CAN 14

Control Area Network 1 channel, 4 type

TECHNICAL MANUAL

grifo®
ITALIAN TECHNOLOGY

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

CAN 14 Edition 5.00 *Rel. 08 June 2001*

abaco®, **GPC®**, **grifo®**, *are registered trademarks of* **grifo®**

CAN 14

Control Area Network 1 channel, 4 type

TECHNICAL MANUAL

Peripheral communication module featurin CAN protocol; interface for **Abaco® I/O BUS** on **26 pins** standard low profile connector; Size: **100x50x40** mm, (110x60x70 mm with container) in **4 type** format; plastic support for **DIN 46277-1** and **DIN 46277-3** Ω rails; **9 pins female D** connector for CAN communication line; complete support of **CAN 2.0B** protocol, managed by PHILIPS **SJA 1000** UART at 24 MHz (compatible with PHILIPS PCx82C200); PHILIPS **82C250** line driver, galvanically insulated; **DC/DC** converter for optocoupling the CAN communication line; supported **baud rates** are up to **400 KBit/sec** on lines long **hundreds** of metres and up to **1 MBit/sec** on lines long **tens** of metres; only **2** consecutives Bytes used for card **I/O** addressing; card **I/O** allocation address defined through proper **dip switch**; **interrupt** requests joined to many events, by software; interrupt connection to **/INT** or **/NMI**, selectable by hardware; single power supply voltage: **+5 Vdc; 93 mA**.

grifo®
ITALIAN TECHNOLOGY

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

CAN 14 Edition 5.00 *Rel. 08 June 2001*

abaco®, **GPC®**, **grifo®**, *are registered trademarks of* **grifo®**

DOCUMENTATION COPYRIGHT BY grifo®, ALL RIGHTS RESERVED

No part of this document may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, either electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, without the prior written consent of **grifo**®.

IMPORTANT

Although all the information contained herein have been carefully verified, **grifo**® assumes no responsability for errors that might appear in this document, or for damage to things or persons resulting from technical errors, omission and improper use of this manual and of the related software and hardware.
grifo® reserves the right to change the contents and form of this document, as well as the features and specification of its products at any time, without prior notice, to obtain always the best product.
For specific informations on the components mounted on the card, please refer to the Data Book of the builder or second sources.

SYMBOLS DESCRIPTION

In the manual could appear the following symbols:

Attention: Generic danger

Attention: High voltage

Trade Marks

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

GENERAL INDEX

INTRODUCTION 1

CARD VERSION 1

GENERAL INFORMATION 2
- INTERFACING AND ADDRESSING 4
- CAN CONTROLLER 4
- CAN LINE INTERFACE 4

TECHNICAL FEATURES 5
- GENERAL FEATURES 5
- PHYSICAL FEATURES 5
- ELECTRIC FEATURES 6

INSTALLATION 8
- CONNECTIONS 8
- CN2 - CAN SERIAL LINE CONNECTOR 8
- CN1 - ABACO® I/O BUS CONNECTOR 10
- BOARD CONNNECTIONS 12
- MECHANICAL MOUNTING 12
- JUMPERS 12
  - 2 PINS JUMPERS 13
  - 3 PINS JUMPERS 13
- CAN LINE TERMINATION 13
- INTERRUPT 14
- RESET 14

HARDWARE DESCRIPTION 16
- BOARD MAPPING 16
- INTERNAL REGISTERS ADDRESSING 17

PERIPHERAL DEVICES SOFTWARE DESCRIPTION 18
- CAN CONTROLLER 18
  - EXAMPLE 18
  - NOTES 18

EXTERNAL CARDS 20

BIBLIOGRAPHY 25

APPENDIX A: ON BOARD COMPONENTS DESCRIPTION A-1

APPENDIX B: ALPHABETICAL INDEX B-1

FIGURES INDEX

FIGURE 1: BLOCK DIAGRAM 3
FIGURE 2: COMPONENTS MAP 7
FIGURE 3: CN2 - CAN SERIAL LINE CONNECTOR 8
FIGURE 4: CAN BUS CONNECTION EXAMPLE 9
FIGURE 5: CN1 - CONNECTOR FOR ABACO® I/O BUS 10
FIGURE 6: CONNECTORS, JUMPERS, DIP SWITCH, ETC. LOCATION 11
FIGURE 7: JUMPERS SUMMARIZING TABLE 12
FIGURE 8: 2 PINS JUMPERS TABLE 13
FIGURE 9: 3 PINS JUMPERS TABLE 13
FIGURE 10: CARD PHOTO 15
FIGURE 11: INTERNAL REGISTERS ADDRESSING TABLE 17
FIGURE 12: INITIALIZATION FLOW CHART 19
FIGURE 13: POSSIBLE CONNECTIONS DIAGRAM 23

INTRODUCTION

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

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

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

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

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

To be on good terms with the products, is necessary guarantee legibility and conservation of the manual, also for future references. In case of deterioration or more easily for technical updates, consult the AUTHORIZED TECHNICAL ASSISTANCE directly.

To prevent problems during card utilization, it is a good practice to read carefully all the informations of this manual. After this reading, the user can use the general index and the alphabetical index, respectly at the begining and at the end of the manual, to find information in a faster and more easy way.

CARD VERSION

The present handbook is reported to the **CAN 14** card release **110498** and later. The validity of the bring informations is subordinate to the number of the card release. The user must always verify the correct correspondence among the two denotations. On the card the release number is present in more points both board printed diagram (serigraph) and printed circuit (in the top right corner between jumper J3 and resistor R1 both on the component side and on the solder side).

GENERAL INFORMATION

The **CAN 14** is a powerfull peripheral card capable to manage the standard Control Area Network serial communication protocol. The most important features of this protocol are: high data transfer rate, long distance connection support, full communication error checking with automatic retrasmission, self recognition of the slave in a multi system network, multimaster and multislave network connections, management of reception and transmission masks, wide range of powerfull commands, etc. In details the **CAN 14** includes all the **BasicCAN** and/or **PeliCAN 2.0B** protocol features to which the user can refer to get further information.
The electric connection of the card is performed through two confortable connectors: one for the control cards with **Abaco®** I/O BUS and the other for the CAN line; the mechanical mounting is simplified by a proper plastic support for **DIN 46277-1** and **DIN 46277-3** omega rails. The CAN line that is available on the connector is galvanically insulated from the other electronic component; this feauture ensures immunity to external noise and prevents possible damaging of the control electronics.
The **CAN 14** typical application fields are the automotive and general industrial environments, where it is necessary to transfer data amongst two or more intelligent decives without having to worry about all the characteristic situations of serial communication, especially when it is required an high level communication, a multimaster network, a noise resistant link or a connection with another CAN device.
A wide range of demo programs on how to use this card, allow an immediate use of the same. These programs are available for the whole CPUs of **Abaco®** family. They are duly documented and supplied under "source" form in the many different languages in which **Abaco®** cards can be programmed.

- Interface for **Abaco® I/O BUS** on **26 pins** standard low profile connector.
- Dimension: **100x50x40** mm, (110x60x70 mm with container) in **4 type** format.
- Plastic support for **DIN 46277-1** and **DIN 46277-3** Ω rails.
- **9 pins female D** connector for the CAN communication line.
- Complete support of **CAN 2.0B** protocol, managed by PHILIPS **SJA 1000** UART at 24 MHz (compatible with PHILIPS PCx82C200).
- PHILIPS **82C250** line driver, galvanically insulated.
- **DC/DC** converter for optocoupling the CAN communication line.
- Supported **baud rate**: up to **400 KBit/sec** on lines long **hundreds** of metres , **1 MBit/sec** on lines long **tens** of metres.
- Only **2** consecutives Bytes used for card **I/O** addressing.
- Card **I/O** allocation address defined through proper **dip switch**.
- **Interrupt** requests joined to many events, by software.
- Interrupt connection to **/INT** or **/NMI**, selectable by hardware.
- Single power supply voltage: **+5 Vdc; 93 mA**.

Here follows a description of the board's functional blocks, indicating the operation each block performs. To easily locate these blocks and to verify their connections, please refer to figure1.

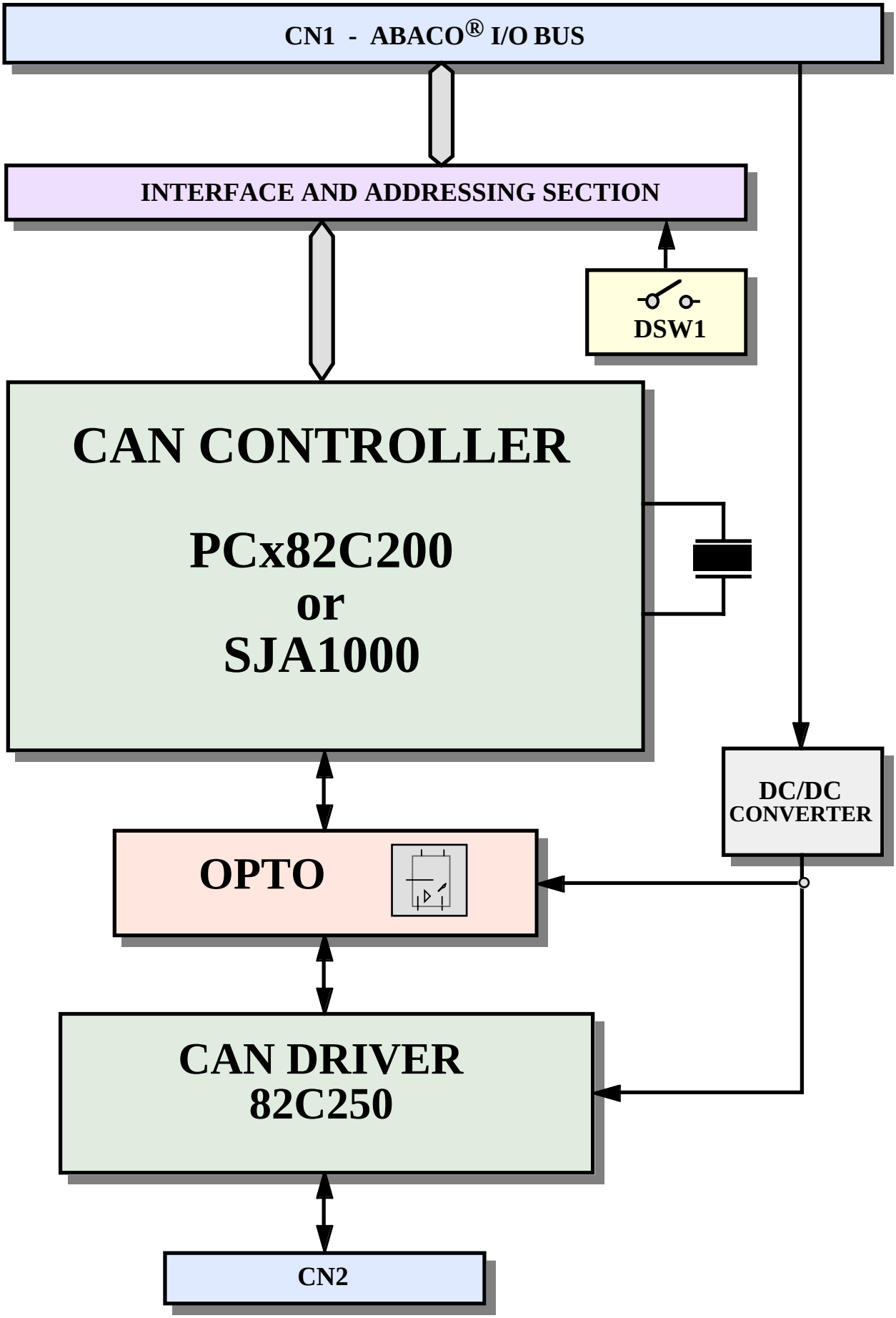


FIGURE 1: BLOCK DIAGRAM

INTERFACING AND ADDRESSING

This section manages the data exchange between CAN controller and command board; in particular, all written or read data transit across this section that, in addition, provides the board I/O management in its addressing space and an eventual interrupt generation by setting the dip switch **DSW1**.
Physical connection to command boards is performed through **ABACO®** I/O BUS, featuring 8 data bits, but it can be extended to **ABACO®** BUS using specific conversion modules like **ABB 05** or **ABB 03**.
Interfacing and addressing section is based on a Programmable Array Logic device and some secondary components carefully selected to warrant a correct working under any condition and minimize the room taken.

CAN CONTROLLER

This section is basedn on one of PHILIPS **PCx82C200** or **SJA 1000** controllers, it is charged to perform the software management of all CAN protocol modalities and aspects. Overall features of this section are:

	PCx82C200	**SJA 1000**
- support for BasicCAN protocol:	YES	YES
- support for PeliCAN 2.0B protocol:	NO	YES
- identifier lengths supported:	11 bits	11 and 29 bits
- transmission buffer size:	10 bytes	13 bytes
- reception buffer size:	10 bytes	64 bytes
- baud rate programmabie up to:	1M Bit/sec	1M Bit/sec
- elimination of reception comparatore:	NO	YES
- configurable messages acceptance filters:	YES	YES
- programmable output driver:	YES	YES
- maximum working frequence:	16 MHz	24 MHz

SJA1000 controller is totally compatible with **PCx82C200**, for which it is an improved update. So the user can switch to use this new controller keeping the firmware or the software already developed and tested.
For further information the user can refer to the manufacturer documentation or to appendix A of this manual.

CAN LINE INTERFACE

Electrically, the **CAN 14** board is provided with the electric driver PHILIPS **82C250** in galvanic isolation. This component brings all the features required by CAN protocol to interface to the field without any need of software intervent. In addition, the CAN serial line is galvanically isolated from the rest of the board, to warrant immunity from eventual electric nois; this feature is essential in case of connection to remote systems with different potential or when the connection cables must be installed in environments electrically very noisy.
A specific DC/DC converter is charged to generate all the galvanically isolated tensions required by the line drivers, the interfacing to the CAN controller communication lines is performed through specific high frequence optocouplers. Connection to the field is made using a 9 pins D type connector to easy the signals untangling and warrants a good signal transmission.

TECHNICAL FEATURES

GENERAL FEATURES

On board resources:	1 CAN serial line 1 eight pins dip switch to set I/O address
BUS type:	Industrial **ABACO®** 8 bits data BUS
Addressing space:	256 bytes
Bytes taken:	2
On board peripherals:	CAN controller PCx82C200 CAN controller SJA1000
Clock frequence:	16 MHz (PCx82C200) 24 MHz (SJA1000)
Maximum baud rate:	1 Mbit/sec
Protocols supported:	BasicCAN (PCx82C200 and SJA1000) PeliCAN 2.0B (only SJA1000)
Interrupt management:	Selectable between /INT and /NMI

PHYSICAL FEATURES

Size:	100 x 50 x 20 mm 110 x 60 x 70 mm	(without container) (with Ω rail container)
Weight:	50 g 110 g	(without container) (with Ω rail container)
Connectors:	CN1: 26 pins low profile M vertical CN2: 9 pins D type F 90°	
Temperature range:	from 0 to 50° C	
Relative humidity:	20% up to 90% (without condensing)	

ELECTRIC FEATURES

Power supply:	+5 Vdc ± 5%
Current consumption:	93 mA
CAN line impedance:	60 Ω
CAN termination network:	120 Ω resistor, unpluggable
TTL voltage levels:	0 Vdc (low level); +5 Vdc (high level)

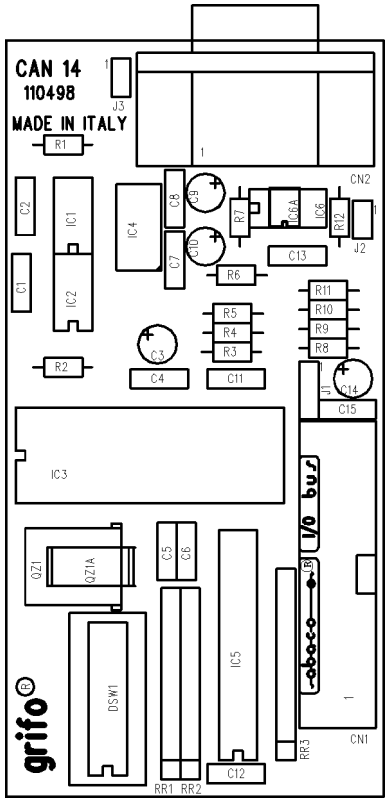


FIGURE 2: COMPONENTS MAP

INSTALLATION

In this chapter there are the information for a right installation and correct use of **CAN 14** card. The user can find the location and functions of each connector, dip switch etc. and some explanatory diagrams.

CONNECTIONS

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

CN2 - CAN SERIAL LINE CONNECTOR

CN2 is a 9 pins D type 90° female connector. It allows to connect the CAN serial communication line according to the specifications defined by the CAN standard. Signals placement on the connector has been designed to reduce problems of noise and interference and to warrant a good transmission quality.

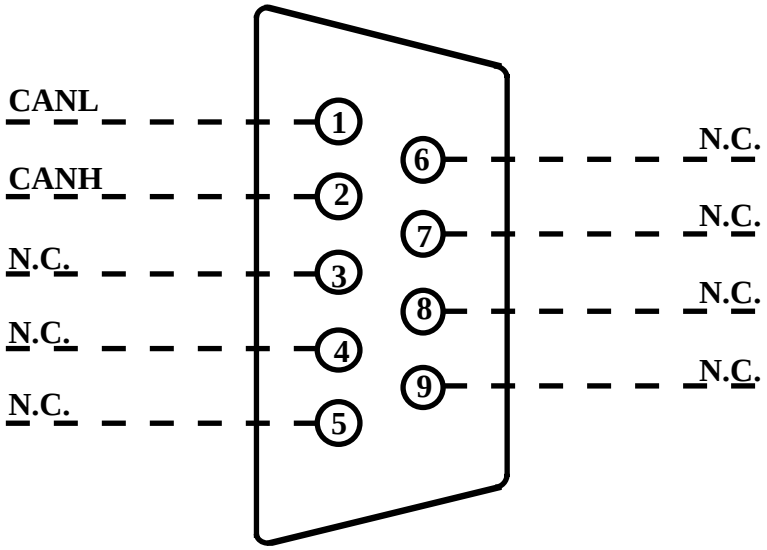


FIGURE 3: CN2 - CAN SERIAL LINE CONNECTOR

Signals description:

CANH	=	I/O	-	Differential high line for CAN BUS.
CANL	=	I/O	-	Differential low line for CAN BUS.
N.C.	=		-	Not connected.

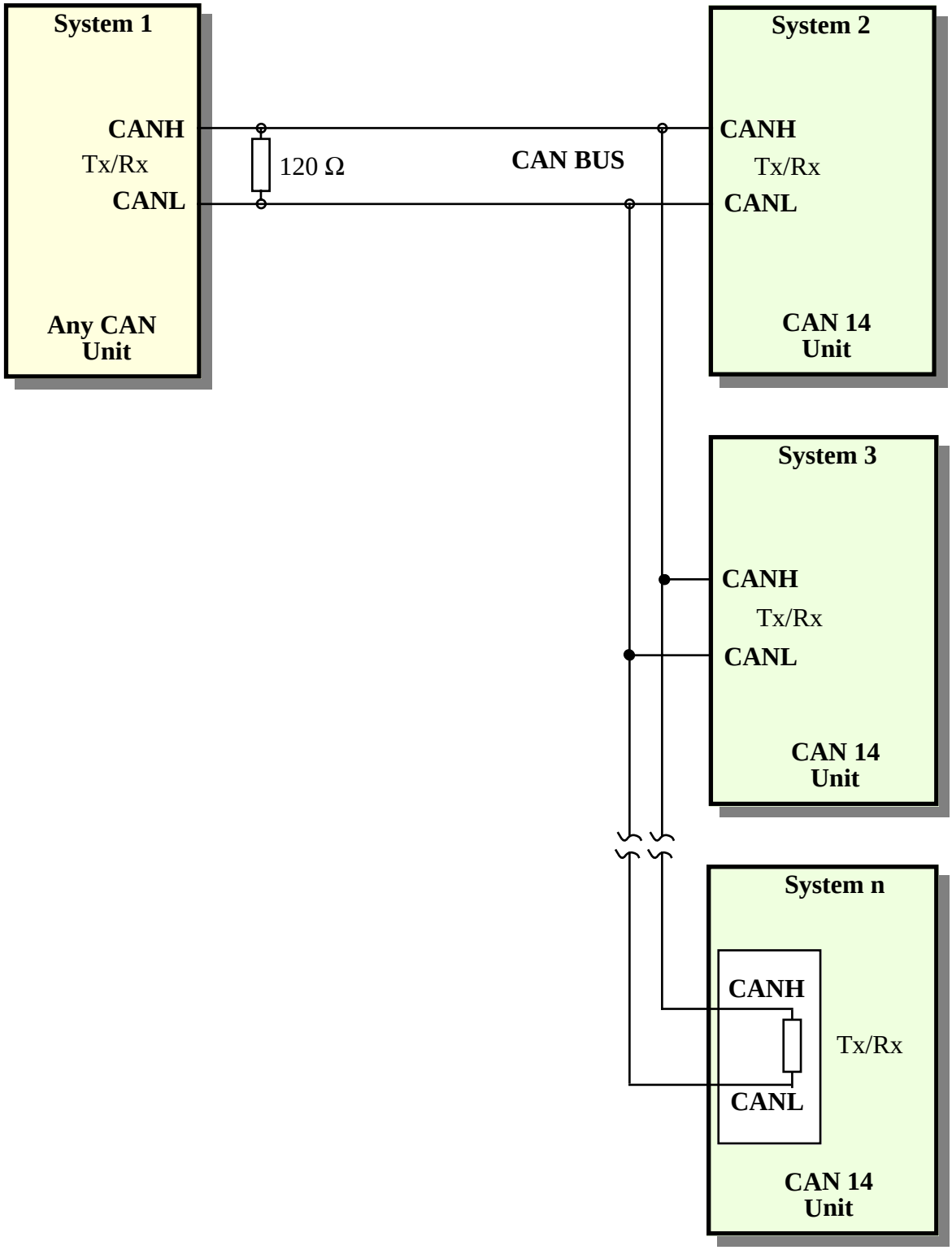


FIGURE 4: CAN BUS CONNECTION EXAMPLE

CN1 - ABACO® I/O BUS CONNECTOR

CN1 is a 26 pins low profile vertical male connector with pitch 2.54 mm. CN1 allows the connection between CAN 14 and the several **GPC®** cards. Such connection is performed through the TTL singals of **ABACO®** I/O BUS which are reported here.

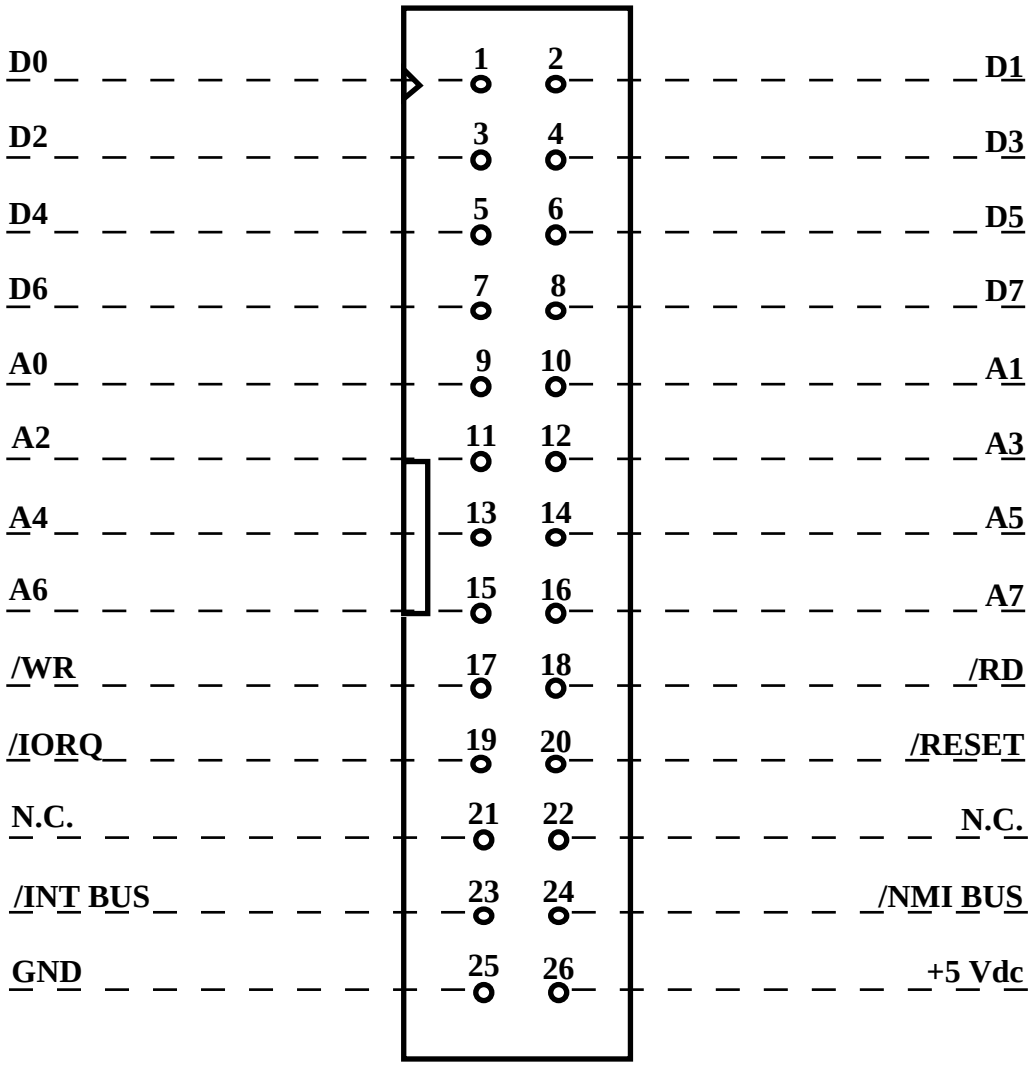


FIGURE 5: CN1 - CONNECTOR FOR ABACO® I/O BUS

Signals description:

A0-A7	=	I	- Address BUS.
D0-D7	=	I/O	- Data BUS.
/INT BUS	=	O	- Interrupt request (open collector).
/NMI BUS	=	O	- Non Maskable Interrupt.
/IORQ	=	I	- Input Output Request.
/RD	=	I	- Read cycle status.
/WR	=	I	- Write cycle status.
/RESET	=	I	- Reset.
+5 Vdc	=	I	- Power supply +5 Vdc.
GND	=		- Ground.
N.C.	=		- Not connected.

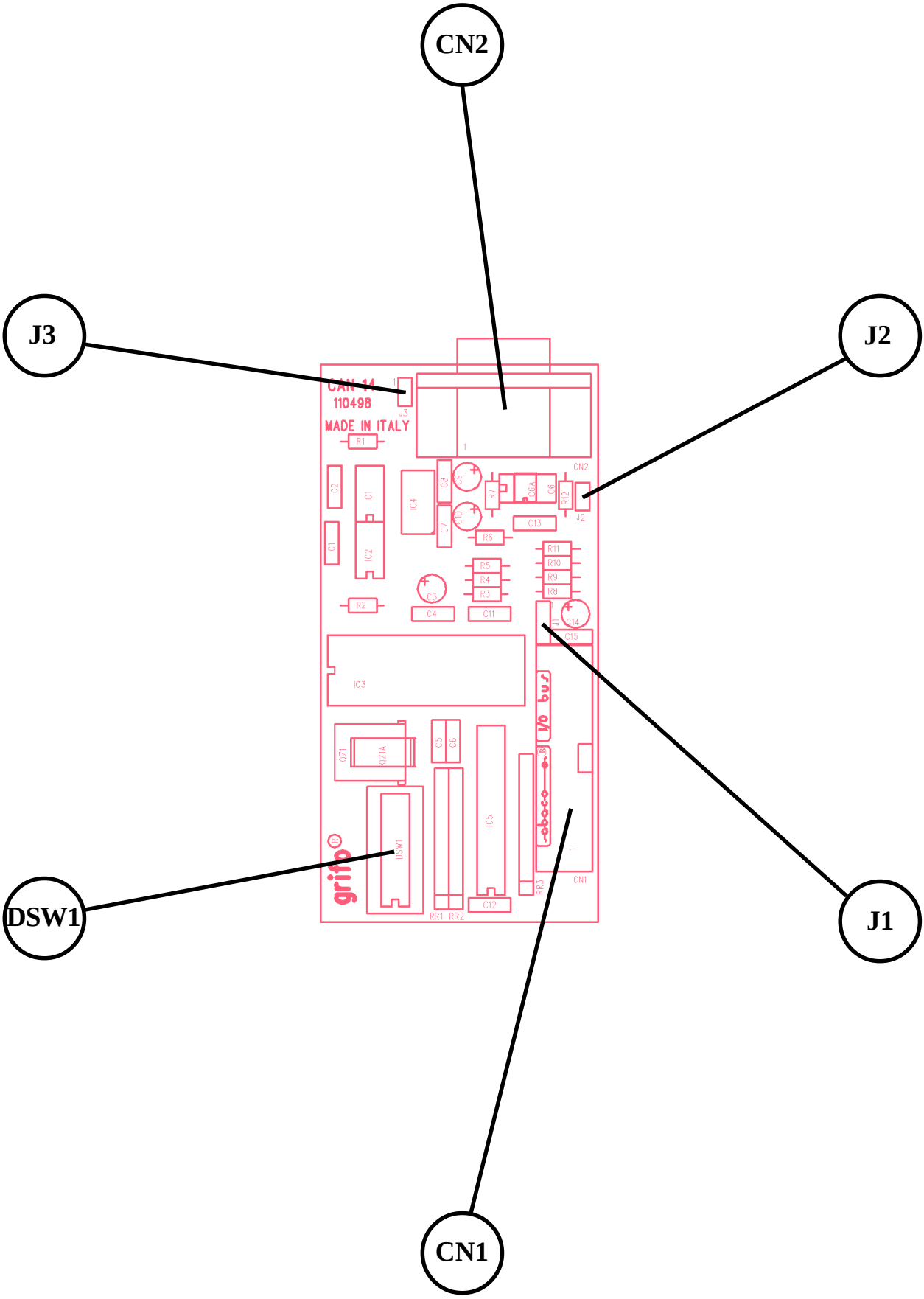


FIGURE 6: CONNECTORS, JUMPERS, DIP SWITCH, ETC. LOCATION

BOARD CONNNECTIONS

To prevent possible connecting problems between **CAN 14** board and the external systems, the user has to read carefully the information of the previous paragraphs and he must follow these instructions:

- The TTL signals can be connected directly only to a device featuring the same type of interface. About the correspondance between logic signals and TTL output status, remember that a logic **0** generates a TTL 0 Vdc, while a logic **1** generates a TTL +5 Vdc. The high frequence signals of **ABACO®** I/O BUS can be connected by a flat cable not longer than about 10 centimeters.

- For information about the serial CAN BUS communication protocol signals, please refer to the standard specification of the protocol. To warrant the galvanic isolation between the CAN communtication line and the control board, it is suggestable not to connect directly or indirectly the **CAN 14** supply ground (pin 25 of CN1) to other systems installed on the CAN BUS.

MECHANICAL MOUNTING

By default, CAN 14 board is provided without any container, but optionally are available items that easy remarkably its mechanical mounting. These items are plastic container designed to be installed on Ω rails type **DIN 46277-1** and **DIN 46277-3**. Should the **CAN 14** be matched to other boards provided with **ABACO®** I/O BUS, it is possible to order an unique container for all the boards, to simplify the mounting and reduce the costs. For further information please call **grifo®** directly.

JUMPERS

On **CAN 14** board there are 3 jumpers for card configuration. Below there is the jumpers list, location and function.

JUMPER	N. PINS	PURPOSE
J1	3	Selects the connection for interrupt signal .
J2	2	Selects the connection for termination resistor.
J3	2	Selects the connection for CN2 connector's armour.

FIGURE 7: JUMPERS SUMMARIZING TABLE

The following tables describe all the right connections of **CAN 14** jumpers with their relative functions. To recognize these valid connections, please refer to the board printed diagram (serigraph) or to figure 2 of this manual, where the pins numeration is listed; for recognizing jumpers location, please refer to figure 6.
The "*" used in the following tables, denotes the default connection, or on the other hand the connection set up at the end of testing phase, that is the configuration the user receives.

2 PINS JUMPERS

JUMPERS	CONNECTION	PURPOSE	DEF.
J2	not connected	Does not connect the 120 Ωtermination resistor to CAN line.	*
	connected	Connects the 120 Ωtermination resistor to CAN line.	
J3	not connected	Does not connect the armour of connector CN2 to the galvanically isolated ground of line interface.	*
	connected	Connects the armour of connector CN2 to the galvanically isolated ground of line interface.	

FIGURE 8: 2 PINS JUMPERS TABLE

3 PINS JUMPERS

JUMPERS	CONNECTION	PURPOSE	DEF.
J1	position 1-2	Connects interrupt of CAN controller to signal /INT of **ABACO®** I/O BUS.	
	position 2-3	Connects interrupt of CAN controller to signal /NMI of **ABACO®** I/O BUS.	
	not connected	Interrupt of CAN controller not connected.	*

FIGURE 9: 3 PINS JUMPERS TABLE

CAN LINE TERMINATION

Jumper J2 can connect or not connect the specific CAN line termination resistor as described in the table on figure 8. CAN BUS must always be configured physically as a differential signal line with 60 Ω of impedance so the termination resistors must be connected to recreate such configuration. In detail the resistor must be always connected in case of point-to-point communication, while for multipoint communication the termination must be connected only on the farest terminal, that is the board at the end of the communication line (please refer to example on figure 4).
The correct termination of CAN line improves remarkably the communcation reliability, in fact **CAN 14** on -board line interface can suppress transients and keep untouched by radio frequence and electromagnetic noise only if the connection to the field is performed correctly.

INTERRUPT

CAN 14 is provided with a comfortable and efficent interrupt generation circuitery, that, if enabled, can generate an interrupt to the **GPC®** control card when predetermined conditions occour. Such circuitery allows to optimize the time needed to manage the board, in fact the **GPC®** intelligent control card is not obliged to poll **CAN 14** registers, but can simply wait for an interrupt and manage the new data interchange.
The board interrput signal can be connected in three different modalities through jumper J1, as described in the table of figure 8; the signal type is open collector so it can be connected to one of the two **ABACO®** I/O BUS interrupt request lines even if other peripherals are already connected to it.
For further information about interrupt management modalities, please refer to appendix A.

RESET

When a power on occours or the /RESET signal from **ABACO®** I/O BUS is actived, **CAN 14** stops any message reception/transmission and enters in reset mode. On the 1-to-0 transition of the reset request bit, the CAN controller returns to the operating mode.
For further information about reset mode please refer to appendix A of this manual.

FIGURE 10: CARD PHOTO

HARDWARE DESCRIPTION

This chapter provides all the hardware informations needed to use **CAN 14** board. Here the user will find information about I/O card mapping and on board peripheral devices addressing.

BOARD MAPPING

CAN 14 board is mapped into a **2** bytes I/O addressing space, that can be allocated starting from different base addresses according to how the board is configured. This feature allows to use several **CAN 14** cards on the same **ABACO®** I/O BUS or **ABACO®** BUS, or to install them on a BUS where other peripheral modules are installed obtaining a structure that can be expanded without any difficulty or modifications to the application software.
These bytes allow the complete control of board settings and status and the complete flow of input and output data.
The base address can be defined through the specific BUS interface circuitry on the board itself; this circuitry uses the eight pins dip switch called **DSW1**, from which it reads the address set by the user. Here follows the corrispondance between dips configuration and address signals, to easily locate such component please refer to figure 6.

DSW1.1	->	Not used
DSW1.2	->	Bit A1
DSW1.3	->	Bit A2
DSW1.4	->	Bit A3
DSW1.5	->	Bit A4
DSW1.6	->	Bit A5
DSW1.7	->	Bit A6
DSW1.8	->	Bit A7

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

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

As an example, dip configuration to set address 048H is repored here:

DSW1.1	->	Indifferent
DSW1.2	->	ON
DSW1.3	->	ON
DSW1.4	->	OFF
DSW1.5	->	ON
DSW1.6	->	ON
DSW1.7	->	OFF
DSW1.8	->	ON

INTERNAL REGISTERS ADDRESSING

Indicating the board base address with <**baseaddr**>, that is the address set using dip switch DSW1, as indicated in the previous paragraph **CAN 14** internal registers are addressable as explained in the following table.

DEVICE	REG.	ADDRESS	R/W	MEANING
PCx82C200	ADDR	<baseaddr>+00H	W	Internal register addressing register.
SJA1000	DATA	<baseaddr>+01H	R/W	Data read/write register.

FIGURE 11: INTERNAL REGISTERS ADDRESSING TABLE

Please remark that the previous table reports the description of only the registers available addressing **CAN 14** board. For a more detailed description of all the CAN controller internal registers and their access modalities refer to next chapter or appendix A.

PERIPHERAL DEVICES SOFTWARE DESCRIPTION

In the previous paragraph allocation addresses of all the peripherals have been reported, here follows a detailed description of function and meaning of internal registers (please always refer to the peripheral mapping tables to understand completely the following informations). Should the present documentaion be inadequate please refer to the component's manufacturer documentation or to appendix A of this manual.
In the following paragraphs the indications **D0÷D7** or **D0÷D15** are used to refer the bits of the byte or word involved in the I/O operations.

CAN CONTROLLER

To access on the **PCx82C200** or **SJA 1000** CAN controllers several registers, it is essential to perform the following operations:

1) Write to register ADDR the register address required.
2) Read from or write to register DATA to get or set the previously selected register.

Appendix A reports the complete list of CAN controller registers and the functional description of all their bits.

EXAMPLE

Referring to appendix A, if data **170** must be written to **Acceptance code register** (register no. 4), the following operations are required:

1) Write to register ADDR the value 4.
2) Write to register DATA the value 170.

NOTES

1) Communication Baud Rate, as described in the information reported in appendix A, can be obtained from the following formula:

BAUD RATE = Freq / 2 * (BRP + 1) * (3 + TSEG1 + TSEG2)

where:
Freq = Frequence of **CAN 14** quartz expressed in Hz.
BRP = Value of bit **BRP.x** in **Bus Timing Register 0** (BTR0, address **6**).
TSEG1 = Value of bit **TSEG1.x** in **Bus Timing Register 1** (BTR1, address **7**).
TSEG2 = Value of bit **TSEG2.x** in **Bus Timing Register 1** (BTR1, address **7**).

2) For a correct interfacement between **PCx82C200** or **SJA1000** CAN controller and 82C250 line driver, it is essential to program **Output control register** (OCR, address **8**) with value **FA Hex**; this configures the device in "Normal output mode", with outputs TX0 and TX1 in "Push-Pull".

The following page reports the flow chart corresponding to **PCx82C200** or **SJA1000** CAN controllers initialization.

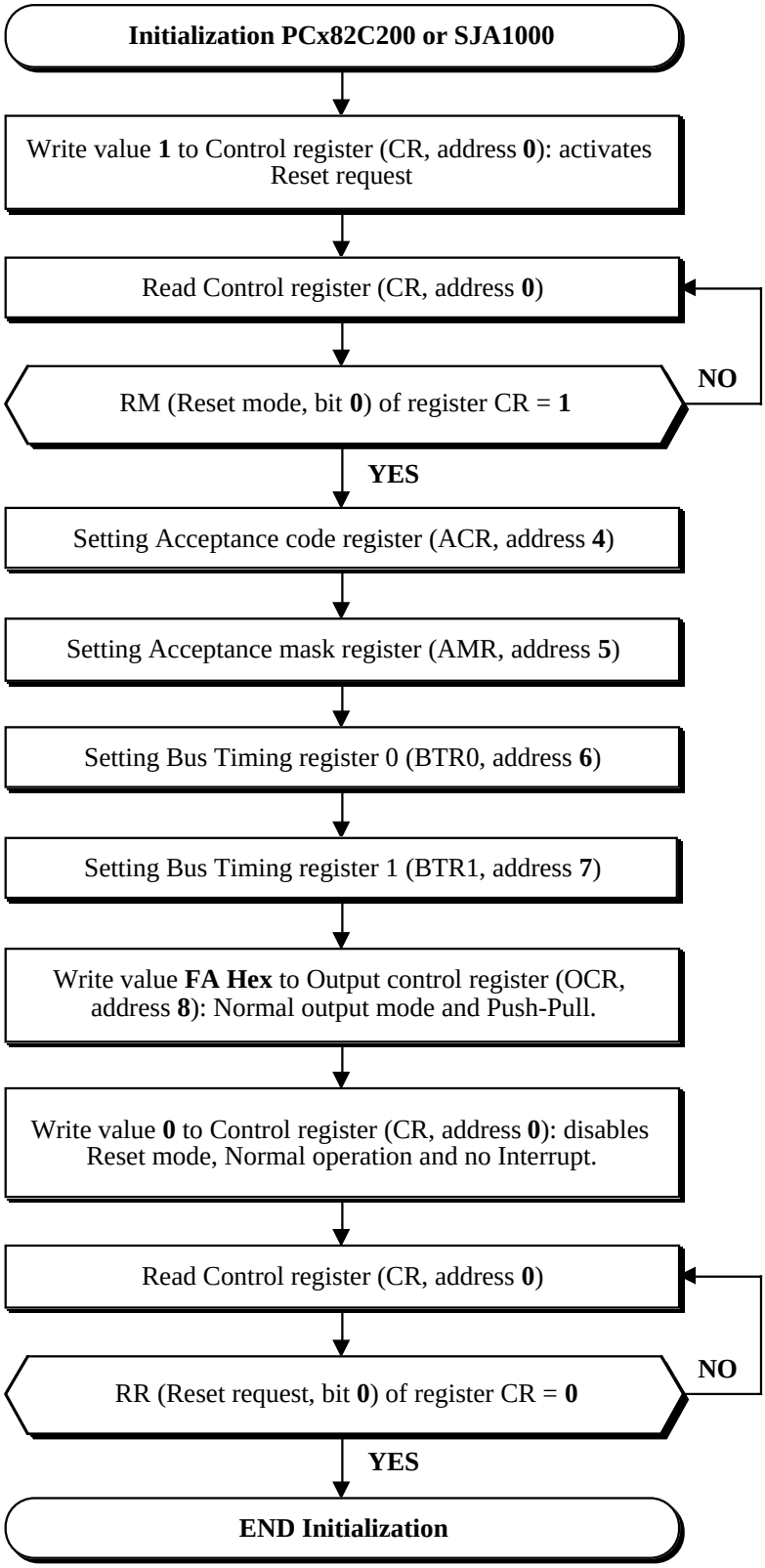


FIGURE 12: INITIALIZATION FLOW CHART

As you can see this initialization uses no interrupt; for their eventual use it is essential to set opportunely the bits of Control Register (CR, indirizzo 0).
Please refer to appendix A and to the example programs provided with **CAN 14** board for further information.

EXTERNAL CARDS

CAN 14 can be connected to a wide range of block modules and operator interface system produced by **grifo®**, or to many system of other companies.The on board resources can be expanded with a simple connection to the numerous peripheral **grifo®** boards,both intelligent and not, thanks to its standard **ABACO®** I/O BUS connector. Even cards with **ABACO®** BUS can be connected, by using the proper mother boards.
Hereunder some of these cards are briefly described; ask the detailed information directly to **grifo®**, if required.

QTP G28

Quick Terminal Panel - LCD Graphic, 28 keys

LCD display 240x128 pixels, CFC backlit; Optocoupled RS 232 line and additional RS 232/422/485/ Current Loop line; CAN line controller; E^2 for set up; RTC and RAM lithium backed; primary graphic object; possibility of re-naming keys, LEDs and panel name; 28 keys and 16 LEDs with blinking attribute and buzzer manageable by software; Buzzer; built in power supply; reader of magnetic badge and relay option.

MB3 01-MB4 01-MB8 01

Mother Board 3, 4, 8 slots

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

SPB 04-SPB 08

Switch Power BUS 4-8 slots

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

ABB 03

ABACO® Block BUS 3 slots

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

ABB 05

ABACO® Block BUS 5 slots

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

FBC 20-120

Flat Block Contact 20 vie

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

FBC D9 M/F

Flat Block Contact vaschetta D 9 vie Male/Female

Interfacce between 1 D type 9 pins connector (male or female) and the field wires (quick release connectors). Container for DIN 46277-1 and 3.

SPC 03.5S

Switch Power Card +5 Vdc

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

SPC 512

Switch Power Card +5 Vdc +12 Vdc

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

GPC® 51

General Purpose Controller fam. 51

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

GPC® 188F

General Purpose Controller 80C188

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

GPC® 15A

General Purpose Controller 84C15

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

GPC® 150

General Purpose Controller 84C15

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

GPC® 15R

General Purpose Controller 84C15

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

GPC® 323

General Purpose Controller 51 family

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

GPC® 553

General Purpose Controller 80C552

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

GPC® 153

General Purpose Controller Z80

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

GPC® 183

General Purpose Controller Z180

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

GPC® 324/D

“4” Type General Purpose Controller 80C32/320

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

GPC® 554

General Purpose Controller 80C552

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

GPC® 154

“4” Type General Purpose Controller Z80

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

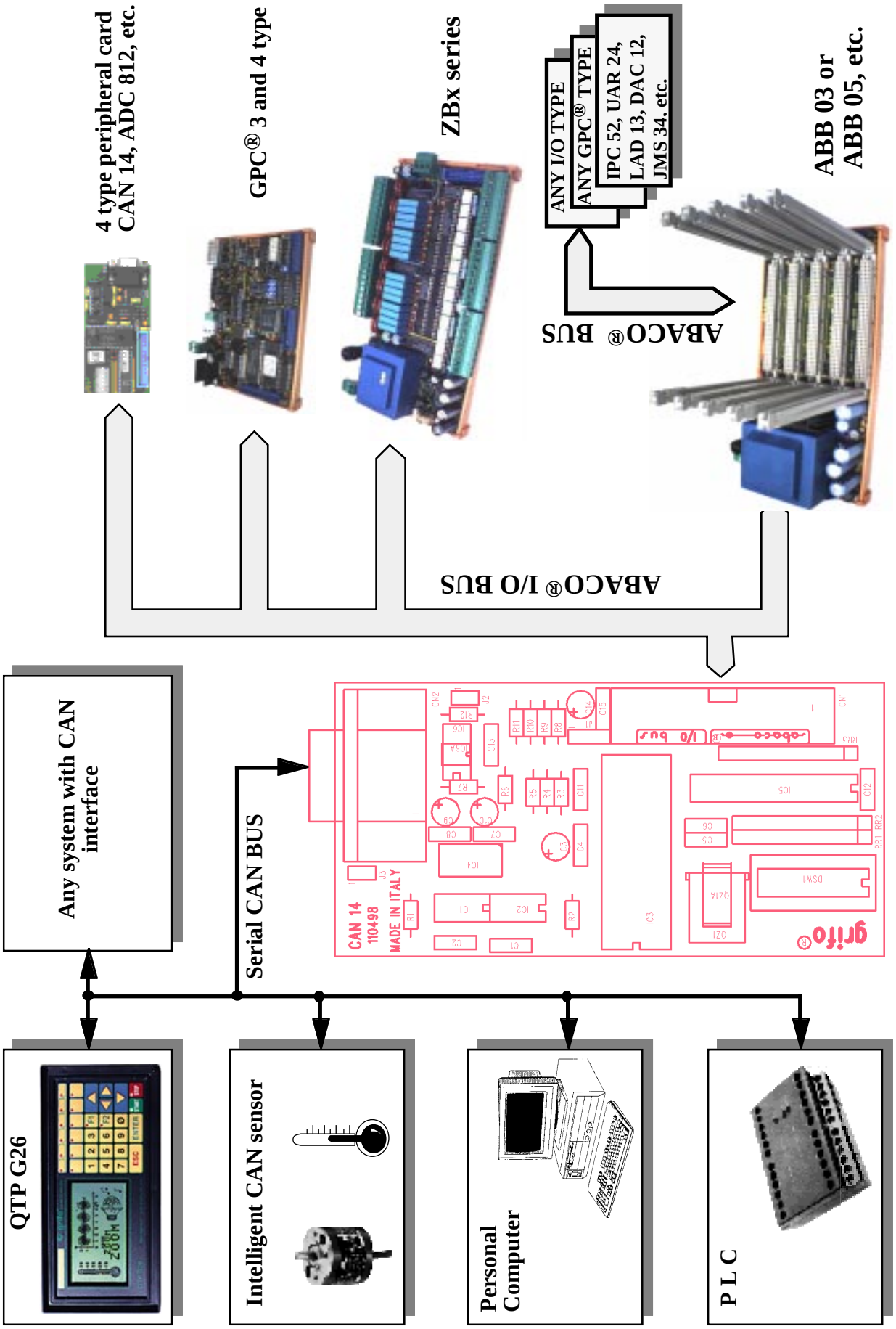


FIGURE 13: POSSIBLE CONNECTIONS DIAGRAM

GPC® 884

General Purpose Controller Am188ES

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

GPC® 114

General Purpose Controller 68HC11

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

GPC® 184

General Purpose Controller Z80195

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

GPC® AM4

General Purpose Controller ATmega103

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

BIBLIOGRAPHY

Here follows a list of manuls thst can be a source of further informations about the devices installed on **CAN 14**.

Manual PHILIPS: *Application notes and development tools for 80C51 microcontrollers*

Manual SGS-THOMSON: *Programmable logic manual - GAL products*

Manual TOSHIBA: *Photo couplers Data Book*

Manual NEWPORT: *DC-DC Converters*

Please connect to the manifactures Web sites to get the latest version of all manuals and data sheets.

1 FEATURES

- Pin compatibility to the PCA82C200 stand-alone CAN controller
- Electrical compatibility to the PCA82C200 stand-alone CAN controller
- Software-compatibility mode to the PCA82C200 (BasicCAN mode is default)
- Extended receive buffer (64-byte FIFO)
- CAN 2.0B protocol compatibility (extended frame passive in PCA82C200 compatibility mode)
- Supports 11-bit identifier as well as 29-bit identifier
- Bit rates up to 1 Mbits/s
- PeliCAN mode extensions:
  - Error counters with read/write access
  - Programmable error warning limit
  - Last error code register
  - Error interrupt for each CAN-bus error
  - Arbitration lost interrupt with detailed bit position
  - Single-shot transmission (no re-transmission)
  - Listen only mode (no acknowledge, no active error flags)
  - Hot plugging support (software driven bit rate detection)
  - Acceptance filter extension (4-byte code, 4-byte mask)
  - Reception of 'own' messages (self reception request)
- 24 MHz clock frequency
- Interfaces to a variety of microprocessors
- Programmable CAN output driver configuration
- Extended ambient temperature range (–40 to +125 °C).

2 GENERAL DESCRIPTION

The SJA1000 is a stand-alone controller for the Controller Area Network (CAN) used within automotive and general industrial environments. It is designed to be hardware and software compatible to the PCA82C200 CAN controller (BasicCAN) from Philips Semiconductors. Additionally, a new mode of operation is implemented (PeliCAN) which supports the CAN 2.0B protocol specification with several new features.

3 ORDERING INFORMATION

TYPE NUMBER	PACKAGE		
	NAME	DESCRIPTION	VERSION
SJA1000	DIP28	plastic dual in-line package; 28 leads (600 mil)	SOT117-1
SJA1000T	SO28	plastic small outline package; 28 leads; body width 7.5 mm	SOT136-1

4 BLOCK DIAGRAM

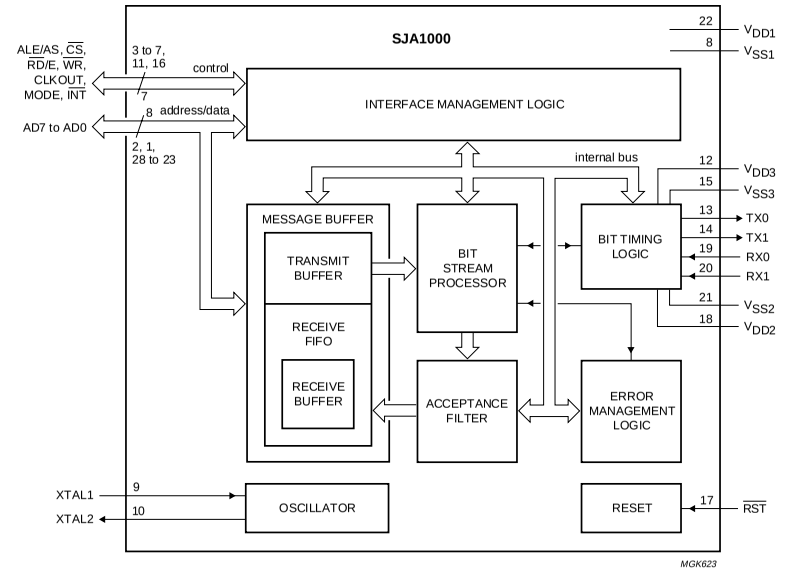


Fig.1 Block diagram.

6 FUNCTIONAL DESCRIPTION

6.1 Description of the CAN controller blocks

6.1.1 Interface Management Logic (IML)

The interface management logic interprets commands from the CPU, controls addressing of the CAN registers and provides interrupts and status information to the host microcontroller.

6.1.2 Transmit Buffer (TXB)

The transmit buffer is an interface between the CPU and the Bit Stream Processor (BSP) that is able to store a complete message for transmission over the CAN network. The buffer is 13 bytes long, written to by the CPU and read out by the BSP.

6.1.3 Receive Buffer (RXB, RXFIFO)

The receive buffer is an interface between the acceptance filter and the CPU that stores the received and accepted messages from the CAN-bus line. The Receive Buffer (RXB) represents a CPU-accessible 13-byte window of the Receive FIFO (RXFIFO), which has a total length of 64 bytes.
With the help of this FIFO the CPU is able to process one message while other messages are being received.

6.1.4 Acceptance Filter (ACF)

The acceptance filter compares the received identifier with the acceptance filter register contents and decides whether this message should be accepted or not. In the event of a positive acceptance test, the complete message is stored in the RXFIFO.

6.1.5 Bit Stream Processor (BSP)

The bit stream processor is a sequencer which controls the data stream between the transmit buffer, RXFIFO and the CAN-bus. It also performs the error detection, arbitration, stuffing and error handling on the CAN-bus.

6.1.6 Bit Timing Logic (BTL)

The bit timing logic monitors the serial CAN-bus line and handles the bus line-related bit timing. It is synchronized to the bit stream on the CAN-bus on a 'recessive-to-dominant' bus line transition at the beginning of a message (hard synchronization) and re-synchronized on further transitions during the reception of a message (soft synchronization). The BTL also provides programmable time segments to compensate for the propagation delay times and phase shifts (e.g. due to oscillator drifts) and to define the sample point and the number of samples to be taken within a bit time.

6.1.7 Error Management Logic (EML)

The EML is responsible for the error confinement of the transfer-layer modules. It receives error announcements from the BSP and then informs the BSP and IML about error statistics.

6.2 Detailed description of the CAN controller

The SJA1000 is designed to be software and pin-compatible to its predecessor, the PCA82C200 stand-alone CAN controller. Additionally, a lot of new functions are implemented. To achieve the software compatibility, two different modes of operation are implemented:

- BasicCAN mode; PCA82C200 compatible
- PeliCAN mode; extended features.

The mode of operation is selected with the CAN-mode bit located within the clock divider register. Default mode upon reset is the BasicCAN mode.

6.2.1 PCA82C200 compatibility

In BasicCAN mode the SJA1000 emulates all known registers from the PCA82C200 stand-alone CAN controller. The characteristics, as described in Sections 6.2.1.1 to 6.2.1.4 are different from the PCA82C200 design with respect to software compatibility.

6.2.1.1 Synchronization mode

The SYNC bit in the control register is removed (CR.6 in the PCA82C200). Synchronization is only possible by a recessive-to-dominant transition on the CAN-bus. Writing to this bit has no effect. To achieve compatibility to existing application software, a read access to this bit will reflect the previously written value (flip-flop without effect).

6.2.1.2 Clock divider register

The clock divider register is used to select the CAN mode of operation (BasicCAN/PeliCAN). Therefore one of the reserved bits within the PCA82C200 is used. Writing a value between 0 and 7, as allowed for the PCA82C200, will enter the BasicCAN mode. The default state is divide by 12 for Motorola mode and divide by 2 for Intel mode. An additional function is implemented within another of the reserved bits. Setting of bit CBP (see Table 49) enables the internal RX input comparator to be bypassed thereby reducing the internal delays if an external transceiver circuit is used.

6.2.1.3 Receive buffer

The dual receive buffer concept of the PCA82C200 is replaced by the receive FIFO from the PeliCAN controller. This has no effect to the application software except for the data overrun probability. Now more than two messages may be received (up to 64 bytes) until a data overrun occurs.

6.2.1.4 CAN 2.0B

The SJA1000 is designed to support the full CAN 2.0B protocol specification, which means that the extended oscillator tolerance is implemented as well as the processing of extended frame messages. In BasicCAN mode it is possible to transmit and receive standard frame messages only (11-bit identifier). If extended frame messages (29-bit identifier) are detected on the CAN-bus, they are tolerated and an acknowledge is given if the message was correct, but there is no receive interrupt generated.

6.2.2 Differences between BasicCAN and PeliCAN mode

In the PeliCAN mode the SJA1000 appears with a re-organized register mapping with a lot of new features. All known bits from the PCA82C200 design are available as well as several new ones. In the PeliCAN mode the complete CAN 2.0B functionality is supported (29-bit identifier).

Main new features of the SJA1000 are:

- Reception and transmission of standard and extended frame format messages
- Receive FIFO (64-byte)
- Single/dual acceptance filter with mask and code register for standard and extended frame
- Error counters with read/write access
- Programmable error warning limit
- Last error code register
- Error interrupt for each CAN-bus error
- Arbitration lost interrupt with detailed bit position
- Single-shot transmission (no re-transmission on error or arbitration lost)
- Listen only mode (monitoring of the CAN-bus, no acknowledge, no error flags)
- Hot plugging supported (disturbance-free software driven bit rate detection)
- Disable CLKOUT by hardware.

6.3 BasicCAN mode

6.3.1 BasicCAN address layout

The SJA1000 appears to a microcontroller as a memory-mapped I/O device. An independent operation of both devices is guaranteed by a RAM-like implementation of the on-chip registers.

The address area of the SJA1000 consists of the control segment and the message buffers. The control segment is programmed during an initialization download in order to configure communication parameters (e.g. bit timing). Communication over the CAN-bus is also controlled via this segment by the microcontroller. During initialization the CLKOUT signal may be programmed to a value determined by the microcontroller.

A message, which should be transmitted, has to be written to the transmit buffer. After a successful reception the microcontroller may read the received message from the receive buffer and then release it for further use.

The exchange of status, control and command signals between the microcontroller and the SJA1000 is performed in the control segment. The layout of this segment is shown in Table 3. After an initial download, the contents of the registers acceptance code, acceptance mask, bus timing registers 0 and 1 and output control should not be changed. Therefore these registers may only be accessed when the reset request bit in the control register is set HIGH.

For register access, two different modes have to be distinguished:

- Reset mode
- Operating mode.

The reset mode (see Table 3, control register, bit Reset Request) is entered automatically after a hardware reset or when the controller enters the bus-off state (see Table 5, status register, bit Bus Status). The operating mode is activated by resetting of the reset request bit in the control register.

CAN 14 Rel. 5.00

abaco bus grifo® ITALIAN TECHNOLOGY

Table 1 BasicCAN address allocation; note 1

CAN ADDRESS	SEGMENT	OPERATING MODE		RESET MODE	
		READ	WRITE	READ	WRITE
0	control	control	control	control	control
1		(FFH)[2]	command	(FFH)[2]	command
2		status	–	status	–
3		interrupt	–	interrupt	–
4		(FFH)[2]	–	acceptance code	acceptance code
5		(FFH)[2]	–	acceptance mask	acceptance mask
6		(FFH)[2]	–	bus timing 0	bus timing 0
7		(FFH)[2]	–	bus timing 1	bus timing 1
8		(FFH)[2]	–	output control	output control
9		test	test	test	test
10	transmit buffer	identifier (10 to 3)	identifier (10 to 3)	(FFH)[2]	–
11		identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC	(FFH)[2]	–
12		data byte 1	data byte 1	(FFH)[2]	–
13		data byte 2	data byte 2	(FFH)[2]	–
14		data byte 3	data byte 3	(FFH)[2]	–
15		data byte 4	data byte 4	(FFH)[2]	–
16		data byte 5	data byte 5	(FFH)[2]	–
17		data byte 6	data byte 6	(FFH)[2]	–
18		data byte 7	data byte 7	(FFH)[2]	–
19		data byte 8	data byte 8	(FFH)[2]	–
20	receive buffer	identifier (10 to 3)	identifier (10 to 3)	identifier (10 to 3)	identifier (10 to 3)
21		identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC	identifier (2 to 0), RTR and DLC
22		data byte 1	data byte 1	data byte 1	data byte 1
23		data byte 2	data byte 2	data byte 2	data byte 2
24		data byte 3	data byte 3	data byte 3	data byte 3
25		data byte 4	data byte 4	data byte 4	data byte 4
26		data byte 5	data byte 5	data byte 5	data byte 5
27		data byte 6	data byte 6	data byte 6	data byte 6
28		data byte 7	data byte 7	data byte 7	data byte 7
29		data byte 8	data byte 8	data byte 8	data byte 8
30		(FFH)[2]	–	(FFH)[2]	–
31		clock divider	clock divider; note 3	clock divider	clock divider

Notes

1. It should be noted that the registers are repeated within higher CAN address areas (the most significant bits of the 8-bit CPU address are not decoded: CAN address 32 continues with CAN address 0 and so on).
2. During read-out of this register a zero is always given.
3. Some bits are writeable in reset mode only (CAN mode and CBP).

6.3.2 RESET VALUES

Detection of a 'reset request' results in aborting the current transmission/reception of a message and entering the reset mode. On the '1-to-0' transition of the reset request bit, the CAN controller returns to the operating mode.

Table 2 Reset mode configuration; notes 1 and 2

REGISTER	BIT	SYMBOL	NAME	VALUE	
				RESET BY HARDWARE	SETTING BIT CR.0 BY SOFTWARE OR DUE TO BUS-OFF
Control	CR.7	–	reserved	0	0
	CR.6	–	reserved	X	X
	CR.5	–	reserved	1	1
	CR.4	OIE	Overrun Interrupt Enable	X	X
	CR.3	EIE	Error Interrupt Enable	X	X
	CR.2	TIE	Transmit Interrupt Enable	X	X
	CR.1	RIE	Receive Interrupt Enable	X	X
	CR.0	RR	Reset Request	1 (reset mode)	1 (reset mode)
Command	CMR.7	–	reserved	note 3	note 3
	CMR.6	–	reserved		
	CMR.5	–	reserved		
	CMR.4	GTS	Go To Sleep		
	CMR.3	CDO	Clear Data Overrun		
	CMR.2	RRB	Release Receive Buffer		
	CMR.1	AT	Abort Transmission		
	CMR.0	TR	Transmission Request		
Status	SR.7	BS	Bus Status	0 (bus-on)	X
	SR.6	ES	Error Status	0 (ok)	X
	SR.5	TS	Transmit Status	0 (idle)	0 (idle)
	SR.4	RS	Receive Status	0 (idle)	0 (idle)
	SR.3	TCS	Transmission Complete Status	1 (complete)	X
	SR.2	TBS	Transmit Buffer Status	1 (released)	1 (released)
	SR.1	DOS	Data Overrun Status	0 (absent)	0 (absent)
	SR.0	RBS	Receive Buffer Status	0 (empty)	0 (empty)
Interrupt	IR.7	–	reserved	1	1
	IR.6	–	reserved	1	1
	IR.5	–	reserved	1	1
	IR.4	WUI	Wake-Up Interrupt	0 (reset)	0 (reset)
	IR.3	DOI	Data Overrun Interrupt	0 (reset)	0 (reset)
	IR.2	EI	Error Interrupt	0 (reset)	X; note 4
	IR.1	TI	Transmit Interrupt	0 (reset)	0 (reset)
	IR.0	RI	Receive Interrupt	0 (reset)	0 (reset)

CAN 14 Rel. 5.00 Page A-3
ITALIAN TECHNOLOGY grifo® abaco bus

REGISTER	BIT	SYMBOL	NAME	VALUE	
				RESET BY HARDWARE	SETTING BIT CR.0 BY SOFTWARE OR DUE TO BUS-OFF
Acceptance code	AC.7 to 0	AC	Acceptance Code	X	X
Acceptance mask	AM.7 to 0	AM	Acceptance Mask	X	X
Bus timing 0	BTR0.7	SJW.1	Synchronization Jump Width 1	X	X
	BTR0.6	SJW.0	Synchronization Jump Width 0	X	X
	BTR0.5	BRP.5	Baud Rate Prescaler 5	X	X
	BTR0.4	BRP.4	Baud Rate Prescaler 4	X	X
	BTR0.3	BRP.3	Baud Rate Prescaler 3	X	X
	BTR0.2	BRP.2	Baud Rate Prescaler 2	X	X
	BTR0.1	BRP.1	Baud Rate Prescaler 1	X	X
	BTR0.0	BRP.0	Baud Rate Prescaler 0	X	X
Bus timing 1	BTR1.7	SAM	Sampling	X	X
	BTR1.6	TSEG2.2	Time Segment 2.2	X	X
	BTR1.5	TSEG2.1	Time Segment 2.1	X	X
	BTR1.4	TSEG2.0	Time Segment 2.0	X	X
	BTR1.3	TSEG1.3	Time Segment 1.3	X	X
	BTR1.2	TSEG1.2	Time Segment 1.2	X	X
	BTR1.1	TSEG1.1	Time Segment 1.1	X	X
	BTR1.0	TSEG1.0	Time Segment 1.0	X	X
Output control	OC.7	OCTP1	Output Control Transistor P1	X	X
	OC.6	OCTN1	Output Control Transistor N1	X	X
	OC.5	OCPOL1	Output Control Polarity 1	X	X
	OC.4	OCTP0	Output Control Transistor P0	X	X
	OC.3	OCTN0	Output Control Transistor N0	X	X
	OC.2	OCPOL0	Output Control Polarity 0	X	X
	OC.1	OCMODE1	Output Control Mode 1	X	X
	OC.0	OCMODE0	Output Control Mode 0	X	X
Transmit buffer	–	TXB	Transmit Buffer	X	X
Receive buffer	–	RXB	Receive Buffer	X; note 5	X; note 5
Clock divider	–	CDR	Clock Divider Register	00000000 (Intel); 00000101 (Motorola)	X

Notes

1. X means that the value of these registers or bits is not influenced.
2. Remarks in brackets explain functional meaning.
3. Reading the command register will always reflect a binary ‘11111111’.
4. On bus-off the error interrupt is set, if enabled.
5. Internal read/write pointers of the RXFIFO are reset to their initial values. A subsequent read access to the RXB would show undefined data values (parts of old messages). If a message is transmitted, this message is written in parallel to the receive buffer but no receive interrupt is generated and the receive buffer area is not locked. So, even if the receive buffer is empty, the last transmitted message may be read from the receive buffer until it is overridden by the next received or transmitted message.
   Upon a hardware reset, the RXFIFO pointers are reset to the physical RAM address ‘0’. Setting CR.0 by software or due to the bus-off event will reset the RXFIFO pointers to the currently valid FIFO start address which is different from the RAM address ‘0’ after the first release receive buffer command.

6.3.3 Control Register (CR)

The contents of the control register are used to change the behaviour of the CAN controller. Bits may be set or reset by the attached microcontroller which uses the control register as a read/write memory.

Table 3 Bit interpretation of the control register (CR); CAN address 0

BIT	SYMBOL	NAME	VALUE	FUNCTION
CR.7	–	–	–	reserved; note 1
CR.6	–	–	–	reserved; note 2
CR.5	–	–	–	reserved; note 3
CR.4	OIE	Overrun Interrupt Enable	1	enabled; if the data overrun bit is set, the microcontroller receives an overrun interrupt signal (see also status register; Table 5)
			0	disabled; the microcontroller receives no overrun interrupt signal from the SJA1000
CR.3	EIE	Error Interrupt Enable	1	enabled; if the error or bus status change, the microcontroller receives an error interrupt signal (see also status register; Table 5)
			0	disabled; the microcontroller receives no error interrupt signal from the SJA1000
CR.2	TIE	Transmit Interrupt Enable	1	enabled; when a message has been successfully transmitted or the transmit buffer is accessible again, (e.g. after an abort transmission command) the SJA1000 transmits a transmit interrupt signal to the microcontroller
			0	disabled; the microcontroller receives no transmit interrupt signal from the SJA1000
CR.1	RIE	Receive Interrupt Enable	1	enabled; when a message has been received without errors, the SJA1000 transmits a receive interrupt signal to the microcontroller
			0	disabled; the microcontroller receives no transmit interrupt signal from the SJA1000

Page A-4
CAN 14 Rel. 5.00
abaco bus grifo ITALIAN TECHNOLOGY

BIT	SYMBOL	NAME	VALUE	FUNCTION
CR.0	RR	Reset Request; note 4	1	present; detection of a reset request results in aborting the current transmission/reception of a message and entering the reset mode
			0	absent; on the '1-to-0' transition of the reset request bit, the SJA1000 returns to the operating mode

Notes

1. Any write access to the control register has to set this bit to logic 0 (reset value is logic 0).
2. In the PCA82C200 this bit was used to select the synchronization mode. Because this mode is not longer implemented, setting this bit has no influence on the microcontroller. Due to software compatibility setting this bit is allowed. This bit will not change after hardware or software reset. In addition the value written by users software is reflected.
3. Reading this bit will always reflect a logic 1.
4. During a hardware reset or when the bus status bit is set to logic 1 (bus-off), the reset request bit is set to logic 1 (present). If this bit is accessed by software, a value change will become visible and takes effect first with the next positive edge of the internal clock which operates with $\frac{1}{2}$ of the external oscillator frequency. During an external reset the microcontroller cannot set the reset request bit to logic 0 (absent). Therefore, after having set the reset request bit to logic 0, the microcontroller must check this bit to ensure that the external reset pin is not being held HIGH. Changes of the reset request bit are synchronized with the internal divided clock. Reading the reset request bit reflects the synchronized status.
   After the reset request bit is set to logic 0 the SJA1000 will wait for:
   a) One occurrence of bus-free signal (11 recessive bits), if the preceding reset request has been caused by a hardware reset or a CPU-initiated reset
   b) 128 occurrences of bus-free, if the preceding reset request has been caused by a CAN controller initiated bus-off, before re-entering the bus-on mode; it should be noted that several registers are modified if the reset request bit was set (see also Table 2).

6.3.4 Command Register (CMR)

A command bit initiates an action within the transfer layer of the SJA1000. The command register appears to the microcontroller as a write only memory. If a read access is performed to this address the byte '11111111' is returned. Between two commands at least one internal clock cycle is needed to process. The internal clock is divided by two from the external oscillator frequency.

Table 4 Bit interpretation of the command register (CMR); CAN address 1

BIT	SYMBOL	NAME	VALUE	FUNCTION
CMR.7	–	–	–	reserved
CMR.6	–	–	–	reserved
CMR.5	–	–	–	reserved
CMR.4	GTS	Go To Sleep; note 1	1	sleep; the SJA1000 enters sleep mode if no CAN interrupt is pending and there is no bus activity
			0	wake up; SJA1000 operates normal
CMR.3	CDO	Clear Data Overrun; note 2	1	clear; data overrun status bit is cleared
			0	no action
CMR.2	RRB	Release Receive Buffer; note 3	1	released; the receive buffer, representing the message memory space in the RXFIFO is released
			0	no action
CMR.1	AT	Abort Transmission; note 4	1	present; if not already in progress, a pending transmission request is cancelled
			0	absent; no action
CMR.0	TR	Transmission Request; note 5	1	present; a message will be transmitted
			0	absent; no action

Notes

1. The SJA1000 will enter sleep mode if the sleep bit is set to logic 1 (sleep); there is no bus activity and no interrupt is pending. Setting of GTS with at least one of the previously mentioned exceptions valid will result in a wake-up interrupt. After sleep mode is set, the CLKOUT signal continues until at least 15 bit times have passed, to allow a host microcontroller clocked via this signal to enter its own standby mode before the CLKOUT goes LOW.
   The SJA1000 will wake up when one of the three previously mentioned conditions is negated: after 'Go To Sleep' is set LOW (wake-up), there is bus activity or $\overline{\text{INT}}$ is driven LOW (active). On wake-up, the oscillator is started and a wake-up interrupt is generated. A sleeping SJA1000 which wakes up due to bus activity will not be able to receive this message until it detects 11 consecutive recessive bits (bus-free sequence). It should be noted that setting of GTS is not possible in reset mode. After clearing of reset request, setting of GTS is possible first, when bus-free is detected again.
2. This command bit is used to clear the data overrun condition indicated by the data overrun status bit. As long as the data overrun status bit is set no further data overrun interrupt is generated. It is allowed to give the clear data overrun command at the same time as a release receive buffer command.
3. After reading the contents of the receive buffer, the microcontroller can release this memory space of the RXFIFO by setting the release receive buffer bit to logic 1. This may result in another message becoming immediately available within the receive buffer. This event will force another receive interrupt, if enabled. If there is no other message available no further receive interrupt is generated and the receive buffer status bit is cleared.
4. The abort transmission bit is used when the CPU requires the suspension of the previously requested transmission, e.g. to transmit a more urgent message before. A transmission already in progress is not stopped. In order to see if the original message had been either transmitted successfully or aborted, the transmission complete status bit should be checked. This should be done after the transmit buffer status bit has been set to logic 1 (released) or a transmit interrupt has been generated.
5. If the transmission request was set to logic 1 in a previous command, it cannot be cancelled by setting the transmission request bit to logic 0. The requested transmission may be cancelled by setting the abort transmission bit to logic 1.

CAN 14 Rel. 5.00 Page A-5
ITALIAN TECHNOLOGY grifo® abaco bus

6.3.5 STATUS REGISTER (SR)

The content of the status register reflects the status of the SJA1000. The status register appears to the microcontroller as a read only memory.

Table 5 Bit interpretation of the status register (SR); CAN address 2

BIT	SYMBOL	NAME	VALUE	FUNCTION
SR.7	BS	Bus Status; note 1	1	bus-off; the SJA1000 is not involved in bus activities
			0	bus-on; the SJA1000 is involved in bus activities
SR.6	ES	Error Status; note 2	1	error; at least one of the error counters has reached or exceeded the CPU warning limit
			0	ok; both error counters are below the warning limit
SR.5	TS	Transmit Status; note 3	1	transmit; the SJA1000 is transmitting a message
			0	idle; no transmit message is in progress
SR.4	RS	Receive Status; note 3	1	receive; the SJA1000 is receiving a message
			0	idle; no receive message is in progress
SR.3	TCS	Transmission Complete Status; note 4	1	complete; the last requested transmission has been successfully completed
			0	incomplete; the previously requested transmission is not yet completed
SR.2	TBS	Transmit Buffer Status; note 5	1	released; the CPU may write a message into the transmit buffer
			0	locked; the CPU cannot access the transmit buffer; a message is waiting for transmission or is already in process
SR.1	DOS	Data Overrun Status; note 6	1	overrun; a message was lost because there was not enough space for that message in the RXFIFO
			0	absent; no data overrun has occurred since the last clear data overrun command was given
SR.0	RBS	Receive Buffer Status; note 7	1	full; one or more messages are available in the RXFIFO
			0	empty; no message is available

Notes

1. When the transmit error counter exceeds the limit of 255 [the bus status bit is set to logic 1 (bus-off)] the CAN controller will set the reset request bit to logic 1 (present) and an error interrupt is generated, if enabled. It will stay in this mode until the CPU clears the reset request bit. Once this is completed the CAN controller will wait the minimum protocol-defined time (128 occurrences of the bus-free signal). After that the bus status bit is cleared (bus-on), the error status bit is set to logic 0 (ok), the error counters are reset and an error interrupt is generated, if enabled.
2. Errors detected during reception or transmission will affect the error counters according to the CAN 2.0B protocol specification. The error status bit is set when at least one of the error counters has reached or exceeded the CPU warning limit of 96. An error interrupt is generated, if enabled.
3. If both the receive status and the transmit status bits are logic 0 (idle) the CAN-bus is idle.
4. The transmission complete status bit is set to logic 0 (incomplete) whenever the transmission request bit is set to logic 1. The transmission complete status bit will remain at logic 0 (incomplete) until a message is transmitted successfully.
5. If the CPU tries to write to the transmit buffer when the transmit buffer status bit is at logic 0 (locked), the written byte will not be accepted and will be lost without being indicated.
6. When a message that shall be received has passed the acceptance filter successfully (i.e. earliest after arbitration field), the CAN controller needs space in the RXFIFO to store the message descriptor. Accordingly there must be enough space for each data byte which has been received. If there is not enough space to store the message, that message will be dropped and the data overrun condition will be indicated to the CPU only, if this received message has no errors until the last but one bit of end of frame (message becomes valid).
7. After reading a message stored in the RXFIFO and releasing this memory space with the command release receive buffer, this bit is cleared. If there is another message available within the FIFO this bit is set again with the next bit quantum (t_{scl}).

abaco bus grifo® ITALIAN TECHNOLOGY

CAN 14 Rel. 5.00

6.3.6 Interrupt Register (IR)

The interrupt register allows the identification of an interrupt source. When one or more bits of this register are set, the $\overline{INT}$ pin is activated (LOW). After this register is read by the microcontroller, all bits are reset what results in a floating level at $\overline{INT}$. The interrupt register appears to the microcontroller as a read only memory.

Table 6 Bit interpretation of the interrupt register (IR); CAN address 3

BIT	SYMBOL	NAME	VALUE	FUNCTION
IR.7	–	–	–	reserved
IR.6	–	–	–	reserved
IR.5	–	–	–	reserved
IR.4	WUI	Wake-Up Interrupt; note 1	1	set; this bit is set when the sleep mode is left
			0	reset; this bit is cleared by any read access of the microcontroller
IR.3	DOI	Data Overrun Interrupt; note 2	1	set; this bit is set on a '0-to-1' transition of the data overrun status bit, when the data overrun interrupt enable is set to logic 1 (enabled)
			0	reset; this bit is cleared by any read access of the microcontroller
IR.2	EI	Error Interrupt	1	set; this bit is set on a change of either the error status or bus status bits if the error interrupt enable is set to logic 1 (enabled)
			0	reset; this bit is cleared by any read access of the microcontroller
IR.1	TI	Transmit Interrupt	1	set; this bit is set whenever the transmit buffer status changes from logic 0 to logic 1 (released) and transmit interrupt enable is set to logic 1 (enabled)
			0	reset; this bit is cleared by any read access of the microcontroller
IR.0	RI	Receive Interrupt; note 3	1	set; this bit is set while the receive FIFO is not empty and the receive interrupt enable bit is set to logic 1 (enabled)
			0	reset; this bit is cleared by any read access of the microcontroller

Notes

1. A wake-up interrupt is also generated if the CPU tries to set go to sleep while the CAN controller is involved in bus activities or a CAN interrupt is pending.
2. The overrun interrupt bit (if enabled) and the data overrun status bit are set at the same time.
3. The receive interrupt bit (if enabled) and the receive buffer status bit are set at the same time.
   It should be noted that the receive interrupt bit is cleared upon a read access, even if there is another message available within the FIFO. The moment the release receive buffer command is given and there is another message valid within the receive buffer, the receive interrupt is set again (if enabled) with the next t_{sci}.

6.3.7 Transmit Buffer Layout

The global layout of the transmit buffer is shown in Table 7. The buffer serves to store a message from the microcontroller to be transmitted by the SJA1000. It is subdivided into a descriptor and data field. The transmit buffer can be written to and read out by the microcontroller in operating mode only. In reset mode a 'FFH' is reflected for all bytes.

Table 7 Layout of transmit buffer

CAN ADDRESS	FIELD	NAME	BITS							
			7	6	5	4	3	2	1	0
10	descriptor	identifier byte 1	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5	ID.4	ID.3
11		identifier byte 2	ID.2	ID.1	ID.0	RTR	DLC.3	DLC.2	DLC.1	DLC.0
12	data	TX data 1	transmit data byte 1							
13		TX data 2	transmit data byte 2							
14		TX data 3	transmit data byte 3							
15		TX data 4	transmit data byte 4							
16		TX data 5	transmit data byte 5							
17		TX data 6	transmit data byte 6							
18		TX data 7	transmit data byte 7							
19		TX data 8	transmit data byte 8							

6.3.7.1 Identifier (ID)

The identifier consists of 11 bits (ID.10 to ID.0). ID.10 is the most significant bit, which is transmitted first on the bus during the arbitration process. The identifier acts as the message's name. It is used in a receiver for acceptance filtering and also determining the bus access priority during the arbitration process. The lower the binary value of the identifier the higher the priority. This is due to a larger number of leading dominant bits during arbitration.

6.3.7.2 Remote Transmission Request (RTR)

If this bit is set, a remote frame will be transmitted via the bus. This means that no data bytes are included within this frame. Nevertheless, it is necessary to specify the correct data length code which depends on the corresponding data frame with the same identifier coding.

If the RTR bit is not set, a data frame will be sent including the number of data bytes as specified by the data length code.

6.3.7.3 Data Length Code (DLC)

The number of bytes in the data field of a message is coded by the data length code. At the start of a remote frame transmission the data length code is not considered due to the RTR bit being at logic 1 (remote). This forces the number of transmitted/received data bytes to be logic 0. Nevertheless, the data length code must be specified correctly to avoid bus errors if two CAN controllers start a remote frame transmission with the same identifier simultaneously.

The range of the data byte count is 0 to 8 bytes and is coded as follows:

DataByteCount = 8 × DLC.3 + 4 × DLC.2 + 2 × DLC.1 + DLC.0

For reasons of compatibility no data length code >8 should be used. If a value >8 is selected, 8 bytes are transmitted in the data frame with the data length code specified in DLC.

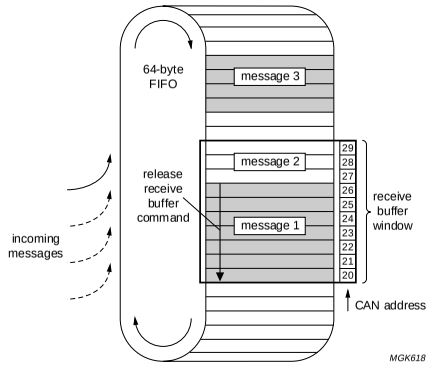
6.3.7.4 Data field

The number of transferred data bytes is determined by the data length code. The first bit transmitted is the most significant bit of data byte 1 at address 12.

6.3.8 Receive Buffer

The global layout of the receive buffer is very similar to the transmit buffer described in Section 6.3.7. The receive buffer is the accessible part of the RXFIFO and is located in the range between CAN address 20 and 29.

CAN 14 Rel. 5.00 Page A-7
ITALIAN TECHNOLOGY grifo® abaco bus



Message 1 is now available in the receive buffer.

Fig.4 Example of the message storage within the RXFIFO.

Identifier, remote transmission request bit and data length code have the same meaning and location as described in the transmit buffer but within the address range 20 to 29.

As illustrated in Fig.4 the RXFIFO has space for 64 message bytes in total. The number of messages that can be stored in the FIFO at any particular moment depends on the length of the individual messages. If there is not enough space for a new message within the RXFIFO, the CAN controller generates a data overrun condition. A message which is partly written into the RXFIFO, when the data overrun condition occurs, is deleted. This situation is indicated to the microcontroller via the status register and the data overrun interrupt, if enabled and the frame was received without any errors until the last but one bit of end of frame (RX message becomes valid).

6.3.9 ACCEPTANCE FILTER

With the help of the acceptance filter the CAN controller is able to allow passing of received messages to the RXFIFO only when the identifier bits of the received message are equal to the predefined ones within the acceptance filter registers. The acceptance filter is defined by the acceptance code register (ACR; see Section 6.3.9.1) and the acceptance mask register (AMR; see Section 6.3.9.2).

6.3.9.1 Acceptance Code Register (ACR)

Table 8 ACR bit allocation; can address 4

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
AC.7	AC.6	AC.5	AC.4	AC.3	AC.2	AC.1	AC.0

This register can be accessed (read/write), if the reset request bit is set HIGH (present). When a message is received which passes the acceptance test and there is receive buffer space left, then the respective descriptor and data field are sequentially stored in the RXFIFO. When the complete message has been correctly received the following occurs:

- The receive status bit is set HIGH (full)
- If the receive interrupt enable bit is set HIGH (enabled), the receive interrupt is set HIGH (set).

The acceptance code bits (AC.7 to AC.0) and the eight most significant bits of the message's identifier (ID.10 to ID.3) must be equal to those bit positions which are marked relevant by the acceptance mask bits (AM.7 to AM.0). If the conditions as described in the following equation are fulfilled, acceptance is given:

(ID.10 to ID.3) ≡ (AC.7 to AC.0)] ∨ (AM.7 to AM.0) ≡ 11111111

6.3.9.2 Acceptance Mask Register (AMR)

Table 9 AMR bit allocation; CAN address 5

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
AM.7	AM.6	AM.5	AM.4	AM.3	AM.2	AM.1	AM.0

This register can be accessed (read/write), if the reset request bit is set HIGH (present). The acceptance mask register qualifies which of the corresponding bits of the acceptance code are 'relevant' (AM.X = 0) or 'don't care' (AM.X = 1) for acceptance filtering.

6.3.9.3 Other registers

The other registers are described in Section 6.5.

6.4 PeliCAN mode

6.4.1 PELICAN ADDRESS LAYOUT

The CAN controller's internal registers appear to the CPU as on-chip memory mapped peripheral registers. Because the CAN controller can operate in different modes (operating/reset; see also Section 6.4.3), one has to distinguish between different internal address definitions.

Starting from CAN address 32 the complete internal RAM (80-byte) is mapped to the CPU interface.

Table 10 PeliCAN address allocation; note 1

CAN ADDRESS	OPERATING MODE				RESET MODE	
	READ		WRITE		READ	WRITE
0	mode		mode		mode	mode
1	(00H)		command		(00H)	command
2	status		–		status	–
3	interrupt		–		interrupt	–
4	interrupt enable		interrupt enable		interrupt enable	interrupt enable
5	reserved (00H)		–		reserved (00H)	–
6	bus timing 0		–		bus timing 0	bus timing 0
7	bus timing 1		–		bus timing 1	bus timing 1
8	output control		–		output control	output control
9	test		test		test	test
10	reserved (00H)		–		reserved (00H)	–
11	arbitration lost capture		–		arbitration lost capture	–
12	error code capture		–		error code capture	–
13	error warning limit		–		error warning limit	error warning limit
14	RX error counter		–		RX error counter	RX error counter
15	TX error counter		–		TX error counter	TX error counter
16	RX frame information SFF; note 2	RX frame information EFF; note 3	TX frame information SFF; note 2	TX frame information EFF; note 3	acceptance code 0	acceptance code 0
17	RX identifier 1	RX identifier 1	TX identifier 1	TX identifier 1	acceptance code 1	acceptance code 1
18	RX identifier 2	RX identifier 2	TX identifier 2	TX identifier 2	acceptance code 2	acceptance code 2
19	RX data 1	RX identifier 3	TX data 1	TX identifier 3	acceptance code 3	acceptance code 3
20	RX data 2	RX identifier 4	TX data 2	TX identifier 4	acceptance mask 0	acceptance mask 0
21	RX data 3	RX data 1	TX data 3	TX data 1	acceptance mask 1	acceptance mask 1
22	RX data 4	RX data 2	TX data 4	TX data 2	acceptance mask 2	acceptance mask 2
23	RX data 5	RX data 3	TX data 5	TX data 3	acceptance mask 3	acceptance mask 3
24	RX data 6	RX data 4	TX data 6	TX data 4	reserved (00H)	–
25	RX data 7	RX data 5	TX data 7	TX data 5	reserved (00H)	–
26	RX data 8	RX data 6	TX data 8	TX data 6	reserved (00H)	–

CAN ADDRESS	OPERATING MODE				RESET MODE	
	READ		WRITE		READ	WRITE
27	(FIFO RAM); note 4	RX data 7	–	TX data 7	reserved (00H)	–
28	(FIFO RAM); note 4	RX data 8	–	TX data 8	reserved (00H)	–
29	RX message counter		–		RX message counter	–
30	RX buffer start address		–		RX buffer start address	RX buffer start address
31	clock divider		clock divider; note 5		clock divider	clock divider
32	internal RAM address 0 (FIFO)		–		internal RAM address 0	internal RAM address 0
33	internal RAM address 1 (FIFO)		–		internal RAM address 1	internal RAM address 1
↓	↓		↓		↓	↓
95	internal RAM address 63 (FIFO)		–		internal RAM address 63	internal RAM address 63
96	internal RAM address 64 (TX buffer)		–		internal RAM address 64	internal RAM address 64
↓	↓		↓		↓	↓
108	internal RAM address 76 (TX buffer)		–		internal RAM address 76	internal RAM address 76
109	internal RAM address 77 (free)		–		internal RAM address 77	internal RAM address 77
110	internal RAM address 78 (free)		–		internal RAM address 78	internal RAM address 78
111	internal RAM address 79 (free)		–		internal RAM address 79	internal RAM address 79
112	(00H)		–		(00H)	–
↓	↓		↓		↓	↓
127	(00H)		–		(00H)	–

Notes

1. It should be noted that the registers are repeated within higher CAN address areas (the most significant bit of the 8-bit CPU address is not decoded: CAN address 128 continues with CAN address 0 and so on).
2. SFF = Standard Frame Format.
3. EFF = Extended Frame Format.
4. These address allocations reflect the FIFO RAM space behind the current message. The contents are random after power-up and contain the beginning of the next message which is received after the current one. If no further message is received, parts of old messages may occur here.
5. Some bits are writeable in reset mode only (CAN mode, CBP, RXINTEN and clock off).

CAN 14 Rel. 5.00
Page A-9
ITALIAN TECHNOLOGY grifo® abaco bus

6.4.2 RESET VALUES

Detection of a set reset mode bit results in aborting the current transmission/reception of a message and entering the reset mode. On the '1-to-0' transition of the reset mode bit, the CAN controller returns to the mode defined within the mode register.

Table 11 Reset mode configuration; notes 1 and 2

REGISTER	BIT	SYMBOL	NAME	VALUE	
				RESET BY HARDWARE	SETTING MOD.0 BY SOFTWARE OR DUE TO BUS-OFF
Mode	MOD.7 to 5	–	reserved	0 (reserved)	0 (reserved)
	MOD.4	SM	Sleep Mode	0 (wake-up)	0 (wake-up)
	MOD.3	AFM	Acceptance Filter Mode	0 (dual)	X
	MOD.2	STM	Self Test Mode	0 (normal)	X
	MOD.1	LOM	Listen Only Mode	0 (normal)	X
	MOD.0	RM	Reset Mode	1 (present)	1 (present)
Command	CMR.7 to 5	–	reserved	0 (reserved)	0 (reserved)
	CMR.4	SRR	Self Reception Request	0 (absent)	0 (absent)
	CMR.3	CDO	Clear Data Overrun	0 (no action)	0 (no action)
	CMR.2	RRB	Release Receive Buffer	0 (no action)	0 (no action)
	CMR.1	AT	Abort Transmission	0 (absent)	0 (absent)
	CMR.0	TR	Transmission Request	0 (absent)	0 (absent)
Status	SR.7	BS	Bus Status	0 (bus-on)	X
	SR.6	ES	Error Status	0 (ok)	X
	SR.5	TS	Transmit Status	1 (wait idle)	1 (wait idle)
	SR.4	RS	Receive Status	1 (wait idle)	1 (wait idle)
	SR.3	TCS	Transmission Complete Status	1 (complete)	X
	SR.2	TBS	Transmit Buffer Status	1 (released)	1 (released)
	SR.1	DOS	Data Overrun Status	0 (absent)	0 (absent)
	SR.0	RBS	Receive Buffer Status	0 (empty)	0 (empty)
Interrupt	IR.7	BEI	Bus Error Interrupt	0 (reset)	0 (reset)
	IR.6	ALI	Arbitration Lost Interrupt	0 (reset)	0 (reset)
	IR.5	EPI	Error Passive Interrupt	0 (reset)	0 (reset)
	IR.4	WUI	Wake-Up Interrupt	0 (reset)	0 (reset)
	IR.3	DOI	Data Overrun Interrupt	0 (reset)	0 (reset)
	IR.2	EI	Error Warning Interrupt	0 (reset)	X; note 3
	IR.1	TI	Transmit Interrupt	0 (reset)	0 (reset)
	IR.0	RI	Receive Interrupt	0 (reset)	0 (reset)

REGISTER	BIT	SYMBOL	NAME	VALUE	
				RESET BY HARDWARE	SETTING MOD.0 BY SOFTWARE OR DUE TO BUS-OFF
Interrupt enable	IER.7	BEIE	Bus Error Interrupt Enable	X	X
	IER.6	ALIE	Arbitration Lost Interrupt Enable	X	X
	IER.5	EPIE	Error Passive Interrupt Enable	X	X
	IER.4	WUIE	Wake-Up Interrupt Enable	X	X
	IER.3	DOIE	Data Overrun Interrupt Enable	X	X
	IER.2	EIE	Error Warning Interrupt Enable	X	X
	IER.1	TIE	Transmit Interrupt Enable	X	X
	IER.0	RIE	Receive Interrupt Enable	X	X
Bus timing 0	BTR0.7	SJW.1	Synchronization Jump Width 1	X	X
	BTR0.6	SJW.0	Synchronization Jump Width 0	X	X
	BTR0.5	BRP.5	Baud Rate Prescaler 5	X	X
	BTR0.4	BRP.4	Baud Rate Prescaler 4	X	X
	BTR0.3	BRP.3	Baud Rate Prescaler 3	X	X
	BTR0.2	BRP.2	Baud Rate Prescaler 2	X	X
	BTR0.1	BRP.1	Baud Rate Prescaler 1	X	X
	BTR0.0	BRP.0	Baud Rate Prescaler 0	X	X
Bus timing 1	BTR1.7	SAM	Sampling	X	X
	BTR1.6	TSEG2.2	Time Segment 2.2	X	X
	BTR1.5	TSEG2.1	Time Segment 2.1	X	X
	BTR1.4	TSEG2.0	Time Segment 2.0	X	X
	BTR1.3	TSEG1.3	Time Segment 1.3	X	X
	BTR1.2	TSEG1.2	Time Segment 1.2	X	X
	BTR1.1	TSEG1.1	Time Segment 1.1	X	X
	BTR1.0	TSEG1.0	Time Segment 1.0	X	X

REGISTER	BIT	SYMBOL	NAME	VALUE	
				RESET BY HARDWARE	SETTING MOD.0 BY SOFTWARE OR DUE TO BUS-OFF
Output control	OCR.7	OCTP1	Output Control Transistor P1	X	X
	OCR.6	OCTN1	Output Control Transistor N1	X	X
	OCR.5	OCPOL1	Output Control Polarity 1	X	X
	OCR.4	OCTP0	Output Control Transistor P0	X	X
	OCR.3	OCTN0	Output Control Transistor N0	X	X
	OCR.2	OCPOL0	Output Control Polarity 0	X	X
	OCR.1	OCMODE1	Output Control Mode 1	X	X
	OCR.0	OCMODE0	Output Control Mode 0	X	X
Arbitration lost capture	–	ALC	Arbitration Lost Capture	0	X
Error code capture	–	ECC	Error Code Capture	0	X
Error warning limit	–	EWLR	Error Warning Limit Register	96	X
RX error counter	–	RXERR	Receive Error Counter	0 (reset)	X; note 4
TX error counter	–	TXERR	Transmit Error Counter	0 (reset)	X; note 4
TX buffer	–	TXB	Transmit Buffer	X	X
RX buffer	–	RXB	Receive Buffer	X; note 5	X; note 5
ACR 0 to 3	–	ACR0 to ACR3	Acceptance Code Registers	X	X
AMR 0 to 3	–	AMR0 to AMR3	Acceptance Mask Registers	X	X
RX message counter	–	RMC	RX Message Counter	0	0
RX buffer start address	–	RBSA	RX Buffer Start Address	0000 0000	X
Clock divider	–	CDR	Clock Divider Register	0000 0000 Intel; 0000 0101 Motorola	X

Notes

1. X means that the value of these registers or bits is not influenced.
2. Remarks in brackets explain functional meaning.
3. On bus-off the error warning interrupt is set, if enabled.
4. If the reset mode was entered due to a bus-off condition, the receive error counter is cleared and the transmit error counter is initialized to 127 to count-down the CAN-defined bus-off recovery time consisting of 128 occurrences of 11 consecutive recessive bits.
5. Internal read/write pointers of the RXFIFO are reset to their initial values. A subsequent read access to the RXB would show undefined data values (parts of old messages).
   If a message is transmitted, this message is written in parallel to the receive buffer. A receive interrupt is generated only if this transmission was forced by the self reception request. So, even if the receive buffer is empty, the last transmitted message may be read from the receive buffer until it is overwritten by the next received or transmitted message.
   Upon a hardware reset, the RXFIFO pointers are reset to the physical RAM address '0'. Setting CR.0 by software or due to the bus-off event will reset the RXFIFO pointers to the currently valid FIFO start address (RBSA register) which is different from the RAM address '0' after the first release receive buffer command.

6.4.3 Mode Register (MOD)

The contents of the mode register are used to change the behaviour of the CAN controller. Bits may be set or reset by the CPU which uses the control register as a read/write memory. Reserved bits are read as logic 0.

Table 12 Bit interpretation of the mode register (MOD); CAN address '0'

BIT	SYMBOL	NAME	VALUE	FUNCTION
MOD.7	–	–	–	reserved
MOD.6	–	–	–	reserved
MOD.5	–	–	–	reserved
MOD.4	SM	Sleep Mode; note 1	1	sleep; the CAN controller enters sleep mode if no CAN interrupt is pending and if there is no bus activity
			0	wake-up; the CAN controller wakes up if sleeping
MOD.3	AFM	Acceptance Filter Mode; note 2	1	single; the single acceptance filter option is enabled (one filter with the length of 32 bit is active)
			0	dual; the dual acceptance filter option is enabled (two filters, each with the length of 16 bit are active)
MOD.2	STM	Self Test Mode; note 2	1	self test; in this mode a full node test is possible without any other active node on the bus using the self reception request command; the CAN controller will perform a successful transmission, even if there is no acknowledge received
			0	normal; an acknowledge is required for successful transmission

CAN 14 Rel. 5.00 Page A-11
ITALIAN TECHNOLOGY grifo® abaco bus

BIT	SYMBOL	NAME	VALUE	FUNCTION
MOD.1	LOM	Listen Only Mode; notes 2 and 3	1	listen only; in this mode the CAN would give no acknowledge to the CAN-bus, even if a message is received successfully
			0	normal; the error counters are stopped at the current value
MOD.0	RM	Reset Mode; note 4	1	reset; detection of a set reset mode bit results in aborting the current transmission/reception of a message and entering the reset mode
			0	normal; on the '1-to-0' transition of the reset mode bit, the CAN controller returns to the operating mode

Notes

1. The SJA1000 will enter sleep mode if the sleep mode bit is set to logic 1 (sleep); then there is no bus activity and no interrupt is pending. Setting of SM with at least one of the previously mentioned exceptions valid will result in a wake-up interrupt. After sleep mode is set, the CLKOUT signal continues until at least 15 bit times have passed, to allow a host microcontroller clocked via this signal to enter its own standby mode before the CLKOUT goes LOW. The SJA1000 will wake up when one of the three previously mentioned conditions is negated: after SM is set LOW (wake-up), there is bus activity or $\overline{\text{INT}}$ is driven LOW (active). On wake-up, the oscillator is started and a wake-up interrupt is generated. A sleeping SJA1000 which wakes up due to bus activity will not be able to receive this message until it detects 11 consecutive recessive bits (bus-free sequence). It should be noted that setting of SM is not possible in reset mode. After clearing of reset mode, setting of SM is possible first, when bus-free is detected again.
2. A write access to the bits MOD.1 to MOD.3 is only possible, if the reset mode is entered previously.
3. This mode of operation forces the CAN controller to be error passive. Message transmission is not possible. The listen only mode can be used e.g. for software driven bit rate detection and 'hot plugging'.
4. During a hardware reset or when the bus status bit is set to logic 1 (bus-off), the reset mode bit is also set to logic 1 (present). If this bit is accessed by software, a value change will become visible and takes effect first with the next positive edge of the internal clock which operates at half of the external oscillator frequency. During an external reset the microcontroller cannot set the reset mode bit to logic 0 (absent). Therefore, after having set the reset mode bit to logic 1, the microcontroller must check this bit to ensure that the external reset pin is not being held HIGH. Changes of the reset request bit are synchronized with the internal divided clock. Reading the reset request bit reflects the synchronized status. After the reset mode bit is set to logic 0 the CAN controller will wait for:
   a) One occurrence of bus-free signal (11 recessive bits), if the preceding reset has been caused by a hardware reset or a CPU-initiated reset.
   b) 128 occurrences of bus-free, if the preceding reset has been caused by a CAN controller initiated bus-off, before re-entering the bus-on mode.

6.4.4 COMMAND REGISTER (CMR)

A command bit initiates an action within the transfer layer of the CAN controller. This register is write only, all bits will return a logic 0 when being read. Between two commands at least one internal clock cycle is needed in order to proceed. The internal clock is half of the external oscillator frequency.

Table 13 Bit interpretation of the command register (CMR); CAN address 1

BIT	SYMBOL	NAME	VALUE	FUNCTION
CMR.7	–	reserved	–	–
CMR.6	–	reserved	–	–
CMR.5	–	reserved	–	–
CMR.4	SRR	Self Reception Request; notes 1 and 2	1	present; a message shall be transmitted and received simultaneously
			0	– (absent)
CMR.3	CDO	Clear Data Overrun; note 3	1	clear; the data overrun status bit is cleared
			0	– (no action)
CMR.2	RRB	Release Receive Buffer; note 4	1	released; the receive buffer, representing the message memory space in the RXFIFO is released
			0	– (no action)
CMR.1	AT	Abort Transmission; notes 5 and 2	1	present; if not already in progress, a pending transmission request is cancelled
			0	– (absent)
CMR.0	TR	Transmission Request; notes 6 and 2	1	present; a message shall be transmitted
			0	– (absent)

Notes

1. Upon self reception request a message is transmitted and simultaneously received if the acceptance filter is set to the corresponding identifier. A receive and a transmit interrupt will indicate correct self reception (see also self test mode in mode register).
2. Setting the command bits CMR.0 and CMR.1 simultaneously results in sending the transmit message once. No re-transmission will be performed in the event of an error or arbitration lost (single-shot transmission). Setting the command bits CMR.4 and CMR.1 simultaneously results in sending the transmit message once using the self reception feature. No re-transmission will be performed in the event of an error or arbitration lost. Setting the command bits CMR.0, CMR.1 and CMR.4 simultaneously results in sending the transmit message once as described for CMR.0 and CMR.1. The moment the transmit status bit is set within the status register, the internal transmission request bit is cleared automatically. Setting CMR.0 and CMR.4 simultaneously will ignore the set CMR.4 bit.
3. This command bit is used to clear the data overrun condition indicated by the data overrun status bit. As long as the data overrun status bit is set no further data overrun interrupt is generated.
4. After reading the contents of the receive buffer, the CPU can release this memory space in the RXFIFO by setting the release receive buffer bit to logic 1. This may result in another message becoming immediately available within the receive buffer. If there is no other message available, the receive interrupt bit is reset.

CAN 14 Rel. 5.00
abaco bus grifo® ITALIAN TECHNOLOGY

5. The abort transmission bit is used when the CPU requires the suspension of the previously requested transmission, e.g. to transmit a more urgent message before. A transmission already in progress is not stopped. In order to see if the original message has been either transmitted successfully or aborted, the transmission complete status bit should be checked. This should be done after the transmit buffer status bit has been set to logic 1 or a transmit interrupt has been generated.
   It should be noted that a transmit interrupt is generated even if the message was aborted because the transmit buffer status bit changes to 'released'.
6. If the transmission request was set to logic 1 in a previous command, it cannot be cancelled by setting the transmission request bit to logic 0. The requested transmission may be cancelled by setting the abort transmission bit to logic 1.

6.4.5 Status Register (SR)

The content of the status register reflects the status of the CAN controller. The status register appears to the CPU as a read only memory.

Table 14 Bit interpretation of the status register (SR); CAN address 2

BIT	SYMBOL	NAME	VALUE	FUNCTION
SR.7	BS	Bus Status; note 1	1	bus-off; the CAN controller is not involved in bus activities
			0	bus-on; the CAN controller is involved in bus activities
SR.6	ES	Error Status; note 2	1	error; at least one of the error counters has reached or exceeded the CPU warning limit defined by the Error Warning Limit Register (EWLR)
			0	ok; both error counters are below the warning limit
SR.5	TS	Transmit Status; note 3	1	transmit; the CAN controller is transmitting a message
			0	idle
SR.4	RS	Receive Status; note 3	1	receive; the CAN controller is receiving a message
			0	idle
SR.3	TCS	Transmission Complete Status; note 4	1	complete; last requested transmission has been successfully completed
			0	incomplete; previously requested transmission is not yet completed
SR.2	TBS	Transmit Buffer Status; note 5	1	released; the CPU may write a message into the transmit buffer
			0	locked; the CPU cannot access the transmit buffer; a message is either waiting for transmission or is in the process of being transmitted
SR.1	DOS	Data Overrun Status; note 6	1	overrun; a message was lost because there was not enough space for that message in the RXFIFO
			0	absent; no data overrun has occurred since the last clear data overrun command was given

BIT	SYMBOL	NAME	VALUE	FUNCTION
SR.0	RBS	Receive Buffer Status; note 7	1	full; one or more complete messages are available in the RXFIFO
			0	empty; no message is available

Notes

1. When the transmit error counter exceeds the limit of 255, the bus status bit is set to logic 1 (bus-off), the CAN controller will set the reset mode bit to logic 1 (present) and an error warning interrupt is generated, if enabled. The transmit error counter is set to 127 and the receive error counter is cleared. It will stay in this mode until the CPU clears the reset mode bit. Once this is completed the CAN controller will wait the minimum protocol-defined time (128 occurrences of the bus-free signal) counting down the transmit error counter. After that the bus status bit is cleared (bus-on), the error status bit is set to logic 0 (ok), the error counters are reset and an error warning interrupt is generated, if enabled. Reading the TX error counter during this time gives information about the status of the bus-off recovery.
2. Errors detected during reception or transmission will effect the error counters according to the CAN 2.0B protocol specification. The error status bit is set when at least one of the error counters has reached or exceeded the CPU warning limit (EWLR). An error warning interrupt is generated, if enabled. The default value of EWLR after hardware reset is 96.
3. If both the receive status and the transmit status bits are logic 0 (idle) the CAN-bus is idle. If both bits are set the controller is waiting to become idle again. After a hardware reset 11 consecutive recessive bits have to be detected until the idle status is reached. After bus-off this will take 128 of 11 consecutive recessive bits.
4. The transmission complete status bit is set to logic 0 (incomplete) whenever the transmission request bit or the self reception request bit is set to logic 1. The transmission complete status bit will remain at logic 0 until a message is transmitted successfully.
5. If the CPU tries to write to the transmit buffer when the transmit buffer status bit is logic 0 (locked), the written byte will not be accepted and will be lost without this being indicated.
6. When a message that is to be received has passed the acceptance filter successfully, the CAN controller needs space in the RXFIFO to store the message descriptor and for each data byte which has been received. If there is not enough space to store the message, that message is dropped and the data overrun condition is indicated to the CPU at the moment this message becomes valid. If this message is not completed successfully (e.g. due to an error), no overrun condition is indicated.
7. After reading all messages within the RXFIFO and releasing their memory space with the command release receive buffer this bit is cleared.

CAN 14 Rel. 5.00
Page A-13
ITALIAN TECHNOLOGY grifo® abaco bus

6.4.6 Interrupt Register (IR)

The interrupt register allows the identification of an interrupt source. When one or more bits of this register are set, a CAN interrupt will be indicated to the CPU. After this register is read by the CPU all bits are reset except for the receive interrupt bit.

The interrupt register appears to the CPU as a read only memory.

Table 15 Bit interpretation of the interrupt register (IR); CAN address 3

BIT	SYMBOL	NAME	VALUE	FUNCTION
IR.7	BEI	Bus Error Interrupt	1	set; this bit is set when the CAN controller detects an error on the CAN-bus and the BEIE bit is set within the interrupt enable register
			0	reset
IR.6	ALI	Arbitration Lost Interrupt	1	set; this bit is set when the CAN controller lost the arbitration and becomes a receiver and the ALIE bit is set within the interrupt enable register
			0	reset
IR.5	EPI	Error Passive Interrupt	1	set; this bit is set whenever the CAN controller has reached the error passive status (at least one error counter exceeds the protocol-defined level of 127) or if the CAN controller is in the error passive status and enters the error active status again and the EPIE bit is set within the interrupt enable register
			0	reset
IR.4	WUI	Wake-Up Interrupt; note 1	1	set; this bit is set when the CAN controller is sleeping and bus activity is detected and the WUIE bit is set within the interrupt enable register
			0	reset
IR.3	DOI	Data Overrun Interrupt	1	set; this bit is set on a '0-to-1' transition of the data overrun status bit and the DOIE bit is set within the interrupt enable register
			0	reset
IR.2	EI	Error Warning Interrupt	1	set; this bit is set on every change (set and clear) of either the error status or bus status bits and the EIE bit is set within the interrupt enable register
			0	reset
IR.1	TI	Transmit Interrupt	1	set; this bit is set whenever the transmit buffer status changes from '0-to-1' (released) and the TIE bit is set within the interrupt enable register
			0	reset
IR.0	RI	Receive Interrupt; note 2	1	set; this bit is set while the receive FIFO is not empty and the RIE bit is set within the interrupt enable register
			0	reset; no more message is available within the RXFIFO

Notes

1. A wake-up interrupt is also generated, if the CPU tries to set the sleep bit while the CAN controller is involved in bus activities or a CAN interrupt is pending.
2. The behaviour of this bit is equivalent to that of the receive buffer status bit with the exception, that RI depends on the corresponding interrupt enable bit (RIE). So the receive interrupt bit is not cleared upon a read access to the interrupt register. Giving the command 'release receive buffer' will clear RI temporarily. If there is another message available within the FIFO after the release command, RI is set again. Otherwise RI remains cleared.

6.4.7 Interrupt Enable Register (IER)

The register allows to enable different types of interrupt sources which are indicated to the CPU.

The interrupt enable register appears to the CPU as a read/write memory.

Table 16 Bit interpretation of the interrupt enable register (IER); CAN address 4

BIT	SYMBOL	NAME	VALUE	FUNCTION
IER.7	BEIE	Bus Error Interrupt Enable	1	enabled; if an bus error has been detected, the CAN controller requests the respective interrupt
			0	disabled
IER.6	ALIE	Arbitration Lost Interrupt Enable	1	enabled; if the CAN controller has lost arbitration, the respective interrupt is requested
			0	disabled
IER.5	EPIE	Error Passive Interrupt Enable	1	enabled; if the error status of the CAN controller changes from error active to error passive or vice versa, the respective interrupt is requested
			0	disabled
IER.4	WUIE	Wake-Up Interrupt Enable	1	enabled; if the sleeping CAN controller wakes up, the respective interrupt is requested
			0	disabled
IER.3	DOIE	Data Overrun Interrupt Enable	1	enabled; if the data overrun status bit is set (see status register; Table 14), the CAN controller requests the respective interrupt
			0	disabled
IER.2	EIE	Error Warning Interrupt Enable	1	enabled; if the error or bus status change (see status register; Table 14), the CAN controller requests the respective interrupt
			0	disabled
IER.1	TIE	Transmit Interrupt Enable	1	enabled; when a message has been successfully transmitted or the transmit buffer is accessible again (e.g. after an abort transmission command), the CAN controller requests the respective interrupt
			0	disabled
IER.0	RIE	Receive Interrupt Enable; note 1	1	enabled; when the receive buffer status is 'full' the CAN controller requests the respective interrupt
			0	disabled

Note

1. The receive interrupt enable bit has direct influence to the receive interrupt bit and the external interrupt output $\overline{INT}$. If RIE is cleared, the external $\overline{INT}$ pin will become HIGH immediately, if there is no other interrupt pending.

6.4.8 Arbitration Lost Capture register (ALC)

This register contains information about the bit position of losing arbitration. The arbitration lost capture register appears to the CPU as a read only memory. Reserved bits are read as logic 0.

Table 17 Bit interpretation of the arbitration lost capture register (ALC); CAN address 11

BIT	SYMBOL	NAME	VALUE	FUNCTION
ALC.7 to ALC.5	–	reserved	For value and function see Table 18	
ALC.4	BITNO4	bit number 4		
ALC.3	BITNO3	bit number 3		
ALC.2	BITNO2	bit number 2		
ALC.1	BITNO1	bit number 1		
ALC.0	BITNO0	bit number 0		

On arbitration lost, the corresponding arbitration lost interrupt is forced, if enabled. At the same time, the current bit position of the bit stream processor is captured into the arbitration lost capture register. The content within this register is fixed until the users software has read out its contents once. The capture mechanism is then activated again.

The corresponding interrupt flag located in the interrupt register is cleared during the read access to the interrupt register. A new arbitration lost interrupt is not possible until the arbitration lost capture register is read out once.

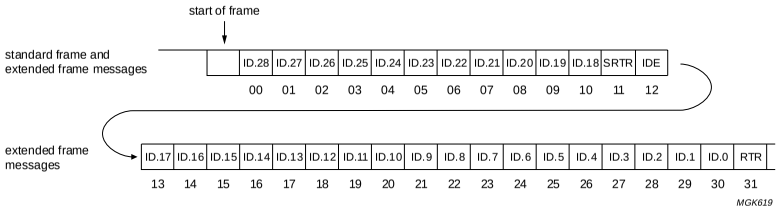


Fig.5 Arbitration lost bit number interpretation.

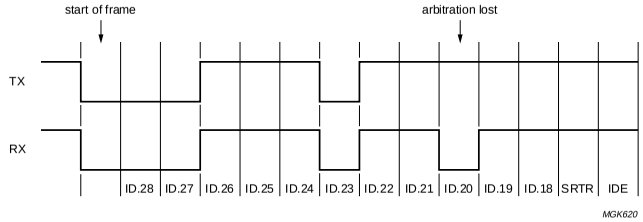


Fig.6 Example of arbitration lost bit number interpretation; result: ALC = 08.

CAN 14 Rel. 5.00
Page A-15
ITALIAN TECHNOLOGY grifo® abaco bus

Table 18 Function of bits 4 to 0 of the arbitration lost capture register

BITS[1]					DECIMAL VALUE	FUNCTION
ALC.4	ALC.3	ALC.2	ALC.1	ALC.0		
0	0	0	0	0	00	arbitration lost in bit 1 of identifier
0	0	0	0	1	01	arbitration lost in bit 2 of identifier
0	0	0	1	0	02	arbitration lost in bit 3 of identifier
0	0	0	1	1	03	arbitration lost in bit 4 of identifier
0	0	1	0	0	04	arbitration lost in bit 5 of identifier
0	0	1	0	1	05	arbitration lost in bit 6 of identifier
0	0	1	1	0	06	arbitration lost in bit 7 of identifier
0	0	1	1	1	07	arbitration lost in bit 8 of identifier
0	1	0	0	0	08	arbitration lost in bit 9 of identifier
0	1	0	0	1	09	arbitration lost in bit 10 of identifier
0	1	0	1	0	10	arbitration lost in bit 11 of identifier
0	1	0	1	1	11	arbitration lost in bit SRTR; note 2
0	1	1	0	0	12	arbitration lost in bit IDE
0	1	1	0	1	13	arbitration lost in bit 12 of identifier; note 3
0	1	1	1	0	14	arbitration lost in bit 13 of identifier; note 3
0	1	1	1	1	15	arbitration lost in bit 14 of identifier; note 3
1	0	0	0	0	16	arbitration lost in bit 15 of identifier; note 3
1	0	0	0	1	17	arbitration lost in bit 16 of identifier; note 3
1	0	0	1	0	18	arbitration lost in bit 17 of identifier; note 3
1	0	0	1	1	19	arbitration lost in bit 18 of identifier; note 3
1	0	1	0	0	20	arbitration lost in bit 19 of identifier; note 3
1	0	1	0	1	21	arbitration lost in bit 20 of identifier; note 3
1	0	1	1	0	22	arbitration lost in bit 21 of identifier; note 3
1	0	1	1	1	23	arbitration lost in bit 22 of identifier; note 3
1	1	0	0	0	24	arbitration lost in bit 23 of identifier; note 3
1	1	0	0	1	25	arbitration lost in bit 24 of identifier; note 3
1	1	0	1	0	26	arbitration lost in bit 25 of identifier; note 3
1	1	0	1	1	27	arbitration lost in bit 26 of identifier; note 3
1	1	1	0	0	28	arbitration lost in bit 27 of identifier; note 3
1	1	1	0	1	29	arbitration lost in bit 28 of identifier; note 3
1	1	1	1	0	30	arbitration lost in bit 29 of identifier; note 3
1	1	1	1	1	31	arbitration lost in bit RTR; note 3

Notes

1. Binary coded frame bit number where arbitration was lost.
2. Bit RTR for standard frame messages.
3. Extended frame messages only.

6.4.9 Error Code Capture register (ECC)

This register contains information about the type and location of errors on the bus. The error code capture register appears to the CPU as a read only memory.

Table 19 Bit interpretation of the error code capture register (ECC); CAN address 12

BIT	SYMBOL	NAME	VALUE	FUNCTION
ECC.7[1]	ERRC1	Error Code 1	–	–
ECC.6[1]	ERRC0	Error Code 0	–	–
ECC.5	DIR	Direction	1	RX; error occurred during reception
			0	TX; error occurred during transmission
ECC.4[2]	SEG4	Segment 4	–	–
ECC.3[2]	SEG3	Segment 3	–	–
ECC.2[2]	SEG2	Segment 2	–	–
ECC.1[2]	SEG1	Segment 1	–	–
ECC.0[2]	SEG0	Segment 0	–	–

Notes

1. For bit interpretation of bits ECC.7 and ECC.6 see Table 20.
2. For bit interpretation of bits ECC.4 to ECC.0 see Table 21.

Table 20 Bit interpretation of bits ECC.7 and ECC.6

BIT ECC.7	BIT ECC.6	FUNCTION
0	0	bit error
0	1	form error
1	0	stuff error
1	1	other type of error

Table 21 Bit interpretation of bits ECC.4 to ECC.0; note 1

BIT ECC.4	BIT ECC.3	BIT ECC.2	BIT ECC.1	BIT ECC.0	FUNCTION
0	0	0	1	1	start of frame
0	0	0	1	0	ID.28 to ID.21
0	0	1	1	0	ID.20 to ID.18
0	0	1	0	0	bit SRTR
0	0	1	0	1	bit IDE
0	0	1	1	1	ID.17 to ID.13
0	1	1	1	1	ID.12 to ID.5
0	1	1	1	0	ID.4 to ID.0
0	1	1	0	0	bit RTR
0	1	1	0	1	reserved bit 1
0	1	0	0	1	reserved bit 0
0	1	0	1	1	data length code
0	1	0	1	0	data field
0	1	0	0	0	CRC sequence
1	1	0	0	0	CRC delimiter
1	1	0	0	1	acknowledge slot
1	1	0	1	1	acknowledge delimiter
1	1	0	1	0	end of frame
1	0	0	1	0	intermission
1	0	0	0	1	active error flag
1	0	1	1	0	passive error flag
1	0	0	1	1	tolerate dominant bits
1	0	1	1	1	error delimiter
1	1	1	0	0	overload flag

Note

1. Bit settings reflect the current frame segment to distinguish between different error events.

If a bus error occurs, the corresponding bus error interrupt is always forced, if enabled. At the same time, the current position of the bit stream processor is captured into the error code capture register. The content within this register is fixed until the users software has read out its content once. The capture mechanism is then activated again.

The corresponding interrupt flag located in the interrupt register is cleared during the read access to the interrupt register. A new bus error interrupt is not possible until the capture register is read out once.

6.4.10 Error Warning Limit Register (EWLR)

The error warning limit can be defined within this register. The default value (after hardware reset) is 96. In reset mode this register appears to the CPU as a read/write memory. In operating mode it is read only.

Note, that a content change of the EWLR is only possible, if the reset mode was entered previously. An error status change (see status register; Table 14) and an error warning interrupt forced by the new register content will not occur until the reset mode is cancelled again.

Table 22 Bit interpretation of the error warning limit register (EWLR); CAN address 13

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
EWL.7	EWL.6	EWL.5	EWL.4	EWL.3	EWL.2	EWL.1	EWL.0

6.4.11 RX Error Counter Register (RXERR)

The RX error counter register reflects the current value of the receive error counter. After a hardware reset this register is initialized to logic 0. In operating mode this register appears to the CPU as a read only memory. A write access to this register is possible only in reset mode.

If a bus-off event occurs, the RX error counter is initialized to logic 0. The time bus-off is valid, writing to this register has no effect.

Note, that a CPU-forced content change of the RX error counter is only possible, if the reset mode was entered previously. An error status change (see status register; Table 14), an error warning or an error passive interrupt forced by the new register content will not occur, until the reset mode is cancelled again.

Table 23 Bit interpretation of the RX error counter register (RXERR); CAN address 14

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
RXERR.7	RXERR.6	RXERR.5	RXERR.4	RXERR.3	RXERR.2	RXERR.1	RXERR.0

6.4.12 TX Error Counter Register (TXERR)

The TX error counter register reflects the current value of the transmit error counter.

In operating mode this register appears to the CPU as a read only memory. A write access to this register is possible only in reset mode. After a hardware reset this register is initialized to logic 0. If a bus-off event occurs, the TX error counter is initialized to 127 to count the minimum protocol-defined time (128 occurrences of the bus-free signal). Reading the TX error counter during this time gives information about the status of the bus-off recovery.

If bus-off is active, a write access to TXERR in the range from 0 to 254 clears the bus-off flag and the controller will wait for one occurrence of 11 consecutive recessive bits (bus-free) after the reset mode has been cleared.

Table 24 Bit interpretation of the TX error counter register (TXERR); CAN address 15

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
TXERR.7	TXERR.6	TXERR.5	TXERR.4	TXERR.3	TXERR.2	TXERR.1	TXERR.0

Writing 255 to TXERR allows to initiate a CPU-driven bus-off event. It should be noted that a CPU-forced content change of the TX error counter is only possible, if the reset mode was entered previously. An error or bus status change (see status register; Table 14), an error warning or an error passive interrupt forced by the new register content will not occur until the reset mode is cancelled again. After leaving the reset mode, the new TX counter content is interpreted and the bus-off event is performed in the same way, as if it was forced by a bus error event. That means, that the reset mode is entered again, the TX error counter is initialized to 127, the RX counter is cleared and all concerned status and interrupt register bits are set.

Clearing of reset mode now will perform the protocol-defined bus-off recovery sequence (waiting for 128 occurrences of the bus-free signal).

If the reset mode is entered again before the end of bus-off recovery (TXERR > 0), bus-off keeps active and TXERR is frozen.

ITALIAN TECHNOLOGY grifo® abaco bus

6.4.13 TRANSMIT BUFFER

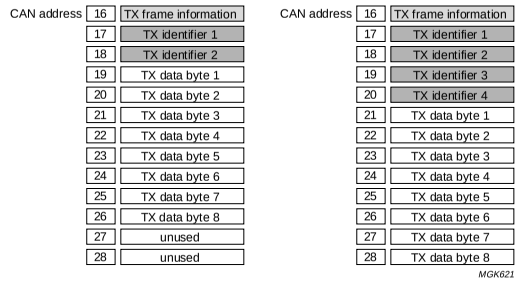
The global layout of the transmit buffer is shown in Fig.7. One has to distinguish between the Standard Frame Format (SFF) and the Extended Frame Format (EFF) configuration. The transmit buffer allows the definition of one transmit message with up to eight data bytes.

6.4.13.1 Transmit buffer layout

The transmit buffer layout is subdivided into descriptor and data fields where the first byte of the descriptor field is the frame information byte (frame information). It describes the frame format (SFF or EFF), remote or data frame and the data length. Two identifier bytes for SFF or four bytes for EFF messages follow. The data field contains up to eight data bytes.

The transmit buffer has a length of 13 bytes and is located in the CAN address range from 16 to 28.

Note, that a direct access to the transmit buffer RAM is possible using the CAN address space from 96 to 108. This RAM area is reserved for the transmit buffer. The three following bytes may be used for general purposes (CAN address 109, 110 and 111).



a. Standard frame format. b. Extended frame format.

Fig.7 Transmit buffer layout for standard and extended frame format configurations.

6.4.13.2 Descriptor field of the transmit buffer

The bit layout of the transmit buffer is represented in Tables 25 to 27 for SFF and Tables 28 to 32 for EFF. The given configuration is chosen to be compatible with the receive buffer layout (see Section 6.4.14.1).

Table 25 TX frame information (SFF); CAN address 16

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
FF(1)	RTR(2)	X(3)	X(3)	DLC.3(4)	DLC.2(4)	DLC.1(4)	DLC.0(4)

Notes

1. Frame format.
2. Remote transmission request.
3. Don't care; recommended to be compatible to receive buffer (0) in case of using the self reception facility (self test).
4. Data length code bit.

Table 26 TX identifier 1 (SFF); CAN address 17; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21

Note

1. ID.X means identifier bit X.

Table 27 TX identifier 2 (SFF); CAN address 18; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.20	ID.19	ID.18	X(2)	X(3)	X(3)	X(3)	X(3)

Notes

1. ID.X means identifier bit X.
2. Don't care; recommended to be compatible to receive buffer (RTR) in case of using the self reception facility (self test).
3. Don't care; recommended to be compatible to receive buffer (0) in case of using the self reception facility (self test).

Table 28 TX frame information (EFF); CAN address 16

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
FF(1)	RTR(2)	X(3)	X(3)	DLC.3(4)	DLC.2(4)	DLC.1(4)	DLC.0(4)

Notes

1. Frame format.
2. Remote transmission request.
3. Don't care; recommended to be compatible to receive buffer (0) in case of using the self reception facility (self test).
4. Data length code bit.

Table 29 TX identifier 1 (EFF); CAN address 17; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21

Note

1. ID.X means identifier bit X.

Table 30 TX identifier 2 (EFF); CAN address 18; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13

Note

1. ID.X means identifier bit X.

Table 31 TX identifier 3 (EFF); CAN address 19; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5

Note

1. ID.X means identifier bit X.

Table 32 TX identifier 4 (EFF); CAN address 20; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.4	ID.3	ID.2	ID.1	ID.0	X(2)	X(3)	X(3)

Notes

1. ID.X means identifier bit X.
2. Don't care; recommended to be compatible to receive buffer (RTR) in case of using the self reception facility (self test).
3. Don't care; recommended to be compatible to receive buffer (0) in case of using the self reception facility (self test).

Table 33 Frame Format (FF) and Remote Transmission Request (RTR) bits

BIT	VALUE	FUNCTION
FF	1	EFF; extended frame format will be transmitted by the CAN controller
	0	SFF; standard frame format will be transmitted by the CAN controller
RTR	1	remote; remote frame will be transmitted by the CAN controller
	0	data; data frame will be transmitted by the CAN controller

6.4.13.3 Data Length Code (DLC)

The number of bytes in the data field of a message is coded by the data length code. At the start of a remote frame transmission the data length code is not considered due to the RTR bit being logic 1 (remote). This forces the number of transmitted/received data bytes to be 0. Nevertheless, the data length code must be specified correctly to avoid bus errors, if two CAN controllers start a remote frame transmission with the same identifier simultaneously.

The range of the data byte count is 0 to 8 bytes and is coded as follows:

DataByteCount = 8 × DLC.3 + 4 × DLC.2 + 2 × DLC.1 + DLC.0

For reasons of compatibility no data length code >8 should be used. If a value >8 is selected, 8 bytes are transmitted in the data frame with the Data Length Code specified in DLC.

6.4.13.4 Identifier (ID)

In Standard Frame Format (SFF) the identifier consists of 11 bits (ID.28 to ID.18) and in Extended Frame Format (EFF) messages the identifier consists of 29 bits (ID.28 to ID.0). ID.28 is the most significant bit, which is transmitted first on the bus during the arbitration process. The identifier acts as the message's name, used in a receiver for acceptance filtering, and also determines the bus access priority during the arbitration process.

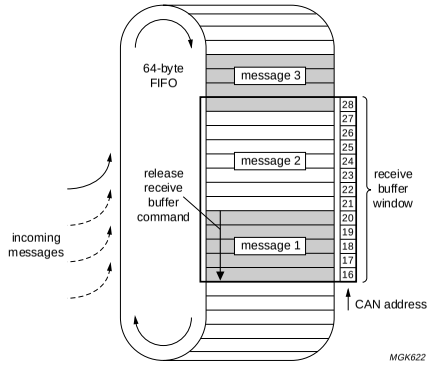
The lower the binary value of the identifier the higher the priority. This is due to the larger number of leading dominant bits during arbitration.

6.4.13.5 Data field

The number of transferred data bytes is defined by the data length code. The first bit transmitted is the most significant bit of data byte 1 at CAN address 19 (SFF) or CAN address 21 (EFF).

6.4.14 RECEIVE BUFFER

The global layout of the receive buffer is very similar to the transmit buffer described in the previous section.
The receive buffer is the accessible part of the RXFIFO and is located in the range between CAN address 16 and 28. Each message is subdivided into a descriptor and a data field.



Message 1 is now available in the receive buffer.

Fig.8 Example of the message storage within the RXFIFO.

6.4.14.1 Descriptor field of the receive buffer

The bit layout of the receive buffer is represented in Tables 34 to 36 for SFF and Tables 37 to 41 for EFF. The given configuration is chosen to be compatible with the transmit buffer layout (see Section 6.4.13.2).

Table 34 RX frame information (SFF); CAN address 16

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
FF(1)	RTR(2)	0	0	DLC.3(3)	DLC.2(3)	DLC.1(3)	DLC.0(3)

Notes

1. Frame format.
2. Remote transmission request.
3. Data length code bit.

CAN 14 Rel. 5.00 Page A-19
ITALIAN TECHNOLOGY grifo® abaco bus

Table 35 RX identifier 1 (SFF); CAN address 17; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21

Note

1. ID.X means identifier bit X.

Table 36 RX identifier 2 (SFF); CAN address 18; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.20	ID.19	ID.18	RTR(2)	0	0	0	0

Notes

1. ID.X means identifier bit X.
2. Remote transmission request.

Table 37 RX frame information (EFF); CAN address 16

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
FF(1)	RTR(2)	0	0	DLC.3(3)	DLC.2(3)	DLC.1(3)	DLC.0(3)

Notes

1. Frame format.
2. Remote transmission request.
3. Data length code bit.

Table 38 RX identifier 1 (EFF); CAN address 17; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21

Note

1. ID.X means identifier bit X.

Table 39 RX identifier 2 (EFF); CAN address 18; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13

Note

1. ID.X means identifier bit X.

Table 40 RX identifier 3 (EFF); CAN address 19; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5

Note

1. ID.X means identifier bit X.

Table 41 RX identifier 4 (EFF); can address 20; note 1

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
ID.4	ID.3	ID.2	ID.1	ID.0	RTR(2)	0	0

Notes

1. ID.X means identifier bit X.
2. Remote transmission request.

Remark: the received data length code located in the frame information byte represents the real sent data length code, which may be greater than 8 (depends on sender). Nevertheless the maximum number of received data bytes is 8. This should be taken into account by reading a message from the receive buffer.

As described in Fig.8 the RXFIFO has space for 64 message bytes in total. It depends on the data length how many messages can fit in it at one time. If there is not enough space for a new message within the RXFIFO, the CAN controller generates a data overrun condition the moment this message becomes valid and the acceptance test was positive. A message which is partly written into the RXFIFO, when the data overrun situation occurs, is deleted. This situation is indicated to the CPU via the status register and the data overrun interrupt, if enabled.

6.4.15 Acceptance filter

With the help of the acceptance filter the CAN controller is able to allow passing of received messages to the RXFIFO only when the identifier bits of the received message are equal to the predefined ones within the acceptance filter registers.

The acceptance filter is defined by the Acceptance Code Registers (ACRn) and the Acceptance Mask Registers (AMRn). The bit patterns of messages to be received are defined within the acceptance code registers.
The corresponding acceptance mask registers allow to define certain bit positions to be 'don't care'.

Two different filter modes are selectable within the mode register (MOD.3, AFM; see Section 6.4.3):

- Single filter mode (bit AFM is logic 1)
- Dual filter mode (bit AFM is logic 0).

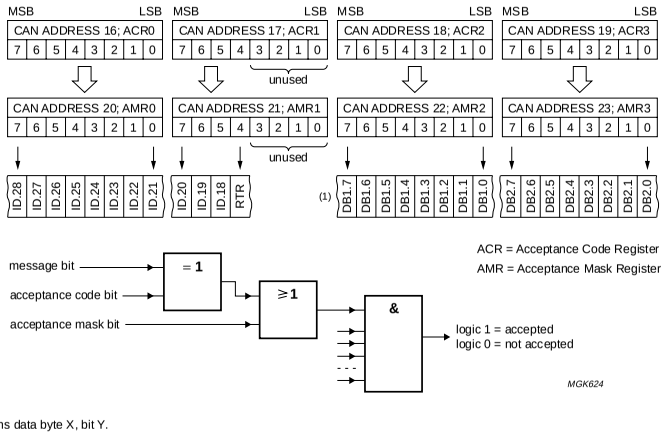
6.4.15.1 Single filter configuration

In this filter configuration one long filter (4-bytes) could be defined. The bit correspondences between the filter bytes and the message bytes depend on the currently received frame format.

Standard frame: if a standard frame format message is received, the complete identifier including the RTR bit and the first two data bytes are used for acceptance filtering. Messages may also be accepted if there are no data bytes existing due to a set RTR bit or if there is none or only one data byte because of the corresponding data length code.

For a successful reception of a message, all single bit comparisons have to signal acceptance.
Note, that the 4 least significant bits of AMR1 and ACR1 are not used. In order to be compatible with future products these bits should be programmed to be 'don't care' by setting AMR1.3, AMR1.2, AMR1.1 and AMR1.0 to logic 1.

abaco bus grifo® ITALIAN TECHNOLOGY



DBX.Y means data byte X, bit Y.

Fig.9 Single filter configuration, receiving standard frame messages.

Extended frame: if an extended frame format message is received, the complete identifier including the RTR bit is used for acceptance filtering.

For a successful reception of a message, all single bit comparisons have to signal acceptance.
It should be noted that the 2 least significant bits of AMR3 and ACR3 are not used. In order to be compatible with future products these bits should be programmed to be 'don't care' by setting AMR3.1 and AMR3.0 to logic 1.

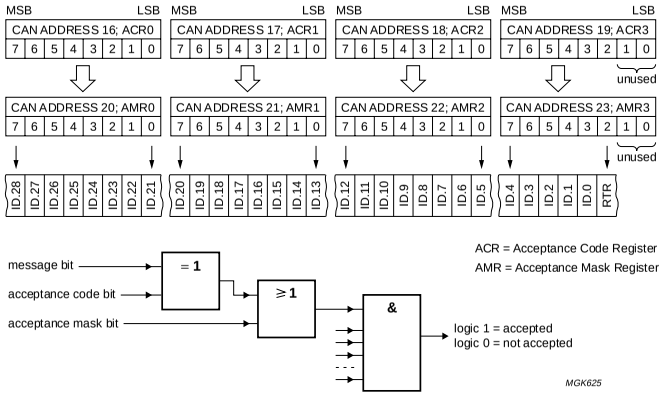


Fig.10 Single filter configuration, receiving extended frame messages.

6.4.15.2 Dual filter configuration

In this filter configuration two short filters can be defined. A received message is compared with both filters to decide, whether this message should be copied into the receive buffer or not. If at least one of the filters signals an acceptance, the received message becomes valid. The bit correspondences between the filter bytes and the message bytes depends on the currently received frame format.

Standard frame: if a standard frame message is received, the two defined filters are looking different. The first filter compares the complete standard identifier including the RTR bit and the first data byte of the message. The second filter just compares the complete standard identifier including the RTR bit.

For a successful reception of a message, all single bit comparisons of at least one complete filter have to signal acceptance. In case of a set RTR bit or a data length code of logic 0 no data byte is existing. Nevertheless a message may pass filter 1, if the first part up to the RTR bit signals acceptance.

If no data byte filtering is required for filter 1, the four least significant bits of AMR1 and AMR3 have to be set to logic 1 (don't care). Then both filters are working identically using the standard identifier range including the RTR bit.

ITALIAN TECHNOLOGY grifo® abaco bus

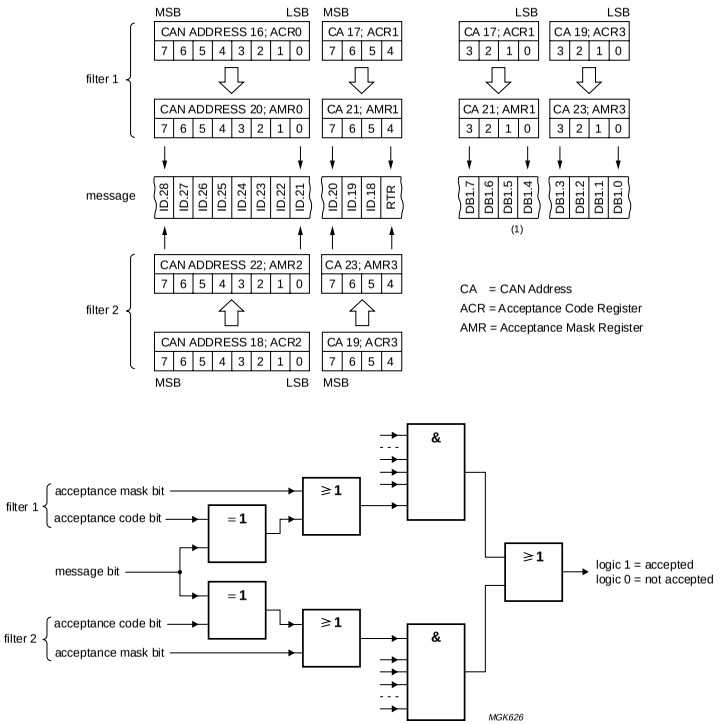


Fig.11 Dual filter configuration, receiving standard frame messages.

Extended frame: if an extended frame message is received, the two defined filters are looking identically. Both filters are comparing the first two bytes of the extended identifier range only.

For a successful reception of a message, all single bit comparisons of at least one complete filter have to indicate acceptance.

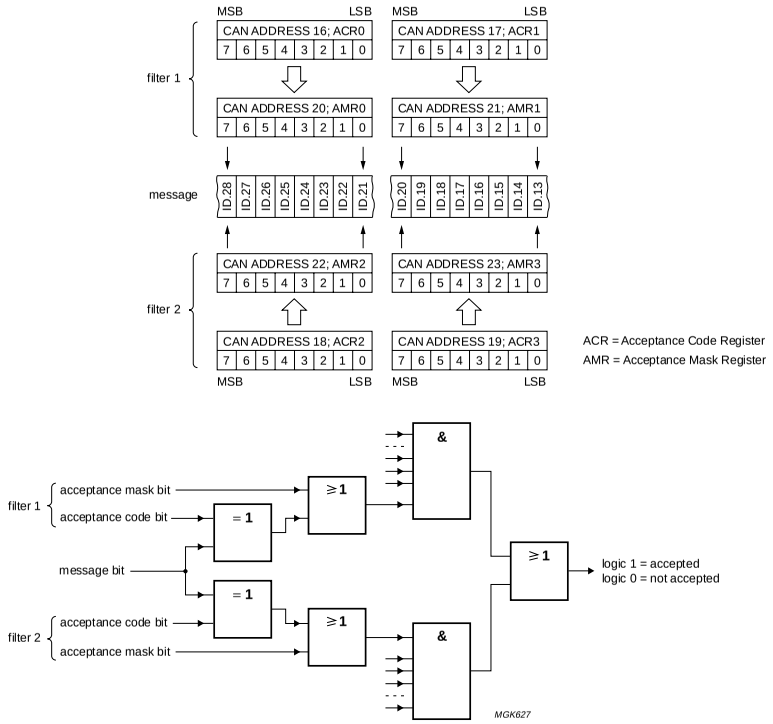


Fig.12 Dual filter configuration, receiving extended frame messages.

6.4.16 RX MESSAGE COUNTER (RMC)

The RMC register (CAN address 29) reflects the number of messages available within the RXFIFO. The value is incremented with each receive event and decremented by the release receive buffer command. After any reset event, this register is cleared.

Table 42 Bit interpretation of the RX message counter (RMC); CAN address 29

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
(0)[1]	(0)[1]	(0)[1]	RMC.4	RMC.3	RMC.2	RMC.1	RMC.0

Note

1. This bit cannot be written. During read-out of this register always a zero is given.

6.4.17 RX BUFFER START ADDRESS REGISTER (RBSA)

The RBSA register (CAN address 30) reflects the currently valid internal RAM address, where the first byte of the received message, which is mapped to the receive buffer window, is stored. With the help of this information it is possible to interpret the internal RAM contents.
The internal RAM address area begins at CAN address 32 and may be accessed by the CPU for reading and writing (writing in reset mode only).

Example: if RBSA is set to 24 (decimal), the current message visible in the receive buffer window (CAN address 16 to 28) is stored within the internal RAM beginning at RAM address 24. Because the RAM is also mapped directly to the CAN address space beginning at CAN address 32 (equal to RAM address 0) this message may also be accessed using CAN address 56 and the following bytes
(CAN address = RBSA + 32 > 24 + 32 = 56).

If a message exceeds RAM address 63, it continues at RAM address 0.

The release receive buffer command is always given while there is at least one more message available within the FIFO. RBSA is updated to the beginning of the next message.

On hardware reset, this pointer is initialized to '00H'. Upon a software reset (setting of reset mode) this pointer keeps its old value, but the FIFO is cleared; this means that the RAM contents are not changed, but the next received (or transmitted) message will override the currently visible message within the receive buffer window.

The RX buffer start address register appears to the CPU as a read only memory in operating mode and as read/write memory in reset mode. It should be noted that a write access to RBSA takes effect first after the next positive edge of the internal clock frequency, which is half of the external oscillator frequency.

Table 43 Bit interpretation of the RX buffer start address register (RBSA); CAN address 30

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
(0)[1]	(0)[1]	RBSA.5	RBSA.4	RBSA.3	RBSA.2	RBSA.1	RBSA.0

Note

1. This bit cannot be written. During read-out of this register always a zero is given.

6.5 Common registers

6.5.1 BUS TIMING REGISTER 0 (BTR0)

The contents of the bus timing register 0 defines the values of the Baud Rate Prescaler (BRP) and the Synchronization Jump Width (SJW). This register can be accessed (read/write) if the reset mode is active.

In operating mode this register is read only, if the PeliCAN mode is selected. In BasicCAN mode a 'FFH' is reflected.

Table 44 Bit interpretation of bus timing register 0 (BTR0); CAN address 6

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SJW.1	SJW.0	BRP.5	BRP.4	BRP.3	BRP.2	BRP.1	BRP.0

6.5.1.1 Baud Rate Prescaler (BRP)

The period of the CAN system clock t_{scl} is programmable and determines the individual bit timing. The CAN system clock is calculated using the following equation:

$$t_{scl} = 2 \times t_{CLK} \times (32 \times BRP.5 + 16 \times BRP.4 + 8 \times BRP.3 + 4 \times BRP.2 + 2 \times BRP.1 + BRP.0 + 1)$$

where t_{CLK} = time period of the XTAL frequency = $\frac{1}{f_{XTAL}}$

6.5.1.2 Synchronization Jump Width (SJW)

To compensate for phase shifts between clock oscillators of different bus controllers, any bus controller must re-synchronize on any relevant signal edge of the current transmission. The synchronization jump width defines the maximum number of clock cycles a bit period may be shortened or lengthened by one re-synchronization:

$$t_{SJW} = t_{scl} \times (2 \times SJW.1 + SJW.0 + 1)$$

6.5.2 BUS TIMING REGISTER 1 (BTR1)

The contents of bus timing register 1 defines the length of the bit period, the location of the sample point and the number of samples to be taken at each sample point. This register can be accessed (read/write) if the reset mode is active.

In operating mode, this register is read only, if the PeliCAN mode is selected. In BasicCAN mode a 'FFH' is reflected.

Table 45 Bit interpretation of bus timing register 1 (BTR1); CAN address 7

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SAM	TSEG2.2	TSEG2.1	TSEG2.0	TSEG1.3	TSEG1.2	TSEG1.1	TSEG1.0

6.5.2.1 Sampling (SAM)

BIT	VALUE	FUNCTION
SAM	1	triple; the bus is sampled three times; recommended for low/medium speed buses (class A and B) where filtering spikes on the bus line is beneficial
	0	single; the bus is sampled once; recommended for high speed buses (SAE class C)

ITALIAN TECHNOLOGY grifo® abaco bus

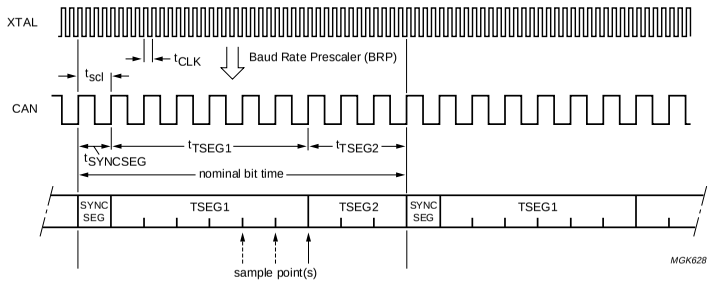
6.5.2.2 Time Segment 1 (TSEG1) and Time Segment 2 (TSEG2)

TSEG1 and TSEG2 determine the number of clock cycles per bit period and the location of the sample point, where:

$t_{SYNCSEG} = 1 \times t_{scl}$

$t_{TSEG1} = t_{scl} \times (8 \times TSEG1.3 + 4 \times TSEG1.2 + 2 \times TSEG1.1 + TSEG1.0 + 1)$

$t_{TSEG2} = t_{scl} \times (4 \times TSEG2.2 + 2 \times TSEG2.1 + TSEG2.0 + 1)$



Possible values are BRP = 000001, TSEG1 = 0101 and TSEG2 = 010.

Fig.13 General structure of a bit period.

6.5.3 OUTPUT CONTROL REGISTER (OCR)

The output control register allows the set-up of different output driver configurations under software control.

This register may be accessed (read/write) if the reset mode is active. In operating mode, this register is read only, if the PeliCAN mode is selected. In BasicCAN mode a 'FFH' is reflected.

Table 46 Bit interpretation of the output control register (OCR); CAN address 8

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
OCTP1	OCTN1	OCPOL1	OCTP0	OCTN0	OCPOL0	OCMODE1	OCMODE0

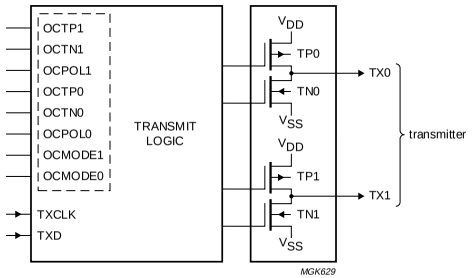


Fig.14 Transceiver input/output control logic.

If the SJA1000 is in the sleep mode a recessive level is output on the TX0 and TX1 pins with respect to the contents within the output control register. If the SJA1000 is in the reset state (reset request = HIGH) or the external reset pin $\overline{RST}$ is pulled LOW the outputs TX0 and TX1 are floating.

The transmit output stage is able to operate in different modes. Table 47 shows the output control register settings.

Table 47 Interpretation of OCMODE bits

OCMODE1	OCMODE0	DESCRIPTION
0	0	bi-phase output mode
0	1	test output mode; note 1
1	0	normal output mode
1	1	clock output mode

Note

1. In test output mode TXn will reflect the bit, detected on RX pins, with the next positive edge of the system clock. TN1, TN0, TP1 and TP0 are configured in accordance with the setting of OCR.

6.5.3.1 Normal output mode

In normal output mode the bit sequence (TXD) is sent via TX0 and TX1. The voltage levels on the output driver pins TX0 and TX1 depend on both the driver characteristic programmed by OCTPx, OCTNx (float, pull-up, pull-down, push-pull) and the output polarity programmed by OCPOLx.

6.5.3.2 Clock output mode

For the TX0 pin this is the same as in normal output mode. However, the data stream to TX1 is replaced by the transmit clock (TXCLK). The rising edge of the transmit clock (non-inverted) marks the beginning of a bit period. The clock pulse width is $1 \times t_{scl}$.

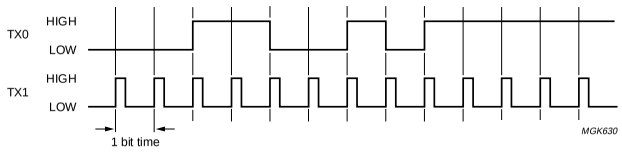


Fig.15 Example of clock output mode.

6.5.3.3 Bi-phase output mode

In contrast to the normal output mode the bit representation is time variant and toggled. If the bus controllers are galvanically decoupled from the bus line by a transformer, the bit stream is not allowed to contain a DC component. This is achieved by the following scheme. During recessive bits all outputs are deactivated (floating). Dominant bits are sent with alternating levels on TX0 and TX1, i.e. the first dominant bit is sent on TX0, the second is sent on TX1, and the third one is sent on TX0 again, and so on. One possible configuration example of the bi-phase output mode timing is shown in Fig.16.

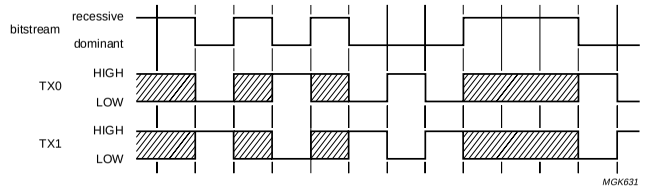


Fig.16 Bi-phase output mode example (output control register = F8H).

6.5.3.4 Test output mode

In test output mode the level connected to RX is reflected at TXn with the next positive edge of the system clock $\frac{f_{osc}}{2}$ corresponding to the programmed polarity in the output control register.

Table 48 shows the relationship between the bits of the output control register and the output pins TX0 and TX1.

CAN 14 Rel. 5.00
Page A-25
ITALIAN TECHNOLOGY grifo® abaco® bus

Table 48 Output pin configuration; note 1

DRIVE	TXD	OCTPX	OCTNX	OCPOLX	TPX[2]	TNX[3]	TXX[4]
Float	X	0	0	X	off	off	float
Pull-down	0	0	1	0	off	on	LOW
	1	0	1	0	off	off	float
	0	0	1	1	off	off	float
	1	0	1	1	off	on	LOW
Pull-up	0	1	0	0	off	off	float
	1	1	0	0	on	off	HIGH
	0	1	0	1	on	off	HIGH
	1	1	0	1	off	off	float
Push-pull	0	1	1	0	off	on	LOW
	1	1	1	0	on	off	HIGH
	0	1	1	1	on	off	HIGH
	1	1	1	1	off	on	LOW

Notes

1. X = don't care.
2. TPX is the on-chip output transistor X, connected to V_{DD}.
3. TNX is the on-chip output transistor X, connected to V_{SS}.
4. TXX is the serial output level on pin TX0 or TX1. It is required that the output level on the CAN-bus line is dominant when TXD = 0 and recessive when TXD = 1.

The bit sequence (TXD) is sent via TX0 and TX1.
The voltage levels on the output driver pins depends on both the driver characteristics programmed by OCTP, OCTN (float, pull-up, pull-down, push-pull) and the output polarity programmed by OCPOL.

6.5.4 Clock Divider Register (CDR)

The clock divider register controls the CLKOUT frequency for the microcontroller and allows to deactivate the CLKOUT pin. Additionally a dedicated receive interrupt pulse on TX1, a receive comparator bypass and the selection between BasicCAN mode and PeliCAN mode is made here. The default state of the register after hardware reset is divide-by-12 for Motorola mode (00000101) and divide-by-2 for Intel mode (00000000).

On software reset (reset request/reset mode) this register is not influenced.

The reserved bit (CDR.4) will always reflect a logic 0. The application software should always write a logic 0 to this bit in order to be compatible with future features, which may be 1-active using this bit.

Table 49 Bit interpretation of the clock divider register (CDR); CAN address 31

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
CAN mode	CBP	RXINTEN	(0)[1]	clock off	CD.2	CD.1	CD.0

Note

1. This bit cannot be written. During read-out of this register always a zero is given.

6.5.4.1 CD.2 to CD.0

The bits CD.2 to CD.0 are accessible without restrictions in reset mode as well as in operating mode. These bits are used to define the frequency at the external CLKOUT pin. For an overview of selectable frequencies see Table 50.

Table 50 CLKOUT frequency selection; note 1

CD.2	CD.1	CD.0	CLKOUT FREQUENCY
0	0	0	$\frac{f_{osc}}{2}$
0	0	1	$\frac{f_{osc}}{4}$
0	1	0	$\frac{f_{osc}}{6}$
0	1	1	$\frac{f_{osc}}{8}$
1	0	0	$\frac{f_{osc}}{10}$
1	0	1	$\frac{f_{osc}}{12}$
1	1	0	$\frac{f_{osc}}{14}$
1	1	1	f_{osc}

Note

1. f_{osc} is the frequency of the external oscillator (XTAL).

6.5.4.2 Clock off

Setting of this bit allows to disable the external CLKOUT pin of the SJA1000. A write access is possible only in reset mode (reset request bit is set in BasicCAN mode).

6.5.4.3 RXINTEN

This bit allows to use the TX1 output as a dedicated receive interrupt output. When a received message has passed the acceptance filter successfully, a receive interrupt pulse with the length of one bit time is always output at the TX1 pin (during the last bit of end of frame). The polarity and output drive are programmable via the output control register (see also Section 6.5.3). A write access is only possible in reset mode (the reset request bit is set in BasicCAN mode).

6.5.4.4 CBP

Setting of CDR.6 allows to bypass the CAN input comparator and is only possible in reset mode. This is useful in the event that the SJA1000 is connected to an external transceiver circuit. The internal delay of the SJA1000 is reduced, which will result in a longer maximum possible bus length. If CBP is set, only RX0 is active. The unused RX1 input should be connected to a defined level (e.g. V_{SS}).

6.5.4.5 CAN mode

CDR.7 defines the CAN mode. If CDR.7 is at logic 0 the CAN controller operates in BasicCAN mode. If set to logic 1 the CAN controller operates in PeliCAN mode. Write access is only possible in reset mode.

APPENDIX B: ALPHABETICAL INDEX

SYMBOLS
+5 VDC **6**, **10**
/INT **10**
/NMI **10**
/RESET **14**
82C250 **4**

A

ABACO® I/O BUS **4**, **12**, **14**, **16**
ABACO® I/O BUS CONNECTOR **10**
ABB 03 **4**
ABB 05 **4**
ACCEPTANCE **4**
ADDR **18**
ADDRESSING **4**, **16**
ADDRESSING SPACE **5**

B

BASICCAN **4**
BAUD RATE **5**, **18**
BIBLIOGRAPHY **25**
BOARD CONNNECTIONS **12**
BOARD MAPPING **16**
BUFFER SIZE **4**
BYTES TAKEN **5**

C

CAN INITIALIZATION **18**
CAN SIGNALS **12**
CAN CONTROLLER **4**, **18**
CAN LINE CONNECTOR **8**
CAN LINE IMPEDANCE **6**
CAN LINE INTERFACE **4**
CAN LINE TERMINATION **13**
CAN TERMINATION NETWORK **6**
CARD VERSION **1**
CLOCK FREQUENCE **5**
CONNECTIONS **8**
  CN1 **10**
  CN2 **8**
CONNECTORS **5**
CURRENT CONSUMPTION **6**

D

DATA **18**
DC/DC CONVERTER **4**
DIN 46277-1 AND 3 **12**
DIP SWITCH **5**
DSW1 **4**, **16**

E

ELECTRIC FEATURES **6**
EXTERNAL CARDS **20**

G

GALVANIC ISOLATION **4**, **12**
GENERAL FEATURES **5**
GENERAL INFORMATION **2**

H

HARDWARE DESCRIPTION **16**

I

IDENTIFIER **4**
INSTALLATION **8**
INTERFACING **4**
INTERNAL REGISTERS ADDRESSING **17**
INTERRUPT **14**
INTERRUPT MANAGEMENT **5**
INTRODUCTION **1**

J

J1 **14**
JUMPERS **12**
  2 PINS JUMPERS **13**
  3 PINS JUMPERS **13**

M

MECHANICAL MOUNTING **12**

O

ON BOARD PERIPHERALS **5**
ON BOARD RESOURCES **5**

P

PCX82C200 **4**, **18**
PELICAN 2.0B **4**
PERIPHERAL DEVICES SOFTWARE DESCRIPTION **18**
PHYSICAL FEATURES **5**
POWER SUPPLY **6**
PROTOCOLS SUPPORTED **5**

R

RELATIVE HUMIDITY **5**
RESET **14**

S

SIZE **5**
SJA 1000 **4**, **18**

T

TECHNICAL FEATURES **5**
TEMPERATURE RANGE **5**
TTL **6**, **12**

W

WEIGHT **5**