Users Manual

C167CR EvaBoard

C167CR Eva Small Edition

i4ME
Company
of the ACTIA group
ACTIA
Systems, Support & More

Manual
C167CR EvaBoard
Contents

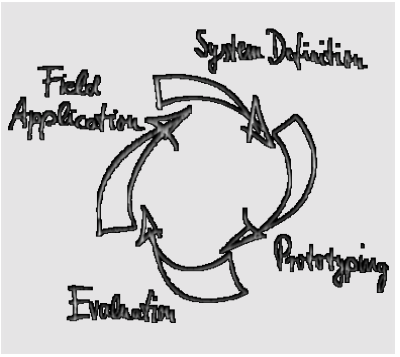
Administration of dokument

General dokument version 1.04

Welcome to ***I+ME*** ACTIA

Before acquainting you with your new ***I+ME*** Hardware we would first like to thank you for purchasing our product. We are extremely pleased that you have chosen to place your trust in ***I+ME*** ACTIA and will do our best to satisfy whatever needs you may have. The following is a brief explanation highlighting our background, areas of expertise and general product lines. This products and the list of our world-wide branch offices show that you have found a competent partner in ***I+ME*** ACTIA.

Since its foundation in 1986, ***I+ME*** ACTIA has made quite a name for itself. Our employees are dedicated to producing high-quality solutions in the field bus and multiplexed systems sectors. The knowledge of our experts allows to develop a spectrum of products which have been used in the automotive field as well as in general industrial environments. Our products can be used in all phases of system development: system definition, prototyping, evaluation and field application.



I+ME ***Informatik und MikroElektronik***

Whether your professional background is into industry-process-control or development and test tools, we offer six product groups to fulfill your sophisticated needs. Tried and tested under the most severe conditions the automotive industry has to offer, our products have proved themselves again and again.
Our six products groups are:

1 CAN System Test & Design Tools

Support of various user application phases: Learning, proto-typing, testing and evaluation of networked systems. Comfortable real-time simulation of message transfer characteristics in CAN networks. Tools for mobile diagnosis and tests.

2 CAN PC Interfaces

Easy interfacing between PCs, Laptops, notebooks and networks with automotive fieldbus – protocols. Available for all PC standard interfaces such as ISA, PCI, backplane, RS232, Centronics and PCMCIA. Development of applications under Windows according to real-time requirements is supported

3 CAN Industrial I/O

CAN-IO is an intelligent hardware concept for sensor / actuator interfacing. A modular architecture allows the flexible change target micro controllers for process control.

4 CAN System Application Software

Enabling real-time system modeling, testing of networked systems as well as application support. Offering basic services for network communication which is applicable for various processors and programming languages. Facilitating the application interface for distributed industrial process control according to the CAL standard by CiA. Support of all Windows 32bit platforms.

5 CAN System Know How

Promoting the understanding of various network protocols in practice. Understanding of CAN networks with CAL in practice. Developing HW/SW solutions for customer specific problems. We offer CAN / CAL workshops and in-house seminars to enable CAN users to benefit from I+ME ACTIA's extensive knowledge.

6 Automotive Diagnostics

Assistance during the development phases. Diagnostic tools for quality control in production lines as well as after sales diagnostic, control and servicing tools are provided to manufacturers, suppliers and dealers of the car industry by I+ME ACTIA.

If you have any questions concerning our products or you look for specific solutions within our product groups,

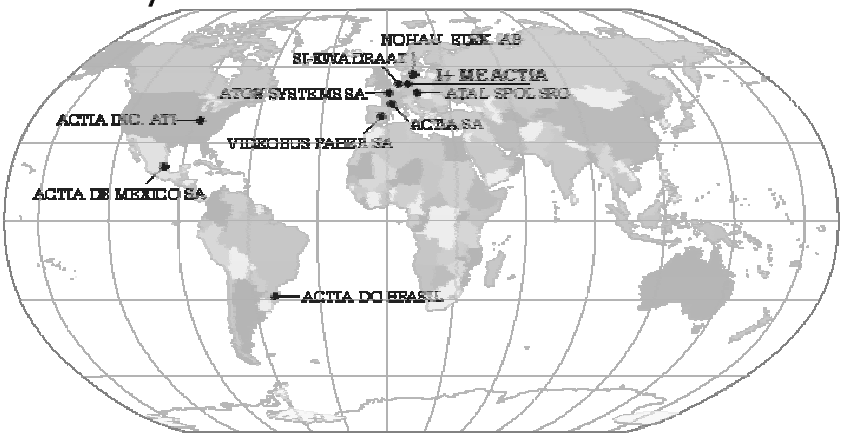
don’t hesitate to call us and benefit from I+ME’s extensive knowledge - your need is our desire.

Our merger with the French corporation ACTIA in 1995 allowed us to become a powerful supplier for the European automotive industry. ACTIA products include diagnostic systems for automotive service and maintenance as well as development and production of high quality on-board electronics. Joining forces with ACTIA has enabled I+ME to better service it’s international customers not only in Europe, but throughout the world.

Headquarter:

I+ME ACTIA GmbH
Rebenring 33
38106 Braunschweig
Germany
T: +49 531 38701 0
F: +49 531 38701 88

I+ME WORLDWIDE REPRESENTATIVES



I+ME ACTIA Representatives

SI-KWADRAAT
Nuenen, Netherlands
T: +31 40 2.631.185
F: +31 40 2.838.092

ACTIA SA
Toulouse, France
T: +33 (0)5 61.17.61.61
F: +33 (0)5 61.55.42.31

NOHAU ELEK. AB
Malmö, Sweden
T: +46 40 59.22.00
F: +46 40 59.22.29

ACTIA INC. - ATI
Elkhart, Indiana USA
T: +1 (219) 293.75.05
F: +1 (219) 293.75.39

If needed, please contact our associates below.

ATAL SPOL SRO
Tabor, Czech Rep.
T: +420 361 251 791
F: +420 361 253 043

ACTIA DO BRASIL
Porte Alegre, Brazil
T: +55 51 337.90.17
F: +55 51 337.60.81

ATON SYSTEMS SA
France
T: +33 0 1.42.07.18.00
F: +33 0 1.42.07.85.55

ACTIA de Mexico SA
Mexico
T: +525 119.22.78
F: +525 119.22.79

VIDEOBUS Paher SA
Getafe Madrid, Spain
T: +34 91 665 26 26
F: +34 91 665 23 24

Your I+ME *C167CR EVA Board*. Overview, System Requirements, Delivery Contents, and Specifications.

1 Introduction

1.1 Your I+ME C167CR Eva Board

The I+ME C167CR Eva Board or C167CR Eva Small Edition is a universal evaluation system for the Siemens C167CR microcontroller family. On-board memory makes it ideally suitable for stand-alone operation. Besides an on-board ISO-highspeed transceiver for the C167's on-chip CAN-controller of the C167CR the board is ready for the whole family of I+ME's protocol and transceiver babyboards. Thus you can use this hardware in a wide variety of different networks. Using both the C167C's CAN-controller and any other protocol babyboard you can even realize gateways and bridges between different networks.
In addition all relevant processor signals are externally available; So you can easily add your own hardware extension. Although the on board hardware is designed for 16-bit non-multiplrexed bus external components can use any other bus mode using a different bus configuration.
A set of 8+1 LED's and 8 DIP switches complement the system for simple digital input/output operations. One ADC channel and one PWM channel are available via on-board amplifiers for simple analog input/output. A standard RS232 serial interface offers easy connection to almost any host system. With a hardware reset option on the RS232 interface it offers a convenient host controlled procedure for **B**oot**S**trap **L**oader (BSL) operation.
Most functions of the board and the microcontroller are jumper selectable. This includes memory types, complete system startup configuration and network source.

I+ME ACTIA is always eager to fulfil the needs of our customers. If problems should occur, please refer to **Troubleshooting**. If the problem persists, then feel free to contact our after-sales support hotline using the following number:

After-sales service

I+ME ACTIA GmbH
Rebenring 33
D-38106 Braunschweig
Germany.

Fon: ++ 49 (531) 38 701 38
Fax: ++ 49 (531) 38 701 88

mail : info@ime-actia.de

1.2 Startup with C167CR Evaboard

This chapter is only relevant for user of the C167CR Small Edition. Owner of an full equipped C167CR EvaBoard can skip the chapter, because the stand-alone firmware is already implemented in the mounted EPROM‘s. Only the CAN-Bus must be connected and power supplier must be plug in, to use the board as automatic transmission node.

User who got an C167CR EvaSmall Edition and want to use these board as well must load down firmware in to the RAM.

Step 1:
Install delivered software.

Step 2:
Plug in power supplier and connected the serial port with the serial interface of your PC.

Step 3:
Change in DOS Mode. Select the directory BSL and call the program" ldserial sample (x)". (x) COM port number of the PC (1 or 2).

Now the firmware for automatic transmission is loaded down into the RAM. Description of these program see chapter 6 "Samples".

1.3 System Requirements

For interface using:

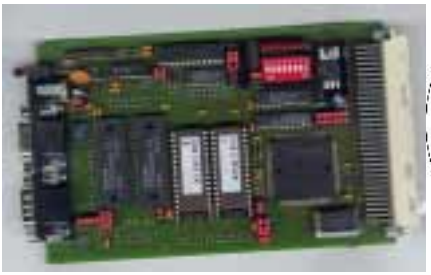
- PC with min. Pentium 133 MHz,
- 32 MB RAM,
- RS232 Interface.
- Operating system Windows 9x / NT.

For designing own firmware:

- Compiler for C167C e.g. Keil

1.4 Delivery Contents

1.4.1 Standard



- 1 C167CR Eva Board (IME 2308 205).
- 1 set EPROM with stand-alone firmware.
- 1 CD with I+ME universal CAN firmware as HEX file for standalone modus, DOS applications and all electrical schemes. Software LevelX and PcCANControl

1.4.2 Small Edition

- 1 C167CR Eva Small Edition (IME 2308 204)
- 1 CD with I+ME universal CAN firmware as HEX file for standalone modus, DOS applications and all electrical schemes. Software LevelX and PcCANControl

1.4.3 Supplementary Support

- Protocol-Baby-Board (PrBB) for other CAN chips.
- Transceiver-Baby-Board (TxBB)for other transceiver.
- Powersupply.
- EPROMs with stand-alone firmware.
- Compiler for C167.
- Potentiometer extension.
- Indication extension.

1.5 Technical Specifications

General characteristics of C167CR EVA-board

- Supports all members of the C167 family.
- 128k*16 SRAM on-board.
- 64k/128k/256k*16 on-board EPROM / Flash / SRAM (optional).
- Supports the full 16 MByte address range of the C167 family.
- 8+1 LEDs and 8 DIP switches on-board.
- All CPU signals externally available on 160pin DIN connector (optional).
- On-board 5V voltage regulator.
- On-board voltage supervisor & reset generator.
- On-board RS232 interface.
- Support for Bootstrap Loader via RS232.
- On-board CAN interface compatible to ISO DIS 11898 and CiA (optionally galv. decoupled).
- On-board one channel audio input and output amplifiers/filters.
- System “Startup Configuration” completely jumper selectable.

Your I+ME C167CR EvaBoard.Software Installation.

2 Installation

2.1 Installation

The I+ME C167CR EvaBoard comes with a PC-based control software. Follow these steps in order to get the control software running on your host PC:

- Make sure all jumpers are set correctly, especially BOOTSTRAP-jumper (refer to chapter 'Jumper description')
- Connect the RS232 interface of the C167CR EvaBoard to your host PC
- Connect the EvaBoard to a suitable power supply: 7-20V DC, ~300mA
- insert the delivered installation CD, start “START.EXE” if no autostart is activ and follow the requests on screen.

2.2 Installation trouble shooting

If the control software gives an error message check the following points:

- Make sure you have connected the EvaBoard to the same serial port that you stated when invoking the control software
- Make sure you have connected the EvaBoard to suitable power supply (see step 3)
- Check all jumpers (see step 2)
- Remove any external hardware and try again

If you still face problems contact I+ME!

Your I+ME C167CR Eva Board. Overview, and Specifications.

3 Hardware

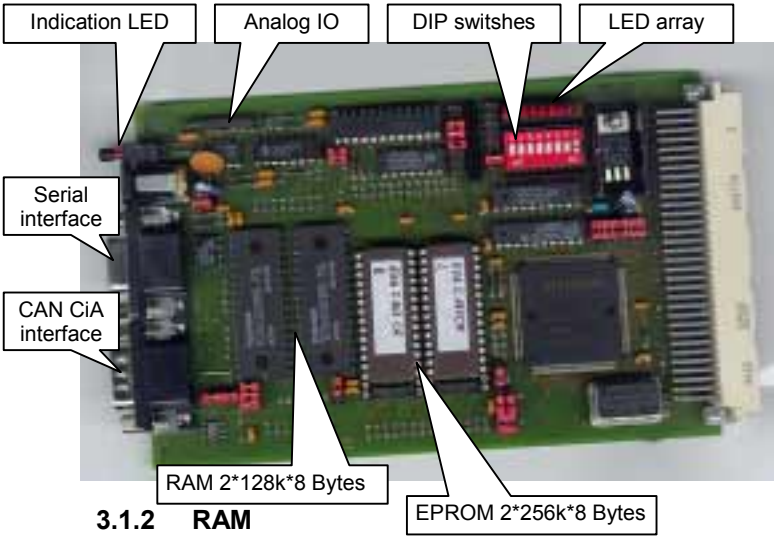
3.1 Functional Blocks

The I+ME C167CR EvaBoard by default is equipped with a C167CR. Nevertheless, the board is designed to work with all known members of the C167 family. Ask us for other versions.

In order to give you full access to all features of the C167 microcontroller the EvaBoard has a complete set of jumpers for the Startup Configuration. Even bits which are currently marked as reserved are available. Thus the hardware is prepared for future extensions.
Nevertheless, while you are using the on-board components you are restricted in some settings. For a detailed description of the Startup Configuration jumpers refer to chapter 'Jumper Description'.

The EvaBoard was designed to keep as much resources of the C167 as possible to you. Therefore only one CS-output of the C167 is used. All other address decoding is done in an external GAL. For more information on the address map refer to chapter 'Memory Map'.

3.1.1 Board



3.1.2 RAM

The I+ME C167CR EvaBoard is equipped with 256 kbytes (2*128k*8) of world-wide static RAM. In order to allow maximum performance it uses the 16-bit non-muxed bus mode.

3.1.3 ROM

By providing 2 standard memory sockets the C167CR EvaBoard can be equipped with up to 512 kbytes (2*256*8) of EPROM. Alternatively you can use FlashEPROMS and even additional RAMs.
These memory sockets are to be used on a 16-bit non-muxed bus, too.

3.1.4 DIP Switch

An 8-fold dip-switch is available for simple input purposes. The switches have no predefined meaning but are completely software dependant. From the application's point of view the switches are a read only memory location. Due to limited resolution of the address decoder the switches actually show up at several consecutive locations. For further information refer to chapter 'Memory Map'.
A switch in position 'on' reads as '1'.

Since the switches are a one address 8-bit device they can be accessed in any bus mode.

3.1.5 LED Array

For simple indication tasks the C167CR EvaBoard has an array of 8 LEDs. From application point of view the LEDs are a write only memory location. Due to limited resolution of the address decoder the LEDs actually show up at several consecutive locations. For further information refer to chapter 'Memory Map'.
Writing a '1' will turn on the corresponding LED.

Since the LEDs are a one address 8-bit device they can be accessed in any bus mode.

3.1.6 Indication LED

One additional LED is driven directly from a processor port. It is especially useful for single bit status indications. Note that this LED is turned on by setting the corresponding port to '0'! Of course you have to program this port line as output when you use it from software. Anyway, since the LED is buffered it will

always show the actual (inverted) logic state at this port line even if you use it as input.

3.1.7 Serial Interface

ASC0 of the C167 is hard-wired to an on-board RS232 transceiver. This allows an easy connection to almost any host system. Although the corresponding lines are externally available at least the RxD line must remain dedicated to the transceiver.
Two special features make this standard interface especially suitable for the Bootstrap Loader: The 'DTR' signal can be jumpered to act as (active low) hardware reset. Thus the host can initiate a BSL sequence without manual reset! In addition the signal 'Ring Indicator' is always low. This can be used as a first check whether a C167CR EvaBoard hardware is connected.

3.1.8 Analog Input / Output

The EvaBoard contains on-board analog input/output amplifiers/filters. With these components you can easily make first steps towards analog signal processing. However, this is no high-end data acquisition module. Due to the immediate vicinity of analog and digital components you have to make compromises in noise and accuracy.

PWM channel 0 is used as analog output. An amplifier/filter gives an output amplitude of 1 Vss and reduces the bandwidth to about 7 kHz in order to smoothed the signal. Although PWM 0 is used its output is still directly available. The amplifier load is just about 10 kW.

ADC channel 0 is used for the analog input. An anti-aliasing filter reduces the bandwidth to about 100 kHz. A fixed amplification by 5 allows an input voltage of 1

Vss. Note that two inputs are available - a DC coupled one and an AC coupled one. The input impedance is about 22 kW.

3.2 CAN Network

The C167CR EvaBoard has a very versatile network interface. A standard D-sub connector can be fed from an on-board ISO DIS 11898 transceiver or from an additional transceiver babyboard (TxBB). The data can be derived from the C167C's on-chip CAN-controller or from an additional protocol babyboard (PrBB). Thus you can choose from a wide variety of physical interface / protocol combinations.
Without additional babyboards you have a CiA (CAN in Automation) conformant CAN interface which can optionally be galvanically decoupled.

3.2.1 On-Board Transceiver

The EvaBoard has an ISO DIS 11898 conformant CAN transceiver on-board. It can be driven from a protocol baby board or directly from the C167C. Optionally this transceiver can be galvanically decoupled to reduce problems with ground offsets on long bus lines.

3.2.2 Transceiver Babyboard

Optionally a transceiver babyboard can be plugged on the EvaBoard. It lies in parallel with the on-board transceiver. Thus you should not have them simultaneously on your board.

3.2.3 On-Chip CAN Controller

The C167C's CAN-controller can be jumpered to drive the on-board transceiver (and the TxBB which lies in

parallel). The CAN-signals are always externally available.

3.2.4 Protocol Babyboard

For other protocols or just other CAN controllers a protocol babyboard can be installed. However, in this case the C167C's CAN must be disconnected from the transceiver! Of course it can still be used with an external transceiver.
The PrBB shows up as memory mapped device on the C167's bus. Available bus modes, bus timings and exact memory map depend on the actual babyboard.

The PrBB uses several special port lines of the C167. For detailed information refer to chapter 'Reserved I/O-Ports'. Of course these lines can be universally used as long as no PrBB is installed.

3.3 Boot Modes

In order to make the C167CR EvaBoard a flexible development system it offers two operational modes concerning the memory scheme. By supporting the C167's Bootstrap Loader (or any other download procedure) for loading complete applications into on-board RAM the EvaBoard allows short cycles during software development and debugging. In order to let you test your application at the same addresses in RAM and in final EPROM the EvaBoard has two memory schemes.

- In the first mode 'ROM-Boot' the on-board EPROM is located at address 0h. Thus, after reset the C167 will execute the application available in the EPROM.
- Mode two 'RAM-Boot' locates the on-board RAM at address 0h. In this mode the application previously loaded into the RAM will be executed.

The operational mode is selected by a hardware flipflop. This flipflop is reset to 'ROM-Boot' with every hardware reset. Thus ensuring that the EvaBoard will power up with a (hopefully) valid application from EPROM. Once running the software can set this flipflop, too. Therefore after booting from EPROM you can switch to 'RAM-Boot' mode. Now all interrupt vectors lie in RAM area and can be altered.

A software reset will not alter the flipflop state. This is especially useful for application downloading. The download procedure - which will probably run in EPROM or in internal RAM - can select 'RAM-Boot' mode and perform a software reset. Thus the just loaded application will start with a true reset and at the final addresses (not just a restoration of some registers and then a jump).

For further information on the flipflop access and memory schemes refer to chapter 'Memory Map'.

3.4 Memory Map

The C167CR EvaBoard has basically two different memory schemes. They are determined by the external address decoder. In addition some periphals are selected by the C167's CS1 signal. Thus they overlay the basic memory map. On third level the internal memories and XBUS devices of the C167 override all external memories.
The following chapters will discuss the basic memory map and the overlaid periphal functions in detail.

3.4.1 ROM-Boot Mode

In this mode the on-board EPROM is mirrored at address 0h. Since this area is overlaid by the C167's internal memories and some peripheral functions not the whole EPROM is usable in this tile. To overcome this restriction you can use the 'original' image in the 2nd tile. Above all this tile will remain untouched when you switch to 'RAM-Boot' mode.
The Boot-Mode flipflop is a special function located in the 2nd tile. Since you don't need permanent access to this function it needs not to always overlay this tile but only if you need it.

Address	Device	Bus mode	Addr.decoder
000000h – 03FFFFh (1st 256kB tile)	On-board EPROM(1)	16bit nonmux	External
	Internal memories(2)	n/a	n/a
	LED-array(3)	any	CS1 + external
	Dip-switch(3)	any	CS1 + external
	PrBB(3)	8bit mux	CS1 + external
040000h – 07FFFFh (2nd 256kB tile)	on-board EPROM	16bit nonmux	External
	Boot-Mode flipflop(3)	any	CS1 + external
080000h – 0BFFFFh (3rd 256kB tile)	on-board RAM	16bit nonmux	external
0C0000h – FFFFFFh (4th - 64th 256kB tile)	free / user specific	user specific	user specific

(1) Mirrored from 040000h-07FFFFh.
(2) The exact size of the internal memories depends on the actual C167-type.
(3) Refer to the chapter 'Periphals'.

3.4.2 RAM-Boot Mode

In this mode the on-board RAM is mirrored at address 0h. As long as this area is overlaid by the C167's internal memories and some peripheral functions, not the whole RAM is usable. To overcome this restriction you can use the 'original' image in the 3rd tile. This mode is especially useful for downloading applications into RAM (for testing) or for EPROM applications which must modify their interrupt vectors.

The Boot-Mode flipflop is a special function located in the 2nd tile. Since you don't need permanent access to this function it needs not to always overlay this tile but only if you really use it.

Address	Device	Bus mode	Address decoder
000000h - 03FFFFh (1st 256kB tile)	on-board RAM(1)	16bit nonmux	External
	internal memories(2)	n/a	n/a
	LED-array(3)	any	CS1 + external
	dip-switch(3)	any	CS1 + external
	PrBB(3)	8bit mux	CS1 + external
040000h - 07FFFFh (2nd 256kB tile)	on-board EPROM	16bit nonmux	External
	Boot-Mode flipflop(3)	any	CS1 + external
080000h - 0BFFFFh (3rd 256kB tile)	on-board RAM	16bit nonmux	external
0C0000h - FFFFFFh (4th-64th 256kB tile)	free / user specific	user specific	user specific

(1) Mirrored from 080000h-0BFFFFh

(2) The exact size of the internal memories depends on the actual C167-type.

(3) Refer to the chapter 'Periphals'

3.4.3 Periphals

On the C167CR EvaBoard the first three 256kB tiles are basically used for memory. Nevertheless there are some periphal functions which overlay this basic memory map. The memories are selected by the external address decoder without using any of the C167's CS lines. All periphal functions are accessed via CS1. Therefore your application has to set up ADDRSEL1 and BUSCON1 in order to access these functions. On the other side these functions do not overlay the basic memory if you do not activate CS1!

The following functions are accessible with this mechanism:

- on-board PrBB socket
- on-board LED-array
- on-board dip-switches
- Boot-Mode switch

Which periphal function is accessed depends on the address range in which CS1 becomes active. The address decoder evaluates the address lines A10, A18-A23. Therefore in the first place it decodes the 256kB tile. In addition it can distinguish even and odd 1kB tiles. In conjunction with the CS1 signal - which allows a resolution of 4kB - this results in the following overlay scheme:

tile #[3]	address[1]	function	bus mode
000000h - 03FFFFh (1st 256kB tile)	xxx000h-xxx3FFh xxx400h - xxx7FFh dto. xxx800h -	PrBB read: dip-switch write: LED-array mirrored	8bit mux any dto

	xxxFFFh	from 000 - 7FF	
040000h - 07FFFFh (2nd 256kB tile)	xxx000h - xxx3FFh	write: set ROM-Boot mode(2)	Any
	xxx400h - xxx7FFh	write: set RAM-Boot mode(2)	any
	xxx800h – xxxFFFh	mirrored from 000 - 7FF	

(1) The relative position within a 256 kB tile is not determined by hardware. Nevertheless we recommend to map the overlays at the top of each tile. Also it is not inforced by hardware to program the CS1 area to 4kB. Nevertheless we recommend it because even this size is larger than actually necessary.

(2) The Boot mode flipflop is set be the mere write access. The data do not matter. Also read accesses will not influence this flipflop but will access the underlying EPROM.

(3) Do not program CS1 to other tiles than listed here.

3.5 Reserved I/O-Ports

Although all CPU pins are externally available some of them are used for on-board purposes.
Especially with inputs you must **be careful** not **to drive** a **line** from **outside** which is already driven on-board.

- **P0.0-P0.15: Address/data bus**
  The C167CR EvaBoard is meant to be used with external memory. Therefore this port is normally reserved. Nevertheless if your application fits into internal memory you can remove the on-board memory and use this port as universal I/O.
- **P1.0-P1.15: Address bus**
  See P0.
- **P2.0: PrBB INT**
  This is the low-active interrupt line from PrBB to the C167. Nevertheless it can be safely used as long as no PrBB is present.
- **P2.1: PrBB Reset**
  This is a low-active hardware reset for the PrBB. The signal is **OR**ed with the system reset. As long as no PrBB is installed you can use this line as normal I/O.
- **P2.2: PrBB Control**
  This is general purpose input from the PrBB. You may use this line as long as no PrBB is present.
- **P2.3: Indication LED**
  This output drives the Indication LED buffer. A logical '0' will turn the LED on. You can use this line universally if you don't need the function of the LED.
- **P3.10: TxD**
  This output is used as TxD for the on-board

RS232 interface. Use this line only if you don't need the RS232 link.

- **P3.11: RxD**
  This input is driven by the on-board RS232 interface. Never drive this line from outside!

- **P3.12: #WRH**
  This line is used as bus control signal.

- **P4.0-P4.7**
  These lines are used as address lines or as on-chip CAN RxD/TxD. Be sure not to interfere with these special functions.

- **P5.0: ADC0**
  This analog input is used by the on-board audio input. Drive this line only if you have removed the input amplifier!

- **P6.1: #CS1**
  This signal is needed to access the on-board periphery. You can use it for additional purposes but be careful not to interfere with the on-board function.

- **P7.0: PWM0**
  This output drives the on-board audio output. Since the output amplifier has a rather high input impedance it won't interfere with your own applications.

3.6 Jumper Description

The C167CR EvaBoard has a lot of options which are accessible by jumpers. The following list describes the function of each jumper setting. The table shows which pins of each strip are connected.

The default setting configures the board for EPROM-less BSL operation. The jumpers to be set are marked with an asterisk (*).

Definition:

*	mark the default settings
1-2	connect pin 1 and 2
0-0	no Jumper is set
x-x	the jumper set is no matter

JP1,JP2	CPU clock distribution
0-0,0-0*	CPU clock just on-board
1-2,0-0	CPU clock available on DIN connector
2-3,0-0	CPU clock driven from PrBB
x-x,1-2	dto. & PrBB clock available on DIN connector

JP3	ROM type selection
1-2	ROM socket: pin 30 = Vcc
2-3*	ROM socket: pin 30 = A17

JP5,JP7	RAM/ROM selection both strips should be configured identically. There are two jumpers on each strip.
1-2 & 3-4	ROM-socket: pin 31 = WR, pin 29 = A15
2-3 & 4-5*	ROM-socket: pin 31 = A15, pin 29 = WR
JP8	RAM type selection
1-2*	RAM-socket: pin 30 = Vcc
2-3	RAM-socket: pin 30 = A17
JP9, JP10 JP13, JP14	Onboard transceiver power all strips should be configured identically.
1-2*	all jumper set - direct power from EvaBoard
0-0	no jumper set - galv. decoupled power from EvaBoard
JP11, JP12	Input selection for transceiver Both strips should be configured identically.
1-2*	On-chip CAN
0-0	PrBB - Protocol babyboard
JP16	Boot selection
1-2	#EA = Vcc: Boot from on-chip ROM
2-3*	#EA = GND: Boot from external memory

JP17-JP32			System Startup Configuration
		1-2	Pull down
		0-0	Float
AD0	J17	0-0	Emulation mose
AD1	J18	0-0	Adapt mode
AD2	J19	0-0	reserved
AD3	J20	0-0	reserved
AD4	**J21**	**1-2**	**Bootstrap loader mode (BSL)**
AD5	J22	0-0	reserved
AD6	J23	1-2*	Bus mode - demuxed
		0-0	muxed
AD7	J24	0-0*	Data width – 16Bit
		1-2	8Bit
AD8	J25	0-0*	WRC mode - BHE
		1-2	WRL, WRH
AD9	J26	0-0	Num. of chip selects (CSSEL0)
AD10	J27	1-2	Num. of chip selects (CSSEL1)
		J26 J27	
		0-0 & 0-0	5 CS lines
		0-0 & 1-2	0 CS lines
		1-2 & 0-0	2 CS lines
		1-2 & 1-2	3 CS lines
AD11	J28	1-2	Num.seg.addr.lines (SALSEL0)
AD12	J29	1-2	Num.seg.addr.lines (SALSEL1)
		J28 J29	
		0-0 & 0-0	A17 (256k)
		0-0 & 1-2	A23 (16M)
		1-2 & 0-0	A15 (64k)
		1-2 & 1-2	A19 (1M)
AD13	J30	0-0	reserved
AD14	J31	0-0	reserved
AD15	J32	0-0*	direct clock (c167 only)
		0-0	*4 or *1/2
		1-2	direct clock

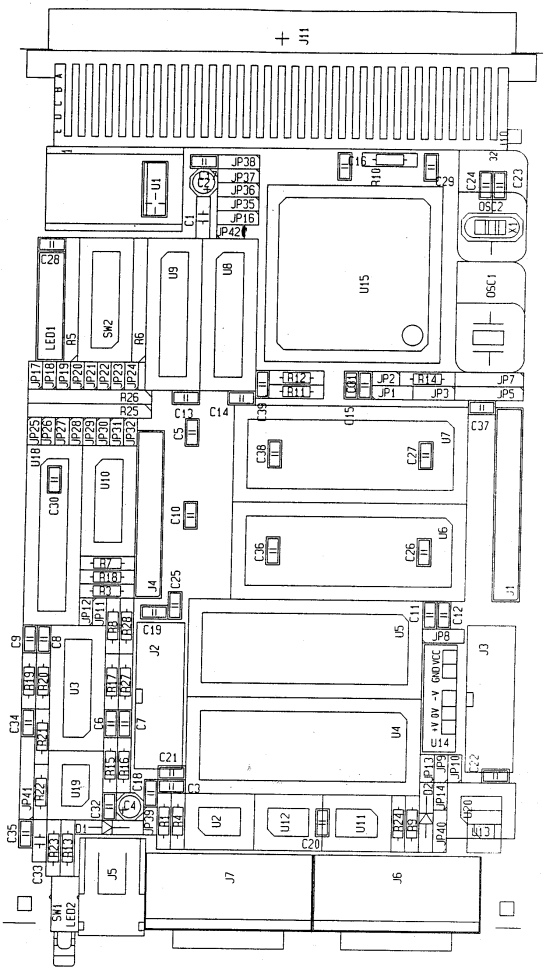
Set by default: JP21, JP23, JP27, JP28, JP29

JP35-JP38	High address lines

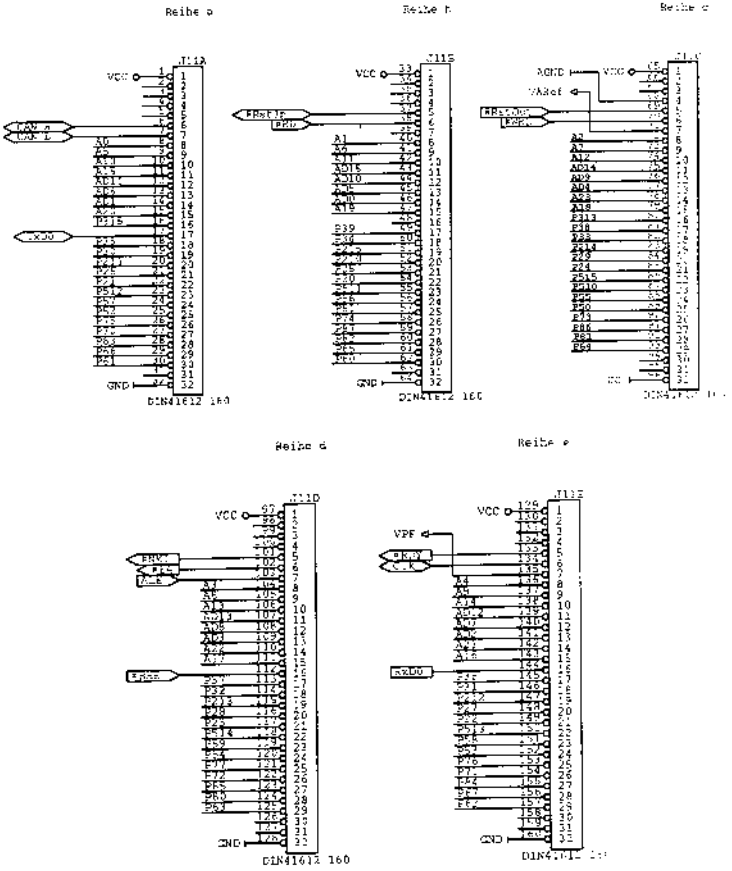
		If not all address lines are enabled the adress decoder must get '0' instead! JP35-38 correspond to A20-A23.
	1-2	Address decoder gets Ax from CPU.
	2-3*	Address decoder gets constant '0' for Ax.
JP39		RS232 reset
	1-2*	Enable hardware reset via RS232.DTR
	0-0	disable
JP40		Transceiver slope control Depends on transceiver!
	1-2	Rs = 0 to Vcc
	2-3*	Rs = 0 to GND
	0-0	Rx = x
JP42		Vpp source
	1-2*	Vpp = Vcc (Vcc is now present on DIN connector, too!)
	0-0	Vpp supplied via DIN connector

3.7 Schematics

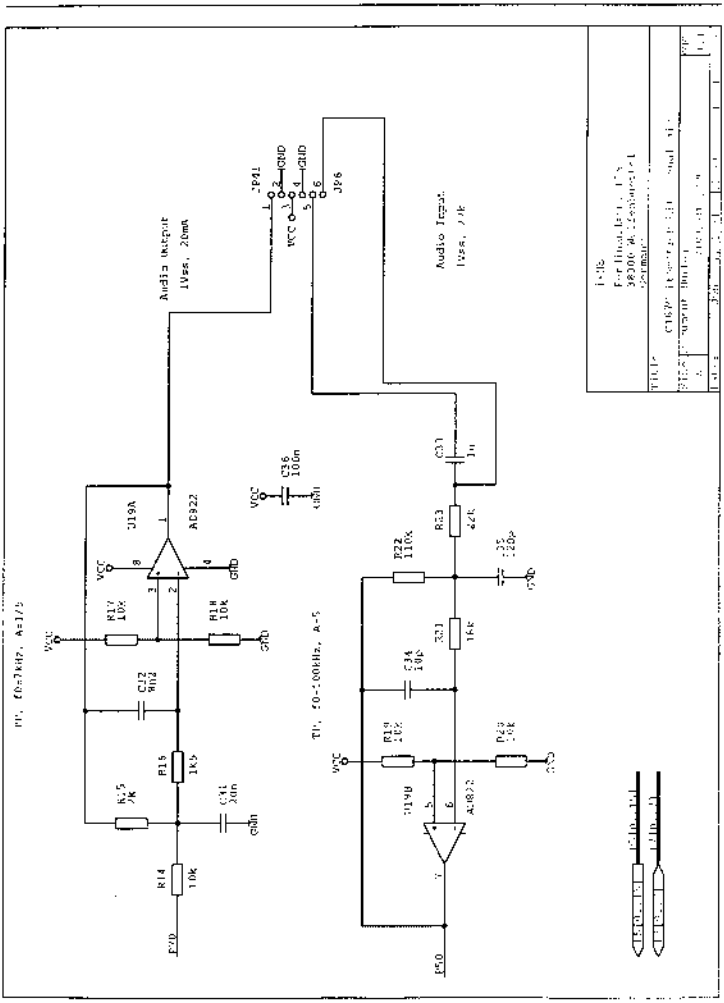
3.7.1 Component placement



3.7.2 Conector 160pin



3.7.3 Conector analog



3.8 Connector Pinnings

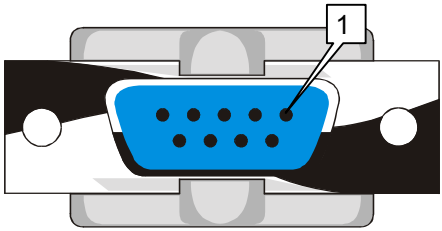
3.8.1 Power Supply

The C167CR EvaBoard has a standard DIN 45323 connector for power supply.

Center pin	= +Vbatt
Outer contact	= GND

3.8.2 RS232 Interface

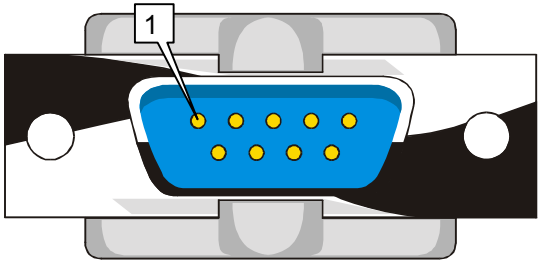
The RS232 interface has a standard female 9-pin D-Sub connector.



Pin 1 = n/c
Pin 2 = TxD (output)
Pin 3 = RxD (input)
Pin 4 = DTR (logic-low active hardware reset input)
Pin 5 = GND
Pin 6 = n/c
Pin 7 = n/c
Pin 8 = n/c
Pin 9 = RI (permanent logic-low)

3.8.3 Network Interface

The C167CR EvaBoard uses a standard male 9-pin D-Sub connector for the network. The pinning conforms to the CiA recommendation. Of course this only applies to CAN networks.



Pin 1 = n/c
Pin 2 = CAN-L
Pin 3 = GND
Pin 4 = n/c
Pin 5 = n/c
Pin 6 = GND
Pin 7 = CAN-H
Pin 8 = n/c
Pin 9 = Bus power supply (input)

3.8.4 Analog Signal Interface

The on-board analog input/output amplifiers end on a 6-pin strip connector.

Pin 1 = analog output
Pin 2 = GND
Pin 3 = +5V
Pin 4 = GND
Pin 5 = AC coupled input
Pin 6 = DC coupled input

3.9 GAL Listing

```
PAGE0 = (/A23 & /A22 & /A21 & /A20 & /A19 & /A18);
PAGE1 = (/A23 & /A22 & /A21 & /A20 & /A19 &  A18);
PAGE2 = (/A23 & /A22 & /A21 & /A20 &  A19 & /A18);
PAGE3 = (/A23 & /A22 & /A21 & /A20 &  A19 &  A18);

/_CSROM = 'PAGE1'
      + 'PAGE0' & /RAMLO & /(/_CSLED + /_CSDIP + /_CSPRB);

/_CSRAM = 'PAGE2'
      + 'PAGE0' &  RAMLO & /(/_CSLED + /_CSDIP + /_CSPRB);

/_CSLED = 'PAGE0' & /_CS1 &  A10 & /_WRL;

/_CSDIP = 'PAGE0' & /_CS1 &  A10 & _WRL & _WRH;

/_CSPRB = 'PAGE0' & /_CS1 & /A10;

/_WRH   = /_BHE & _RIN;


RAMLO  = /_RAMLO +         ('PAGE1' & /_CS1 & /_WRL &  A10);

_RAMLO = /RAMLO  + /_RST  + ('PAGE1' & /_CS1 & /_WRL & /A10);


*PAL

TYPE=GAL20V8

*PINS

A10=2,
A18=3,
A19=4,
A20=5,
A21=6,
A22=7,
A23=8,
_CS1=9,
_WRL=10,
_BHE=11,

_CSROM=15,
_CSRAM=16,
```

```
_CSLED=17,
_CSDIP=18,
_CSPRB=19,
_RST=14,
_RIN=23,
RAMLO=20,
_RAMLO=21,
_WRH=22;
```

3.10 Electrical Characteristics

3.10.1 Power supply

The I+ME C167CR EvaBoard needs a DC power supply:

U = 7 - 20 DC
I ~ 300 mA (without external load)

3.10.2 RS232 Interface

The RS232 interface conforms to the standard ± 10V signal levels.

3.10.3 Analog Signal Interface

The analog input/output amplifiers on the I+ME C167CR EvaBoard have the following characteristics:

Output:	~ 1 V_{ss} symmetrical to $V_{cc}/2$, < 20mA
AC Input:	~ 1 V_{ss} on > 22kΩ
DC Input:	~ 1 V_{ss} on > 22kΩ, referenced to $V_{cc}/2$

3.10.4 External CPU Signals

All signals of the C167CR are available on a standard DIN connector. These signals are not buffered! Therefore beware of excessive loads. In order to ensure proper operation with external hardware you should use buffers as near as possible to the C167CR EvaBoard. Keep in mind that these buffers will influence the bus timing.

Your I+ME C167CR Eva Board sample program.

4 Samples

4.1 System Programming

This chapter describes the basic configuration of the C167 in order to work with the I+ME C167 EvaBoard.

SYSCON

STKSZ	user specific
ROMS1	user specific
SGTDIS	user specific (0 = enable recommended)
ROMEN	user specific
BYTDIS	0 = enabled
CLKEN	user specific
WRCFG	1 = #WRL/#WRH

BUSCON0

RDYEN0	0 = ready disabled
BUSACT0	1 = bus enabled
ALECTL0	0 = lengthening disabled
BTYP	2h = 16-bit non-mux
MTTC0	1 = 0 Tri-state waitstates
RWDC0	1 = Read/Write delay disabled
MCTC	Dh = 2 waitstates (tacc<150ns)

The bus timing shown is rather conservative. When using fast memories (RAM instead of EPROM) you may try 0 waitstates. Anyway you can set up an ADDRSEL especially for the RAM area and make it faster than the EPROM area. You don't need to activate a CS signal for this purpose!

In order to access the periphal functions LED-array/Dip-switch/PrBB via CS1 we suggest the following setup:

ADDRSEL1

RGSAD	03Fh = 03F000h - 03FFFFh
RGSZ	0h = 4 kBytes

BUSCON1

CSWEN1	1 } = Read/Write chip select
CSREN1	1 }
RDYEN1	0 = Ready disabled
BUSACT1	1 = bus enabled
ALECTL1	1 = lengthening enabled
BTYP	1h = 8-bit mux
MTTC1	0 = 1 Tri-state waitstate
RWDC1	1 = delay disabled
MCTC	Ch = 3 waitstates

Again, this is a very conservative timing. Depending on the PrBB a faster timing may be possible. Especially if you don't anticipate a PrBB at all you can make the timing as fast as you like since the LED-array and dip-switch are not time critical. Even the bus type doesn't matter in this case. So if you stay with a 16-bit non-mux bus you save the waitstates during bus type switching.

4.2 Default CAN sample

This default sample is delivered within the EEPROMs if you bought the standart edition. If you got the smal edition you must downloading the "sample.h86"-file into RAM. At both situations (RAM/EEPROM) the programs funktion is the same, only the startup situation is an other.
The program simulate an CAN member on differend adresses. You can send and receive one byte massages depend on the state of the DIP-switches. The output of send or receive massage can be display on the LED-array.

For detailed information look up into the source code you found in the next chapter.

The function of this programs DIP-switches is the following:

DIP switches:

8 = 1	PC control→ The PC controls the Eva board. The other switches have no function.
8 = 0	Stand-alone → only now the DIP switches will control the board in the following way.
7 = 1	display rx data on the LEDs.
7 = 0	display tx data on the LEDs.
6,5	*Receive message type. Data byte is displayed on LED if DIP7 = 1.*
6,5 = 00	receive messages is standard ID 000h, data length 1.
6,5 = 01	receive messages is standard ID 001h, data length 1.

6,5 = 10 receive messages is extended ID 0000000h, data length 1.
6,5 = 11 receive messages is extended ID 0040000h, data length 1.

4,3 transmit message type. Data byte is displayed on LED if DIP7 = 1 Repeat time = see transmission speed control DIP2,1
4,3 = 00 transmit messages is standard ID 000h, data length 1.
4,3 = 01 transmit messages is standard ID 001h, data length 1.

4,3 = 10 transmit messages is extended ID 0000000h, data length 1.
4,3 = 11 transmit messages is extended ID 0040000h, data length 1.

2,1 Transmission speed control. Defines the time up to a new transmission.
2,1 = 00 fast
2,1 = 01 medium
2,1 = 10 slow
2,1 = 11 very slow

4.2.1 Default sample source code

```
/***********************************************

          program for ExtendedCAN C167CW EVA

                   I+ME 1994

         Written for C-166/167 Compiler
          Boston System Office/Tasking
              Tasking Software BV
                  P.O. Box 899
           Amersfoort, the Netherlands
***********************************************/

#include <reg167.h>
#include "167Creg.h"

#define XP0INT 0x40

#define PrBB   ( (volatile char far*)0x3F000ul)
#define LED     (*(volatile char far*)0x3F400ul)  /* Write
only ! */
#define DIP     (*(volatile char far*)0x3F400ul)  /* Read
only ! */

#define QUARTZ   40    /* 20 in reality; CW-version runs
without prescaler ! */


/***********************************************
       Global variables
***********************************************/
unsigned char tx_value,
              Show_Received_Data;

unsigned char Transmitter, Trans_Old, Trans_New,
              Receiver,    Receiv_Old, Receiv_New;

/***********************************************
       Protocolspecific variables and constants
***********************************************/

/* Setup for 75 kBit/s, 16 MHz */
#define cBTR0            0xC5
/* CAN-Bus-Timing-Reg. 0 Setup Value */
#define cBTR1            0x6D
/* CAN-Bus-Timing-Reg. 1 Setup Value */


/*---------------------------------------------*/
/* Function: init_can                          */
/*---------------------  ----------------------*/
void init_can (void)
  {
```

```
    unsigned char i;

    C167CControl      = 0x41;
    for (i=0; i<0xFF; i++) {}

    C167CGlobalMaskStandard0 = 0xFF;/* prepare masks */
    C167CGlobalMaskStandard1 = 0xFF;
    C167CGlobalMaskExtended0 = 0xFF;
    C167CGlobalMaskExtended1 = 0xFF;
    C167CGlobalMaskExtended2 = 0xFF;
    C167CGlobalMaskExtended3 = 0xFF;

    for (i=0x00; i <= 0xEF; i++)
    {
      C167CMessage1 [i] = 0xFF;       /* init memory   */
    }

    C167CBitTiming0 = cBTR0;
    C167CBitTiming1 = cBTR1;

    C167CMessage15 [0] = 0x55;         /* Msg15 = "invalid"
*/

    C167CMessage1 [0] = 0x55;     /* setup Receiver mailbox
*/
    C167CMessage1 [1] = 0x55;     /* invalid, RX Int active
*/
    C167CMessage1 [2] = 0x00;     /* ID 3        */
    C167CMessage1 [3] = 0x60;
    C167CMessage1 [4] = 0x00;
    C167CMessage1 [5] = 0x00;
    C167CMessage1 [6] = 0x10;
    C167CMessage1 [0] = /*0x99; activate mailbox */

    C167CMessage2  [0]  =  0x55;        /*  setup  transmitter
mailbox */
    C167CMessage2  [1]  =  0x55;        /*  invalid,  no  Ints
*/
    C167CMessage2 [2] = 0x00;     /* ID 0        */
    C167CMessage2 [3] = 0x00;
    C167CMessage2 [4] = 0x00;
    C167CMessage2 [5] = 0x00;
    C167CMessage2 [6] = 0x18;
    C167CMessage2  [0]  =  0x95;        /*  activate  mailbox
*/

    C167CControl = 0x02;   /* activate CAN-chip  */
  }

/*------------------------ --------------------*/
/* Function: Receive  (Interrupt service routine)*/
/*-----------------------------------------------*/

interrupt (XP0INT) using ( CC_regs )
void Receive (void)
{
```

```
/* ****************************************** */
/* * Protocolspecific Receive Routine          * */
/* ****************************************** */

  if (C167CInterrupt == 0x03) /* Rx Int from Msg 1 */
  {
    C167CMessage1 [1] = 0xFD;             /* Reset "NewDat"
bit */
    C167CMessage1 [0] = 0x99;             /* Reset "IntPnd"
bit */
  }
  if (Show_Received_Data) LED = C167CMessage1[7];
}

/*-----------------------------------------------*/
/* Function: Setup_Transmitter                   */
/*-----------------------------------------------*/
void Setup_Transmitter (void)
{
  unsigned char Type;

  Type = ((DIP & 0x0C) >> 2);            /* Dipswitch 3 & 4
*/

/* ****************************************** */
/* * Protocolspecific Message Definition For   * */
/* * Transmitter                               * */
/* ****************************************** */

  Trans_New = Type;
  if (Trans_New != Trans_Old)
  {
    C167CMessage2 [0] = 0x55;        /* deactivate message
*/
    Transmitter = Trans_Old = Trans_New;
    if (Transmitter > 0x01)    /* Extended ? */
       C167CMessage2 [6] = 0x1C;
    else C167CMessage2 [6] = 0x18;
    C167CMessage2 [3] = (Transmitter & 0x01) << 5;
    C167CMessage2 [0] = 0x95;/* activate message */
  }
}

/*-----------------------------------------------*/
/* Function: Setup_Receiver                      */
/*-----------------------------------------------*/
void Setup_Receiver (void)
{
  unsigned char Type;

  Type = ((DIP & 0x30) >> 4); /* Dipswitch 5 & 6 */

/* ****************************************** */
/* * Protocolspecific Message Definition For   * */
/* * Receiver                                  * */
/* ****************************************** */
```

```
  Receiv_New = Type;
  if (Receiv_New != Receiv_Old)
  {
    Receiver = Receiv_Old = Receiv_New;
    C167CMessage1 [0] = 0x55;      /* deactivate mailbox */
    if (Receiver > 0x01)         /* Extended ? */
       C167CMessage1 [6] = 0x14;
    else C167CMessage1 [6] = 0x10;
    C167CMessage1 [3] = (Receiver & 0x01) << 5;
    C167CMessage1 [0] = 0x99;                              /*
activate message with int */
  }
}

/*-----------------------------------------------*/
/* Function: Setup_LED                           */
/*-----------------------------------------------*/
void Setup_LED (void)
{                          /* Dipswitch 7      */
  Show_Received_Data = (DIP & 0x40) >> 6;
}

/*-----------------------------------------------*/
/* Function: Delay                               */
/*-----------------------------------------------*/
void Delay (void)
{
   const unsigned int dly[] = { 1, 10, 50, 100 };

   unsigned int Max;


   /*
   ** Use T3 as timer
   */

   T3CON =   0x0006     /* T3I  : prescaler 1024 */
           | 0x0000     /* T3M  : timer  */
           | 0x0040     /* T3R  : run */
           | 0x0000;    /* T3UD : up */

         T3 = 0;

         Max = ((35154 / 40) * QUARTZ) / 100 * dly[DIP &
0x03];

         while (T3 < Max);

}

/*-----------------------------------------------*/
/* Function: Transmit                            */
/*-----------------------------------------------*/
void Transmit (void)
{
```

```
/* ****************************************** */
/* * Protocolspecific Transmit Routine        * */
/* ****************************************** */

  C167CMessage2 [0] = 0x55;    /* deactivate mailbox */
  C167CMessage2 [7] = tx_value++;   /* put data */
  C167CMessage2 [0] = 0x95;           /* activate mailbox
*/
  C167CMessage2 [1] = 0xE5;           /* transmit request
*/

  if (!Show_Received_Data)
  {
    LED = tx_value;
  }
}

/*-----------------------------------------------*/
/* Function: Init                                */
/*-----------------------------------------------*/
void Init (void)
{

  BUSCON3=0;
  BUSCON4=0;
  ADDRSEL1 = 0x03F0;     /* 4 kB block in 1st 256 kB page
03F000h - 03FFFFFh */
  BUSCON1  = 0xC65C;    /* 11*0*11* 01011100
                                        //   ||   |   ||
||||++++-- 3 wait states
                                        // || | ||  |||+-
----- RD/WR delay disabled
                                        // || | ||  ||+--
----- Memory Tri-State delay enabled
                                        // || | ||  ++---
----- Bus type: 8bit mux
                                        // || | |+-------
----- ALE lengthning enabled
                                        // || | +--------
----- Bus active
                                        // || +----------
----- Ready disabled
                                       // ++--------------
--- RD/WR chip select */


   XP0IC = 0x48;                      /* Enable interrupt -
ILVL = 2, GLVL = 0. */

   IEN   = 1;

   tx_value = 0;
   Receiv_New = Trans_New = 0x55;
   Receiv_Old = Trans_Old = 0xAA;
   init_can ();
   Setup_Transmitter ();
```

```
   Setup_Receiver ();
   Setup_LED ();
}

/*---------------------------------------------*/
/* Main program                                */
/*---------------------------------------------*/

void main (void)
{
  Init ();

  while (1)
  {
    Setup_Transmitter ();
    Setup_Receiver ();
    Setup_LED ();
    Transmit ();
    Delay ();
  }
}
```

Common problems and how to solve them. How to get in touch with our after-sales support experts if you so desire.

5 Techn. Support

5.1 What to do if you have problems:

First and foremost, please read **Installation** very closely and make sure that you performed your installation exactly as described.

For developers:
The Key is often used in developing environments in combination with the API and/or DLL. If the PcCANControl software is functioning properly, then there is no problem with general CAN access. You should check your usage of the API of DLL.

If the PcCANControl software is not functioning, please consult the list of common problems below and their possible solutions.

5.2 Solutions for all parts ...

General problem to install and run the hardware:

Under NT: Check if the carddriver is started under the device manager. Status "started"
Under 9x: Check if the card is correct listed under the device manager, no yellow spot on it.

The system crashes after choose the hardware at PcCANControl:

The selected memory area is not free, or the selected interrupt is being used by another application. Make sure that no conflicts exist on your system. If you get a blue screen under Win NT it is most likely that a memory or IRQ conflict is occur.

PcCANControl launches correctly, but CAN access is not possible:

Check the transceiver cable and CAN connectors. If a connector is not correctly plugged, connect it right. If the sub-min-D connectors at the end of the cables are hot, remove the Key and call I+ME.
*If you use **higher baudrates**, remember to protect the end of line with **resistors** [120Ohm]*

5.3 Solutions for PCNetboard II and III

The card is plugged in but no contact was possible:

Under NT: *Compare the ressources in the device manager with the jumper on the board. Both must be equal! You can change it with help of the tool "LevelX PcNet...Driver Configuration" under Software installation/deinstallation.*

Under Windows 9x*: Compare the ressources in the device manager with the jumper on the board. Both must be equal! You can change it with device manager.*

5.4 Solutions for PCMCIA

The system crashes after the installation of the Key client:

The selected memory area is not free, or the selected interrupt is being used by another application. Refer to Installation.

The system crashes when the Key is inserted or removed:

The selected memory area is not free, or the selected interrupt is being used by another application. Refer to Installation.

Windows 95/98
Windows will not launch after the installation of the client and services:

Make sure that there is a memory area reserved in the SYSTEM.INI file. The selected memory area is not free, or the selected interrupt is being used by another application. Refer to Installation.

Windows 95/98
The acknowledge sound after inserting or removing the Key is sounded twice:

There are two services installed on your system. One can be found in the CONFIG.SYS and the other in the device manager. In the device manager, under the topic PCMCIA, you can disable one of the services.

Windows 95/98

The Key is not inserted and PcCANControl starts with the message: I+ME CARD not available or drive not ready.

Insert the Key.

The Key is inserted and PcCANControl starts with the message: I+ME CARD not available or drive not ready.

*Some aspect of the resource allocation is wrong. Refer to **Installation** for more information.*

Windows NT
After installation the PC is restart via Powerdown?

In most cases the reset of the PCMCIA hardware is not correct done, so it is neseccarry to start with powerdown.

Windows NT
Is the PCMCIA-Service started correct?

You can check it under
"start/control/systemconfiguration/
PC-driver(PCMCIA)"
if the service is correct started. The driver must be visible with its two sockets.
The I+ME PCMCIA-key should be visible under one slot.
trouble: No service is started!
You have to start the PCMCIA service under "Devices" on the same directory.
The "starttype" should be "automatic". Change the state if necessarry.
Start the driver manual in this case.

trouble: The service is started but PCCANcontrol starts with ...
"Error-DriverOpen: CLIxpm1.exe not started or PCMCIA card not inserted?"

a) Check if card is inserted into one slot.
note: under WinNT you don't exchange cards under power !!!
b) Start the Client software "CLIxpm1.exe".
You should start the programm "Initdrv.bat" you found under...
//WinNT/system32/drivers/ime/lxn4pm1j/...
If trouble occure:
Install the driver again and restart. You can overwrite the driverinstallation.

Are your BIOS settings correct for WinNT?

The BIOS settings "Plug&Play" must be on state "off" - WinNT is not a plug&play OS!
Correct the setting if necessarry.

5.4.1 Non-Supported PCMCIA Drives

In the current version there are some PCMCIA drives which are incompatible with the *PCMCIA Key*. Today, as a general rule, some PCI connected drives result in compatibility problems. The following is a list of known incompatible drives and any laptops which are known to use these drives.

Drive	Known Laptops Using Drive
Cirrus Logic PCIC compatible	SIEMENS NIXDORF Scenic Mobile 700 COMPAQ Armada

5.5 Solutions for NetPorty II

You start PcCANControl at parallel port but the connection doesn't work correct:

For parallel port usage it is necessary to have the EPP-loader-firmware (BOOTLOAD.H86) in the NetPorty II -Flash-EPROM. All Porty's are delivered with this loader!

Firmware is correct but the NetPorty II doesn't work under parallel port:

Use the right EPP mode on BIOS. It can be only one of the EPP modes work with one Win-System together. In past we find out that EPP 1.7 is necessary for Win9x and EPP 1.9 is necessary for Win NT. Please try out the right mode.

Using a printer and NetPorty II under Win NT:

If you use NetPorty II under NT note that you are not able to print!

Using a printer and NetPorty II under Win 9x:

If you use NetPorty II under Win9x the normal printer driver is active but it can make problems while printing with the active driver.! One way to solve this problem is to load the firmware into the flash of the NetPorty II.

You have overwrite the delivered EPP-Loader by your own firm*ware.*

In this case it is possible to download the EPP-firmware again into the NetPorty II -Flash. This software is a Win32-Console-Application and runs under Win95 and WinNT.

How to use:

- *Connect your NetPorty II with one serial port.*

- *If you do not use COM1 please change it into the file L.BAT.*
- *Start L.BAT.*
  If you get an error messages like "BSL: no response from the target hardware" it is possible that you have used the wrong COM port.
- *If you get the message below "-- COMx Loading bootload.h86..Ready" without any error message in the next line, all activities are correct finished*
- *Leave the software with pressing the ESC key, sometimes Alt X does not work.*

***Info**:*
The program LDFSER.EXE is a loader that burns firmware in the Porty-Flash and also it is a serial monitor The serial parameter are 115200,8,n,1. I+ME also can deliver a loader to start firmware in the RAM and a little firmware demo (Keil-C). Please contact us if you want to develop your own firmware.

5.6 Solutions for PCI-Intellican

Problems under Windows 2000:

The SW installation is successful and the card is inserted correct but no commuication is possible

- Try to plug in the card into an other PCI port.

The SW installation is successful and the card is detect correct but no commuication is possible

General problems with ACPI exist, one of the next cases can help.

- Disable the “Assign IRQ for USB” (similar Text possible) in BIOS. The BIOS map this IRQ to a common IRQ, the IRQ for USB is set by Windows 2000 special, so the funktion of USB has no restrictions. **Note**: These case is not possible by all PC’s.
- Force a Window 2000 conform mapping by blocking the PCI-plug&play mechanism into the BIOS settings: Set under “PNP/ISA Configuration” the “Resource controlled by manual” funktion, after this set all but one IRQ’s to “Legacy ISA”.

Tips to last point:

✓ Check the IRQ under Windows 2000 device manager. Choose “view” and then “Recource to connections”. There is a list of all IRQ’s of PCI/ISA devices. All PCI-devices should be set to one IRQ (by activ ACPI !). You should choose the IRQ with the most entries in list. Use these IRQ to set him into the BIOS to PCI/ISA PNP.

- Disable of the ACPI interface.
  Note:
  Depends on BIOS type (AMI, AWARD, Compaq,..)
  Depends on ServicePack you use.
  General:
  If you will disable the ACPI mechanism you **must first** disable it under Windows 2000 and then disable it into BIOS. If not, Windows 2000 will never restart correct.
  If you have install the SP1 from Win2000 this BIOS settings are not necessary.

  For more infos please contact our
  After Sales Service.

  In this mode our PCI-card works, if the IRQ is shared or not.

If you encounter difficulties which are not discussed in the manual, or if you need more help than is offered in **Installation** and **Troubleshooting**, please call our after-sales service. Our experts will do their best to solve whatever problem you might have.

After-sales service

✉ I+ME ACTIA GmbH
Rebenring 33
D-38106 Braunschweig.

☎ ++ 49 (531) 38 701 38
Fax : ++ 49 (531) 38 701 88

e-mail : info@ime-actia.de

6 Glossary

Glossary

ACPI
Advanced Configuration and Power Interface.
Managing the Powersystem of you computer. New since Windows 2000.

CAN
Controller Area Network
Fieldbus network system useful in EMV critical environment

-BCAN
Basic-CAN
The used chip has only a small buffer structure. All filter has to be defined in software and have to managed by the application.

-ECAN
Extended-CAN
The used chip supports extended identifiers (CAN 2.0B / 29bit IDs)

-FCAN
Full-CAN
The used chip has an internal buffer structure with mailbox architecture and supports Remote Frames.

-SCAN
Standard-CAN
The used chip supports standard identifiers (CAN 2.0A / 11bit IDs)

BIOS
Basic Input Output System
An abbreviation for Basic Input / Output System. A set of instructions/routines stored in ROM. These routines work closely with hardware devices (memory chips, disk drives and monitor) to input and output interrupt requests indicating when a device is ready to accept or send data.

DPRAM
Dual Ported Random Access Memory.
The data for communication between PC and CAN-hardware are exchange via a DPRAM.

I/O
An abbreviation for Input Output. Refers to the sending (input) and receiving (output) of data through an 110 channel in the CPU. *Example:* The keyboard inputs data to the 110 channel in the CPU that in turn is output to the monitor.

IRQ
An abbreviation for Interrupt Request. A signal sent by a device and routed through the BIOS indicating when a device is ready to accept or send data.

PCMCIA
Personal Computer Memory Card International Association. A trade association of leading hardware and software vendors, established to adopt a set of standards pertaining to adapter slots and PC cards for portable PC accessories.

Slot / Socket
A receptacle on a micro, portable, laptop or palmtop computer that is used to insert and operate PCMCIA PC Cards. Also referred to as a *Socket.*

Socket-Controller
A PC system hardware component that manages the operation of PCMCIA sockets in conjunction with system software.

Upper-Memory
Memory area within the PC address space between 640 KB and 1 MB. This area is used by hardware devices like graphics controller. The DPRAM of the CANcard is located in the Upper Memory.

For more information's ...

7 Literature

[1] **PCCANControl User manual**
I+ME Actia documentation (available on WEB)

[2] **LevelX User Manual**
I+ME Actia documentation (available on WEB)

[3] **CAN - Controller Area Network**
Grundlagen und Praxis
Hüthig GmbH, Heidelberg
ISBN 3-7785-2263-7

[4] **SAB 80C167CR User Manual**
Siemens AG

[5] **SAB C167CR Description of the On-chip CAN-Module**
Siemens AG

[6] **CiA**
DS 102-1CAN in Automation e.V
www.can-cia.com