@C5xx

Operating manual

Edition date/Rev. date:	25.08.2006
Document no./Rev. no.:	01
Software version:	-
File name:	@C5xx-TRS-V-BA-GB-0000-01
Author:	WIJ/HIE

TR-Systemtechnik GmbH
Eglishalde 16
D-78647 Trossingen
Germany
Tel. +49 - (0) 7425 / 228-0
Fax +49 - (0) 7425 / 228-34

Imprint

TR-Systemtechnik GmbH
D-78647 Trossingen
Eglishalde 16
Tel.: (++49) 07425/228-0
Fax: (++49) 07425/228-34
info@tr-systemtechnik.de
http://www.tr-systemtechnik.de/

Right of modification

We reserve the right to modify the technical information contained in this document without prior notice in the interests of ongoing improvements to our products and documentation.

Printing

This manual was produced on a personal computer using MS-Word for Windows. The text was printed in Arial type.

Fonts

Italics and **bold** type are used for the title of a document or to emphasize text passages.

Passages written in `Courier` show text which is visible on the display as well as software menu selections.

"< >" refers to keys on your computer keyboard (e.g. <RETURN>).

Note

Text following the "NOTE" symbol describes important features of the respective product.

Copyright Information ©

MS-WORD is a registered trademark of Microsoft Corporation.

Literature

Revision History

Note
The cover of this document shows the current revision status and the corresponding date. Since each individual page has its own revision status and date in the footer, there may be different revision statuses within the document.

Drawings which can be in the appendix have their own revision history.

Document created: 23.11.2004

	Changes	Date
01	presentation of control-register improved, new features of FPGA added	25.08.2006

Contents

1. Functional description

PC104 Master @C500, @C550 and 570 Power Supply PCB
The @C5xx base modul consists out of following logical parts:

Power supply
Input voltage: 20-28V
Output voltages: 3,3V / 3,5A
5V / 3,5A
12V / 2,5A
Address decoder
@IO system bus / supervisor chip / UPS

Supervisor chip
Temperature / van speed / voltages

Watchdog (default not active)

FPGA interface system bus (TRS deliver C-Functions) programmable interrupt timer.

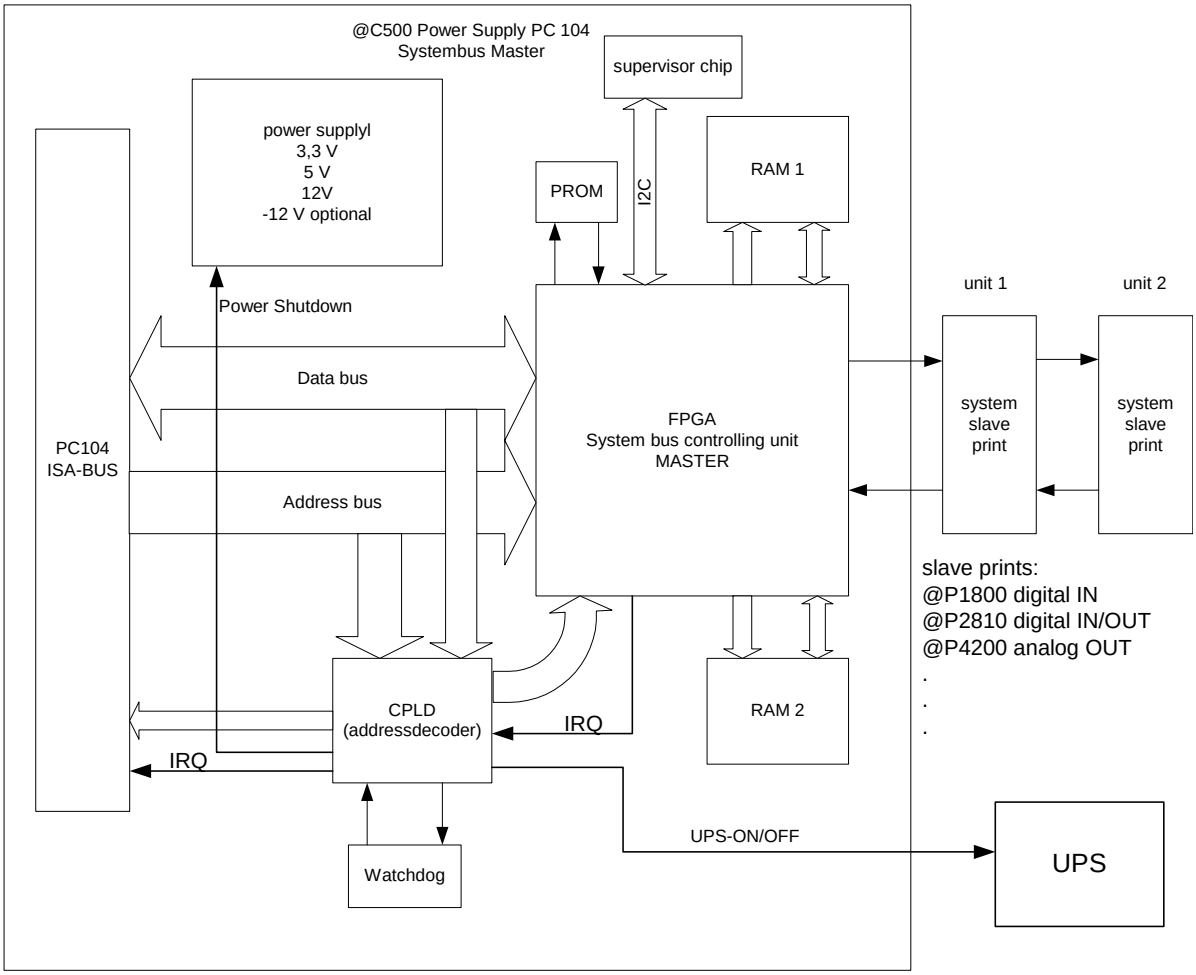

Figure 1: functional parts of @C500/570 power supply pcb

1.1 optional variations

Power supply -12V / 50mA

- FPGA PROM:
  Normally the FPGA must be loaded after each power on by software from the CPU over the ISA-BUS.
  With PROM on board this is not necessary. But updates of the FPGA firmware is only by changing the PROM possible. In other case it must only changed one file on the harddisk or compact flash of the PC104-CPU.
- The IRQ channel could be set by instrumentation or DIP switches.
- Following channels are possible:
  > IRQ3
  > IRQ5
  > IRQ7
  > IRQ9
  > IRQ10
  > IRQ11
  > IRQ12
  > IRQ15

Sources of IRQ:

1. Supervisor chip I²C-Interrupt
2. FPGA system bus (parity error, or timeout of data transmission)
3. FPGA timer

After IRQ you must delete the cause manually.

2 Address range

Real basis address: 0xD0000 (C-prog: 0xD000000) used address space 4 kByte

Table 1: Address range

Address space	Absolute byteaddress	Description
0xD0000 Memory area green area only 16-bit access allowed	0xD0000 0xD00FF	send-page
	0xD0100 0xD01FF	receive-page
	0xD0200 0xD02FF	dual memory send-page (only useable if no transmission runs on system bus)
	0xD0400 0xD0406	command register
	0xD040C	optional hardware-watchdog, trigger with write access on these address (random data) it exist a second watchdog (twd=200ms) in the FPGA which will be triggered automatically by each RAM_SWITCH
I2C-interface	0xD0410	I2C clock prescale register
	0xD0412	I2C control register
	0xD0414	I2C transmit command register
	0xD0416	I2C receive status register
0xD07FF		
0xD0800 yellow area only byte access allowed 0xD0FFF	0xD0800	write access with random data start programming cycle of FPGA
	0xD0802	IRQ-status register; IRQ-Sources FPGA System bus, LM81 supervisor chip (only with CPLD V03 after 24.3.2003)
	0xD0FE0 0xD0FF0	Power-shutdown, power supply will be restarted with write access data 0x10. If valid 24V external source, than power supply runs automatically again, but if 24V is disconnected and only UPS runs the system will not start again. switch on UPS with byte access binary data xxxx xxx1, switch off xxxx xxx0 (x=any value) Status of UPS could be read back at this address

Calculating byte address offsets to word address offsets in C progamming language:

Move byte address offset 1Bit right (Division with 2)
Neccessary for green memory area. 16-Bit word access in C makes word offsets neccessary, in other programming languages it should be similar.

3 Control register of system bus master

Name	Data-Bit	Word offset Address	Description	Annotation
ANZ_BYTES	0-7	0x200	write to this register number of bytes in systembus -1	
	8-15		free	
ST_NORM	0		change from initialisation to normal mode	
BUSY	1		set busy if processor read/write systembus memory	not necessary, no effect at time
STOP_MOV	2		prevent new transmission on systembus (trigger event RAM_SWITCH)	
RESET_MASTER	3		RESET of @ctiveIO-System	will be automatically deletet after execution
FREE_RUN	4		continious automatic mode of systembus transfer	to read/write systembus memory stop this mode
WD_DISABLE	5		disable watchdog if set	
IRQ_DEL	6		delete IRQ of System	should be executed in software interrupt routine
SECURE_MODE	7	0x201	TIMEOUT cause rapid reset of systembus slaves	
SLOW_MOTION	8		slow down the systembus clock from 5 MHz to 2 MHz	
FAST_ACT	9		change mode of systembus to get more actual values	version 10 or higher
EN_TIMEOUT_PARITY_IRQ	10		Interrupt Enable for TIMEOUT and Parity	version 10 or higher
reserved	11			
reserved	12			
reserved	13			
reserved	14			
reserved	15			
			Toggle Register (data bit will be inverted with each writeaccess)	
			Set only Register, will be automatically deletet after execution	

Table 2: control register 0x200 and 0x201

Control register of system bus master

		Word offset		
Name	Data-Bit	Address	Description	Annotation
INIT	0		manual initialisation of systembus slaves (mov_dat necessary)	
L_FPD	1		start/stop firmware update slave	not implemented
RESET_SLAVE	2		reset of systembus slaves	
RAM_SWITCH	3		change access on rams and start normally transmission on systembus	
MOV_DAT	4		move data on systembus without actualisation of IO-Register	
FST_TRAN	5		automatical initialisation of systembus slaves (RESET, INIT, MOV_DAT)	
DATA_CHG	6		data exchange on systembus slaves	
STATUS	7	0x202	if set, next transmission on systembus exchange status/parameter information	
READ_INP	8		copy systembus input values in register	version 10 or higher
reserved	9		reserved	
EN_IRQ_ERRORS	10		interrupt enable for TIMEOUT and P_ERROR	version 10 or higher
reserved	11			
reserved	12			
reserved	13			
reserved	14			
reserved	15			

Set only Register, will be automatically deletet after execution

Table 3: control register 0x202

Name	Data-Bit	Word offset Address	Description	Annotation
STATUS				
P_ERROR	0	0x203	Parity error on systembus data receive process	reset bit write 0x1 to Register
TIMEOUT	1		TIMEOUT of systembus data transmission	reset bit write 0x2 to Register
M_WARTEN	2		no transmission on systembus	
RAM_CHANGED	3		ram access changed (after manuall ram_switch)	
PROZ_RAM1_ACCESS	4		shows actual access to ram	1=access to ram1, 0=access to ram2
WD_TIMEOUT	5		shows watchdog timeout, write to this bit to trigger watchdog manually	reset or manually trigger -> write 0x20 to register
IRQ_STATUS	6			
reserved	7			
VERSION	8-15		FPGA version	
CLK_DIFF	0-7	0x204	difference between sended and received clock´s	on TIMEOUT values >0 possible
TIMER_PERIOD	8-15		timer period register for timer1 multiplier 10µs	FPGA Version <14 multiplier 100µs
TIMER1 (FPGA version 14 or higher)				
TIMER_ON	0	0x205	enable bit for timer1	enables timer for IRQ and automatically systembus transfer
TIMER_IRQ	1		signal of finished timer cycle	after read access, TIMER_IRQ will be deleted
TIMER_VAL_L	2-7		actual timer count (multiplier 125 ns)	
TIMER_VAL_H	8-15		actual timer count (multiplier 10 µs)	

Table 4: control register 0x203, 0x204, 0x205

Name	Data-Bit	Word offset Address	Description	Annotation
TIMER2 (FPGA version 16 or higher)				
SBUS_CYCLE	0-7	0x206	actual value duration of systembus cycle in µs	
MAX_CYCLE	8-14		stored max value of SBUS_CYCLE (only bit 1-7)	reset with RESET_MASTER
OVERFLOW_MAX_CYCLE	15		MAX_CYCLE value is bigger than 255µs	if set, ignore value of MAX_CYCLE
SYSTEMBUS Info				
SBUS_REPEATS	0-7	0x207	if TIMER_ON then P_ERROR or TIMEOUT increment this counter	TIMER_ON -> single P_ERROR or TIMEOUTS were not shown on it´s STATUS-Bit
reserved	8-15			

Table 5: control register 0x206, 0x207

Name	Data-Bit	Byte offset Address	Description	Annotation
STATUS				
IRQ_STAT_FPGA	0	0x802	IRQ status FPGA	lowactive, 0=interrupt
IRQ_STAT_LM81	1		IRQ status LM81 supervisorchip	lowactive, 0=interrupt
reserved	2			
reserved	3			
CPLD_VERSION	4-7		CPLD version	
POWER_SHUTDOWN	4	0xFE0		
UPS_ON	0	0xFF0	switch on for external UPS	
reserved	1			
reserved	2			
reserved	3			
VOLTAGE_24_OK	4		24V power supply OK, (1 >19V ; 0 <17,8V)	only with external UPS, else LEFT_CON_IN0
BATTERY_OK	5		battery OK (1 > 10V; 0 <7,8V)	only with external UPS, else LEFT_CON_IN1
LEFT_CON_IN2	6		digital Input on left connector (max 5V)	
reserved	7			

Table 6: byte register 0x802, 0xFFE, 0xFF0

4 UPS

If your system is equipped with an UPS you must them activate by software.

To activate: write on address **0xD0**FF0 data 0x1 (UPS_ON).

Now if external 24V supply voltage disappear, the system will be supplied for 45 seconds by the UPS. Then it will be switched off. You can detect this by reading on **0xD0**FF0 data bit 4 (VOLTAGE_24_OK). If zero, then no valid 24V supply -> finish your applications

5 Interrupt timer (FPGA version V14 or higher)

The FPGA interrupt timer trigger periodically an IRQ (if enabled) or a simple Register. The timer period can be programmed in the period register with word offset 0x204 high byte (bit 8-15).
The time which result from this register could be calculated as follows:

Timer period = programmed value x 10µs. (FPGA Version 14 and higher)

Start timer set bit TIMER_ON: word offset 0x205 bit 0.

The Timer IRQ will be deleted by each read access on the timer register 0x205. Read actual IRQ level of timer at bit 1.

Word address offset 0x204	Bit 0-7	Systembus Info (Lost Clock's), after finished cycles always 0.
	Bit 8-15	Timer period register, value x 10µs = Timer cycle.
Word address offset 0x205:	Bit 0	TIMER_ON; 1= Timer run, 0= Timer are in reset.
	Bit 1	TIMER IRQ Status; 1=timer interrupt, 0=no timer interrupt.

6 Systembus timer (FPGA version V17 or higher)

The systembus timer shows the duration of a systembus data transmission cycle in µs. You can read the actual value on word offset 0x206 bit 0-7.
This function is very useful if you use the interrupt timer. Then you should first determine the systembus cycle duration by this function. So you can choose a timer period which is certainly higher than the systembus cycle duration.

Word address offset 0x206	Bit 0-7 actual value duration of systembus cycle in µs

7 Internal watchdog (FPGA)

In the FPGA is a watchdog integrated with a timeout of 200ms. It will be triggered with each **ram_switch** (see command register). In the operating mode FREE_RUN you must trigger the watchdog manually by writing status register 0x203 bit 5=1.

7.1.1 What happens if watchdog occurs

The watchdog generate a reset of the sytem bus. All slave prints go in reset state. For new datatransfer you must initialisate the system bus again.

7.1.2 Disable Watchdog

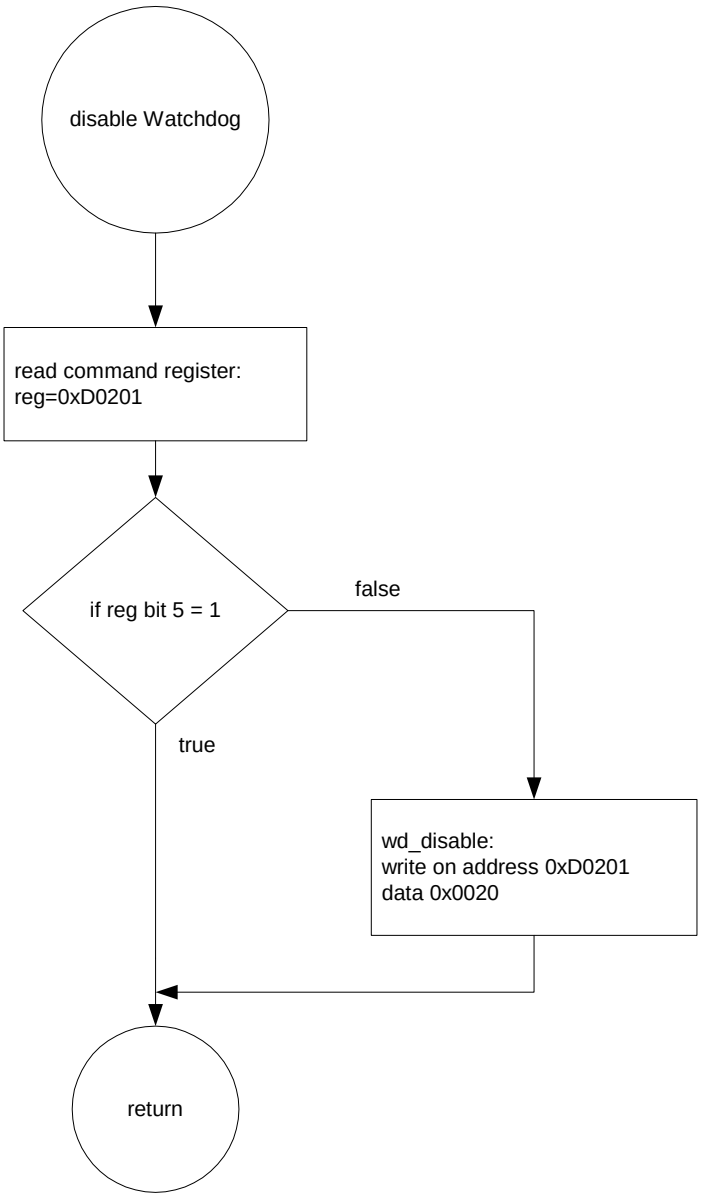

Figure 2: Disable Watchdog

1.1.1 Enable Watchdog

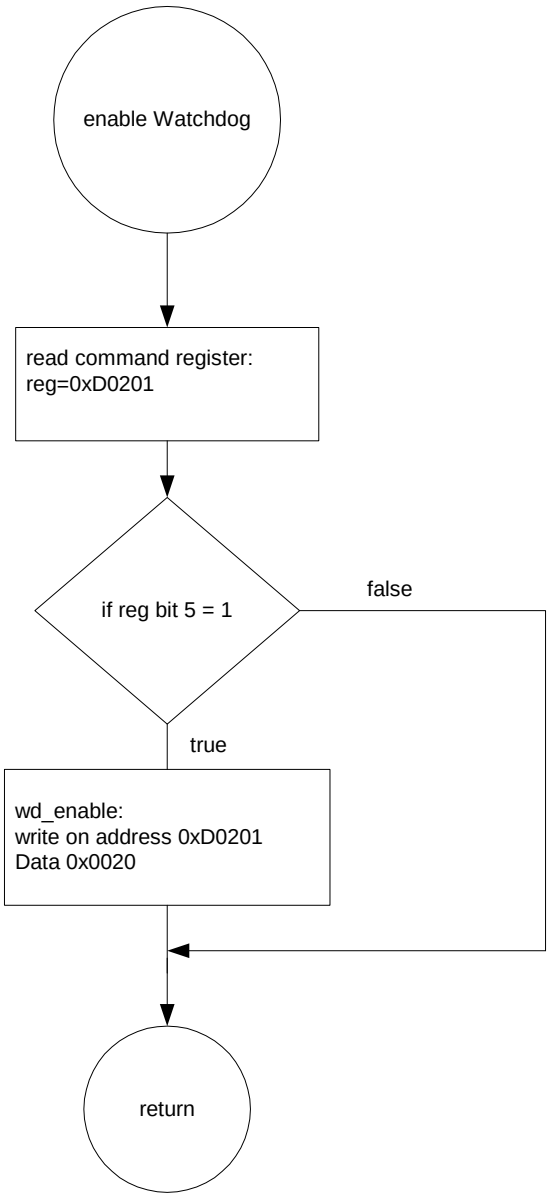

Figure 3: Enable Watchdog

8 Initialisation of system bus

For easy initialisation you can use the command FST_TRAN. Then you must manually calculate the amount of system bus data by looking at RAM1 or RAM2. The calculated value must be written in register ANZ_BYTES (0x200 word offset). Then you must set the ST_NORM bit.

Example:

RESET_MASTER
    wait 2ms
RESET_SLAVE
WD_DISABLE
    clear receive page
RAM_SWITCH
    clear receive page
FST_TRAN
    wait 2ms
ANZ_BYTES= (bytes of systembus slave moduls) –1
    example SSI 2K @P5200 64 Bit= 8 Byte on Systembus: ANZ_BYTES= 8 – 1 = 7
ST_NORM

Now you can set watchdog enable and start with normal datatransfer by the command *RAM_SWITCH.*

CAUTION:
Parameter assignment on slave prints must be done after each reset or power on of system bus.

9 I²C-Interface

The I²C-Interface is only used for the communication of the supervisior chip LM81.

9.1.1 Register

9.1.1.1 Clock prescale register address offset 0x208 (word offset)

Sets transfer rate of interface. Change only if EN-Bit of control register is deleted.

Prescale (Prexx): value in register
Transferspeed

Prescale= 40 MHz /(4x transfer speed)

Example: 100kHz transfer speed
Prescale = 40MHz / (4x0,1MHz) =100 = 0x64

Reset value of Prescale: 0xFFFF

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Function	Pre15	Pre14	Pre13	Pre12	Pre11	Pre10	Pre9	Pre8	Pre7	Pre6	Pre5	Pre4	Pre3	Pre2	Pre1	Pre0

Table 4: Clock prescale register 0x208 (value of 0x64 recommended)

9.1.1.2 Control Register Addressoffset 0x209

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Function	res	res	res	res	res	res	res	res	EN_I2C	IEN	res	res	res	res	res	res

Table 5: Control Register 0x209

res: reserved bit
EN_I2C: Enable bit of I2C-Interface. During normal operation set to one (after setting clock prescale).

IEN: Interrupt Enable Bit. (not implemented)

9.1.1.3 Transmit Command Register Adressoffset 0x20A

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Function	tdat7	tdat6	tdat5	tdat4	tdat3	tdat2	tdat1	tdat0	STA	STO	RD	WR	ACK	res	res	IACK

Table 6: Transmit Command Register 0x20A

tdat7-1: next byte which should be transmitted by I²C Bus
tdat0: on data transmission LSB
on Slave Address transmission RW Bit (Read/Write)
1 read from Slave
0 write to Slave
STA: generate start transmission
STO: generate stop transmission
RD: read from slave
WR: write to slave
ACK: Acknowledge, on RD, Slave transmit ACK =0 or ACK=1
Res: reserved
IACK: Interrupt Acknowledge. if set, interrupt will be deleted

STA, STO, RD, WR and IACK Bits will be automaticalle deleted.

9.1.1.4 Receive Status Register Addressoffset 0x20B

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Function	rdat7	rdat6	rdat5	rdat4	rdat3	rdat2	rdat1	rdat0	RxACK	BUSY	res	res	res	res	TIP	IF

Table 7: Receive Status Register 0x20B

rdat7-0: last received Byte of I²C Bus
RxACK: received Acknowledge from Slave, "1" No Acknowledge received
"0" Acknowledge received
BUSY: I²C BUSY, "1" after START Signal occured
"0" after STOP Signal occured
Res: reserved
TIP: Transfer In Progress, "1" transmission runs
"0" transmission finished
IF: Interrupt Flag
IF = 1 if bytetransfer complete (only if IEN enabled)

9.1.2 Initialisation

1. Write in clock prescale register ADR 0x208 data 0x64
2. Write in control register ADR 0x209 data 0x80 or 0xC0

9.1.3 Data transmission to supervisor chip

9.1.3.1 Write

For an write access you must first transmit the device address of LM81, than the register address and than the datas.
On each transmission you must wait until it is finished before you can start the next one. You have to read the TIP. bit of receiver status register. For stabel function you should combine the wait phase with a timeout of 250µs.

Example: to the supervisor chip LM81 device address 0x58 should be written the configuration register 0x40 data value 0x1 (Start sampling of analog inputs)

- Write to Transmit Command Register 0x20A: 0x5890 (Device Adresse 0x58 WR, STA + WR)
- Wait until TIP=0 OR TIMEOUT
- Write to Transmit Command Register 0x20A: 0x4010 (Register Adresse 0x40, WR)
- Wait until TIP=0 OR TIMEOUT
- Write to Transmit Command Register 0x20A: 0x0150 (Register Daten 0x01, STO + WR)
- Wait until TIP=0 OR TIMEOUT

Hint: If TIMEOUT occurs print an error message.

9.1.3.2 Read

For an read access must first the device address of the lm81 chip be transmitted, than the whished register address and than could be the datas read. Bsp.: LM81 mit der Deviceadresse 0x58 wird die Temperatur im Register 0x27 gelesen

- Write to Transmit Command Register 0x20A: 0x5890 (Device Address 0x58 WR, STA + WR)
- Wait until TIP=0 OR TIMEOUT
- Write to Transmit Command Register 0x20A: 0x2750 (Register Address 0x27, STO + WR)
- Wait until TIP=0 OR TIMEOUT
- Write to Transmit Command Register 0x20A: 0x5990 (Device Address 0x58 RD, STA + WR)
- Wait until TIP=0 OR TIMEOUT
- Write to Transmit Command Register 0x20A: 0x0068 (Data Dummy 0x00, STO + RD + ACK)
- Wait until TIP=0 OR TIMEOUT

Temperatur = Receive Status Register 0x20B rdat7..rdat0

For direct following temperature requests, you need only a read on den device address. You must not tranmit the register address again and again.

- write to Transmit Command Register 0x20A: 0x5990 (Device Address 0x58 RD, STA + WR)
- wait until TIP=0 OR TIMEOUT
- write to Transmit Command Register 0x20A: 0x0068 (Data Dummy 0x00, STO + RD + ACK)
- wait until TIP=0 OR TIMEOUT

Temperature = Receive Status Register 0x20B rdat7..rdat0

10 LM81 supervisor chip

- See datasheet (http://www.national.com)
- Caution reading of ADC values only each 56ms possible
- Reading of all ADC values only each 0.82 sec possible
- Critical values should be watched by internal limits sets of LM81 chip, its much faster than reading all ADC values
- Calculating factors of voltages:

LM81 Register	name / Function	factor for values in volt
0x20	V_BAT (battery voltage)	0.0746
0x21	24V (Supply voltage)	0.15686
0x22	3,3V (Logic supply)	0.017188
0x23	5V (Logic supply)	0.02604
0x24	12V (van supply)	0.0625