SDI 02
Secure Digital Interface 2 GBytes

USER MANUAL
The SDI 02 is a complete memory cards interface specifically designed for an industrial use and for a direct mounting on automatic machines. The accepted memory cards are the SD (Secure Digital) and the MMC (Multi Media Card). The SDI 02 is a complete peripheral device, capable to write and read data on the listed memories supports, that must be linked to a master system through a serial communication in different types. A list of proper commands allow the master to exchange data with SDI 02 and let manage this data with typical resources of a disk as the Folders, the Files, the Attributes, the Protections, etc.

The memory card is organized with FAT16 standard and it can be used by any personal computer, as an external removable disk. This really simplify the data transfer and use in fact each PC.

Some additional features as the presence of a Power Supply Section that acceptts a wide range input, a Plastic Container that speed up the installation and an Optional Real Time Clock that adds a time reference to saved data, complete the product with an optimum price/performance ratio.

The SDI 02 is the suitable component in all the applications where big quantity of data must be managed, up to 2 GigaBytes maximum limit, by using a communication serial line that is normally provided by each automation and control systems, available on the market. In the last case the memory card is removed from SDI 02 and then used by any other system capable to manage the FAT16 standard; thus it can be developed applications for data logger, data exchange, configuration, update, production data collection, etc.

The serial communication with the master system can be established through an Asynchronous Line buffered in RS 232 or TTL or alternatively through an I2C BUS Synchronous Line.
IMPORTANT

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

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

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

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

HARDWARE AND FIRMWARE VERSION ............................................................................ 3

GENERAL INFORMATION ...................................................................................................... 4
REQUIREMENTS ................................................................................................................ .... 7

TECHNICAL FEATURES .......................................................................................................... 8
GENERAL FEATURES ............................................................................................................ 8
ELECTRIC FEATURES .......................................................................................................... 9
PHYSICAL FEATURES ......................................................................................................... 10

INSTALLATION ................................................................................................................... ..... 11
CONNECTIONS ................................................................................................................... .... 12
    CN3 - POWER SUPPLY CONNECTOR ........................................................................... 12
    CN1 - SERIAL COMMUNICATION LINE AND SWITCHING CONNECTOR ........ 14
    CN2 - MEMORY CARDS CONNECTOR ........................................................................ 18
JUMPERS ................................................................................................................................ 19
BACK UP ..................................................................................................................... ............ 20
ASYNCHRONOUS SERIAL LINE CONFIGURATION ................................................... 20
SIGNALLING LED ................................................................................................................ 21
POWER SUPPLY ................................................................................................................ .... 22

SOFTWARE DESCRIPTION ................................................................................................... 24
WORK PARAMETERS ......................................................................................................... 26
ELECTRIC PROTOCOL ...................................................................................................... 26
PHYSICAL PROTOCOL ..................................................................................................... 27
LOGIC PROTOCOL ............................................................................................................ 27
PERFORMED OPERATIONS .............................................................................................. 31
INSERTION AND REMOVING MEMORY CARD ........................................................... 31
HANDSHAKE COMMUNICATION .................................................................................... 31
ERRORS .................................................................................................................................. 33
NOTE FOR SYNCHRONOUS COMMUNICATION I2C BUS ........................................ 34
TIMEOUT IN COMMUNICATION ..................................................................................... 36
FILES DATE AND TIME ....................................................................................................... 36
TYPICAL APPLICATIONS ................................................................................................... 36
DEMO PROGRAMS .............................................................................................................. 38

COMMANDS ....................................................................................................................... ....... 39
GENERAL COMMANDS ...................................................................................................... 40
GET VERSIONS ............................................................................................................. ..... 40
RESET ERRORS .............................................................................................................. 40
GET STATUS ..................................................................................................................... 40
SET REAL TIME CLOCK .................................................................................................. 42
GET REAL TIME CLOCK ................................................................................................. 42
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRITE BACK UP SRAM</td>
<td>44</td>
</tr>
<tr>
<td>READ BACK UP SRAM</td>
<td>44</td>
</tr>
<tr>
<td>SET CONFIGURATION</td>
<td>45</td>
</tr>
<tr>
<td>GET CONFIGURATIONS</td>
<td>46</td>
</tr>
<tr>
<td>DISK COMMANDS</td>
<td>47</td>
</tr>
<tr>
<td>GET DISK FEATURES</td>
<td>47</td>
</tr>
<tr>
<td>FORMAT DISK</td>
<td>47</td>
</tr>
<tr>
<td>CREATE FOLDER</td>
<td>48</td>
</tr>
<tr>
<td>CHANGE FOLDER</td>
<td>48</td>
</tr>
<tr>
<td>REMOVE FOLDER</td>
<td>49</td>
</tr>
<tr>
<td>COMMANDS FOR FILES</td>
<td>50</td>
</tr>
<tr>
<td>GET FILE LIST</td>
<td>50</td>
</tr>
<tr>
<td>GET FILE INFORMATION</td>
<td>51</td>
</tr>
<tr>
<td>ERASE FILE</td>
<td>52</td>
</tr>
<tr>
<td>RENAME FILE</td>
<td>52</td>
</tr>
<tr>
<td>GET AVAILABLE FILE MANAGER</td>
<td>53</td>
</tr>
<tr>
<td>OPEN FILE</td>
<td>54</td>
</tr>
<tr>
<td>READ FILE</td>
<td>55</td>
</tr>
<tr>
<td>WRITE FILE</td>
<td>56</td>
</tr>
<tr>
<td>EMPTY BUFFER ON FILE</td>
<td>58</td>
</tr>
<tr>
<td>GET FILE POSITION</td>
<td>58</td>
</tr>
<tr>
<td>CLOSE FILE</td>
<td>59</td>
</tr>
<tr>
<td>HOW TO START</td>
<td>60</td>
</tr>
<tr>
<td>PRELIMINARY OPERATIONS</td>
<td>60</td>
</tr>
<tr>
<td>CONFIGURATION</td>
<td>61</td>
</tr>
<tr>
<td>USING</td>
<td>63</td>
</tr>
</tbody>
</table>

APPENDIX A: COMMANDS SUMMARY TABLES            A-1

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

APPENDIX C: ALPHABETICAL INDEX                  C-1
FIGURES INDEX

FIGURE 1: LOCATION OF HARDWARE AND FIRMWARE VERSION ......................................................... 3
FIGURE 2: VIEW WITHOUT BOX ........................................................................................................... 5
FIGURE 3: CONSUMPTION TABLE ........................................................................................................... 9
FIGURE 4: LOCATION OF JUMPERS, CONNECTORS, LED, BATTERY, ETC. ........................................... 11
FIGURE 5: CN3 - POWER SUPPLY CONNECTOR .................................................................................. 12
FIGURE 6: +5 VDC REGULATED POWER SUPPLY CONNECTION ....................................................... 13
FIGURE 7: 8+24 VDC ALTERNATE POWER SUPPLY CONNECTION (WITH .SW OPTION) ............. 13
FIGURE 8: 10+38 VDC DIRECT POWER SUPPLY CONNECTION (WITH .SW OPTION) ................... 13
FIGURE 9: CN1 SERIAL COMMUNICATION LINE AND SWITCHING CONNECTOR .............................. 14
FIGURE 10: CONNECTION EXAMPLE IN RS 232 WITH THREE WIRES (WITHOUT HANDSHAKE) ........... 15
FIGURE 11: CONNECTION EXAMPLE IN RS 232 WITH FIVE WIRES (WITH HANDSHAKE) ..................... 15
FIGURE 12: CONNECTION EXAMPLE IN TTL WITH THREE WIRES (WITHOUT HANDSHAKE) ............ 15
FIGURE 13: CONNECTION EXAMPLE IN TTL WITH FIVE WIRE (WITH HANDSHAKE) ......................... 15
FIGURE 14: CONNECTION EXAMPLE POINT TO POINT IN I2C BUS WITHOUT POWER SUPPLY ........... 16
FIGURE 15: CONNECTION EXAMPLE POINT TO POINT IN I2C BUS WITH POWER SUPPLY ............... 16
FIGURE 16: NETWORK CONNECTION EXAMPLE I2C BUS COMMUNICATION ...................................... 17
FIGURE 17: COMPONENTS MAP COMPONENTS SIDE AND SOLDER SIDE ........................................... 18
FIGURE 18: JUMPERS TABLE ............................................................................................................... 19
FIGURE 19: ASYNCHRONOUS SERIAL RS 232 JUMPER CONFIGURATION ............................................. 21
FIGURE 20: ASYNCHRONOUS SERIAL TTL JUMPER CONFIGURATION ................................................. 21
FIGURE 21: SIGNALLING LED .............................................................................................................. 21
FIGURE 22: POWER SUPPLY EL 12 ..................................................................................................... 23
FIGURE 23: WORK PARAMETERS DEFINITION .................................................................................... 25
FIGURE 24: I2C BUS SYNCHRONOUS COMMUNICATION FROM EXTERNAL SYSTEM TO SDI 02 ............ 28
FIGURE 25: I2C BUS SYNCHRONOUS COMMUNICATION FROM SDI 02 TO EXTERNAL SYSTEM ............. 29
FIGURE 26: ASYNCHRONOUS COMMUNICATION RS 232, TTL ........................................................... 30
FIGURE 27: NETWORK OF I2C BUS DIFFERENT DEVICES ................................................................... 35
FIGURE 28: DATA LOGGER FOR TEMPERATURES WITH GMM XXX CONNECTS IN RS 232 .................. 37
FIGURE 29: DATA LOGGER FOR TEMPERATURES WITH GMM XXX CONNECTS IN I2C BUS ................. 37
FIGURE 30: STATUS AND ERRORS PARAMETERS ............................................................................... 41
FIGURE 31: REAL TIME CLOCK PARAMETERS .................................................................................... 42
FIGURE 32: POTENTIAL CONNECTION DIAGRAM .............................................................................. 43
FIGURE 33: BACK UP SRAM PARAMETERS ......................................................................................... 44
FIGURE 34: CONFIGURATION PARAMETERS ........................................................................................ 45
FIGURE 35: DISK FEATURES PARAMETERS .......................................................................................... 47
FIGURE 36: COMPLETE VIEW ............................................................................................................. 49
FIGURE 37: FILE INFORMATION PARAMETERS .................................................................................... 51
FIGURE 38: FILE MANAGER PARAMETER ............................................................................................. 53
FIGURE 39: TOP VIEW .......................................................................................................................... 53
FIGURE 40: OPEN FILE PARAMETERS .................................................................................................. 54
FIGURE 41: READ FILE PARAMETERS .................................................................................................. 55
FIGURE 42: WRITE FILE PARAMETERS .................................................................................................. 56
FIGURE 43: DATA SAVING APPLICATION WITH GAB 844 ................................................................. 57
FIGURE 44: RS 232 CONNECTION WITH PC ......................................................................................... 60
FIGURE 45: PC SERIAL SETTING WITH DEMO_SD102 FOR CONFIGURATION .................................. 61
FIGURE 46: GETTING VERSIONS WITH DEMO_SDI02 ................................................................. 62
FIGURE 47: COMMUNICATION CONFIGURATION WITH DEMO_SDI02 ........................................ 62
FIGURE 48: PC SERIAL SETTING FOR APPLICATION WITH DEMO_SDI02 .................................. 64
FIGURE 49: DISK FEATURES GETTING ....................................................................................... 64
FIGURE 50: DISK FORMATTING WITH SDI 02 ........................................................................ 65
FIGURE 51: FILE OPENING WITH DEMO_SDI02 ...................................................................... 66
FIGURE 52: FILE WRITING WITH DEMO_SDI02 ....................................................................... 67
FIGURE 53: FILE INFORMATION ACQUIRING CON DEMO_SDI02 ............................................... 68
FIGURE 54: FILE READING WITH DEMO_SDI02 ...................................................................... 69
FIGURE 55: ERRORS STATUS MANAGEMENT WITH DEMO_SDI02 ........................................... 70
FIGURE A1: CONFIGURATION COMMANDS TABLE (1 OF 2) .................................................... A-1
FIGURE A2: CONFIGURATION COMMANDS TABLE (2 OF 2) .................................................... A-2
FIGURE B1: DEFAULT CONFIGURATION TABLE ........................................................................ B-1
FIGURE B2: DEFAULT JUMPERS CONFIGURATION TABLE ............................................................ B-1
FIGURE B3: OPTIONS TABLE ....................................................................................................... B-2
FIGURE B4: AMP4.CABLE CONNECTION ACCESSORY .............................................................. B-2
FIGURE B5: CKS.AMP4 CONNECTION ACCESSORY ................................................................. B-3
FIGURE B6: AMP8.CABLE CONNECTION ACCESSORY .............................................................. B-3
FIGURE B7: CKS.AMP8 CONNECTION ACCESSORY ................................................................. B-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, respectfully at the begining and at the end of the manual, to find information in a faster and more easy way.

_grifo®_ provid 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 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.
HARDWARE AND FIRMWARE VERSION

This handbook make reference to printed circuit version 110907 and to firmware version 4.1 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.

![Figure 1: Location of Hardware and Firmware Version](image)

Obviously if the version must be checked, then it must be extracted from the metallic container: the firmware version can be checked on bottom of the container too, near the configuration ordered. Moreover both numbers version can be also directly required to SDI 02 by using dedicated command.

Normally the SDI 02 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.

In this manual there are information referred to other programs that built SDI 02 too: each ones have proper number version that, if needed, is shown in this manual.

In case of necessity of technical assistance is important that user supply version numbers of his products.
GENERAL INFORMATION

The **SDI 02** is a complete memory cards interface specifically designed for an industrial use and for a direct mounting on automatic machines. The accepted memory cards are the SD (Secure Digital) and the MMC (Multi Media Card), that are the ones actually provided of the best features concerning price, handling and reliability. The **SDI 02** is a complete peripheral device, capable to write and read data on the listed memories supports, that must be linked to a master system through a serial communication in different types. A list of proper commands allow the master to exchange data with **SDI 02** and let manage this data with typical resources of a disk as the Folders, the Files, the Attributes, the Protections, etc.

The memory card is organized with FAT16 standard and it can be used by any personal computer, as an external removable disk. This really simplify the data transfer and use in fact each PC, provided of memory cards interface, can read, write, copy, open the available files, without any limits. Some additional features as the presence of a Power Supply Section that accepts a wide range input, a Plastic Container that speed up the installation and an Optional Real Time Clock that adds a time reference to saved data, complete the product with an optimum price/performance ratio.

The **SDI 02** is the suitable component in all the applications where big quantity of data must be managed, up to 2 GigaBytes maximum limit, by using a communication serial line that is normally provided by each automation and control systems, available on the market. The data can be managed only on site and in this condition the interface acts as a memory expansion of the master system or they can be transfer to another system. In the last case the memory card is removed from **SDI 02** and then used by any other system capable to manage the FAT16 standard; thus it can be developed applications for data logger, data exchange, configuration, update, production data collection, etc.

The serial communication with the master system can be established through an Asynchronous Line buffered in RS 232 or TTL or alternatively through an I2C BUS Synchronous Line. The selection of the used serial line and of the relative parameters is comfortably executed with a specific Configuration Modality, directly accessible to the final user.

Hardware features of SDI 02, complete of possible options, are listed below:

- Modular plastic container DIN 50022 modulbox, M3 HC53 model
- Overall dimension: 90x53 mm frontal size; 58 mm height
- Weight 0,084 Kg.
- Mounting on DIN Omega rails DIN 46277-1 and DIN 46277-3
- Connector for memory cards in SD and MMC formats
- Circuit for recognition of memory card Insertion and its Write Protect status
- 1 bicolour LED for visual signalations, driven by firmware with different modalities
- Real Time Clock backed by proper Lithium battery
- 240 bytes of backed SRAM
- Asynchronous communication serial line configurable in RS 232 or TTL
- I2C BUS synchronous communication serial line
- 1 digital output in Open Collector, connected to internal connector
- 2 handy connectors for a fast wiring of:
  - Power supply
  - Communication lines
- Power supply section capable to support also small external loads
- Two different power supply types:
  - 5 Vdc regulated
  - unregulated wide range from 10 to 38 Vdc or from 8 to 24 Vac
- Minimum required power equal to about 0,390 W + memory card consumption
- On board protection against voltage peaks by TransZorb
- Supplied with default management Firmware, that supports a complete File System
- Customized firmware and program under specific request of the user

For specific requirements about functionality and price, please contact directly grifo®

---

**Figures 2: View without box**

In order to simplify the use of SDI 02 the grifo® has realized a management firmware that speed up the development of the user application program. Generally the firmware allows to use all the hardware resources of the interface through an high level File System; in other words the user doesn't have to interact directly with the described hardware sections, but simply manage the provided commands. The general features of the firmwares, including possible options, are the following ones:

- Management of memory cards with capacity up to 2 GBytes.
- Memory cards managed as a Disk with FAT16 format.
- Data on disk are organized in Folders and Files, compatible with the most diffused Operating Systems for PC (MS-DOS, Windows, etc.).
- Files and Folders names are managed with 8.3 format (8 characters for name and 3 for extension): the extended names are converted into this format.
- Management of file Attributes either in read and write operations.
- Up to 2 Files contemporaneously managed.
- Management of current Date and Time from optional RTC that are used as a temporal reference for the files.
- Possibility to set and acquire the Real Time Clock (RTC) and all its temporal parameters (hour, minutes, seconds, day, month, year).
- Possibility to write and read the backed SRAM locations in order to keep user data even when
power supply is not available and memory card is not inserted.
- Recognition and management of the memory card Insertion.
- Recognition and management of the memory card Write Protect selector.
- Recognition and management of the files Read Only attribute.
- Memory card status visualized by Bicolour LED, that shows up to 5 different condition.
- Complete management of the possible Errors: 12 general and 6 for the disk.
- The errors are managed through specific fields in the communication protocol and proper commands.
- Tens of Commands dedicated to typical requirements of data management (see details in following table).
- The firmware lets the SDI 02 work as a Peripheral device that executes the commands received from an external system.
- All the Commands, and the relative Parameters and Responses, are exchanged through a Communication Line by following a proper logic, physic and electric protocol.
- ASCII communication Logic Protocol specifically designed in order to simplify and speed up the commands execution and the data management.
- Proper modality and Configuration commands allow the user to select the Physic and Electric Protocols, that will be used for communication.
- The provided Electric Protocol are:
  - Asynchronous Serial Line buffered in RS 232 or TTL.
  - I2C BUS Synchronous Serial Line.
- Selectable Physic Protocol for asynchronous serial communication:
  - 4.800, 9.600, 19.200, 38.400, 57.600 or 115.200 Bauds;
  - 8 bits for character;
  - No, Even, Odd Parity;
  - 1 or 2 Stop bits.
- Hardware (RTS, CTS) or Software (XON, XOFF) communication Handshaking
- Selectable Physic Protocol for I2C BUS synchronous serial communication:
  - Bit Rate from 225 to 460.799 bps;
  - Slave mode;
  - SDI 02 Slave Address range from 00H to FEH with step of 2, equal to 128 different values
- Serial Network connection through I2C BUS synchronous protocol.
- Medium Speed for data transfer from master to memory card are:
  - 4.000 Bytes/second in Write and 10.000 Bytes/second in Read, with asynchronous serial line at 115.200 Baud
  - 6.000 Bytes/second in Write and 10.000 Bytes/second in Read, with synchronous I2C BUS serial line at 400 KBits/second
Supplied with Demo Programs that shows and simplify the use of all the commands. One of these can be executed on PC and allows the management of SDI 02 connected through a free RS 232 serial line of computer.
The interface can be driven even by hand by using any system with an asynchronous serial line and a Terminal Emulation program (i.e. a standard PC that executes Hyperterminal).
REQUIREMENTS

Following there is a list of necessary material to use SDI 02:

a) Purpose documentation, in other words, this user manual.

b) A power supply compatible with needed configuration (please see POWER SUPPLY paragraph for details).

c) An external system able to talk one of three serial interfaces of SDI 02, according with physical and electric protocol selected.

d) A serial communication cable able to communicate with the electric standard selected, between external device described at point c and SDI 02. To realize this connection please see the several examples and figures on this manual.

e) One memory card SD or MMC up to 2 GB capacity, in FAT or FAT16 format.

Moreover isn't needed but is recommended:

f) One personal computer able to execute Demo_SDI02 program for SDI 02 configuration and make it ready to work in the system to realize. This PC must have following basic features:

   Personal Computer: IBM compatible
   RAM memory: ≥ 64 MBytes
   Hard disk: ≥ 8 MByte liberi
   Video board: ≥ 800x600 pixels, 65536 colours
   Monitor: Colours
   Interfaces: One serial line COM free
   Operating system: Windows 98, ME, 2000, XP

Conf_SLOG program doesn’t proper drivers but uses those ones already presents in the operating system so every item above described has to be configurated and working accordig with the operating system.

To realize the final application, there are available more examples programs made for talk with SDI 02. The customer has to find before his interesting components and then to use them how is described in the same programs or in following chapter of this manual.

Some of described elements, can be downloaded from grifo® web sites.
TECHNICAL FEATURES

GENERAL FEATURES

Resources:
1 status bicolour LED
Asynchronous serial line, full duplex in RS 232, or TTL
Synchronous I2C BUS line slave mode
RTC backed by lithium battery
240 back up SRAM byte from lithium battery
Interface and connector for memory cards SD and MMC
Recognizing inserting and protection memory card
Configuration mode selecting
Wide range power supply section

Turn off time: 70 ms
Timing resolution: 20 ms
LEDs intermittent time: 640 ms
Tempo rip. handshake software: 1000 ms

Communication timeout: 0÷5000 ms or disabled

Network unity number: 128
with synchronous serial I2C BUS

Communication:
Select between: asynchronous (RS 232, TTL) and synchronous (I2C BUS)
Default: asynchronous

Asynchronous communication protocol (RS 232, TTL):
Baud rate: 4800, 9600, 19200, 38400, 57600, 115200
Bit per character: 8
Parity: off, even, odd
Stop Bit: 1, 2
Handshake: off, hardware, software, software repeated
Default: 19200 Baud, 8 Bit, None parity, 1 Stop, None handshake

Synchronous communication protocol (I2C BUS):
Bit rate: from 225 to 460.799 bits per second
Mode: Slave
Slave Address: from 00H to FEH step of 2
Default: Slave Address = 128 = 80H

Reception buffer size: 5 characters
Commands buffer commands: 57 characters
Max. file name length: 35 characters
Number of files at once: 2
Max. number of read, write file with one command: 65535

Max. reading speed: 10.000 Bytes/sec, with asynchronous serial at 115.200 Baud 10.000 Bytes/sec, with synchronous serial I2C BUS at 400 KBits/sec

Max. writing speed: 4.000 Bytes/sec, with asynchronous serial at 115.200 Baud 6.000 Bytes/sec, with synchronous serial I2C BUS at 400 KBits/sec

Memory card format: FAT, FAT16

**ELECTRIC FEATURES**

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

Power consumption: please see next table (*)

<table>
<thead>
<tr>
<th>Measurement Conditions</th>
<th>Consumption max. +5 Vdc</th>
<th>Consumption max. 10÷38 Vdc 8÷24 Vac</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDI 02 without SD, not connected to other systems</td>
<td>44 mA</td>
<td>0,23 W</td>
</tr>
<tr>
<td>SDI 02.RTC with SD, connected to Master system, at work</td>
<td>60 mA</td>
<td>0,39 W</td>
</tr>
</tbody>
</table>

**FIGURE 3: CONSUMPTION TABLE**

Output power supply voltage: +5,0 Vdc

Current available on +5Vdc output: 400 mA - consumption max. +5 Vdc (*)

RS 232 extravoltage protection: ±15 KV

I2C BUS pull up resistor: 4,7K Ω

Back up battery: 3 V Lithium; 180 mAh; CR 2032 model

Back up current: 3,2 µA

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

The previous table lists the **SDI 02** power consumption referred to tipical operating condition;
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.

**PHYSICAL FEATURES**

**Size:**
90 x 53 x 58 mm (L x A x P) with container
86 x 50 x 30 mm (L x A x P) without container

**Container:**
DIN 50022 modulbox, M3 HC53 model, plastic

**Mounting:**
On Omega rail DIN 46277-1 and DIN 46277-3

**Weight:**
84 g

**Temperature range:**
From 0 to 50 °C

**Relative umidity:**
20% up to 90% (without condense)

**Connectors:**
CN1: 8 pins AMP MODU II, vertical, male, pitch 2.54
CN2: SD, MMC, connector 90 degree
CN3: quick release screw terminal, 2 pins, male, pitch 3.5
INSTALLATION

In this chapter there are the information for a right installation and correct use of SDI 02. 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. All components that are not described in this chapter are for internal use and they can't be used used by user.

**Figure 4: Location of jumpers, connectors, LED, battery, etc.**
CONNECTIONS

The SDI 02 is provided of 3 connectors that are used for all field connections and with other system boards to realize. 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 figure 4 show the connectors position on the board, to simplify their recognitions. All the connectors are accessible through a proper breaking in the container that allows comfortable insertion and deinsertion.

CN3 - POWER SUPPLY CONNECTOR

CN3 is a vertical, 2 pins, male, quick release screw terminal connector, with 3.5 mm pitch. On CN3 must be connected the single power supply voltage for the module when is requested .SW option and is present the power supply section, able to accept a wide range voltage.

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, to next figures and the CN1 -SERIAL COMMUNICATION LINE AND SWITCHING CONNECTOR paragraph.
**Figure 6:** +5 Vdc regulated power supply connection

**Figure 7:** 8÷24 Vac alternate power supply connection (with .SW option)

**Figure 8:** 10÷38 Vdc direct power supply connection (with .SW option)
CN1 - SERIAL COMMUNICATION LINE AND SWITCHING CONNECTOR

CN1 is a 8 pins, male, vertical, AMP MODU II 4+4 connector with 2.54 mm pitch. On this connector there are the signals for communication in RS 232 or TTL, performed through hardware serial line of SDI 02. Signals placement has been designed to reduce interferences and electrical noises and to simplify the connections with other system; the electric protocols follow the CCITT directives of the used standard. Moreover on CN1 there is the regulated power voltage +5 Vdc that can be supplied or drawn; it depend on SDI 02 configuration. Female connector for CN1 is directly available between grifo® accessories, and it can be ordered by using the codes CKS.AMP8 or AMP8.Cable, as described in APPENDIX B of the manual.

![Diagram of CN1 Serial Communication Line and Switching Connector](image)

**Figura 9: CN1 Serial Communication Line and Switching Connector**

Legenda:

- **SDA** = I/O - Data signal of 'I2C BUS.
- **SCL** = I/O - Clock signal of 'I2C BUS.
- **RTS RS232** = O - Handshake reception line in RS 232.
- **RX RS232** = I - Reception line in RS 232.
- **TX RS232** = O - Transmission line in RS 232.
- **CTS RS232** = O - Handshake transmission line in RS 232.
- **RTS TTL** = O - Handshake reception line in TTL.
- **RX TTL** = I - Reception line in TTL.
- **TX TTL** = O - Transmission line in TTL.
- **CTS TTL** = O - Handshake transmission line in TTL.
- **+5 Vdc** = O - Power supply line +5 Vdc.
- **GND** = - Ground line.
- **N.C.** = - Not connected.

For details on board power supply, please see CN3 - wide range POWER SUPPLY CONNECTOR, POWER SUPPLY paragraphs and see figures 6+8. For details about asynchronous cominication we recommend to examine the ASYNCHRONOUS SERIAL LINE COMMUNICATION paragraph and to see figures 10+16 that show connection examples.
Figure 10: Connection example in RS 232 with three wires (without handshake)

Figure 11: Connection example in RS 232 with five wires (with handshake)

Figure 12: Connection example in TTL with three wires (without handshake)

Figure 13: Connection example in TTL with five wire (with handshake)
- For details about I2CBUS we recommend to examine the NOTE FOR SYNCHRONOUS I2CBUS COMMUNICATION paragraph and to see figures 14+16 that show connection examples both in point to point and network.

![Diagram of connection example point to point in I2C BUS without power supply](image1)

**Figure 14:** Connection example point to point in I2C BUS without power supply

![Diagram of connection example point to point in I2C BUS with power supply](image2)

**Figure 15:** Connection example point to point in I2C BUS with power supply

In connecting power supply is important to know that +5 Vdc voltage has to be generated by only one connected system and this one is able to supply powerful for the other system too. The SDI 02 can get and supply power voltage on CN1 connector too, as described in POWER SUPPLY paragraph, and for this purpose that J4 jumper can connect pin 8 of the connector.
Please remind that in a **I2C BUS** must be present two pull up resistors at the limits of the line, one next to master unit and one next slave unity. On **SDI 02** are always present the resistor (*1) and their value is reported in the ELECTRIC FEATURES paragraph. User has to choose and/or configure I2C BUS devices to connect, keeping in mind this feature. In detail on **SDI 02** these resistors have to be disconnected from the unity for those line as shown in the previous figure on slaves 1 and 2.

For further information see "**THE I2C-BUS SPECIFICATION**" document of PHILIPS Semiconductors.
CN2 - MEMORY CARDS CONNECTOR

CN2 is a connector for memory cards like SD and MMC type. In this chapter there are no description of connector pin out, in fact is perfectly compatible with used standard on memory card, but only a short bill of expedient and recommendation for use:

a) The max. capacity of utilizable cards is **2 GigaBytes**.
b) Utilizable cards have to be provided of **SPI** interface.
c) Cards have to be **formatted before with FAT** or **FAT16 standard**.
d) In case of problems of insertion in CN2 connector, verify the integrity of the same connector and than make sure of not finding not related pieces by mistake.
e) When the memory card is provided of **mechanical adapter**, before insertion of the card into the adapter, be sure that the anchorage is between card and adapted is good. A bag connection with the adapter can cause a bad inserting or removing and/or create malfunctions.
f) Avoid that card electric contacts get dirty or in touch with any substances.
g) Before insertion make sure that writing protection selector is disableb; otherwise the **SDI 02** can read data only and in case of writing it shows an error.

**FIGURE 17: COMPONENTS MAP COMPONENTS SIDE AND SOLDER SIDE**
JUMPERS

On SDI 02 there are nine 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>position 1-2</td>
<td>At power on selects the configuration mode.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>At power on does not select the configuration mode.</td>
<td></td>
</tr>
<tr>
<td>J2</td>
<td>not connected</td>
<td>Does not connect BT1 battery to back up circuitry.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td>Connects BT1 battery back up circuitry.</td>
<td>*</td>
</tr>
<tr>
<td>J4</td>
<td>position 1-2</td>
<td>Does not connect the power supply +5 Vdc to CN1 connector.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>Connects the power supply +5 Vdc to CN1 connector.</td>
<td></td>
</tr>
<tr>
<td>J5</td>
<td>position 1-2</td>
<td>Connects pin 7 of CN1 to RTS RS 232 signal.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>Connects pin 7 of CN1 to RTS TTL signal.</td>
<td></td>
</tr>
<tr>
<td>J6</td>
<td>position 1-2</td>
<td>Connects pin 1 of CN1 to CTS RS 232 signal.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>Connects pin 1 of CN1 to CTS TTL signal.</td>
<td></td>
</tr>
<tr>
<td>J7</td>
<td>position 1-2</td>
<td>Connects pin 3 of CN1 to TX RS 232 signal.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>Connects pin 3 of CN1 to TX TTL signal.</td>
<td></td>
</tr>
<tr>
<td>J8</td>
<td>position 1-2</td>
<td>Connect pin 5 of CN1 to RX RS 232 signal.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>Connect pin 5 of CN1 to RX TTL signal.</td>
<td></td>
</tr>
<tr>
<td>J9</td>
<td>position 1-2</td>
<td>Connects reception serial signal of the board to the RS 232 driver.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>Does not connect reception serial signal of the board to the RS 232 driver.</td>
<td></td>
</tr>
<tr>
<td>J10</td>
<td>position 1-2</td>
<td>Connects CTS handshake serial signal of the board to the RS 232 driver.</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>position 2-3</td>
<td>Does not connect CTS handshake serial signal of the board to the RS 232 driver.</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 18: JUMPERS table**

To recognize the valid connections and locations of these jumpers, please refer to the board printed diagram (serigraph) or to figure 4 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.
Further information about purpose of the **SDI 02** jumpers are reported in the following paragraphs, that describe the sections where the same jumpers are used.

**BACK UP**

When **SDI 02** contain a real time clock (**RTC**) feature, it is provided of a lithium battery that keeps hours, minutes, seconds, days, months and years, and the content of SRAM of 240 bytes. This device is managed by user to set and get current date and time used as reference for file on memory card, or read or write files which has to stay in power off condition too.

The clock is managed by **SDI 02** and information tell to user a temporal indication respect of saved data, so the external system that uses can check elapsed time, can obtain the the time of some events, calculate production data in a period, etc... The SRAM is for the external system to save and draw data which has been taken on **SDI 02** when the memory card has been removed and the power supply is off. For example can be used for configuration data, identification data, values that change quickly, etc. The clock and SRAM section is present only when is specified in the order phase, with **RTC** option and includes a lithium battery too that makes a back up of devices when the power supply is off.

The user can connect or not this battery to back up circuitry, by acting on dedicated jumper J2, as described in figure 18. The card is supplied with the jumper connected to preserve the clock counting and the SRAM content in each operating condition.

Whenever the **SDI 02** 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 J2. Obviously if the J2 connection must be changed, then the card must be extracted from the container making a pressure on the hooks. When printed circuit is extracted from container, the J2 location can be easily found by using figure 4, when for further information on clock function, please see FILES DATE AND TIME paragraph.

**ASYNCHRONOUS SERIAL LINE CONFIGURATION**

Serial communication line of **SDI 02** can be in three kind: asynchronous RS 232, asynchronous TTL or synchronous I2C BUS. The choice between asynchronous and synchronous line, of physical and logic communication protocol is via software by proper configuration modality (see bearing the same name paragraph).

When for configuration is chosen asynchronous serial, selection of electric protocol (RS 232 or TTL) is via hardware and reach a proper jumper configuration, as described on figure 18.

Following there are two figure that show jumper configuration involved in this selection, to make realization easy:
FIGURE 19: ASYNCHRONOUS SERIAL RS 232 JUMPER CONFIGURATION

FIGURE 20: ASYNCHRONOUS SERIAL TTL JUMPER CONFIGURATION

Used communication signals in both configurations always are that connect to 1, 3, 5, 7 and 2 pin of CN1, as described in figures 10-13.

NOTE. Two RS 232 and TTL protocols are electrically not compatible and user, before connect asynchronous serial of SDI 02 to external system, has to carefully verify current serial interface on the system and configures jumpers properly. The RS 232 line connection to SDI 02 in TTL configured can also cause the SDI 02 damaging.

SIGNALLING LED

On SDI 02 is present the bicolour LED DL1 that signal some operative conditions, as is described in the following table:

<table>
<thead>
<tr>
<th>STATUS</th>
<th>CONDITIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>switch off</td>
<td>S-LOG not supplied or out of order.</td>
</tr>
<tr>
<td>red, green, yellow alternating in about each half second</td>
<td>S-LOG working but without memory card.</td>
</tr>
<tr>
<td>green</td>
<td>S-LOG is working with memory card inserted.</td>
</tr>
<tr>
<td>red</td>
<td>S-LOG is saving received data, on memory card.</td>
</tr>
<tr>
<td>yellow</td>
<td>S-LOG is working and is performing a configuration command.</td>
</tr>
<tr>
<td>blinking yellow</td>
<td>S-LOG is working but with errors.</td>
</tr>
</tbody>
</table>

FIGURE 21: SIGNALLING LED

As shown in figure 4 the LED is easily visible in the insertion side if memory card; for more information on signalled conditions, please see SOFTWARE DESCRIPTION.
POWER SUPPLY

SDI 02 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. When SDI 02 is ordered in default version the power supply section is partially assembled and accepts only one regulated voltage of +5 Vdc. For alternate current it can be ordered with a proper power supply switching that accepts a wide range voltage.

Here follow the voltages required from SDI 02 according to card configuration together with the relative right connection:

Default version: Here the J4 jumper has to be in position 2-3. 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, at pin 2 (GND) and pin 8 (+5 Vdc) of CN1 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 condition is comfortable because with only one cable it can connect the power supply and the communication line, always on CN1.

SW Version This configuration includes a switching power supply that requires 10÷38 Vdc or 8÷24 Vac provided through CN1 (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 cheap solution for default version power supply can be the EL12 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 SDI 02 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 CN1 - POWER SUPPLY CONNECTOR.

With J4 jumper uses can decide to connect the regulated voltage to +5 Vdc, generated from switching, to CN1 connector: if J4 is in position 2-3 that voltage is connected and can be used for supply external components; if J4 is in 1-2 the pin 8 of CN1 is not connected and generated voltage is used only by SDI 02.

The complete version of power supply section described has to be properly ordered using the SDI 02 SW indication

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 **SDI 02** 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.

**Figure 22: Power supply EL 12**
SOFTWARE DESCRIPTION

The **SDI 02** module is a complete data logger interface projected to manage SD (Secure Digital) or MMC (MultiMedia Card) card through a serial communication line. Every external system is provided of compatible serial line that can directly communicate with **SDI 02** and indirectly with the memory card. All these things is done through a command series that allow to main system to read and write data and to manage them as classic disk resources; like folders, files, attributes, protections, errors, etc.

In this chapter are described the main **SDI 02** functional features while in the next one you can find the detailed documentation of the available configuration commands: these last ones allow the users to satisfy every market request.

MEMORY CARD ORGANIZATION

The **SDI 02** organizes the memory card as described in the following points:

With the standard FAT 16 which assures the use of every kind of personal computer like a removable external disk. The standard FAT16 is in fact supported by the most operative systems available on the market, such as MS-DOS, Windows, Linux, Mac-Os, etc… This vastly simplifies the transportation and the data use, in fact every pc, provided with an interface for memory cards, can read, write, copy, open files on the card, without any limitation.

Before the **SDI02** uses the memory card for the first time it has to be previously formatted with the standard FAT 16, by using a pc.

To a 2 Gigabytes maximum capacity, as described in the paragraph CN2 connector for memory cards.

CONFIGURATION

In order to make the **SDI 02** functions usable to the most of the external systems and in order to satisfy the requests of many applications, it is available a kind of configuration mode which allows to set the used communication protocol.

The serial communication towards the main system can happen with a RS 232 or TTL buffered asynchronous line or with a 12C BUS synchronous line. The selection of the used line and its parameters easily happens through the configuration mode, directly manageable by the final user.

In detail:

a) The configuration mode is enabled by the proper jumper J1:
   - J1 position 1-2 -> enabled configuration mode.
   - J1 position 2-3 -> not enabled configuration mode.

b) The jumper J1 is acquired from **SDI 02** only after a power on and so the variation of configuration enabling during work is not allowed.

c) When at power on of **SDI 02** the configuration mode is selected, are set the communication line and the default physical protocol. So makes sure always to check and/or reconfigure the module. Default selections are reported in figure B1, that are asynchronous communication at 19200 Baud, 8 bit, None parity, 1 Stop and none handshake.

d) Only when the configuration mode is enabled, the **SDI 02** recognize and preform the SET...
CONFIGURATION command, and the user can modify all work parameters described in the following paragraph. Viceversa if the configuration mode is not enabled, the command is ignored.

e) To modify the status of J1, before it has to extract the board from the container, performing a pressure on the closing hooks, in the bottom of the box. This choice is justified by that the configuration mode is normally disabled and is enabled occasionally (before the installation or for the beginning tests). The user that build the entire system will have to open only one time the container, enable the configuration mode and put the jumper in position 2-3 and re-close the container.

With the J1 access it avoids the accidental selection too, that should cause variations of the communication protocol set.

**Figure 23: Work Parameters Definition**

- **Asynchronous Communication Management (RS 232, TTL):**
  - Communication line = asynchronous (RS 232, TTL)
  - Timeout = 250 * 20 ms = 5 seconds
  - Baud rate = 19200
  - Bit per character = 8
  - Parity = none
  - Stop bit = 1
  - Handshake = none
  - Slave address = 128 (not used)

- **Synchronous Communication Management (I2C BUS):**

---

**SDI 02 Rel. 5.00**

Page 25
WORK PARAMETERS

The SDI 02 working in any its operative modality depends on a series of work parameters that define both communication aspects and data storage on memory card.
Current parameters are:

Work parameters for communication:
- **com.line** -> Serial line used for communication with external system.
- **baud** -> Serial Baud rate used for asynchronous serial communication in RS 232, TTL with external system.
- **parity** -> Parity check used for asynchronous serial communication in RS 232, TTL with external system.
- **stop** -> Stop bit used for asynchronous communication serial in RS 232, TTL with external system.
- **handshake** -> Handshake used for asynchronous serial communication in RS 232, TTL with external system.
- **sla.add** -> Recognizing SDI 02 address used for synchronous serial I2C BUS communication with external system.
- **tout** -> Max. waiting time during communication with external system.

A better description of work parameters is available in following paragraph and figures and in COMMANDS chapter; In that chapter are shew all available values as parameter and their default value.

Work parameters are very important and for their storage it was choosen a EEPROM to be sure of good performance and saved data preservation, naturally without power supply too. When saved values in EEPROM are not valid, work parameters will be set as default values as figure 23 and B1.

ELECTRIC PROTOCOL

Serial communication with SDI 02 can be done with all electric protocols available: asynchronous TTL, asynchronous RS 232 and synchronous I2C BUS, all availables on CN1.
The choise of the communication line depends on several factors as described in figure 23. In details:

Asynchronous serial line RS 232, TTL:
- Asynchronous communication uses signals on CN1 and they can be:
  - GND, RX xxxx, TX xxxx when external system don't use handshake;
  - GND, RX xxxx, TX xxxx, RTS xxxx, CTS xxxx when external system use hardware handshake;
  - GND, RX xxxx, TX xxxx when external system use software handshake (XON, XOFF);
  - Not listed signals are not managed and can be connected or not.
This communication is good for point to point connections in RS 232 or TTL, as described in ASYNCHRONOUS SERIAL LINE CONFIGURATION paragraph.
Synchronous serial line I2C BUS:
Synchronous communication uses signals on CN1 and they can be:
- GND, SCL, SDA when system not provides or not needs power supply;
- GND, SCL, SDA, +5 Vdc when external system provides or needs power supply.
This communication is good both for point to point connections and as network.

PHYSICAL PROTOCOL
Serial communication with SDI 02 can be done with all physical protocols, that are correspondent with possible combination of work parameters involved (baud, parity, stop, handshake, sla.add). Available values for physical protocol are listed in GENERAL FEATURES paragraph and in command paragraph.
The definition of physical protocol used by SDI 02 depends on work parameters configuration and on other factors described in figure 23.

LOGIC PROTOCOL
Serial communication with SDI 02 can be done with a proper logic protocol in order to simplify and increase speed the preparation and check out of the module by user. Main features of this protocol are summarized in the follow points, while flow charts in figures 25 and 26 show graphically operations performed by two involved communication configuration units.

a) Logic configuration protocol is based on proper commands that external system transmits to SDI 02; than SDI 02 executes them always transmits a response.
b) Both in the command and in the response can be present some added data that meet configuration parameters.
c) All command and response data are always coded in ASCII: moreover in case of numeric values it uses the decimal code, only with more significant numbers. This thing increase the number of communicated characters but in the same time provides an important benefit for SDI 02 that can be configured by hand by any user. It is sufficient a simple program of terminal emulation with the selected electric and physical protocols and no other devices, ecc.
d) If the command or the response contain more parameters, these ones are separated by a space.
e) In logic protocol a method for recognize the end of data of communication is used. This method meet with an end character adding (CR in command and CRLF> in response) when is used the asynchronous communication or with transmission of the numbers of response byte when is used the synchronous communication. This is because a lot of parameters have a variable length (see path and file name, strings to save on file, suffix and prefix, etc.) and that length is not known by reception unit.
f) During configuration are managed potential communication handshakes and timeout, as described in paragraphs bearing the same name.
g) Some commands, that shift big amount of data, use exception to logic protocol as described in the GET FILES LIST, READ FILE and WRITE FILE commands.
Figure 24: I2C BUS synchronous communication from external system to SDI 02

**EXTERNAL SYSTEM**

- **Start**
  - Initialize I2C BUS line in master mode with physical protocol (Bit rate) from 250 to 460000 Bit per second
  - Action management of master unity

**SDI 02**

- **Synchronous communication management (I2C BUS)**
  - Initialize I2C BUS line in slave mode with slave address set in work parameters
  - SDI 02 actions management (LED signalling, insertion and removing of memory card, initialization memory card, status drawing, etc.)

**Transmitted command with exception**

- **NO**
  - Build a communication as master transmit, like:
    1) send START sequence;
    2) send slave address set on SDI 02 writing (with R/W=0);
    3) verify reception of ACK;
    4) send the first byte of the command to perform;
    5) verify reception of ACK;
    6) send the second byte of the command to perform;
    7) verify reception of ACK;
    ... n) send the last byte of the command to perform;
    n+1) verify reception of ACK;
    n+2) send STOP sequence.

- **YES**
  - Receive response with synchronous communication (I2C BUS)
    - Use data of received response

**Received command with exception**

- **NO**
  - Build communication as slave receive, like:
    1) recognize the START sequence;
    2) receive and compare the slave address set on SDI 02:
      - if they meet it goes on otherwise it removes all I2C BUS data until the site next STOP sequence.
      - Check the type of operation R/W, if in writing (R/W=0) it goes on in reception otherwise sets sequence error of invalid communication;
    3) send ACK;
    4) receive and save on the first byte of the reception buffer of command to perform;
    5) send ACK;
    6) receive and save on reception buffer the second byte of command to perform;
    7) send ACK;
    ... n) receive and save on reception buffer the last byte of command to perform;
    n+1) send ACK;
    n+2) recognize the STOP sequence.

- **YES**
  - Send the response with synchronous communication (I2C BUS)

**Perform the received command setting the response**
h) The first character of response always meet the command of the executed flag. This is a single character that can assume the values:

- 0 (ASCII 48=30H code) -> not performed command
- 1 (ASCII 49=31H code) -> correctly performed command

In the first case the SDI 02 does not report any response data and the external system can recognize the communication and/or configuration problems occurs, through the GET STATUS command. So in the second case all response data are sent from the SDI 02.

i) During the communication for the commands, the LED of SDI 02 is yellow.

**FIGURE 25: I2C BUS SYNCHRONOUS COMMUNICATION FROM SDI 02 TO EXTERNAL SYSTEM**
**EXTERNAL SYSTEM**

Start

Initialize the serial line with physical protocol (Baud rate, 8 Bit per chr, Parity, Stop bit, Handshake) set on SDI 02

External unit process management

Input Handshake enabled and inactive

Transmit command byte to execute, to SDI 02

Command completely transmitted

Transmitted command with an exception

Transmit expection bytes command with input handshake check, if enabled

Received bytes from SDI 02

Save received bytes which are the reply to transmitted command, activating output handshake, if enabled and needed

Reply completely received

Received response with exception

Received bytes of exception as response activating the output handshake, if enabled and needed

Use received response data

**SDI 02**

Asynchronous communication management (RS 232, TTL)

Initialize asynchronous serial with physical protocol (Baud rate, 8 Bit per chr, Parity, Stop bit, Handshake) defined at power on in the work parameters

Actions management SDI 02 (signal on LED, insertion or removing memory card, memory card initialization, status drawing, etc.)

Received bytes from master unit

Save received bytes in reception buffer

Reception buffer full

Reception buffer with complete command with not expired timeout

Received command with exception

Receive exception bytes command with timeout, and activate output handshake, if enabled

Performe received command building the response

Input enabled and inactive handshake e disattivo and not expired timeout

Transmit response bytes of performed command, to master unity

Response completely transmitted

Transmitted response with exception

Transmit bytes of exception response with timeout, and input handshake check, if enabled

**FIGURE 26: ASYNCHRONOUS COMMUNICATION RS 232, TTL**
PERFORMED OPERATIONS

The **SDI 02** works as a peripheral device that performs the received commands from serial line and at the end that returns the response. There are other function are performed too:

- At the insertion time of memory card is verified and initialized in order to make easy all the other operations; if the initialization occurs correctly is activated the LED green.
- When the command to perform needs the writing on memory card, the **SDI 02** verify and manage the selector status for writing protection and naturally it manages the command when is disabled only. If the **SDI 02** receives a command that has to be written on the card and the protection activate, an error is signalled. Vice versa all commands that are not to be written on memory card, are always performed, with or without protection protection.
- Adding to communication with associated commands the **SDI 02** manages potential handshakes and timeout, as described in the following paragraphs.
- During the data writing operations, the **SDI 02** activates its LED of red colour in order to indicates the current operation. It is very important that the user does not remove the memory card when LED is red because he can lose his data and can damage the memory card. Please see the following paragraph for details.

INSERTION AND REMOVING MEMORY CARD

The **SDI 02** can correctly manages the inserted memory card only if this one is in **FAT16** format. This preparation can be done by a normal PC ans is performed only one time before first using.

At this point the memory card can be inserted in **SDI 02** and here can be performed the initialization operations needed. If in this phase there are errors, LED is blinking yellow (errors can be cleared by the proper command only); vice versa if there are not errors the LED become green. These operations described are performed after a power on of the board when the memory card is already inserted. In this case the LED is green too.

When the memory card was used by **SDI 02** can be removed but only when there are not saving in progress, that is when the LED is not red. If user want to remove the card but the LED is red he has to wait that the LED become green or blinking yellow to remove the memory card and be sure.

If these rules are not respected, saving data can be lost and the memory card can be damaged.

Once the memory card is removed, if no errors occured, the LED is blinking red, green and yellow. If there were errors the LED continues to blinking yellow.

HANDSHAKE COMMUNICATION

Reception performing commands the **SDI 02** makes some characterized by a variable execution time. Some of these operations, like the work parameters saving, physical writing on received data files, the formatting of the memory card, and many others, take a long execution time with the possible loss of transmitted data from the external system.

In order to avoid this loss, and connect external system with execution speed notably different, or variable work load, the **SDI 02** foresees several kind of handshake communication. Remind that the handshakes manage only the asynchronous communication, where the two systems don’t have any temporal correlation, in I2C BUS communication, systems are already synchronized by the communication itself.
The foreseen technicals are:
- Handshake HW: it is based on a couple of hardware signals which join the communication ones, always available on the CN1 connector.
  RTSxxxx SDI 02 -> CTS external system
  When the SDI 02 is ready to get data, it activates its RTS and its external system subordinates the transmission to the status of its active CTS.
  CTS xxxx SDI 02 <- RTS external system
  When the external system is ready to get data, it activates its RTS and the SDI 02 subordinates the transmission to the status of its active CTS.

- Handshake SW: it is based on the characters XON (code 17=11H) and XOFF (code 19=13H) which are added to the communication.
  SDI 02 -> external system
  When the SDI 02 can’t get any data it transmits XOFF, followed by XON when it is ready again. The external system subordinates its transmission to the last XON, XOFF received status.
  External system -> SDI 02
  When the external system can’t get any data, it transmits XOFF followed by XON when it is ready again. The external system subordinated its transmission to the last XON, XOFF received status.

- SW Repeated Handshakes: the condition is similar to the software handshakes that we have previously described, but it is transmitted every second by the SDI 02.

The SDI 02 can’t get any data when its reception buffer is 90% full, but it can get data again when the same percentage is lower than the 20%. The small size of reception buffer can often cause a variation of reception status, so there is a handshake variation. The external system has to be ready to recognize and manages all these variations or the communication can works not properly (in some cases the Windows operating system can lose some variations with problems in execution commands). It is important to remember that the described handshakes are managed only if one of the three techniques has been set through the proper configuration command; otherwise other handshakes are not foreseen.
ERRORS

The **SDI 02** checks a several of condition during his work, both in configuration mode and in reception and saving data. Several error condition are included, they meet defects, malfunctions and/or not foreseen status.

The complete list of recognized errors is shown in GET STATUS paragraph that describes the command able to get the total status of **SDI 02**.

When an error is recognized the LED of **SDI 02** become yellow blinking, in order to signalling the wrong condition; this blinking length continues as long as errors are reset with proper command.

In detail to correctly manages errors, user has to perform these following steps:

a) Use the **SDI 02** with necessary commands.

b) When an error occurs, the LED becomes blinking yellow.

c) With the external unit provide the GET STATUS command and examine the activate errors in its response. When the external unit has not this features, the user can replace it with another unit that is able to do it, like a PC.

d) Once seen errors, try to fix them (for example unlock the write protection on memory card, replace the full card, increase communication timeout, etc.).

e) Provide RESET ERRORS command that disables all error status and stop the blinking of the LED and become fixed in red if memory card is inserted or red, green, yellow by turns if the card is not inserted.

f) Re-activate the previous working of external unit in order to take the control of commands of **SDI 02**; if at the point c, the unit was replaced, please, reconnect the original one

Keep in mind that error are rare phenomena when application is check out and installed. The operations listed above normally have to be performed only at beginning phase preparation of **SDI 02**.

Otherwise, if the user want to realize with high certainly, the errors have to be managed by external system: in this case the listed points have to be executed after each command, or at least when the command was not correctly performed. So user can examine the first character of commands response, that meet the executed command flag (please, see the LOGIC PROTOCOL paragraph or the COMMANDS chapter).
NOTE FOR SYNCHRONOUS COMMUNICATION I2C BUS

When is using synchronous communication I2C BUS in both operative modes of **SDI 02** the rules to follow are:

a) The systems that communicates have to follow the rules of I2C BUS standard protocol detailed described in the document "THE I2C-BUS SPECIFICATIONS", from PHILIPS semiconductors.

b) The **SDI 02** always works as Slave, in particular as Slave receive when reception and saving data mode is selected and as Slave receive+transmit when configuration mode is selected. The external unit has to works as Master transmit to send data da save or configuration commands and as Master receive to receive the configuration commands response.

c) Local network communicartion is possible both with unit of the same type and different type as shown in figures 16 and 27.

d) I2C BUS communication has a synchronism between the two communication systems, as shown in figures 24 and 25. So is not necessary none handshake used only in asynchronous communication.

e) Each communication interest only the used slave address of **SDI 02** that meet the express one in work parameter sla.add. In case of a I2C BUS communication network, each **SDI 02**, have to be set with a different slave, different address and different from other ones of I2C BUS device potential present in the same network.

f) In order to semplify the management, the first data give back from **SDI 02**, always meet the number of character of response, that is the number of data that the external system has to receive. The last one has to finish comunication with a STOP sequence, after all data reception.

g) In single communication between external unit and **SDI 02** can be trasferred several data to save, taking care to doen't fill up the reception buffer.

h) The slave address defined in local setup is 7 bit but is set to 8 bit with less significant bit (R/W) set to 0: it is possible to use the 128 different values even, in range 00H÷FEH.

i) In case of I2C BUS network connection perform configuration described in paragraph CN1 SERIAL COMINICATION LINE AND SWITCHING CONNECTOR making sure that the line is correctly terminated (see figure 16).

j) The **SDI 02** does not support I2C BUS protocol extension (addressing 10 bits, fast communication, etc.) and reserved slave address; these thing has not to be used by user in the external unit.
FIGURE 27: NETWORK OF I2C BUS DIFFERENT DEVICES

TIMEOUT IN COMMUNICATION

To avoid long waiting during the communication there is the tout work parameter that ensures that SDI 02 is always operative. The tout value can be define and acquire by proper configuration.
command and it is express in twenties of milliseconds, as described in referred paragraph. In correspondence to any command in communication, the SDI 02 actives a counter of elapsed time and during the waithing it verify if the counter got the timeout parameter, if enabled. In case it is disabled it continue to waiting, viceversa step to following operation and set the expired timeout error. The timeout is check for each communication data, so if SDI 02 has to wait several events, it will be operative again after a complessive time equal to the product of two ones. When the SDI 02 communication is managed manually by an operator we recommend to disable timeout or to set it to a very high time.

FILES DATE AND TIME

Among the features of SDI 02 used files on memory card, we remind the reference of time of creation and last modify. Both these times are composed of date and timethat assume different values in according with the needed configuration:

SDI 02 -> Data and time in according to files is created by firmware with "increasing" method: at power on of SDI 02 it is set the initial time 01/01/1980 00:00:00 that is increased of 2 seconds in correspondence with each files access. So each access has a different time a will have a time reference for all files and all performed operation.

SDI 02.RTC -> Data and time in according to files is those ones get from the back up clock in real time (Real time clock) that manages hours, minutes, seconds, days, months and year. If needed this clock can be set and/or get through proper commands. For work with clock in absence od power supply too, we recomend to see the BACK UP CLOCK paragraph.

The association between date and time is managed by SDI 02 and provides to user a time reference in according to used data files: the external system that use them, will can perform some check of elapsed time, get the creation and modify time, calculate production data in a range of time, list the files, etc.

TYPICAL APPLICATIONS

The SDI 02 is the ideal device for those application where you have to manage big amount of data that can pass on one of serial communication. The following points lists some of these typical using:

-Data collection system that acquire digital/analog signals from the field and save them on memory card. Among these signals we can remind those for civil and industrial sectors, as temperatures, pressures, limit switch, voltages, currents, frequencies, positions, speed, levels, alarms, etc.
-Configuration data exchanges between systems and plants already installed.
-Update and relocation of working parameters, work programs, and production data.
-Memory enlargement for main system connected.
-Connection to a PLC or other programmable control systems.
-Connection to a programmable testers.
-Direct mounting on automatic machines.
- Application field can be enlarged with proper firmwares developed in base of customer specifics. These ones can requested to grifo® directly.

DEMO PROGRAMS

For SDI 02 module are available demo programs both in source level and executable level; these programs can be used without no changes for a first check out of the product and than modified, or used in pieces, to sodisfy user's requirements. These programs are available for more high level programming languages (C, PASCAL, BASIC, etc.) both for PC and microprocessor GPC® or GMM board of grifo® family. User can examine the comments and decide if test them or not. All demo programs can be directly used or modified by user with no any authorization or added costs.
In case of particular requirements or use combination you can requires specific demo programs in accordance with grifo®.
Moreover is available a PC program, called Demo_SD102, that allow to manage configuration modality of the board, with a normal PC connected to asynchronous serial line RS 232. The purpouse of this program is easy and intuitive, but for who need help, we recommend to see HOW TO START chapter, where there are an example use.
COMMANDS

In this chapter are described all available commands in configuration mode of SDI 02 with relative input and output parameters. Commands are divided in subgroups in according with their function and for each one is made a triple description: mnemonic one, through ASCII characters and the numeric one express both in decimal and exadecimal format.

All commands and associated parameters are coded with rules already described in LOGIC PROTOCOL paragraph and summarized below:

a) The external system transmits the command and the SDI 02 give back a response.
b) Potential command and response parameters always are coded in ASCII.
c) If the parameter is numeric it can be used the decimal code, with only significant numbers.
d) If the command or the response include more parameters, these are separated by space.
e) In the follow description is not give back the performed command flag that meet the first data response, with the following code:

   0 (48=30H) -> not performed command: the rest of response don't give back;
   1 (49=31H) -> command performed correctly: the rest of response give back.
f) In the following description there aren't data to recognize the end of communication, that is:

   CR (13=0DH) -> at the end of second command with asynchronous communication;
   CRLF> -> at the end of response with asynchronous communication. This
   (13,10,62=)
   0DH,0AH,3EH) from a terminal emulator; <
   bytes number -> at the begin of response with synchronous communication,
   of response> the performed command flag.

g) In the following descriptions there aren't handshaking data of communication (XON,XOFF) that can exchanged when asynchronous communication is selected and software handshake are active.
h) Some commands that have to exchange large amounts of data don't use parameters but use exceptions of logic protocol. These exceptions meet adding communications, of prefixed length, located among the command or response parameters. For further information, please see the paragraphs that describe those commands.
i) The command parameters that meet the path and/or the file or folder name, have to be in 8.3 format, that is 8 characters for the name and 3 for the extension. The valid characters are those numeric (0÷9), and alphabetic (A÷Z, a÷z) and separation ones (_,-).

For a quickly search of available commands we recommend to see summarizing tables in APPENDIX A. In figure B1 there are default settings of all configuration commands described in this chapter. During the management of a command (reception, execution and response transmission) the SDI 02 keeps the LED fixed in yellow.
GENERAL COMMANDS

Following we are relative commands to current status of SDI 02, its potential errors, real time clock and version.

GET VERSIONS

Command: v  
Command Dec: 118  
Command Hex: 76  
Response: ver.hw ver.fw  
Response Dec: ver.hw 32 ver.fw  
Response Hex: ver.hw 20 ver.fw

The command give back hardware and firmware versions of SDI 02 like ASCII strings. The two strings have the following constant format:

- ver.hw -> 6 numeric characters;
- ver.fw -> 3 numeric characters in unit.tens format.

For examples as shown in HARDWARE, FIRMWARE VERSION beginning chapter are give back the following characters:

110907 4.2 or
49 49 48 57 48 55 32 52 46 49 or
31 31 30 39 30 37 20 34 2E 31 Hex

RESET ERRORS

Command: Z  
Command Dec: 90  
Command Hex: 5A  
Response:  
Response Dec:  
Response Hex:

This command reset all error managed from SDI 02. The reset can be verified with GET STATUS command, there is the complete list of errors too.

The command don't give back none response parameter.

GET STATUS

Command: z  
Command Dec: 122  
Command Hex: 7A  
Response: general.status disk.status  
Response Dec: general.status 32 disk.status  
Response Hex: general.status 20 disk.status

This command give back the SDI 02 status as two ASCII strings. These strings has numeric format with variable length of status and current errors with the following relation to bit:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>dec. weight</th>
<th>Hex Weight</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>general status.00</td>
<td>00001</td>
<td>0001</td>
<td>Communication overflow error.</td>
</tr>
<tr>
<td>general status.01</td>
<td>00002</td>
<td>0002</td>
<td>Parity error in asynchronous line.</td>
</tr>
<tr>
<td>general status.02</td>
<td>00004</td>
<td>0004</td>
<td>Status error in synchronous line.</td>
</tr>
<tr>
<td>general status.03</td>
<td>00008</td>
<td>0008</td>
<td>Invalid communication sequence error.</td>
</tr>
<tr>
<td>general status.04</td>
<td>00016</td>
<td>0010</td>
<td>RTC access error.</td>
</tr>
<tr>
<td>general status.05</td>
<td>00032</td>
<td>0020</td>
<td>Invalid configuration error (set work parameter with default values).</td>
</tr>
<tr>
<td>general status.06</td>
<td>00064</td>
<td>0040</td>
<td>Not recognized command error.</td>
</tr>
<tr>
<td>general status.07</td>
<td>00128</td>
<td>0080</td>
<td>Command parameters error.</td>
</tr>
<tr>
<td>general status.08</td>
<td>00256</td>
<td>0100</td>
<td>Memory card present.</td>
</tr>
<tr>
<td>general status.09</td>
<td>00512</td>
<td>0200</td>
<td>Memory card write protect (WP selector enabled).</td>
</tr>
<tr>
<td>general status.10</td>
<td>01024</td>
<td>0400</td>
<td>Configuration mode active (J1 in 1-2 at power on).</td>
</tr>
<tr>
<td>general status.11</td>
<td>02048</td>
<td>0800</td>
<td>Elapsed Timeout of communication.</td>
</tr>
<tr>
<td>general status.12</td>
<td>04096</td>
<td>1000</td>
<td>Not used.</td>
</tr>
<tr>
<td>general status.13</td>
<td>08192</td>
<td>2000</td>
<td>Not used.</td>
</tr>
<tr>
<td>general status.14</td>
<td>16384</td>
<td>4000</td>
<td>Not used.</td>
</tr>
<tr>
<td>general status.15</td>
<td>32768</td>
<td>8000</td>
<td>Not used.</td>
</tr>
<tr>
<td>disk status.00</td>
<td>00001</td>
<td>0001</td>
<td>Disk initialization error.</td>
</tr>
<tr>
<td>disk status.01</td>
<td>00002</td>
<td>0002</td>
<td>Disk formatting error.</td>
</tr>
<tr>
<td>disk status.02</td>
<td>00004</td>
<td>0004</td>
<td>Creating folder error.</td>
</tr>
<tr>
<td>disk status.03</td>
<td>00008</td>
<td>0008</td>
<td>Changing folder error.</td>
</tr>
<tr>
<td>disk status.04</td>
<td>00016</td>
<td>0010</td>
<td>Removing folder error.</td>
</tr>
<tr>
<td>disk status.05</td>
<td>00032</td>
<td>0020</td>
<td>File information error.</td>
</tr>
<tr>
<td>disk status.06</td>
<td>00064</td>
<td>0040</td>
<td>Deleting file error.</td>
</tr>
<tr>
<td>disk status.07</td>
<td>00128</td>
<td>0080</td>
<td>Rename file error.</td>
</tr>
<tr>
<td>disk status.08</td>
<td>00256</td>
<td>0100</td>
<td>Not valid management error.</td>
</tr>
<tr>
<td>disk status.09</td>
<td>00512</td>
<td>0200</td>
<td>Opening file error.</td>
</tr>
<tr>
<td>disk status.10</td>
<td>01024</td>
<td>0400</td>
<td>Reading file error.</td>
</tr>
<tr>
<td>disk status.11</td>
<td>02048</td>
<td>0800</td>
<td>Writing file error.</td>
</tr>
<tr>
<td>disk status.12</td>
<td>04196</td>
<td>1000</td>
<td>Buffer empty on file error.</td>
</tr>
<tr>
<td>disk status.13</td>
<td>08192</td>
<td>2000</td>
<td>Position file error.</td>
</tr>
<tr>
<td>disk status.14</td>
<td>16384</td>
<td>4000</td>
<td>Closing file error.</td>
</tr>
<tr>
<td>disk status.15</td>
<td>32768</td>
<td>8000</td>
<td>Writing on protected disk error.</td>
</tr>
</tbody>
</table>

**Figure 30: Status and Errors Parameters**

Keep in mind that error bit are set by **SDI 02**, during its work and reset only by its **RESET ERRORS** proper command. In this mode every error is memorized and kept as long as the user reset it, after get it, as described in **ERRORS** paragraph. In previous description was used the simply disk term to indicate used memory card, for PC analogies that manages it like a disk.

If the **SDI 02** is in configuration mode, with a memory card protected inserted with already data received but non saved because of the protection, when receives this command gives back the following response:

```
1792 32768
49 55 57 50 32 51 50 55 54 56
31 37 39 32 20 33 32 37 36 38
```

or

```
Hex
```

or
SET REAL TIME CLOCK

**Command:** \( T \) gg/mm/aaaa oo:mm:ss \n**Response:**

**Command Dec:** 84 32 dd 47 mm 47 yyyy 32 hh 58 mm 58 ss \n**Response Dec:**

**Command Hex:** 54 20 dd 2F mm 2F yyyy 20 hh 3A mm 3A ss \n**Response Hex:**

The command set the real time clock of SDI 02, with contained data in passed parameters. Following are reported in detail the means of 6 values, divided in two parameters, with the relative range of validity:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
<th>Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>dd</td>
<td>1÷31</td>
<td>Day</td>
</tr>
<tr>
<td>mm</td>
<td>1÷12</td>
<td>Month</td>
</tr>
<tr>
<td>yyyy</td>
<td>0÷9999</td>
<td>Year</td>
</tr>
<tr>
<td>hh</td>
<td>0÷23</td>
<td>Hours</td>
</tr>
<tr>
<td>mm</td>
<td>0÷59</td>
<td>Minutes</td>
</tr>
<tr>
<td>ss</td>
<td>0÷59</td>
<td>Seconds</td>
</tr>
</tbody>
</table>

**Figure 31: Real time clock parameters**

The command is always performed, if a parameter has a value out of range too: the external system have guarantee the goodness of values. This command can be only used if the option SDI 02.RTC is ordered, that is with the clock option and the presence of SRAM; otherwise if there is not this option, the command will not performed and will set the clock (RTC) access or back up RAM error (general status.04 ->00016=0010H).

In order to guarantee the update of clock, without power supply too, please see BACK UP REAL TIME CLOCK and SRAM paragraph. The main function of clock is to provide current date and time used to define the creating and modify time of files, as described in DATE AND TIME FILES paragraph. FOR the external system without real time clock, it can use the clock of SDI 02 to save current time in files, with acquired data. For examples to set the clock to midday of July the first 2008, the user has to provide the following command:

\( T\) 01/07/2008 12:00:00

**Note:** Because of the year with four numbers managed by clock has to be always valid, it is necessary to provide the set clock command, once every 4 years.

GET REAL TIME CLOCK

**Command:** \( t \) \n**Response:** dd/mm/yyyy hh:mm:ss

**Command Dec:** 116 \n**Response Dec:** dd 47 mm 47 yyyy hh 58 mm 58 ss

**Command Hex:** 74 \n**Response Hex:** dd 2F mm 2F yyyy hh 3A mm 3A ss

Are reported 6 data, grouped in two parameters, that meet temporal parameters with current date and time, get from the clock of SDI 02.

The means of these parameters are shown in previous figure. This command can be only used if the option SDI 02.RTC is ordered, that is with the clock option and the presence of SRAM; otherwise if there is not this option, the command will not performed and will set the clock (RTC) access or back up RAM error (general status.04 ->00016=0010H).
**Figure 32: Potential Connection Diagram**
WRITE BACK UP SRAM

**Command:** B address byte  
**Response:**

**Command Dec:** 66 32 address 32 byte  
**Response Dec:**

**Command Hex:** 42 20 address 20 byte  
**Response Hex:**

The command writes a location of back up SRAM on SDI 02, using the passed parameters.

The two parameters that execute the command have the following range of validity:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
<th>Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>32÷255</td>
<td>Location address</td>
</tr>
<tr>
<td>byte</td>
<td>0÷255</td>
<td>Data to write</td>
</tr>
</tbody>
</table>

**Figure 33: Back up SRAM parameters**

On the first parameter is verified the validity and, if the passed value is not in the valid range, is set the relative *Command parameters error (general status 07 -> 00128=0080H)*.

The second parameter is not verified and the system and the external system has to guarantee its validity.

This command can only be used if the option SDI 02.RTC is ordered, that is with the clock option and the presence of SRAM; otherwise if there is not this option, the command will not be performed and will set the *clock (RTC) access or back up RAM error (general status 04 -> 00016=0010H)*.

In order to guarantee the update of clock, without power supply too, please see BACK UP REAL TIME CLOCK and SRAM paragraph.

The main function of back up SRAM is to save and give back data the stay on SDI 02 when the memory card is removed and the power supply is off too. For example it can be used for data configuration, identifier data, values that have quick changes, to work with some information to save on memory card, etc.

For example to save the data 85 at the address 100 of SRAM, it has to provide the following command:

B 100 85

READ BACK UP SRAM

**Command:** b address  
**Response:** byte

**Command Dec:** 98 32 address  
**Response Dec:** byte

**Command Hex:** 62 20 address  
**Response Hex:** byte

It gives up one data that meet the saved byte in back up SRAM location write by address parameter.

The means and check on parameters are the same reported in the previous figure.

This command can be only used if the option SDI 02.RTC is ordered, that is with the clock option and the presence of SRAM; otherwise if there is not this option, the command will not be performed and will set the *clock (RTC) access or back up RAM error (general status 04 -> 00016=0010H)*.

For example the SRAM location at address 100 contains the data 85, and to read it, the right command is

B 100

and the response is

85
SET CONFIGURATION

Command:  S ncnf conf  
Command Dec:  83 32 ncnf 32 conf  
Command Hex:  53 20 ncnf 20 conf

Response:
Response Dec:
Response Hex:

The command configures the settings for serial communication between SDI 02 and external system using the passed parameters. The first parameter identify the configuration to perform while the second one the value to assign, as is described below:

<table>
<thead>
<tr>
<th>ncnf</th>
<th>conf</th>
<th>means</th>
</tr>
</thead>
<tbody>
<tr>
<td>C (67=43H)</td>
<td></td>
<td>Selects the used communication line</td>
</tr>
<tr>
<td>R (82=52H)</td>
<td></td>
<td>Asynchronous communication line (RS 232,TTL)</td>
</tr>
<tr>
<td>I (73=49H)</td>
<td></td>
<td>Synchronous communication line (I2C BUS)</td>
</tr>
<tr>
<td>T (84=54H)</td>
<td></td>
<td>Selects waiting timeout communication</td>
</tr>
<tr>
<td>0 (00H)</td>
<td></td>
<td>None waiting</td>
</tr>
<tr>
<td>1÷250 (01÷FAH)</td>
<td></td>
<td>waiting 20÷5000 milliseconds</td>
</tr>
<tr>
<td>255 (FFH)</td>
<td></td>
<td>endless waiting</td>
</tr>
<tr>
<td>0 (66=42H)</td>
<td></td>
<td>Selects baud rate for synchronous communication</td>
</tr>
<tr>
<td>4800</td>
<td></td>
<td>4800 baud</td>
</tr>
<tr>
<td>9600</td>
<td></td>
<td>9600 baud</td>
</tr>
<tr>
<td>19200</td>
<td></td>
<td>19200 baud</td>
</tr>
<tr>
<td>38400</td>
<td></td>
<td>38400 baud</td>
</tr>
<tr>
<td>57600</td>
<td></td>
<td>57600 baud</td>
</tr>
<tr>
<td>115200</td>
<td></td>
<td>115200 baud</td>
</tr>
<tr>
<td>S (83=53H)</td>
<td></td>
<td>Selects stop bit for asynchronous communication</td>
</tr>
<tr>
<td>1 (49=31H)</td>
<td></td>
<td>1 stop bit</td>
</tr>
<tr>
<td>2 (50=32H)</td>
<td></td>
<td>2 stop bit</td>
</tr>
<tr>
<td>P (80=50H)</td>
<td></td>
<td>Selects parity for asynchronous communication</td>
</tr>
<tr>
<td>N (78=4EH)</td>
<td></td>
<td>None parity</td>
</tr>
<tr>
<td>E (69=45H)</td>
<td></td>
<td>Even parity</td>
</tr>
<tr>
<td>O (49=4FH)</td>
<td></td>
<td>Odd parity</td>
</tr>
<tr>
<td>H (72=48H)</td>
<td></td>
<td>Selects handshake for asynchronous communication</td>
</tr>
<tr>
<td>N (78=4EH)</td>
<td></td>
<td>None handshake</td>
</tr>
<tr>
<td>H (72=48H)</td>
<td></td>
<td>Handshake Hardware (RTS, CTS)</td>
</tr>
<tr>
<td>S (83=53H)</td>
<td></td>
<td>Handshake Software 4XON, XOFF</td>
</tr>
<tr>
<td>A (65=41H)</td>
<td></td>
<td>Selects slave address for synchronous communication</td>
</tr>
<tr>
<td>0÷254 (00÷FEH)</td>
<td></td>
<td>Recognizing address (even values only)</td>
</tr>
</tbody>
</table>

**FIGURE 34: CONFIGURATION PARAMETERS**

The command is always performed, if the parameter is has a out fo range value not valid too: the external system has to guarantee the validity.

In order to storage these settings from casual modify, the command is recognized and performed if the SDI 02 is enabled in configuration mode only, that is in correspondence of the last switch on the jumper J1 was in 1-2 position. To configure all parameters, the command has to be provided 7 times, using the relative 7 identifiers; at the end of this operation the jumper J1 has to be in 2-3 position and re-switch on the SDI 02 board. For example to set the speed at 38400 baud, the command to perform is:

**S B 38400**
GET CONFIGURATIONS

Command:  s  
Response:  C=com.line  T=tout  B=baud  S=stop  
P=parity  H=handshake  A=sla.add

Command Dec:  115  
Response Dec:  67 61 com.line 3284 61 tout 32 66 61  
baud 32 83 61 stop 32 80 61 parity 32 72 61  
handshake 32 65 61 sla.add

Command Hex:  73  
Response Hex:  43 3D com.line 20 54 3D tout 20 42 3D  
baud 20 53 3D stop 20 50 3D parity 20 46 3D  
handshake 20 41 3D sla.add

Are reported 7 parameters that meet the current configuration of SDI 02 for the serial communication with the external system.

In order to make easy the parameters recognizing are reported with a prefix that meet the same identifier; these identifier and the potential values of each parameter are reported in previous figure.

This command reports all configuration in one calling.

For example with the default configuration that show in appendix B, and the response is:

C=R  T=250  B=19200  P=N  S=1  H=N  A=128
DISK COMMANDS

Followind are reported, the commands for the memory card (called disk, because is similar to PC) management, inserted in SDI 02. Among them there are those relative to disk folder, to formatting and other features.

GET DISK FEATURES

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>size free label protection n.series</td>
</tr>
<tr>
<td>68</td>
<td>size 32 free 32 label 32 protection 32 n.series</td>
</tr>
<tr>
<td>44</td>
<td>size 20 free 20 label 20 protection 20 n.series</td>
</tr>
</tbody>
</table>

Are reported 5 parameters that provide information on disk (=memory card) actually inserted in SDI 02.

The means of these parameters is following described:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>Disk global size in kbytes (the value is reported with the suffix K to avoid errors in reading provided values)</td>
</tr>
<tr>
<td>free</td>
<td>Disk global free space in Kbytes (the value is reported with the suffix K to avoid errors in reading of provided value)</td>
</tr>
<tr>
<td>label</td>
<td>String with labeled or disk name, defined on PC. This parameter is reported only if it is selected the main folder of the disk.</td>
</tr>
<tr>
<td>protection</td>
<td>Writing protection status of the disk (0=unlocked, 1=locked).</td>
</tr>
<tr>
<td>n.series</td>
<td>Series or disk production number.</td>
</tr>
</tbody>
</table>

**FIGURE 35: DISK FEATURES PARAMETERS**

If the memory card is not inserted on SDI 02, the command is not performed and none parameter is reported. Viceversa the time of execution and restoring of parameters can have some seconds endurance.

For example, with one SD of 1 Gigabyte, unlocked, write for one third exactly, with label as OVEN_12 is reported with the same response:

1005696K 670464K OVEN_12 0 1894276491

FORMAT DISK

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>FU¬</td>
<td></td>
</tr>
<tr>
<td>70 85 170</td>
<td></td>
</tr>
<tr>
<td>46 55 AA</td>
<td></td>
</tr>
</tbody>
</table>

The command format the disk (=memory card) currently inserted on SDI 02.

The command does not report response parameter and is performed only the memory card is inserted on SDI 02.

Keep in mind that the command perform a logic format not physical, so there is the total lost of data already saved on disk.
CREATE FOLDER

**Command:** M path  
**Response:**
**Command Dec:** 77 32 path  
**Response Dec:**
**Command Hex:** 4D 20 path  
**Response Hex:**
The command creates one folder on a disk (= memory card) currently inserted on **SDI 02**, with name and the path specified in the parameter.
The command does not report response parameter and is performed only the memory card is inserted on **SDI 02** and is not writing protected.
The **path** parameter meet the string that defines folders with "\" symbol, has max length of 8 characters for each folder and 32 characters max. for the all the path.
When the passed path does not respect the 8.3 rules, is converted in the following format; in detail are kept the the first 6 characters, added the separation character "~" and one progressive value: for example "DATA_FOLDER" become "FOLDER~1", "FOLDER_PARAMETERS" become "FOLDER~2".

In the **path** parameter can be used typical indications for manage file for folder changing:
- .  -> stay in the current folder
- .. -> come back to previous level folder
- \  -> come back to disk main folder

For example to select the folder called LINE1, it has to provide the following command:

```
M LINE1
```

CHANGE FOLDER

**Command:** P path  
**Response:**
**Command Dec:** 80 32 path  
**Response Dec:**
**Command Hex:** 50 20 path  
**Response Hex:**
The command changes the selected folder on disk (= memory card) currently inserted on **SDI 02**, with the one specified in the parameter. The chosen folder is used from several other commands, like GET FILES LIST that reports the content of the current folder.
The command does not report response parameter and is performed only the memory card is inserted on **SDI 02**.
The **path** parameter meet the string that defines folders with "\" symbol, has max length of 8 characters for each folder and 32 characters max. for the all the path.
When the passed path does not respect the 8.3 rules, is converted in the following format; in detail are kept the the first 6 characters, added the separation character "~" and one progressive value: for example "DATA_FOLDER" become "FOLDER~1", "FOLDER_PARAMETERS" become "FOLDER~2".

In the **path** parameter can be used typical indications for manage file for folder changing:
- .  -> stay in the current folder
- .. -> come back to previous level folder
- \  -> come back to disk main folder

For example to select the folder called LINE1, it has to provide the following command:

```
P LINE1
```

to select the main folder, it has to provide the following command:

```
P\``
REMOVE FOLDER

Command:   K path  
Command Dec:  75 32 path  
Command Hex:  4B 20 path  
Response:  
Response Dec:  
Response Hex:  

The command deletes the folder on a disk (=memory card) currently inserted in **SDI 02**, with name and the path specified in the parameter:

The command does not report response parameter and is performed only the memory card is inserted on **SDI 02** and is not writing protected, the folder to delete empty and if the folder is not the current one.

The **path** parameter meet the string that defines folders with "\" symbol, has max length of 8 characters for each folder and 32 characters max. for the all the path

When the passed path does not respect the 8.3 rules, is converted in the following format; in detail are kept the the first 6 characters, added the separation character "~" and one progressive value: for example "DATA_FOLDER" become "FOLDER~1", "FOLDER_PARAMETERS" become "FOLDER~2".

For example to select the folder called LINE1, it has to provide the following command:

K LINE1
COMMANDS FOR FILES

Followind are reported, the commands for management of files saved on memory card inserted on SDI 02. Among them we remind those ones for creation, deleting, writing, reading, list of file, etc.

GET FILES LIST

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
<th>nfiles</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td></td>
<td>file1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>file2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>filen</td>
</tr>
</tbody>
</table>

Command Dec: 76
Response Dec:
- nfiles 08 file1 13 10 08 file2
- 13 10 .........08 filen 13 10

Command Hex: 4B 20 path
Response Hex:
- nfiles 08 file1 0D 0A 08 file2
- 0D 0A .........08 filen 0D 0A

The command report the list of present files name and subfolder in the current folder of disk (=memory card) inserted on SDI 02.
The command hasn't input parameters and it is performed only if the memory card is inserted on SDI 02.
The names filex reported represent an exception to logic protocol in fact they aren't in the response to command and are reported after the real response nfiles transmission of SDI 02, with the number of files and folders. In detail for each name is reported a string of 15 characters with:
- ranging TAB;
- file or folder, up to 8 characters length;
- separation .;
- file extension, up to 3 characters length;
- CRLF.

In case of asynchronous communication RS232 or TTL, the characters will arrive immediately and the external system, that know which are made by nfiles*15, have to wait them.
In case of synchronous communication those characters have to be received with a new I2C BUS communication of the external system that read all the nfiles*15 characters, always in master receive mode. The names of folders are reported with the name closed to square brackets for a global 10 (8+2) characters max., that are followed by two filling spaces at least.
Naturally if the current board does not contain files and subfolders (nfile=0) after the command response there are no transmissions (none characters to receive with the asynchronous communication or none new synchronous communication).
An example of response of command is when the current folder contains 3 files and one folder, complete of exception can be:

```plaintext
4
TAIR.TXT
TOVEN.TXT
TFLAME.TXT
[TESTS]
```
GET FILES INFORMATION

Command: I path
Command Dec: 73 32 path
Command Hex: 49 20 path

Response: size.t.creation.t.modify attributes
Response Dec: size 32 t.creation 32 t.modify 32 attributes
Response Hex: size 20 t.creation 20 t.modify 20 attributes

The command reports the current files information of the disk (=memory card) inserted on **SDI 02**, with the name and the path specified in the parameter.

The command is performed only if the memory card is inserted on **SDI 02** and the that file exists.

In this case the 4 parameters returned have the following means:

- **size** -> Size of file in bytes.
- **t.creation** -> Date and time of creation in dd/mm/yyyy-hh:mm:ss format.
- **t.modify** -> Date and time of the last modify in dd/mm/yyyy-hh:mm:ss format.
- **attributes** -> String with files attributes.
  
  R= only reading 
  H= hided 
  A=archives 
  S= system 
  that can be combined among them.

**Figure 37: File information parameters**

Keep in mind that the time of creation and modify file, are reported in one parameter with the indication date and time divided by "-", in order with the rules of logic protocol that use the space as parameter separator. More info on time value are available in the DATE AND TIME OF FILES paragraphs.

The **path** parameter meet the string that defines folders with "\" symbol, has max length of 8 characters for each folder and 32 characters max. for the all the path.

When the passed path does not respect the 8.3 rules, is converted in the following format; in detail are kept the the first 6 characters, added the separation character "~" and one progressive value: for example "PARAMETERS\WORK_PARAMETERS.DAT" become "PARAME~1\WORK_P~2.DAT", for example, providing an archives file, 1 KBytes length, called **SETTINGS.TXT**, saved in the current folder of the inserted SD:

**I SETTINGS.TXT**

the potential response is:

1024 01/07/2008-12:00:00 05/08/2008 14:10:30 A
ERASE FILE

Command:  E path
Command Dec:  69 32 path
Command Hex:  45 20 path

The command erases the file present in the disk (=memory card) currently inserted in SDI 02, with the name and the path specified in the parameter.
The command is performed only if the memory card is inserted on SDI 02, the writing protection is not activate and the that file exists.
The path parameter meet the string that defines folders with "\" symbol, has max length of 8 characters for each folder and 32 characters max. for the all the path
When the passed path does not respect the 8.3 rules, is converted in the following format; in detail are kept the the first 6 characters, added the separation character "~" and one progressive value: for example "PARAMETERS\WORK_PARAMETERS.DAT" become "PARAME~1\WORK_P~2.DAT", for example, providing an archives file, 1 KBytes length, called SETTINGS.TXT, saved in the current folder of the inserted SD and the command is:

E SETTINGS.TXT

RENAME FILE

Command:  X prev.name new.name
Command Dec:  69 32 prev.name 32 new.name
Command Hex:  45 20 prev.name 20 new.name

The command rename the present file in the disk (=memory card) currently inserted on SDI 02, the specified name in the first parameter with the specified name in the second parameter.
The command is performed only if the memory card is inserted on SDI 02 and the file to rename already exists.
The path parameter meet the string that defines folders with "\" symbol, has max length of 8 characters for each folder and 32 characters max. for the all the path
When the passed path does not respect the 8.3 rules, is converted in the following format; in detail are kept the the first 6 characters, added the separation character "~" and one progressive value: for example "WORK_PARAMETERS.DAT" become "WORK_P~1.DAT".
For example to rename the SETTINGS.TXT file with the PREVSET.TXT name, both saved in the current folder of SD card inserted, we have to provide the command:

X SETTINGS.TXT PREVSET.TXT
GET AVAILABLE FILE MANAGER

Command: A  
Command Dec: 65  
Command Hex: 41 

Response: man.file  
Response Dec: man.file  
Response Hex: man.file 

The command reports the number of the first free file manager in the disk (=memory card) currently inserted on SDI 02.

The function of this manager is to make easy and quick the file identification, in fact before is assigned to a file through its path, and than used by the following commands.

So the user and the external system can use and communicate the single character of numeric value reported, instead of several numeric characters of path that identifies the file itself.

The command is always executed and reports the **man.file** parameter with the following means:

<table>
<thead>
<tr>
<th>Value</th>
<th>Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-&gt; None file manager available</td>
</tr>
<tr>
<td>1</td>
<td>-&gt; First file manager available</td>
</tr>
<tr>
<td>2</td>
<td>-&gt; Second file manager available</td>
</tr>
</tbody>
</table>

**Figure 38: File manager parameter**

Currently the number of file managers is limited to two that is the max. number of files that **SDI 02** can use in the same time, as described in GENERAL FEATURES paragraph. For the first view this one can seem a limit, but with a little of experience, it can be develop a lot of applications without worries.
OPEN FILE

Command: O man.file path mode attributes
Response:
Command Dec: 79 32 man.file 32 path 32 mode 32 attributes
Response Dec:
Command Hex: 4F 20 man.file 20 path 20 mode 20 attributes
Response Hex:

The command opens or creates one file in the disk (=memory card) currently inserted on SDI 02, with passed parameters, that can has the following means:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>man.file</td>
<td>Free file manager that works with the file to open; the value of this parameter can be the that one reported by the previous command. The SDI 02, check the man.file parameter verifying that it is lower than the referred max. number of managed files. If these conditions are not respected the command is not performed and the relative not valid management error is set (disk.status.08-&gt;00256=0100H).</td>
</tr>
<tr>
<td>path</td>
<td>String with path and file name to open. In the parameter the potential folders are delimited by &quot;&quot; symbol, has max length of 8 characters for each folder and 32 characters max. for the all path. When the passed path does not respect the 8.3 rules, is converted in the following format; in detail are kept the first 6 characters, added the separation character &quot;<del>&quot; and one progressive value: for example &quot;WORK_PARAMETERS.DAT&quot; become &quot;WORK_P</del>1\WORK_P~2.DAT&quot;,</td>
</tr>
<tr>
<td>mode</td>
<td>Specifies the kind of file opening and managing it has to use, with the following correspondence: R (82,52H) = Opens an existing file for reading operation only from the beginning; W (87,57H) = Opens an existing file for writing operation only from the beginning; A (65,41H) = Opens an existing file to adding data (both in reading and in writing), from the end; C (67,43H) = Creates a new file for reading and writing operation from the beginning.</td>
</tr>
<tr>
<td>attributes</td>
<td>This parameter assumes means only when the command creates a file (C mode) and defines the attributes for the creation, with the following correspondence: R (82,52H) = Reading only file; H (72,48H) = Hidden file; A (65,41H) = Archives file; S (83,53H) = System file. The SDI 02 combines the attributes only in the GET FILE INFORMATION command while is creating an hidden file it has only one attribute.</td>
</tr>
</tbody>
</table>

Figure 40: Open file parameters
For example to create a file called TAIR.TXT, inside the existing folder LINE1, with archives attribute and connected to file 1 manager, we have to provide this command:

```
O 1\LINE1\TAIR.TXT C A
```

**READ FILE**

**Command:**  
```plaintext
R man.file n.bytes address
```

**Response:**  
```plaintext
byte1...byten
```

**Command Dec:**  
`82 32 man.file 32 n.bytes 32 address`

**Response Dec:**  
`byte1...byten`

**Command Hex:**  
`4F 20 man.file 20 n.bytes 20 address`

**Response Hex:**  
`byte1...byten`

The command read from a file in the disk (=memory card) currently inserted on SDI 02, the data series in according to the passed parameters, that have the following means:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>man.file</td>
<td>File manager in according to an open file: the value of this parameter can be the that one reported by the previous command. The SDI 02, check the man.file parameter verifying that is low to the referred max. number of managed files. If these conditions are not respected the command is not performed and the relative not valid management error is set (disk.status.08-&gt;00256=0100H).</td>
</tr>
<tr>
<td>n.bytes</td>
<td>Number of bytes to read from the file, variable in the range 1÷65535.</td>
</tr>
<tr>
<td>address</td>
<td>Specifies the reading start address from file, variable in the range 0÷file length. The start of addresses from 0 can access correctly to blank files.</td>
</tr>
</tbody>
</table>

If the address parameter is high to current file size, the command is not performed and the relative Position file error (disk.status.13_>08192=2000H) is set.

**Figura 41: Read file parameters**

If the memory card is inserted and the parameters are valid, the command is performed and the **SDI 02** report the bytes read from file. These ones are and exception to logic protocol, in fact they are not in the response and are reported after the **SDI 02** finished to transmit its command response performed correctly. In case of asynchronous communication RS 232 or TTL these transmission of n.bytes, have to wait all of them. In case of synchronous communication these bytes have to be received with a new communication I2C BUS of external system, that reads all the n.bytes. Otherwise the read command is not performed, after the flag with the execution status, none characters are reported.

Read byte from file are not in ASCII format but are in pure binary format, so each character meet to one byte.

If the command READ FILE is used for read a huge quantity of data, these ones are trasmitter to the external system, with the max. speed for the selected physical protocol. If the external system is not able to receive them we can follow two ways: we can provide more reading commands with a low number of data, or enable the handshakes.

For example to read the first 5 bytes from file ASCII in according to manager 1, that contain the string "0123456789", the command to provide is:

```
R 1 5 0
```

and the **SDI 02** will report before the flag of executed command and then characters:

```
01234
```
WRITE FILE

Command: \texttt{W man.file n.bytes address}  
\textbf{Response: n.bytes.written byte1...byten}

Command Dec: 87 32 man.file 32 n.bytes 32 address  
\textbf{Response Dec;n.bytes.written byte1...byten}

Command Hex: 57 20 man.file 20 n.bytes 20 address  
\textbf{Response Hex;n.bytes.written byte1...byten}

The command writes in the file of the disk(=memory card) currently inserted on \textbf{SDI 02}, a series of data in according with passed parameters, that have the following means:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{man.file}</td>
<td>File manager in according to an open file: the value of this parameter can be the that one reported by the previous command. The \textbf{SDI 02}, check the \texttt{man.file} parameter verifying that is low to the referred max. number of managed files. If these conditions are not respected the command is not performed and the relative \texttt{not valid management error} is set (disk.status.08-&gt;00256=0100H).</td>
</tr>
<tr>
<td>\texttt{n.bytes}</td>
<td>Number of bytes to read from the file, variable in the range 1+65535.</td>
</tr>
<tr>
<td>\texttt{address}</td>
<td>Specifies the reading start address from file, variable in the range 0+file length. The start of addresses from 0 can access correctly to blank files. If the \texttt{address} parameter is high to current file size, the command is not performed and the relative \texttt{Position file error} (disk.status.13_&gt;08192=2000H) is set.</td>
</tr>
</tbody>
</table>

\textbf{Figure 42: Write file parameters}

If the memory card is inserted, if it is writing unlocked and the parameters are valid, than the execution of the command works and the \textbf{SDI 02} wait for the bytes to write on file. These ones are an exception to logic protocol in fact they are between command and response, and have to be comunicated after the command transmission of external system. In case of asynchronous communication RS 232 or TTL these transmission of \texttt{n.bytes}, have to wait all of them. In case of synchronous communication these bytes have to be send through a new I2C BUS communication by the external system that send all the \texttt{n.byte}. In this case the \textbf{SDI 02} perform a double verify at the end of data: the first one with the \texttt{n.byte} value received before and the second one from the STOP received by external system.

During the waiting of the data to write the \textbf{SDI 02} check the potential timeout too, as described in the \textbf{COMMUNICATION TIMEOUT} paragraph.

The WRITE FILE command perform the writing when all the \texttt{n.byte} characters are received or when the potential timeout is expired. In bath cases the command reports the \texttt{n.bytes.written} response with the number of bytes received and written.

Written byte from file are not in ASCII format but are in pure binary format, so each character meet to one byte.

If the command WRITE FILE is used for read a huge quantity of data, these ones are received by the external system, with the max. speed for the selected physical protocol. If the \textbf{SDI 02} is not able to receive them we can follow two ways: we can provide more reading commands with a low number of data, or enable the handshakes.
To remove the memory card in every moment, after every WRITE FILE command, the command EMPTY BUFFER ON FILE is needed, in order to guarantee that written data are completely saved in the memory card (see the proper paragraph for details).

For example, to read the first 5 bytes from file ASCII in accordance with manager 1, that contain the string "01234", the command to provide is:

```
W 1 5 0
```

that the data to write:

```
01234
```

at the end, we have to wait the SDI 02 response, that will report the number of written data:

```
5
```

**Figure 43: Data saving application with GAB 844**
EMPTY BUFFER ON FILE

Command: U man.file  
Response:  
Command Dec: 85 32 man.file  
Response Dec:  
Command Hex: 55 20 man.file  
Response Hex:  
The command empty the data buffer writing in the file on the disk (=memory card) currently inserted on SDI 02, through the proper manager.
The man.file parameter has to meet the manager already connected, so has to be the that one used in the command calling for the file opening. The SDI 02, check the man.file parameter verifying that is low to the referred max. number of managed files. If these conditions are not respected the command is not performed and the relative not valid management error is set (disk.status.08->00256=0100H).

If the memory card is inserted and the parameters are valid, the command is performed and the SDI 02 writes physically the saved data in the buffer on file. This buffer is managed by SDI 02 to optimize the accesses to memory card and is empty with this command or with theCLOSE FILE command. For example to to make sure the saving of all written data in the connected file to manager 1, the command to provide is:

U 1

GET FILE POSITION

Command: H man.file  
Response: address  
Command Dec: 72 32 man.file  
Response Dec: address  
Command Hex: 48 20 man.file  
Response Hex: address  
The command reports the current position of disk (=memory card) of file, inserted on SDI 02, set by the specified manager.
The man.file parameter has to meet the manager already connected, so has to be the that one used in the command calling for the file opening. The SDI 02, check the man.file parameter verifying that is low to the referred max. number of managed files. If these conditions are not respected the command is not performed and the relative not valid management error is set (disk.status.08->00256=0100H).

If the memory card is inserted and the parameters are valid, the command is performed and the SDI 02 transmit the response address, that meet the position of current reading and/or writing position in bytes, variable in the range 0+file length. Viceversa the command is not performed and the file position error (disk.status.13 ->08192=2000H).

For example to obtain the position of connected file to manager 1, the command to provide is:

H 1

and if the current position is in the sixth byte of file, the SDI 02 will report the command flag of executed command, and the response:
CLOSE FILE

Command:  C man.file  
Command Dec:  67 32 man.file  
Command Hex:  43 20 man.file  
Response:  
Response Dec:  
Response Hex:  

The command close the file in the disk (=memory card) currently inserted on SDI 02 set by the specified manager. The man.file parameter has to meet the manager already connected, so has to be the that one used in the command calling for the file opening. The SDI 02, check the man.file parameter verifying that it is low to the referred max. number of managed files. If these conditions are not respected the command is not performed and the relative not valid management error is set (disk.status.08- >00256=0100H).

If the memory card is inserted and the parameters are valid, the command is performed and the SDI 02, closes the specified file. Moreover before closing the data buffer is empty too on file, in order to guarantee the saving of all written data (please see the EMPTY BUFFER ON FILE paragraph). At the end the passed file manager, is free because now is not connected to the closed file; this one will become available for the use of new files opening.

For example to close the file connected to manager 1, the command to provide is:

C 1
HOW TO START

In this chapter are described the operation to perform for a first use of S-LOG in linear and quickly mode, without none beginning problem. In particular is reported the right operations sequence that user has to perform for configure and for utilize the product.

To semplify the beginning step, in this chapter we use a minimal systel available for all customers, like a PC with a communication line in RS 232, that perform the Demo_SD02 program.

PRELIMINARY OPERATIONS

A1) Read all the documentation received.

A2) Arrange the SDI 02 for work, find a proper power supply source, verify that jumper configuration is for serial line in RS 232 and the back up battery is connected (see figures 18 and 20) and remove the potentially memory card.
So is restored the default hardware configuration.

A3) Perform the serial connection described in follow figure so connect the two communication signals (TX RS232, RX RS232), reference ground (GND) and handshake signals (CTS RS232, RTS RS232) to a communication port COMx of the PC.

A4) Switch on the Personal Computer.

A5) In case of the PC serial line arrives from (for example USB <-> RS 232 interfaces), perform the proper operations described into the documentation and verify that all works fine.

FIGURE 44: RS 232 connection with PC
CONFIGURATION

B1) Install on PC the Demo_SDI02 program, that is the utility realized by grifo® to quickly configure the SDI 02.

B2) Launch the Demo_SDI02 that at the first switch on is set for english language.

B3) Open the SDI 02 box, select the configuration mode through the jumper J1 in 1-2, and provide power supply (A2 point): LED has to blink in red, green and yellow.

B4) Configure the communication line of PC for default physical protocol of S-LOG, through Program | Serial option. In the window displayed select the serial line connected to A3 point, the reported parameters in follow figure, press the Confirm button and check that the operation is working properly.

![FIGURE 45: PC SERIAL SETTING WITH Demo_SDI02 FOR CONFIGURATION](image)

B5) Verify if communication between PC and SDI 02 works fine, selecting the General | Versions option. With this option is displayed the follow info window with hardware and firmware versione of SDI 02 in use and has to closed through Exit button.
B6) Set the kind of serial communication that SDI 02 has to use to receive data to save, through Configure | Communication option.

In the previous example we supposed that an external system communicate in RS 232 serial line at 38400 Baud, none parity, 1 stop bit and hardware handshake.

In window displayed we have to press before the Get configuration button, than select the proper settings as described in follow figure and at the end press the Set configuration button. At this point we have to wait the saving of all configuration commands that is finished after tens of seconds, verify that no errors are found (parameters on green background) and close the window with the Exit button.
B7) Check and potentially set the current date and time of the clock of **SDI 02** through the *General Clock* option.

B8) Turn off the power supply of **SDI 02**, make disable the configuration mode with the jumper in 2-3 position, and close the board into the container.

B9) At this point the **SDI 02** is completely configurated and ready to work in the application.

**USING**

In this paragraph we realize an application example, so we will check, write and read one memory card with ASCII data. To simplify the external system development we use a PC with a serial line in RS 232. The means of this description is to show the commands, the response and the communicated parameters among the unity.

C1) Connect the communication line of **S-LOG** (see at B6 point), to the external system for the communication.
   In the example of application the RS 232 line is already connected to PC at A3 point.

C2) Supply a voltage to **SDI 02** and verify that the LED is blincking red, green and yellow.

C3) Format one memory card with FAT16 or FAT format, through a PC, insert it into the **SDI 02** and verify that the LED become green.

C4) Initialize the line of the external system with the physical protocol already configured on **SDI 02**. In the proposed application re-confugure the serial line of PC through the *Program Serial* option. In the occured window, select the serial line connected at A3 point, the parameted reported in the following figures, press the *Confirm* button and check that the operation happen correctly.

C5) Acquire the features of the inserted memory card, through the **SDI 02 disk Features**. In the occured window press the *Get features* button and wait for some seconds until there are the total filling of the field below with the relative information.
   In the shown window check the *Writing protection* is disabled, if it is enabeld remove the memory card, move its WP selector, re-insert on **SDI 02** and re-press the button. So press the *Exit* button to close the window.
C6) Perform the formatting of memory card through the Disk SDI02 | Format option. Remind that this command will delete all data saved before. Click the Confirm format button and verify that there are no errors during its working. So close the window with the proper command X on top right of the window.
C7) Acquire again the features of the formatted disk, by repeat the C5 point. In this case the size and the free space havo to meet themselves.

![Diagram of SDI02 disk formatting](image)

**Figure 50: Disk Formatting with SDI 02**

C8) Draw the number of the first manager file available through the File SDI 02 | File manager option and remember it for the following points. In the suggested application pushing the Get manager button has to be shown the first one with the number 1, because none file is associated to managers. In order to make easy the work, Demo_SDIO2 represents one window with the connections among managers and files for each command.

C9) Create one file for save the application data, with the name SDITEST.TXT on main folder of the disk, by the File SDI 02 | Open file option. In the displayed window opening command parameters have to be set, that are the file manager got in the previous point, the file name, the Create (from the beginning) mode and Archive attributes, as described in figure 51. Once clicked the Open button, check that there are no errors, and close the window with Exit button.

C10) Write the string "DATA FROM APPLICATION" on created file, by the File SDI 02 | Write file option. In the shown window writing command parameters have to set, that are the file manager connected to C9 point, the number of byte to write meet the 21 characters of the string, the writing address to 0 to write the string from the beginning of file and data to write, as described in figure 52. Once clicked the Write button, check that the LED of SDI 02 perform a red quick blink, that there are no errors, that all 21 bytes are written and close the window with the Exit button.

C11) Close the written file by the File SDI 02 | Close file option. In the shown window the command parameter has to be set, that is the file manager connected to C9 point. Once clicked the Close button, check that there are no errors and close the window with the Exit button.

C12) Remove the memory card from SDI 02, and make sure that LED is not red, that is a saving in progress.
C13) Insert the memory card on SDI 02 on a PC with an interface for memory card: here there is the file SDITEST.TXT, that contain the written string (C10 point). This string can be examine opening the file with a normal editor, as Notepad of Windows. With the editor add the string "DATA FROM PC" at file, on its second line and press enter to bring the cursor at the beginning of the third line. Save the file, exit from the editor and remove the memory card from the PC.

C14) Re-insert the card on SDI 02 and verify that the LED becomes green fix.
**Figure 52: File writing with Demo_SDIO2**

C15) Draw the present file list from the memory card, through the File SDI 02 | Files list option. In the shown window click the Get list button and check that is present only the TESTSDI.TXT file. At this point close the window with the Exit button.

C16) Draw the TESTSDI.TXT file features with the File SDI 02 | File information option. In the shown window enter the file name, click the Get information button and check that the information are represented of the figure 53. In detail the size and the attributes of the file have to meet, while the creation times can be different. At this point close the window with the Exit button.

C17) Re-open the TESTSDI.TXT file, as in C9 point, but in this case selecting the Read (from the beginning) or Add (from the beginning) mode.
C18) Read the entire file, by the File SDI 02 | Read file option. In the shown window the reading command parameters have to be set, that are the file manager connected to C17 point, the number of bytes to read have to meet the 34 characters got at C16 point, the reading address to 0 for read from the beginning of file as described in figure 54. Once clicked the Read button, check that there are no errors, that both written strings on file are present and close the window with the Exit button.

C19) Re-close the read file as described to C11 point.

C20) Who wishes to test all the functionality of Demo_SDI02 program can also use the other options associated to remaining menu. It is important to note that these tests use all the available commands, so you can value the product. Keep in mind that the utility program Demo_SDI02 only provides the described commands in the previous chapter, in an intelligible modality for users and quick to use. In its main window are also reported the communicated data making a separation among those ones direct from PC to SDI 02 (commands), and those ones from SDI 02 to PC (response). Through this window the user check all the properties of the communication, when it is to realize on another system.

C21) Among the command, in the phase of application development, we remember those ones that report the errors status and SDI 02 errors. These command can be test by the General | Status option and, in the shown windows, clicking the Get status and Reset errors buttons to perform the operations, as described in figure 55. In this window, to see the reset effect, is necessary to perform a new getting status.
C22) At the end of verifying, quit from the **Demo_SD102** and come back to Windows operating system by the *Program | Exit* proper option.

C23) As described in some of previous point, the data written with the **SDI02** can be read by a generic PC and vice versa. In the suggested application the data was ASCII strings but these ones can be any data, and with any format, set by the external system. For example, we can save data on compatible files with programs like EXCEL, so they can be printed, transformed, put on graphic, registered, etc. depend on user's requirements.
**Figure 55:** Errors status management with Demo_SD102
APPENDIX A: COMMANDS SUMMARY TABLES

The following tables list of all the commands recognized by S-LOG. In these tables the command sequences and response are reported in nmemonic format and if the user need the decimal or hexadecimal one can see the paragraphs of the COMMANDS chapter. Please remind that the commands are always identified from the first character and that the following data meet the parameters of the same commands.

<table>
<thead>
<tr>
<th>Command</th>
<th>Communication from main system to SDI 02</th>
<th>Communication from SDI 02 to main system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write back up SRAM</td>
<td>B byte address</td>
<td>-</td>
</tr>
<tr>
<td>Read back up SRAM</td>
<td>b address</td>
<td>byte</td>
</tr>
<tr>
<td>Set Real Time Clock</td>
<td>T dd/mm/yyyy hh:mm:ss</td>
<td>-</td>
</tr>
<tr>
<td>Get Real Time Clock</td>
<td>t dd/mm/yyyy hh:mm:ss</td>
<td></td>
</tr>
<tr>
<td>Reset errors</td>
<td>Z</td>
<td>-</td>
</tr>
<tr>
<td>Get status</td>
<td>z gen.status</td>
<td>disk.status</td>
</tr>
<tr>
<td>Get versions</td>
<td>v ver.hw ver.fw</td>
<td></td>
</tr>
<tr>
<td>Set configuration</td>
<td>S ncnf conf</td>
<td>-</td>
</tr>
<tr>
<td>Get configuration</td>
<td>s com.line tout baud stop parity</td>
<td>handshake sla.add</td>
</tr>
<tr>
<td>Get disk features</td>
<td>D free size protection label n.series</td>
<td></td>
</tr>
<tr>
<td>Format disk</td>
<td>Fchr(85)chr(170)</td>
<td>-</td>
</tr>
<tr>
<td>Create folder</td>
<td>M path</td>
<td>-</td>
</tr>
<tr>
<td>Change folder</td>
<td>P path</td>
<td>-</td>
</tr>
<tr>
<td>Remove folder</td>
<td>K path</td>
<td>-</td>
</tr>
</tbody>
</table>

**Figure A1: Configuration commands table (1 of 2)**
<table>
<thead>
<tr>
<th>Command</th>
<th>Communication from main system to SDI 02</th>
<th>Communication from SDI 02 to main system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get file list</td>
<td>L</td>
<td>les</td>
</tr>
<tr>
<td></td>
<td></td>
<td>File number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>file1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>file2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>filen</td>
</tr>
<tr>
<td>Get file info</td>
<td>I path</td>
<td>size t.creation t.modify attributes</td>
</tr>
<tr>
<td>Delete file</td>
<td>E path</td>
<td>-</td>
</tr>
<tr>
<td>Rename file</td>
<td>X prev.name new.name</td>
<td>-</td>
</tr>
<tr>
<td>Get manager available file</td>
<td>A</td>
<td>man.file</td>
</tr>
<tr>
<td>Open file</td>
<td>O man.file path mode attributes</td>
<td>-</td>
</tr>
<tr>
<td>Read file</td>
<td>R man.file n.bytes address</td>
<td>byte1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>byte2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>byten</td>
</tr>
<tr>
<td>Write file</td>
<td>W man.file n.bytes address</td>
<td>n.bytes.written</td>
</tr>
<tr>
<td></td>
<td>byte1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>byte2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>byten</td>
<td></td>
</tr>
<tr>
<td>Empty buffer on file</td>
<td>U man. file</td>
<td>-</td>
</tr>
<tr>
<td>Get file position</td>
<td>H man.file</td>
<td>address</td>
</tr>
<tr>
<td>Close file</td>
<td>C man.file</td>
<td>-</td>
</tr>
</tbody>
</table>

**Figure A2: Configuration commands table (2 of 2)**
APPENDIX B: DEFAULT CONFIG., OPTIONS, ACCESSORIES

In correspondence of the first purchase, or after a reparation, the SDI 02 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>PARAMETER ID.</th>
<th>DEFAULT SETTING</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.line</td>
<td>C</td>
<td>R</td>
<td>Used communication line = serial asynchronous in RS 232 or TTL</td>
</tr>
<tr>
<td>tout</td>
<td>T</td>
<td>250</td>
<td>Timeout data waiting = 250 * 20 ms = 5000 ms = 5 seconds</td>
</tr>
<tr>
<td>baud</td>
<td>B</td>
<td>19200</td>
<td>Baud rate for serial asynchronous communication RS 232, TTL</td>
</tr>
<tr>
<td>parity</td>
<td>P</td>
<td>N</td>
<td>Parity check for serial asynchronous communication RS 232, TTL = none</td>
</tr>
<tr>
<td>stop</td>
<td>S</td>
<td>1</td>
<td>Stop bit number for serial asynchronous communication RS 232, TTL</td>
</tr>
<tr>
<td>handshake</td>
<td>H</td>
<td>N</td>
<td>Handshakemode management for serial asynchronous communication RS 232, TTL = none</td>
</tr>
<tr>
<td>sla.add</td>
<td>A</td>
<td>128</td>
<td>SDI 02 address for serial synchronous communication I2C BUS</td>
</tr>
</tbody>
</table>

**FIGURE B1: DEFAULT CONFIGURATION TABLE**

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

<table>
<thead>
<tr>
<th>JUMPER</th>
<th>DEFAULT CONNECTION</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1</td>
<td>position 2-3</td>
<td>Selects the configuration mode</td>
</tr>
<tr>
<td>J2</td>
<td>connected</td>
<td>Connect BT1 battery to back up circuitry.</td>
</tr>
<tr>
<td>J4</td>
<td>position 1-2</td>
<td>Does not connect power supply +5 Vdc to CN1 connector.</td>
</tr>
<tr>
<td>J5÷J10</td>
<td>position 1-2</td>
<td>Configure serial line asynchronous on CN1 for the RS232 electrical standard</td>
</tr>
</tbody>
</table>

**FIGURE B2: DEFAULT JUMPERS CONFIGURATION TABLE**

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 SDI 02, the following features:

<table>
<thead>
<tr>
<th>OPTION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>.RTC</td>
<td>Selects clock and SRAM, back up from battery</td>
</tr>
<tr>
<td>.SW</td>
<td>Power supply switching section with wide range input</td>
</tr>
</tbody>
</table>

**FIGURE B3: 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, to found 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:

- **AMP4.Cable** complete connector with 4 coloured wires, 1 metre length;

**FIGURE B4: AMP4.CABLE CONNECTION ACCESSORY**
- CKS.AMP4 kit composed by female AMP Mod II 4 pins, plus 4 contacts to crimp;

![CKS.AMP4 Connection Accessory](image)

**Figure B5: CKS.AMP4 Connection Accessory**

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

-- AMP8.Cable complete connector with 8 coloured wires, 1 metre length;

![AMP8.Cable Connection Accessory](image)

**Figure B6: AMP8.Cable Connection Accessory**
- **CKS.AMP8** kit composed by female AMP Mod II 8 pins, plus 8 contacts to crimp;

![CKS.AMP8 Connection Accessory](image)

**FIGURE B7: CKS.AMP8 CONNECTION ACCESSORY**

These components can be acquired directly from AMP dealers by using P/N 280365 and P/N 182206-2.

- **EL 12** power supply for direct connection to mains voltage at 230 Vac, 50 Hz, that generates an output voltage of 12 Vdc, compatible for **SDI 02**. The photo of this accessories is already available in previous pages of manual, on figure 22.
APPENDIX C: ALPHABETICAL INDEX

Simboli

+5 Vdc  14, 22, 27
+Vdc pow  12
.SW  22, B-2

A

Accessories  B-2
AMP4.Cable  B-2
AMP8.Cable  14, B-3
ASCII  27, 39
Assistance  1
Asynchronous  8, 20, 26

B

Back Up  20
Back up  9
Battery  9, 11, 20
Baud  26
Buffer  32

C

CCITT  14
Change folder  48
Characters  8
CKS.AMP4  B-3
CKS.AMP8  14, B-4
Clock  14
Close file  59
CN1  10, 14, 22
CN2  10, 18
CN3  10, 12
Commands  39
Communication  14, 26, 31, 35
Configuration  20
Connection  12, 16
Connector  10, 11, 12, 14, 18, 22
Consumption  9
Container  1, 10
Copyright  2
Create folder  48
CTS  14, 32
Current  9

D

Default  19, 22
Default configuration  B-1
Demo_SD02  60
Demo_SDI02  38, 61
Directive  1
Disk commands  47
Documentation  1, 7

E
EEPROM  26
EL 12  22, B-4
Electrostatic noises  1
Empty buffer on file  58
Erase file  52
ESD  1

F
FAT  18
FAT16  18
Format disk  47

G
Get available file manage  53
Get configurations  46
Get disk features  47
Get file list  50
Get file position  58
Get files information  51
Get files list  50
Get real time clock  42
Get status  40
Get versions  40
GND  12, 14, 22, 26, 27

H
Handshake  14, 26, 31, 32
Hardware  11

I
I2C BUS  8, 9, 20, 26, 27, 30, 34
I2CBUS  16
Introduction  1

J
Jumper  11, 19, 22
L
LED 8, 21

M
Memory card 18
MMC card 7, 18
Mounting 10

N
Network 17

O
Omega rail 10
Open file 54
Option 13, B-2
Output 9

P
Parameter 26
Parity 26
Phase 22
Power supply 9, 12, 22, B-4
Protection 1
Protocol 14, 26

R
RAM 7
Read back up sram 44
Read file 55
Remove folder 49
Rename file 52
RESET ERROR 41
Reset errors 40
Resistor 9
RS 232 9, 14, 20, 26, 38
RTC 8, 20
RTS 14, 26, 32
Rules 1

S
Safety 1
SCL 14, 27
SD card 7, 18
SDA 14, 27
Serial 14
Serial line 20, 26, 27
Set configuration 45
Set real time clock 42
Size 10
Sla.add 26, 34
Software 24
SPI interface 18
SRAM 20
Stop 26
Switching 14
Synchronous 8, 16, 26, 27, 30, 34

T
Temperature 10
Tout 26, 35
Trademarks 2
TransZorb™ 23
TTL 14, 20, 26

U
Umidity 10

V
Vac 12, 22
Voltage 9, 22

W
Warranty 1, 2
Weight 10
Write back up sram 44
Write file 56

X
XOFF 32
XON 32