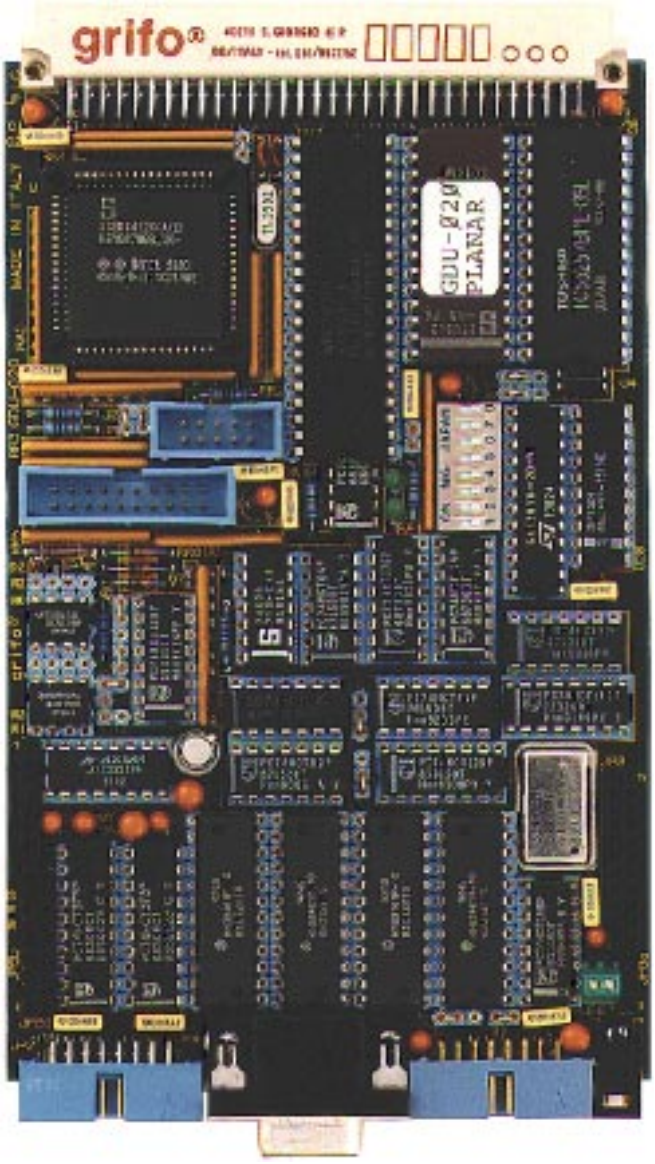
# GDU 020

## Graphic Display Unit

USER MANUAL




grifo®
ITALIAN TECHNOLOGY

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

**GDU 020** Edition 5.10 *Rel. 12 July 2001*

# GDU 020

## Graphic Display Unit

USER MANUAL

Graphic terminal based on GDC µ**P72020**, driven through controller PHILIPS **84c451**; management through 8 bit BUS or serial line buffered in RS 232, RS 422, RS 485 or current loop; connectior for 8 LEDs; capable to acquire a P.C. AT keyboard or a 8x8 keys matrix keyboard; interfaceable to VGA monitor,

grifo®
ITALIAN TECHNOLOGY

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

**GDU 020** Edition 5.10 *Rel. 12 July 2001*

## DOCUMENTATION COPYRIGHT BY grifo® , ALL RIGHTS RESERVED



## IMPORTANT

Although all the information contained herein have been carefully verified, **grifo**® assumes no responsability 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

## Trade Marks

abaco®, **GPC**®, **grifo**® : are trade marks of **grifo**®.
Other Product and Company names listed, are trade marks of their respective companies.

# GENERAL INDEX

# FIGURES INDEX

# INTRODUCTION

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

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 etc. installation are destined - and then executable - always and in exclusive way from specialized warned and educated personnel, or directly from the TECHNICAL AUTHORIZED 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 rispect 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, respectly at the begining and at the end of the manual, to find information in a faster and more easy way.

# CARD VERSION

The present handbook is reported to the **GDU 020** card release **101194** and later. The validity of the bring informations is subordinate to the number of the card release. The user must always verify the correct correspondence among the two denotations. On the card the release number is present in more points both board printed diagram (serigraph) and printed circuit (for example near connector JP8b in the bottom left corner of the card both on the component side and on the solder side).

# GENERAL INFORMATION

**GDU 020** card is a complete and efficent intelligent **Video-Graphic Terminal** based on microprocessor 80c451, therefore the card is capable to work both as a slave peripheral device on **ABACO®** BUS taking as low as two bytes of addressing space and as autonomous device using the serial line as interface to the external world.
The card offers several features that make it the ideal component to build an user interface.
Overall features of **GDU 020** are as follows:

- Eurocard format size 100x160 mm
- Interface to **ABACO®** industrial BUS
- I/O mapping through on board **dip switch**
- Only as low as **2 bytes** taken
- Serial communication line can be RS 232, RS 422, RS 485 or Current Loop
- Graphic controller CMOS **µPD 72020** on board
- CPU **80c451** featuring own ROM and RAM
- **Serial EEPROM** for card configuration
- Two buffers for reception and transmission
- High level communication protocol both serial and parallel
- On board **VRAM** of 128K Bytes for **B/W** expandable to 384K Bytes for **RGB**
- Graphic resolution up to **1024x1024** pixel with 3 color layers
- Direct commands for graphic primitives like segments, circles, box, etc.
- Direct commands to delete characters, rows, screen, etc.
- Character generator EPROM, **8x12 character matrix**
- Mixed alphanumeric and graphic display
- **16** steps of character size **ZOOM** high level manageable
- Contemporary representation of characters having different sizes
- Possibility of absolute or relative cursor addressing
- Output for **CRT** with composite input or separated synchronism
- Direct driving of **PLANAR Graphic Fluorescent Display**
- Direct input for **BG keyboard** or **8x8 matrix keyboard** (optional)
- Input for standard **PC-AT keyboard**
- Direct driving of **8** signalation **LEDs**
- Unique power supply +5 Vdc, except for PLANAR display that needs +12 Vdc

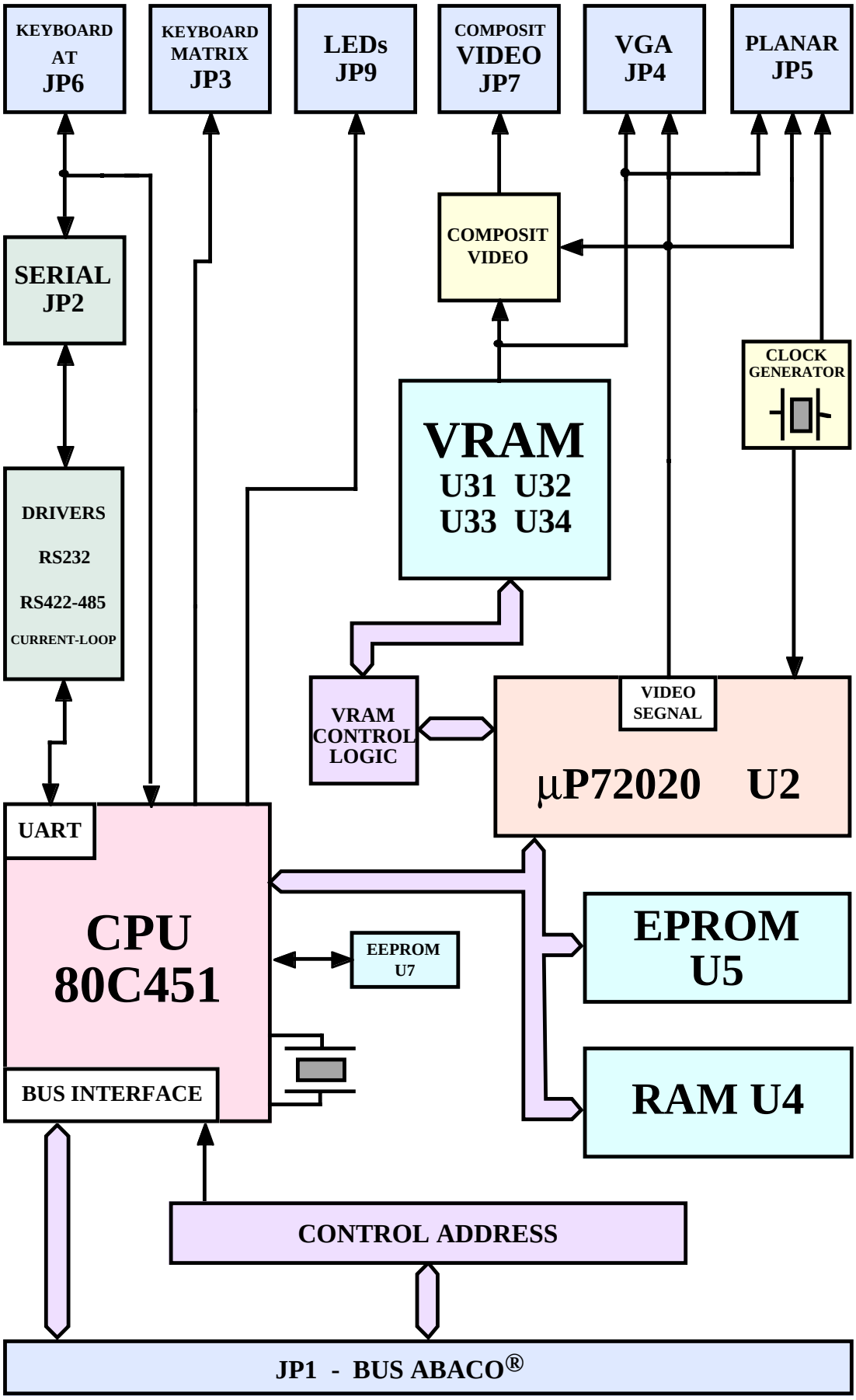



FIGURE 1: BLOCK DIAGRAM

Here follows a description of **GDU 020** board's functional blocks, with an indication of the operations performed by each one. To easily locate these blocks and verify their connections pleare refer to figure 1.

## INTERFACING AND ADDRESSING

This section manages the data exchange between contol logic and command board through **ABACO®** BUS. In particular, all written or read data transit across this section that, in addition, provides the board I/O management in a 256 addressing space, by setting the dip switch **SW1**.
For further information please refer to the chapter dedicated to board's software description.

## CONTROL LOGIC

This section generates all the chip select signals needed to acces the several peripherals on **GDU 020** boards. Using this section the programmer can interact to the board, verifing its status, setting configuration, etc.
All this can be done through a simple software management based on **ABACO®** BUS, to which the contol logic connects through the interfacing and addressing section. For further information please refer to chapter “PERIPHERAL DEVICES SOFTWARE DESCRIPTION”.

## CPU

**GDU 020** card is designed to accept the 8 bit family 51 microprocessor 80c451 manufactured by SINGETICS. The main feature of this CPU is to have an internal peripheral ideal to interface directly to a BUS, **ABACO®** industrial BUS in our implementation, allowing communication to the main CPU as if 80c451 was a peripheral.
Thanks to the high speed parallel communication and to the wide reception buffer **GDU 020** can receive commands as fast as 100,000 Bytes per second, allowing the main CPU to be free from any slowing condition that may occour when using a non intelligent video peripheral.
80c451 controls all the cards sections, making it capable to operate in total autonomy like a real terminal, managing the video section and the keyboard section by the operativity of a program stored in EPROM and basing on the working parametera set by the user and stored in EEPROM.

## GRAPHIC DISPLAY CONTROLLER

The **GDU 020** on board GCD is a NEC **μPD 72020**. Its main features are a high working speed and the ability to use **VRAM** type memories, specific for high speed graphic cards.
This device has all the resources needed to perfrom the operation to display graphics, circles, box, lines, etc. It is also capable to manage in autonomy 16 zoom levels for 8x12 pixel characters, allowing to build a very complex user interface in the most simple way.

## MEMORY DEVICES

The card is provided with seven memory devices, in detail:

| | |
|---|---|
| U5 | -> EPROM for management program and character configuration pattern |
| U4 | -> RAM for management program and buffer |
| U31, U32, U33, U34 | -> **VRAM** type RAM |
| U7 | -> Serial EEPROM to store configuration parameters |

Size of these devices cannot be changed and are selected according to card development considerations. Their management is performed completely by the card and the user in never involved in it.

## SERIAL COMMUNICATION

Serial communication to the external world is managed through the on board serial line. It is **asynchronous full duplex**, can be buffered as RS 232, RS 422, RS 485 or passive current loop line and must be used for communications with the terminal.
All the characters received from such line are interpreted and visualized, while all the characters acquired from the keyboard are sent to the line itself.
Communication protocol can be defined by software, in fact the user can inform **GDU 020** about the working modality during SETUP mode. Serial communication baud rate can vary from 2400 baud up to 19200 baud, but the number of data bits (8), of stop bits (1) and the parity (no parity) cannot be changed.
It is very important to remember that **GDU 020** can manage software protocol XON/XOFF to awarn that the serial buffer is almost full, so even if the hardware handshakes are not available the serial communication can be managed efficently, especially when using protocols like RS 422, RS 485 (optional) or current loop, by standard not provided with hardware handshake.

## CLOCK

**GDU 020** is provided with two indipendent oscillator circuits to generate the clock signals. One circuit generates the CPU clock, the other circuit is charged to control the monitor driving section (it defines the display format).
In fact according to the display type needed the frequence must change; for example a **VGA** monitor needs a 20 MHz pixel-rate, a **PLANAR** display needs 16 MHz pixel-rate and a **CRT** needs a 11 MHz pixel-rate.

## MONITOR INTERFACE

**GDU 020** is provided with a complete and efficent monitor interface section whose horizontal scansion frequence changes according to the kind of monitor required.
This section includes a GDC controller, RAM for video and all the control logic needed. The monitor interface section is completely managed through the on board firmware. There are three different monitor outputs; RGB for VGA connection, PLANAR Fluorescent Graphic Display and composite monitor.

## KEYBOARD INTERFACE

**GDU 020** is provided with two different kinds of keyboard interface; the first (optional) can accept a BG type 8x8 matrix keyboard and is capable to control an auxiliary LED and a buzzer, the other interface can accept a PC-AT keyboard.
When in run mode **GDU 020** manages only one of the two interfaces, the selection of which one to use is made during the setup phase.

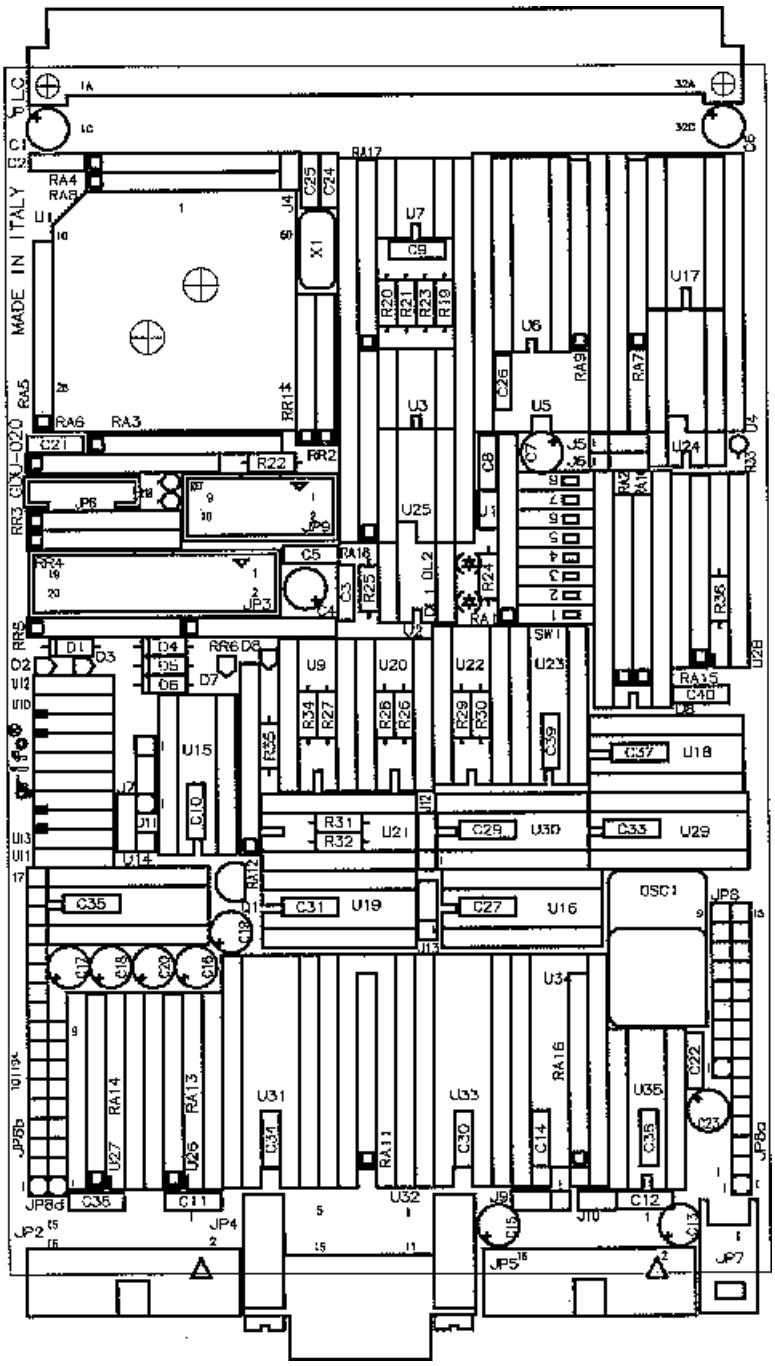



**FIGURE 2: COMPONENTS MAP**

# TECHNICAL FEATURES

## GENERAL FEATURES

| | |
|---|---|
| **BUS type:** | **ABACO® BUS** |
| **On board I/O:** | 8 outputs for LEDs<br>8x8 lines for a BG type matrix keyboard (optional)<br>2 lines for PC-AT keyboard<br>1 full duplex RS232 or RS 422 or RS 485 o current loop serial line<br>1 BAS type video composite signal<br>3 video signals + 2 synchronism signals for VGA monitor<br>1 TTL video signal + 2 synchronism signals for display PLANAR |
| **On board memory:** | U4: RAM 8K x 8 o 32K x 8<br>U5: EPROM 64K x 8 (27c512)<br>U7: serial EEPROM 24C02,04,16<br>U31,U32,U33,U34 : VRAM 64 K x 4 |
| **CPU:** | SIGNETICS 80c451 |

## PHYSICAL FEATURES

| | |
|---|---|
| **Size:** | Standard EUROCARD format 100x160 mm |
| **Weight:** | 220 g |
| **Connectors:** | JP1: 64 pin DIN 41612 Type C<br>JP2: 16 pins low profile 90° male<br>JP3: 20 pins low profile male<br>JP4: 1ç pins D type high density female<br>JP5: 16 pins low profile 90° male<br>JP6: 5 pins vertical male<br>JP7: 2 pins AMP 90° male<br>JP9: 10 pins low profile male |
| **Temperature range:** | from 0 to 70° C |
| **Relative humidity:** | 20% up to 90% (without condensing) |

## ELECTRIC FEATURES

| | |
|---|---|
| **Power supply:** | +5 Vdc ± 5%<br>+12 Vdc for PLANAR display |
| **Current consumption:** | 280 mA on +5 Vdc<br>2.0 A on +12 Vdc for PLANAR display |

INSTALLATION

In this chapter there are the information for a right installation and correct use of **GDU 020** card. The user can find the location and functions of each connectors, LEDs and some explanatory diagrams.

## CONNECTIONS

The board has six connectors that can be linkeded to other devices or directly to the field, according to system requirements. In this paragraph there are connectors pin outs, a short signals description (including the signals direction) and connectors location, plus some figures that describe how the interface signals are connected on the card. To easily locate the connectors please refer to figure 6.

### JP2 - SERIAL LINE AND PC-AT KEYBOARD CONNECTOR

The connector for serial line and PC-AT keyboard, called JP2, is a 16 pins low profile male 90 degreeses connector with 2.54 mm pitch.
The serial line available can be buffered as RS 232, RS 422, RS 485 or current loop. For the selection of which one to use please refer to “SERIAL COMMUNICATION SELECTION”.

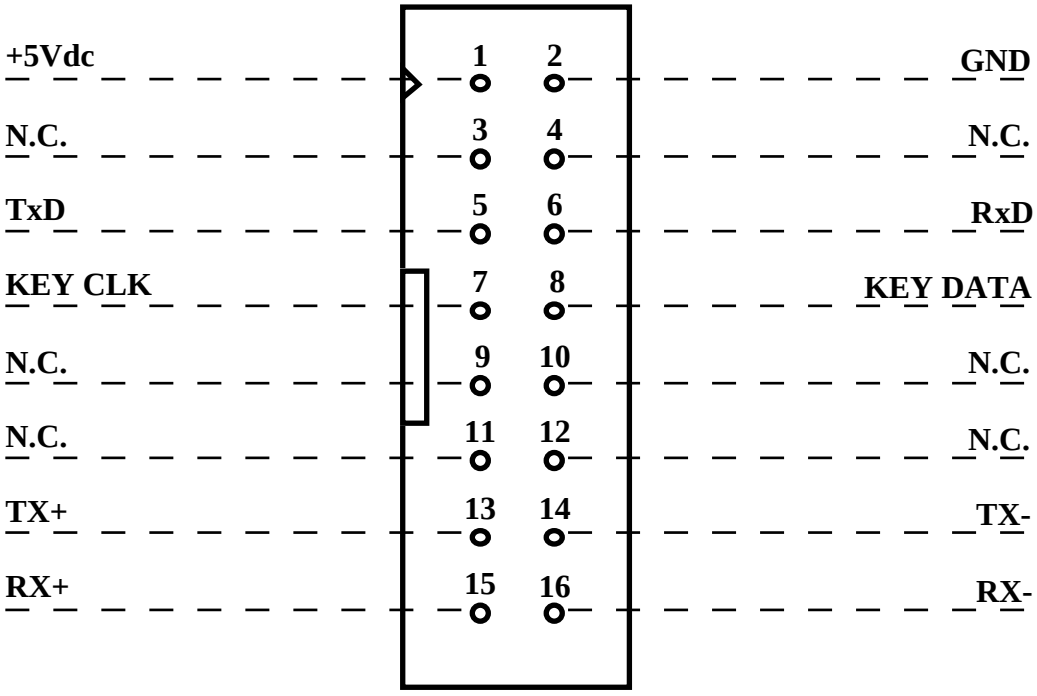



**FIGURE 3: JP2 - SERIAL LINE AND PC-AT KEYBOARD CONNECTOR**

Signals description:

**+5Vdc** = O - Power supply +5 Vdc.
**RxD** = I - RS 232 Receive Data.
**TxD** = O - RS 232 Trasmit Data.
**RX-** = I - RS 422 or RS 485 or current loop Receive Data Negative.
**RX+** = I - RS 422 or RS 485 or current loop Receive Data Positive.
**TX-** = O - RS 422 or RS 485 or current loop Transmit Data Negative.
**TX+** = O - RS 422 or RS 485 or current loop Transmit Data Positive.
**Key Data** = I/O - Data reception or transmission line for PC-AT keyboard.
**Key Clk** = I - Clock signal reception line for PC-AT keyboard.
**N.C.** = - Not connected.
**GND** = - Digital ground.

## JP3 - MATRIX KEYBOARD 8x8 CONNECTOR

The connector for 8x8 matrix keyboard, called JP3, is a 20 pins low profile male connector with 2.54 mm pitch. It allows to interface a BG type 8x8 matrix keyboard with open collector signals for a buzzer and a LED. Use of this keyboard is optional and is available only under request.

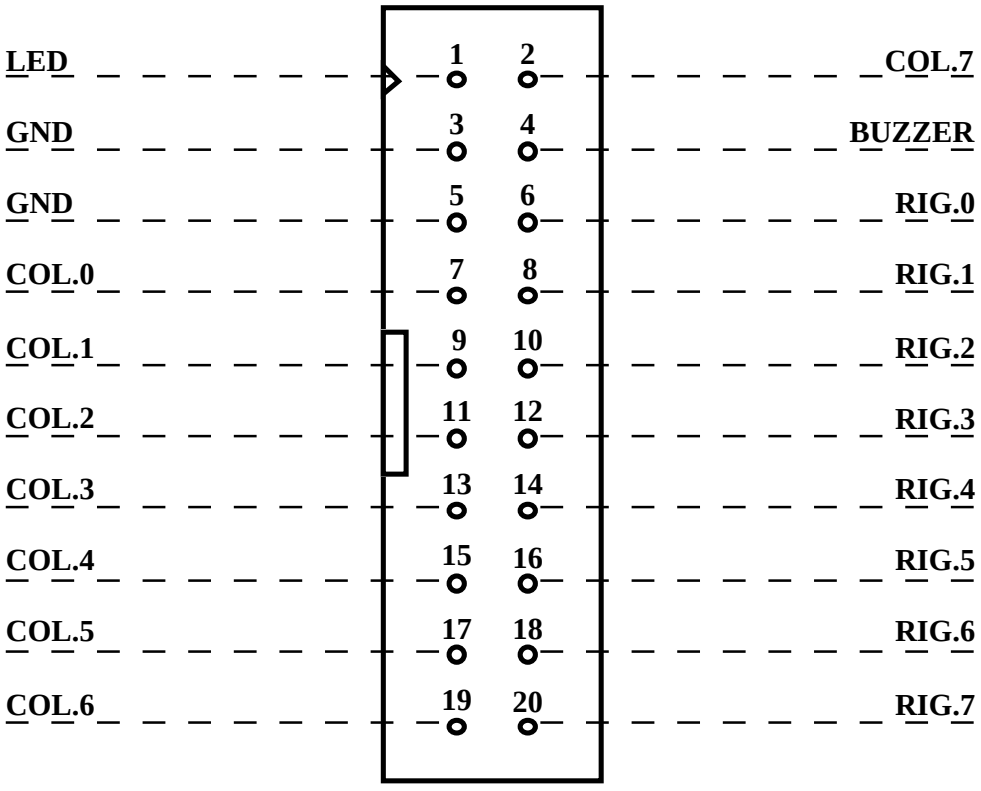



**FIGURE 4: JP3 - MATRIX KEYBOARD 8X8 CONNECTOR**

Signals description:

**COL.n** = O - n-th column driving signal for matrix keyboard.
**RIG.n** = I - n-th row acquisition signal for matrix keyboard.
**BUZZER** = O - External buzzer driving signal.
**LED** = I - External LED driving signal.
**GND** = - Digital ground.

## JP4 - VGA MONITOR CONNECTOR

The connector for VGA monitor, called JP4, is a 15 pins D type high density female connector. JP4 allows to connect **GDU 020** directly to a VGA monitor as all the needed signals are present.

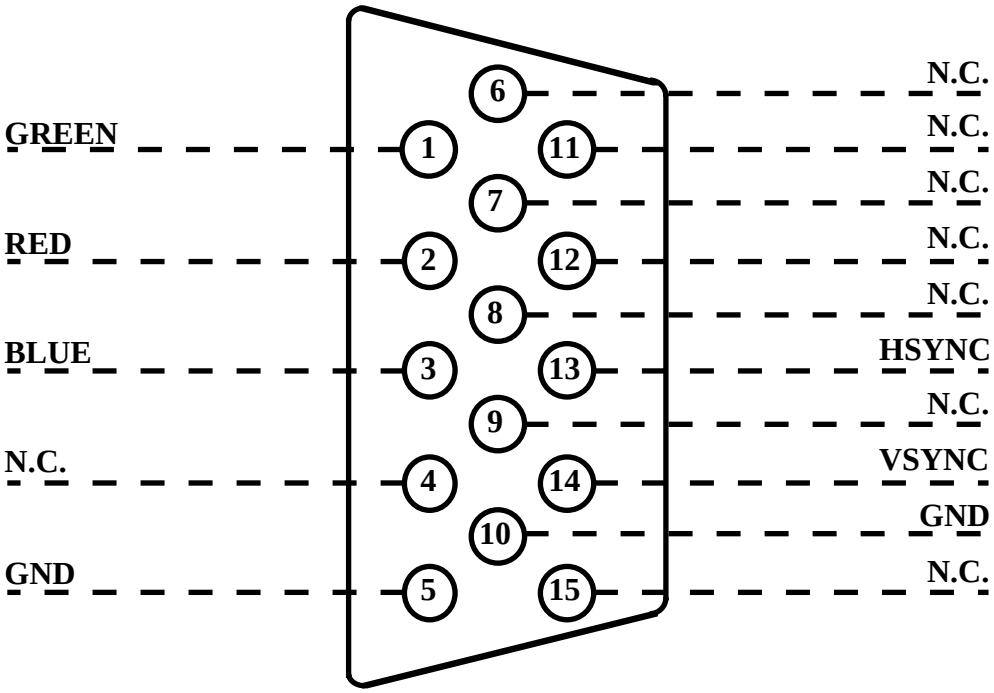



**FIGURE 5: JP4 - VGA MONITOR CONNECTOR**

Signals description:

**GREEN** = O - Green video signal.
**RED** = O - Red video signal.
**BLUE** = O - Blue video signal.
**HSYNC** = O - horizontal synchronism signal.
**VSYNC** = O - vertical synchronism signal.
**GND** = - Digital ground.
**N.C.** = - Not connected.

**NOTE:**
If **GDU 020** is configured for monochrome B/W, the only video signal available on this connector is GREEN.

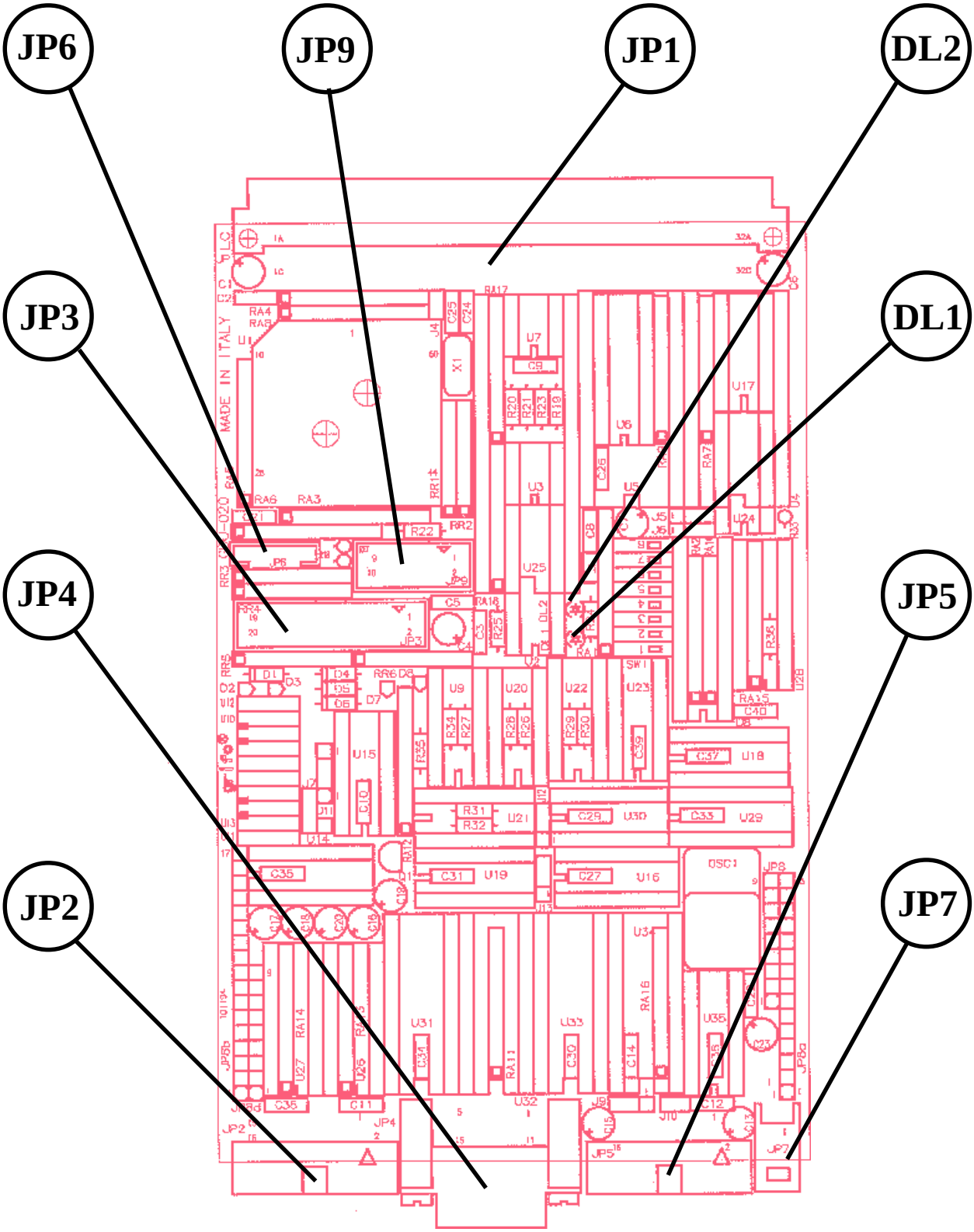



FIGURE 6: CONNECTORS AND LEDS LOCATION

## JP1 - ABACO® BUS CONNECTOR

The connector for **ABACO® industrial BUS**, called K1 on the board,is a DIN 41612, male, a 90 °, type C, A+C.
Here follows the pin-out of the connector installed on **GDU 020**, in addition there is the standard 8 bits and 16 bits**ABACO®** BUS pin-out.
Please remark that all the signals here described are TTL, except for the power supplies.

| A<br>16 bit BUS | A<br>8 bit BUS | A<br>GDU 020 | PIN | C<br>GDU 020 | C<br>8 bit BUS | C<br>16 bit BUS |
|---|---|---|---|---|---|---|
| GND | GND | GND | 1 | GND | GND | GND |
| +5 Vdc | +5 Vdc | +5 Vdc | 2 | +5 Vdc | +5 Vdc | +5 Vdc |
| D0 | D0 | D0 | 3 | N. C. | | D8 |
| D1 | D1 | D1 | 4 | N. C. | | D9 |
| D2 | D2 | D2 | 5 | N. C. | | D10 |
| D3 | D3 | D3 | 6 | N. C. | /INT | /INT |
| D4 | D4 | D4 | 7 | N. C. | /NMI | /NMI |
| D5 | D5 | D5 | 8 | N. C. | /HALT | D11 |
| D6 | D6 | D6 | 9 | N. C. | /MREQ | /MREQ |
| D7 | D7 | D7 | 10 | /IORQ | /IORQ | /IORQ |
| A0 | A0 | A0 | 11 | /RD | /RD | /RDLDS |
| A1 | A1 | A1 | 12 | /WR | /WR | /WRLDS |
| A2 | A2 | A2 | 13 | N. C. | /BUSAK | D12 |
| A3 | A3 | A3 | 14 | N. C. | /WAIT | /WAIT |
| A4 | A4 | A4 | 15 | N. C. | /BUSRQ | D13 |
| A5 | A5 | A5 | 16 | /RESET | /RESET | /RESET |
| A6 | A6 | A6 | 17 | /M1 | /M1 | /IACK |
| A7 | A7 | A7 | 18 | N. C. | /RFSH | D14 |
| A8 | A8 | N. C. | 19 | N. C. | /MEMDIS | /MEMDIS |
| A9 | A9 | N. C. | 20 | N. C. | VDUSEL | A22 |
| A10 | A10 | N. C. | 21 | N. C. | /IEI | D15 |
| A11 | A11 | N. C. | 22 | N. C. | | RISERVATO |
| A12 | A12 | N. C. | 23 | N. C. | CLK | CLK |
| A13 | A13 | N. C. | 24 | N. C. | | /RDUDS |
| A14 | A14 | N. C. | 25 | N. C. | | /WRUDS |
| A15 | A15 | N. C. | 26 | N. C. | | A21 |
| A16 | | N. C. | 27 | N. C. | | A20 |
| A17 | | N. C. | 28 | N. C. | | A19 |
| A18 | | N. C. | 29 | N. C. | /R.T. | /R.T. |
| +12 Vdc | +12 Vdc | +12 Vdc | 30 | N. C. | -12 Vdc | -12 Vdc |
| +5 Vdc | +5 Vdc | +5 Vdc | 31 | +5 Vdc | +5 Vdc | +5 Vdc |
| GND | GND | GND | 32 | GND | GND | GND |

**FIGURE 7: K1 - ABACO® BUS CONNECTOR**

Signals description:

8 bits CPU

| | | | |
|---|---|---|---|
| **A0-A15** | = | O | - Address BUS |
| **D0-D7** | = | I/O | - Data BUS |
| **/INT** | = | I | - Interrupt request |
| **/NMI** | = | I | - Non Maskable Interrupt |
| **/HALT** | = | O | - Halt state |
| **/MREQ** | = | O | - Memory Request |
| **/IORQ** | = | O | - Input Output Request |
| **/RD** | = | O | - Read cycle status |
| **/WR** | = | O | - Write cycle status |
| **/BUSAK** | = | O | - BUS Acknowledge |
| **/WAIT** | = | I | - Wait |
| **/BUSRQ** | = | I | - BUS Request |
| **/RESET** | = | O | - Reset |
| **/M1** | = | O | - Machine cycle one |
| **/RFSH** | = | O | - Refresh for dynamic RAM |
| **/MEMDIS** | = | I | - Memory Display |
| **VDUSEL** | = | O | - VDU Selection |
| **/IEI** | = | I | - Interrupt Enable Input |
| **CLK** | = | O | - System clock |
| **R.B.** | = | I | - Reset button |
| **+5 Vdc** | = | I | - Power supply at +5 Vdc |
| **+12 Vdc** | = | I | - Power supply at +12 Vdc |
| **-12 Vdc** | = | I | - Power supply at -12 Vdc |
| **GND** | = | | - Ground signal |

16 bits CPU

| | | | |
|---|---|---|---|
| **A16-A22** | = | O | - Address BUS |
| **D8-D15** | = | I/O | - Data BUS |
| **/RD UDS** | = | O | - Read Upper Data Strobe |
| **/WR UDS** | = | O | - Write Upper Data Strobe |
| **/IACK** | = | O | - Interrupt Acknowledge |
| **/RD LDS** | = | O | - Read Lower Data Strobe |
| **/WR LDS** | = | O | - Write Lower Data Strobe |

**NOTE**
Directionality indications as above stated are referred to a master (**GPC®**) board and have been kept untouched to avoid ambiguity in case of multi-boards systems.

## JP5 - PLANAR DISPLAY CONNECTOR

The connector for planar display, called JP5, is a 16 pins low profile 90° male connector with 2.54 mm pitch.
It allows to interface directly a PLANAR display because all the TTL signals needed are available on the connector.

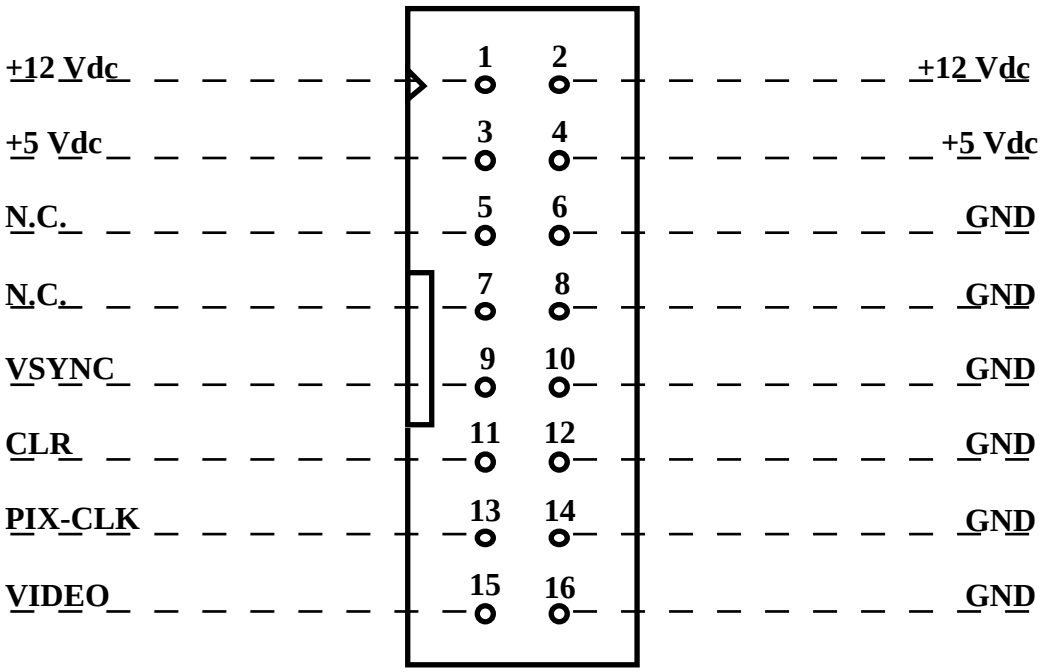



**FIGURE 8: JP5 - PLANAR DISPLAY CONNECTOR**

Signals description:

**VIDEO** = O - Video signal.
**CLR** = O - Clear video signal.
**PIX-CLK** = O - System clk signal.
**VSYNC** = O - Vertical synchronism signal.
**+5 Vdc** = O - Power supply +5 Vdc.
**+12 Vdc** = O - Power supply +12 Vdc.
**GND** = - Digital ground.
**N.C.** = - Not connected.

## JP6 - PC-AT KEYBOARD CONNECTOR

The connector for planar display, called JP6, is a 5 pins male connector with 2.54 mm pitch. It allows to interface directly a PC-AT keyboard, exactly like connector JP2.

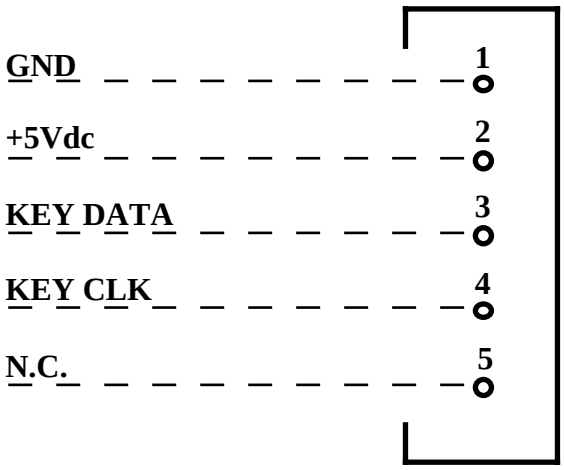



FIGURE 9: JP6 - PC-AT KEYBOARD CONNECTOR

Signals description:

| | | |
|---|---|---|
| **+5 Vdc** | = O | - Power supply +5 Vdc. |
| **KEY DATA** | = I/O | - Reception or transmission signal for PC-AT keyboard . |
| **KEY CLK** | = I | - Clock reception signal for PC-AT . |
| **N.C.** | = | - Not connected. |
| **GND** | = | - Digital ground. |

Here follows an example of connection with a PC-AT keyboard.

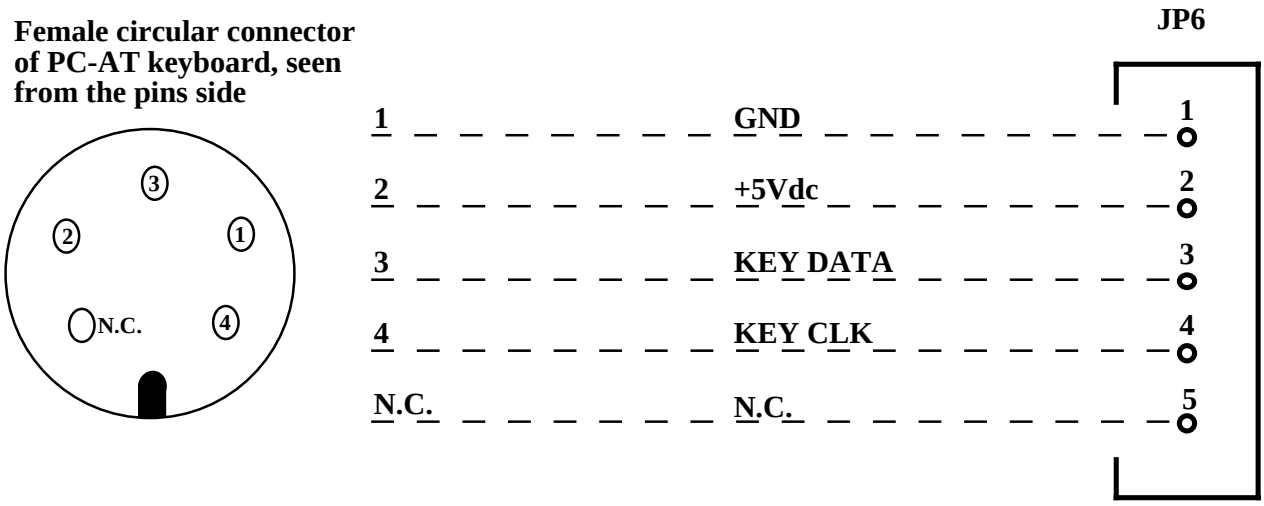



FIGURE 10: EXAMPLE OF CONNECTION WITH A PC-AT KEYBOARD

## JP7 - COMPOSITE VIDEO CONNECTOR

The connector for composite video, called JP7, is a 2 pins male AMP connector with 2.54 mm pitch.
It allows to interface directly a BAS type video composite monitor.

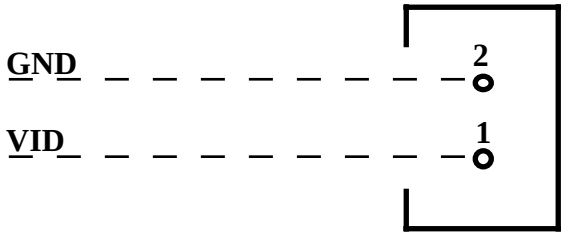



**FIGURE 11: JP7 - COMPOSITE VIDEO CONNECTOR**

Signals description:

**VID** = O - BAS type video composite signal 1.0 Vpp.
**GND** = - Video ground.

## JP9 - EXTERNAL LEDS CONNECTOR

The connector for external LEDs, called JP9, is a 10 pins male low profile connector with 2.54 mm pitch.
It allows to interface directly 8 common anode external LEDs.

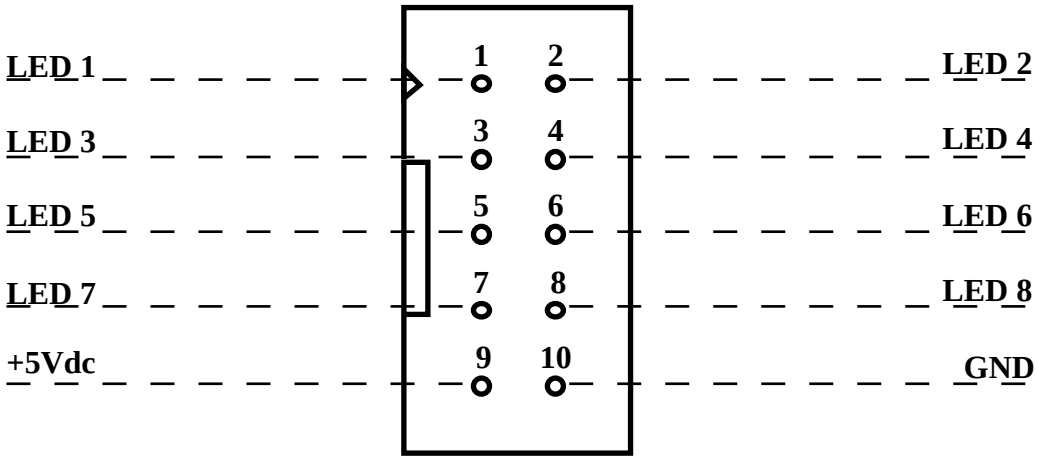



**FIGURE 12: JP7 - EXTERNAL LEDS CONNECTOR**

Signals description:

**+5 Vdc** = O - Power supply +5 Vdc.
**LED.n** = I - n-th LED cathod.
**GND** = - Digital ground.

## VISUAL SIGNALATIONS

**GDU 020** card is provided with two signalation LEDs to show several status informations, as described in the following table:

| LED | COLOUR | PURPOSE |
|---|---|---|
| DL1 | Yellow | The LED is turned ON when there is a logc level 1 on the buzzer drive output on connector JP3. |
| DL2 | Red | The LED is turned ON when the main CPU card performs an operation of read from or write to **GDU 020** through **ABACO® industrial BUS**. |

**FIGURE 13: VISUAL SIGNALATIONS TABLE**

The main purpose of LEDs is to show a visual indication about the card's status, making so easier debug and verify operations. To easily locate these visual signalations please refer to figure 6.

## POWER SUPPLY

**GDU 020** is provided with an efficent circuitery that solves in a comfortable and simple way the problem of the board's supply, under any condition of use.
Here follow the voltages nedded:

**+5 Vdc:** Supplies the on board logic; must be in the range +5 Vdc ± 5% and must be provided through the specific pins of connector K1 (**ABACO®** BUS).

**+12 Vdc:** Supplies the PLANAR display; must be present only if the PLANAR display is to be used as output display and must be provided through the opportune pins of JP5.

To warrant great immunity to external noise and so a correct working of the board, it is essential that **+5 Vdc** and the eventual **+12 Vdc** tensions are galvanically isolated.

## VIDEO SIGNALS

**GDU 020** generates two kinds of video signals; TTL and composite.

Video composite signals can be used with all the monitors capable of a scansion frequence of 16 kHz and resolution 521x288. The signal is available on connector JP7.

TTL signal for VGA monitor has scansion frequence of 31 kHz, pixel-rate of 20 MHz and resolution 512x480. The signal is available on connector JP4.

TTL signal for PLANAR display has scansion frequence of 16 kHz, pixel-rate of 16 MHz and resolution 512x256. The signal is available on connector JP4.

## JUMPERS

On **GDU 020** board there are 10 jumpers for card configuration. Below there is the jumpers list, location and function.

| JUMPERS | N. PINS | PURPOSE |
|---|---|---|
| J1 | 2 | Connects signal /M1 to interfacing and addressing section. |
| J4 | 2 | Abilitation of CPU internal ROM. |
| J5 | 3 | Sets U4 for RAM of 8 or 32 KByte. |
| J6 | 3 | Sets U4 for RAM of 2 or 8 KByte. |
| J7 | 3 | Selects RS 422-485 reception drivers. |
| J9 | 3 | Selects whether to connect pin 5 of connector JP5 to GND or to CLR. |
| J10 | 2 | Connects pin 8 of connector JP5 to GND. |
| J11 | 5 | Selects directionality and activation modality in RS 422-485 serial communication line. |
| J12 | 3 | Selects between normal or complemented VSYNC signal. |
| J13 | 3 | Selects between normal or complemented HSYNC signal. |

**FIGURE 14: JUMPERS SUMMARIZING TABLE**

The following tables describe all the right connections of **GDU 020** jumpers with their relative functions. To recognize these valid connections, please refer to the board printed diagram (serigraph) or to figure 2 of this manual, where the pins numeration is listed; for recognizing jumpers location, please refer to figure 15.
The "*" used in the following tables, 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.

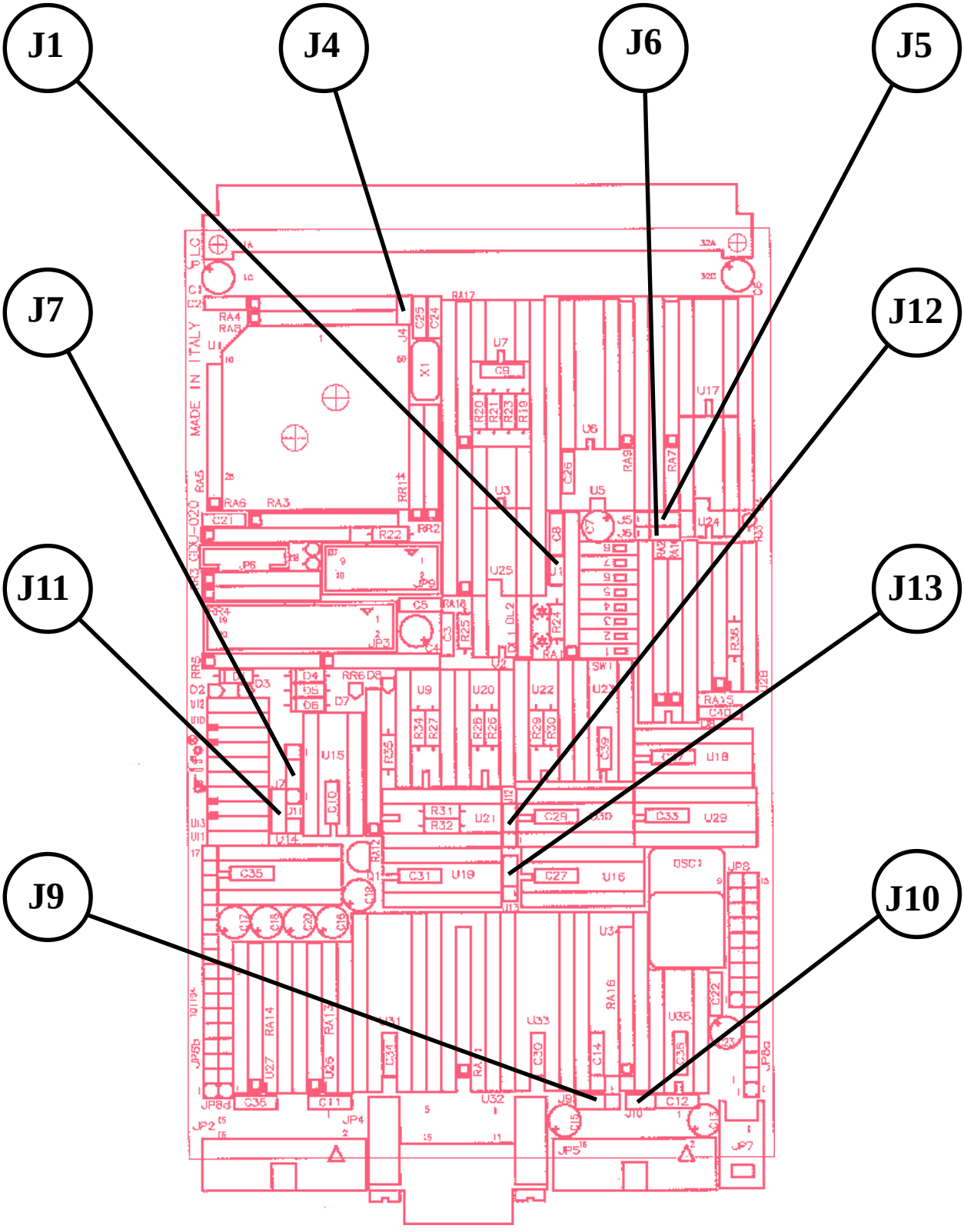



FIGURE 15: JUMPERS LOCATION

## 2 PINS JUMPERS

| JUMPERS | CONNECTION | PURPOSE | DEF. |
|---|---|---|---|
| J1 | not connected | The interfacement and addresing section does not manage BUS signal /M1. | |
| | connected | The interfacement and addresing section manages BUS signal /M1. | * |
| J4 | not connected | Activates CPU internal ROM. | |
| | connected | Does not activate CPU internal ROM. | * |
| J10 | not connected | Does not connect pin 8 of JP5 to board supply ground signal. | * |
| | connected | Connects pin 8 of JP5 to board supply ground signal. | |

**FIGURE 16: 2 PINS JUMPERS TABLE**

The "*" used in the following tables, 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.

## 3 PINS JUMPERS

| JUMPERS | CONNESSIONE | UTILIZZO | DEF. |
|---|---|---|---|
| J5 | position 1-2 | Sets U4 for RAM of 32 KBytes | * |
| | position 2-3 | Sets U4 for RAM of 8 KBytes | |
| J6 | position 1-2 | Sets U4 for RAM of 8 KBytes | * |
| | position 2-3 | Sets U4 for RAM of 2 KBytes | |
| J7 | position 1-2 | Selects driver on U10 for RS 422-485 reception. | |
| | position 2-3 | Selects driver on U10 for RS 422-485 reception. | |
| J9 | position 1-2 | Connects pin 5 of JP5 to board supply ground signal. | |
| | position 2-3 | Connects pin 5 of JP5 to board CLR signal. | |
| J12 | position 1-2 | Selects straight VSYNC signal. | |
| | position 2-3 | Selects complemented VSYNC signal. (suggested for VGA and PLANAR) | * |
| J13 | position 1-2 | Selects straight HSYNC signal. | |
| | position 2-3 | Selects complemented HSYNC signal. (suggested for VGA) | * |

**FIGURE 17: 3 PINS JUMPERS TABLE**

The "*" used in the following tables, 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.

**5 PINS JUMPERS**

| JUMPER | CONNECTION | PURPOSE | DEF. |
|---|---|---|---|
| J11 | posizione 1-2 e 3-4 | Enables full duplex or half duplex 4-wires serial transmission in RS 422. | |
| | posizione 2-3 e 4-5 | Enables half duplex 2-wires serial transmission in RS 485. | |
| | disconnected | Enables RS 232 or Current Loop serial communication, according to the driver installed. | * |

**FIGURE 18: 5 PINS JUMPERS TABLE**

The "*" used in the following tables, 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.

## BOARD CONNNECTIONS

To prevent possible connecting problems between **GDU 020** board and the external systems, the user has to read carefully the information of the previous paragraphs and he must follow these instructions:

- The TTL signals can be connected directly only to a device featuring the same type of interface. About the correspondance between logic signals and TTL output status, remember that a logic **0** generates a TTL 0 Vdc, while a logic **1** generates a TTL +5 Vdc.

- The composite video signal can be connected directly only to a device compliant with BAS specification and 1.0 Vpp.

## SERIAL COMMUNICATION SELECTION

Serial line can be buffered only as RS 232, RS 422, RS 485 or current loop. By hardware can be selected which one of these electric standards is used, through jumpers connection (as described in the previous tables) and drivers installation. By software the baude rate can be set during setup phase; bits per character, parity and stop bits cannot be changed.

In the following paragraphs there are all the information on serial communication configurations. 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. This far the user can change in autonomy the configuration following the informations below:

- SERIAL LINE IN RS 232 (default configuration)

| | | | | |
|---|---|---|---|---|
| | | | U10 | = don't care |
| J7 | = | disconnected | U11 | = don't care |
| J11 | = | disconnected | U12 | = don't care |
| | | | U13 | = don't care |
| | | | U14 | = driver MAX 202 |

- SERIAL LINE IN CURRENT LOOP (option .CLOOP)

| | | | | |
|---|---|---|---|---|
| | | | U10 | = don't care |
| J7 | = | disconnected | U11 | = don't care |
| J11 | = | disconnected | U12 | = HP4200 |
| | | | U13 | = HP4100 |
| | | | U14 | = don't care |

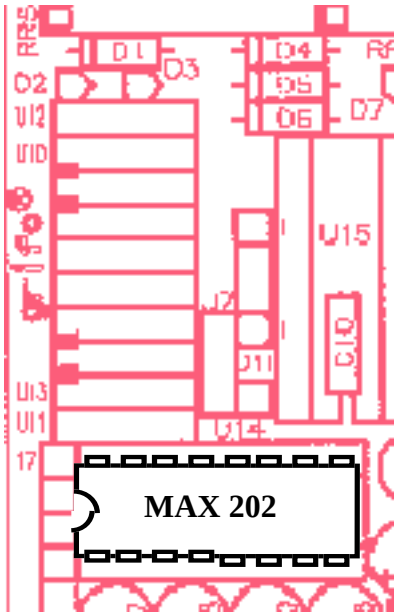
Please remark that Current Loop serial interface is passive, so it must be connected an active Current Loop serial line, that is a line provided with its own power supply.

- SERIAL LINE IN RS 422 (option .RS 422)

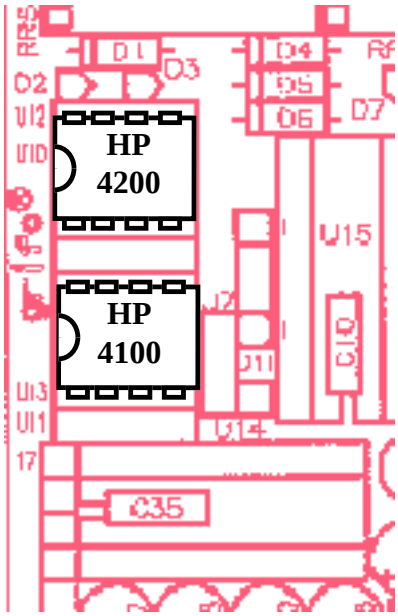
| | | | | |
|---|---|---|---|---|
| | | | U10 | = SN75176 |
| J7 | = | position 1-2 | U11 | = SN75176 |
| J11 | = | position 1-2 and 3-4 | U12 | = don't care |
| | | | U13 | = don't care |
| | | | U14 | = don't care |

- SERIAL LINE IN RS 485 (option .RS 485)

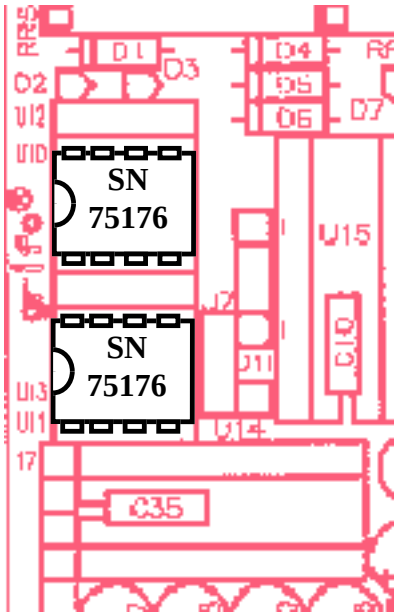
| | | | | |
|---|---|---|---|---|
| | | | U10 | = SN75176 |
| J7 | = | position 1-2 | U11 | = don't care |
| J11 | = | position 2-3 and 4-5 | U12 | = don't care |
| | | | U13 | = don't care |
| | | | U14 | = don't care |



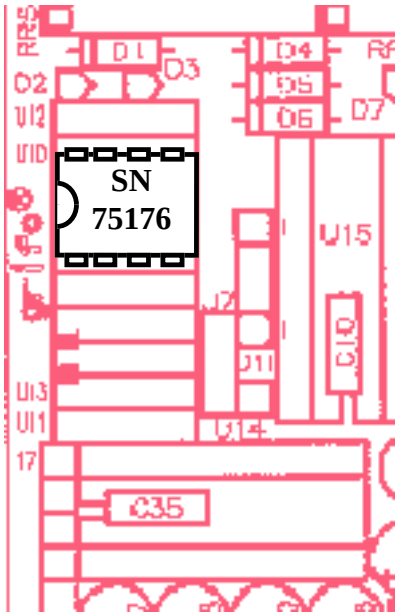

Serial B in RS 232




Serial B in Current Loop




Serial B in RS 422




Serial B in RS 485

**FIGURE 19: SERIAL COMMUNICATION DRIVERS LOCATION**

# SOFTWARE DESCRIPTION

**GDU 020** board is a complete video terminal that can be managed through the serial line or **ABACO®** industrial BUS. The on-board firmware is designed to recognize specific conditions and to react in consequence. This manual contains a complete list of the comand sequences and the recognized combination used to benefit of the main features of **GDU 020**. For each code or codes sequence, there is a double description: the mnemonic one, through the ASCII characters, and the numerical one under decimal and hexadecimal form.
The said commands respect the **ADDS View Point** standard so all the sequences begin with **ESC** character corresponding to the **27** decimal code (**1B Hex**).
Following description is referred to version **1.7** of the firmware.

## SETUP COMMAND

Setup mode allows the card to recognize the configuration command, described in the followin paragraph.
To enter in setup mode, turn off the board, connect pins 7 and 8 of connector JP2, or connect pins 3 and 4 of connector JP6 (please refer to figure 6 to easily locate the connectors), turn on the board and send the "READ PRESENCE BYTE" (ESC N) command, if the board responds with AA Hex then the previously described connections have not been made correctly, please repeat the sequence; otherwise if the board responds with BB Hex it is ready to receive the configuration command.
If the configuration command completes successfully, it is essential to remove the above described connections, then turn off and on the board again to enter the normal working mode.

## CONFIGURATION COMMAND

| | | | | |
|---|---|---|---|---|
| ***Code:*** | ***27*** | ***33*** | ***69*** | ***byte*** |
| ***Hex code:*** | ***1B*** | ***21*** | ***45*** | ***byte*** |
| ***Mnemonic:*** | ***ESC*** | | ***!*** | ***E*** | ***ASCII(byte)*** |

The parameter indicated as "byte" represents the working configuration code for the card, the meaning of its bits is:

| | ***bit7*** | ***bit6*** | ***bit5*** | ***bit4*** | ***bit3*** | ***bit2*** | ***bit1*** | ***bit0*** |
|---|---|---|---|---|---|---|---|---|
| ***BYTE =*** | ***B/S*** | ***P.C.*** | ***NU*** | ***NU*** | ***NU*** | ***NU*** | ***S1*** | ***S0*** |

where:

B/S = selects the communication mode.
- 0 = communication through serial port
- 1 = communication through **ABACO® Industrial BUS**

P.C. = if on (1) enables the PC-AT USA keyboard
NU = Not used
S1 S0 = select the baud rate of serial communication
- 0 0 = selects Baud Rate of 19200 Baud
- 0 1 = selects Baud Rate of 9600 Baud
- 1 0 = selects Baud Rate of 4800 Baud
- 1 1 = selects Baud Rate of 2400 Baud

As can be easily seen, only baud rate can be changed in serial communicatio; data bits are 8, stop bit is 1 and no parity are unchangeable parameters.
Default configuration is 10000000 binary (128 decimal or 80 hexadecimal), which means:
**ABACO®** industrial BUS for communication and matrix keyboard for input.

***Example***
For example, configuration code for using the bus and the PC-AT keyboard is 11000000 binary number (192 decimal or C0 hexadecimal).

***Board response***
After sending the configuration byte, the board can respond:

| | |
|---|---|
| AA Hexadecimal = | New setting correctly saved in EEPROM |
| E4 Hexadecimal = | Could not save new setting in EEPROM |

## PC-AT KEYBOARD (USA)

**GDU 020** becomes a fully functional terminal thaks to the possibility to use a PC-AT keyboard like an ASCII decoded 7 bit keyboard.
To let the user use also the function keys, despite they are not included in ASCII code, there is a matching between each one of them and a double code:

| | | |
|---|---|---|
| ***Code:*** | ***27*** | ***byte*** |
| ***Hex code:*** | ***1B*** | ***byte*** |
| ***Mnemoonic:*** | ***ESC*** | ***ASCII(byte)*** |

Each function key generates two characters, ESC and a **byte** from 1 to 12 corresponding to each one of the twelve function keys.
For example, F10 generates these two characters:

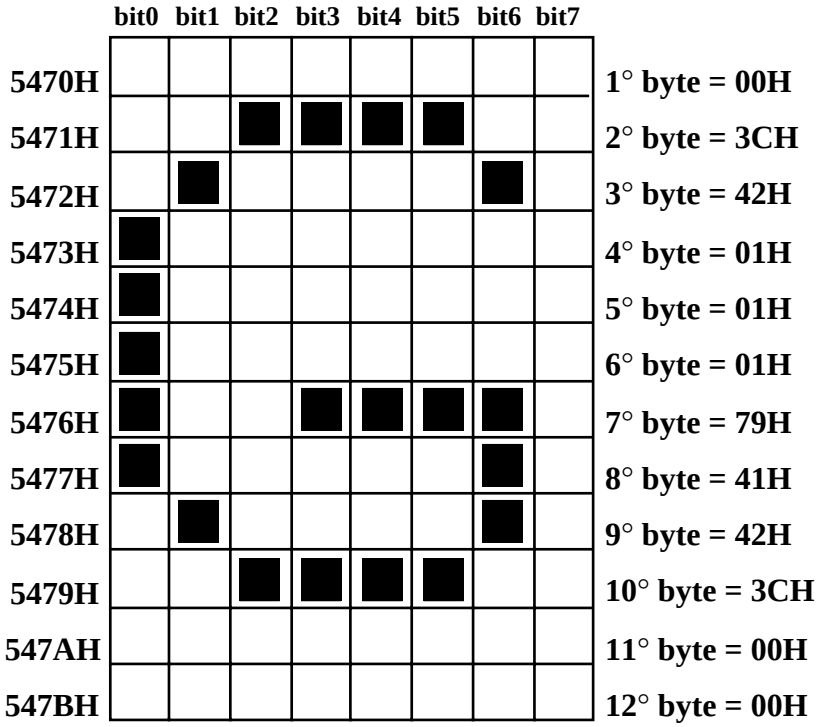
| | | |
|---|---|---|
| ***Code for F10:*** | ***27*** | ***10*** |
| ***Hex code for F10:*** | ***1B*** | ***0A*** |
| ***Mnemonic for F10:*** | ***ESC*** | ***LF*** |

Due to their incompatibility with ASCII code, some keys of PC-AT keyboard do not generate any code, the keys are:

| | |
|---|---|
| ***Alt*** | ***Tab*** |
| ***Print Screen (SysRq)*** | ***Scroll Lock*** |
| ***Pause (Break)*** | ***Insert*** |
| ***Page Up*** | ***Page Down*** |
| ***Home*** | ***End*** |

## REPRESENTATION OF A CHARACTER ON THE DISPLAY

All the ASCII characters whose code is in the range 32÷255 (20÷FF Hex) are visualized on the display, however the graphic patterns have been implemented only up to the character whose code is 129, where character 128 is "µ" and character 129 "C"; any character out of this range, except for commands, is ignored.
Characters rangin from 130 to 255 (82÷FF Hex) are not implemented so the user can take advantage of them to create special symbols.
Characters pattern is stored in EPROM U5 from address 05200 Hex to 05FFF Hex, where each pattern 8x12 pixels is made of 16 bytes (only the first 12 are used).
For example, to get the pattern of "G", whose ASCII code is 47 hexadecimal, its first byte is located at 05470 Hex, the following 15 bytes complete it but only the first 12 bytes are used.
Please refer to the following figure.




**547CH ....547FH = not used**

**FIGURE 20: EXAMPLE PATTERN "G"**

To get the EPROM address in hexadecimal of a character the following calculation should be used:

Address = 5000H + (ASCII(character)H * 10H

For example, the address of character "G" (hexadecimal ASCII code 47):

Address = 5000H + (47H * 10H) = 5470H

Next character will be located at hexadecimal 5480H.
Please remark that the character pattern is a grid 8x12 pixel when Zoom=0, when Zoom is not 0 the characters resolution is (8 x Zoom) x (12 x Zoom).

## CURSOR POSITIONING COMMANDS

Here follows the list of the cursor positioning commands.

### 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.

### 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, il will be moved to the Home position.

### 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 i.e first line, first column of the display, or on the other hand the upper left corner (0, 0).

**CARRIAGE RETURN**

| | |
|---|---|
| ***Code:*** | ***13*** |
| ***Hex code:*** | ***D*** |
| ***Mnemonic:*** | ***CR*** |

The cursor is moved to the beginning of the line where it was located.

**ALPHANUMERIC CURSOR PLACEMENT**

| | |
|---|---|
| ***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**".
These codes are the row and column values of the position, plus a constant offset of **32** (**20 Hex**). If, for example, the user wants to place the cursor at Home position (line 0, column 0), the following byte sequence must be sent:

**27 89 32 32**.

If row and/or column values are not compatible to the installed display, the command is ignored.

**ABSOLUTE CURSOR PLACEMENT**

| | | | | | |
|---|---|---|---|---|---|
| ***Code:*** | ***27 90*** | ***xH*** | ***xL*** | ***yH*** | ***yL*** |
| ***Hex code:*** | ***1B 5A*** | ***xH*** | ***xL*** | ***yH*** | ***yL*** |
| ***Mnemonic:*** | ***ESC Z*** | ***ASCII(xH)*** | ***ASCII(xL)*** | ***ASCII(yH)*** | ***ASCII(yL)*** |

Cursor is placed at absolute coordinaters with origin (0, 0) in the top left corner of the display. The parameter specify the distance of the character from the origin; in detail ***xH*** and ***xL*** specify high byte and low byte of distance along X axis, of course ***yH*** and ***yL*** specify high byte and low byte of distance along Y axis.
Character coordinates refer to bottom left corner of its grid, next characters will keep the same tabulation up to display end, from Home postition normal tabulation will be restored.
When using, for example, 512x256 display, last usable coordinates will be 511x255.
For example, to locate the cursor in Home position (row 0, column 0), the following byte sequence must be sent:

**27 90 0 0 0 11**

## COMMANDS FOR CHARACTERS ERASURE

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

### BACKSPACE

***Code:*** ***8***
***Hex code:*** ***8***
***Mnemonic:*** ***BS***

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

### CLEAR PAGE

***Code:*** ***12***
***Hex code:*** ***C***
***Mnemonic:*** ***FF***

This command clears all data on the display, sets Zomm to 1 and moves the cursor to Home position.

### CLEAR END OF LINE

***Code:*** ***27 75***
***Hex code:*** ***1B 4B***
***Mnemonic:*** ***ESC K***

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.

## ATTRIBUTES SELECTION COMMANDS

Here follow the commands to manage several board attributes.

### REVERSE ATTRIBUTE SELECTION

| | |
|---|---|
| ***Code:*** | ***27 48 80*** |
| ***Hex code:*** | ***1B 30 50*** |
| ***Mnemonic:*** | ***ESC 0 P*** |

Enables or disables the possibility to display characters in reverse mode (please see following commands).

### REVERSE ACTIVATION

| | |
|---|---|
| ***Code:*** | ***14*** |
| ***Hex code:*** | ***E*** |
| ***Mnemonic:*** | ***SO*** |

Characters are displayed in reverse.

### REVERSE DEACTIVATION

| | |
|---|---|
| ***Code:*** | ***15*** |
| ***Hex code:*** | ***F*** |
| ***Mnemonic:*** | ***SI*** |

Characters are not displayed in reverse any more.

### LEDS ACTIVATION WITH MASK

| | |
|---|---|
| ***Code:*** | ***27 50 byte*** |
| ***Hex code:*** | ***1B 32 byte*** |
| ***Mnemonic:*** | ***ESC 2 ASCII(byte)*** |

All LEDs on **GDU 020** are managed at the same time through the code inicated in ***byte***:

| ***bit 0*** | ***bit 1*** | ***bit 2*** | ***bit 3*** | ***bit 4*** | ***bit 5*** | ***bit 6*** | ***bit 7*** |
|---|---|---|---|---|---|---|---|
| **LED 0** | **LED 1** | **LED 2** | **LED 3** | **LED 4** | **LED 5** | **LED 6** | **LED 7** |

Bit set to 0 means LED turned OFF, bit set to 1 means LED turned ON.
For example, to turn ON LED n.5 and OFF all other LEDs, send: **27 50 32**

## LEDS ACTIVATION WITH COMPLEMENTED MASK

***Code:*** ***27 53 byte***
***Hex code:*** ***1B 35 byte***
***Mnemonic:*** ***ESC 5 ASCII(byte)***

All LEDs on **GDU 020** are managed at the same time through the code inicated in ***byte***:

| ***bit 0*** | ***bit 1*** | ***bit 2*** | ***bit 3*** | ***bit 4*** | ***bit 5*** | ***bit 6*** | ***bit 7*** |
|---|---|---|---|---|---|---|---|
| **LED 0** | **LED 1** | **LED 2** | **LED 3** | **LED 4** | **LED 5** | **LED 6** | **LED 7** |

Bit set to 1 means LED turned OFF, bit set to 0 means LED turned ON.
For example, to turn ON LED n.5 and OFF all other LEDs, send: **27 50 223**

## CHARACTER ZOOM SELECTION

***Code:*** ***27 1 byte***
***Hex code:*** ***1B 01 byte***
***Mnemonic:*** ***ESC SOH ASCII(byte)***

Thanks to the on board GDC features, it is possible to set up to 16 (from 0 to 15) character zoom levels, where zoom level 0 means a character pattern 8x12 pixels, each zoom level increment characters pattern becomes 8 pixel larger and 12 pixel higher, up to the maximum (8x16) x( 12x16)=128 x 192. Please remark that whenever this command is used, cursor position is reset to home position.

## READ PRESENCE BYTE

***Code:*** ***27 78***
***Hex code:*** ***1B 4E***
***Mnemonic:*** ***ESC N***

**GDU 020** returns its presence byte.
The byte can have two different values:

| | | |
|---|---|---|
| **AA** Hexadecimal | = | Board ready for use |
| **BB** Hexadecimal | = | Invalid setup, board waiting for a new setup |

This command can be used, for example, to detect the board presence or to check the board correct working.

## READ ERROR BYTE

| | |
|---|---|
| ***Code:*** | ***27 88*** |
| ***Hex code:*** | ***1B 58*** |
| ***Mnemonic:*** | ***ESC X*** |

At power on, **GDU 020** calculaters its error byte, its value can be requested in any moment after power on so the status it reports may not correspond any more to the current board status.
This command makes the board send error byte to serial line, it can have these different values:

| | | |
|---|---|---|
| **E0** Hexadecimal | = | Board ready for use |
| **E1** Hexadecimal | = | PC-AT keyboard internal error, it will not be used |
| **E2** Hexadecimal | = | Communication problems with PC-AT keyboard or not present |
| **E3** Hexadecimal | = | Invalid setup, **ABACO®** BUS communication will be used |

This command can be used, for example, to check the board correct configuration.

## READ VERSION NUMBER

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

**GDU 020** sends to serial line a three characters string containing the on board firmware version number, for example “1” “.” “7”.

## COMMANDS TO MANAGE µPD72020

Here follow two commands that allow the usere to take advantage of the GDC **µP 72020** features not supported by this version of the firmware. To do this it is very important to read very carefully the manual of GDC **µPD 72020**.
During the execution of these two commands, one of which carries instructions while the other one carries data, local CPU acts only as a repeater that does not change in any way the byte flow, it only checks its internal buffer for non-empty condition whenever a byte transfer operation to GDC occours.

### SEND A COMMAND TO µPD 72020

***Code:*** ***2 byte***
***Hex code:*** ***02 byte***
***Mnemonic:*** ***STX ASCII(byte)***

This command allows to send the instruction specified by ***byte*** to **µPD 72020**.

### SEND A DATA TO µPD 72020

***Code:*** ***3 byte***
***Hex code:*** ***03 byte***
***Mnemonic:*** ***ETX ASCII(byte)***

This command allows to send the data specified by ***byte*** to **µPD 72020**.

## COMMANDS FOR GRAPHIC

Here follow all the commands for graphic management available in this version of firmware.
The notation “xnH”, “xnL” or “ynH”, “ynL” will soon be used in the following, x and y represent the coordinates in a cartesian axis system with origin (0, 0) in the top left corner of the display, n is a progressibe number that distinguishes different coordinates in the same command, H and L indicate respectively the high byte and the low byte of a coordinate.
Please remark that, for example, using a 512x256 display the maximum coordinates available are 511x255, further coordinates write to a zone of the display not visible.

### DRAW A RECTANGLE

| | |
|---|---|
| ***Code:*** | ***27 2 x1H x1L y1H y1L x2H x2L y2H y2L*** |
| ***Hex code:*** | ***1B 02 x1H x1L y1H y1L x2H x2L y2H y2L*** |
| ***Mnemonic:*** | ***ESC STX ASCII(x1H) ASCII(x1L) ASCII(y1H) ASCII(y1L) ASCII(x2H) ASCII(x2L) ASCII(y2H) ASCII(y2L)*** |

This command draws a rectangle whose top left corner is located at **x1, y1** and whose bottom right corner is located at **x2, y2**.
For example, to draw a square with 300 pixel per side originating in the home position, send:
**27 2 0 0 0 0 1 44 1 44**

### DRAW A FILLED RECTANGLE

| | |
|---|---|
| ***Code:*** | ***27 4 x1H x1L y1H y1L x2H x2L y2H y2L*** |
| ***Hex code:*** | ***1B 04 x1H x1L y1H y1L x2H x2L y2H y2L*** |
| ***Mnemonic:*** | ***ESC EOT ASCII(x1H) ASCII(x1L) ASCII(y1H) ASCII(y1L) ASCII(x2H) ASCII(x2L) ASCII(y2H) ASCII(y2L)*** |

This command draws a filled rectangle whose top left corner is located at **x1, y1** and whose bottom right corner is located at **x2, y2**. always sent for a correct execution of this command.
For example, to draw a filled square with 300 pixel per side originating in the home position, send:
**27 4 0 0 0 0 1 44 1 44**

### CLEAR AN AREA

| | |
|---|---|
| ***Code:*** | ***27 12 x1H x1L y1H y1L x2H x2L y2H y2L*** |
| ***Hex code:*** | ***1B 0C x1H x1L y1H y1L x2H x2L y2H y2L*** |
| ***Mnemonic:*** | ***ESC FF ASCII(x1H) ASCII(x1L) ASCII(y1H) ASCII(y1L) ASCII(x2H) ASCII(x2L) ASCII(y2H) ASCII(y2L)*** |

This command clears an area whose top left corner is located at **x1, y1** and whose bottom right corner is located at **x2, y2**. always sent for a correct execution of this command.
For example, to clear a square with 300 pixel per side originating in the home position, send:
**27 12 0 0 0 0 1 44 1 44**

## DRAW A LINE

| | |
|---|---|
| ***Code:*** | ***27 3 x1H x1L y1H y1L x2H x2L y2H y2L*** |
| ***Hex code:*** | ***1B 03 x1H x1L y1H y1L x2H x2L y2H y2L*** |
| ***Mnemonic:*** | ***ESC ETX ASCII(x1H) ASCII(x1L) ASCII(y1H) ASCII(y1L) ASCII(x2H) ASCII(x2L) ASCII(y2H) ASCII(y2L)*** |

This command draws a line starting from the point at coordinates **x1, y1** and ending at the point at coordinates **x2, y2**.
For example, to draw a line from the home position and the bottom right corner in a 512x480 display, send: **27 3 0 0 0 0 1 255 1 224.**

## DRAW AN ARC OF CIRCUMFERENCE

| | |
|---|---|
| ***Code:*** | ***27 5 xH xL yH yL rH rL p n*** |
| ***Hex code:*** | ***1B 05 xH xL yH yL rH rL p n*** |
| ***Mnemonic:*** | ***ESC ENQ ASCII(xH) ASCII(xL) ASCII(yH) ASCII(1L) ASCII(rH) ASCII(rL) ASCII(p) ASCII(n)*** |

This command draws an arc of circumference having radius equal to **r**, center at coordinates **x**, **y**, starting arc **p** and made of **n** arcs. The values of these parameters must range from **0** to a maximum that depends on the size of the picture to draw; as it must be always possible to show it completely in the display area. The value of **p** must range from 0 to 7, the value of **n** must range from 1 to 8. Please refer to figure 21 to see how the circumference is divided in arcs by the firmware, remember that arcs can be drawn only at steps of 45 degreeses.
For example, to draw the left half circumference with origin at coordinates (100, 100) and radius 50, send: **27 5 0 100 0 100 0 50 2 4.**

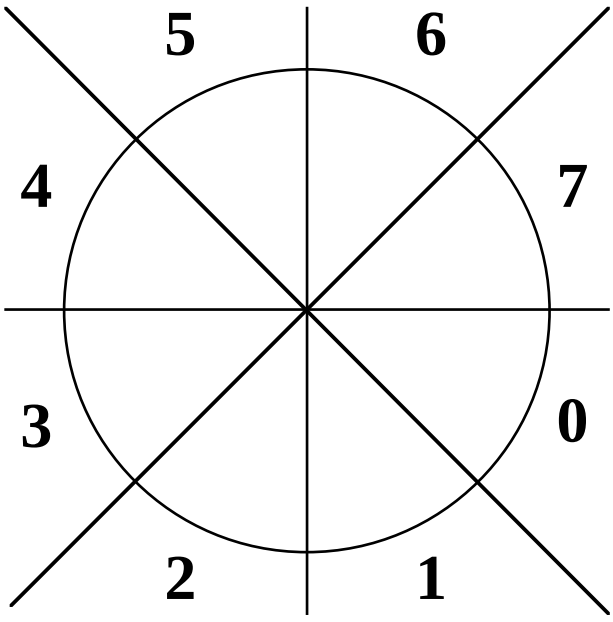



**FIGURE 21: ARCS NUMERATION AND LOCATION**

## DRAW A CIRCUMFERENCE

| | |
|---|---|
| ***Code:*** | ***27 6 xH xL yH yL rH rL*** |
| ***Hex code:*** | ***1B 06 xH xL yH yL rH rL*** |
| ***Mnemonic:*** | ***ESC ACK ASCII(xH) ASCII(xL) ASCII(yH) ASCII(1L) ASCII(rH) ASCII(rL)*** |

This command draws a circumference having radius equal to **r** and center at coordinates **x**, **y**. The values of these parameters must range from **0** to a maximum that depends on the size of the picture to draw; as it must be always possible to show it completely in the display area.
For example, to draw circumference with origin at coordinates (100, 100) and radius 50, send:
**27 6 0 100 0 100 0 50.**

| COMMAND | CODE | HEX CODE | MNEMONIC |
|---|---|---|---|
| **Home** | 01 | 01 | SOH |
| **Cursor Left** | 21 | 15 | NACK |
| **Cursor Right** | 06 | 06 | ACK |
| **Cursor Down** | 10 | 0A | LF |
| **Cursor Up** | 26 | 1A | SUB |
| **Carriage Return** | 13 | 0D | CR |
| **Alphanumeric Cursor Placement** | 27 89 r c | 1B 59 r c | ESC Y<br>ASCII(r) ASCII(c) |
| **Absolute Cursor Placement** | 27 90 xH xL yH yL | 1B 5A xH xL yH yL | ESC Z<br>ASCII(xH, xL)<br>ASCII(yH, yL) |
| **Backspace** | 08 | 08 | BS |
| **Clear Page** | 12 | 0C | FF |
| **Clear End Of Line** | 27 75 | 1B 4B | ESC K |
| **Reverese Attribute Selection** | 27 48 80 | 1B 30 50 | ESC 0 P |
| **Reverse Activation** | 14 | 0E | SO |
| **Reverse Deactivation** | 15 | 0F | SI |
| **LEDs Activation With Mask** | 27 50 byte | 1B 32 byte | ESC 2 byte |
| **LEDs Activation With Complemented Mask** | 27 53 byte | 1B 35 byte | ESC 5 byte |
| **Character Zoom Selection** | 27 1 byte | 1B 01 byte | ESC SOH byte |

**FIGURE 22: COMMAND SUMMARIZING TABLE PART 1**

| COMMAND | CODE | HEX CODE | MNEMONIC |
|---|---|---|---|
| **Read Presence Byte** | 27 78 | 1B 4E | ESC N |
| **Read Error Byte** | 27 88 | 1B 58 | ESC X |
| **Read Version Number** | 27 86 | 1B 56 | ESC V |
| **Character Visualization** | 32÷255 | 20÷FF | “space”÷ASCII(255) |
| **Send A Command To µPD72020** | 2 byte | 02 byte | STX byte |
| **Send A Data To µPD72020** | 3 byte | 03 byte | ETX byte |
| **Draw A Rectangle** | 27 2 N°8 bytes | 1B 02 N°8 bytes | ESC STX x1H x1L y1H y1L x2H x2L y2H y2L |
| **Draw A Filled Rectangle** | 27 4 N°8 bytes | 1B 04 N°8 bytes | ESC EOT x1H x1L y1H y1L x2H x2L y2H y2L |
| **Clear An Area** | 27 12 N°8 bytes | 1B 0C N°8 bytes | ESC FF x1H x1L y1H y1L x2H x2L y2H y2L |
| **Draw A Line** | 27 3 N°8 bytes | 1B 03 N°8 bytes | ESC ETX x1H x1L y1H y1L x2H x2L y2H y2L |
| **Draw A Circumference** | 27 6 N°6 bytes | 1B 06 N°6 bytes | ESC ACK xH xL yH yL rH rL |
| **Draw An Arc Of Cirumference** | 27 5 N°8 bytes | 1B 05 N°8 bytes | ESC ENQ xH xL yH yL rH rL p n |
| **Configuration Command** | 27 33 69 byte | 1B 21 45 byte | ESC ! E byte |

**FIGURE 23: COMMAND SUMMARIZING TABLE PART 2**

# HARDWARE DESCRIPTION

This chapter provides all the hardware informations needed to use **GDU 020** board. Here the user will find information about I/O card mapping and on board peripheral devices addressing.

## BOARD MAPPING

**GDU 020** board is mapped into a **2** bytes I/O addressing space, that can be mapped starting from different base addresses according to how the board is configured. This feature allows to use several **GDU 020** cards on the same **ABACO®** BUS, or to install them on a BUS where other peripheral modules are installed obtaining a structure that can be expanded without any difficulty or modifications to the application software.
The base address can be defined through the specific BUS interface circuitry on the board itself; this circuitry uses the eight pins dip switch called **SW1**, from which it reads the address set by the user. Here follows the corrispondance between dips configuration and address signals.

| | | |
|---|---|---|
| SW1.1 | -> | Enables the **/RESET** signal coming from the BUS |
| SW1.2 | -> | Address A1 |
| SW1.3 | -> | Address A2 |
| SW1.4 | -> | Address A3 |
| SW1.5 | -> | Address A4 |
| SW1.6 | -> | Address A5 |
| SW1.7 | -> | Address A6 |
| SW1.8 | -> | Address A7 |

These dips are driven in complemented logic, this means that if a switch is **ON** generates a **logic zero**, viceversa if a switch is **OFF** generates a **logic one**.

SW1.1 allows to connect or not the **/RESET** signal coming from the BUS, when it is connected the board resets whenever the signal reaches the logic level 0 (0 Volt).

The other dips allow to compose an address for the board from 00H to FEH, even, allowing to select amogst a range of 128 addresses.

Also Jumper J1 affects BUS addressing logic, connecting or not the **/M1** signal from **ABACO®** BUS. If the control card is provided of **/M1** signal, then J1 must be connected and viceversa.

**NOTE**
When allocating the mapping address of the boards, plase be careful not to allocate more than one device in the same addressing space (count also the number of bytes occupied by the card). If this condtion will not be respected, a BUS conflict will happen; such conflict will compromise the correct working of the whole system.

As an example, some possible mappings are reported here.

**1)** Address used to map **GDU 020**: 192 with /RESET enabled
Control board used: /M1 signal connected

| | | |
|---|---|---|
| Jumper J1 | -> | Connected |
| SW1.1 | -> | ON |
| SW1.2 | -> | ON |
| SW1.3 | -> | ON |
| SW1.4 | -> | ON |
| SW1.5 | -> | ON |
| SW1.6 | -> | ON |
| SW1.7 | -> | OFF |
| SW1.8 | -> | OFF |

**2)** Address used to map **GDU 020**: 040H with /RESET not enabled
Control board used: /M1 signal not connected

| | | |
|---|---|---|
| Jumper J1 | -> | Not Connected |
| SW1.1 | -> | ON |
| SW1.2 | -> | ON |
| SW1.3 | -> | ON |
| SW1.4 | -> | ON |
| SW1.5 | -> | ON |
| SW1.6 | -> | ON |
| SW1.7 | -> | OFF |
| SW1.8 | -> | ON |

## INTERNAL REGISTERS ADDRESSING

Indicating the board base address with <**baseaddr**>, that is the address set using dip switch SW1, as indicated in the previous paragraph **GDU 020** internal registers are addressable as explained in the following table.

| REGISTER | ADDRESS | R/W | PURPOSE |
|---|---|---|---|
| DATA | <baseaddr>+00 | R/W | Data register of GDU 020 buffer. |
| STATUS | <baseaddr>+01 | R | Status register of GDU 020 buffer. |

**FIGURE 24: INTERNAL REGISTERS ADDRESSING TABLE**

**NOTE**
When allocating the mapping of the boards, plase be careful not to allocate more than one device in the same addressing space (count also the number of bytes occupied by the card). If this condtion will not be respected, a BUS conflict will happen; such conflict will compromise the correct working of the whole system.

## STATUS REGISTER

Bits of status register have the following meaning:

| | *bit7* | *bit6* | *bit5* | *bit4* | *bit3* | *bit2* | *bit1* | *bit0* |
|---|---|---|---|---|---|---|---|---|
| ***BYTE =*** | ***NU*** | ***NU*** | ***NU*** | ***NU*** | ***NU*** | ***NU*** | ***D/L*** | ***D/T*** |

NU = Not used
D/L = If ON (1) a data in the buffer is ready to be read
D/T = If OFF (0) buffer is empty, a data can be put in it

Data register is the channel to transmit or receive data from the board, to receive eventual keys pressed on the keyboard or to receive responses of commands.

## COMMUNICATION MANAGEMENT

Of course if the serial communication is being used there is no way to reach board internal register so their use is not possible.
Please remark that during serial communication the protocol XON/XOFF is used to awarn the user about the reception buffer status. Code XON (011 Hex) means that the reception buffer is almost full, with still about 500 byte free, so it is opportune to stop the transmission to avoid a possible data loss. Code XOFF (013H) means that the buffer is empty so there is no more danger to lose data.

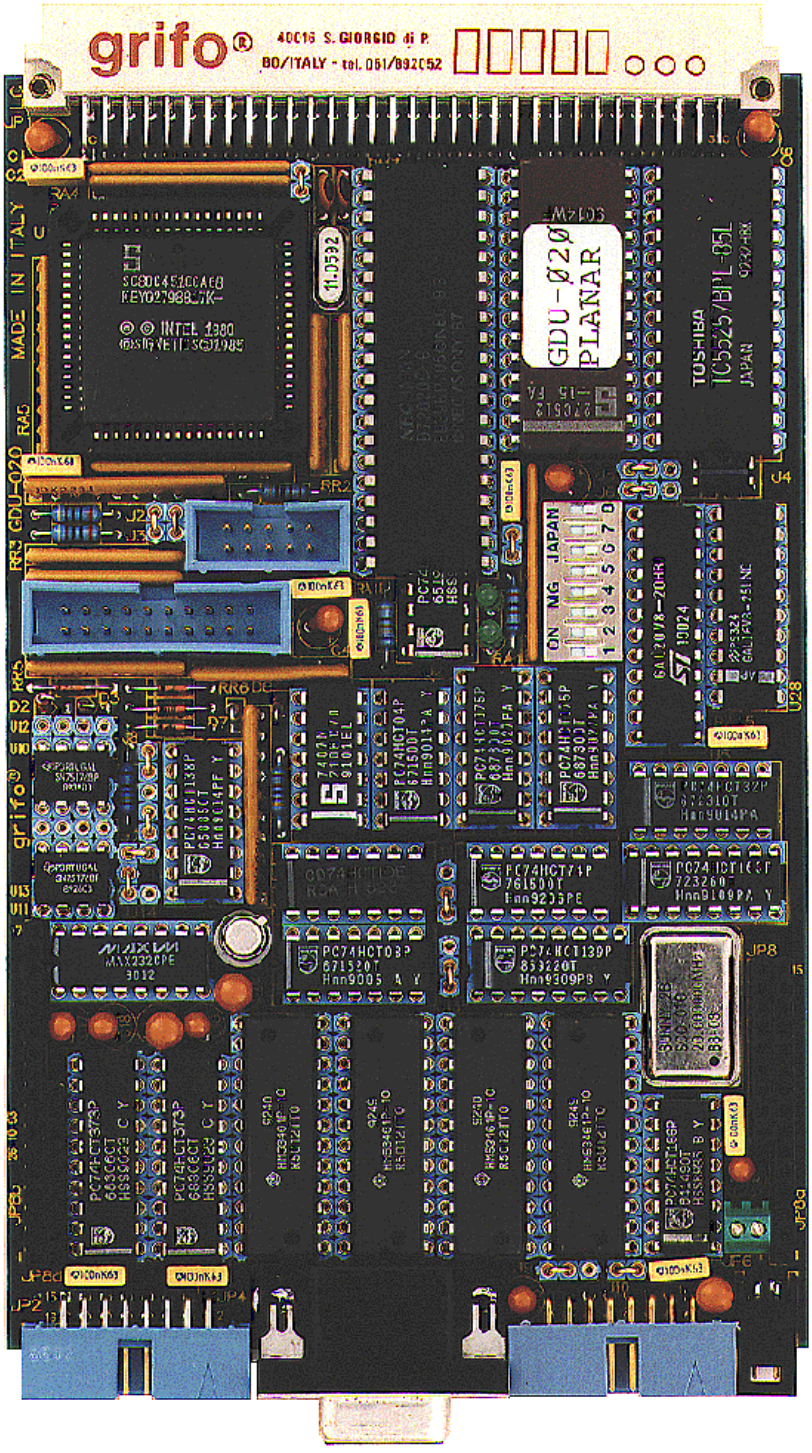



**FIGURE 25: CARD PHOTO**

# EXTERNAL CARDS

**GDU 020** can be connected to a wide range of block modules and operator interface system produced by **grifo®**, or to many system of other companies.The on board resources can be expanded with a simple connection to the numerous peripheral **grifo®** boards,both intelligent and not, thanks to its standard **ABACO®** BUS connector. Even cards with **ABACO®** I/O BUS can be connected, by using the proper mother boards.
Hereunder some of these cards are briefly described; ask the detailed information directly to **grifo®**, if required.

**MB3 01-MB4 01-MB8 01**
Mother Board 3, 4, 8 slots
Motherboard featuring 3, 4 or 8 slots of **ABACO®** industrial BUS; pitch 4 TE; standard power supply connectors; LEDs for visual feed-back of power supply; holes for rack docking.

**SPB 04-SPB 08**
Switch Power BUS 4-8 slots
Motherboard featuring 4-8 slots of **ABACO®** industrial BUS; pitch 4 TE; standard power supply connectors; termination resistances; connector type F for **SPC xxx** supply ; holes for rack docking.

**ABB 03**
**ABACO®** Block BUS 3 slots
3 slots **ABACO®** mother board; 4 TE pitch connectors; **ABACO®** I/O BUS connector; screw terminal for power supply; connection for DIN C type and Ω rails.

**ABB 05**
**ABACO®** Block BUS 5 slots
5 slots **ABACO®** mother board with power supply. Double power supply built in; 5Vdc 2,5A section for powering the on board logic; second section at 24Vdc 400mA galvanically coupled, for the optocoupled input lines. Auxiliary connector for **ABACO®** I/O BUS. Connection for DIN Ω rails.

**SPC 03.5S**
Switch Power Card +5 Vdc
Europe format switching power supply capable to provide +5 Vdc to a load of 4 A; input voltage 12÷24 Vac; power-failure; connector for back-up battery; standard connector for mother board **SPB 0x**.

**SPC 512**
Switch Power Card +5 Vdc +12 Vdc
Europe format switching power supply capable to provide +5 Vdc 5A and +12 Vdc 2.5 A; input voltage 12÷24 Vac; power-failure; connector for back-up battery; standard connector for mother board **SPB 0x**.

**FBC 20-120**
Flat Block Contact 20 vie
Interface for 2 or 1 mounting cable connectors (low profile 20 pins male) and quick release screw terminal connectors; Plastic mount for rails DIN 46277-1 and 3.

**GPC® 51**

General Purpose Controller fam. 51

Microprocessor family 51 INTEL including the masked BASIC chip; the board features: 16 I/O TTL lines; dip switch; 3 timer/counter; RS 232; 4 A/D converter signals resolution 11 bit; buzzer; on board EPROM programmer; RTC and 32K SRAM with Lithium battery back up; controlloer for display and keyboard.

**GPC® 188F**

General Purpose Controller 80C188

80C188 µP 20MHz; 1 RS 232 line; 1 RS 232, RS 422-485 or Current Loop line; 24 TTL I/O lines; 1M EPROM or 512K FLASH; 1M RAM Lithium battery backed; 8K serial EEPROM; RTC; Watch Dog; 8 Dip switch; 3 Timer Counter; 8 13 bit A/D lines; Power failure; activity LEDs; single power supply +5Vdc.

**GPC® 15A**

General Purpose Controller 84C15

Full CMOS card, 10÷20 MHz 84C15 CPU; 512K EPROM or FLASH; 128K RAM; 8K RAM and RTC backed; 8K serial EEPROM; 1 RS 232 line; 1 RS 232 line or RS 422-485 or Current Loop line; 32 or 40 TTL I/O lines; CTC; Watch dog; 2 Dip switches; Buzzer.

**GPC® 150**

General Purpose Controller 84C15

Microprocessor Z80 at 16 MHz; implementation completely CMOS; 512K EPROM or FLASH; 512K SRAM; RTC; Back-Up through external Lithium battery; 4M serail FLASH ; 1 serial line RS 232 plus 1 RS 232 or RS 422-485 or current loop; 40 I/O TTL; 2 timer/counter; 2 watch dog; dip switch; EEPROM; A/D converter with resolution 12 bit; activity LED.

**GPC® 15R**

General Purpose Controller 84C15

84C15 µP, 10÷16 MHz; 1 RS 232 line; 1 RS 232 or RS 422-485 or C. L. line; 16÷24 TTL I/O lines; 16 Opto-in; 8 Relays; 4 Opto Coupled Timers Counters; 512K EPROM or FLASH; 512K RAM and RTC backed; 8K serial EEPROM; 8K Backed RAM modul; Buzzer; 1 Activity LED; Watch dog; 4÷12 readable DIPs; LCD Interface.

**GPC® 323**

General Purpose Controller 51 family

80C32 µP, 14 MHz; Full CMOS; 1 RS 232 line (software); 1 RS 232 or RS 422-485 or Current Loop line; 24 TTL I/O lines; 11 A/D 12 bits lines; 3 Timers Counters; 64K EPROM; 64K RAM; 32K RAM and RTC backed; 32K DIL EEPROM; 8K serial EEPROM; Buzzer; 2 Activity LED; Watch dog; 5 readable DIPs; LCD Interface.

**GPC® 553**

General Purpose Controller 80C552

80C552 µP, 22÷33 MHz; 1 RS 232 line (software); 1 RS 232 or RS 422-485 or Current Loop line; 16 TTL I/O lines; 8 A/D 10 bits lines; 3 Timers Counters; 64K EPROM; 64K RAM; 32K RAM and RTC backed; 32K DIL EEPROM; 8K serial EEPROM; 2 PWM lines; 1 Activity LED; Watch dog; 5 readable DIPs; LCD Interface.

**GPC® 153**

General Purpose Controller Z80

84C15 µP, 10÷16 MHz; Full CMOS; 1 RS 232 line; 1 RS 232 or RS 422-485 or Current Loop line; 16 TTL I/O lines; 8 A/D 12 bits lines; 2÷4 Timers Counters; 512K EPROM or FLASH; 512K RAM and RTC backed; 8K serial EEPROM; Buzzer; 1 Activity LED; Watch dog; 8 readable DIPs; LCD Interface.

**GPC® 183**

General Purpose Controller Z180

Z180 µP, 10÷16 MHz; Full CMOS; 1 RS 232 line; 1 RS 232 or RS 422-485 or Current Loop line; 24 TTL I/O lines; 11 A/D 12 bits lines; 2 Timers Counters; 512K EPROM or FLASH; 512K RAM and RTC backed; 8K serial EEPROM; Buzzer; 2 Activity LED; Watch dog; 4 readable DIPs; LCD Interface.

**GPC® 324/D**

"4" Type General Purpose Controller 80C32/320

80C32 or 80C320 µP, 14÷22 MHz; Full CMOS; 1 RS 232 line; 1 RS 232 or RS 422-485 or Current Loop line; 4÷16 TTL I/O lines; 3 Timers Counters; 64K EPROM; 64K RAM; 32K RAM backed; 32K DIL E2; 8K serial EEPROM; Watch dog; 1 readable DIP; LCD Interface; Abaco® I/O BUS; 5Vdc Power supply; Size: 100x50 mm.

**GPC® 554**

General Purpose Controller 80C552

Microprocessor 80C552 at 22 MHz; implementation completely CMOS; 32K EPROM; 32 K SRAM; 32 K EEPROM or SRAM; EEPROM; 2 RS 232 serial lines; 16 I/O TTL; 2 PWM lines; 16 bits Timer/Counter; Watch Dog; 6 signals A/D converter with resolution 10 bit; interface for **ABACO®** I/O BUS.

**GPC® 154**

"4" Type General Purpose Controller Z80

84C15 µP, 10÷16 MHz; Full CMOS; 1 RS 232 line; 1 RS 232 or RS 422-485 line; 16 TTL I/O lines; 2÷4 Timers Counters; 512K EPROM or FLASH; 512K RAM and RTC backed; 8K serial EEPROM; Watch dog; 2 readable DIPs; LCD Interface; Abaco® I/O BUS; 5Vdc Power supply; Size: 100x50 mm.

**GPC® 884**

General Purpose Controller Am188ES

Microprocessor AMD Am188ES up to 40 MHz16 bits; implementation completely CMOS; serie 4 format; 512K EPROM or FLASH; 512K SRAM backed with Lithium battery; RTC; 1 RS 232 serial line + 1 RS 232 or RS 422-485 or current loop; 16 I/O TTL; 3 timer/counter; watch dog; EEPROM; 11 signals A/D converter with 12 bit resolution; interface for **ABACO®** I/O BUS.

**GPC® 114**

General Purpose Controller 68HC11

Microprocessor 68HC11A1 at 8 MHz; implementation completely CMOS; serie 4 format; 32K EPROM; 32K SRAM backed with Lithium battery; 32K EPROM, SRAM, EEPROM; RTC; 1 serial line RS 232 or RS 422-485; 10 I/O TTL; 3 timer/counter; watch dog; 8 signals A/D converter with resolution 8 bit; 1 asunchronous serial line; extremly low power consumption; interface for **ABACO®** I/O BUS.

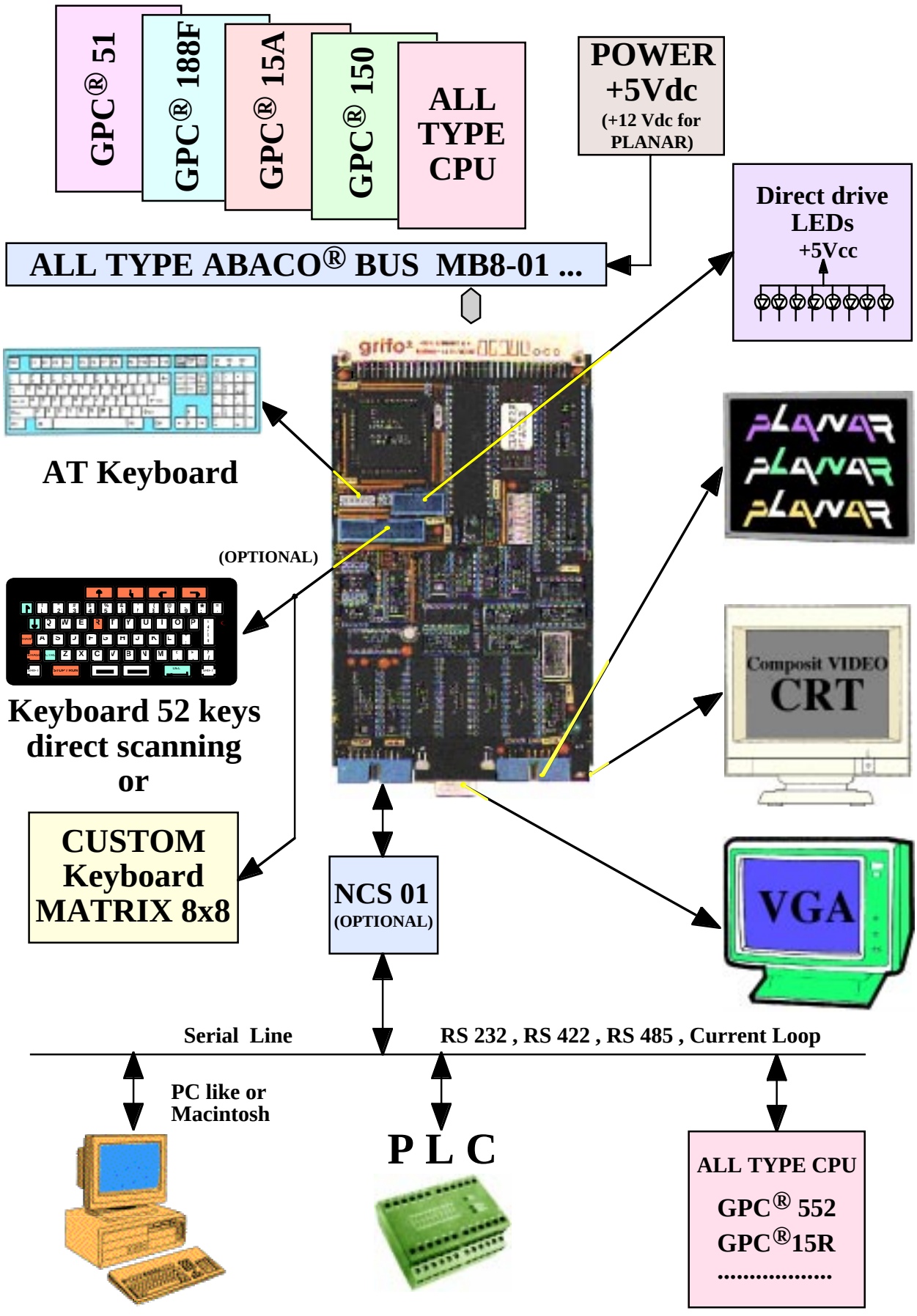



FIGURE 26: POSSIBLE CONNECTIONS DIAGRAM

**GPC® 184**

General Purpose Controller Z80195

Microprocessor Z80195 at 22 MHz; implementation completely CMOS; 512K EPROM or FLASH; 512K RAM; Back-Up with Lithium battery internal or external; 1 serial line RS 232 + 1 RS 232 or RS 422-485 or current loop + 1 TTL; 18 I/O TTL; 4 timer/counter 8 bits; 2 timer 16 bits; Watch Dog; Real Time Clock; activity LED; EEPROM; interface for **ABACO®** I/O BUS.

**GPC® AM4**

General Purpose Controller ATmega103

Microprocessor ATmega103 at 5.5 MHz; implementation completely CMOS; 128K internal FLASH; 32K SRAM; Back-Up with Lithium battery internal or external; 1 serial line RS 232 or RS 422-485 or current loop; 16 I/O TTL; 8 linee A/D resolution 10 bits; 2 timer/counter; Watch Dog; Real Time Clock; 4K internal EEPROM; interface for ISP programming; interface forr **ABACO®** I/O BUS.

# APPENDIX A: ALPHABETICAL INDEX

## SYMBOLS



## A



## B



## C

**D**



**E**



**G**



**H**



**I**

**J**



**K**



**L**



**M**



**O**



**P**



**R**

**S**



**T**



**V**



**W**



**X**



**Z**