ON EEPROM .................................................... 68
FIGURE 48: CONNECTION OF I2C BUS LINE AS MASTER ........................................... 75
FIGURE 49: REAL TIME CLOCK PARAMETERS ............................................................ 77
FIGURE 50: AVAILABLE CONNECTIONS DIAGRAM ................................................... 83
FIGURE A1: COMMAND CODES SUMMARY TABLE (1 OF 4) ........................................ A-1
FIGURE A2: COMMAND CODES SUMMARY TABLE (2 OF 4) ........................................ A-2
FIGURE A3: COMMAND CODES SUMMARY TABLE (3 OF 4) ........................................ A-3
FIGURE A4: COMMAND CODES SUMMARY TABLE (4 OF 4) ........................................ A-4
FIGURE B1: CHARACTERS TABLE OF QTP 16Big-F4 ............................................... B-1
FIGURE B2: CHARACTERS TABLE OF QTP 16Big-C4 ............................................... B-2
FIGURE C1: QTP 16Big DIMENSIONS ................................................................. C-1
FIGURE C2: MOUNTING CLAMP DIMENSIONS ...................................................... C-2
FIGURE C3: QTP 16Big + MOUNTING CLAMP VIEW ................................................ C-2
FIGURE C4: BREAKING FOR INSTALLATION ............................................................. C-3
FIGURE C5: PERSONALIZATION LABEL DIMENSIONS .......................................... C-4
FIGURE C6: PERSONALIZATION LABEL INSERTION ............................................... C-4
FIGURE C7: SCREWS FOR FRONT PANEL FIXING ................................................... C-5
FIGURE D1: DISPLAY DIMENSIONS OF QTP 16Big-C4 ........................................... D-1
FIGURE D2: DISPLAY DIMENSIONS OF QTP 16Big-F4 ........................................... D-2
FIGURE E1: LOCAL SETUP DEFAULT CONFIGURATION ........................................... E-1
FIGURE E2: JUMPERS DEFAULT CONFIGURATION ................................................... E-1
FIGURE E3: OPTIONS TABLE .................................................................................. E-2
FIGURE E4: AMP2.CABLE CONNECTION ACCESSORY ....................................... E-2
FIGURE E5: CKS.AMP2 CONNECTION ACCESSORY ............................................. E-3
FIGURE E6: AMP4.CABLE CONNECTION ACCESSORY ....................................... E-3
FIGURE E7: CKS.AMP4 CONNECTION ACCESSORY ............................................. E-4
INTRODUCTION

The use of these devices has turned - IN EXCLUSIVE WAY - to specialized personnel.

This device is not a safe component as defined in directive 98-37/CE.

Pins of module are not provided with any kind of ESD protection. Many pins of the card are directly connected to their respective pins of on board's components and these last are sensitive to electrostatic noises. So personnel who handles the product/s is invited to take all necessary precautions that avoid possible damages caused by electrostatic discharges.

The purpose of this handbook is to give the necessary information to the cognizant and sure use of the products. They are the result of a continual and systematic elaboration of data and technical tests saved and validated from the manufacturer, related to the inside modes of certainty and quality of the information.

The reported data are destined- IN EXCLUSIVE WAY- to specialized users, that can interact with the devices in safety conditions for the persons, for the machine and for the enviroment, impersonating an elementary diagnostic of breakdowns and of malfunction conditions by performing simple functional verify operations , in the height respect of the actual safety and health norms.

The informations for the installation, the assemblage, the dismantlement, the handling, the adjustment, the reparation and the contingent accessories, devices, installation, etc. are destined - and then executable - always and in exclusive way from specialized warned and educated personnel, or directly from the AUTHORIZED TECHNICAL ASSISTANCE, in the height respect of the manufacturer recommendations and the actual safety and health norms.

The devices can't be used outside a box. The user must always insert the cards in a container that respect the actual safety normative. The protection of this container is not threshold to the only atmospheric agents, but specially to mechanic, electric, magnetic, etc. ones.

To be on good terms with the products, is necessary guarantee legibility and conservation of the manual, also for future references. In case of deterioration or more easily for technical updates, consult the AUTHORIZED TECHNICAL ASSISTANCE directly.
To prevent problems during card utilization, it is a good practice to read carefully all the informations of this manual. After this reading, the user can use the general index and the alphabetical index, respectively at the beginning and at the end of the manual, to find information in a faster and more easy way.

grifo® provides this documentation "as is" without warranty of any kind. In no event shall grifo® be liable for indirect, special, incidental or consequential damages of any kind arising from any error in this documentation, including any loss or interruption of business, profits, use, or data. Moreover, it is not guaranteed the updating of the product for new computers or new operating systems, that will become available in the future.

All trademarks listed in this manual are copyright of the relative manufacturers.
This handbook makes reference to printed circuit version **110705** and to firmware version **2.1** and following ones. The validity of the information contained in this manual is subordinated to the version numbers on the used panel, and so the user must always verify the correct correspondence between the notations. The version numbers are reported in several places on the electronic part of the product, and following figure shows the most accessible ones. Obviously if the version must be checked, then it must be extracted from the metallic container: a simple pressure on **QTP 16Big** connectors, or on the printed circuit reachable from rear container window, is sufficient. When on the front panel there are two black screws, they must be previously unscrewed (for details see **APPENDIX C**).

![Figure 1: Location of hardware and firmware version](image)

The firmware version number can be also directly required to the terminal by using a dedicated command.

Normally the **QTP 16Big** is always supplied with the latest firmware version that is available but, for specific requirements, the user can receive also a different version; he must carefully specify this particular condition in the order.
GENERAL INFORMATION

QTP 16Big basically is an operator interface provided with a large dimension display. Among the most important aims of QTP 16Big we can remind the representation of information that are visible even from a long distance, and the simplified input of user selections; moreover the availability of interesting additional features make it the right component to solve many applications in the civil, domestic and/or industrial fields, always by keeping an optimum price/performace ratio.

QTP 16Big is available with Alphanumeric displays by 20 characters for 4 lines in two different types: LCD with LED backlite or Fluorescent. In addition on the front panel there are: a 16 Keys membrane keyboard, a personalization label slot (used to carry a name or the user's own logo), and five signalling LEDs with different colours.

A practical and robust metallic container, in aluminium shape with the standard DIN 96x192 size allows a direct mounting in front panel modality. A back side openings allows to reach the connectors that can be used for the required wirings. The enclosed brackets supplied with QTP 16Big let the user mount and/or unmount the terminal by performing a simple rectangle digging up, on the support panel, that normally is the front side of the electric box.

QTP 16Big is the best choice whenever the user needs to show information, messages, status, etc. and the 16 keys are sufficient for user interaction. QTP 16Big gives the possibility to store in the onboard serial EEPROM up to 3371 messages. These messages can be shown on the display, also in sliding mode, by simply sending a proper command sequence, through communication line. With this feature the master program space and its execution time are optimized or even erased, in fact messages must not be sent to the panel every time, they are already stored inside EEPROM of the QTP 16Big. Furthermore it is possible to get messages back through the communication line and read them again. So QTP 16Big can be used as little mass memory where the user can save and read set-up informations, passwords, identification codes, etc. The horizontal scrolling attribute for the saved messages, let the user displays more information on less space: on the first row of the display up to 200 characters can be shown in a self managed sliding modality.

The module's asynchronous serial line can be buffered with the most frequently used electric protocols and thanks to this feature the QTP 16Big can be connected to each systems available on the market. Furthermore low cost networks of QTPs can be realized where many different Operator Panel can be contemporaneously managed. Alternatively the terminal can be driven through a synchronous I2C BUS line that allows the connection on local networks. The interconnection with other devices is ensured also by the optional CAN line, that increase the possible applications fields and improves overall network performances.

CPU section features 16K FLASH with ISP interface for comfortable programming through the asynchronous serial line. This allows an easy development environment for the user application that doesn't require any additional system, with a considerable cost reductions. The user can write the application program with 8051 compatible code, by using one of the numerous comfortable development tools.

The QTP 16Big is able to execute an entire range of display commands including: clear the entire screen or part of it, cursor position and movement, buzzer activation, characters definition, messages management, etc., with command codes compatible to ADDS Viewpoint standard. Many other commands allow the use of the other resources of the operator panel, at high level; in other words the user doesn't have to directly interact on the hardware sections but he must simply use the provided commands.

Main features of QTP 16Big, including the available options, are as follows:
- Dimension: front size 96x192 mm; fore depth of 9 mm; back depth of 34 mm.
- Remarkably low price.
- Aluminium container with frontal frame in plastic, provided with mounting clamps.
- Front panel with keyboard and display window protected by anti scratch polyester cover.
- Membrane keypad with 16 keys provided of double serigraphy: numeric and function.
- Debouncing, autorepeat and keycklick functions for the keys pressed.
- The code of the pressed key can be changed and moreover the not used keys can be even disabled.
- Surface or flush panel mounting.
- IP 54 standard protection on front side.
- 2 different models, with different alphanumeric displays type:
  QTP 16Big-C4: LCD display, LED backlite, with 4 lines of 20 characters
  QTP 16Big-F4: Fluorescent display with 4 lines of 20 characters
- Characters dimensions:
  QTP 16Big-C4: 4,8 x 8,1 mm
  QTP 16Big-F4: 3,6 x 7,7 mm
- LED backlite of LCD display with software management in different modes.
- Comfortable regulation of LCD display contrast and VFD display brightness in order to obtain always the best visibility in any environmental conditions.
- **Buzzer** for BELL, keypressed and acoustic signals, all driven by software.
- 5 coloured status **LEDs** indicator managed by software in different modes.
- **Personalization label slot** for panel name, and/or name of one LED indicator.
- **I51 family microprocessor**, with software selectable clock.
- Different memory types: 16K FLASH EPROM; 2K FLASH EPROM for Boot Loader; 0,5K RAM; up to 64K+2K EEPROM; 240 bytes backed SRAM.
- 3 relays outputs, up to 5 A.
- EEPROMs used for permanent storage of setup parameters, messages, user characters, keys codes, etc.
- Memorization on **EEPROMs** and visualization on display, of maximum 3371 different messages, even with auto scrolling mode.
- The text messages managed by firmware reduce the user program and thus the communicated data.
- Possibility to save and load data from the on board non volatile memories (backed SRAM and EEPROM).
- Asynchronous serial line with RS 232 or RS 422, RS 485, passive Current Loop electric protocols.
- Synchronous **I2C BUS** communication line.
- **CAN** communication line provided of proper line driver.
- **Network** connection through the available serial line and proper logic protocols, up to 256 different units.
- Several **physic protocols** selectable for all the serial communication lines.
- Functionality as **serial <-> I2C BUS converter**, capable to manage each peripheral devices with this interface (temperature sensors, A/D and D/A converters, etc.).
- **Local setup** for required operating modalities.
- 8 user characters provided of selectable patterns.
- Up to 256 different characters defined on display and thus visible.
- Real Time Clock (RTC) backed by proper Lithium battery.
- Possibility to set and acquire the Real Time Clock and its seven temporal parameter (hours, minutes, seconds, day, month, year and week day). The current date and time can be autonomously visualized on display, with attributes and positions defined by user.
- Management of a complete **alarm clock** that can be set on hour, minutes, seconds, day and month. When the alarm time is reached a proper relay output is enabled, with a selectable activation endurance time.
- Management of 3 relays digital outputs either with single settings or group settings.
- **Transparent** functional modality: the data received by user program, if they are not commands, are directly visualized on display while the keys pressed and possible responses of the commands are returned to the same program. This modality is normally defined dumb terminal.
- Tens of commands dedicated to visualization and other operations, compatible with ADDS View-Point standard.
- Possibility to enable an autonomous visualization with different attributes that is automatically shown at power on.
- 6 comfortable and standard connectors for a fast cablation.
- Wide range **DC** or **AC** power supply from 5 Vdc to 24 Vac.
- Total **power consumption** change according with used configuration, from 0,3 to 6,2 W.
- On board electronic protection against voltage peaks, by TransZorb™.
- For specific requirements about front panel, consumption, functionality and price, please contact directly grifo®.

Here follows a description of the board's functional blocks, with an indication of the operations performed by each one.
**BUZZER**

*QTP 16Big* has a circuitry that generates a steady sound, based on a capacitive buzzer. By software, through some specific commands, this circuitry can be enabled, disabled or intermittent, it can generate a simple beep, it can signal a key pressed and it can signalize possible malfunctions. When, after a power on, the card generates a fixed or intermittent sound and it doesn't work correctly, there is a wrong condition that must be resolved: please contact *grifo®* technicians.

**EEPROM**

*QTP 16Big* has a base EEPROM (2 KBytes) for storing set up, communication protocol, identification name, keys codes, user characters patterns, messages, and so on. Many of the stored data have vital importance so a serial EEPROM has been chosen to obtain the best warranties on validity and maintenance of the saved information, even when power supply is not available. It is really interesting the feature of 20 characters messages that can be first saved and then read or shown on the display at any moments, just giving a proper command to the terminal, with the right message identification number or numbers. *QTP 16Big* also manages the visualization of these messages in scrolling mode: on a single line it shows more text than it could be visible in normal condition. The number of managed messages can be increased by ordering the *QTP 16Big* with one of the optional and additional EEPROM:

- EE128 (16 Kbytes)
- EE256 (32 Kbytes)
- EE512 (64 Kbytes)

For detailed information about messages please read COMMANDS FOR MESSAGES MANAGEMENT paragraph.

**KEYBOARD**

*QTP 16Big* has a membrane keyboard with 16 keys located around the display that offer a cheap solution for user data input even when the data are heterogeneous and complex. All the keys are metallic dome type so they provide a tactile sensation of the key pressed and they withstand the knocks and bumps of industrial life. All the keys have a standard label (see figure 45) that satisfy the normal man-machine interface requirements. Remarkable is the presence of numeric digits, the whole alphabet and some functions that allows to input any kind of data and to execute any kind of command. Moreover the keys are equipped with autorepeat and they are totally software reconfigurable or on the other hand the code returned when a key is pressed can be changed or disabled. It is also possible to switch on/off the keyclick function, i.e the buzzer short activation each time a key is pressed. Three keys are used to define some of the functional parameters, as described in proper paragraph LOCAL SETUP. In addition, a personalization label can be added on the frontal of the keyboard to customize and/or identify the terminal, as described in APPENDIX C.
DISPLAY

**QTP 16Big** is available with two alphanumeric displays, provided of 20x4 big size characters: fluorescent and backlit LCD. LEDs backlighting of LCD model ensures a good visibility even when the environmental lighting changes and if it necessary the user can modify the contrast and brightness regulation by software. Another important features of **QTP 16Big** displays is their wide viewing angle that allows a good visibility from each frontal position. The big sizes of characters increase the capability to read the information on displays even at a distance of some metres, without any difficulty.

Further information on each display are reported in TECHNICAL FEATURES chapter and APPENDIX B and D.

The user must choose the right display (and so the right **QTP 16Big** model) that is sufficient for his visibility requirements. For specific requirements on current consumption, visibility and price, the card can be provided even with LCD display not backlit; for detailed information about these options and their availability, please contact directly **grifo®** offices.

ON BOARD POWER SUPPLY

One of the most important peculiarity of **QTP 16Big** is its own switching power supply that requires an input voltage variable from 8÷24 Vac or 10÷38 Vdc; this section generates all the voltages used by the module.

As alternative, **QTP 16Big** without power supply can be ordered (by using the codes .5Vdc or .ALIM): in this case a +5 Vdc stabilized power supply must be provided by an external source.

For detailed information about power supply section, please refer to ELECTRIC FEATURES and POWER SUPPLY paragraphs.

I2C BUS LINE

Through the synchronous serial interface in I2C BUS the **QTP 16Big** can perform two different communications:

- **slave mode** = the command unit operates as a master and communicate to **QTP** either the commands and the responses through the I2C BUS line; it is supported the communication in short local networks, with other units of the same and/or different type.
- **master mode** = the **QTP** communicates with peripheral devices in I2C BUS (sensors, A/D, D/A, etc.) and it acts as a converter; naturally the operations to perform on the line are decided by the command unit that communicate with **QTP**, through the asynchronous serial line.

The physic protocol of the described modes is partially configured through the proper program of local setup, that allows to select the values described in TECHNICAL FEATURES chapter, by the simple use of only three keys. Further information about the communication between **QTP 16Big** and other units are reported in next two paragraphs.
CAN INTERFACE

QTP 16Big can have, as option, a complete CAN interface that supports the BasicCAN and PeliCAN 2.0B standards protocols. With this feature the user can afford and solve many problems as: high speed data transfer, long distance communication, autonomous errors management, multimaster and multislave networks support, etc.

The code used to order this option is: .CAN

ASYNCHRONOUS SERIAL LINE

The most diffused communication with the master unit is performed through an asynchronous serial line, that in default configuration, is electrically configured in RS 232 but using a proper indication in the order, it can be configured in:

- RS 422 -> .RS422 option
- RS 485 -> .RS485 option
- Current Loop -> .CLOOP option

The physical protocol of the serial line is completely configurable through a dedicated setup modality that let the user select the values listed in TECHNICAL FEATURES chapter, by the simple use of three keys. Finally the logic protocol can be point to point or master slave type, using the ninth bit technique; this latter, when used in conjunction with one of the options above described, allows the connection of many QTPs on a network and to communicate with terminals of the same or different type, easily and efficiently.

FIGURE 3: QTP 16Big COMPLETE VIEW
REAL TIME CLOCK

**QTP 16Big** can have, as option, a Real Time Clock backed by on board Lithium battery, that manages hours, minutes, seconds, day, month, year and week day. This device is manageable by the user with appropriate software commands which allow to set time and date, to read these data or to display them on display with a given position plus format and to manage a clock alarm. This option adds a complete time information, autonomously managed by **QTP 16Big**, and it makes available a serial real time clock to the external command unit. This unit should control the elapsed time, activate procedures on time based events, calculate production values in a time period, start or stop processes at fixed time of a day, etc.

The code used to order this option is: `.RTC`

RELAYS OUTPUTS

**QTP 16Big** can have, as option, three relays with a 5 Ampere contact. These are enabled or disabled through proper software commands that allows either a single output management or group management. The three normally open contacts of the relays are connected to a dedicated connector, quick release screw terminals type. By this way the connection of the relays output section is completely separated from the other circuitry.

As an example, this option could be necessary when an automatic door must be opened directly by the terminal placed at a short distance; in this case it will be necessary only a power actuator driven directly by one of the described relay.

The code used to order this option is: `.RELAY`

QTP 16BIG LIBRARY

For the **QTP 16Big** it is available a library that allow the user to decide the complete functionality of the operator panel. In this condition the **QTP** firmware is not the one described in this manual but it is developed by the user through comfortable high level programming languages. Anyway the language take advantage from the numerous functions described in the COMMANDS chapter, in fact they are grouped in the library that must be simply linked with the user firmware. This choice really simplify the **QTP** management and reduces the developing time. For detailed information about this possibility please refer to proper manual.
TECHNICAL FEATURES

GENERAL FEATURES

Resources:
- IP54 frontal
- Metallic container complete of mounting clamps
- 5 status LEDs driven by software
- Membrane keyboard with 16 keys, software reconfigurable
- Slot pocket for personalization label
- Buzzer for beep, keyclick or acoustic feedback
- Full duplex, asynchronous serial line, buffered in RS 232 or RS 422, RS 485. Current Loop (options)
- Synchronous I2C BUS serial line in master and/or slave modality
- CAN interface (option)
- Alphanumeric display in two different models
- Circuitry that regulates LCD display contrast
- Circuitry that enables LCD display backlight
- Real Time Clock backed by lithium battery (option)
- 3 relays digital outputs (option)
- Switching power supply

Displays:
- alphanumeric LCD 20x4 big size chars, LED backlight
- alphanumeric fluorescent 20x4 big size chars

CPU:
- 89C5115 or 89C51CC02 with 14.7456 MHz crystal
- Default: 89C5115

Memories:
- 16K FLASH EPROM
- 2K FLASH EPROM for Boot Loader
- 0,5K RAM
- 2K EEPROM
- up to 64K EEPROM (option)
- 240 byte backed SRAM (option)

Power on time: 50 msec

Timing resolution: 2,5 msec

Base EEPROM write time: 8 msec

Optional EEPROM write time: 5 msec

Keys autorepeat time: After 500 msec and then every 100 msec

Buzzer intermittent time: 500 msec

LEDs intermittent time: 500 msec
Messages shift time: 500 msec
RTC visualization time: 500 msec
User EEPROM bytes: 40
Messages number: 95, 914, 1733, 3371
  Default: 95
Max units on network: 256 with asynchronous line and Master-Slave 9 bits
  128 with synchronous I2C BUS line
Communication: Selectable between Normal, Master-Slave 9 bits, I2C BUS
  Default: Normal
Communication physic protocol in Normal, Master-Slave 9 bits:
  Baud rate: 1200, 2400, 4800, 9600, 19200, 38400
  Stop bit: 1 or 2
  Parity: none
  Bits x chr: 8, 9
  Slave Address: from 00H to FFH at step of 1
  Default: 19200 Baud, 1 Stop, No parity, 8 Bits,
           Slave Address = 80H
Communication physic protocol in I2C BUS:
  Bit rate: from 500 to 15000 bits/second
  Modality: Slave
  Slave Address: from 00H to FEH at step of 2
  Default: Slave Address = 80H
Receive buffer size: 40 characters
Transmit buffer size: 20 characters

PHYSICAL FEATURES
Size: DIN 96x192: 192 x 96 x 43 mm (W x H x D)
       204 x 96 x 86 mm (W x H x D) with clamps
  See outline dimension in APPENDIX C
Size of breaking for mount: 186 (min) x 90 (min) x 10 (max) mm (W x H x D)
  See outline dimension in APPENDIX C
Pixels size:
              LCD 20x4 Big: 0.9 x 1.1 mm (W x H)
              Fluorescent 20x4 Big: 0.6 x 0.8 mm (W x H)
  See dimension in APPENDIX D
Characters size:
              LCD 20x4 Big: 5 x 7 dots = 4.8 x 8.1 mm (W x H)
              Fluorescent 20x4 Big: 5 x 7 dots = 3.6 x 7.7 mm (W x H)
  See dimension in APPENDIX D
Viewing area size: LCD 20x4 Big: 118.8 x 38.4 mm (W x H)  
Fluorescent 20x4 Big:  99.6 x 37.6 mm (W x H)  
See dimension in APPENDIX D

Weight: 550 g max.

Mounting: Surface or front panel mounting, through provided clamps  
At sight on a bearing surface

Temperature range: From 0 to 50 °C

Relative humidity: 20% up to 90% (without condense)

Connectors: CN3: AMP MODU II, vertical, 4 pins, male, pitch 2.54  
CN4: quick release screw terminal, 2 pins, male, pitch 5  
CN5: D type connector, vertical, 9 pins, female  
CN6: quick release screw terminal, 3 pins, male, pitch 3.5  
CN7: quick release screw terminal, 6 pins, male, pitch 5  
J1: AMP MODU II, vertical, 2 pins, male, pitch 2.54

Figure 4: Rear view
ELECTRIC FEATURES

Power voltage:  
+10÷38 Vdc, 8÷24 Vac  
or +5 Vdc ± 5% (option)

Power consumption:  
see next table

<table>
<thead>
<tr>
<th>DISPLAY model</th>
<th>Consumption max. +5 Vdc</th>
<th>Consumption max. 10÷38 Vdc 8÷24 Vac</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD 20x4 alphanumeric Big backlight:</td>
<td>660 mA</td>
<td>4.6 W</td>
</tr>
<tr>
<td>QTP 16Big-C4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fluorescent 20x4 alphanumeric Big:</td>
<td>880 mA</td>
<td>6.2 W</td>
</tr>
<tr>
<td>QTP 16Big-F4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

FIGURE 5: CONSUMPTIONS TABLE

Output power supply voltage:  +5.0 Vdc
Current available on +5 Vdc output:  1000 mA - consumption max. +5 Vdc
RS 232 extravoltage protection:  ±15 KV
RS 422-485 line impedance:  60 Ω
RS 422-485 termination:  
line termination resistor:  120 Ω
pull-up resistor on positive:  3.3 KΩ
pull-down resistor on negative:  3.3 KΩ

CAN line impedance:  60 Ω
CAN termination circuit:  120 Ω ressitor, disconnectable
Pull up resistor on I2C BUS:  10 KΩ
Back up battery:  3 V Lithium; 180 mAh; CR2032 model
Back up current:  3.5 µA
Max current on relays:  5 A
Max voltage on relays:  30 Vdc

(*) The data are referenced to 20 C° environmental work temperature (for further information please refer to chapter POWER SUPPLY).

The table on figure 5 lists the QTP 16Big power consumption referred to the different display types that can be ordered; for the wide range power supply are described the required power, in place of the current, already corrected with efficiency factor of the on board power supply section.

To reduce consumptions of QTP 16Big with LCD display it is possible to order particular models without backlight: for further information and availability, please contact directly grifo®. Alternatively the QTP 16Big-C4 backlight can be disabled by software through a dedicated command: in this condition it achieves the minimum consumption of 50 mA, equal to about 0.3 W.
**INSTALLATION**

In this chapter there are the information for a right installation and correct use of the terminal **QTP 16Big**. In detail there are the locations and functions of each connector, of the user settable jumpers, of the battery and any other information concerning hardware configuration.

**Figure 6: Location of Jumpers, Connectors, Buzzer, Battery, etc.**
CONNECTIONS

QTP 16Big terminal has 6 connectors that can be linked to other devices or directly to the field, according to system requirements. Below are reported the pin outs, the meaning of the connected signals (including their directions) and some connection examples, that simplify and speed the installation phase. In addition the figures 4 and 6 show the connectors position on the board, to simplify their recognitions.

All the connectors are accessible from the back of the alluminum container, through a proper breaking in the rear side that allows comfortable insertion and deinsertion.

CN4 - POWER SUPPLY CONNECTOR

CN4 is a vertical, 2 pins, male, quick release screw terminal connector, with 5 mm pitch. On CN4 must be connected the single power supply voltage for the terminal that can be one out of three different types, as described by following figures

![Figure 7: CN4 - Power Supply Connector](image)

Signals description:

- **Vac** = I - AC power supply lines connected to on board switching section; these signals must be in the range 8÷24 Vac.
- **+Vdc pow** = I - DC power supply lines connected to on board switching section (+10÷+38 Vdc) or stabilized (+5 Vdc) voltage connected to on board logic, according to ordered configuration.
- **GND** = - Ground signal for DC power supply.

**NOTE** For further information about power supply configurations, please refer to paragraph POWER SUPPLY.
FIGURE 8: AC POWER SUPPLY 8÷24 VAC

FIGURE 9: DC POWER SUPPLY +10÷38 VDC

FIGURE 10: STABILIZED POWER SUPPLY +5 VDC (OPTION)
CN6 - CAN INTERFACE CONNECTOR

CN6 is a vertical, 3 pins, male, quick release screw terminal connector, with 3.5 mm pitch. Through CN6 must be connected the CAN serial communication line by following the standard rules defined by the same protocol. Signals placement has been designed to reduce interferences and to obtain a fast and comfortable node connection on the field CAN bus. The connector is available only when the .CAN option has been ordered.

Signals description:

- **CANH** = I/O - Differential line high for CAN interface.
- **CANL** = I/O - Differential line low for CAN interface.
- **GND** = - Ground signal.

**Figure 11: CN6 - CAN interface connector**

**Figure 12: CAN line connection**
Please remind that a CAN network must have a line impedance of 60 Ω and for this reason two termination resistors (120 Ω) must be placed at its extrem, respectively near the units that are at the greatest distance. On QTP 16Big the terminating circuitry is already installed: it can be connected or not through specific jumper, as explained later.

When the system to link on the CAN line have very different potentials, it is possible to connect also the grounds of the same systems, that is pin 1 of CN6. In this way any possible problems of communication and/or incorrect working, are solved.
CN3 - I2C BUS LINE CONNECTOR

CN3 is a vertical, 4 pins, male, AMP MODU II connector, with 2.54 mm pitch. Through CN3 can be connected the synchronous communication line in I2C BUS. The signals connected respect the international normatives defined by this standard of communication and include also the power supply voltage generated on board, that can be used to supply power at external devices and/or systems. On the other hand the signals placement has been designed to reduce interferences and it is the same one available on great part of grifo® cards, to speed up the connection of different units.

The female connector for CN3 is directly available between grifo® accessories, and it can be ordered by using the codes CKS.AMP4 or AMP4.Cable, as described in APPENDIX E of the manual.

Signals description:

SDA = I/O - Data signal for I2C BUS communication.
SCL = I/O - Clock signal for I2C BUS communication.
+5 Vdc = O - +5 Vdc power supply signal.
GND = - Ground signal.

A complete description of I2C BUS communication is reported in next paragraphs I2C BUS COMMUNICATION and COMMAND FOR I2C BUS COMMUNICATION AS MASTER. The following figures show a connection example diagram with a generic I2C BUS master unit, both in point to point and network mode:

Figure 14: CN3 - I2C BUS line connector

Figure 15: Connection example for I2C BUS point to point communication
Figure 16: Connection example for I2C BUS network communication

Please remind that in a I2C BUS network must be connected two pull up resistors at the net extremities, respectively near the master unit and the slave unit at the greatest distance from the master. 

On QTP 16Big these resistors are always present in default configuration and they have the value described in ELECTRIC FEATURES paragraph. The user must select or configure the I2C BUS devices to connect, by taking care of this feature. In detail on QTP 16Big the described resistors must be removed on the units that are not at the line extremities, as shown in previous figure, on slaves 1 and 2.

For further information please refer to document “THE I2C-BUS SPECIFICATIONS”, from PHILIPS semiconductors.
CN7 - RELAYS OUTPUTS CONNECTOR

CN7 is a 6 pins, vertical, quick release screw terminal connector with 5 mm pitch. This connector allows to connect the normally open contacts and common signals of the 3 relays outputs available on QTP 16Big, ordered with .RELAY option. Please remind that maximum (resistive) load for each line is 5 A and maximum voltage is 30 Vdc.

These lines are driven by proper commands and among the provided functionalities it is remarkable the alarm clock based on optional on board real time clock.

Signals description:

- **NO OUT n** = O - Normally open contact of relay n.
- **COMMON n** = - Common contact of relay n.

As described in these figures, each relay output is provided of its own common terminal. This allows connection of external loads even when they are supplied by different sources, making the cabling of the whole system very easier.
CN5 - SERIAL LINE CONNECTOR

CN5 is a D type, 9 pins, female, vertical connector. On CN5 are available all the signals of the asynchronous serial line, buffered with one of the diffused electric standards RS 232, RS 422, RS 485 or Current Loop, that allows the complete management of the panel. Placing of the signals has been designed to reduce interferences and electrical noises and to simplify connections with other systems, while the electric protocols follow the CCITT directives of the used standard.

**Figure 19: CN5 - Serial line connector**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>RX RS232</td>
<td>I</td>
<td>Receive data for RS 232.</td>
</tr>
<tr>
<td>3</td>
<td>TX RS232</td>
<td>O</td>
<td>Transmit data for RS 232.</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td></td>
<td>Ground signal.</td>
</tr>
</tbody>
</table>

**RS 232 serial line:**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RX- RS422</td>
<td>I</td>
<td>Negative receive data for RS 422.</td>
</tr>
<tr>
<td>2</td>
<td>RX+ RS422</td>
<td>I</td>
<td>Positive receive data for RS 422.</td>
</tr>
<tr>
<td>3</td>
<td>TX- RS422</td>
<td>O</td>
<td>Negative transmit data for RS 422.</td>
</tr>
<tr>
<td>4</td>
<td>TX+ RS422</td>
<td>O</td>
<td>Positive transmit data for RS 422.</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td></td>
<td>Ground signal.</td>
</tr>
</tbody>
</table>

**RS 422 serial line:**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RXTX- RS485</td>
<td>I/O</td>
<td>Negative receive and transmit data for RS 485.</td>
</tr>
<tr>
<td>2</td>
<td>RXTX+ RS485</td>
<td>I/O</td>
<td>Positive receive and transmit data for RS 485.</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td></td>
<td>Ground signal.</td>
</tr>
</tbody>
</table>

**RS 485 serial line:**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RXTX- RS485</td>
<td>I/O</td>
<td>Negative receive and transmit data for RS 485.</td>
</tr>
<tr>
<td>2</td>
<td>RXTX+ RS485</td>
<td>I/O</td>
<td>Positive receive and transmit data for RS 485.</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td></td>
<td>Ground signal.</td>
</tr>
</tbody>
</table>

**Current Loop serial line:**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>RX- C.L.</td>
<td>I</td>
<td>Negative receive data for Current Loop.</td>
</tr>
<tr>
<td>8</td>
<td>RX+ C.L.</td>
<td>I</td>
<td>Positive receive data for Current Loop.</td>
</tr>
<tr>
<td>7</td>
<td>TX- C.L.</td>
<td>O</td>
<td>Negative transmit data for Current Loop.</td>
</tr>
<tr>
<td>6</td>
<td>TX+ C.L.</td>
<td>O</td>
<td>Positive transmit data for Current Loop.</td>
</tr>
</tbody>
</table>

**QTP 16Big Rel. 5.00**
**FIGURE 20: RS 232 POINT TO POINT CONNECTION EXAMPLE**

![RS 232 Connection Diagram](image)

**FIGURE 21: RS 422 POINT TO POINT CONNECTION EXAMPLE**

![RS 422 Connection Diagram](image)

**FIGURE 22: RS 485 POINT TO POINT CONNECTION EXAMPLE**

![RS 485 Connection Diagram](image)
Please remark that in a RS 485 network two forcing resistors must be connected across the net and two termination resistors (120 Ω) must be placed at its extremis, respectively near the Master unit and the Slave unit at the greatest distance from the Master.

Forcing and terminating circuitry is installed on QTP 16Big board and it can be enabled or disabled through specific jumpers, as explained later.

About termination resistor of Master unit, connect it only if not already present (for example many RS 232-RS 485 converters already have it inside).

For further information please refer to TEXAS INSTRUMENTS Data-Book, "RS 422 and RS 485 Interface Circuits", the introduction about RS 422-485.
**Figure 24: Current Loop 4 Wires Point to Point Connection Example**

**Figure 25: Current Loop 2 Wires Point to Point Connection Example**
Possible Current Loop connections are two: 2 wires and 4 wires. These connections are shown in figures 24+26 where it is possible to see the voltage for VCL and the resistances for current limitation (R). The supply voltage varies in compliance with the number of connected devices and voltage drop on the connection cable.

The choice of the values for these components must be done considering that:
- circulation of a 20 mA current must be guaranteed;
- potential drop on each transmitter is about 2.35 V with a 20 mA current;
- potential drop on each receiver is about 2.52 V with a 20 mA current;
- in case of shortcircuit each transmitter must dissipate at most 125 mW;
- in case of shortcircuit each receiver must dissipate at most 90 mW.

For further info please refer to HEWLETT-PACKARD Data Book, (HCPL 4100 and 4200 devices).
J1- BOOT LOADER ACTIVATION CONNECTOR

J1 is a vertical, 2 pins, male, AMP MODU II connector, with 2.54 mm pitch. This connector enables the DEBUG modality of QTP 16Big that allows the user to reprogram the internal FLASH EPROM. Normally, this operation is necessary only when the user must develop its own management program, for example in conjunction with the library firmware .LIB.

Signals description:

/EBL = I/O - Boot Loader enable signal.
GND = Ground signal.

The J1 connector can be used also as a simple 2 pins jumper (as described on figure 30) in fact to start Boot Loader the /EBL signal must be enabled, that means connect it to ground. In this specific condition the Boot Loader is enabled by simply inserting a jumper on the connector. Viceversa, when the activation must be remoted, the female connector for J1 must be used: it is directly available between grifo® accessories, and it can be ordered by using the codes CKS.AMP2 or AMP2.Cable, as described in APPENDIX E of the manual.
FIGURE 28: COMPONENTS MAP SOLDER SIDE

FIGURE 29: COMPONENTS MAP COMPONENTS SIDE
**JUMPERS**

On **QTP 16Big** there are eight jumpers for card configuration and by connecting them, the user can perform some selections that regards the working conditions of the card. Here below there is the jumpers list and relative functions in the possible connection modalities:

<table>
<thead>
<tr>
<th>JUMPER</th>
<th>CONNECTION</th>
<th>PURPOSE</th>
<th>DEF.</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1</td>
<td>not connected</td>
<td>It selects the RUN modality at power on by executing the program saved on FLASH (used only for .LIB version).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td>It selects the DEBUG modality at power on by executing the Boot Loader (sed only for .LIB version).</td>
<td>*</td>
</tr>
<tr>
<td>J3 , J4</td>
<td>not connected</td>
<td>Do not connect termination and forcing circuitry to RS 422, RS 485 serial line.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td>Connect termination and forcing circuitry to RS 422, RS 485 serial line.</td>
<td></td>
</tr>
<tr>
<td>J5</td>
<td>not connected</td>
<td>Does not connect 120 Ω termination resistor to CAN line.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td>Connects 120 Ω line termination resistor to CAN line.</td>
<td></td>
</tr>
<tr>
<td>J8</td>
<td>position 1-2</td>
<td>Configures serial line for RS 485 standard electric protocol (2 wires half duplex).</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>Configures serial line for RS 422 standard electric protocol (4 wires half duplex or full duplex).</td>
<td></td>
</tr>
<tr>
<td>J9</td>
<td>position 1-2</td>
<td>Write protection of optional EEPROM not enabled.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>Write protection of optional EEPROM enabled.</td>
<td></td>
</tr>
<tr>
<td>J10</td>
<td>not connected</td>
<td>Management of RL2 relay not enabled.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td>Management of RL2 relay enabled.</td>
<td></td>
</tr>
<tr>
<td>J12</td>
<td>not connected</td>
<td>On board battery BT1 not connected to back up circuitry.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td>On board battery BT1 connected to back up circuitry.</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 30: JUMPERS table**

To recognize the valid connections and locations of these jumpers, please refer to the board printed diagram (serigraph) or to figure 6 of this manual, where the pins numeration is listed.

In previous table the "*" denotes the default connection, or on the other hand the connection set up at the end of testing phase, that is the configuration the user receives. The user can check the default configuration of all the modifiable features, also in the APPENDIX E at the end of the manual. Further information about purpose of the **QTP 16Big** jumpers are reported in the following paragraphs, that describe the sections where the same jumpers are used.
OPTIONAL EEPROM PROTECTION

With jumper J9 the user can protect the optional EEPROM of QTP 16Big towards the write operations. In detail with jumper in position 1-2 the EEPROM is not protected and it can be written through proper commands, viceversa in position 2-3 the device can only be read. Please remind that the first 95 messages and all the other data saved in base EEPROM (see paragraph DATA STORED ON EEPROM), are not interested by jumper J9 configuration. The most important purpose of this jumper is to avoid unwanted writing and/or modifications of the numerous messages saved inside optional EEPROM, especially when they are stable. Normally the user must perform the following operations, during the installation phase:

- disable the protection by connecting J9 in position 1-2 (default configuration);

- save all the invariable messages, by taking advantage of specific commands for messages provided of number greater than 95, through a dedicated program (i.e. QTP EDIT), or a proper modality of management program;

- enable the protection by connecting J9 in position 2-3;

- at this point the management program can use the saved and protected messages through the read commands, only.

BACK UP

When QTP 16Big is ordered with the optional real time clock (.RTC), it is provided of a lithium battery that keeps the time and the content of SRAM even when power supply is off. The user can connect or not this battery to back up circuitry, by acting on dedicated jumper J12, as described in figure 30. The card is supplied with the jumper connected to preserve the clock counting and the SRAM content in each operating condition. Whenever the QTP 16Big is not used for a long time, or the application doesn't need the back up circuit, it is suggested to prevent the battery discharge by removing the jumper J12. Obviously if the J12 connection must be changed, then the card must be extracted from the metallic container: a simple pressure on QTP 16Big connectors, or on the printed circuit reachable from rear container window, is sufficient. When on the front panel there are two black screws, they must be previously unscrewed (for details see APPENDIX C).
SERIAL LINE CONFIGURATION

Serial line of QTP 16Big can be buffered in RS 232, RS 422, RS 485 or Current Loop. By software the serial line can be programmed to operate with all the standard physical protocols, in fact the bits per character, parity, stop bits and baud rates can be decided by an opportune local setup procedure (see homonymous paragraph). Through the local setup can be selected also the logic protocol of communication, among the available modalities.

By hardware can be selected which one of the electric standards is used, through jumpers connection (as described in the previous table) and drivers installation. Some devices needed for RS 422, RS 485 and Current Loop configurations are not mounted on the board in standard configuration; this is why each fist non-standard (non RS 232) serial configuration must be always performed by grifo® technicians. At this point the user can change autonomously the configuration following the below information:

- SERIAL LINE IN RS 232 (default configuration)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IC11</td>
<td>= driver MAX 202</td>
</tr>
<tr>
<td>IC12</td>
<td>= no device</td>
</tr>
<tr>
<td>J8</td>
<td>= indifferent</td>
</tr>
<tr>
<td>IC13</td>
<td>= no device</td>
</tr>
<tr>
<td>J3, J4</td>
<td>= not connected</td>
</tr>
<tr>
<td>IC14</td>
<td>= no device</td>
</tr>
<tr>
<td>IC15</td>
<td>= no device</td>
</tr>
</tbody>
</table>

- SERIAL LINE IN CURRENT LOOP (option .CLOOP)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IC11</td>
<td>= no device</td>
</tr>
<tr>
<td>IC12</td>
<td>= no device</td>
</tr>
<tr>
<td>J8</td>
<td>= indifferent</td>
</tr>
<tr>
<td>IC13</td>
<td>= no device</td>
</tr>
<tr>
<td>J3, J4</td>
<td>= not connected</td>
</tr>
<tr>
<td>IC14</td>
<td>= driver HP 4200</td>
</tr>
<tr>
<td>IC15</td>
<td>= driver HP 4100</td>
</tr>
</tbody>
</table>

Please remark that Current Loop serial interface is passive, so it must be connected an active current loop serial line, that is a line provided with its own power supply, like described in figures 24÷26. Current Loop interface can be employed to make both point to point and multi points connections through a 2 wires or a 4 wires connection.

- SERIAL LINE IN RS 422 (option .RS422)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IC11</td>
<td>= no device</td>
</tr>
<tr>
<td>IC12</td>
<td>= driver SN 75176 or MAX 483</td>
</tr>
<tr>
<td>J8</td>
<td>= position 2-3</td>
</tr>
<tr>
<td>IC13</td>
<td>= driver SN 75176 or MAX 483</td>
</tr>
<tr>
<td>J3, J4</td>
<td>= (*)</td>
</tr>
<tr>
<td>IC14</td>
<td>= no device</td>
</tr>
<tr>
<td>IC15</td>
<td>= no device</td>
</tr>
</tbody>
</table>

RS 422 electric protocol can be used to make 4 wires, full duplex, connections either in multi points or point to point systems.

Transmitter abilitation, essential in networks connections, is managed directly by QTP 16Big firmware by selecting the Master-Slave 9 bits logic protocol.
FIGURE 31: LOCATIONS OF DRIVERS FOR SERIAL COMMUNICATION

Serial line in RS 232

Serial line in Current Loop

Serial line in RS 422

Serial line in RS 485
- SERIAL LINE IN RS 485 (option .RS485)

J8 = position 1-2
J3 , J4 = (*)

IC11 = no device
IC12 = driver SN 75176 or MAX 483
IC13 = no device
IC14 = no device
IC15 = no device

In this modality the signals to use are pins 1 and 2 of connector CN5, that become transmission or reception lines according to the status defined by firmware; the last must be configured with logic protocol Master-Slave 9 bits. The RS 485 electric protocol can be used to make 2 wires half duplex connections both in multi points networks and point to point connection.

(*) When the RS 422 or RS 485 serial line are used, it is possible to connect the termination and forcing circuit on the line, by using J3 and J4 jumpers. This circuit must be always connected in case of point to point connections, while in case of multi points connections it must be connected only in the farest boards, that is on the edges of the communication line.

During a power on, the RS 485 driver is in reception and RS 422 transmission driver is disabled, to avoid conflicts on the communication line.

For further information about serial communication please refer to the connection examples of figures 20+26.

**RL2 RELAY ENABLE**

The functionality of digital output NO OUT2, connected to RL2 relay, depends on jumper J10 configuration. This last enables or not the output, as described in figure 30, and avoids unwanted settings of the same, when the QTP 16Big has not been arranged. Infact the RL2 relay is driven by the on board clock device (.RTC option) and this component must be programmed at least one time, to ensure its right functionalities.

Moreover the RL2 relay can work as a generic user output or as alarm output enabled autonomously by the alarm clock. In both the conditions, the user must proceed as below described:

- disable the output management by not connecting J10 (default configuration);
- select the RL2 functionality according to user requirements, through the proper option in local setup;
- define the output status by using the commands that manage relays outputs or those for the clock and clock alarm, according to selected functionality;
- enable the output management by connecting J10;
- at this point the QTP 16Big is completely arranged to correctly drive the RL2 output.

If jumper J12 is connected, and consequently the real time clock is connected to back up circuitry, then the described steps must be performed only the first time the panel is used, that is after a purchase or possible reparations.

To locate J10 position, please refer to figure 6.
Figure 32: QTP 16Big-C4

Figure 33: QTP 16Big-F4
POWER SUPPLY

**QTP 16Big** terminal is provided with a power supply section that solves in an efficient and comfortable way the problem to supply the board, in any situation. It generates energy for all sections of the board: control logic, display, backlight, keyboard, LEDs, serial interfaces, CAN interface, I2C BUS line, real time clock, buzzer and relays. Here follow the voltages required from **QTP** according to card configuration together with the relative right connection:

**Default version:** This configuration includes a switching power supply that requires 10÷38 Vdc or 8÷24 Vac provided through CN4 (**polarity must be respected** in case of DC supply). This allows to supply the terminal using standard industrial and commercial power sources like transformers, batteries, solar cells, etc. A comfortable and inexpensive solution for default version power supply can be the **EXPS-1** product that can be directly connected to the terminal starting from mains. Please remind that on board switching section is provided with single diode rectifier, so in case of DC supply, all ground signals of the terminal (GND) are at the same potential. When a single AC source is used to supply different units (both some **QTP 16Big** or other cards provided of supply section with single diode rectifier), please ensure that the two phases of AC voltage must be connected at the same input pins of power supply connector. Whenever this rule is not satisfied dangerous malfunctions or damages can rise up on all the connected devices. For example, if we call Phase1 and Phase2 the two signals of the AC voltage, then Phase1 must be always connected to positive inputs (Vac, +Vdc pow) and Phase2 must be connected to negative input (Vac, GND). Complete information and details can be found on paragraph CN4- POWER SUPPLY CONNECTOR. This is the default version, normally delivered without further specifications, in the order.

**.5Vdc or .ALIM version:** This configuration is not provided of any power supply section, so a +5 Vdc ±5% stabilized supply voltage must be provided by an external source, through CN4 connector (**polarity must be respected** also in this case). This allows to provide energy to the terminal through laboratory power supply, other cards, etc. This version is a particular OEM configuration only, to directly agree upon **grifo®**.

Selection of power supply section must be performed during the order phase, in fact it involves a different hardware configuration that must be made by **grifo®** technicians. The **QTP 16Big** is always provided with a **TransZorb™** protection circuit to avoid damages from incorrect voltages and/or break down of power supply section. It is also provided with a distributed filtering circuitry that saves the terminal from disturbs or noise from the field, improving the overall system performances. For further information please refer to paragraph ELECTRIC FEATURES.
CANA N INTERFACE CONNECTION

Jumper J5 connects or does not connect termination resistor of CAN line, as described on figure 30. The CAN BUS must be a differential line with 60 Ω of impedance so termination resistors must be connected to obtain this impedance value. In detail, this connection must be always made in case of point to point communications, while in multi points communications it must be connected only in the cards at the greatest distance, that is at the ends of the CAN line (please see example of figure 13).

The right CAN termination contributes considerably to obtain a correct communication; in fact the **QTP 16Big** on board interface can suppress transients and avoids radio frequency and electromagnetic noises, only when connection to the field is correctly made.

CAN line is not galvanically isolated (as described in previous paragraph POWER SUPPLY) from on board generated supply voltage. Ground of CAN line is connected to on board GND and it is available on a pin of CN6 connector. This latter can be used to equilibrate difference of potentials amongst several CAN systems, but also to shield physical connection, when shielded cable is used for CAN line, to obtain the greatest protection against external noise.
SOFTWARE DESCRIPTION

As already stated QTP 16Big is a complete video terminal. It shows on the display any characters received from communication line, except the commands that are recognized and executed, and it transmits back, on the same line, the possible results of the executed commands and the codes of keys pressed. In other words it acts a slave dumb terminal controlled by an external command unit that can be placed even at a long distance. These operations are automatically performed by the on board firmware that is programmed and executed from QTP 16Big CPU. The on board firmware manages also a local setup which allows the user to define some working conditions by using the display and the keyboard of QTP 16Big. This chapter describes the main features of QTP 16Big software functionalities, while the following one reports a detailed description of the recognized command sequences, that can be used to benefit of all the potentiality of the terminal. In correspondence of the first order, on the received grifo® CD, are supplied many complete and useful demo programs either in executable and source format; these can be used as received with no modifications, for a first test of the product and then changed, or partially used, to develop the user application program.

LOCAL SET UP

Thanks to a proper local setup mode the user can select some parameters of communication protocol, define some working conditions and restore the base EEPROM content. This mode can be easily and intuitively used, thanks to the on board display and 3 keys of QTP 16Big.

In detail the user must:

a) Press the keys F1 and F2, simultaneously power on the QTP 16Big and then maintain the keys pressed for at least half of a second.

b) At this point setup mode is entered, on the display appears the “Local Setup V.x.y” string and with keys F4 and F1 the current configuration parameters, and its current values, shall be changed as below described:

c) Press the key F4 to change current menu, recognized by the following messages:
   "COMMUNIC." to change the communication type
   "BAUD RATE" to change the communication baud rate
   "STOP BIT" to change the stop bit number
   "KEY-CLICK" to change the keyclick mode
   "SLAVE ADD." first digit of identification name in hexadecimal
   "SLAVE ADD." second digit of identification name in hexadecimal
   "EE DATA" initializes data in EEPROM
   "RL2 FUNCT." to change function of RL2 relay output
   "SAVE and EXIT" to exit from setup mode

d) Press the key F1 to change current value of displayed menu:
   COMMUNICATION: Norm., I2C, M.S.9 that are the 3 communication mode (def.=Norm.)
   BAUD RATE: 38400, 19200, 9600, 4800, 2400 or 1200 baud (def.=19200)
   STOP BIT: 1 or 2 with Normal protocol (def.=1)
             1 with Master-Slave 9 bits protocol
<table>
<thead>
<tr>
<th>Feature</th>
<th>Options</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY-CLICK:</td>
<td>ON or OFF</td>
<td>(def. = ON)</td>
</tr>
<tr>
<td>SLAVE ADD.:</td>
<td>Changes the digit enclosed in &quot;&lt;&gt;&quot; from 0 to F</td>
<td>(def. = 80H)</td>
</tr>
<tr>
<td>EE DATA:</td>
<td>NOINI or INIT</td>
<td>(def. = NOINI)</td>
</tr>
<tr>
<td>RL2 FUNCT.:</td>
<td>USER or ALARM</td>
<td>(def. = USER)</td>
</tr>
<tr>
<td>SAVE and EXIT:</td>
<td>exits setup and configures QTP 16Big with selected parameters</td>
<td></td>
</tr>
</tbody>
</table>

e) Once the necessary values have been set by using the modalities described in points c and d, select the **SAVE and EXIT** menu and press the ➔ F1 key to confirm.

Once exited from setup mode, the selected parameters are saved on EEPROM and they are maintained until another local setup is executed; immediately after the terminal starts its normal functionality. The *default* values (before reported between round brackets) are those set at the end of testing phase, that is the configuration the user receives.

Available options for menus BAUD RATE and STOP BIT define the physical communication protocol that has other two parameters unchangeable and set to no parity and 8 bits per character or 9 bits when Master-Slave 9 bit is selected. Options of remaining menus are described in the following paragraphs.

**NOTE:** Please remind that local setup mode can be entered only during power on, when previously described conditions are recognized in fact if external keys are pressed at the same time during normal operation then setup mode will not start.

The local setup is normally executed only one time after the first installation, from the customer or installer, that configures the QTP 16Big according with requirements of the developed application.

So it regards expert staff but not the final user that handle it as a simple, ready operator interface unit.

**COMMUNICATION BUFFERS**

QTP 16Big is provided of two communication buffers that simplify the management and increase its flexibility, in fact they reduce the waiting time of the connected command unit.

The first is a receive buffer: it is **40 bytes** long, it memorizes each character received from command unit and then it is examined at the end of the currently executed operation. Naturally when commands that requires a long execution time (delete commands, EEPROMs management commands, messages shift, etc.) are continuously received, the buffer can become full and it will overflow. When overflow occurs the first locations of the buffer are overwritten by each next received characters, and they are definitely lost. The command unit must stop the transmission until the QTP 16Big has emptied the receive buffer and it is still ready to receive other data. In practice the user must insert suitable delays in communication, experimentally calibrated, to avoid overflow of the buffer.

The second is a transmit buffer: it is **20 bytes** long, it memorizes each character that must be sent to command unit and it is filled with the keys pressed codes and the executed commands response. When Normal communication is selected the transmit buffer is not used in fact data are immediately transmitted, vice versa when I2C BUS or Master-Slave 9 bits communications are used the data remain in transmit buffer until the command unit requires them. If the command unit doesn't receive data from QTP 16Big, this buffer become full. When this filling occurs all following data are no more saved in the transmit buffer, and these are definitely lost. So the command unit must manage data reception from QTP 16Big at least in two situations: before to send commands provided of responses (to empty the buffer for the same response) and periodically (to get the possible keys pressed).
DATA STORED ON EEPROM

The base EEPROM of QTP 16Big stores a set of data used and/or changed through the specific commands and local setup. The choice of EEPROM memory type has been performed to obtain the best warranties on data validity and endurance, naturally even when power supply is not available. The detailed description on each one of the data saved on EEPROM is reported in the following chapter, in the paragraphs relative to commands that directly use them.

With menu EE DATA of local setup the user can select to leave unchanged these data (NOINI option) or to set them at their default values (INIT option) that is the configuration received after an order or a reparation. In details, by selecting the INIT option, the base EEPROM data will have the following values:

- presence byte → 255 (FFH)
- keys codes → those described in table of figure 36
- power on visualization → none
- patterns of user defineable characters → 255 (FFH)
- messages → 255 (FFH)
- user EEPROM bytes → 255 (FFH)

Once exit from local setup a string is shown on the display together with a scrolling bar of * (asterisk) that inform about the status progress of the operation. The displayed * are 10 and the execution time of the described initialization phase is 20 seconds approximately.

Please remind that the INIT option will initialize only the base EEPROM, while the optional EEPROM maintain its original contents. With this option of all the available messages, only the first 95 are deleted.

The user must be very careful with EEPROM initialization, in fact all data previously saved are definitly lost.

KEYBOARD ACQUISITION

When QTP 16Big recognizes a key pressure, it transmits the relative code on communication line. This happens immediately when Normal communication is selected, while in case of I2C BUS or Master-Slave 9 bits communications, the code is saved in the transmission buffer and then it is sent only upon reception of specific request from command unit, by using the rules described in the following paragraphs.

Moreover an auto repeat function of the stroked key is implemented so when QTP 16Big recognizes the pressure on a key, for a time greater than 0.5 sec. it will start the transmission of its code about each 0.1 sec. and it lasts until that key is released.

If the keyclick function is enabled when the code of the pressed key is transmitted, the on board buzzer also generates a loud beep that sonorously signalize the event to the user. Whenever the buzzer is already enabled or intermittent, then the keyclick disables it for a little time period, to ensure the acoustic event recognition in any circumstance.

When two or more keys are contemporaneously pressed it is transmitted only the code of the key with higher number, or on the other end, the key with number 15 (+.- 0) has the highest priority while the key number 0 (↑ F3) has the lower priority.

Another feature provided by QTP 16Big is the complete reconfiguration of the key codes performed by user application program; in other words it is possible to change the code returned when a key is pressed and even disable the key.
KEYS CODES

Below there is a figure that shows the keys numbers and locations and another figure with the default codes, that QTP 16Big terminal sends on communication line, when a key is pressed. As for the command sequences the code are shown in decimal, hexadecimal and ASCII mnemonic format, through the standard ASCII symbol table:

![Figure 35: Keys numbers and location](image)

The keys numbers on figure 35 are unchangeable and they are necessary for the command KEY CODE RECONFIGURATION, described in following chapter, to univocally identify all the keys. On the other end, the keys codes are programmable and they can be used by command unit to recognizes the keys pressed. The table of figure 36 lists the codes in default configuration, that is the one the user receive after an order and reparation, or after an EEPROM initialization. In addition the user can comfortably reconfigure the keys codes: this features really simplifies the management software development in fact the command unit can change the codes according with his requirements and it can also disable the keys.
Figure 36: Default keys codes

Characters visualization on display

QTP 16Big shows on its display all the received characters having a code included in the range 0÷255 (00÷FF Hex) including the one that identifies a command sequence (27 = 1BH), as described later. The character is visualized on the current cursor position and this latter will go to the next position; if it is placed on the last character of the display (right down corner), it will be placed on Home position (left up corner).

The correspondence between codes and displayed characters is defined by the following rules:

<table>
<thead>
<tr>
<th>KEY NUMBER</th>
<th>SERIGRAPHY</th>
<th>CODE</th>
<th>HEX CODE</th>
<th>MNEMONIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>↑</td>
<td>F3</td>
<td>62</td>
<td>3E</td>
</tr>
<tr>
<td>1</td>
<td>TUV</td>
<td>8</td>
<td>56</td>
<td>38</td>
</tr>
<tr>
<td>2</td>
<td>GHI</td>
<td>4</td>
<td>52</td>
<td>34</td>
</tr>
<tr>
<td>3</td>
<td>→</td>
<td>F1</td>
<td>58</td>
<td>3A</td>
</tr>
<tr>
<td>4</td>
<td>↓</td>
<td>F4</td>
<td>63</td>
<td>3F</td>
</tr>
<tr>
<td>5</td>
<td>PQRS</td>
<td>7</td>
<td>55</td>
<td>37</td>
</tr>
<tr>
<td>6</td>
<td>DEF</td>
<td>3</td>
<td>51</td>
<td>33</td>
</tr>
<tr>
<td>7</td>
<td>←</td>
<td>F2</td>
<td>59</td>
<td>3B</td>
</tr>
<tr>
<td>8</td>
<td>ENTER #</td>
<td>35</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>MNO</td>
<td>6</td>
<td>54</td>
<td>36</td>
</tr>
<tr>
<td>10</td>
<td>ABC</td>
<td>2</td>
<td>50</td>
<td>32</td>
</tr>
<tr>
<td>11</td>
<td>ESC</td>
<td>*</td>
<td>42</td>
<td>2A</td>
</tr>
<tr>
<td>12</td>
<td>WXYZ</td>
<td>9</td>
<td>57</td>
<td>39</td>
</tr>
<tr>
<td>13</td>
<td>JKL</td>
<td>5</td>
<td>53</td>
<td>35</td>
</tr>
<tr>
<td>14</td>
<td>FUNZ</td>
<td>1</td>
<td>49</td>
<td>31</td>
</tr>
<tr>
<td>15</td>
<td>+-.</td>
<td>0</td>
<td>48</td>
<td>30</td>
</tr>
</tbody>
</table>

To allow visualization of special characters, that have same codes of the one character commands, a specific command has been provided that selects the operating mode of QTP 16Big among the two available:

`command` the characters are not displayed and the relative commands are executed;
`representation` the characters are always displayed.
After a power on it is automatically selected the command mode to make immediately utilizable every functionalities. The commands composed by a sequence of two or more characters, that always start with ESC = 27 = 1BH, are anyhow interpreted and executed independently from the selected operating mode.

Every models of QTP 16Big has 8 user characters that can be defined and/or stored according to application requirements, and then shown on the display, as explained in the further paragraph COMMANDS FOR USER CHARACTERS.

About special characters please refer to APPENDIX B and remind that it is possible to get also different display models, provided of different special characters, but everything must be directly prearranged with grifo®.

**CONTRAST REGULATION**

On QTP 16Big board there is a circuit that defines the contrast on LCD display. This circuit is set by grifo® to obtain the best visibility in each working conditions and normally the user must not change its setting. In case of specific requirements, as external light very low or very high, the user can change the regulation by performing little modifications and checking if the visibility is improved.

The regulation is performed as described in the following steps:

- Mantain the key F4 pressed for at least 3 seconds to enter the contrast regulation modality. During the 3 seconds the code of the pressed entry key is anyway returned to command unit with the traditional timing of keyboard.
- At this point press the keys F3 and F4 to decrease and increase the contrast, without saving it. The down arrow is voluntarily associated to contrast increase to ensure the display visibility, even when the user mantain pressed the regulation entry key. To obtain a mnemonic function of the used arrows keys, they can be associated to text brightness, that is:
  - ↓ F4 -> reduce text brightness (= increase contrast)
  - ↑ F3 -> raise text brightness (= decrease contrast)
- When a different key is pressed or no keys are pressed for 2.5 seconds, the regulation modality is terminated and the last contrast set is saved.
- In the contrast regulation modality, in correspondence of every variation the codes of the pressed key is not transmitted and it is produced a keyclick with a shorter duration time, when enabled. This shorter keyclick period helps the user to recognize both the entry and the permanency of contrast regulation modality.

Once the contrast has been regulated and saved, it will be maintained until another regulation will be executed, and it is automatically set up after every power off and power on of QTP 16Big.

**NOTE**  The contrast regulation modality is not supported during execution of local setup; furthermore the regulation is possible only with LCD display, installed on QTP 16Big-C4. When QTP 16Big-F4, with fluorescent display, is used the contrast regulation process still works but it has no effects on visibility. The user can change the VFD display visibility only by using the command FLUORESCENT DISPLAY BRIGHTNESS SETTING.
COMMUNICATION MODALITIES

QTP 16Big supports three different serial communication modalities:

**Norm.** Normal communication uses the asynchronous serial line on CN5 and it supports 8 bits per character, no parity plus stop bit and baud rate selected by user, through local setup. This communication mode is suitable for point to point connections in RS 232, RS 422 and Current Loop.

For detailed information about this modality please read proper paragraph NORMAL COMMUNICATION.

**I2C** I2C BUS communication uses the synchronous serial line on CN3 and it supports a bit rate from 500 to 15000 bits per second, as slave (either receive or transmit), with a 7 bits Slave Address selected by user, through local setup. This communication mode is suitable for point to point or network connections.

For detailed information about this modality please read proper paragraph I2C BUS COMMUNICATION.

**M.S.9** Master-Slave 9 bits communication uses the asynchronous serial line on CN5 and it supports 9 bits per character, no parity, one stop bit plus baud rate selected by user, through local setup. This communication mode is suitable for point to point connections (with all electric protocols) or network (with RS 485, RS 422 and Current Loop electric protocols).

For detailed information about this modality please read proper paragraph MASTER-SLAVE 9 BITS COMMUNICATION.

Local setup allows to select communication modality, as described in the specific paragraph, while electric protocol must be defined when the terminal is ordered or changed as described in SERIAL LINE CONFIGURATION paragraph.

MASTER-SLAVE 9 BITS COMMUNICATION

The Master-Slave 9 bits mode uses a particular communication technique; in addition to the 8 data bits also a ninth bit is managed and it discriminates between a call coming from the "master" device to any of the "slave" units, and a normal transmission between master and the currently selected slave. When 9th bit is placed at 1, the 8 data bits of the same character has to contain the identification address, of the device required for communication, while by placing this bit at 0, it is possible to take out or supply info at the selected device.

When QTP 16Big is used, the identification address must be that one selected through the local setup program, on the "SLAVE ADD." menus.

When this byte is sent (with 9th bit set to 1) the QTP 16Big recognizes itself and it waits the string containing chars, data or commands. In this string there could be only a command that involves the return of a response, to send via serial line from QTP part; if there is more than one command with response, the results of the remaining ones are ignored.

Between the transmission of a character and the next one there must be a time interval shorter than **Time Out**, in fact when this time period is elapsed, the QTP 16Big will consider the command sequence terminated and it will begin the answering phase. The Time Out values for each baud rates are below described:
Baud Rate | Time Out | Character transmission time
---|---|---
38400 Baud | 550 µsec | 287 µsec
19200 Baud | 990 µsec | 573 µsec
9600 Baud | 1540 µsec | 1146 µsec
4800 Baud | 3080 µsec | 2292 µsec
2400 Baud | 6105 µsec | 4584 µsec
1200 Baud | 12100 µsec | 9167 µsec

Master unit, once completed the transmission of the last character of the command sequence, must wait for a time equal to:

\[ \text{Character transmission time} + \text{Time Out} \]

before to receive the first character of the response string, transmitted by the QTP 16Big. The answer consists in a character containing the possible code of key pressed (255 = FF Hex means no keys pressed), or a characters sequence that coincide with the response of the command sent in the previous interrogation. Please remind that response is provided also when master unit transmit a command sequence with only the identification name: this simplifies the check for available keys pressed or invalid command.

Several demo programs, coded in different programming languages, are provided with QTP 16Big. They implement Master-Slave 9 bits communication and can be used directly by the user or modified according to the specific needs.

When the master unit is a PC, the user can also take advantage of comfortable DLL libraries that allow to manage Master-Slave 9 bits communication at high level, this means without having to worry about nineth bit, timings, possible electric protocol converters, etc. Also these libraries are provided with the first purchase, complete of user documentation, on a floppy disk or a CD rom.

**NOTES:**

1) To ensure right commands execution, between a call and the next one it is necessary to wait for a time that is proportional to the number of commands sent, and type of operations they involve.

2) If the master unit doesn't support 9 bits communication, it is possible to simulate this bit by using the parity bit and programming its value properly, before any characters transmission, according to this scheme:

   **If the character to transmit has even number of "1" bits**
   - If 9th bit must be 1 -> Program parity to ODD
   - If 9th bit must be 0 -> Program parity to EVEN

   **If the character to transmit has odd number of "1" bits**
   - If 9th bit must be 1 -> Program parity to EVEN
   - If 9th bit must be 0 -> Program parity to ODD

3) When automatic visualizations on display are enabled (scrolling messages, date and time visualization, etc.) the time between two calls, in addition to the time indicated at point 1, must be about 12000 µsec.

4) In a single communication between master unit and QTP 16Big can be transmitted many characters to show and some commands to execute, taking care to doesn't fill the receive buffer, as described in COMMUNICATION BUFFER paragraph.
The following flow chart shows all the described features:

**Figure 37: Flow chart for Master-Slave 9 bits communication**
To explain better the Master-Slave 9 bits protocol, here follows an example where master unit sends three commands to **QTP 16Big** (reading of version number, a string to show on display and a check for possible keys pressed) with a 38.4K baud rate and identification address (SLAVE ADD.) set to 80H value:

<table>
<thead>
<tr>
<th><strong>Master</strong></th>
<th><strong>QTP 16Big</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sends “Reading of version number” command, that is the characters sequence: 80H with nineth bit set to 1 1BH with nineth bit set to 0 56H with nineth bit set to 0 with a delay between the characters lower than 550 µsec</td>
<td>Receives characters of the command and verifies the end with a 550 µsec Time out</td>
</tr>
<tr>
<td>Waits for 837 µsec</td>
<td>Recognizes command sequence, executes the command and stores response for next interrogation</td>
</tr>
<tr>
<td>Receives one character response</td>
<td>Sends the response, which is the code 255=data not available, with nineth bit set to 0</td>
</tr>
<tr>
<td>Sends a string to show on the display, that is the characters sequence: 80H with nineth bit set to 1 1° character of string with nineth bit set to 0 2° character of string with nineth bit set to 0 : : : : : : with a delay between the characters lower than 550 µsec</td>
<td>Receives characters of the command and verifies the end with a 550 µsec Time Out</td>
</tr>
<tr>
<td>Waits for 837 µsec</td>
<td>Recognizes command sequence and shows on the display the characters of the string</td>
</tr>
<tr>
<td>Receives three response characters with the version number previously requested</td>
<td>Transmits saved response which is the version number required by previous command, with nineth bit set 0</td>
</tr>
<tr>
<td>Sends check command for answer data and/or keys pressed, that is the characters sequence: 80H with nineth bit set to 1</td>
<td>Receives characters of the command and verifies the end with a 550 µsec Time Out</td>
</tr>
<tr>
<td>Waits for 837 µsec</td>
<td>Recognizes sequence without commands so performs no operation</td>
</tr>
<tr>
<td>Receives one or more characters corresponding to codes of possible keys pressed</td>
<td>Sends the response, which is the code 255 or possible key pressed code, with nineth bit set to 0</td>
</tr>
</tbody>
</table>

**FIGURE 38:** **EXAMPLE OF MASTER-SLAVE 9 BITS COMMUNICATION**
I2C BUS COMMUNICATION

The system that communicates with **QTP 16Big** in this modality must operate as master, either in transmit and receive mode, following the rules of I2C BUS standard protocol detailed described in the document "THE I2C-BUS SPECIFICATIONS", from PHILIPS semiconductors. This modality requires a synchronization between the systems in communication, as illustrated in the following flow charts:

**FIGURE 39: FLOW CHART FOR MASTER -> QTP 16Big COMMUNICATION IN I2C BUS**
The master must perform a communication with write data direction to supply the characters to visualize and/or the command sequences to execute, and perform a communication with read data direction to get the possible codes of keys pressed and/or the possible answers to the supplied commands.

Each communication involves only the **QTP 16Big** with the slave address equal to those defined in local setup of the terminal, inside "SLAVE ADD," menus. When an I2C BUS network is used, each **QTP 16Big** must be set with a different slave address, and different from the slave addresses of the other possible I2C BUS devices connected to same network.

In order to simplify the complete management, the first data returned by **QTP 16Big** after a read communication, always coincides with the number of characters available in the transmission buffer, that is the number of data the master must receive. Thus the master unit could terminate the communication with proper STOP sequence, only when it has received all these data.

---

**FIGURE 40: FLOW CHART FOR QTP 16Big -> MASTER COMMUNICATION IN I2C BUS**

---

**QTP 16Big**

1. **Power on**
   - Initializes I2C BUS line in slave mode with the slave address selected in local setup

2. **Key pressed**
   - Performs communication as slave transmit, that is:
     1. recognizes START sequence;
     2. receives and compare slave address selected on **QTP 16Big**: if they are equal proceeds otherwise discards all the I2C BUS data until next STOP sequence. Checks communication direction R/W, if read (R/W=1) proceeds in transmission otherwise it goes on in reception (see previous figure);
     3. sends ACK pulse;
     4. transmit number of bytes available in transmit buffer: key codes and/or responses of the executed commands;
     5. checks ACK pulse;
     6. transmit first byte saved in transmit buffer;
     7. checks ACK pulse;
     8. : : :
     9. transmit last byte saved in transmit buffer;
     10. n+1) recognizes STOP sequence.

3. **Bytes received from QTP 16Big**
   - Manages active processes (keyboard, buzzer, visualizations, shift messages, clock, etc.)

---

**MASTER**

1. **Start**
   - Initializes I2C BUS line in master mode with physical protocol (Bit rate) from 500 to 15000 Bit/sec

2. **Manages processes of master unit, including possible I2C BUS communication with other devices provided of a slave address different from those selected on **QTP 16Big**.

3. **Performs communication as master receive, that is:**
   1. sends START sequence;
   2. sends slave address selected on **QTP 16Big** with read direction (R/W=1);
   3. checks ACK pulse;
   4. receive number of bytes to receive from **QTP 16Big**: key codes and/or responses of the executed commands;
   5. sends ACK pulse;
   6. receives first byte from **QTP 16Big**;
   7. sends ACK pulse;
   8. : : :
   9. n) receives last byte from **QTP 16Big**;
   10. n+1) sends STOP sequence.

4. **Manages keys pressed and/or responses to commands previously transmitted**

---

---
NOTES:

1) To ensure right commands execution, between a communication and the next one it is necessary to wait for a time that is proportional to the number of commands sent and type of operations they involve.

2) When automatic visualizations on display are enabled (scrolling messages, date and time visualization, etc.) the time between two calls, in addition to the time indicated at point 1, must be about 12000 µsec.

3) During a communication from master unit to QTP 16Big it can be transferred many characters to visualize and command to execute, taking care to doesn't overflow the receive buffer, as described in paragraph COMMUNICATION BUFFERS.

4) The communications from QTP 16Big to master unit must be planed to doesn't overflow the transmit buffer, as described in paragraph COMMUNICATION BUFFERS.

5) The slave address defined in local setup is 7 bits wide but it is managed as 8 bits value, with the least significant bit (R/W) fixed to 0; so 128 different even values can be selected, in the range 00÷FEH. Moreover when the options .EExxx and .RTC are installed the addresses 160 (A0 Hex) and 162 (A2 Hex) can't be used.

6) When an I2C BUS network connection is used, performs all the configurations described in CN3 - I2C BUS LINE CONNECTOR paragraph, in order to ensure that the line is correctly terminated, from the electric point of view (see figure 16).

7) The QTP 16Big doesn't support the enhancements of I2C BUS protocol (as 10 bits addressing, fast mode, high speed mode, etc.) and the reserved slave addresses: these features can't be used by master unit.
NORMAL COMMUNICATION

The system that communicates with **QTP 16Big** (defined master) in this mode must only transmit the characters to visualize and/or the command sequences to execute, and manage the reception of characters that are the codes of the possible key pressed and/or the possible response to the supplied commands. This mode doesn’t require any synchronization between the two systems in communication and each event is immediately processed from **QTP 16Big**, as illustrated in the following flow chart:

**Figure 42: Flow chart for Normal Communication**
HOW TO START

In this paragraph are listed the operations that must be performed to start using the **QTP 16Big** in a practical and fast way, solving the typical beginners problems. The paragraph contains interesting information even for the users that already know the product and its operating modes, in fact there is the description of a fast functional test. The following steps assume that the command unit is a **Personal Computer** (provided of one free RS 232 serial line and a generic operating system, up to Windows 98), to allow any user to execute them correctly.

A) **Establish connections:**
A1) Perform the serial connection described in figure 43 or on the other hand connect the two communication signals (TX RS232, RX RS232) and the reference ground signal (GND), to free COMx serial port of the PC. It can be easily discovered that this connection cable is reversed and it can be conveniently ordered to **grifo®**, with the code **CCR 9+9R**.

![Figure 43: RS 232 connection with PC](image)

A2) Supply power voltage on CN4 and check that buzzer is immediately disabled and a blinking block cursor is displayed in the left up corner of the display.

B) **Use of demo program:**
B1) On the floppy disks or **grifo®** CD rom received with the first purchase, it is available the file **PRQTP16B.EXE**, that is a demo program for PC that comunicate through RS 232 line with **QTP**. This file once found, must be copied in a comfortable folder on the hard disk of the used PC.

B2) Execute the program copied at point B1 and compile its start questions, by selecting the mounted display type and the possible mounted options. At this point press a key on PC to continue without execute the local setup, in fact the shown configuration coincides with the default one already set on the received **QTP 16Big**.

B3) Carry on demo program execution and check that the operations described on PC monitor are correctly executed on **QTP**; when required interact with the same program in order to test all the available commands, until the end of demo program is reached.

C) **Use of terminal emulation:**
C1) Found the **HYPERTERMINAL** communication program on the PC, that normally is located on Windows menu: "Start | Program | Accessories | Communication", and execute it.
C2) Through the HYPERTERMINAL properties windows, define the communication parameters to:

- Connect directly to COM x (those used at point A1)
- Bit rate 19200
- Data Bits 8
- Parity No
- Stop Bit 1
- Flow control None

and wait the presentation of communication window.

C3) At this point type something on PC keyboard and check that pressed keys are shown on QTP 16Big display. For completeness it can be tested also the effects of some commands by typing their codes sequences always on PC keyboard (this operation is simplified by contemporaneous pression of ALT key and of digits of the decimal code, on the numeric pad: for example to transmit the clear page command with decimal code 12, you can press contemporaneously the ALT key and first the keys 1 and then 2).

C4) Press the keys of QTP 16Big and check that they are correctly shown on PC monitor, with the codes correspondence reported on figure 36.

When during execution of the steps above described a problem or a malfunction is found, we suggest to read and repeat again all the steps carefully and then ensure that default configuration values are saved, through local setup. If malfunction persists please contact directly grifo® technician.

DEMO PROGRAMS

In correspondence of the first purchase together with QTP 16Big it is supplied a floppy disk or a grifo® CD where are saved numerous demo programs that allow to test and estimate immediately the received product. These programs are provided both in executable and source format and they are coded with many high level programming languages (C, PASCAL, BASIC, etc.) either for PC platforms or grifo® microprocessor cards (as GPC®, Mini Modules, etc).

As described in HOW TO START paragraph the programs named PRQTP16B.* use all the commands of QTP through a simple interaction with the user; but many other demos are supplied capable, for example, to: drive QTP connected to a serial network, manage messages, use the Master-Slave 9 bits protocol with DLL libraries, perform scenographic presentation, communicate with I2C BUS line, etc. The user can examine the remarks of these demos and decide himself if they are interesting.

All the demo programs can be used directly or modified or partially used, according to applications requirements, without any authorization, license or additional cost. Furthermore in case of unusual requirements or combinations, specific new demo programs can be obtained, after proper agreement with grifo®.
COMMENDS

This chapter describes all the commands available in QTP16Big firmwares and each relative input and output parameters. The commands are divided in subgroups according with their functions and for each code, or codes sequence, there is a double description: the mnemonic one through the ASCII characters and the numeric one under decimal and hexadecimal format.

The commands respect the ADDS Viewpoint standard so all the sequences begin with ESC character corresponding to the 27 decimal code (1B Hex).

A rich list of demo programs (supplied in source and executable format) shows the practical use modalities of commands: we suggest to add these demo programs, received during first purchase on CD or floppy disk, to this chapter documentation.

A summarized descriptions of all the available commands, their parameters and possible responses, are reported in the tables of APPENDIX A.

COMMENDS FOR CURSOR POSITION

Here follows the list of the commands that acts on cursor position.

CURSOR LEFT

Code: 21
Hex code: 15
Mnemonic: NACK

The cursor is shifted of one position to the left without modifying the display contents. If the cursor is in Home position, it will be placed in the last position of the last row of the display (down, right corner).

CURSOR RIGHT

Code: 6
Hex code: 6
Mnemonic: ACK

The cursor is shifted of one position to the right. If the cursor is placed in the last position of the last row, it will be moved to the Home position that is the first position in the first row (up, left corner).

CURSOR DOWN

Code: 10
Hex code: A
Mnemonic: LF

The cursor will be moved to the line below but it will remain in the same column. If the cursor is in the last display line, it will be moved to the first display line.
CURSOR UP

Code: 26  
Hex code: 1A  
Mnemonic: SUB
The cursor will be moved to the line above but it will remain in the same column. If the cursor is in the first display line, it will be moved to the last display line.

HOME

Code: 1  
Hex code: 1  
Mnemonic: SOH
The cursor is moved to Home position that is the first line, first column of the display, or on the other hand the up, left corner.

CARRIAGE RETURN

Code: 13  
Hex code: D  
Mnemonic: CR
The cursor is moved to the beginning of the line where it was located.

CARRIAGE RETURN+LINE FEED

Code: 29  
Hex code: 1D  
Mnemonic: GS
The cursor is moved to the beginning of line below the one where it was located. If the cursor is at the last display line, it will be moved to the beginning of the first line, i.e Home position.

ABSOLUTE PLACEMENT OF ALPHANUMERIC CURSOR

Code: 27 89  r  c  
Hex code: 1B 59  r  c  
Mnemonic: ESC Y ASCII(r)  ASCII(c)
The cursor is moved to the absolute position indicated by r and c parameters. These characters are the row and column values of the new desired position referred to coordinate 0, 0 of the Home position, plus a constant offset of 32 (20 Hex). The position is expressed in alphanumeric mode so their valid values ranges respectively are 32÷35 and 32÷51. When row and/or column values are not compatible with the specified ranges, the command is ignored. If, for example, the user wants to place the cursor on the second line, third column (row 1, column 2), then the following sequence must be sent: 

27 89 33 34  or  1B 59 21 22 Hex  or  ESC Y ! " 
COMMANDS FOR CHARACTERS ERASURE

Below are described all the commands that deletes one or more characters from the display.

BACKSPACE

<table>
<thead>
<tr>
<th>Code</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code</td>
<td>8</td>
</tr>
<tr>
<td>Mnemonic</td>
<td>BS</td>
</tr>
</tbody>
</table>

This command moves the cursor one character to the left and it erase the contents of the reached position. If the cursor is in Home position, it will be erased the last character of the last row of the display.

CLEAR PAGE

<table>
<thead>
<tr>
<th>Code</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code</td>
<td>C</td>
</tr>
<tr>
<td>Mnemonic</td>
<td>FF</td>
</tr>
</tbody>
</table>

This command clears all data on the display and it moves the cursor to Home position.

CLEAR LINE

<table>
<thead>
<tr>
<th>Code</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code</td>
<td>19</td>
</tr>
<tr>
<td>Mnemonic</td>
<td>EM</td>
</tr>
</tbody>
</table>

This command erases all characters displayed on the current line and it moves the cursor to the first column of the same line.

CLEAR END OF LINE

<table>
<thead>
<tr>
<th>Code</th>
<th>27 75</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code</td>
<td>1B 4B</td>
</tr>
<tr>
<td>Mnemonic</td>
<td>ESC K</td>
</tr>
</tbody>
</table>

This command erases all characters displayed from the current cursor position to the end of line inclusive. The cursor mantains the previous position. If, for example, the cursor is at the beginning of a display line, the complete line will be erased.

CLEAR END OF PAGE

<table>
<thead>
<tr>
<th>Code</th>
<th>27 107</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code</td>
<td>1B 6B</td>
</tr>
<tr>
<td>Mnemonic</td>
<td>ESC k</td>
</tr>
</tbody>
</table>

This command erases all characters displayed from the current cursor position to the end of display inclusive. The cursor mantains the previous position. If, for example, the cursor is at Home position, the complete display will be erased.
COMMANDS FOR CURSOR ATTRIBUTES MANAGEMENT

Below are listed the commands that define the possible cursor attribute and styles.

CURSOR OFF

- **Code:** 27 80
- **Hex code:** 1B 50
- **Mnemonic:** ESC P

The cursor is disabled and it is not more visible.

STEADY STATIC CURSOR ON

- **Code:** 27 79
- **Hex code:** 1B 4F
- **Mnemonic:** ESC O

The cursor is enabled and so it is visible as a not blinking line placed under the character displayed on the current cursor position.

BLINKING BLOCK CURSOR ON

- **Code:** 27 81
- **Hex code:** 1B 51
- **Mnemonic:** ESC Q

The cursor is enabled and it is visible as a blinking rectangular block that is alternatively visualized with the character displayed on the current cursor position.
COMMANDS FOR GENERAL FUNCTIONS

In the following paragraphs are described all the general purpose commands that manage some features of QTP 16Big firmwares. These commands do not come into the other subgroups and for this reason they are described in this specific paragraph.

READ FIRMWARE VERSION

**Code:** 27 86  
**Hex code:** 1B 56  
**Mnemonic:** ESC V

The command returns a string of 3 characters containing the management firmware version that is resident and executed by QTP 16Big.
For example with firmware version 2.1 the following characters will be returned:

```
50 46 49 or 32 2E 31 Hex or 2.1
```

READ CARD CODE

**Code:** 27 160  
**Hex code:** 1B A0  
**Mnemonic:** ESC ASCII(160)

The firmware returns the card code that in case of QTP 16Big coincides with value 10 (0A Hex).
This command is useful especially when on the same serial network there are many different devices and the command unit must recognize them. Naturally the card code identifies the product in a univocal manner.

FLUORESCENT DISPLAY BRIGHTNESS SETTING

**Code:** 27 108 lum  
**Hex code:** 1B 6C lum  
**Mnemonic:** ESC 1 ASCII(lum)

Sets fluorescent display brightness to one of the four possible values, passed in lum parameter, with the following correspondence:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Brightness at 100%</td>
</tr>
<tr>
<td>1</td>
<td>Brightness at 75%</td>
</tr>
<tr>
<td>2</td>
<td>Brightness at 50%</td>
</tr>
<tr>
<td>3</td>
<td>Brightness at 25%</td>
</tr>
</tbody>
</table>

If brightness parameter is not valid, command is ignored.

**NOTE** This command is available only with model QTP 16Big-F4, provided of fluorescent display. In case of QTP 16Big-C4 with LCD display, command must not be sent because it produces the visualization of an undesired character and a shift in all the next received data (alternatively see the CONTRAST REGULATION paragraph).
OPERATING MODE SELECTION

**Code:** 27 65 mode  
**Hex code:** 1B 41 mode  
**Mnemonic:** ESC A ASCII(mode)

It defines the operating mode for the first special characters (those provided of code less than 32 = 20H) and the single character commands. The selected modality is defined by value of `mode` parameter, with the following correspondence:

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (00 Hex)</td>
<td>Command mode</td>
</tr>
<tr>
<td>255 (FF Hex)</td>
<td>Representation mode</td>
</tr>
</tbody>
</table>

If `mode` value is not one of the above described, the command is ignored. Further information about operating mode are reported in CHARACTER VISUALIZATION ON DISPLAY paragraph.

COMMUNICATION RESET

**Code:** 27 163  
**Hex code:** 1B A3  
**Mnemonic:** ESC ASCII(163)

This command reinitializes communication, with no modifications on the other process in execution. Naturally communication is referenced to data exchange between **QTP 16Big** and command unit, so it is independent from used vehicle. In detail the command performs the following operations:

- clears receive buffer;
- eliminates possible characters of response still to return, from transmit buffer;
- eliminates possible pressed keys still to return;
- terminates the managements of all the commands under execution that redirect the supplied characters (message storage, I2C BUS communication as master, etc.).

GENERAL RESET

**Code:** 27 162  
**Hex code:** 1B A2  
**Mnemonic:** ESC ASCII(162)

This command performs a general reset of **QTP 16Big** and it set again an initial condition similar to those available after a power on. In detail the command performs the following operations:

- resets communication as described in previous command;
- clears display and stops possible scrolling messages;
- enables backlight, disables all status LED, buzzer and possible intermittent attributes;
- disables and deactivates the optional clock alarm;
- disables the optional relays outputs;
- reloads the setting saved on EEPROM that are keyclick mode, key codes, user characters patterns, identification slave address, communication protocol, etc.

**NOTE:** This command uses the on board EEPROM, so before executing it is better to check the EEPROM ready status through the proper command. The execution time of this command is some tens of msec.: when after this command many other commands must be sent, it is preferable insert a suitable delay that avoid receive buffer overflow.
BEEP

Code: 7
Hex code: 7
Menomonic: BEL

The buzzer is enabled for a time of 0.1 second. If buzzer was already enable then it is disabled for the same time period, so the audible effect of this command is always recognizable.

BUZZER, LEDS, BACKLIGHT ACTIVATION

Code: 27 50 dev attr
Hex code: 1B 32 dev attr
Mnemonic: ESC 2 ASCII(dev) ASCII(attr)

The on board device selected by dev parameter is driven using attribute specified in attr parameter. In details the devices have the following correspondence:

0÷4 (00÷04 Hex) -> Status LEDs
128 (80 Hex) -> LCD display backlight
255 (FF Hex) -> Buzzer

while the attribute can assume the following values:

0 (00 Hex) -> device OFF
255 (FF Hex) -> device ON
85 (55 Hex) -> device intermittent

If parameters are not valid, command is ignored.

The 5 LEDs numeration is those described in following figure 44: this number is the value of dev parameter and its identifies univocally each LED of QTP 16Big.

With LCD display backlight is referenced the LED bright plane that supply backlight only to LCD display; this backlight is always enabled after a power on, to ensure the best visibility of display, but it can changed by the user to reduce power consumption, to recall operator attention, to signalize an alarm, etc.

The intermittent function is completely autonomous and it doesn't requires any intervent from user side.

For example, to activate the LED number 0 (the one besides the personalization label) with intermittent attribute, the following sequence must be sent:

27 50 0 85 or 1B 32 00 55 Hex or ESC 2 NUL U

NOTE: The use of this command with device associated to backlight on QTP 16Big-F4 model, provided of fluorescet display, has no effects.
**Figure 44: LEDs numeration, position and colour**
COMANDS FOR EEPROM

In this paragraph are described some of the commands that directly manages the data saved on EEPROM/s of QTP 16Big; there are other commands that indirectly use this memory devices but they are described in next paragraphs.

REQUEST FOR EEPROM AVAILABILITY

Code: 27 51
Hex code: 1B 33
Mnemonic: ESC 3

This command checks if the QTP 16Big is ready for management of its on board EEPROM/s. This command must be executed any time there are data to be read or write on this type of memories.

When QTP 16Big receives this command, it returns one of the following codes:

- 6 (06 Hex) (ACK) -> QTP 16Big ready
- 21 (15 Hex) (NACK) -> QTP 16Big not ready

If firmware sends back the NACK code, it is not yet possible to memorize a new data on EEPROM or get an already saved one.

WRITE OF PRESENCE BYTE

Code: 27 33 78 byte
Hex code: 1B 21 4E byte
Mnemonic: ESC ! N ASCII(byte)

This command sets the card presence byte with the value indicated in the byte parameter that must be included in 0÷255 (0÷FF Hex) range.

This byte has a reserved allocation on the on board base EEPROM that, once it is set with the desired value, it allows for example, to verify that QTP 16Big runs correctly, or if there are some communication problems on the serial line.

NOTE: This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is ignored.

READ PRESENCE BYTE

Code: 27 33 110
Hex code: 1B 21 6E
Mnemonic: ESC ! n

The command sends back the value of its presence byte.

For example, it can be useful to verify the presence, or the correct running, of the card and its firmware.

NOTE: This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is not executed and the code 21 (15 Hex) = NAK is returned.
WRITE BYTE ON EEPROM

**Code:** 27 164 addl addh byte  
**Hex code:** 1B A4 addl addh byte  
**Mnemonic:** ESC ASCII(164) ASCII(addl) ASCII(addh) ASCII(byte)

The value passed in `byte` parameter, included in range 0-255 (0-FF Hex), is write in the user EEPROM location for general use, identified by `addl addh` address.

The user EEPROM is a reserved area in the base EEPROM for general purpose, directly managed at byte level with no use of the other commands for messages, presence bytes, etc. The typical uses of this area are the memorization of configurations, setups, identifications, etc. that must be maintained also when power supply is absent. The address that identifies the used location is 16 bits wide and `addl`, `addh` are respectly the most and less significant part. The user EEPROM with the QTP 16Big firmwares has a size of 40 bytes, so the `addl` parameter must be included in range 0-39 (0-27H) while `addh` must always be 0. This choice has been made for compatibility with future expansions and other terminals.

When the command sequence contains not valid data, the command is ignored.

If, for example, the user wants to write the value 100 at address 35 of user EEPROM, then the following sequence must be sent:

```text
27 164 35 0 100 or 1B A4 23 00 64 Hex or ESC ASCII(164) # NUL d
```

**NOTE:** This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is ignored.

READ BYTE FROM EEPROM

**Code:** 27 165 addl addh  
**Hex code:** 1B A5 addl addh  
**Mnemonic:** ESC ASCII(165) ASCII(addl) ASCII(addh)

The value saved in user EEPROM location identified by `addl addh` address is read and returned. As described in the previous command the value of first parameter must be included in range 0-39 (0-27H) while the value of second parameter must always be 0. The returned data is a single character that is included in 0-255 (0-FF Hex) range.

When the command sequence contains not valid data, the command is ignored.

**NOTE:** This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is not executed and the code `21` (15 Hex) = NAK is returned.
COMMANDS FOR KEYBOARD MANAGEMENT

Below are described the commands that can be used to manage the keys available on **QTP 16Big**. Detailed information about keys management and codes transmitted by the terminal, are available in KEYBOARD ACQUISITION paragraph.

KEY RECONFIGURATION

<table>
<thead>
<tr>
<th>Code:</th>
<th>27 55 key n. code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code:</td>
<td>1B 37 key n. code</td>
</tr>
<tr>
<td>Mnemonic:</td>
<td>ESC 7 ASCII(key n.) ASCII(code)</td>
</tr>
</tbody>
</table>

When the selected key n. is reconfigured, each time it is pressed, the card will send the new specified code on communication line.

The value of key n. to be reconfigured must be included in the range \(0\div15\) (\(00\div0F\) Hex) otherwise the command is ignored, and it will substitute the key described in figure 35.

The code value can vary in the range \(0\div254\) (\(00\divFE\) Hex) to obtain the same code when key is pressed, but value 255 (FF Hex) indicates that the key is disabled and when it will be pressed the QTP will not send any code.

Figure 36 reports the default key codes and the paragraph DATA STORED ON EEPROM indicates how to restore these codes in case of unwanted changes.

**NOTE:** This command writes data on the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is ignored.

KEYCLICK ON WITHOUT MEMORIZATION

<table>
<thead>
<tr>
<th>Code:</th>
<th>27 53</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code:</td>
<td>1B 35</td>
</tr>
<tr>
<td>Mnemonic:</td>
<td>ESC 5</td>
</tr>
</tbody>
</table>

The keyclick function is switched on so there is a timed sound feedback when a key is pressed (whenever the buzzer is already enabled, then it is disabled for the same time, in order to always ensure the key pressed recognition). This setting is not saved inside the on board EEPROM so if the terminal is powered off and on it goes back to the previous condition, defined and saved in local setup mode.

KEYCLICK OFF WITHOUT MEMORIZATION

<table>
<thead>
<tr>
<th>Code:</th>
<th>27 54</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code:</td>
<td>1B 36</td>
</tr>
<tr>
<td>Mnemonic:</td>
<td>ESC 6</td>
</tr>
</tbody>
</table>

The keyclick function is disabled so there is not sound feedback when a key is pressed. This setting is not saved inside the on board EEPROM so if the terminal is powered off and on it goes back to the previous condition, defined and saved in local set up mode.
KEYCLICK ON WITH MEMORIZATION

*Code:* 27 33 53  
*Hex code:* 1B 21 35  
*Mnemonic:* ESC ! 5

This command enables **keyclick** function, so there is an audible feedback when a key is pressed (whenever the buzzer is already enabled, then it is disabled for the same time, in order to always ensure the key pressed recognition). This setting is stored on the on board EEPROM so if the card is turned off and on, it keeps the current condition.

**NOTE:** This command writes data on the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is ignored.

KEYCLICK OFF WITH MEMORIZATION

*Code:* 27 33 54  
*Hex code:* 1B 21 36  
*Mnemonic:* ESC ! 6

This command disables **keyclick** function, so there is not audible feedback when a key is pressed. This setting is stored on the on board EEPROM so if the card is turned off and on, it keeps the current condition.

**NOTE:** This command writes data on the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is ignored.

![Figure 45: Front panel with keyboard](image-url)
COMMANDS FOR USER CHARACTERS

**QTP 16Big** lets the user define and show up to 8 user characters; those characters can be used to represent on display special characters, pseudo graphic characters, special symbols, etc. that are not still available in the same display (please refer to tables in appendix B). The user characters can be defined and saved with a pattern equal to a 5 x 8 pixels matrix, so organized:

![Figure 46: User Characters Pattern](image)

The user characters representation is really simple in fact it is sufficient to send the proper code (0 to 7 equal to 8 to 15) with a previous setting of representation mode, through OPERATING MODE SELECTION command.

When the user characters are saved, their patterns are written on EEPROM and then they are reloaded on display any time the terminal is powered on or when GENERAL RESET command is executed.

**NOTE:** Please remind that on **QTP 16Big-F4** with fluorescent displays the character has a 5 x 7 pixels matrix (Pat 0÷Pat 6) and the last row of the pattern is not displayed. Furthermore the value of Pat 7.4 pixel defines the status of all the five pixels Pat 7.4÷Pat 7.0, or in other words it defines the status of underline attribute of the defined character.
DEFINITION OF USER CHARACTER

Code: 27 66 nchar Pat0 ÷ Pat7
Hex code: 1B 42 nchar Pat0 ÷ Pat7
Mnemonic: ESC B ASCII(nchar) ASCII(Pat0) ÷ ASCII(Pat7)

After the two command identification codes, other 9 bytes must be sent with the following meaning:

- \text{nchar} (0÷7) (00÷7 Hex) \rightarrow Number of user character to define
- \text{Pat0} (0÷31) (00÷1F Hex) \rightarrow First byte of pattern equal to first high row of character.

This command loads on the display the pattern of the user character \text{nchar} with the value placed in the eight bytes \text{Pat0 ÷ Pat7}, as described in figure 46; the pattern is only defined but not saved, so when QTP 16Big is turned off and on, the user character \text{nchar} doesn't maintain the supplied pattern.

For example to define the user character 5 as an empty rectangle with maximum size, the following sequence has to be sent:

\begin{verbatim}
27 66  5   31  17  17  17  17  17  31
\end{verbatim}

or

\begin{verbatim}
1B 42  05  1F  11  11  11  11  11  11  1F  Hex
\end{verbatim}

DEFINITION AND MEMORIZATION OF USER CHARACTER

Code: 27 33 66 nchar Pat0 ÷ Pat7
Hex code: 1B 21 42 nchar Pat0 ÷ Pat7
Mnemonic: ESC ! B ASCII(nchar) ASCII(Pat0) ÷ ASCII(Pat7)

After the three command identification codes, other 9 bytes must be sent with the following meaning:

- \text{nchar} (0÷7) (00÷7 Hex) \rightarrow Number of user character to define and save
- \text{Pat0} (0÷31) (00÷1F Hex) \rightarrow First byte of pattern equal to first high row of character.

This command loads on the display the pattern of the user character \text{nchar} with the value placed in the eight bytes \text{Pat0 ÷ Pat7}, as described in figure 46; moreover the pattern is also saved on EEPROM, so if QTP 16Big is turned off and on, the user character \text{nchar} maintain the supplied pattern.

NOTE: This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is ignored.

Execution time of the command is about 80 msec. When the command has been transmitted and several commands must follow, it is better to insert a delay to avoid receive buffer overflow.
COMMANDS FOR MESSAGE MANAGEMENT

In the following paragraphs are described all the commands that manage messages, available in QTP 16Big terminal. The messages are 20 characters sequence that can be saved on board EEPROM/s and then reloaded or represented on display, simply by suppling the same message identification number. The most important function of messages is the possibility to show constant information on the display (i.e. alarms, equipment status, user instructions, etc.) without the transmission of the numerous characters of these information but only the few characters of the command. The QTP 16Big firmware manages the scrolling messages visualization, too; with this feature on a single line of display can be shown more text that continuously shift from right to left. Additionally the messages coincide with the entity used by power on visualization command, described in a following paragraph.

Please remind that a comfortable program for PC, named QTP EDIT, allows any user to edit the messages, save and load them on PC disks and transmit/receive them directly to/from QTP serially connected to same PC.

In the default configuration the QTP 16Big install a base EEPROM with a size of 2048 bytes, that can store up to 95 messages; through an optional EEPROM, that must be specified in the order phase, the maximum number of messages can be increased up to 3371, as described in next table. When the user has special requirements about EEPROM sizes, other different dimensions can be obtained, but they must be previously agreed upon grifo®.

READING OF MAX MESSAGE NUMBER

| Code:  | 27 110 |
| Hex code: | 1B 6E |
| Mnemonic: | ESC n |

This command returns the number of the last messages that can be saved. It varies in compliance with the size of the EEPROM/s installed on the card, as reported in the below table:

<table>
<thead>
<tr>
<th>Version (option)</th>
<th>EEPROM size</th>
<th>Messages n°</th>
<th>Last message n°</th>
<th>Last group n° (max.grp) and last message n° of group (max.msg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>2K Bytes</td>
<td>95 (005FH)</td>
<td>94 (5EH)</td>
<td>0 (00H) 94 (5EH)</td>
</tr>
<tr>
<td>.EE128</td>
<td>2+16K Bytes</td>
<td>914 (0392H)</td>
<td>255 (FFH)</td>
<td>3 (03H) 145 (91H)</td>
</tr>
<tr>
<td>.EE256</td>
<td>2+32K Bytes</td>
<td>1733 (06C5H)</td>
<td>255 (FFH)</td>
<td>6 (06H) 196 (C4H)</td>
</tr>
<tr>
<td>.EE512</td>
<td>2+64K Bytes</td>
<td>3371 (0D2BH)</td>
<td>255 (FFH)</td>
<td>13 (0DH) 42 (2AH)</td>
</tr>
</tbody>
</table>

**Figure 47: Number of messages on EEPROM**

This command has been implemented for compatibility and interchangeability with other QTP operator panels and it returns a valid response only when optional EEPROM is not available. To obtain always the right messages number it is suggested the following command.
READING OF LAST GROUP AND MESSAGE MANAGED

| Code:          | 27 33 109 |
| Hex code:      | 1B 21 6D  |
| Mnemonic:      | ESC ! m   |

This command returns the number of the last group of messages that can be saved and the number of the last message inside this group. Both these values change according to EEPROM/s sizes installed on the board, as described in the last column of previous table. As convention we define \texttt{max.grp} and \texttt{max.msg} the two numbers returned as response by this command, and they will be used in all the following descriptions.

All the numerous messages of \texttt{QTP 16Big} have been divided in groups of 256 messages, to allow their identification. With this technique it is really easy and fast to convert the two returned values in the number of the last message = \texttt{max.grp} * 256 + \texttt{max.msg}.

SELECT CURRENT MESSAGE GROUP

| Code:          | 27 33 77 grp |
| Hex code:      | 1B 21 4D grp |
| Mnemonic:      | ESC ! M ASCII(grp) |

It selects the message group \texttt{grp} that must be used with the following commands for messages management. The message group identifies a set of 256 messages, as described in previous command, that has been adopted to easily address all the messages with a byte codification of the message number. The valid values for message group are those included in range \texttt{0-max.grp} (where \texttt{max.grp} is the last group reported on figure 47), otherwise the command is not executed.

After a power on or a GENERAL RESET command, it is always selected the first message group, that is those with number \texttt{0} (\texttt{00 Hex}).

If, for example, the user wants to select the group for message number 300, then the following sequence must be sent:

\begin{verbatim}
27 33 77 1  or  1B 21 4D 01 Hex  or  ESC ! M SOH
\end{verbatim}

MESSAGE STORAGE

| Code:          | 27 33 67 mess.n. chr.0 ÷ chr.19 |
| Hex code:      | 1B 21 43 mess.n. chr.0 ÷ chr.13 Hex |
| Mnemonic:      | ESC ! C ASCII(mess.n.) ASCII(chr.0) ÷ ASCII(chr.19) |

This command stores the 20 characters message, identified by \texttt{mess.n.} parameter, on the on board EEPROM/s, in the currently selected message group. The 20 chars which form the message must be visualizable on the display so they must be included in the range \texttt{0+255 (00+FF Hex)}. The message number must be included in the range \texttt{0+max.msg} when the last message group is selected, otherwise in the range \texttt{0+255 (00+FF Hex)}, or in other words identify one of the available messages.

If this number is out of range, the command is ignored.

NOTE: This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is ignored.

The execution of this command depends on jumper J9 configuration, as described in paragraph OPTIONAL EEPROM PROTECTION.
MESSAGE READING

**Code:** 27 33 69 mess.n.
**Hex code:** 1B 21 45 mess.n.
**Mnemonic:** ESC ! E ASCII(mess.n.)

This command reads from EEPROM/s the 20 characters message identified by mess.n. parameter, in the currently selected message group, and it returns this message, beginning from the first char of the string.

The message number must be included in the range 0 to max.msg when the last message group is selected, otherwise in the range 0 to 255 (00÷FF Hex), or in other words identify one of the available messages. If this number is out of range, the command is ignored.

**NOTE:** This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is not executed and 20 characters with the code 21 (15 Hex) = NAK are returned.

VISUALIZATION OF MESSAGES

**Code:** 27 33 68 mess.n. n
**Hex code:** 1B 21 44 mess.n. n
**Mnemonic:** ESC ! D ASCII(mess.n.) ASCII(n)

This command visualizes n 20 characters messages on the display, beginning from current cursor position.

The first of the n messages is that one having the number corresponding to mess.n., in the currently selected message group, while the remaining messages are those immediately subsequents in EEPROM/s.

The mess.n. value and the number of the following messages defined by n, must be included in the range 0 to max.msg when the last message group is selected, otherwise in the range 0 to 255 (00÷FF Hex), or in other words identify one of the available messages. If these numbers are out of range, the command is ignored.

The n quantity of messages to be visualized depends on the maximum number of characters of the installed display; on QTP 16Big this number is 80 and so the maximum number of messages is 4. In other words the n parameter can be set with a value in the range 1 to 4 and if it is out of this range, the command is ignored.

Once the command is executed the cursor is placed in the next position of the last character visualized; if the last character of the said message occupies the last position of the display, the cursor will be placed in Home position.

For example, to visualize the messages number 10 and 11, first of all it will be necessary to send the command that select message group 0, and then the following sequence:

```
27 33 68 10 2  
1B 21 44 0A 02 Hex  
ESC ! D LF STX
```

**NOTE:** This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is delayed until the operation under execution is completed.
SCROLLING MESSAGES VISUALIZATION

**Code:** 27 33 83 mess.n. n.chr  
**Hex code:** 1B 21 53 mess.n. n.chr  
**Mnemonic:** ESC ! S ASCII(mess.n.) ASCII(n.chr)

This command visualizes a string, long n.chr characters, on the first line of display in sliding mode. The string is shifted from right to left and so the user can visualize on a single line (the first) many information, more than the normal 20 characters.

The string of n.chr characters, begins with the first character of the mess.n. message already stored in EEPROM/s, in the currently selected message group, and continues with next characters (always saved in following EEPROM/s messages).

The mess.n. value must be included in the range 0÷max.msg when the last message group is selected, otherwise in the range 0÷255 (00÷FF Hex), or in other words identify one of the available messages. If this number is out of range, the command is ignored.

Instead the n.chr parameter must range in the following values:

- **0** Stops the scrolling messages visualization (the mess.n. value doesn't care).
- **20÷200** Enables sliding visualization of the specified characters.

If n.chr. value is out of the specified ranges or it points after the last character of the last message stored in EEPROM/s, the command will be ignored.

The scrolling messages visualization is always performed on the first display line and the cursor position and attributes are mantained.

For example, if you wish to visualize a 35 characters string in sliding mode, composed by message 10 (20 characters) and by the first 15 characters of message 11, first of all it will be necessary to send the command that select message group 0, and then the following sequence:

27 33 83 10 35 or 1B 21 53 0A 23 Hex or ESC ! S LF #

**NOTE:** This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is delayed until the operation under execution is completed.

The message visualization in sliding mode is managed in background and so there is an increased firmware execution time, that causes a subsequent slowing down on interpretation of the data coming from command unit. This is the reason why it is necessary to wait for few msec between the transmission of 20÷30 bytes data blocks when many information and/or commands are sent to QTP 16Big. In this way misunderstanding and interpreting problems of the received data, caused by receive buffer overflow, are completely avoid.
SET AUTOMATIC VISUALIZATION

Code: 27 150 255 mess.n len shift r c
Hex code: 1B 96 FF mess.n len shift r c
Mnemonic: ESC ASCII(150) ASCII(255) ASCII(mess.n) ASCII(len) ASCII(shift) ASCII(r) ASCII(c)

This command sets the power on visualization of QTP 16Big that is a possible representation automatically displayed, immediately after a power on phase, and that stay on display until the first data is received from command unit.

In this paragraph the term visualization and representation always refer to the visual results on the display and it can be selected among many possibilities, defined by the proper parameters of the command. In detail it is possible to display: a single message in any position, a static messages sequence (screen) in any position and an auto scrolling messages sequence only on the first row.

For this purpose the parameters have the following meaning:

- **mess.n**: it coincides with the number of the first message to show, of the group 0, and the possible others, are those immediately successive on EEPROM. The mess.n value, in order to be valid, must respect the conditions below described:
  - $0 \leq \text{mess.n} \leq \text{msggrp0}$ -> to enable visualization or, in other words, it must select an available message (where maxgrp0 is the number of the last message in group 0, described in the fourth column of figure 47)
  - mess.n=255 (FFH) -> to disable visualization

- **len**: it coincides with the length of visualization and it is expressed in a different way according with visualization attribute:
  - static visualization (shift=0): it corresponds to messages number. The len value, in order to be valid, must respect the following conditions:
    - mess.n+len<=$\text{msggrp0}$ -> it must select available messages in group 0 (where msggrp0 is the number of the last message in group 0, described in the fourth column of figure 47)
    - $1 \leq \text{len} \leq 4$ -> it must not exceed the display dimensions equal to 4 messages of 20 characters max.
  - scrolling visualization (shift=255): it corresponds to characters number. The len value, in order to be valid, must respect the following conditions:
    - len=0 -> stops the scrolling in execution
    - $20 \leq \text{length} \leq 200$ -> the range of shiftable characters

- **shift**: it coincides with the scrolling visualization attribute and it can assume two possible values:
  - 0 00H NUL -> static visualization
  - 255 FFH ASCII(255) -> scrolling visualization

- **r**: when the representation is static (shift=0) it coincides with the row where the visualization starts and its valid values range is 0-3. When the visualization has the scroll attribute (shift=255) the representation occurs always on the first line of display and the parameter value doesn’t care.

- **c**: when the representation is static (shift=0) it coincides with the column where the visualization starts and it valid values range is 0-19. When the visualization has the scroll attribute (shift=255) the representation occurs always on the first line of display and the parameter value doesn’t care.
Whenever in the received sequence there are not valid data, the command is ignored vice versa the automatic visualization is immediately saved on EEPROM in order to maintain it when power off and on occurs. In fact this command arranges the QTP 16Big as a system, that show the visualization before than any communications occur with the command unit, and for this reason the visualization is saved, recalled and managed by the single terminal.

The power on visualization uses only the messages that belong to first group 0 and the message group currently selected when the command is received, is completely indifferent.

Among the typical use of this command there are: the visualization of the general information (name, address, telephone, etc.) of the company that developed the application, the timed visualization of the firmware and/or software version in execution, the immediate visualization of instructions for the operator that has just powered on the machine. Moreover the command adds another application field for the QTP 16Big as a stand alone visualization system, that can even work without any external command unit.

For example, if you wish to enable a power on view, with static visualization of messages 10 and 11 on the second row of display, it will be necessary to send the following sequence:

```
27 150 255 10 2 0 1 0
or
1B 96 FF 0A 02 00 01 00 Hex
or
ESC ASCII(150) ASCII(255) LF STX NUL SOH NUL
```

NOTE: This command uses the on board EEPROM, so before executing it is better to check the EEPROM availability through the proper command; in fact if it is not ready the command is ignored.

When a command unit is connected to QTP 16Big ensures that it doesn't transmit any characters (even spurious ones) during power on phase, or the power on messages will be immediately removed from display.
COMMANDS FOR I2C BUS COMMUNICATION AS MASTER

On QTP 16Big are available a group of commands let the command unit communicate in master mode to all the I2C BUS peripherals that operates as slaves. This commands coincide with the essential elements that once properly combined allows to communicate with any device provided of this standard (temperature sensors, A/D and D/A converters, etc.).

Naturally these commands are superfluous when communication with QTP 16Big is already performed on I2C BUS line in fact in this case the command unit can directly exchange data with the I2C BUS peripherals, as it already does with QTP. Viceversa the commands become really useful when the command unit communicate through the asynchronous serial line and in this condition QTP 16Big acts as a serial <-> I2C BUS converter.

The figure 48 shows a possible connection diagram for some I2C BUS peripherals that can be managed by these commands; certainly the operator panel resources can be expanded with a reduced cost and a short development time of management software.

About physical protocol of I2C BUS line managed by these commands, the following features are used:

- Bit rate: 50000 bits per second
- Mode: Master (transmit and receive)
- Slave Address: all the even addresses in range 0÷254 (00÷FE Hex), except the values 160 (A0 Hex), 162 (A2 Hex) already used on board and the one defined in local setup of QTP.

About electric connection, please remind that the I2C BUS line in Master mode coincides with the one used for Slave mode. The connection is always performed through CN3, by following the indications already reported in paragraph CN3 - I2C BUS LINE CONNECTOR.

START I2C BUS

<table>
<thead>
<tr>
<th>Code:</th>
<th>27 250</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code:</td>
<td>1B FA</td>
</tr>
<tr>
<td>Mnemonic:</td>
<td>ESC ASCII(250)</td>
</tr>
</tbody>
</table>

This command generates the start sequence on the I2C BUS line.

STOP I2C BUS

<table>
<thead>
<tr>
<th>Code:</th>
<th>27 251</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex code:</td>
<td>1B FB</td>
</tr>
<tr>
<td>Mnemonic:</td>
<td>ESC ASCII(251)</td>
</tr>
</tbody>
</table>

This command generates the stop sequence on the I2C BUS line.
TRANSMIT BYTE ON I2C BUS

Code: 27 252 byte
Hex code: 1B FC byte
Mnemonic: ESC ASCII(252) ASCII(byte)

This command transmits the byte passed as parameter on the I2C BUS line and it reads the acknowledge bit from peripheral. At the end of operation it always returns a character with the read bit status that can assume the possible values 0 (00 Hex) or 1 (01 Hex).

The command can be used to perform many operations required by I2C BUS communication, infact all the data exchanged with this standard are organized in bytes (Slave Address, commands, addresses, etc.) that must be sent to peripheral devices.

RECEIVE BYTE FROM I2C BUS

Code: 27 253 ack
Hex code: 1B FD ack
Mnemonic: ESC ASCII(253) ASCII(ack)

This command receive a byte from the I2C BUS line and returns it. Then it performs the following acknowledge operations, according with ack parameter value:

0 (00 Hex) -> sends bit at 0 (ACK)
1 (01 Hex) -> sends bit at 1 (NAK)

The command can be used to perform many operations required by I2C BUS communication, infact all the data exchanged with this standard are organized in bytes (data, status, addresses, etc.) that must be received from peripheral devices.
COMMANDS FOR SRAM AND CLOCK

Here follow commands that manage the backed Real Time Clock and SRAM available on QTP 16Big.RTC. Whenever the .RTC option is not ordered all the commands described in this paragraph are ignored because the necessary hardware components are not installed.

The correct updating of the clock and the SRAM data preservation is ensured also when power supply is not available, as described in BACK UP paragraph.

Among the most important functions of the SRAM+RTC we remind the following examples: the automatic visualization on display of current time and date; the use of a complete clock through one of the communication lines; the management of a clock alarm that autonomously check a predefined time; the activation of one of the relay outputs when the clock alarm time is reached; the memorization and the acquisition of data that change values continuously; etc.

WRITE BYTE ON BACKED SRAM

**Code:** 27 33 71 addr byte  
**Hex code:** 1B 21 47 addr byte  
**Mnemonic:** ESC ! G ASCII(addr) ASCII(byte)

It writes the value passed in byte parameter, variable in range 0÷255 (00÷FF Hex), on the backed SRAM. The address where memorize this byte is passed in addr parameter and it must be included in the range 32÷255 (20÷FF Hex) otherwise the command is ignored.

READ BYTE FROM BACKED SRAM

**Code:** 27 33 103 addr  
**Hex code:** 1B 21 67 addr  
**Mnemonic:** ESC ! g ASCII(addr)

It returns a value that is the byte stored on backed SRAM, at the address specified by the addr parameter. The address must be included in the range 32÷255 (20÷FF Hex) otherwise the command is ignored.
SET CLOCK

Code: 27 33 70 hou min sec day mon yea wee
Hex Code: 1B 21 46 hou min sec day mon yea wee
Mnemonic: ESC ! F ASCII(hou) ASCII(min) ASCII(sec) ASCII(day) ASCII(mon) ASCII(yea) ASCII(wee)

The on board Real Time Clock is set with the data contained in the passed parameters; if one of these ones has a value not included in the allowed range, the command is ignored and none of the RTC fields are changed.

Here under it is listed the detailed meaning of the 7 parameters above described, and their validity ranges.

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>RANGE</th>
<th>MEANING</th>
</tr>
</thead>
<tbody>
<tr>
<td>hou</td>
<td>0÷23 (00H÷17H)</td>
<td>Hours</td>
</tr>
<tr>
<td>min</td>
<td>0÷59 (00H÷3BH)</td>
<td>Minutes</td>
</tr>
<tr>
<td>sec</td>
<td>0÷59 (00H÷3BH)</td>
<td>Seconds</td>
</tr>
<tr>
<td>day</td>
<td>0÷31 (00H÷1FH)</td>
<td>Day of month</td>
</tr>
<tr>
<td>mon</td>
<td>1÷12 (00H÷0CH)</td>
<td>Month</td>
</tr>
<tr>
<td>yea</td>
<td>0÷99 (00H÷63H)</td>
<td>Year</td>
</tr>
<tr>
<td>wee</td>
<td>0÷6 (00H÷06H)</td>
<td>Day of week: 0 -&gt; Sunday</td>
</tr>
</tbody>
</table>

**FIGURE 49: REAL TIME CLOCK PARAMETERS**

**NOTE:** To ensure the validity of the two digits year, managed by clock of QTP 16Big_RTC, it is absolutely necessary to send the SET CLOCK command at least one time each 4 years. This is a quite normal condition in fact tipically the clock time is adjusted one or two times for any year.

ACQUIRE CLOCK

Code: 27 33 102
Hex Code: 1B 21 66
Mnemonic: ESC ! f

The command returns 7 values, named hou, min, sec, day, mon, yea, wee, that corresponds to the current time and date parameters, acquired from the on board Real Time Clock.

The meaning of these bytes is the one explained in previous table.
SHOW TIME ON DISPLAY

**Code:** 27 33 116 r c frm  
**Hex Code:** 1B 21 74 r c frm  
**Mnemonic:** ESC ! t ASCII(r) ASCII(c) ASCII(frm)

The time acquired from the on board Real Time Clock is displayed starting from the display position passed in r and c parameters. These express the row and column values of display referred to Home position with coordinate 0, 0, plus a constant offset of 32 (20 Hex). The position is expressed in alphanumeric mode so their valid values ranges respectively are 32÷35 and 32÷51. When row and/or column values are not compatible with the specified ranges, the command is ignored.

The **frm** parameter is used to specify the visualization format, with the following meaning:

- **Bit 0** -> 1 Enables the automatic visualization of time in the display position defined by r and c.
  - 0 Disables the automatic visualization of time. The values of r and c are not used.
- **Bit 1** -> 1 The time is visualized in AM/PM format: HH:MM:SSm where HH are the hours, MM the minutes, SS the seconds and m is the AM (a) or PM (p) indication.
  - 0 The time is visualized in 24H format: HH:MM:SS where HH are the hours, MM the minutes and SS the seconds.
- **Bit 2** -> 1 Enables the alternate visualization of time and date on the same position of display.
  - 0 Disables the alternate visualization of time and date.
- **Bit 3+7** -> 0 Reserved for future expansions. They must be set to 0 value.

For example, if you wish to visualize the time in 24 H format, starting from the 10th character of the 2nd row (row 1, column 9), it will be necessary to send the following sequence:

```
27 33 116 33 41 1 or 1B 21 74 21 29 01 Hex or ESC ! t ! ) SOH
```

**NOTE:** The time visualization is managed in background and so there is a slowing down on interpretation of data coming from command unit. This is the reason why it is necessary to wait for few msec between the transmission of 20÷30 bytes data blocks when many information and/or commands are sent to QTP 16Big. In this way misunderstanding and interpreting problems of the received data, caused by receive buffer overflow, are completely avoid.
SHOW DATE ON DISPLAY

**Code:**  27 33 100  r c frm  
**Hex code:**  1B 21 64  r c frm  
**Mnemonic:** ESC ! d ASCII(r) ASCII(c) ASCII(frm)

The date acquired from the on board Real Time Clock is displayed starting from the display position passed in r and c parameters. These express the row and column values of display referred to Home position with coordinate 0, 0, plus a constant offset of 32 (20 Hex). The position is expressed in alphanumeric mode so their valid values ranges respectively are 32÷35 and 32÷51. When row and/or column values are not compatible with the specified ranges, the command is ignored.

The *frm* parameter is used to specify the visualization format, with the following meaning:

- **Bit 0** ➔ 1 Enables the automatic visualization of date in the display position defined by r and c.
  0 Disables the automatic visualization of date. The values of r and c are not used.

- **Bit 1** ➔ 1 The date is visualized in English format: MM-DD-YY, where MM is the month, DD the day and YY the year.
  0 The date is visualized in Italian format: DD-MM-YY, where DD is the day, MM the month and YY the year.

- **Bit 2** ➔ 1 Enables the alternate visualization of date and time on the same position of display.
  0 Disables the alternate visualization of date and time.

- **Bit 3÷7** ➔ 0 Reserved for future expansions. They must be set to "0" value.

Please note that the week day is not displayed.

The alternate visualization of date and time, on the same display position, is obtained by sending both the commands SHOW TIME and SHOW DATA with the bit *frm*.2=1 and with the same coordinate r, c values.

For example, if you wish to visualize the date in English format MM-DD-YY, starting from the 10th character of the 2nd row (row 1, column 9), it will be necessary to send the following sequence:

27 33 100 33 41 3 or 1B 21 64 21 29 03 Hex or ESC ! d ! ) ETX

**NOTE:** The date visualization is managed in background and so there is a slowing down on interpretation of data coming from command unit. This is the reason why it is necessary to wait for few msec between the transmission of 20÷30 bytes data blocks when many information and/or commands are sent to **QTP 16Big**. In this way misunderstanding and interpreting problems of the received data, caused by receive buffer overflow, are completely avoid.
SET CLOCK ALARM

**Code:** 27 33 70 hou min sec day mon dton  
**Hex Code:** 1B 21 46 hou min sec day mon dton  
**Mnemonic:** ESC ! F ASCII(hou) ASCII(min) ASCII(sec) ASCII(day) ASCII(mon) ASCII(dton)

The command sets and enables the on board clock alarm with the data contained in the passed parameters; if one of these ones has a value not included in the allowed range (see figure 49), the command is ignored.

Once received the command, the **QTP 16Big** clock alarm is set and it is also enabled and deactivated. The word enabled means that the current time and date are compared with those of the clock alarm and, when all the parameters match, the clock alarm is activated. At this point the clock remains active for the time period passed in dton parameter and then it is deactivated and disabled. The dton alarm activation time is expressed in tens of seconds, with the following correspondence:

- dton = 0 -> alarm remains active 1 second
- 1 -> alarm remains active 10 seconds
- 2 -> alarm remains active 20 seconds
- ...: : : : :   :
- 255 -> alarm remains active 2550 seconds

When the menu **RL2 FUNCT.** of local setup is set to **ALARM** option, during the clock alarm activation time, it is enabled also the RL2 relay, equal to NO OUT2 output on CN7, in a completely automatic mode (when J10 is connected). In this condition the user can act even on an external actuator with no requirements of additional operations (read paragraph COMMANDS FOR RELAYS OUTPUTS MANAGEMENT for additional information on the use of this output).

The user must remind the following notes that concern the clock alarm:

**NOTE:**
- After a power on the alarm clock is always disabled and deactivated; so a possible enable, performed before the the power off, is lost. Viceversa the defined alarm parameters are maintained and they can be acquired with the proper ACQUIRE CLOCK ALARM command.
- The command SET CLOCK doesn't affect the clock alarm setting and functionality.
- The described SET CLOCK ALARM command always enables and deactivates the clock alarm, independently from the previous status. A possible alarm activation time under execution it is interrupted and the NO OUT2 output is not active.
- At the end of activation time the alarm clock is disabled and deactivated; in other words it can be re-enabled only with a new SET CLOCK ALARM command.
- The current status of the clock alarm, included enable and active conditions, can be comfortably acquired through the following command.
ACQUIRE CLOCK ALARM

**Code:** 27 33 97  
**Hex Code:** 1B 21 61  
**Mnemonic:** ESC ! a

The command returns the 5 clock alarm parameters (hou, min, sec, day, mon), defined with the last command SET CLOCK ALARM, and a sixth value with the current clock alarm status, with the following information:

- **Bit 0** -> clock alarm enabled status
  - 1 Clock alarm enabled and waiting for the defined time
  - 0 Clock alarm disabled (no controls are performed)

- **Bit 1** -> clock alarm active status
  - 1 Clock alarm activated, that means predefined time reached and it waits the elapsing of the alarm activation time `dton`. In this condition the NO OUT2 output is active.
  - 0 Clock alarm deactivated, that means predefined time is not still reached or it has already been reached and the alarm activation time `dton` is finished. In this condition the NO OUT2 output is not active.

- **Bit 2-7** -> 0 Not used.

Through this command the user can easily check if the current time and date have reached and matched the time already set with the previous command. So he must not provide complicated time controls, that involves numerous crossed checks, in his application software.
COMMANDS FOR RELAYS OUTPUTS MANAGEMENT

Below are listed the commands that manage the optional digital relays outputs, available in the QTP 16Big.RELAY. Please remind that these outputs are available on proper connector (as described in CN7- RELAYS OUTPUTS CONNECTOR) and that the same outputs can directly drive the field signals, when they respect the limits reported in the ELECTRIC FEATURES paragraph. Whenever the .RELAY option is not ordered all the commands described in this paragraph are ignored because the necessary hardware components are not installed. Moreover the commands related to NO OUT2 output are executed only if the menu RL2 FUNCT. of the local setup is set to USER option.

Among the most important features of the relays outputs it can be listed, for example, the management of power relays, lamps, electric valves, motors, heaters, etc. or any other actuators that assume only the two state of active/deactive (ON/OFF).

As described in all following paragraphs the digital outputs are referenced by the same name used in connectors descriptions (see figures 17 and 18) and when required, a sequential numeration starting from 1.

WRITE ALL DIGITAL OUTPUTS

Code: 27 166 out
Hex code: 1B A6 out
Mnemonic: ESC ASCII(166) ASCII(out)

All the digital relays outputs are set with out value, according to following correspondence:

(MSB) 0 0 0 0 0 NO OUT 3 NO OUT2 NO OUT1 (LSB)

Where NO OUT n stands for the logic state, 0 (output deactivate=relay contact open) or 1 (output active=relay contact closed), that the respective relay outputs, on CN7, must assume.

When the received sequence contains invalid data the command is ignored.

If, for example, only the NO OUT3 and NO OUT1 outputs must be enabled, then the following sequence must be sent:

27 166 5 or 1B A6 05 Hex or ESC ASCII(166) ENQ

ENABLE SINGLE DIGITAL OUTPUT

Code: 27 168 bit
Hex code: 1B A8 bit
Mnemonic: ESC ASCII(168) ASCII(bit)

This command sets to logic state 1 (output active=relay contact closed) the relay digital output identified by bit parameter, that has the following correspondence with CN7 signals:

1 -> NO OUT1  2 -> NO OUT2  3 -> NO OUT3

When the received sequence contains invalid data, the command is ignored.

If, for example, the output NO OUT2 must be enabled with no modifications on the remaining outputs, then the following sequence must be sent:

27 168 2 or 1B A8 02 Hex or ESC ASCII(168) STX
FIGURE 50: AVAILABLE CONNECTIONS DIAGRAM
DISABLE SINGLE DIGITAL OUTPUT

**Code:** 27 170 bit

**Hex code:** 1B AA bit

**Mnemonic:** ESC ASCII(170) ASCII(bit)

This command sets to logic state 0 (output deactive=relay contact opened) the relay digital output identified by bit parameter, that has the following correspondence with CN7 signals:

1 -> NO OUT1  
2 -> NO OUT2  
3 -> NO OUT3

When the received sequence contains invalid data, the command is ignored.

If, for example, the output NO OUT1 must be disabled with no modifications on the remaining outputs, then the following sequence must be sent:

27 170 1 or 1B AA 01 Hex or ESC ASCII(170) SOH
APPENDIX A: COMMANDS SUMMARY TABLES

The following tables list a summary of all the commands recognized by QTP 16Big firmware. Please remind that these commands are compatible with ADDS Viewpoint standard. As in all the other descriptions of the manual, the codes are reported in three formats: decimal, hexadecimal and mnemonic, while the last column reports the number of data returned by command as response.

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>CODE</th>
<th>HEX CODE</th>
<th>MNEMONIC</th>
<th>Ret.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>01</td>
<td>01</td>
<td>SOH</td>
<td>0</td>
</tr>
<tr>
<td>Cursor left</td>
<td>21</td>
<td>15</td>
<td>NACK</td>
<td>0</td>
</tr>
<tr>
<td>Cursor right</td>
<td>06</td>
<td>06</td>
<td>ACK</td>
<td>0</td>
</tr>
<tr>
<td>Cursor down</td>
<td>10</td>
<td>0A</td>
<td>LF</td>
<td>0</td>
</tr>
<tr>
<td>Cursor up</td>
<td>26</td>
<td>1A</td>
<td>SUB</td>
<td>0</td>
</tr>
<tr>
<td>Carriage return</td>
<td>13</td>
<td>0D</td>
<td>CR</td>
<td>0</td>
</tr>
<tr>
<td>Carriage return+line feed</td>
<td>29</td>
<td>1D</td>
<td>GS</td>
<td>0</td>
</tr>
<tr>
<td>Alphanumeric cursor absolute position</td>
<td>27 89 r c</td>
<td>1B 59 r c</td>
<td>ESC Y ASCII(r) ASCII(c)</td>
<td>0</td>
</tr>
<tr>
<td>Back space</td>
<td>08</td>
<td>08</td>
<td>BS</td>
<td>0</td>
</tr>
<tr>
<td>Clear page</td>
<td>12</td>
<td>0C</td>
<td>FF</td>
<td>0</td>
</tr>
<tr>
<td>Clear line</td>
<td>25</td>
<td>19</td>
<td>EM</td>
<td>0</td>
</tr>
<tr>
<td>Clear end of line</td>
<td>27 75</td>
<td>1B 4B</td>
<td>ESC K</td>
<td>0</td>
</tr>
<tr>
<td>Clear end of page</td>
<td>27 107</td>
<td>1B 6B</td>
<td>ESC k</td>
<td>0</td>
</tr>
<tr>
<td>Cursor off</td>
<td>27 80</td>
<td>1B 50</td>
<td>ESC P</td>
<td>0</td>
</tr>
<tr>
<td>Steady cursor on</td>
<td>27 79</td>
<td>1B 4F</td>
<td>ESC O</td>
<td>0</td>
</tr>
<tr>
<td>Blinkling block cursor on</td>
<td>27 81</td>
<td>1B 51</td>
<td>ESC Q</td>
<td>0</td>
</tr>
<tr>
<td>Reading of version number</td>
<td>27 86</td>
<td>1B 56</td>
<td>ESC V</td>
<td>3</td>
</tr>
<tr>
<td>Reading card code</td>
<td>27 160</td>
<td>1B A0</td>
<td>ESC ASCII(160)</td>
<td>1</td>
</tr>
<tr>
<td>Operating mode selection</td>
<td>27 65 mode</td>
<td>1B 41 mode</td>
<td>ESC A ASCII(mode)</td>
<td>0</td>
</tr>
<tr>
<td>Fluorescent display brightness setting</td>
<td>27 108 lum</td>
<td>1B 6C lum</td>
<td>ESC 1 ASCII(lum)</td>
<td>0</td>
</tr>
<tr>
<td>Beep</td>
<td>07</td>
<td>07</td>
<td>BEL</td>
<td>0</td>
</tr>
<tr>
<td>LEDs, Buzzer, Backlite activation</td>
<td>27 50 dev attr</td>
<td>1B 32 dev attr</td>
<td>ESC 2 ASCII(dev) ASCII(attr)</td>
<td>0</td>
</tr>
</tbody>
</table>

FIGURE A1: COMMAND CODES SUMMARY TABLE (1 OF 4)
<table>
<thead>
<tr>
<th>COMMAND</th>
<th>CODE</th>
<th>HEX CODE</th>
<th>MNEMONIC</th>
<th>Ret.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request of EEPROM availability</td>
<td>27 51</td>
<td>1B 33</td>
<td>ESC 3</td>
<td>1</td>
</tr>
<tr>
<td>Writing presence byte</td>
<td>27 33</td>
<td>1B 21 4E</td>
<td>ESC ! N ASCII(byte)</td>
<td>0</td>
</tr>
<tr>
<td>Reading presence byte</td>
<td>27 33 110</td>
<td>1B 21 6E</td>
<td>ESC ! n</td>
<td>1</td>
</tr>
<tr>
<td>Write byte on EEPROM</td>
<td>27 164 addl addh byte</td>
<td>1B A4 addl addh byte</td>
<td>ESC ASCII(164) ASCII(addl) ASCII(addh) ASCII(byte)</td>
<td>0</td>
</tr>
<tr>
<td>Read byte from EEPROM</td>
<td>27 165 addl addh</td>
<td>1B A5 addl addh</td>
<td>ESC ASCII(165) ASCII(addl) ASCII(addh)</td>
<td>1</td>
</tr>
<tr>
<td>Key code reconfiguration</td>
<td>27 55 key n. cod.</td>
<td>1B 37 key n. cod.</td>
<td>ESC 7 ASCII(key n.) ASCII(cod.)</td>
<td>0</td>
</tr>
<tr>
<td>Keyclick on without memorization</td>
<td>27 53</td>
<td>1B 35</td>
<td>ESC 5</td>
<td>0</td>
</tr>
<tr>
<td>Keyclick off without memorization</td>
<td>27 54</td>
<td>1B 36</td>
<td>ESC 6</td>
<td>0</td>
</tr>
<tr>
<td>Keyclick on with memorization</td>
<td>27 33 53</td>
<td>1B 21 35</td>
<td>ESC ! 5</td>
<td>0</td>
</tr>
<tr>
<td>Keyclick off with memorization</td>
<td>27 33 54</td>
<td>1B 21 36</td>
<td>ESC ! 6</td>
<td>0</td>
</tr>
<tr>
<td>Definition of user character</td>
<td>27 66 nchar Pat0÷Pat7</td>
<td>1B 42 nchar Pat0÷Pat7</td>
<td>ESC B ASCII(nchar) ASCII(Pat0)÷ASCII(Pat7)</td>
<td>0</td>
</tr>
<tr>
<td>Definition and memorization of user character</td>
<td>27 33 66 nchar Pat0÷Pat7</td>
<td>1B 21 42 nchar Pat0÷Pat7</td>
<td>ESC ! B ASCII(nchar) ASCII(Pat0)÷ASCII(Pat7)</td>
<td>0</td>
</tr>
<tr>
<td>Write all digital outputs</td>
<td>27 166 out</td>
<td>1B A6 out</td>
<td>ESC ASCII(166) ASCII(out)</td>
<td>0</td>
</tr>
<tr>
<td>Enable single digital output</td>
<td>27 168 bit</td>
<td>1B A8 bit</td>
<td>ESC ASCII(168) ASCII(bit)</td>
<td>0</td>
</tr>
<tr>
<td>Disable single digital output</td>
<td>27 170 bit</td>
<td>1B AA bit</td>
<td>ESC ASCII(170) ASCII(bit)</td>
<td>0</td>
</tr>
</tbody>
</table>

_Figure A2: Command codes summary table (2 of 4)_
<table>
<thead>
<tr>
<th>COMMAND</th>
<th>CODE</th>
<th>HEX CODE</th>
<th>MNEMONIC</th>
<th>Ret.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reading of max message number</td>
<td>27 110</td>
<td>1B 6E</td>
<td>ESC n</td>
<td>1</td>
</tr>
<tr>
<td>Reading of last group and message managed</td>
<td>27 33 109</td>
<td>1B 21 6D</td>
<td>ESC ! m</td>
<td>2</td>
</tr>
<tr>
<td>Select current message group</td>
<td>27 33 77 grp</td>
<td>1B 21 4D grp</td>
<td>ESC ! M grp</td>
<td>0</td>
</tr>
<tr>
<td>Message storage</td>
<td>27 33 67 mess.n. chr.0÷chr.19</td>
<td>1B 21 43 mess.n. chr.0÷chr.13</td>
<td>ESC ! C ASCII(mess.n.) ASCII(chr.0)÷ASCII(chr.19)</td>
<td>0</td>
</tr>
<tr>
<td>Message reading</td>
<td>27 33 69 mess.n.</td>
<td>1B 21 45 mess.n.</td>
<td>ESC ! E ASCII(mess.n.)</td>
<td>20</td>
</tr>
<tr>
<td>Visualization of messages</td>
<td>27 33 68 mess.n. n</td>
<td>1B 21 44 mess.n. n</td>
<td>ESC ! D ASCII(mess.n.) ASCII(n)</td>
<td>0</td>
</tr>
<tr>
<td>Scrolling messages visualization</td>
<td>27 33 83 mess.n. n.chr</td>
<td>1B 21 53 mess.n. n.chr</td>
<td>ESC ! S ASCII(mess.n.) ASCII(n.chr)</td>
<td>0</td>
</tr>
<tr>
<td>Set automatic visualization</td>
<td>27 150 255 mess.n. len shift r c</td>
<td>1B 96 FF mess.n. len shift r c</td>
<td>ESC ASCII(150) ASCII(255) ASCII(mess.n.) ASCII(len) ASCII(shift) ASCII(r) ASCII(c)</td>
<td>0</td>
</tr>
<tr>
<td>Start I2CBUS</td>
<td>27 250</td>
<td>1B FA</td>
<td>ESC ASCII(250)</td>
<td>0</td>
</tr>
<tr>
<td>Stop I2CBUS</td>
<td>27 251</td>
<td>1B FB</td>
<td>ESC ASCII(251)</td>
<td>0</td>
</tr>
<tr>
<td>Transmit byte on I2CBUS</td>
<td>27 252</td>
<td>1B FC byte</td>
<td>ESC ASCII(252) ASCII(byte)</td>
<td>1</td>
</tr>
<tr>
<td>Receive byte from I2CBUS</td>
<td>27 253 ack</td>
<td>1B FD ack</td>
<td>ESC ASCII(253) ASCII(ack)</td>
<td>1</td>
</tr>
</tbody>
</table>

**FIGURE A3: COMMAND CODES SUMMARY TABLE (3 OF 4)**
<table>
<thead>
<tr>
<th>COMMAND</th>
<th>CODE</th>
<th>HEX CODE</th>
<th>MNEMONIC</th>
<th>Ret.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write byte on backed SRAM</td>
<td>27 33 71</td>
<td>1B 21 47</td>
<td>ESC ! G ASCII(addr) ASCII(byte)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>addr byte</td>
<td>addr byte</td>
<td>ASCII(addr) ASCII(byte)</td>
<td></td>
</tr>
<tr>
<td>Read byte from backed SRAM</td>
<td>27 33 103</td>
<td>1B 21 67</td>
<td>ESC ! g ASCII(addr)</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>addr</td>
<td>addr</td>
<td>ASCII(addr)</td>
<td></td>
</tr>
<tr>
<td>Set clock</td>
<td>27 33 70</td>
<td>1B 21 46</td>
<td>ESC ! F ASCII(hou) ASCII(min) ASCII(sec) ASCII(day) ASCII(mon) ASCII(wea) ASCII(yea) ASCII(wee)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>hou min sec day</td>
<td>hou min sec day</td>
<td>ASCII(hou) ASCII(min) ASCII(sec) ASCII(day) ASCII(mon) ASCII(wea) ASCII(yea) ASCII(wee)</td>
<td></td>
</tr>
<tr>
<td>Acquire clock</td>
<td>27 33 102</td>
<td>1B 21 66</td>
<td>ESC ! f</td>
<td>7</td>
</tr>
<tr>
<td>Show time on display</td>
<td>27 33 116</td>
<td>1B 21 74</td>
<td>ESC ! t ASCII(r) ASCII(c) ASCII(frm)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>r c frm</td>
<td>r c frm</td>
<td>ASCII(r) ASCII(c) ASCII(frm)</td>
<td></td>
</tr>
<tr>
<td>Show data on display</td>
<td>27 33 100</td>
<td>1B 21 64</td>
<td>ESC ! d ASCII(r) ASCII(c) ASCII(frm)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>r c frm</td>
<td>r c frm</td>
<td>ASCII(r) ASCII(c) ASCII(frm)</td>
<td></td>
</tr>
<tr>
<td>Set clock alarm</td>
<td>27 33 65</td>
<td>1B 21 41</td>
<td>ESC ! A ASCII(hou) ASCII(min) ASCII(sec) ASCII(day) ASCII(mon) ASCII(dton)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>hou min sec day</td>
<td>hou min sec day</td>
<td>ASCII(hou) ASCII(min) ASCII(sec) ASCII(day) ASCII(mon) ASCII(dton)</td>
<td></td>
</tr>
<tr>
<td>Acquire clock alarm</td>
<td>27 33 97</td>
<td>1B 21 61</td>
<td>ESC ! a</td>
<td>6</td>
</tr>
</tbody>
</table>

**FIGURE A4: COMMAND CODES SUMMARY TABLE (4 OF 4)**
APPENDIX B: DISPLAY CHARACTERS

The following tables show the characters sets displayed on QTP 16Big for all the possible received codes, according with ordered display, and so model, and according with functionality mode preselected through proper commands. Even the not ASCII characters (or special characters) change when the display type changes; if the user requires a characters set different from those described in the following figures, he can directly contact grifo®.

<table>
<thead>
<tr>
<th>L</th>
<th>00</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
<th>A0</th>
<th>B0</th>
<th>C0</th>
<th>D0</th>
<th>E0</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>User chr 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01</td>
<td>User chr 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>User chr 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>User chr 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>User chr 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>05</td>
<td>User chr 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>06</td>
<td>User chr 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>07</td>
<td>User chr 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>08</td>
<td>User chr 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>09</td>
<td>User chr 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0A</td>
<td>User chr 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0B</td>
<td>User chr 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0C</td>
<td>User chr 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0D</td>
<td>User chr 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0E</td>
<td>User chr 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0F</td>
<td>User chr 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**FIGURE B1: CHARACTERS TABLE OF QTP 16Big-F4**
<table>
<thead>
<tr>
<th>D3 D2 D1 D0</th>
<th>0000 0</th>
<th>0001 1</th>
<th>0010 2</th>
<th>0011 3</th>
<th>0100 4</th>
<th>0101 5</th>
<th>0110 6</th>
<th>0111 7</th>
<th>1000 8</th>
<th>1001 9</th>
<th>1010 A</th>
<th>1011 B</th>
<th>1100 C</th>
<th>1101 D</th>
<th>1110 E</th>
<th>1111 F</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7 D6 D5 D4</td>
<td>0000 000000000000111111111111</td>
<td>0001 0000011111000001111111</td>
<td>0010 00100110100110110111</td>
<td>0011 0101010101010101</td>
<td>0100 0110011001100110</td>
<td>0101 0111011101110111</td>
<td>0110 0111011101110111</td>
<td>0111 0111011101110111</td>
<td>1000 0000011111111111111111</td>
<td>1001 0000011111111111111111</td>
<td>1010 0000011111111111111111</td>
<td>1011 0000011111111111111111</td>
<td>1100 0000011111111111111111</td>
<td>1101 0000011111111111111111</td>
<td>1110 0000011111111111111111</td>
<td>1111 0000011111111111111111</td>
</tr>
<tr>
<td>User chr 0</td>
<td>00P'P</td>
<td>1AQaa</td>
<td>2BRbr</td>
<td>3CScs</td>
<td>4DTdt</td>
<td>5EUeu</td>
<td>6FVfv</td>
<td>7GWgw</td>
<td>8HXhx</td>
<td>9IYiy</td>
<td>JZjz</td>
<td>KKKk</td>
<td>LhIl</td>
<td>MInn</td>
<td>NOoO</td>
<td></td>
</tr>
<tr>
<td>User chr 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User chr 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User chr 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User chr 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User chr 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User chr 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User chr 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure B2: Characters Table of QTP 16Big-C4**
**APPENDIX C: MOUNTING NOTES**

**QTP 16Big** is provided complete of metallic container and some additional parts that simplify the mounting. Inside this appendix there are all the information concerning this operations, together with the instructions to open the container and to personalize it.

**TERMINAL DIMENSIONS**

In the following figure there are dimensions of terminal **QTP 16Big** complete of external metallic container, attached frontal plastic frame and mounting clamps. Dimensions are in mm and the drawing is in scale.

![Figure C1: QTP 16Big Dimensions](image)

The dimensions of previous figure refer to container only, but occupied area can be slightly greater by considering also mounting clamps and screws, described in following figures, up to a maximum of 204 x 96 x 86 mm (W x H x D).
**Figure C2: Mounting Clamp Dimensions**

**Figure C3: QTP 16Big + Mounting Clamp View**
FRONT PANEL MOUNTING

The provided mounting mode is the front panel one that is possible on any panel with 10 mm maximum thickness and fixing is done by two clamps provided with QTP 16Big. Installation operations are extremly easy and they are below summarized:

1) make a rectangular breaking on mounting panel like those in the following figure;

![Figure C4: Breaking for installation](image)

2) screw the two screws on the two clamps, keeping the sharpened part close to the screw cut hole of clamp;

3) insert QTP 16Big in the breaking made at point 1;

4) dock the two clamps prepared at point 2 to the specific side breakings of QTP 16Big container, taking care that the first hook of the clamp, the one near screw-cut hole, enters correctly in the proper lateral buttonhole of the container (figure C3 shows the result of these instructions);

5) screw the screws of the two clamps until the QTP 16Big container is firmly docked to mounting panel;

6) insert the connettors on the back side.
PERSONALIZATION LABEL INSERTION

Frontal of QTP 16Big is provided with a pocket where the user can insert a personalization label with his logo, an identification code, the terminal function, or anything else.

If the label is required please insert it before mounting QTP. Label must be thin but rather rigid, for example made of 160 g/m² paper or polyester or polycarbonate sheets. Here follow the suggested dimensions, in millimeters, of personalization label; please note that the white zone is the area contained in the transparent window, or in other words, the visible part:

![Figure C5: Personalization label dimensions](image)

Here follow the operations required to insert personalization label inside the QTP 16Big:

1) Unscrew the two black screws on frontal panel (if present).
2) Separate the group metallic carter + plastic frame from the group front panel + printed circuit. A simple pressure on QTP 16Big connectors, or on the printed circuit always from the backside connectors window, it is sufficient.
3) Now the front panel is ready to insert the personalization label; this latter must be inserted from the bottom side, using the specific pocket located on the back of front panel, as shown in following figure. As described on figure C5, length of label must be greater than height of window to simplify the insertion and extraction.
4) Remount terminal QTP 16Big, following the previous steps in reversed order.

![Figure C6: Personalization label insertion](image)
**FIXING FRONT PANEL TO CONTAINER**

**QTP 16Big** by default is provided with front panel (keyboard+printed circuit board) jointed in plastic frame of the back metallic container. Terminal anyway allows a better mechanical docking between this two groups, by using two specific screws; this avoid accidental separations of front panel.

Here follows the operations that must be performed, to ensure such docking:

1) Separate the group metallic carter + plastic frame from the group front panel + printed circuit. A simple pressure on backside **QTP 16Big** connectors, or on the printed circuit always from the backside connectors window, is normally sufficient.

2) Of the six screws, that dock the plastic frame to the back metallic container, unscrew the two central ones.

3) On front panel, in correspondence with these central screws, there are two holes provided with flare, visible only from the back side. It is sufficient to perforate the polyester layer that covers the frontal, to make these holes accessible.

4) Remount everything, using the same screws removed at point 2, that will be screwed on the front panel with keyboard and not on the plastic frame any more.

**AUXILIARY FIXING SCREWS**

**FIGURE C7: SCREWS FOR FRONT PANEL FIXING**
APPENDIX D: VIEW AREA AND CHARACTERS DIMENSIONS

The following figures report the dimensions expressed in mm and in scale of the two display model used on QTP 16Big. Whenever a larger visible area and/or a greater number of characters are necessary, please remind that are available many other QTP models and/or display types; in these conditions it is suggested to contact directly grifo®.

**FIGURE D1: DISPLAY DIMENSIONS OF QTP 16Big-C4**
FIGURE D2: DISPLAY DIMENSIONS OF QTP 16BIG-F4
APPENDIX E: DEFAULT CONFIG., OPTIONS, ACCESSORIES

In correspondence of the first purchase, or after a reparation, the QTP 16Big is supplied in its base configuration. The features of this configuration has been described many times in the manual (by using also the name default configuration) and in this appendix they are summarized, opportunely divided in the following tables.

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>DEFAULT SETTING</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMUNIC.</td>
<td>Norm.</td>
<td>Serial communication on asynchronous serial line in normal mode</td>
</tr>
<tr>
<td>BAUD RATE</td>
<td>19200</td>
<td>Communication speed on asynchronous serial line</td>
</tr>
<tr>
<td>BIT x CHR</td>
<td>8</td>
<td>Bits per character on asynchronous serial line</td>
</tr>
<tr>
<td>STOP BIT</td>
<td>1</td>
<td>Stop bits on asynchronous serial line</td>
</tr>
<tr>
<td>PARITY</td>
<td>None</td>
<td>Parity check on asynchronous serial line</td>
</tr>
<tr>
<td>KEY-CCLICK</td>
<td>ON</td>
<td>Keyclick enabled when keys are pressed</td>
</tr>
<tr>
<td>SLAVE ADD.</td>
<td>80H</td>
<td>QTP address for network communication</td>
</tr>
<tr>
<td>EE DATA</td>
<td>INIT</td>
<td>Data of base EEPROM, initialized</td>
</tr>
<tr>
<td>RL2 FUNCT</td>
<td>USER</td>
<td>NO OUT2 output (connected to RL2 relay) configured as user digital output</td>
</tr>
</tbody>
</table>

**Figure E1: Local setup default configuration**

The values listed in previous table can be modified through the local setup modality, as described with details in the homonimous paragraph.

<table>
<thead>
<tr>
<th>JUMPER</th>
<th>DEFAULT CONNECTION</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1</td>
<td>not connected</td>
<td>Selects the RUN modality</td>
</tr>
<tr>
<td>J3, J4</td>
<td>not connected</td>
<td>Do not connect termination and forcing circuitry to RS 422, RS 485 serial line.</td>
</tr>
<tr>
<td>J5</td>
<td>not connected</td>
<td>Does not connect 120 Ω termination resistor to CAN line.</td>
</tr>
<tr>
<td>J8</td>
<td>position 2-3</td>
<td>Configures serial line for RS 422 standard electric protocol</td>
</tr>
<tr>
<td>J9</td>
<td>position 1-2</td>
<td>Write protection of optional EEPROM not enabled.</td>
</tr>
<tr>
<td>J0</td>
<td>not connected</td>
<td>Management of RL2 relay not enabled.</td>
</tr>
<tr>
<td>J12</td>
<td>connected</td>
<td>On board battery BT1 connected to back up circuitry.</td>
</tr>
</tbody>
</table>

**Figure E2: Jumpers default configuration**
Please remind that the jumpers default configuration proposed is the one relative to base version of terminal, that is without any options.

During the order phase the user can add to **QTP 16Big**, the following features:

<table>
<thead>
<tr>
<th>OPTION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>.CAN</td>
<td>CAN communication line</td>
</tr>
<tr>
<td>.RS422</td>
<td>Asynchronous serial communication line in RS 422</td>
</tr>
<tr>
<td>.RS485</td>
<td>Asynchronous serial communication line in RS 485</td>
</tr>
<tr>
<td>.CLOOP</td>
<td>Asynchronous serial communication line in passive Current Loop</td>
</tr>
<tr>
<td>.EE128</td>
<td>Additional EEPROM with 16K Bytes size</td>
</tr>
<tr>
<td>.EE256</td>
<td>Additional EEPROM with 32K Bytes size</td>
</tr>
<tr>
<td>.EE512</td>
<td>Additional EEPROM with 64K Bytes size</td>
</tr>
<tr>
<td>.RTC</td>
<td>Section with Real Time Clock and SRAM backed by battery</td>
</tr>
<tr>
<td>.RELAY</td>
<td>Three digital outputs with relays</td>
</tr>
<tr>
<td>.5Vdc or .ALIM</td>
<td>Power supply voltage at +5 Vdc</td>
</tr>
</tbody>
</table>

**FIGURE E3: OPTIONS TABLE**

All these options are described in the paragraphs of the manual that illustrate the functionalities and the use of the same additional features. It is suggested to use the final alphabetical index, placed in following APPENDIX F, to find these paragraphs in a short time.

In addition there are a list of accessories that simplify and speed up the use of the module. Among these ones we remind the following available products:

- **AMP2.Cable**  complete connector with 2 coloured wires, 1 metre length;
- **CKS.AMP2**  kit composed by female AMP Mod II 2 pins, plus 2 contacts to crimp;

![Figure E5: CKS.AMP2 Connection Accessory](image)

These components can be acquired directly from AMP dealers by using P/N 280358 and P/N 182206-2.

- **AMP8.Cable**  complete connector with 4 coloured wires, 1 metre length;

![Figure E6: AMP4.Cable Connection Accessory](image)
- **CKS.AMP4** kit composed by female AMP Mod II 4 pins, plus 4 contacts to crimp;

![CKS.AMP4 Connection Accessory](image)

**Figure E7: CKS.AMP4 Connection Accessory**

These components can be acquired directly from AMP dealers by using P/N 280359 and P/N 182206-2.

- **EXPS-1** power supply for direct connection to mains voltage at 230 Vac, 50 Hz, that generates an output voltage of 24 Vdc, 300 mA compatible for **QTP 16Big**. The photo of this accessories is already available in previous pages of manual, on figure 34.
Symbols

+Vdc pow 16, 36
.5Vdc pow 36, E-2
.ALIM option 36, E-2
.CAN option 9, 18, E-2
.CLOP option 9, 32, E-2
.EEExxx option 7, 68, E-2
.RELAY option 10, 22, 82, E-2
.RS422 option 9, 32, E-2
.RS485 option 9, 34, E-2
.RTC option 10, 31, 76, E-2
9 bits 9, 39, 44

A

ABSOLUTE PLACEMENT OF ALPHANUMERIC CURSOR, command 55
Accessories 20, 28, 36, 52, E-2
ACQUIRE CLOCK ALARM, command 81
ACQUIRE CLOCK, command 77
Addressing 50
ADD Viewpoint 54, A-1
Alarm activation time 80
AMP2.Cable E-2
AMP4.Cable E-3
ASCII 42, B-1
Assistance 1
Autorepeat 7, 40

B

Back up 31
Backlight 8, 14, 60
BACKSPACE, command 56
BasicCAN 9
Battery 15, 31
Baud rate 12, 38, 45, 53, E-1
BEEP, command 60
Bit rate 12, 74
Bits x chr 12, 39, 53, E-1
BLINKING BLOCK CURSOR ON, command 57
Boot Loader 28, 30
Brightness 43, 58
Buffer 39
Buzzer 7, 60, 64
BUZZER, LEDS, BACKLIGHT ACTIVATION, command 60

C

CAN 9, 14, 18, 37, E-2
CAN termination 30
Card code 58
CARRIAGE RETURN, command 55
CARRIAGE RETURN+LINE FEED, command 55
CD rom 38, 52
Characters 12, B-1, D-1
Characters visualization 42
CKS.AMP2 E-3
CKS.AMP4 E-4
Clamps 13, C-1, C-3
CLEAR END OF LINE, command 56
CLEAR END OF PAGE, command 56
CLEAR LINE, command 56
CLEAR PAGE, command 56
Clock alarm 80, 81
Column 55, 72, 78, 79
Command mode 42, 59
Commands 54, A-1
Commands for characters erasure 56
Commands for cursor attributes 57
Commands for cursor position 54
Commands for EEPROM 62
Commands for general functions 58
Commands for I2C BUS communication as master 74
 Commands for keyboard 64
Commands for messages 68
Commands for relays outputs 82
Commands for SRAM and clock 76
Commands for user characters 66
Common 22
Communication 59
Electric protocol 30
I2C BUS 48
Logic protocol 38, 44, 48, 51
 Normal 51
Physic protocol 39, 53
Communication mode 44
COMMUNICATION RESET, command 59
Communication type 38
Components map 29
COMx 52, 53
Connectors 13, 15
 CN3 20
 CN4 16
 CN5 23
 CN6 18
 CN7 22
 J1 28
Contacts 22
Container 1, C-1, C-5
Contrast 43
CPU 11
Current Loop 9, 23, 26, 32, E-2
Current Loop network 27
Cursor 54, 57
CURSOR DOWN, command 54
CURSOR LEFT, command 54
CURSOR OFF, command 57
CURSOR RIGHT, command 54
CURSOR UP, command 55
D
Data endurance 40
Default configuration 12, 30, 39, 68, E-1
DEFINITION AND MEMORIZATION OF USER CHARACTER, command 67
DEFINITION OF USER CHARACTER, command 67
Delay 39, 45, 67, 71, 78, 79
Demo programs 52, 53
Dimensions 12, C-1, D-1
Directive 1, 23
DISABLE SINGLE DIGITAL OUTPUT, command 84
Display 8, 11, 14, B-1, D-1
Distance 37
DLL libraries 53
Documentation 1

E
EEPROM 7, 11, 38, 40, 68, 73, E-2
EEPROM for general use 63
Electrostatic noises 1
ENABLE SINGLE DIGITAL OUTPUT, command 82
ESC 54
ESD 1
EXPS-1 36, 37, E-4
Extra voltages 14, 36

F
Firmware 3, 48, 51, 58, A-1
First purchase 53
Flow charts 48, 49, 51
Flow control 53
FLUORESCENT DISPLAY BRIGHTNESS SETTING, command 58
Front panel C-3
Front panel fixing C-5

G
General information 4
GENERAL RESET, command 59
GND 16, 36
Ground 37

H
Handshake 53
HOME, command 55
Humidity 13
HYPERTERMINAL 53

I
I2C BUS 12, 20, 38, 48, 74
I2C BUS communication 44, 48
Identification address 38, 44
Impedance 14
Initialization 59
Installation 15
Intermittent attribute 60
Introduction 1
IP54 11

J
Jumpers 15, 30, E-1

K
KEY RECONFIGURATION, command 64
Keyboard 7, 40
Keyclick 7, 38, E-1
keyclick 40, 64
KEYCLICK OFF WITH MEMORIZATION, command 65
KEYCLICK OFF WITHOUT MEMORIZATION, command 64
KEYCLICK ON WITH MEMORIZATION, command 65
KEYCLICK ON WITHOUT MEMORIZATION, command 64
Keys 7, 38, 40, 43

L
Label 7, C-4
LEDs 60, 61
Library 10, 53
License 53
Loads 22
Local setup 38

M
Malfunctions 53
Master 44, 48, 51
Master-Slave 9 bits communication 44
Master-Slave 9 bits example 47
Membrane 7
MESSAGE READING, command 70
MESSAGE STORAGE, command 69
Messages 12, 40, 68, 72
Mounting 13, C-1, C-3
Mounting breaking C-3

N
Network 12, 19, 21, 25, 27, 44, 49, 50
Noisy 37
Normal communication 44, 51
Normally open 22
Normative 21, 48

O
Operating mode 59
OPERATING MODE SELECTION, command 59
Outline 12
Overflow 67, 71, 78, 79

P
Parity 12, 39, 45, 53, E-1
Patterns 40, 66
PC 53
PC connection 52
PeliCAN 9
Personalization C-4
Phases 36
Physic protocol 12, 39, 53
Pixels 66, D-1
Pocket C-4
Polarity 36
Power on 11, 59, 67
Power on visualization 73
Power supply 8, 14, 16, 36, E-4
Precision 11
Presence byte 40, 62
Protection 1, 11, 36
Protocols 12, 44
PROQTP16B.* 52
Pull up resistors 21

Q
QTP EDIT 68

R
READ BYTE FROM BACKED SRAM, command 76
READ BYTE FROM EEPROM, command 63
READ CARD CODE, command 58
Read data 49
READ FIRMWARE VERSION, command 58
READ PRESENCE BYTE, command 62
READING OF MAX MESSAGE NUMBER, command 68
Receive buffer 12, 39
Relays 22, 34, 82, E-2
Remarks 53
Representation mode 42, 59
REQUEST FOR EEPROM AVAILABILITY, command 62
Reset 59
Resources 11
RL2 34, 80, 82, E-1
Row 55, 72, 78, 79
RS 232 9, 23, 32, 52
RS 422 9, 14, 23, 30, 32, E-2
RS 422-485 Termination 14
RS 485 9, 23, 25, 30, 34, E-2
RTC 76, E-2
RTC parameters 77
Rules 1
S
Safety 1
Screws C-5
Scrolling 71, 72
SCROLLING MESSAGES VISUALIZATION, command 71
Serial line 32, 52
SET CLOCK ALARM, command 80
SET CLOCK, command 77
Setup 38
SHOW DATE ON DISPLAY, command 79
SHOW TIME ON DISPLAY, command 78
Size 12, C-1, D-1
Slave 44
Slave Address 38, 49, 50, 74, E-1
Sound 60
Special characters 42, B-1
STEADY STATIC CURSOR ON, command 57
Stop bit 38, 53, E-1
Syncronization 48, 51

T
Temperature 13
Terminal emulation 52
Termination 19, 25, 30
Time out 45
Timing 11, 45
Trademarks 2
Transmission time 45
Transmit buffer 12, 39
TransZorb™ 36

U
Underline 66
User characters 40, 43, 66, B-1
User EEPROM 12, 40, 63

V
Vac 16, 36
Version 3, 58
VISUALIZATION OF MESSAGES, command 70

W
Warranty 1, 2
Weight 13
WRITE ALL DIGITAL OUTPUTS, command 82
WRITE BYTE ON BACKED SRAM, command 76
Write data 49
WRITE OF PRESENCE BYTE, command 62