

CAN 14

Control Area Network 1 channel, 4 type

MANUALE TECNICO



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>

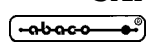
Tel. +39 051 892.052 (r.a.) FAX: +39 051 893.661



CAN 14

Edizione 3.00

Rel. 06 Novembre 1998

, GPC[®], grifo[®], sono marchi registrati della ditta grifo[®]

CAN 14

Control Area Network 1 channel, 4 type

MANUALE TECNICO

Modulo periferico di comunicazione con protocollo CAN; interfaccia per **Abaco® I/O BUS** su connettore a scatola a 26 vie; ingombro di 100x50x20 mm, (110x60x70 mm con contenitore) nel formato serie 4; contenitore per guide ad Ω tipo DIN 46277-1 e DIN 46277-3; connettore vaschetta D a 9 vie per linea di comunicazione CAN; supporto protocollo CAN 2.0B gestito dall'UART PHILIPS SJA 1000 a 24 MHz (compatibile con PHILIPS PCx82C200); driver di linea PHILIPS 82C250 galvanicamente isolato; DC/DC converter per optoisolamento della linea di comunicazione CAN; Baud rate supportati: fino a 400 KBit/sec per linee da centinaia di metri ed 1 MBit/sec per linee di decine di metri; spazio d'indirizzamento occupato di soli 2 bytes consecutivi; dip switch per settare il mappaggio in I/O della scheda; generazione di interrupts in corrispondenza di eventi definibili via software; collegamento interrupt ad /INT o /NMI selezionabile via hardware; unica tensione di alimentazione: +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>

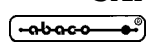
Tel. +39 051 892.052 (r.a.) FAX: +39 051 893.661



CAN 14

Edizione 3.00

Rel. 06 Novembre 1998

, GPC®, grifo®, sono marchi registrati della ditta grifo®

Vincoli sulla documentazione grifo® Tutti i Diritti Riservati

Nessuna parte del presente manuale può essere riprodotta, trasmessa, trascritta, memorizzata in un archivio o tradotta in altre lingue, con qualunque forma o mezzo, sia esso elettronico, meccanico, magnetico ottico, chimico, manuale, senza il permesso scritto della **grifo®**.

IMPORTANTE

Tutte le informazioni contenute sul presente manuale sono state accuratamente verificate, ciononostante **grifo®** non si assume nessuna responsabilità per danni, diretti o indiretti, a cose e/o persone derivanti da errori, omissioni o dall'uso del presente manuale, del software o dell' hardware ad esso associato.

grifo® altresì si riserva il diritto di modificare il contenuto e la veste di questo manuale senza alcun preavviso, con l' intento di offrire un prodotto sempre migliore, senza che questo rappresenti un obbligo per **grifo®**.

Per le informazioni specifiche dei componenti utilizzati sui nostri prodotti, l'utente deve fare riferimento agli specifici Data Book delle case costruttrici o delle seconde sorgenti.

LEGENDA SIMBOLI

Nel presente manuale possono comparire i seguenti simboli:

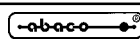


Attenzione: Pericolo generico



Attenzione: Pericolo di alta tensione

Marchi Registrati

 , GPC®, **grifo®** : sono marchi registrati della **grifo®**.

Altre marche o nomi di prodotti sono marchi registrati dei rispettivi proprietari.

INDICE GENERALE

INTRUDIZIONE	1
VERSIONE SCHEDA	1
INFORMAZIONI GENERALI	2
SEZIONE DI INTERFACCIA ED INDIRIZZAMENTO	4
CONTROLLORE CAN	4
INTERFACCIA DI LINEA CAN	4
SPECIFICHE TECNICHE	5
CARATTERISTICHE GENERALI	5
CARATTERISTICHE FISICHE	5
CARATTERISTICHE ELETTRICHE	5
INSTALLAZIONE	6
CONNESSIONI CON IL MONDO ESTERNO	6
CN2 - CONNETTORE PER LINEA CAN	6
CN1 - CONNETTORE PER ABACO® I/O BUS	8
INTERFACCIAMENTO DELLA SCHEDA	10
MONTAGGIO MECCANICO	10
JUMPERS	10
JUMPERS A 2 VIE	11
JUMPERS A 3 VIE	11
TERMINAZIONE LINEA CAN	11
INTERRUPT	12
MAPPAGGI ED INDIRIZZAMENTI	13
MAPPAGGIO DELLA SCHEDA	13
INDIRIZZAMENTO REGISTRI INTERNI	14
DESCRIZIONE SOFTWARE	15
CONTROLLORE CAN	15
SCHEDE ESTERNE	17
BIBLIOGRAFIA	20
APPENDICE A: DESCRIZIONE COMPONENTI DI BORDO	A-1
APPENDICE B: INDICE ANALITICO	B-1

INDICE DELLE FIGURE

FIGURA 1: SCHEMA A BLOCCHI	3
FIGURA 2: CN2 - CONNETTORE PER LINEA CAN	6
FIGURA 3: ESEMPIO COLLEGAMENTO IN RETE CON BUS CAN	7
FIGURA 4: CN1 - CONNETTORE PER ABACO® I/O BUS	8
FIGURA 5: DISPOSIZIONE CONNETTORI, JUMPER, DIP SWITCH, ECC.	9
FIGURA 6: TABELLA RIASSUNTIVA JUMPERS	10
FIGURA 7: TABELLA JUMPERS A 2 VIE	11
FIGURA 8: TABELLA JUMPERS A 3 VIE	11
FIGURA 9: FOTO DELLA SCHEDA	12
FIGURA 10: TABELLA INDIRIZZI DEI REGISTRI INTERNI	14
FIGURA 11: PIANTA COMPONENTI	14
FIGURA 12: FLOW CHART DI INIZIALIZZAZIONE	16
FIGURA 13: SCHEMA DELLE POSSIBILI CONNESSIONI	19

INTRODUZIONE

L'uso di questi dispositivi é rivolto - **IN VIA ESCLUSIVA** - a personale specializzato.

Scopo di questo manuale é la trasmissione delle informazioni necessarie all'uso competente e sicuro dei prodotti. Esse sono il frutto di un'elaborazione continua e sistematica di dati e prove tecniche registrate e validate dal Costruttore, in attuazione alle procedure interne di sicurezza e qualità dell'informazione.

I dati di seguito riportati sono destinati - **IN VIA ESCLUSIVA** - ad un utenza specializzata, in grado di interagire con i prodotti in condizioni di sicurezza per le persone, per la macchina e per l'ambiente, interpretando un'elementare diagnostica dei guasti e delle condizioni di funzionamento anomale e compiendo semplici operazioni di verifica funzionale, nel pieno rispetto delle norme di sicurezza e salute vigenti.

Le informazioni riguardanti installazione, montaggio, smontaggio, manutenzione, aggiustaggio, riparazione ed installazione di eventuali accessori, dispositivi ed attrezzature, sono destinate - e quindi eseguibili - sempre ed in via esclusiva da personale specializzato avvertito ed istruito, o direttamente dall'**ASSISTENZA TECNICA AUTORIZZATA**, nel pieno rispetto delle raccomandazioni trasmesse dal costruttore e delle norme di sicurezza e salute vigenti.

I dispositivi non possono essere utilizzati all'aperto. Si deve sempre provvedere ad inserire i moduli all'interno di un contenitore a norme di sicurezza che rispetti le vigenti normative. La protezione di questo contenitore non si deve limitare ai soli agenti atmosferici, bensì anche a quelli meccanici, elettrici, magnetici, ecc.

Per un corretto rapporto coi prodotti, é necessario garantire leggibilità e conservazione del manuale, anche per futuri riferimenti. In caso di deterioramento o più semplicemente per ragioni di approfondimento tecnico ed operativo, consultare direttamente l'Assistenza Tecnica autorizzata.

Al fine di non incontrare problemi nell'uso di tali dispositivi, é conveniente che l'utente - **PRIMA DI COMINCIARE AD OPERARE** - legga con attenzione tutte le informazioni contenute in questo manuale. In una seconda fase, per rintracciare più facilmente le informazioni necessarie, si può fare riferimento all'indice generale e all'indice analitico, posti rispettivamente all'inizio ed alla fine del manuale.

VERSIONE SCHEDA

Il presente manuale è riferito alla scheda **CAN 14** versione **110498** e successive. La validità delle informazioni riportate è quindi subordinata al numero di versione della scheda in uso e l'utente deve quindi sempre verificare la giusta corrispondenza tra le due indicazioni. Sulla scheda il numero di versione è riportato in più punti sia a livello di serigrafia che di stampato (ad esempio sull'angolo in alto a sinistra nel lato componenti).

INFORMAZIONI GENERALI

Il modulo **CAN 14** é una potente scheda periferica in grado di gestire la comunicazione seriale secondo lo standard industriale Control Area Network. Quest'ultimo é un protocollo standard di comunicazione caratterizzato da: alte velocità di trasferimento dati, comunicazione anche su lunghe distanze, gestione autonoma degli eventuali errori di comunicazione con ritrasmissione automatica; riconoscimento autonomo degli slave in una rete di più sistemi, collegamenti in modalità multimaster e multislave, gestione di maschere di ricezione e trasmissione, disponibilità di una vasta serie di potenti comandi, ecc. In particolare la **CAN 14** supporta completamente gli standard **BasicCAN** e/o **PeliCAN 2.0B** a cui l'utente può fare riferimento per ogni chiarimento necessario.

Il collegamento elettrico del modulo avviene tramite una coppia di comodi connettori di cui uno per il collegamento alle schede di controllo tramite l'**Abaco**® I/O BUS e l'altro per il collegamento alla linea CAN, mentre il montaggio meccanico é facilitato dall'apposito supporto plastico provvisto degli attacchi per le guide ad **Omega** tipo **DIN 46277-1** e **DIN 46277-3**. La linea seriale CAN di bordo é galvanicamente isolata dal resto della scheda, in modo da garantire l'immunità dagli eventuali disturbi del campo su tutta l'elettronica di controllo.

Le applicazioni tipiche in cui il **CAN 14** può essere utilizzato sono tutte quelle in cui si devono trasferire dati tra due o più sistemi intelligenti senza doversi preoccupare di tutte le condizioni caratteristiche della comunicazione seriale, oppure per attraversare ambienti rumorosi e/o distanti, per gestire sistemi multimaster o quando più semplicemente, ci si deve interfacciare ad un altro sistema che utilizza lo standard CAN.

Una ricca serie di programmi dimostrativi ed esempi di utilizzo, consentono un immediato uso della scheda. Detti programmi sono disponibili per numerose schede di CPU presenti nel vasto carteggio **Abaco**®. Gli esempi sono ampiamente commentati e sono forniti sotto forma di sorgenti nei vari linguaggi in cui é possibile programmare le schede del carteggio **Abaco**®.

- Interfaccia per **Abaco**® I/O BUS su connettore a scatolino a **26 vie**.
- Ingombro di **100x50x20** mm, (110x60x70 mm con contenitore) nel formato **serie 4**.
- Contenitore per guide ad Ω tipo **DIN 46277-1** e **DIN 46277-3**.
- Connettore **vaschetta D** a **9 vie** per linea di comunicazione CAN.
- Supporto protocollo **BasicCAN** gestito dal PHILIPS **PCx82C200** a 16M Hz, oppure protocollo **PeliCAN 2.0B** gestito solo dal PHILIPS **SJA 1000** a 24M Hz.
- Driver di linea PHILIPS **82C250** galvanicamente isolato.
- **DC/DC** converter per optoisolamento della linea di comunicazione CAN.
- **Baud rate** supportati: fino a **400 KBit/sec** per linee da **centinaia** di metri, **1 MBit/sec** per linee di **decine** di metri.
- **Spazio d'indirizzamento** occupato di soli **2 bytes** consecutivi.
- **Dip switch** per settare il mappaggio in I/O della scheda.
- Generazione di **interrupts** in corrispondenza di eventi definibili via software.
- Collegamento interrupt ad **/INT** o **/NMI** selezionabile via hardware.
- Unica tensione di alimentazione: **+5 Vdc; 93 mA**.

Viene di seguito riportata una descrizione dei blocchi funzionali della scheda, con indicate le operazioni effettuate da ciascuno di essi. Per una più facile individuazione di tali blocchi e per una verifica delle loro connessioni, fare riferimento alla figura 1.

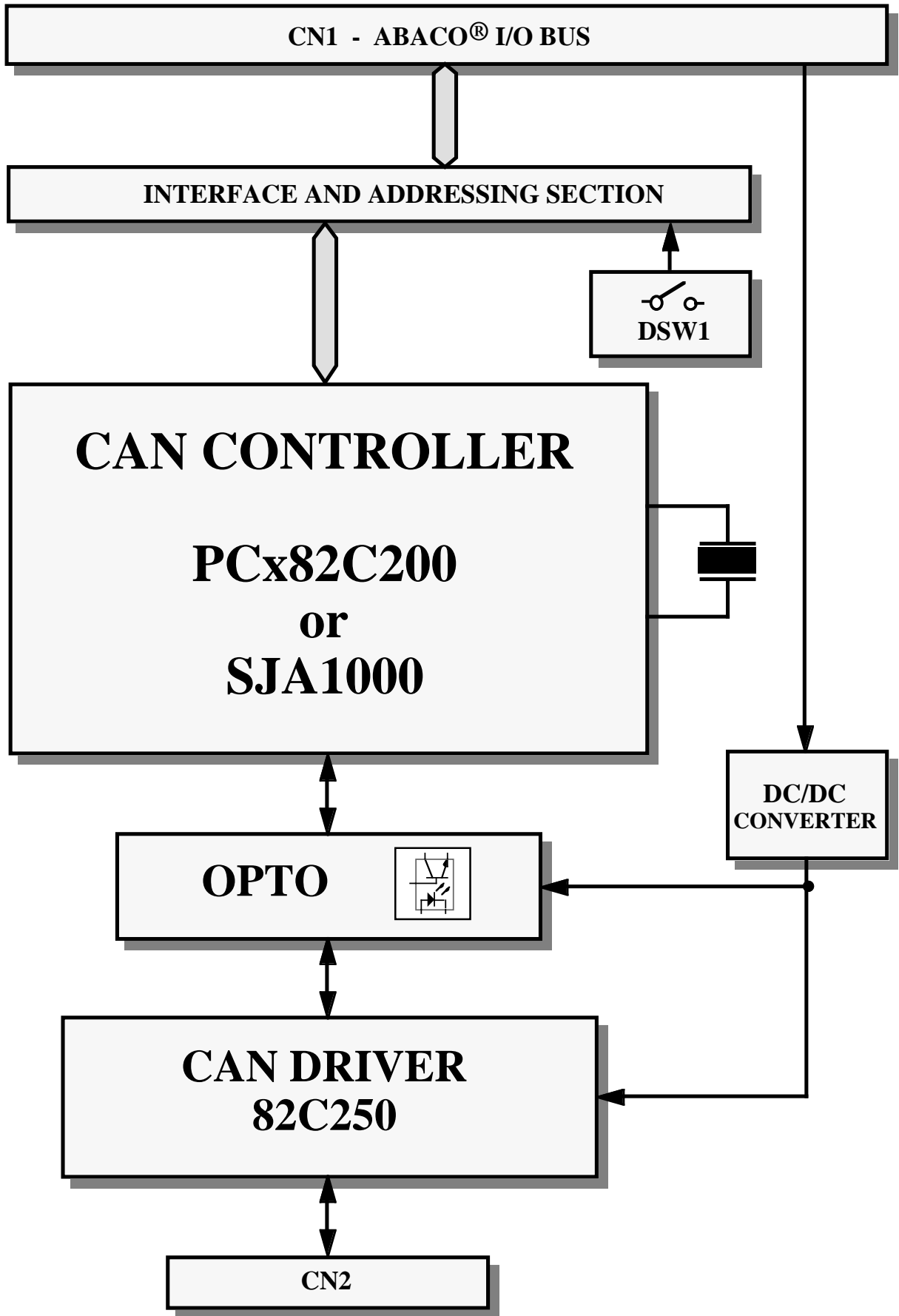


FIGURA 1: SCHEMA A BLOCCHI

SEZIONE DI INTERFACCIA ED INDIRIZZAMENTO

Questa sezione gestisce il colloquio tra il controllore CAN e la scheda di comando di tipo **GPC®**; in particolare tutti i dati di programmazione, gestione e comunicazione passano attraverso questa sezione che inoltre provvede a gestire il mappaggio della scheda in I/O, tramite un comodo dip switch, ed a configurare l'eventuale gestione di interrupt. Il collegamento fisico con le schede di comando é effettuato tramite il comodo **ABACO® I/O BUS** ad 8 bit, ma può essere esteso anche al BUS industriale **ABACO®** sfruttando appositi moduli di conversioni come l'**ABB 05** o l'**ABB 03**. La sezione di interfaccia ed indirizzamento é basata su una logica programmabile ed alcuni componenti di contorno, che garantiscono il funzionamento in ogni condizione operativa ed allo stesso tempo riducono al minimo l'ingombro.

CONTROLLORE CAN

Questa sezione é basata su uno dei controllori **PCx82C200** o **SJA 1000** della PHILIPS e si preoccupa della gestione software del protocollo CAN in tutte le sue modalità ed aspetti. Le caratteristiche fondamentali di questa sezione sono:

	PCx82C200	SJA 1000
- supporto protocollo BasicCAN:	SI	SI
- supporto del protocollo PeliCAN 2.0B:	NO	SI
- gestione identificatori da:	11 bits	11 e 29 bits
- buffer di trasmissione da:	10 bytes	13 bytes
- buffer di ricezione da:	10 bytes	64 bytes
- baud rate programmabile fino a:	1M Bit/sec	1M Bit/sec
- eliminazione del comparatore di ricezione:	NO	SI
- filtri di accettazione messaggi configurabili:	SI	SI
- driver di uscita programmabile:	SI	SI
- frequenza massima di lavoro:	16M Hz	24M Hz

Il controllore **SJA1000** é completamente compatibile con il **PCx82C200**, di cui costituisce un aggiornamento. L'utente può quindi passare a questo nuovo controllore salvaguardando il firmware od il software già sviluppato.

Per ogni chiarimento necessario l'utente può fare riferimento all'apposita documentazione della casa costruttrice, oppure all'appendice A di questo manuale.

INTERFACCIA DI LINEA CAN

Dal punto di vista elettrico la scheda **CAN 14** é dotata dell'apposito driver di linea **82C250** della PHILIPS, galvanicamente isolato. Questo componente si preoccupa di soddisfare tutte le specifiche di collegamento con il campo, definite nel protocollo CAN senza richiedere alcun intervento software. Inoltre la linea seriale CAN di bordo é galvanicamente isolata dal resto della scheda, in modo da garantire l'immunità agli eventuali disturbi del campo; questa caratteristica é di fondamentale importanza soprattutto nel caso di collegamento con sistemi remoti a diversi potenziali oppure di collegamenti con cavi che attraversano ambienti elettricamente rumorosi.

Un apposito DC/DC converter si preoccupa di generare le tensioni galvanicamente isolate richieste dal driver di linea, mentre l'interfacciamento con le linee di comunicazione del controllore CAN sono effettuati tramite appositi optoisolatori per alte frequenze.

Il collegamento con il campo della linea CAN é effettuato tramite un connettore a vaschetta D a 9 vie che facilita il cablaggio e garantisce una buona trasmissione del segnale.

SPECIFICHE TECNICHE

CARATTERISTICHE GENERALI

Tipo di BUS:	ABACO® I/O BUS
Numero linee di I/O:	1 linea seriale CAN
Numero byte di indirizzamento:	256
Numero byte occupati:	2
Periferiche di bordo:	CAN controller PCx82C200 CAN controller SJA1000
Quarzo:	16M Hz (PCx82C200) 24M Hz (SJA1000)
Baud Rate massimo:	1M Bit/sec
Protocolli gestiti:	BasicCAN (PCx82C200 e SJA1000) PeliCAN 2.0B (solo SJA1000)
Gestione interrupt:	Selezionabile tra /INT e /NMI

CARATTERISTICHE FISICHE

Dimensioni (L x A x P):	100 x 50 x 20 mm	(senza contenitore)
	110 x 60 x 70 mm	(con contenitore per guide DIN)
Peso:	50 g	(senza contenitore)
	110 g	(con contenitore per guide DIN)
Connettori:	CN1: 26 vie scatola verticale M	
	CN2: 9 vie vaschetta D femmina	
Range di temperatura:	da 0 a 50 gradi Centigradi	
Umidità relativa:	20% fino a 90%	(senza condensa)

CARATTERISTICHE ELETTRICHE

Tensione di alimentazione:	5 Vdc
Corrente assorbita:	93 mA
Impedenza di linea CAN:	60 Ω
Rete terminazione CAN:	Resistenza da 120 Ω , disinseribile

INSTALLAZIONE

In questo capitolo saranno illustrate tutte le operazioni da effettuare per il corretto utilizzo della scheda. A questo scopo viene riportata l'ubicazione e la funzione dei jumpers, dei connettori, dei dip switch, ecc. presenti sulla **CAN 14**.

CONNESSIONI CON IL MONDO ESTERNO

Il modulo **CAN 14** è provvisto di 2 connettori con cui vengono effettuati tutti i collegamenti con il campo e con le altre schede del sistema di controllo da realizzare. Di seguito viene riportato il loro pin out ed il significato dei segnali collegati; per una facile individuazione di tali connettori, si faccia riferimento alla figura 5.

CN2 - CONNETTORE PER LINEA CAN

CN2 è un connettore a vaschetta D a 9 vie femmina a 90°. Tramite CN2 deve essere collegata la linea di comunicazione seriale CAN secondo gli standard definiti dallo stesso protocollo; la disposizione dei segnali è stata studiata in modo da ridurre al minimo le interferenze ed in modo da facilitare la connessione con il campo.

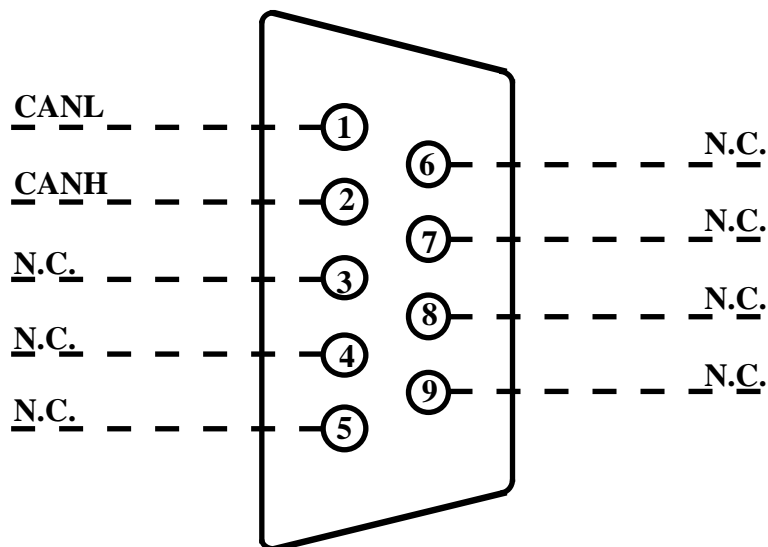


FIGURA 2: CN2 - CONNETTORE PER LINEA CAN

Legenda:

CANH	=	I/O	-	Linea differenziale high per CAN BUS.
CANL	=	I/O	-	Linea differenziale low per CAN BUS.
N.C.	=		-	Non connesso.

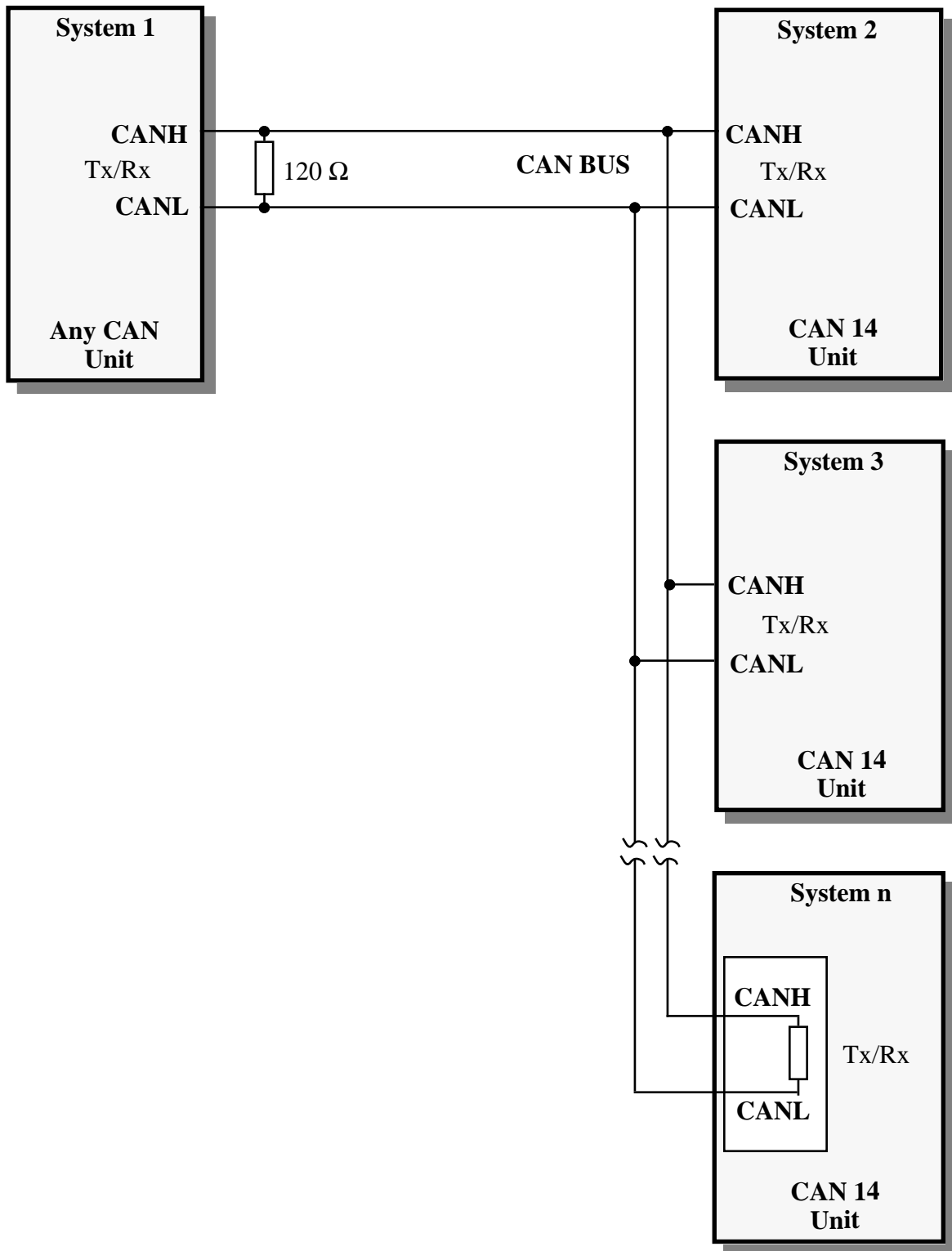
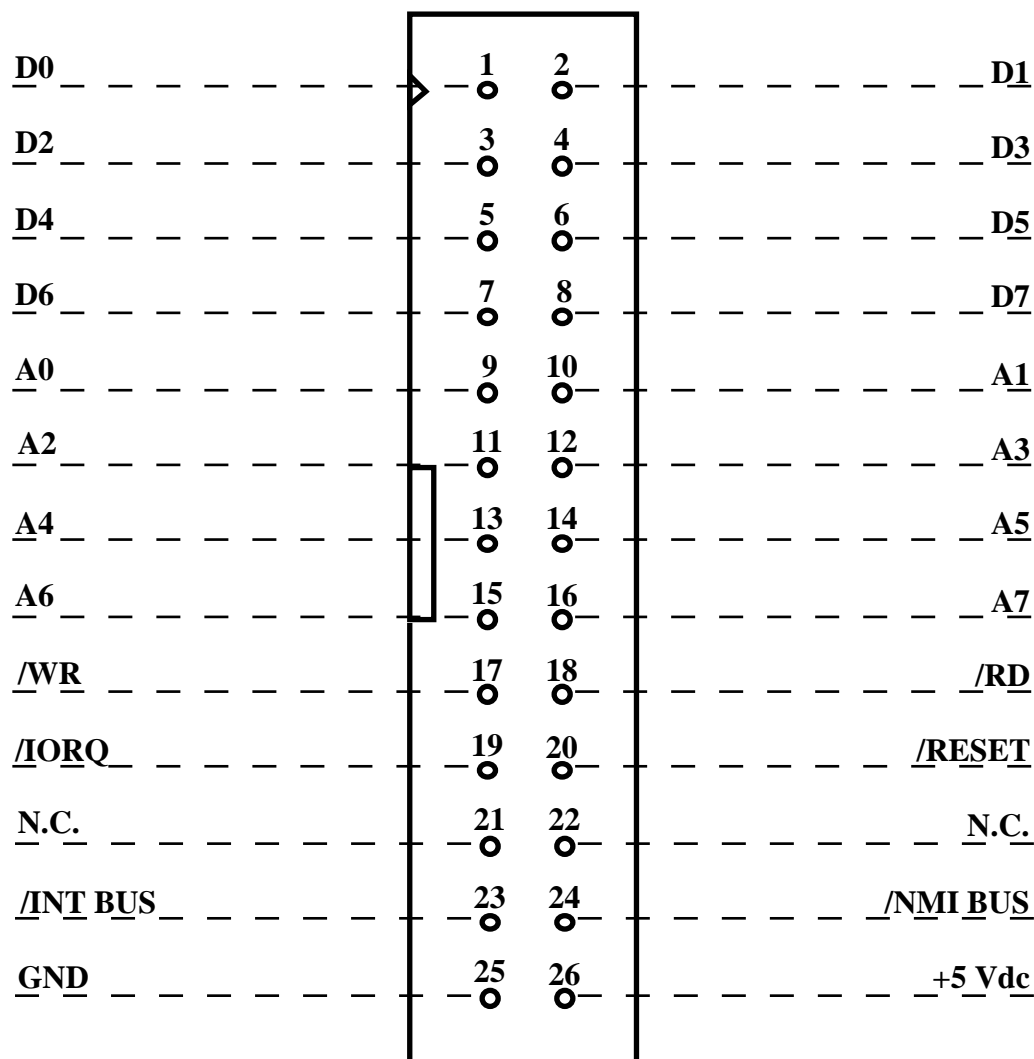


FIGURA 3: ESEMPIO COLLEGAMENTO IN RETE CON BUS CAN

CN1 - CONNETTORE PER ABACO® I/O BUS

CN1 è un connettore a scatolino verticale con passo 2.54 mm a 26 piedini. Tramite CN1 si effettua la connessione tra la scheda e la serie di moduli di controllo della serie **GPC®**. Tale collegamento è effettuato tramite l'**ABACO®** I/O BUS di cui questo connettore riporta tutti i segnali a livello TTL.


FIGURA 4: CN1 - CONNETTORE PER ABACO® I/O BUS

Legenda:

A0-A7	=	I	- Address BUS: BUS degli indirizzi.
D0-D7	=	I/O	- Data BUS: BUS dei dati.
/INT BUS	=	O	- Interrupt request: richiesta d'interrupt.
/NMI BUS	=	O	- Non Mascable Interrupt: richiesta d'interrupt non mascherabile.
/IORQ	=	I	- Input Output Request: richiesta operazione Input Output su I/O BUS.
/RD	=	I	- Read cycle status: richiesta di lettura.
/WR	=	I	- Write cycle status: richiesta di scrittura.
/RESET	=	I	- Reset: azzeramento.
+5 Vdc	=	I	- Linea di alimentazione a +5 Vcc.
GND	=		- Linea di massa.
N.C.	=		- Non collegato.

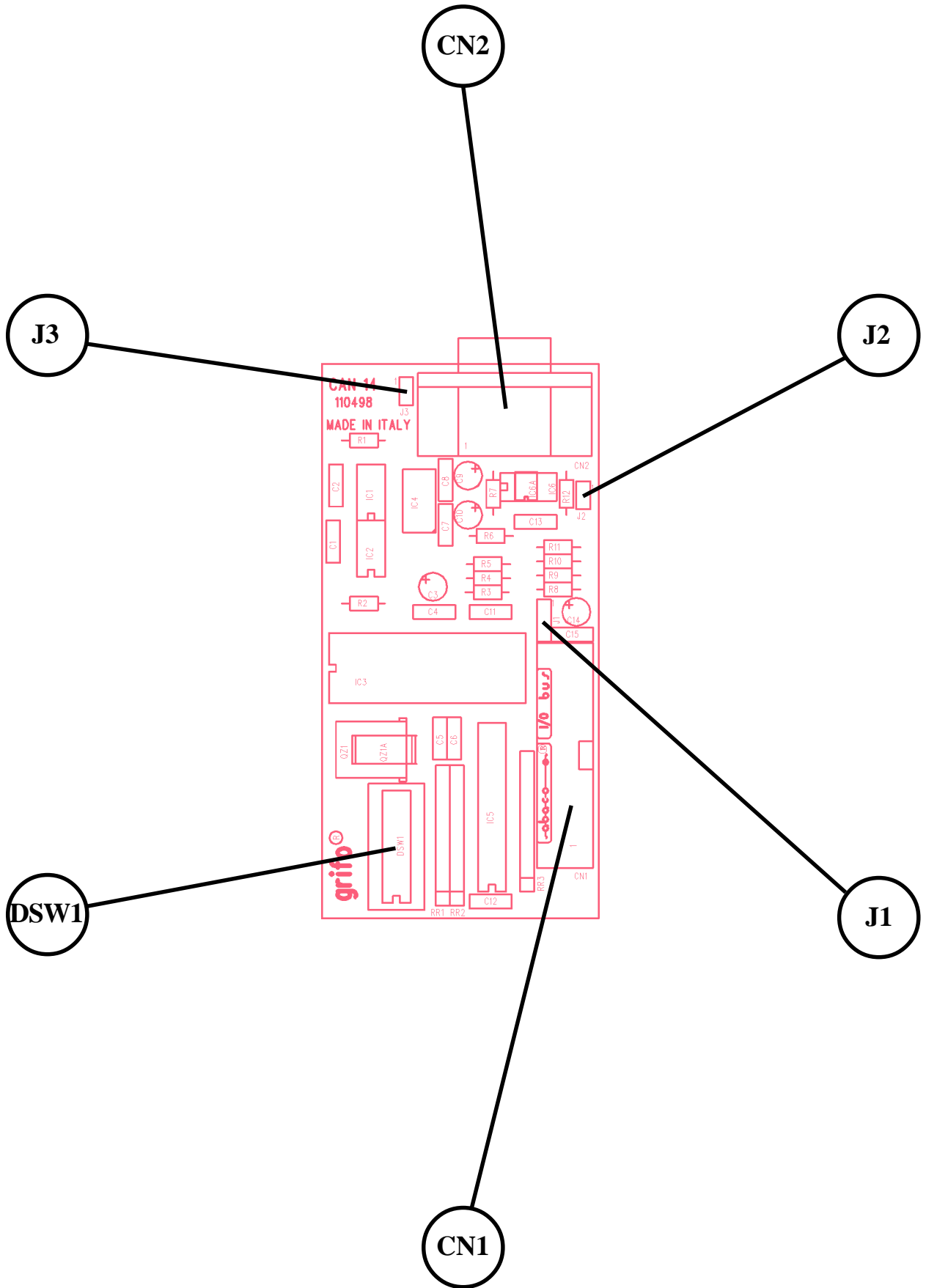


FIGURA 5: DISPOSIZIONE CONNETTORI, JUMPER, DIP SWITCH, ECC.

INTERFACCIAMENTO DELLA SCHEDA

Al fine di evitare eventuali problemi di collegamento della scheda con tutta l'elettronica del campo e di controllo a cui la **CAN 14** si deve interfacciare, si devono seguire le informazioni riportate nei precedenti paragrafi e le relative figure che illustrano le modalità interne di connessione.

- Per i segnali che riguardano la comunicazione seriale con il protocollo CAN BUS, fare riferimento alle specifiche standard di questo protocollo. Al fine di mantenere l'isolamento galvanico tra la linea di comunicazione CAN e l'elettronica di controllo è conveniente che la massa di alimentazione della **CAN 14** (pin 25 di CN1) non sia direttamente o indirettamente collegata ad altri sistemi presenti sul BUS CAN.
- Tutti i segnali a livello TTL possono essere collegati a linee dello stesso tipo riferite alla massa digitale della scheda. Il livello 0V corrisponde allo stato logico 0, mentre il livello 5V corrisponde allo stato logico 1. Nel caso di segnali ad alta frequenza come quelli dell'**ABACO**[®] I/O BUS è consigliabile non superare la lunghezza di una decina di centimetri.

MONTAGGIO MECCANICO

La scheda **CAN 14** normalmente è fornita senza alcun contenitore, ma a livello di opzioni esistono alcuni articoli che facilitano notevolmente il suo montaggio meccanico. Queste soluzioni coincidono con dei contenitori plastici predisposti per il montaggio su guide ad Ω tipo **DIN 46277-1** e **DIN 46277-3**. In caso di accoppiamento della **CAN 14** ad altre schede con **ABACO**[®] I/O BUS è possibile ordinare un unico contenitore che contenga tutte le schede in modo da semplificare il montaggio e da ridurre i costi. Per maggiori informazioni o per sigle dei contenitori da ordinare, rivolgersi direttamente alla **grifo**[®].

JUMPERS

Esistono a bordo della **CAN 14** 3 jumpers a cavaliere, con cui è possibile effettuare alcune selezioni che riguardano il modo di funzionamento della stessa. Di seguito ne è riportato l'elenco, l'ubicazione e la loro funzione nelle varie modalità di connessione.

NOME	N. VIE	UTILIZZO
J1	3	Selezione collegamento segnale d'interrupt.
J2	2	Selezione collegamento della resistenza di terminazione.
J3	2	Selezione collegamento della corazza del connettore CN2.

FIGURA 6: TABELLA RIASSUNTIVA JUMPERS

Nelle successive tabelle è riportata una descrizione dettagliata delle possibili connessioni dei 2 jumpers con la loro relativa funzione. Per riconoscere tali connessioni sulla scheda si faccia riferimento alla serigrafia della stessa o alla figura 10 di questo manuale, dove viene riportata la numerazione dei pin dei jumpers, che coincide con quella utilizzata nella seguente descrizione. Per l'individuazione dei jumpers a bordo della scheda, si utilizzi invece la figura 5. In tutte le seguenti tabelle l'* indica la connessione di default, ovvero quella impostata in fase di collaudo, con cui la scheda viene fornita.

JUMPERS A 2 VIE

JUMPERS	CONNESSIONE	UTILIZZO	DEF.
J2	non connesso	Non collega la resistenza di terminazione da 120 Ω alla linea CAN.	*
	connesso	Collega la resistenza di terminazione da 120 Ω alla linea CAN.	
J3	non connesso	Non collega la carcassa del connettore CN2 alla massa galvanicamente isolata dell'interfaccia di linea.	*
	connesso	Collega la carcassa del connettore CN2 alla massa galvanicamente isolata dell'interfaccia di linea.	

FIGURA 7: TABELLA JUMPERS A 2 VIE

JUMPERS A 3 VIE

JUMPERS	CONNESSIONE	UTILIZZO	DEF.
J1	posizione 1-2	Collega interrupt del controllore CAN al segnale /INT BUS dell'ABACO® I/O BUS.	*
	posizione 2-3	Collega interrupt del controllore CAN al segnale /NMI BUS dell'ABACO® I/O BUS.	
	non connesso	Interrupt del controllore CAN non collegato.	

FIGURA 8: TABELLA JUMPERS A 3 VIE

TERMINAZIONE LINEA CAN

Il jumper J2 ha il compito di collegare o meno l'apposita resistenza di terminazione della linea CAN come descritto nella tabella di figura 7. Il CAN BUS deve fisicamente coincidere con una linea differenziale con impedenza di 60 Ω e per questo le resistenze di terminazione devono essere collegate in modo da ricreare questa impedenza. In particolare tale collegamento deve essere sempre effettuato in caso di sistemi punto punto, mentre nel caso di sistemi multipunto, deve essere collegata solo sulle schede che risultano essere alla maggior distanza, ovvero ai capi della linea di comunicazione CAN (vedere esempio di figura3).

La corretta terminazione della linea CAN contribuisce notevolmente al funzionamento della comunicazione, infatti l'interfaccia di linea della **CAN 14** é in grado di sopprimere i transienti e di essere immune ai disturbi di radio frequenza ed elettromagnetici, solo se il collegamento con il campo é effettuato correttamente.

INTERRUPT

Il modulo **CAN 14** é provvisto di una comoda ed efficace circuiteria di generazione interrupt, che provvede se collegata, a richiedere l'attenzione della scheda di controllo **GPC®** in corrispondenza di stati predeterminati. Tale circuiteria tende ad ottimizzare i tempi di gestione della scheda, infatti tramite la generazione d'interrupt, la scheda di controllo é liberata dal compito di testare continuamente lo stato della **CAN 14**; in questo modo é la stessa scheda che, quando pronta, lo segnala alla scheda di controllo che provvederà quindi all'interscambio di nuovi dati.

Il segnale d'interrupt della scheda può essere collegato in tre diverse modalità tramite il jumper J1, come descritto nella tabella di figura 8; il segnale é in open collector e può essere quindi collegato ad uno dei due segnali d'interrupt dell'**ABACO®** I/O BUS, anche se questi sono già utilizzati da altre periferiche.

Per ulteriori informazioni sulle modalità di gestione interrupt, si faccia riferimento all'appendice A.



FIGURA 9: FOTO DELLA SCHEDA

MAPPAGGI ED INDIRIZZAMENTI

In questo capitolo ci occuperemo di fornire tutte le informazioni relative all'utilizzo della scheda, dal punto di vista della programmazione via software. Tra queste si trovano le informazioni riguardanti il mappaggio della scheda e l'indirizzamento delle sezioni componenti.

MAPPAGGIO DELLA SCHEDA

La **CAN 14** occupa un'indirizzamento di soli 2 bytes consecutivi che possono essere allocati a partire da un indirizzo di base diverso a seconda di come viene mappata la scheda. Questa prerogativa consente di poter utilizzare più **CAN 14** sullo stesso **ABACO®** I/O BUS o **BUS ABACO®**, oppure di montare la scheda su di un BUS su cui sono già presenti altre schede periferiche, ottenendo così una struttura espandibile senza difficoltà e senza alcuna modifica del software già realizzato.

I 2 bytes occupati sono utilizzati sia in fase di Output che di Input, quindi permettono sia la programmazione della scheda che la lettura del suo stato oltre alle normali operazioni di ricezione e trasmissione.

L'indirizzo di mappaggio della **CAN 14** é definibile tramite l'apposita circuiteria d'indirizzamento ed interfaccia al BUS, presente sulla scheda; questa circuiteria utilizza il dip switch ad 8 vie DSW1, da cui preleva lo stesso indirizzo di mappaggio impostato dall'utente. Di seguito viene riportata la corrispondenza del dip switch con le linee d'indirizzamento dell'**ABACO®** I/O BUS, mentre per una più facile individuazione di tale componente si faccia riferimento alla figura 5.

DSW1.1	->	Non usato
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

Tali dip switch sono collegati con logica negata, quindi se posti in **ON** generano uno **zero logico**, mentre se posti in **OFF** generano un **uno logico**.

A titolo di esempio, viene riportata di seguito la configurazione del DSW1, necessaria per mappare la scheda all'indirizzo 48H:

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

INDIRIZZAMENTO REGISTRI INTERNI

Indicando con <indbase> l'indirizzo di mappaggio della scheda, ovvero l'indirizzo impostato tramite il DSW1 come descritto nel paragrafo precedente, i registri interni della **CAN 14** sono visti agli indirizzi riportati nella seguente tabella.

DISP.	REG.	IND.	R/W	SIGNIFICATO
PCx82C200	ADDR	<indbase>+00H	W	Registro di settaggio indirizzo registro
SJA1000	DATA	<indbase>+01H	R/W	Registro di interscambio dati

FIGURA 10: TABELLA INDIRIZZI DEI REGISTRI INTERNI

Se si utilizzano più schede sull'**ABACO®** I/O BUS o **BUS ABACO®**, in fase di impostazione dell'indirizzo di mappaggio delle schede, si deve fare attenzione a non allocare più schede agli stessi indirizzi (considerare per questo indirizzo di mappaggio e numero di byte occupati). Nel caso questa condizione non venga rispettata, si viene a creare una conflittualità sul BUS che pregiudica il funzionamento di tutto il sistema e delle stesse schede.

Si ricorda che la precedente tabella riporta la descrizione dei soli registri disponibili a livello della scheda **CAN 14** e che per una descrizione dettagliata di tutti i registri interni dei controllori CAN e delle relative modalità d'accesso si può fare riferimento al capitolo successivo ed all'appendice A.

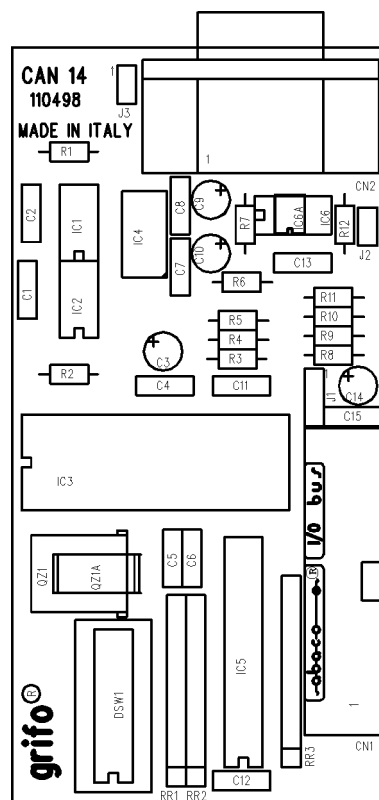


FIGURA 11: PIANTA COMPONENTI

DESCRIZIONE SOFTWARE

Nel paragrafo precedente sono stati riportati gli indirizzi di allocazione di tutte le periferiche e di seguito viene riportata una descrizione dettagliata della funzione e del significato dei relativi registri (al fine di comprendere le successive informazioni, fare sempre riferimento alla tabella indirizzi dei registri interni). Qualora la documentazione riportata fosse insufficiente fare riferimento direttamente alla documentazione tecnica della casa costruttrice del componente o all'appendice A di questo manuale.

CONTROLLORE CAN

Per accedere ai vari registri del controlllore CAN, **PCx82C200** o **SJA1000**, é necessario effettuare le seguenti operazioni:

- 1) Scrivere all'indirizzo ADDR in numero del registro che si desidera gestire.
- 2) Leggere o scrivere all'indirizzo DATA per acquisire o settare il registro selezionato con la precedente operazione.

Nell'appendice A é riportato l'elenco completo dei registri del controlllore CAN e la descrizione di tutti i bit che li compongono.

Esempio:

Facendi riferimento all'appendice A, se si deve scrivere il dato **170** nell'**Acceptance code register** (registro n. 4), sarà necessario effettuare le seguenti operazioni:

- 1) Scrivere all'indirizzo ADDR il dato 4.
- 2) Scrivere all'indirizzo DATA il dato 170.

Note:

- 1) Il Baud Rate di comunicazione, ricavabile dalle informazioni dell'appendice A, é ottenibile dalla seguente formula:

$$\text{BAUD RATE} = \text{Freq} / 2 * (\text{BRP} + 1) * (3 + \text{TSEG1} + \text{TSEG2})$$

dove:

Freq = Frequenza del quarzo della scheda **CAN 14** in Hz.

BRP = Valore espresso dai bit **BRP.x** del **Bus Timing Register 0** (BTR0, indirizzo 6).

TSEG1 = Valore espresso dai bit **TSEG1.x** del **Bus Timing Register 1** (BTR1, indirizzo 7).

TSEG2 = Valore espresso dai bit **TSEG2.x** del **Bus Timing Register 1** (BTR1, indirizzo 7).

- 2) Per un corretto interfacciamento fra il controlllore CAN, **PCx82C200** o **SJA1000**, ed il driver **82C250**, é necessario programmare l'**Output control register** (OCR, indirizzo 8), con il dato **FA Hex**; in questo modo si configura il dispositivo in "Normal output mode", con le uscite TX0 e TX1 in "Push-Pull".

Nelle pagina seguente viene riportata la flow chart relativa all'inizializzazione del controlllore CAN, **PCx82C200** o **SJA1000**.

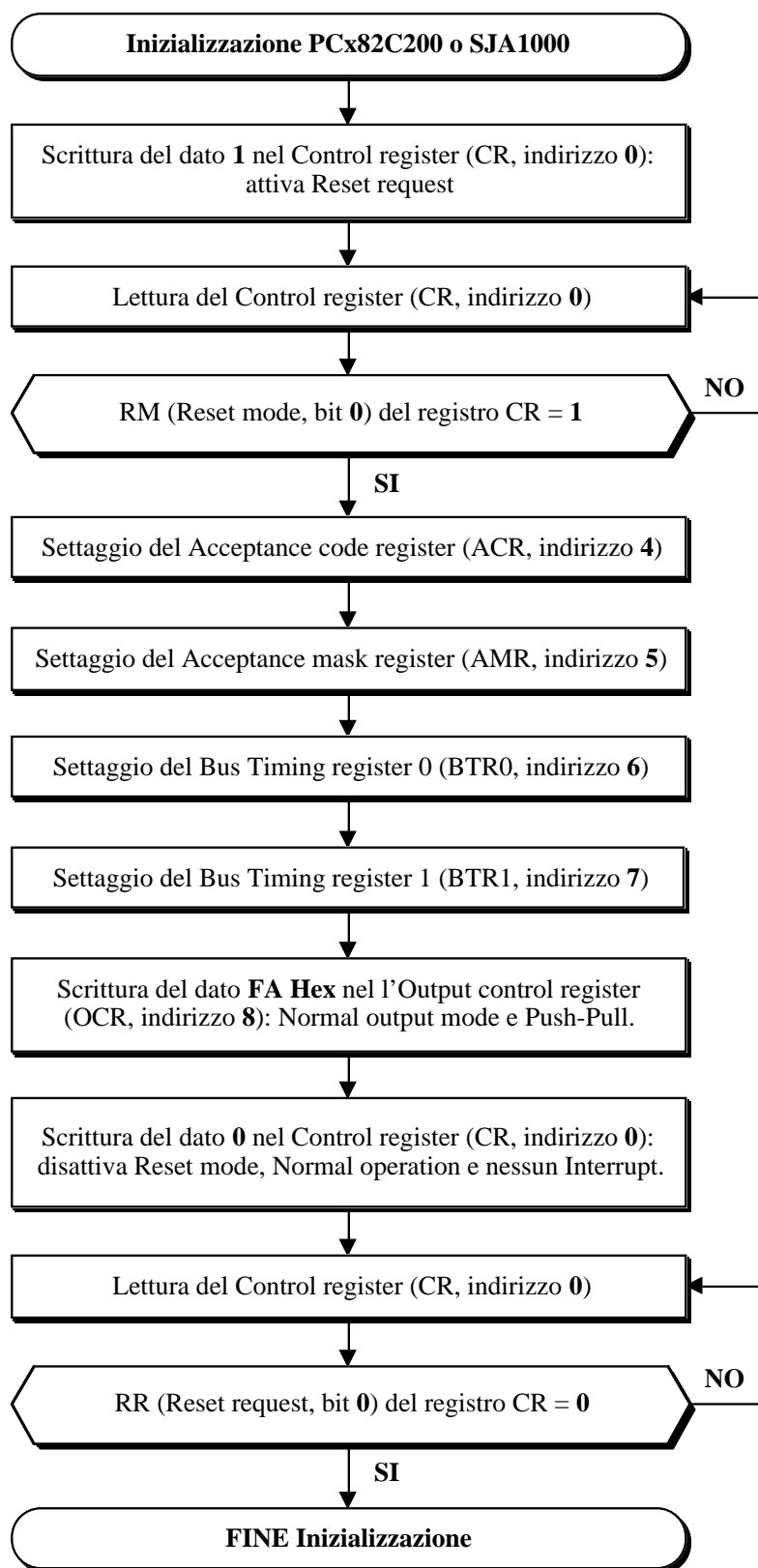


FIGURA 12: FLOW CHART DI INIZIALIZZAZIONE

Come si può notare, questa inizializzazione non prevede l'uso di Interrupt; per una loro eventuale gestione, è necessario settare opportunamente i bit del Control Register (CR, indirizzo 0). A questo proposito si faccia riferimento all'appendice A ed ai programmi di esempio forniti insieme alla scheda CAN 14.

SCHEDE ESTERNE

Le applicazioni caratteristiche della **CAN 14** sono il trasferimento dati ad alta velocità tra sistemi remotati usati nel settore automobilistico e del controllo di processo, l'attraversamento di ambienti rumorosi e/o distanti, la gestione di sistemi multimaster o quando più semplicemente, ci si deve interfacciare ad un altro sistema che utilizza lo standard CAN.

La scheda **CAN 14** si interfaccia a buona parte dei moduli della serie BLOCK e di interfaccia utente. Le risorse di bordo possono essere facilmente aumentate collegando la **CAN 14** alle numerose schede periferiche del carteggio **grifo®** tramite l'**ABACO®** I/O BUS. Anche schede in formato Europa con **BUS ABACO®** possono essere collegate, sfruttando gli appositi mother boards. A titolo di esempio ne riportiamo un elenco con una breve descrizione delle caratteristiche di massima; per maggiori informazioni richiedere la documentazione specifica:

QTP G26

Quick Terminal Panel 26 tasti con LCD grafico

Interfaccia operatore provvista di display grafico da 240x128 pixel retroilluminato a catodo freddo; tastiera a membrana da 26 tasti di cui 6 configurabili dall'utente; 16 LEDs di stato; alimentatore a bordo scheda; interfaccia seriale in RS 232, RS 422-485, current loop o CAN; linea seriale ausiliaria in RS 232. Tasti ed etichette personalizzabili dall'utente tramite serigrafie da inserire in apposite tasche; contenitore metallico e plastico; EEPROM di set up; 256K EPROM o FLASH; Real Time Clock; 128K RAM; buzzer. Firmware di gestione che svolge funzione di terminale con primitive grafiche.

FBC D9 M/F

Flat Block Contact vaschetta D 9 vie Maschio/Femmina

Interfaccia tra 1 connettore a vaschetta D a 9 vie (maschio o femmina) e la filatura da campo (morsettiere a rapida estrazione). Attacco rapido per guide tipo DIN 46277-1 e 3.

ABB 05

Abaco® Block BUS 5 slots

Mother board **ABACO®** da 5 slots; passo 4 TE; guidaschede; connettori normalizzati di alimentazione; tasto di reset; LEDs per alimentazioni; interfaccia **ABACO®** I/O BUS; sezione alimentatrice per +5 Vdc; sezione alimentatrice per +V Opto; sezioni alimentatrici galvanicamente isolate; tre tipi di alimentazione: da rete, bassa tensione o stabilizzata. Attacco rapido per guide Ω .

ABB 03

Abaco® Block BUS 3 slots

Mother board **ABACO®** da 3 slots; passo 4 TE; guidaschede; connettori normalizzati di alimentazione; tasto di reset; LEDs per alimentazioni; interfaccia **ABACO®** I/O BUS. Attacco rapido per guide Ω .

GPC® 51

General Purpose Controller fam. 51

Microprocessore famiglia 51 INTEL compreso il tipo mascherato BASIC; comprende: 16 linee di I/O TTL; dip switch; 3 Timer Counter; linea RS 232; 4 linee di A/D da 11 bit; buzzer; EPROM programmer a bordo; RTC e 32K RAM con back up al litio; controllore display e tastiera.

GPC® 553

General Purpose Controller 80C552

Microprocessore 80C552 a 22 o 30 MHz. Completa implementazione CMOS; 32K EPROM; 32 K RAM; 32 K EEPROM o RAM; RTC; EEPROM; 1 linea RS 232 + 1 RS 232 o RS 422-485 o current loop; 16 I/O TTL; 2 linee di PWM; timer/counter da 16 bits; watch dog; dip switch; 8 linee di A/D da 12 bit; alimentazione in DC o AC; attacco rapido per guide DIN 46277-1 e 3.

GPC® 188F

General Purpose Controller 80C188

Microprocessore 80C188 INTEL. 1 linea RS 232 ed 1 RS 232, 422-485 o current loop; 24 linee di I/O TTL; 256K EPROM e 256K RAM tamponate con batteria al litio; RTC; 3 timer counter; 8 linee di A/D da 12 bit; watch dog; write protect; EEPROM; 2 LEDs di attività; dip switch.

GPC® 15A

General Purpose Controller 84C15

Microprocessore Z80 a 10 MHz. Completa implementazione CMOS; 512K EPROM o 256K FLASH; RAM tamponata+RTC da 2K o 8KRTC ; 128K RAM; 1 linea RS 232 + 1 RS 232 o RS 422-485 o current loop; 32 I/O TTL; 4 counter; 2 watch dog; dip switch; buzzer; EEPROM.

GPC® 15R

General Purpose Controller 84C15

Microprocessore Z80 a 10 MHz. Completa CMOS. 512K EPROM o FLASH; RAM tamponata+RTC da 2K o 8KRTC ; 512K RAM tamponata da batteria esterna; EEPROM; 1 linea RS 232 + 1 RS 232 o RS 422-485 o current loop; 24 I/O TTL; 4 counter; 2 watch dog; dip switch; buzzer; 8 output a relé 3A; 16 input optoisolati NPN; alimentatore di bordo anche per I/O, galvanicamente isolato; power failure; alimentazione da rete 220 Vac; attacco rapido per guide DIN 46277-1 e 3.

GPC® 153

General Purpose Controller 84C15

Microprocessore Z80 a 10 MHz. Completa implementazione CMOS. 512K EPROM o 256K FLASH; RAM tamponata+RTC da 2K o 8KRTC ; 128K RAM; Back-Up con batteria al litio esterna; 1 linea RS 232 + 1 RS 232 o RS 422-485 o current loop; 16 I/O TTL; 4 counter; 2 watch dog; dip switch; buzzer; EEPROM; 8 linee di A/D da 12 bit; alimentazione in DC o AC; attacco rapido per guide DIN 46277-1 e 3.

GPC® 884

General Purpose Controller 80C188ES

Microprocessore AMD 80C188ES fino a 40M Hz. Completa implementazione CMOS; formato serie 4; 512K EPROM o FLASH; 512K RAM tamponata con batteria al litio; RTC; 1 linea RS 232 + 1 RS 232 o RS 422-485 o current loop; 16 I/O TTL; 3 timer counter; watch dog; EEPROM; 11 linee di A/D da 12 bit; attacco rapido per guide DIN 46277-1 e 3.

GPC® 114

General Purpose Controller 68HC11

Microprocessore 68HC11A1 a 8M Hz. Completa implementazione CMOS; formato serie 4; 32K EPROM; 32K RAM tamponata con batteria al litio; 32K EPROM, RAM, EEPROM; RTC; 1 linea RS 232 o RS 422-485; 10 I/O TTL; 3 timer counter; watch dog; 8 linee di A/D da 8 bit; 1 linea seriale sincrona; bassissimo assorbimento; attacco rapido per guide DIN 46277-1 e 3.

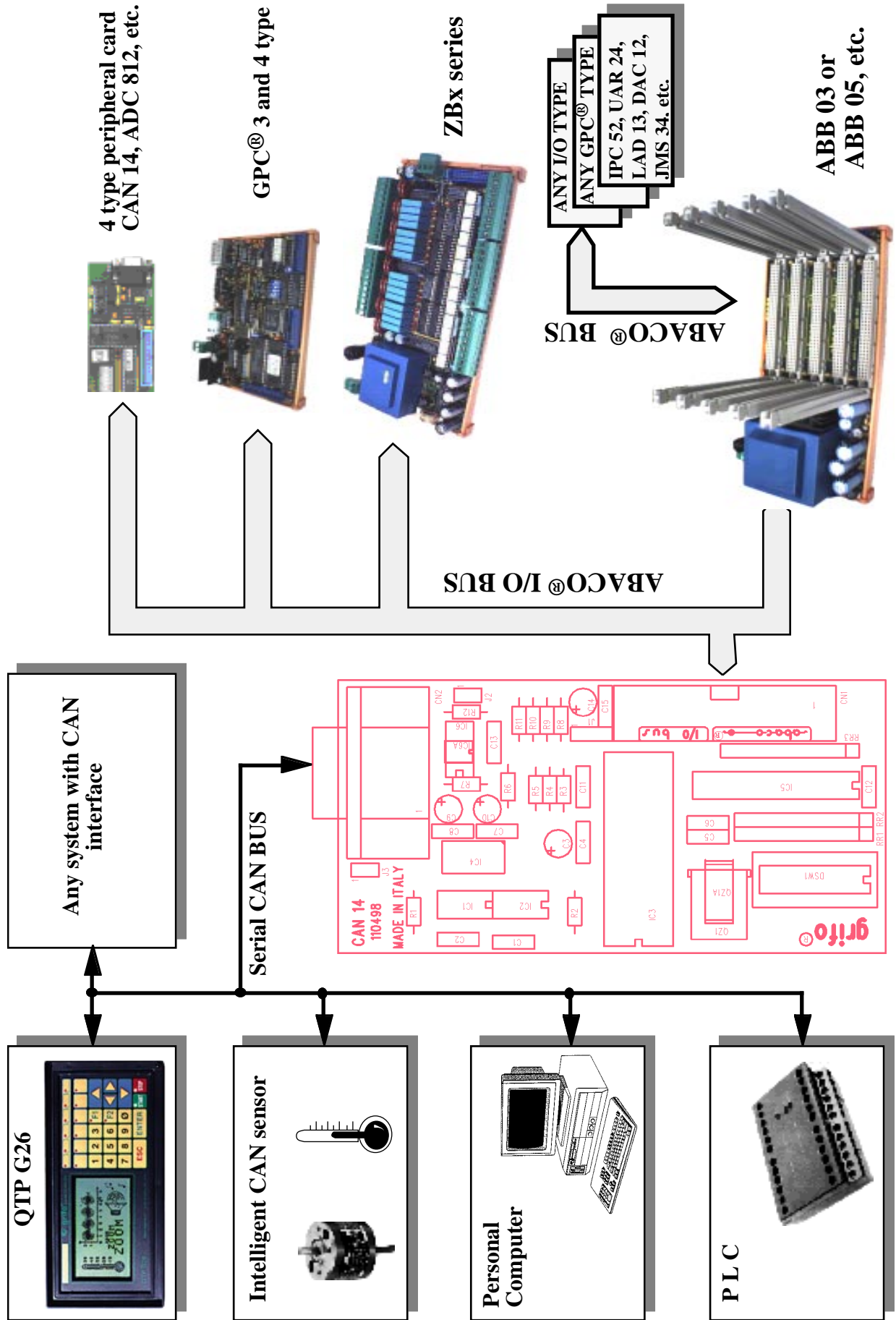


FIGURA 13: SCHEMA DELLE POSSIBILI CONNESSIONI

BIBLIOGRAFIA

E' riportato di seguito, un elenco di manuali e note tecniche, a cui l'utente può fare riferimento per avere maggiori chiarimenti, sui vari componenti montati a bordo della scheda **CAN 14**.

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

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

Manuale TOSHIBA: *Photo couplers Data Book*

Manuale NEWPORT: *DC-DC Converters*

Per reperire questi manuali fare riferimento alle case produttrici ed ai relativi distributori locali. In alternativa si possono ricercare le medesime informazioni o gli eventuali aggiornamenti ai siti internet delle case elencate.

APPENDICE A: DESCRIZIONE COMPONENTI DI BORDO

Philips Semiconductors Preliminary specification

Stand-alone CAN controller SJA1000

Stand-alone CAN controller SJA1000

4 BLOCK DIAGRAM

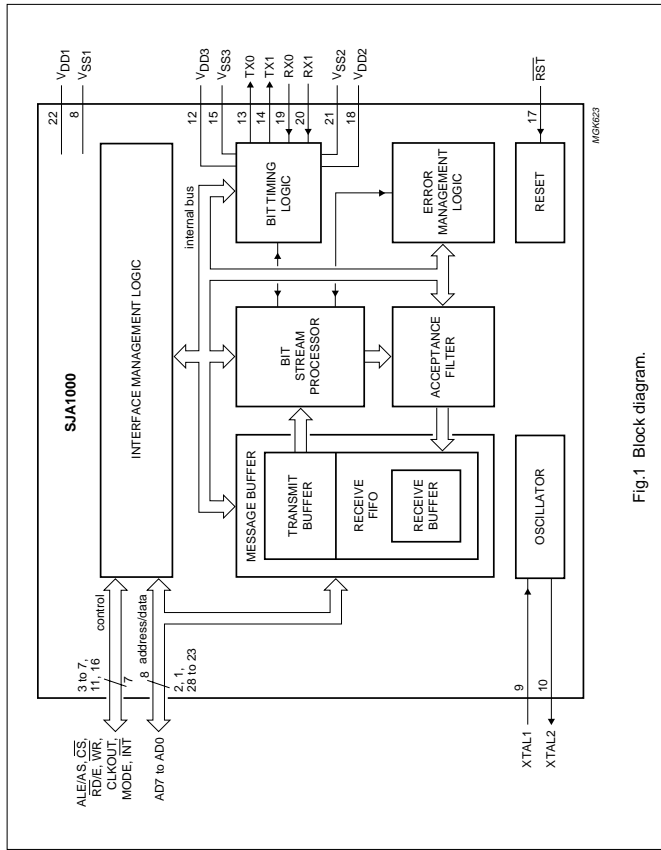


Fig.1 Block diagram.

Philips Semiconductors Preliminary specification

Stand-alone CAN controller SJA1000

Stand-alone CAN controller SJA1000

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
- Pellican 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 (Pellican) 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	S028	plastic small outline package; 28 leads; body width 7.5 mm	SOT136-1



Stand-alone CAN controller

SJA1000

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

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.

1997 Nov 04

8

Stand-alone CAN controller

SJA1000

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.

1997 Nov 04

7

Stand-alone CAN controller

SJA1000

Stand-alone CAN controller

SJA1000

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)	



Stand-alone CAN controller

SJA1000

REGISTER	BIT	SYMBOL	NAME	VALUE		
				RESET BY HARDWARE	SETTING BIT CR.0 BY SOFTWARE OR DUE TO BUS-OFF	
Acceptance code Acceptance mask Bus timing 0	AC.7 to 0	AC	Acceptance Code	X	X	
	AM.7 to 0	AM	Acceptance Mask	X	X	
	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	–	–	Clock Divider Register	00000000 (Intel); 00000101 (Motorola)	X	

1997 Nov 04

11

Stand-alone CAN controller

SJA1000

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

1997 Nov 04

12



Stand-alone CAN controller

SJA1000

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

- Any write access to the control register has to set this bit to logic 0 (reset value is logic 0).
- 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.
- Reading this bit will always reflect a logic 1.
- 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 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:
 - 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
 - 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.

Stand-alone CAN controller

SJA1000

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

- 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 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.
- 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.
- 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.
- 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.
- 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.



Stand-alone CAN controller

SJA1000

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

Stand-alone CAN controller

SJA1000

Notes

- 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.
- 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.
- If both the receive status and the transmit status bits are logic 0 (idle) the CAN-bus is idle.
- 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.
- 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.
- 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).
- 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_{seq}).



Stand-alone CAN controller

SJA1000

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 INT pin is activated (LOW). After this register is read by the microcontroller, all bits are reset what results in a floating level at 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 $t_{50\%}$.

Stand-alone CAN controller

SJA1000

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.

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:

$$\text{DataByteCount} = 8 \times \text{DLC.3} + 4 \times \text{DLC.2} + 2 \times \text{DLC.1} + \text{DLC.0}$$

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.

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.

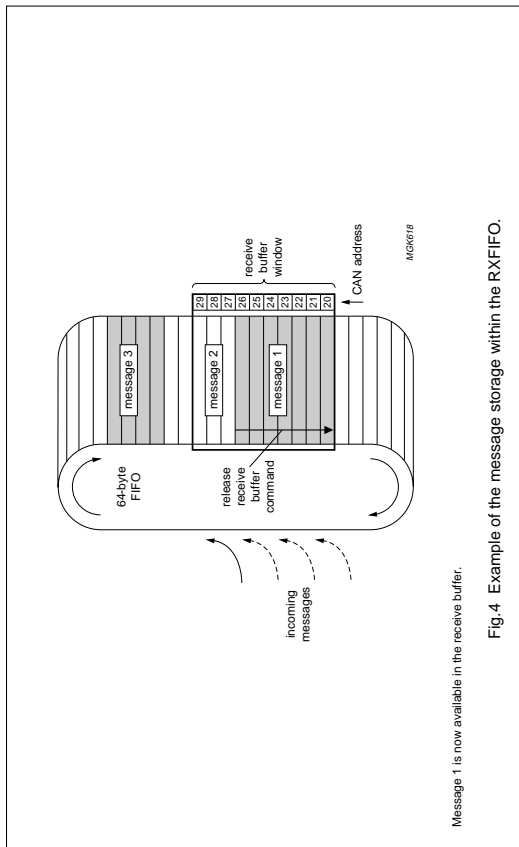
6.3.7.3 Data Length Code (DLC)

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



Stand-alone CAN controller

SJA1000



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).

6.3.9.1 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

1987 Nov 04

19

Stand-alone CAN controller

SJA1000

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:

$$(ID.10 \text{ to ID.3}) = (AC.7 \text{ to AC.0}) \vee (AM.7 \text{ to AM.0}) \\ \equiv 11111111$$

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.

1987 Nov 04

20

Stand-alone CAN controller

SJA1000

Table 10 PeilCAN 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	-
7	bus timing 1	-	bus timing 1	-
8	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	-
14	RX error counter	-	RX error counter	-
15	TX error counter	-	TX error counter	-
16	RX frame information SFF; note 2	RX frame information SFF; note 2	TX frame information EFF; note 3	TX frame information EFF; note 3
17	RX identifier 1	RX identifier 1	TX identifier 1	TX identifier 1
18	RX identifier 2	RX identifier 2	TX identifier 2	TX identifier 2
19	RX data 1	RX identifier 3	TX identifier 3	TX identifier 3
20	RX data 2	RX identifier 4	TX identifier 4	TX identifier 4
21	RX data 3	RX data 1	TX data 1	TX data 1
22	RX data 4	RX data 2	TX data 2	TX data 2
23	RX data 5	RX data 3	TX data 3	TX data 3
24	RX data 6	RX data 4	TX data 4	TX data 4
25	RX data 7	RX data 5	TX data 5	TX data 5
26	RX data 8	RX data 6	TX data 6	TX data 6

Stand-alone CAN controller

SJA1000

CAN ADDRESS	OPERATING MODE		RESET MODE	
	READ	WRITE	READ	WRITE
27	(FIFO RAM); note 4	RX data 7	reserved (00H)	-
28	(FIFO RAM); note 4	RX 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

- 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).
- SFF = Standard Frame Format.
- EFF = Extended Frame Format.
- 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.
- Some bits are writeable in reset mode only (CAN mode, CBP, RXINTEN and clock off).



Stand-alone CAN controller

SJA1000

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)
Interrupt	SR.1	DOS	Data Overrun Status	0 (absent)	0 (absent)
	SR.0	RBS	Receive Buffer Status	0 (empty)	0 (empty)
	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)	

1997 Nov 04

23

Stand-alone CAN controller

SJA1000

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
Bus timing 0	IER.1	TIE	Transmit Interrupt Enable	X	X
	IER.0	RIE	Receive Interrupt Enable	X	X
	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
	BTR1.7	SAM	Sampling	X	X
	Bus timing 1	BTR1.6	TSEG2.2	Time Segment 2.2	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

1997 Nov 04

24



Stand-alone CAN controller

SJA1000

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	OOMODE1	Output Control Mode 1	X	X
	OCR.0	OOMODE0	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

Stand-alone CAN controller

SJA1000

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



Stand-alone CAN controller

SJA1000

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

- 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 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.
- A write access to the bits MOD.1 to MOD.3 is only possible, if the reset mode is entered previously.
- 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'.
- 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:
 - 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.
 - 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.

1997 Nov 04

27

Stand-alone CAN controller

SJA1000

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

- 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).
- 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. 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.
- 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.

1997 Nov 04

28



Stand-alone CAN controller

SJA1000

- 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'.
- 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)
SR.5	TS	Transmit Status; note 3	0	ok; both error counters are below the warning limit
			1	transmit; the CAN controller is transmitting a message
SR.4	RS	Receive Status; note 3	0	idle
			1	receive; the CAN controller is receiving a message
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

Stand-alone CAN controller

SJA1000

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

- 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- After reading all messages within the RXFIFO and releasing their memory space with the command release receive buffer this bit is cleared.



Stand-alone CAN controller

SJA1000

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

1997 Nov 04

31

Stand-alone CAN controller

SJA1000

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

1997 Nov 04

32



Stand-alone CAN controller

SJA1000

Note

- The receive interrupt enable bit has direct influence to the receive interrupt bit and the external interrupt output INT. If RIE is cleared, the external 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.

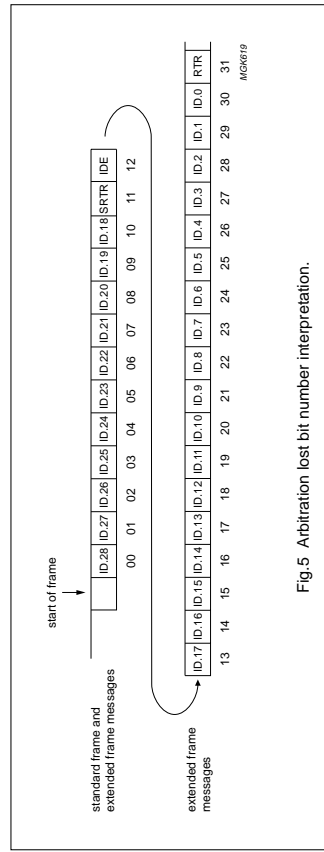


Fig.5 Arbitration lost bit number interpretation.

Stand-alone CAN controller

SJA1000

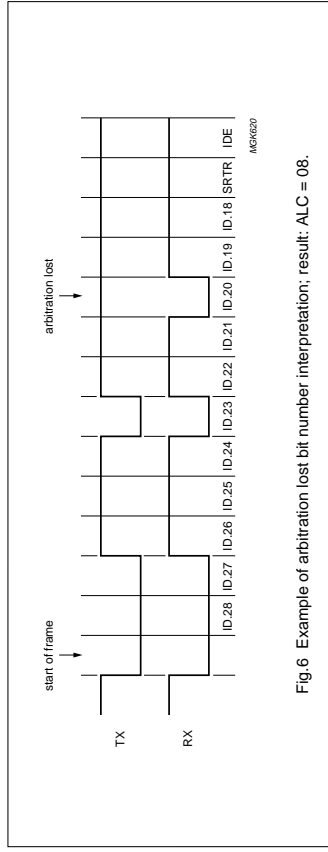


Fig.6 Example of arbitration lost bit number interpretation; result: ALC = 08.



Stand-alone CAN controller

SJA1000

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

ALC.4	BITS ⁽¹⁾				DECIMAL VALUE	FUNCTION
	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

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

1997 Nov 04

35

Stand-alone CAN controller

SJA1000

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

- For bit interpretation of bits ECC.7 and ECC.6 see Table 20.
- 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

1997 Nov 04

36



Stand-alone CAN controller

SJA1000

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	1	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 (EWLRL)

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 EWLRL 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.

Stand-alone CAN controller

SJA1000

Table 22 Bit interpretation of the error warning limit register (EWLRL); 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.



Stand-alone CAN controller

SJA1000

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).

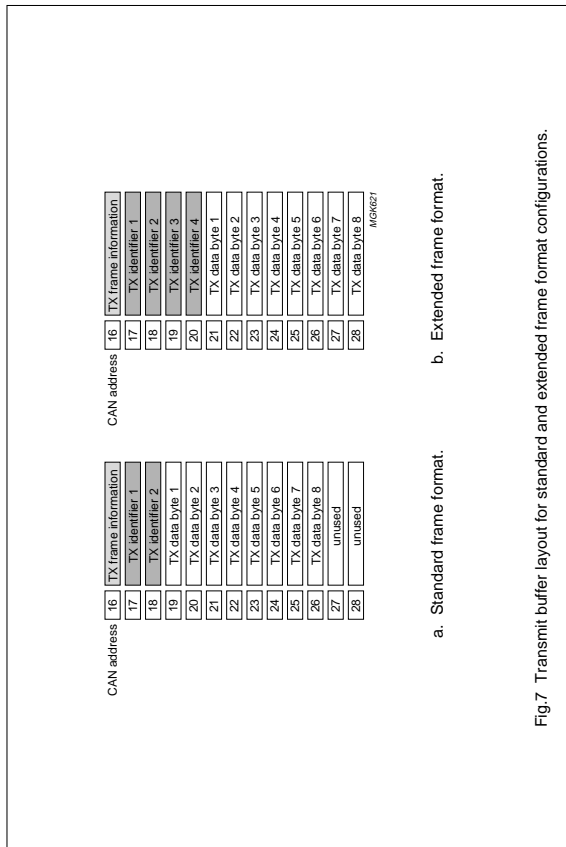


Fig.7 Transmit buffer layout for standard and extended frame format configurations.

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).

Stand-alone CAN controller

SJA1000

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

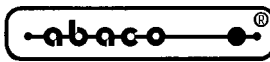
1. Frame format.
2. Remote transmission request.
3. Don't care; recommended to be compatible to receive buffer (0) in case of using the self reception facility (self test).
4. Data length code bit.

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.



Stand-alone CAN controller

SJA1000

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:

$$\text{DataByteCount} = 8 \times \text{DLC.3} + 4 \times \text{DLC.2} + 2 \times \text{DLC.1} + \text{DLC.0}$$

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.

Stand-alone CAN controller

SJA1000

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.

6.4.14 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.

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).

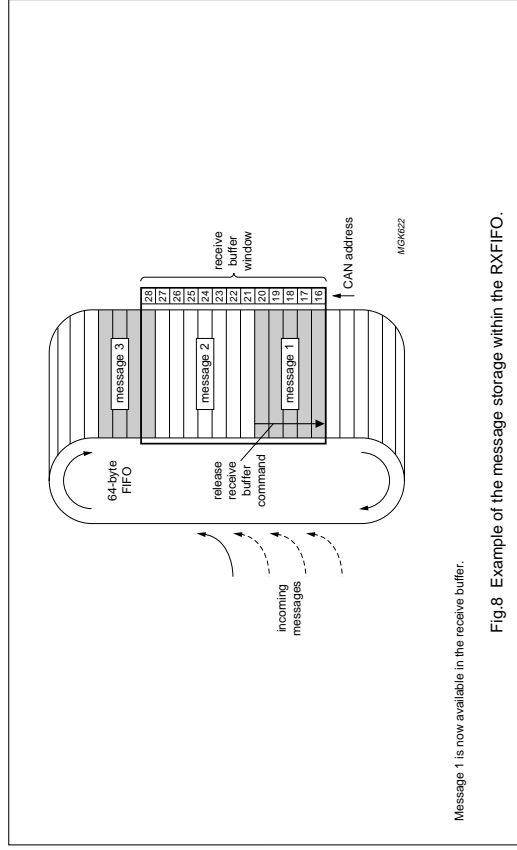


Fig.8 Example of the message storage within the RXFIFO.

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.



Stand-alone CAN controller SJA1000

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.

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.

Stand-alone CAN controller

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.

Stand-alone CAN controller SJA1000

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).

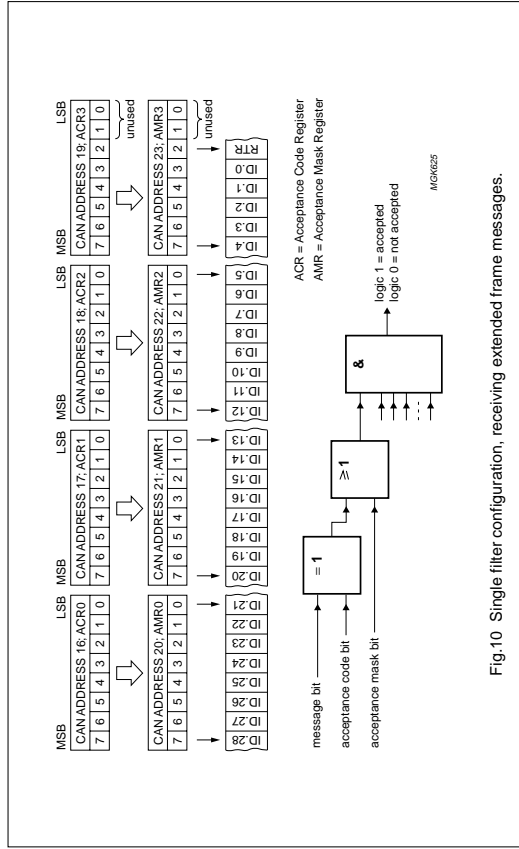


Fig.10 Single filter configuration, receiving extended frame messages.

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.

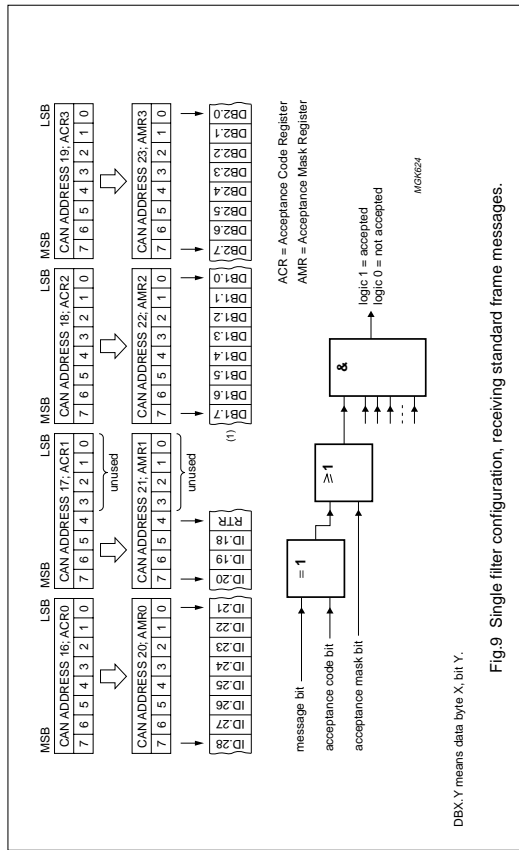


Fig.9 Single filter configuration, receiving standard frame messages.

Extended frame: if an extended frame format message is received, the complete identifier including the RTR bit is used for acceptance filtering. It should be noted that the 2 least significant bits of AMR3 and AMR3 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.

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 AMR3 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.



Stand-alone CAN controller

SJA1000

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_{sei} is programmable and determines the individual bit timing. The CAN system clock is calculated using the following equation:

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

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

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:

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

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)

Stand-alone CAN controller

SJA1000

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

- 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 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 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.

(CAN address = RBSA + 32 > 24 + 32 = 56).
If a message exceeds RAM address 63, it continues at RAM address 0.

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

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



Stand-alone CAN controller

SJA1000

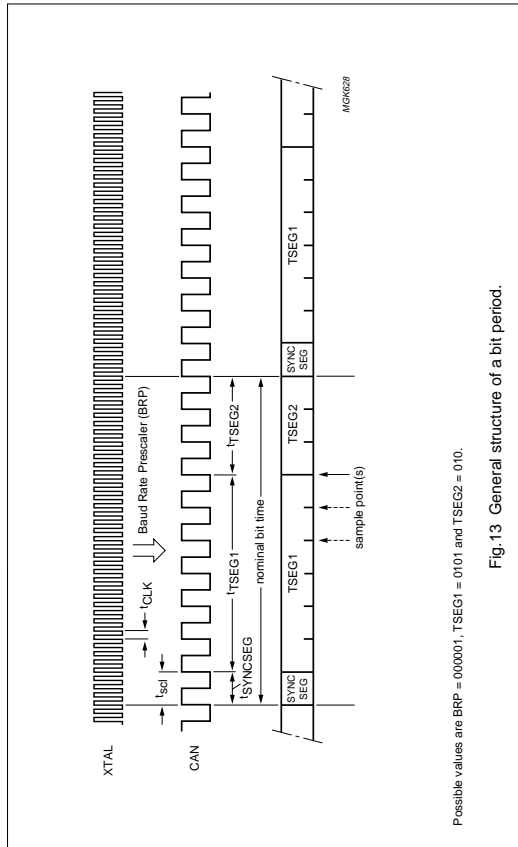
6.5.2.2 Time Segment 1 (TSEG1) and Time Segment 2 (TSEG2)

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

$$t_{\text{SYNCSEG}} = 1 \times t_{\text{scd}}$$

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

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



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

Stand-alone CAN controller

SJA1000

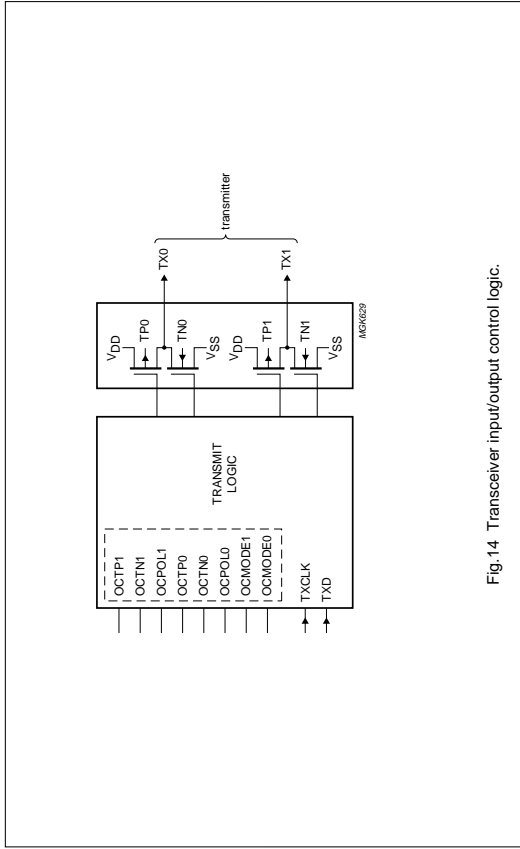


Fig. 14 Transceiver input/output control logic.

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 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

- 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.



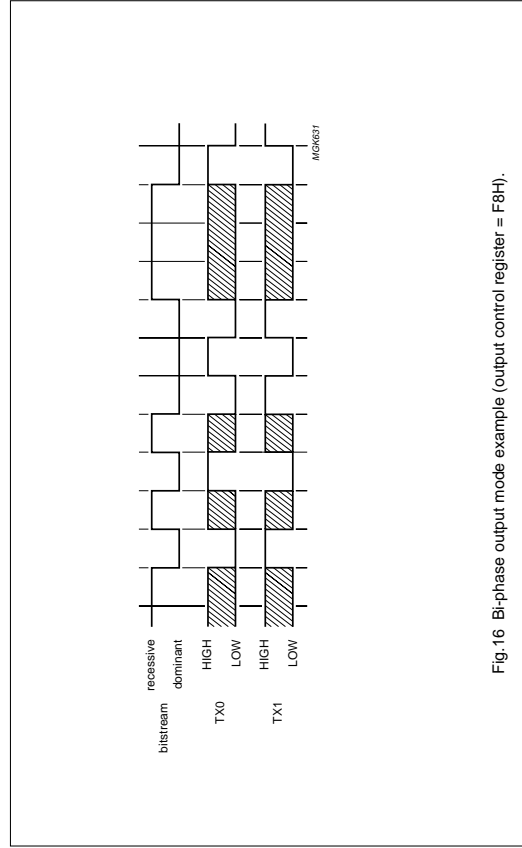


Fig. 16 Bi-phase output mode example (output control register = F8H).

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.

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 $1 \times t_{sep}$.

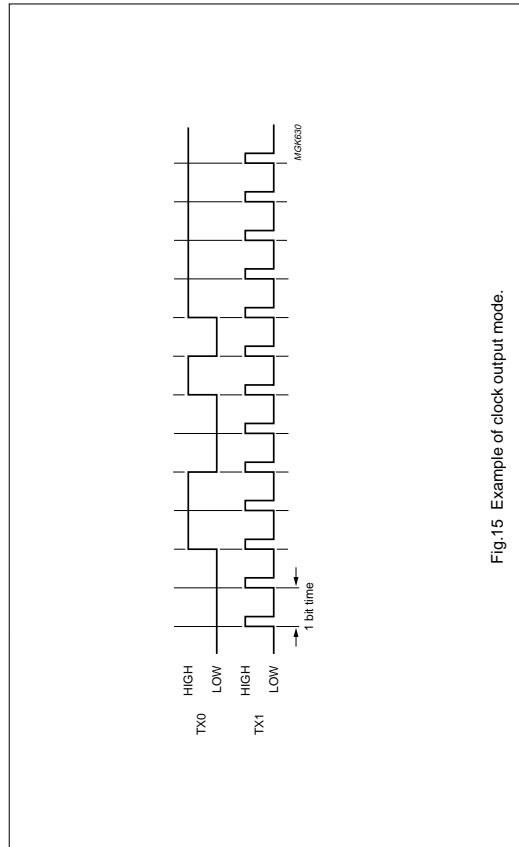


Fig. 15 Example of clock output mode.

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.



Stand-alone CAN controller

SJA1000

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
Pull-up	1	0	1	1	off	on	LOW
	0	1	0	0	off	off	float
	1	1	0	0	on	off	HIGH
Push-pull	0	1	0	1	on	off	HIGH
	1	1	0	1	off	off	float
	0	1	1	0	off	on	LOW
	1	1	1	0	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 PelicCAN 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.

Stand-alone CAN controller

SJA1000

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	$f_{osc} / 2$
0	0	1	$f_{osc} / 4$
0	1	0	$f_{osc} / 6$
0	1	1	$f_{osc} / 8$
1	0	0	$f_{osc} / 10$
1	0	1	$f_{osc} / 12$
1	1	0	$f_{osc} / 14$
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 PelicCAN mode. Write access is only possible in reset mode.



APPENDICE B: INDICE ANALITICO

Simboli

82C250 4

A

ABACO® I/O BUS 4, 8, 13

B

BIBLIOGRAFIA 20

C

CARATTERISTICHE

ELETTRICHE 5

FISICHE 5

GENERALI 5

COLLEGAMENTO IN RETE CON BUS CAN 7

CONNETTORI 6

CN1 8

CN2 6

CONSUMO 5

CONTROLLORE CAN 4, 15, A-1

D

DESCRIZIONE SOFTWARE 15

DIMENSIONI 5

DISPOSIZIONE CONNETTORI, JUMPER, DIP SWITCH 9

F

FLOW CHART DI INIZIALIZZAZIONE 16

FOTO 12

I

INFORMAZIONI GENERALI 2

INSTALLAZIONE 6

INTERFACCIA DI LINEA CAN 4

INTERFACCIA ED INDIRIZZAMENTO 4, 13

INTERFACCIAMENTO DELLA SCHEDA 10

INTERRUPT 12

INTRODUZIONE 1

J

JUMPERS 10

2 VIE 11

3 VIE 11

M

MAPPAGGIO DELLA SCHEDA 13
MONTAGGIO MECCANICO 10

P

PCx82C200 4, 15, A-1
PIANTA COMPONENTI 14

R

REGISTRI INTERNI 14
RETE BUS CAN 11
RETE CON BUS CAN 7

S

SCHEDE ESTERNE 17
SCHEMA A BLOCCHI 3
SCHEMA DELLE POSSIBILI CONNESSIONI 19
SJA1000 4, 15, A-1
SPECIFICHE TECNICHE 5

T

TERMINAZIONE LINEA CAN 11

V

VERSIONE SCHEDA 1