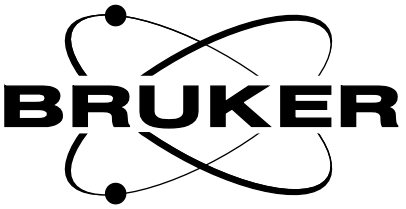


AQX

CCU
Technical Manual

Version 002

BRUKER

The information in this manual may be altered without notice.

This manual was written by

J. Tucek and M. Dudek

Rheinstetten, Germany

Updated for BASH 2.0 by UR, December 1996

P/N: Z31341

DWG-Nr: 1050 002

1 : AQX CCU_4600

J. Tucek / M. Dudek

Contents

Figures

Tables

1. Technical Description

1. 1. General Information

The AQX_CCU is a CPU board specially designed for use in the AQX-Rack. ”AQX_CCU_4600” is another type of CCU and is called version CU08.
This design is made of the memory and IO part of CU06 and a new on board ”Processor Module” using the R4600 and replacing the R3000 modul RPM3330 used on CU04 through CU07

Its version sign is ”CU08” used in this description.

CCU Versions	Part No.	Layout No.	EC Level	Software Constrains	
				AspectStation	SGI Computers
CU05	H2570	H3P2050A/B/C	EC ≤ 19	Boot Tape 940501	all releases
CU06	H2570	H3P2050D	EC ≥ 20	or later	all releases
CU07	H5830	H3P2140	(EC00)		
CU08	H2579	H3P2130A	EC ≥ 0	needs Unix for R4600 avail. on	
CU08	H2570	H3P2130A	EC ≥ 30	Boot Tape 950901 or later	

Table 1: CCU versions

1. 2. Features

- operating frequency 33 MHz on board, 100 MHz on chip
- processor peak performance is 55 VAX MIPS (related on drystones)
- data and instruction cache size 16kbyte each
- 2 mbyte firmware
- VME bus master interface and slave interface to the dynamic RAM
- Thin wire ethernet using Am7990
- 16-,32-,64-Mbyte dynamic RAM
- 8 RS232 channels using Z85C230, labeled as ”console”or ”tty00”, and ”tty01,...,tty09”;
  2 RS485 channels using Z85C230, labeled as ”tty10” and ”tty20”
  (connectable through one 9-pin SubD (”console”) and two 50-pin connectors (”tty01,...,tty09”) at the front edge)
- timer DP8571A
- real time clock MK48T02 with 2 kbyte non volatile memory
- configuration register, status register, interrupt register
- component identification channel to handle the AQX component identification system (CIS)

1. 3. Related Devices

There are two devices which can be used to adapt ”tty01,...tty09” to separate 9-pin SubD-Connectors (see Figure 1:):

- AspectStation1 RS232 Router, (only for RS232)P/N H5468
- AQX RS232/485 Extension UnitP/N H5714

Both devices have to be connected to the CCU by one or two cables of the following type:

- SCSI Cable P/N 73104

Figure 1: Attaching usual SubD connectors to the RS232/485 interfaces of CCU

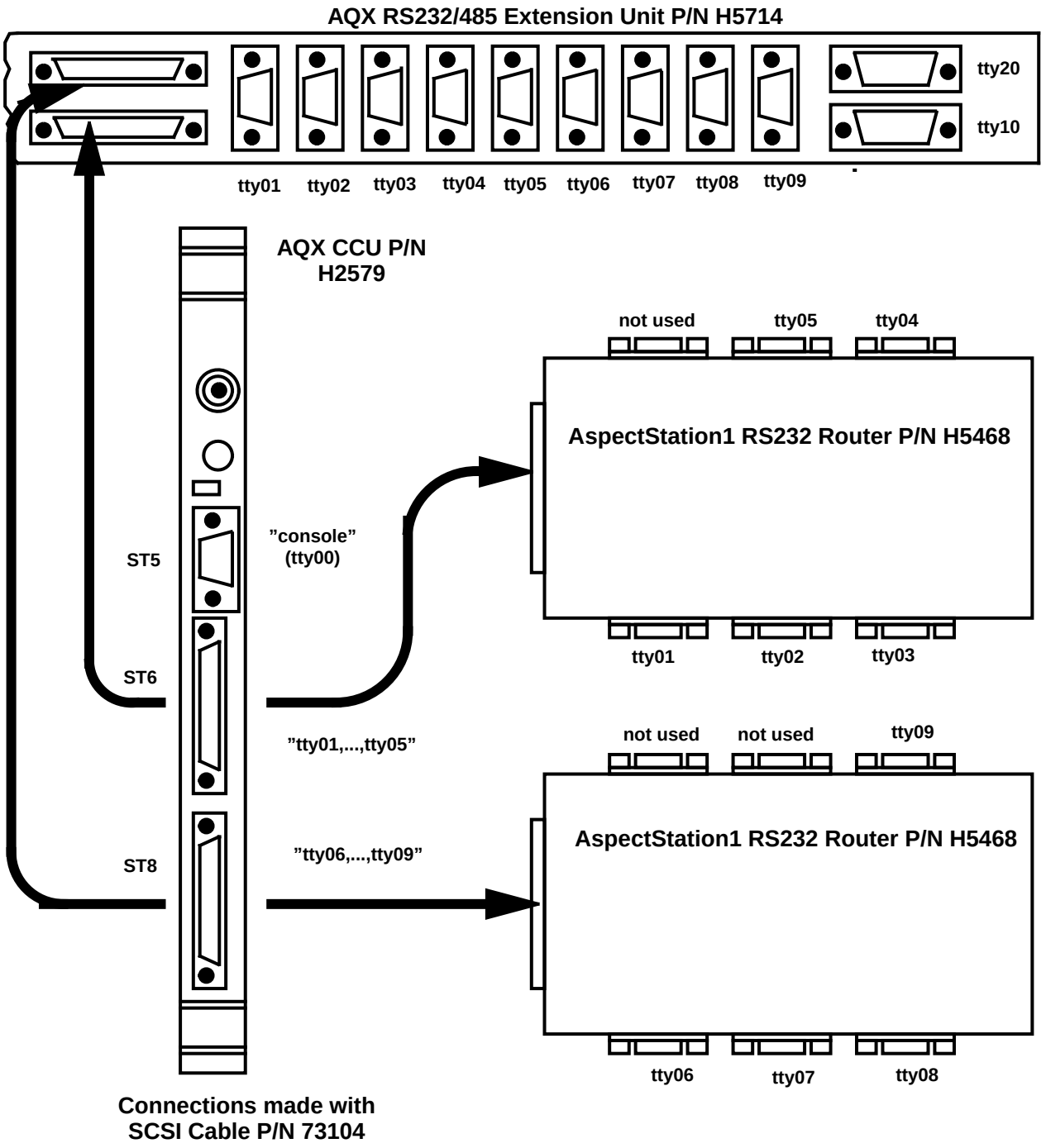


Table 2: Configuration of Serial Interfaces

	CU08				
UNIX Special File	/dev/tty00	/dev/tty01,...,05	/dev/tty06,...,09	/dev/tty10	/dev/tty20
Function	(Console)	RS232		RS485	
CCU Connector	ST5	ST6	ST8	ST6	ST8
AQX Ext. U. Connector	N/A	ST2,...,ST6	ST7,...,ST10	ST13	ST11

1. 4. Architecture

The CPU module, the ethernet subsystem and the VME bus slave interface share the local bus mastership and can access to VME bus and DRAM. All the other onboard devices which are connected to the local bus can only be accessed by the CPU module.

1. 4. 1. Block Diagram of the CCU

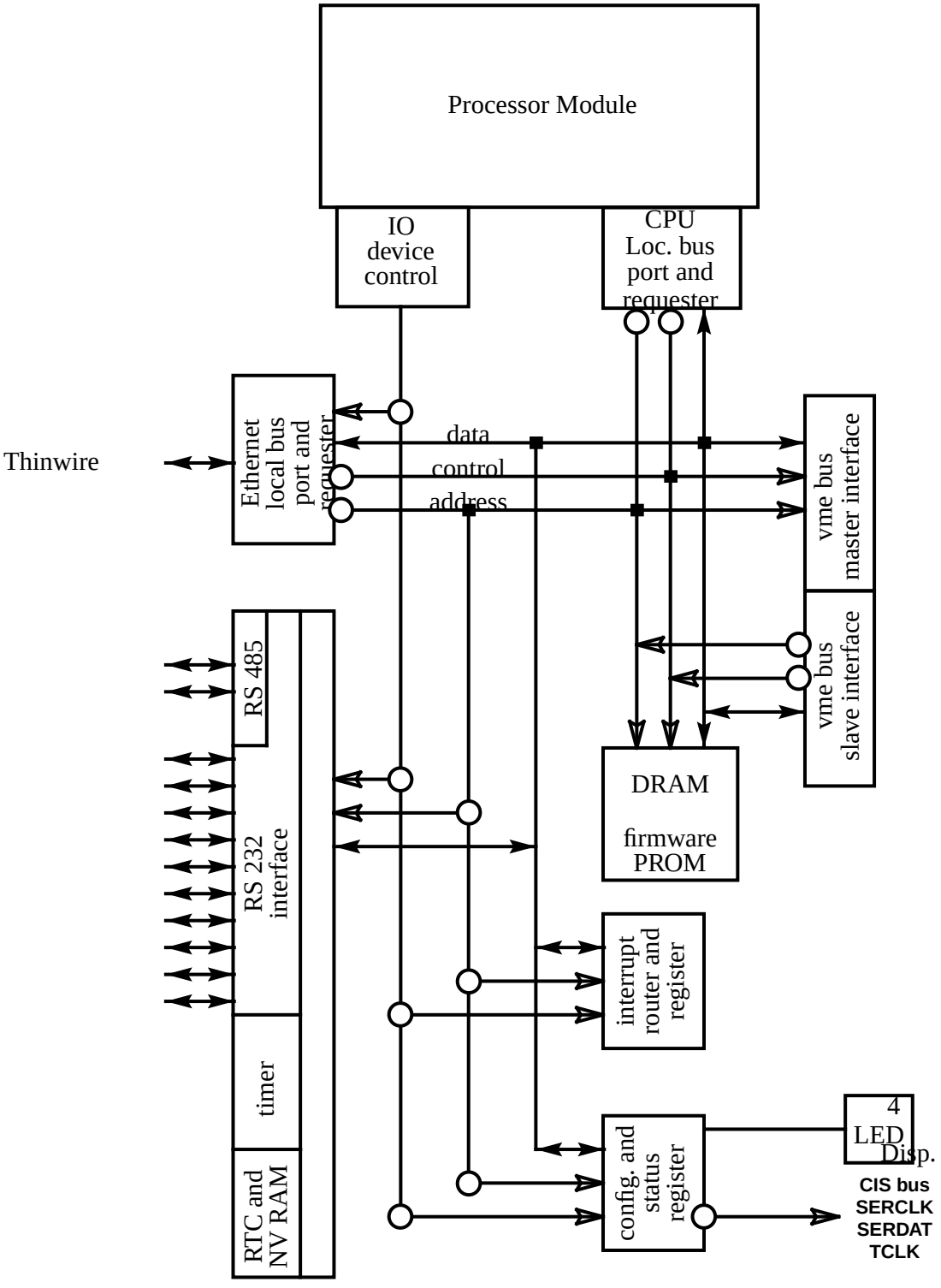


Figure 2: Block Diagram

1. 5. Processor Modul

This compact unit is built arround of R4600 Processor (Orion - Inmos). It contains only circuits necessary for function of the Processor and Interfacing to the Processor Modul Bus (PMB).

- The Processor is used in ” No-Secondary-Cache Mode ”.
- The Processor uses the ” Write re-issue ” option for back-to-back write protocol.
- In Block Write Mode, the Processor suplyes the new 64 bit data each Tclk (Sclk) period. CCU1 Circuits accept 32 bit data each second (third) Tclk period. Four 64 Bit words are buffered and adjacend with max . posible speed written into the memory.
- During the Block Read Response is the Data Rate controlled by the ” ValidIn* ” signal to the Processor. The 32 bit wide Data words are suplyed by CCU1 Circuits (each third Tclk period by memory) at the maximal speed..
- In Block Read Mode is the sequence of Doublewords subject of ’ Subblock Ordering ” (see Kap. 2.4.1.2).
- The Processor is the only Master on the Processor ModulBus (PMB).
- Only Processor requests are served on PMB.
- The Processor werks internally with clock 100 Mhz.
- System Interface uses clock 33.33 Mhz. This clock is used in PMB and it is the basic clock for CCU1 Circuits. The PMB and CCU1 circuite are designed for the Max. clock frequency 50 Mhz.
- The Next Processor access to any device on the PMB Will be allowed first after the previous operation is totaly completed (writen into the memory or peripheral). This should guatarantie that the data written in the previous processor access cycle will be read by immediately following processor read access to the same address (not the old -unchanged data). This means the strict sequential processing of the accesses. Care must be taken during Block Write accesses followed by read access to the same address.
- Link address retained/not retained and
  Cache line replaced/not replaced and
  in SysCmd :
  are Ignored !!!
- Error identification is fixed to ” No error ” in the Data Identifier !!!

1. 5. 1. Block Diagram of the Processor Modul

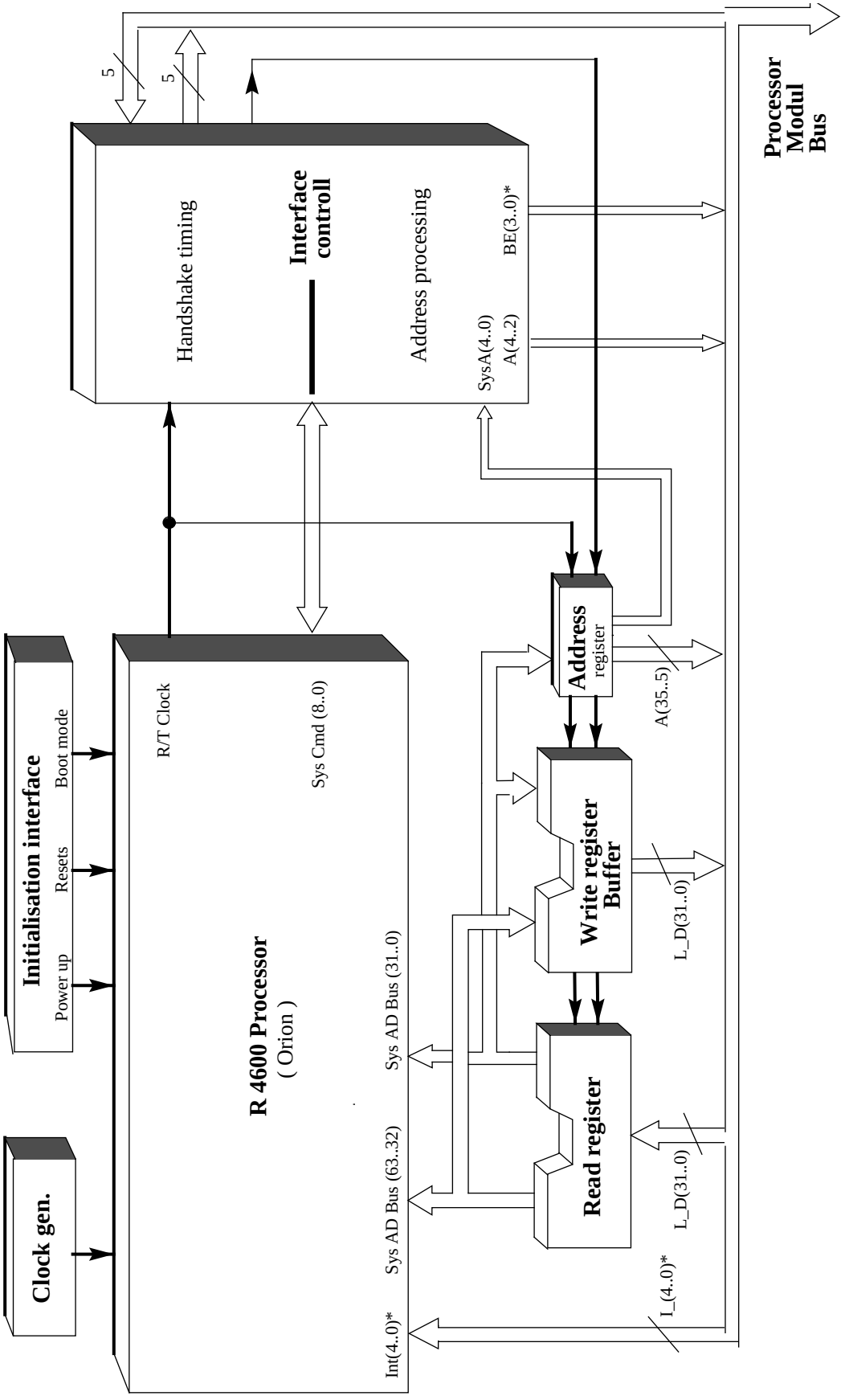


Figure 3: Processor Module

1. 5. 2. Functional settings and conditions

1. 5. 2. 1. Boot-Mode Settings

Serial Bit :	Value :	Name:	Mode setting:
0	0	reserved	-
4..1	0	XmitDatPat	DDDD
7..5	1	SysCkRatio	Divide by 3
8	1	EndBit	Big-endian ord.
10..9	3	Non-block Write	Write re-issue
11	0	TmrIntEn	Enabled Timer Int.
12	0	reserved	-
14..13	11	Drv-Out	83% strength
255..15	0	reserved	-

1. 5. 2. 2. Subblock Ordering

Doubleword addresses on the PMB [A (4..3)] during Block read Cycles in dependence on the Start Block address [SysAD (4..3)] :

Starting Block address : >>> SysAD (4..3)	00	01	10	11
Cycle :	Doubleword address : A (4..3)	Doubleword address : A (4..3)	Doubleword address : A (4..3)	Doubleword address : A (4..3)
1	00	01	10	11
2	01	00	11	10
3	10	11	00	01
4	11	10	01	00

1. 5. 2. 3. Partial Word Transfer Byte addressing

Bytes (Sys-Cmd)	Addr. Mod 8	SysAD byte lanes used (big endian only !)								A2	BE* (3:0)
		63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0		
1 (000)	0	*								1	7
	1		*							1	b
	2			*						1	d
	3				*					1	e
	4					*				0	7
	5						*			0	b
	6							*		0	d
	7								*	0	e
2 (001)	0	*	*							1	3
	2			*	*					1	c
	4					*	*			0	3
	6							*	*	0	c
3 (010)	0	*	*	*						1	1
	1		*	*	*					1	8
	4					*	*	*		0	1
	5						*	*	*	0	8
4 (011)	0	*	*	*	*					1	0
	4					*	*	*	*	0	0
5 (100)	0	*	*	*	*	*				1	0
	0	*	*	*	*	*				0	7
	3				*	*	*	*	*	1	e
	3				*	*	*	*	*	0	0
6 (101)	0	*	*	*	*	*	*			1	0
	0	*	*	*	*	*	*			0	3
	2			*	*	*	*	*	*	1	c
	2			*	*	*	*	*	*	0	0
7 (110)	0	*	*	*	*	*	*	*		1	0
	0	*	*	*	*	*	*	*		0	1
	1		*	*	*	*	*	*	*	1	8
	1		*	*	*	*	*	*	*	0	0
8 (111)	0	*	*	*	*	*	*	*	*	1	0
	0	*	*	*	*	*	*	*	*	0	0

1. 5. 2. 4. Command and Data Identifier

(Relevante only)

Command Identifier :

Read request

SysCmd	8	7	6	5	4	3	2	1	0	Noncoherent
Command	0	0	0	0	1	0		0	1	Block read (8 Words)
11 or 15 (hex)	0	0	0	0	1	0	X	0	1	Block read 8 Words

SysCmd	8	7	6	5	4	3	2	1	0	Doubleword,Partialdoubleword,
Command	0	0	0	0	1	1				word or partial word read
18 (hex)	0	0	0	0	1	1	0	0	0	1 byte valid (Byte)
19	0	0	0	0	1	1	0	0	1	2 bytes valid (Halfword)
1A	0	0	0	0	1	1	0	1	0	3 bytes valid (Tribyte)
1B	0	0	0	0	1	1	0	1	1	4 bytes valid (Word)
1C	0	0	0	0	1	1	1	0	0	5 bytes valid (Quintibyte)
1D	0	0	0	0	1	1	1	0	1	6 bytes valid (Sextibyte)
1E	0	0	0	0	1	1	1	1	0	7 bytes valid (Septibyte)
1F	0	0	0	0	1	1	1	1	1	8 bytes valid (Doubleword)

Write request

SysCmd	8	7	6	5	4	3	2	1	0	Noncoherent
Command	0	0	1	0	1	0		0	1	Block write (8 Words)
51 or 55 (hex)	0	0	1	0	1	0	X	0	1	Block write 8 Words

SysCmd	8	7	6	5	4	3	2	1	0	Doubleword,Partialdoubleword,
Command	0	0	1	0	1	1				word or partial word write
58 (hex)	0	0	1	0	1	1	0	0	0	1 byte valid (Byte)
59	0	0	1	0	1	1	0	0	1	2 bytes valid (Halfword)
5A	0	0	1	0	1	1	0	1	0	3 bytes valid (Tribyte)
5B	0	0	1	0	1	1	0	1	1	4 bytes valid (Word)
5C	0	0	1	0	1	1	1	0	0	5 bytes valid (Quintibyte)
5D	0	0	1	0	1	1	1	0	1	6 bytes valid (Sextibyte)
5E	0	0	1	0	1	1	1	1	0	7 bytes valid (Septibyte)
5F	0	0	1	0	1	1	1	1	1	8 bytes valid (Doubleword)

Data Identifier :

Precessor write (Output) Data Identifier Encoding :

SysCmd	8	7	6	5	4	3	2	1	0	Meaning :
	1	0	X	X	X	X	X	X	X	Last data element
	1	1	X	X	X	X	X	X	X	Not the last data element

Processor read (Input) Data Identifier Encoding :

SysCmd	8	7	6	5	4	3	2	1	0	Meaning :
	1	0	0	0	1	1	1	1	1	Last data element
	1	1	0	0	1	1	1	1	1	Not the last data element

1. 6. Logical References to memory and IO devices

Access Characteristics

- Depending on configuration, the cpu can work in big or little endian mode.For independence the operands within io-device code ranges should be referred to as word (32 bit) operands with the valid bytes indicated by ”b” at ”Byte format” (see below).

- ”Byte 0” means the byte connected to data lines D0 to D7

- Byte, halfword (2 byte), word (4 byte) and block (4 word) accesses occur

- The cpu reads at least 4 bytes at once from the memory range, but it accesses to the vme io-code range and it writes everywhere only the bytes the processor is referring to.

CU04 through CU08 predestinate the following address map:

Address	Range of destination	Access to	Block mode
00000000 . up to border of installed DRAM	main memory	local bus if DRAM installed, otherwise vme bus	yes
border of installed DRAM 0FFFFFFF	main memory	vme bus	yes
10000000 17FFFFFF	(former graphic memory)	vme bus	no
18000000 1DFFFFFF	coprocessor memory	vme bus	no
1E000000 1EFFFFFF	vme bus io-codes	vme bus	no
1F000000 1F3FFFFF	CU08 onboard io-codes	onboard local bus access	no
1F800000 1FFFFFFF	CU08 firmware	onboard prom access	no
20000000 FFFFFFFF	main memory	vme bus	yes

1. 6. 1. Firmware

- The reserved firmware range amounts to 8 mbyte.

- The firmware memory is 32 bits wide, containing 2 mbyte and is connected to the DRAM subsystem. It consists of 2 proms, 16 bits wide each. The start address is1FC00000.

1. 6. 2. Onboard memory

It is possible to install 16, 32, or 64 Mbyte DRAM on board. This RAM has to be enabled by setting jumper W12 and is then located in the lowest address space starting with address 0x0. The installed memory volume must properly be selected with the setting of jumper W1 (see ”Operational settings”). The current UNIX version does not support memory volumes smaller than 16 Mbyte.

1. 6. 3. VME bus port

1. 6. 3. 1. Master Port

Special Features

- In general the cpu provides as bus master the so called **”extended address AM-Code”** indicating the validity of all address lines. But there are two special device code ranges that causes the cpu to escort their bus activities with the **”standard address AM-Code 3D”** or the **”short address AM-Code 2D”**. These ranges are reserved for VME-Devices that are incapable of decoding address lines A24 to A31 (standard) or A16 to A31 (short).

- The interrupt vectors of the vme bus can be read by accessing special vme device codes.

Device Codes

Address	Destination	Read/Write	Byte Format3 2 1 0
1E9xxxxx	Short AM-Code range 2D	R/W	b b b b
1EA00000 1EBFFFFF	Standard AM-Code range 3D	R/W	b b b b
1E89xxxx	interrupt vector of IRQ1	R	x x x b
1E8Axxxx	interrupt vector of IRQ2	R	x x x b
1E8Bxxxx	interrupt vector of IRQ3	R	x x x b
1E8Cxxxx	interrupt vector of IRQ4	R	x x x b
1E8Dxxxx	interrupt vector of IRQ5	R	x x x b
1E8Exxxx	interrupt vector of IRQ6	R	x x x b
1E8Fxxxx	interrupt vector of IRQ7	R	x x x b

1. 6. 3. 2. Slave Port

VME bus accesses can reach only the on board DRAM addresses via this port. The address range depends on the installed volume and starts always with VME bus address 0x0.
Byte, word, longword, 2-, 3-, 4-word nibbles and pagemode accesses are possible. Pagemode is not recommended because of its impact to the on board ethernet operation.
Single word writes and reads use a single read/write register. A pipeline structure is made for multiple writes and higher bandwidth.
The slave port accepts VME bus references with extended 32-bit addresses escorted with the following AM-Codes:

0F, 0E, 0D, 0B, 09;

1. 6. 4. Ethernet

Notes

- The ethernet subsystem is based on the Am7990 with 32 bit address and a 32 bit wide data port.

- DMA address bits 24 to 31 are clamped to zero.

Device Codes

Address	Destination	Read/Write	Byte Format3 2 1 0
1F060xxx	Am7990 Register Data Port	R/W	x x b b
1F061xxx	Am7990 Register Address Port	R/W	x x b b

1. 6. 5. RS232 Interface

Notes

- The RS232/RS485 Interfaces reside in six Z85C230 providing 12 separate channels.

- Channel A is intended to be the ”console”, channel B to L to be ”tty01”, ”tty02”,..., ”tty09”,”tty10”,”tty20”

- tty10 and tty20 are configured as RS485 channels

- The frequency of PCLK at pin 23 of Z85C230 is 10 MHz

Device Codes

Address	Destination	Read/Write	Byte Format3 2 1 0
1F020xxx	Channel B Control Register, tty01	R/W	x x x b
1F021xxx	Channel B Data Register, tty01	R/W	x x x b
1F022xxx	Channel A Control Register, cons.	R/W	x x x b
1F023xxx	Channel A Data Register, cons.	R/W	x x x b
1F024xxx	Channel D Control Register, tty03	R/W	x x x b
1F025xxx	Channel D Data Register, tty03	R/W	x x x b
1F026xxx	Channel C Control Register, tty02	R/W	x x x b
1F027xxx	Channel C Data Register, tty02	R/W	x x x b
1F030xxx	Channel F Control Register, tty05	R/W	x x x b
1F031xxx	Channel F Data Register, tty05	R/W	x x x b
1F032xxx	Channel E Control Register, tty04	R/W	x x x b
1F033xxx	Channel E Data Register, tty04	R/W	x x x b
1F034xxx	Channel H Control Register, tty07	R/W	x x x b
1F035xxx	Channel H Data Register, tty07	R/W	x x x b
1F036xxx	Channel G Control Register, tty06	R/W	x x x b
1F037xxx	Channel G Data Register, tty06	R/W	x x x b
1F038xxx	Channel J Control Register, tty20	R/W	x x x b
1F039xxx	Channel J Data Register, tty20	R/W	x x x b
1F03Axxx	Channel I Control Register, tty10	R/W	x x x b
1F03Bxxx	Channel I Data Register, tty10	R/W	x x x b
1F03Cxxx	Channel L Control Register, tty9	R/W	x x x b
1F03Dxxx	Channel L Data Register, tty9	R/W	x x x b
1F03Exxx	Channel K Control Register, tty8	R/W	x x x b
1F03Fxxx	Channel K Data Register, tty8	R/W	x x x b

1. 6. 6. Timer

Notes

- Timer is the DP8571A made by National Semiconductor

- Programmers have to pay attention to the following hardwired conditions

There is no battery back up for this device. Pin 9 (VBB) is connected to pin 12 (GND).

The Power Fail function can't be used. Pin 23 is connected to pin 24 (VCC).

Pin 14 and 13 are wired to be utilized as timer interrupts TIM_I0 and TIM_I1

Pin 10 and pin 11 are connected to a crystal of 4.9152 MHz .

Device Codes

Address	Destination	Read/Write	Byte Format3 2 1 0
1F0100xx 1F011Fxx	Device code range 00 to 1F of DP8571A	R/W	x x x b

1. 6. 7. Real Time Clock and NVRAM

Notes

- The MK48T02 of SGS-Thomson is installed to implement these two functions by one device.

- It is a 2K-Byte-Static-RAM with its eight upmost cells being reserved as hold register for the real time clock information.

- The lower half of the 2 kbyte is normally protected and can only be written by means of the special PAL device "CU08Sx02 " of the service staff. So far, this range contains only the "ethernet physical address".

- Besides the timing registers, the upper half of NVRAM contains all the other boot parameters read with "printenv" and changed with "setenv" in the monitor program.

Device Codes

Address	Destination	Read/Write	Byte Format3 2 1 0
1F00000x 1F006FFx	NVRAM range, lower half	R	x x x b
1F00700x 1F007F7x	NVRAM range, upper half	R/W	x x x b
1F007F8x	Control Register	R/W	x x x b
1F007F9x	Seconds	R/W	x x x b
1F007FAx	Minutes	R/W	x x x b
1F007FBx	Hour	R/W	x x x b
1F007FCx	Day	R/W	x x x b
1F007FDx	Date	R/W	x x x b
1F007FEx	Month	R/W	x x x b
1F007FFx	Year	R/W	x x x b

1. 6. 8. Configuration and Status Register

Notes

- There are 4 configuration register and 1 status register. Each of them is 8 bit wide.

- Each configuration register provides two hexadecimal digits indicating the version of two individual onboard subsystems as follows:

CPU version	CPUV = 0x8 on CU08
Prozessor version	PRZV = 0x1
Real Time Clock version	RTCV = 0x1
Keyboard Controller version	KBV = 0x0
RS232 Interface version	RSV = 0x2
Ethernet version	ETHV = 0x1
SCSI Controller version	SCSIV = 0x0

A subsystem is not available if the respective digit equals to zero. All configuration register outputs are provided in their negated form

The **STATUS register** accommodates the temperature and VME bus time out status bits, 4 LED bits and 3 control bits of the ”CIS” channel (component identification system).

Read access to the STATUS register provides the occurrence of TMP_INT (temperature interrupt) on bit 1 and the occurrence of vme bus time out on bit 0 since last read. Both are low active and cleared in sequence of a read.
Bit 3 of the status reflects the state of SDA, this is the serial data line of the VME-Bus.

Write access stores the upper 4 bits (D7,...,D4) of the transferred byte to the 4Bit-Led-Register. D3, D2 and D1 control the serial bus lines SDA, SCLK and ACLK.

	Bit Allocation of the STATUS Register							
	D7	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**
Write	Leftmost LED	LED at front edge		Rightmost LED	SDA at J1/B22	SCLK at J1/B21	ACLK at J2/B3	
Read							TMP_INT	VME bus Time out

Device Codes

Address	**Destination**	**Read/Write**	**Byte Format3 2 1 0**
1F05Fxxx	STATUS	R/W	x x x b
1F050xxx	SCSIV, --	R	x x x b
1F051xxx	RSV, ETHV	R	x x x b
1F052xxx	RTCV, KBV	R	x x x b
1F053xxx	CPUV, PRZV	R	x x x b

1. 6. 9. Interrupt Router

Notes

- The processor reaches its 6 interrupt levels by means of the 6 interrupt inputs, INT_5..INT_0.

- INT_5 (highest level) is assigned and hardwired to the floating point interrupt.

- The router distributes all onboard and vme bus interrupt sources to level 0 to 4. Source interrupts intended to a certain level are ored to produce the processor interrupt INT_i. The source interrupts of each level form an Interrupt Register that can be read via device code providing the status of that level.

- Once activated a source interrupt signal has to keep its level up to being acknowledged with a special activity like reading an assigned status register or the assigned vme interrupt vector.

- The router contains only for the temperature interrupt "TMP_INT" an interrupt mask. This mask is opened after reset and by writing to the device code with D1=1 and closed by writing with D1=0.

Routed Source Interrupts

IRQ1..7	7 VME bus interrupts
RS_INT0..5	5 interrupts of 5 individual RS232 subsystems RS_INT0 is the output of Z85C230, Channel A and B RS_INT1 is interrupt of Channel C and D RS_INT2 is interrupt of Channel E and F RS_INT3 is interrupt of Channel G and H RS_INT4 is interrupt of Channel I and J RS_INT5 is the output of Z85C230, Channel K and L
TIM_INT0..1	2 timer interrupts, connected to pin 14 and 13 of the DP8571A
TMP_INT	High Temperature interrupt
LAN_INT	Ethernet Interrupt, pin 6 of Am7990

The routed distribution of source interrupts and their location in the corresponding interrupt register are as follows:

Interrupt Level					**Location**
INT_0	**INT_1**	**INT_2**	**INT_3**	**INT_4**	**in Reg.**
IRQ1	IRQ4	IRQ6	0	IRQ7	D0
TMP_INT	IRQ3	IRQ5	0	0	D1
TIM_INT1	IRQ2	0	0	0	D2
0	0	0	0	0	D3
0	0	0	0	0	D4
0	0	0	0	0	D5
0	0	0	0	0	D6
RS_INT	0	0	TIM_INT0	LAN_INT	D7

RS_INT is the ored sum of RS_INT0,...,RS_INT5

Device Codes

Address	Destination	Read/Write	Byte Format3 2 1 0
1F040xxx	INT0 Register	R	x x x b
1F041xxx	INT1 Register	R	x x x b
1F042xxx	INT2 Register	R	x x x b
1F043xxx	INT3 Register	R	x x x b
1F044xxx	INT4 Register	R	x x x b
1F048xxx	INT0 Mask (only D1 implemented)	W	x x x b

1. 6. 10. CIS access

The information EEPROM of the Component Identification System is connected to the CCU controlled CIS channel. The EEPROM of all CCU's have one and the same unique address out of all possible channel addresses as follows:

Device	SBA binary										SBA hex	
	9	8	7	6	5	4	3	2	1	0	old	new
	Proto-col bit	Group address						EEPROM address				
CCU	0	0	0	0	0	0	1	0	0	0	008	010

1. 7. Operational Settings

1. 7. 1. Configuration

Some features or properties of CU08 were adjusted in the factory or may be later configured.

- Firmware version
  - The originally installed and labeled firmware Prom's are "40Hjjmmdd" containing the two high bytes and "40Ljjmmdd" containing the low bytes.
    Firmware Prom's of the R3000 CCU versions must not be used on CU08 and vice versa.
- Installed type of firmware Prom's
  - A choice out of 2 types is possible, 2 devices in PLCC-Package are required.
  - W11 removed: 272048 may be used resulting in 512 kbyte
  - W11 inserted: 274096 can be used resulting in 1 megabyte firmware space
  - Access time is required to be 120 nsec or less
- SYSCLK on VME Bus
  - Jumper W10 connects if inserted the cpu clock to SYSCLK of the VME Bus.
- Ethernet/Cheapernet Configuration
  - Jumper W2,...,W7 have to be in position B for using the normally mounted Cheapernet BNC connector.
  - Position A of W2,...,W7 is necessary if the Ethernet AUI connector have been installed and should be used
  - Jumper W8 inserted in position B makes the cheapernet to send a SQE test sequence, W8 in position A disables the SQE test mode, e.g. if there is a Repeater connected.

- DRAM Configuration

  - It is possible to equip the CCU with 16, 32 or 64 Mbyte using 1 or 2 of
    8-Mbyte DRAM modules (P/N 65173),
    16-Mbyte DRAM modules (P/N 66123) or
    32-Mbyte DRAM modules (P/N 66124).
    The modules have to be inserted as
    U19 or
    U19 and U20.
    Furthermore this needs W12 to be set and the following jumper setting of W1 position A and B :

Inserted Jumper in position A and B of W1	Corresponding logic signals SEL0, SEL1	CU08	
		DRAM Size	Module Equipment
A,B	0,0	16 Mbyte	2 x 8 Mbyte as U19,U20
-,B	1,0	16 Mbyte	1 x 16 Mbyte as U19
A,-	0,1	32 Mbyte	2 x 16 Mbyte as U19,U20
-,-	11	64 Mbyte	2 x 32 Mbyte as U19,U20

Table 3: DRAM configuration

  - Any installed DRAM is not available if W12 is removed

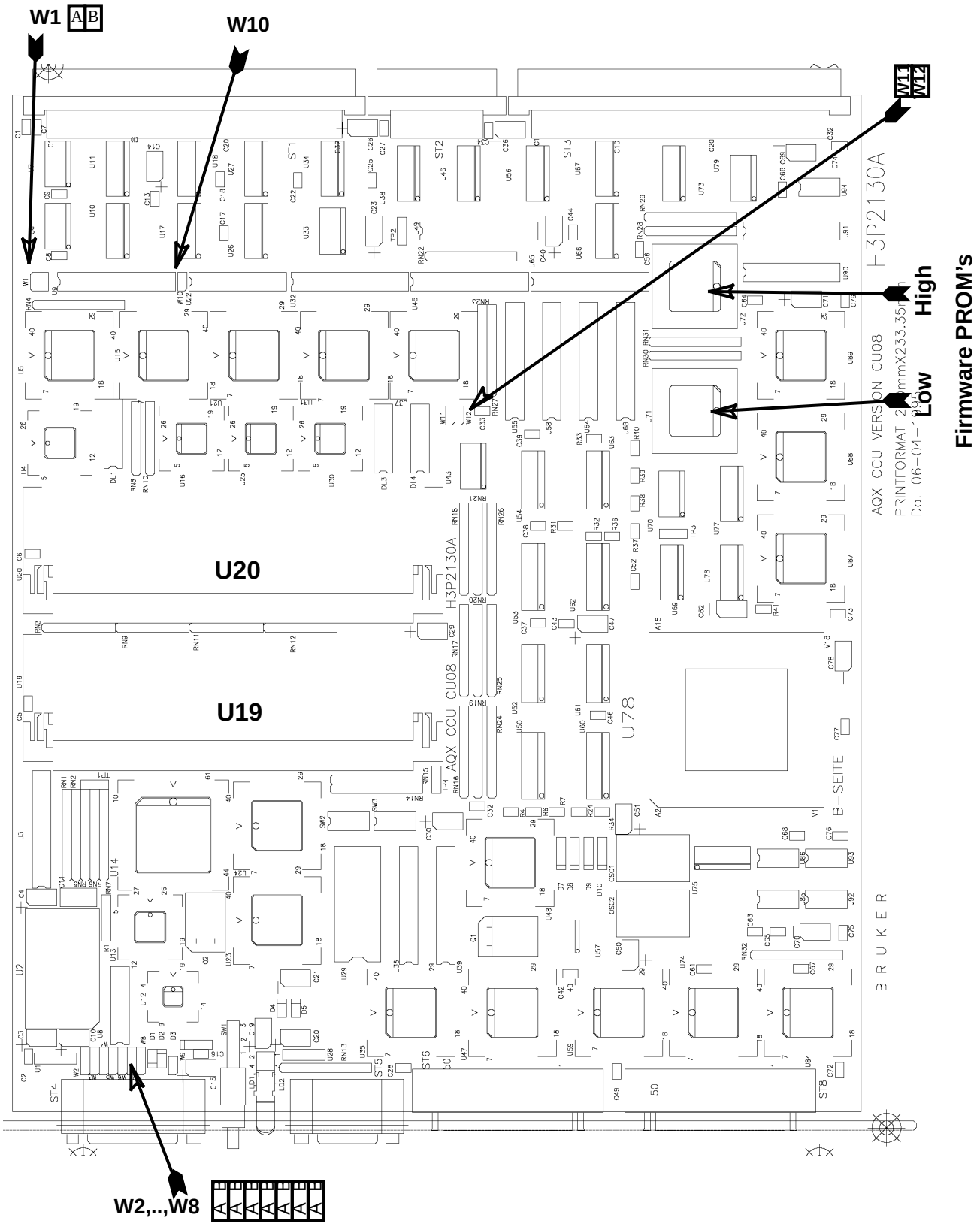
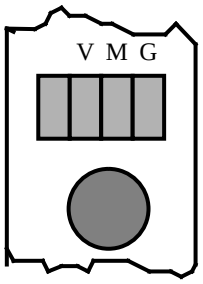


Figure 4: Jumper, Prom and DRAM Locations

1. 7. 2. Controlling Elements



LED Display

Reset Button

Figure 5: Front View at LED Display and Reset Button

Reset

- The general hardware reset signal of CCU and VME-Bus stays active for about 80 milliseconds.

- It is initiated either by one of the following activities

Pushing the reset button

- A low active signal for about 100 nanoseconds at the rear panel connector ST2 pin B10

LED Display

- The 4 digit LED-Display forms a hexadecimal coded value with MSB at the left side. Lighting reflects a logical ”One”.

- The display shows the 4 bit contents of the Status-Register (see Figure 5:) written on D7-D4 if this value is less than 8, usually after power-up or reset.

- But if MSB is set in the Status-Register the lower 3 digits from right to left reflect the activities of the Ethernet-DMA, the CPU on DRAM and the activity of the CPU on vme bus.

- The intention is that after power-up or reset a basic test procedure changes the LED value at success in steps from zero up to eight where eight means all tests have been passed and the monitor is running.

1. 8. Specifications and Connections

The CCU consists of one printed circuit board.

1. 8. 1. Board Size

The real size is 233.35 mm by 280 mm . This is the so called ”Double European Standard” format with a nominal plug in depth of 280 mm.

1. 8. 2. Connectors and Controlling Elements

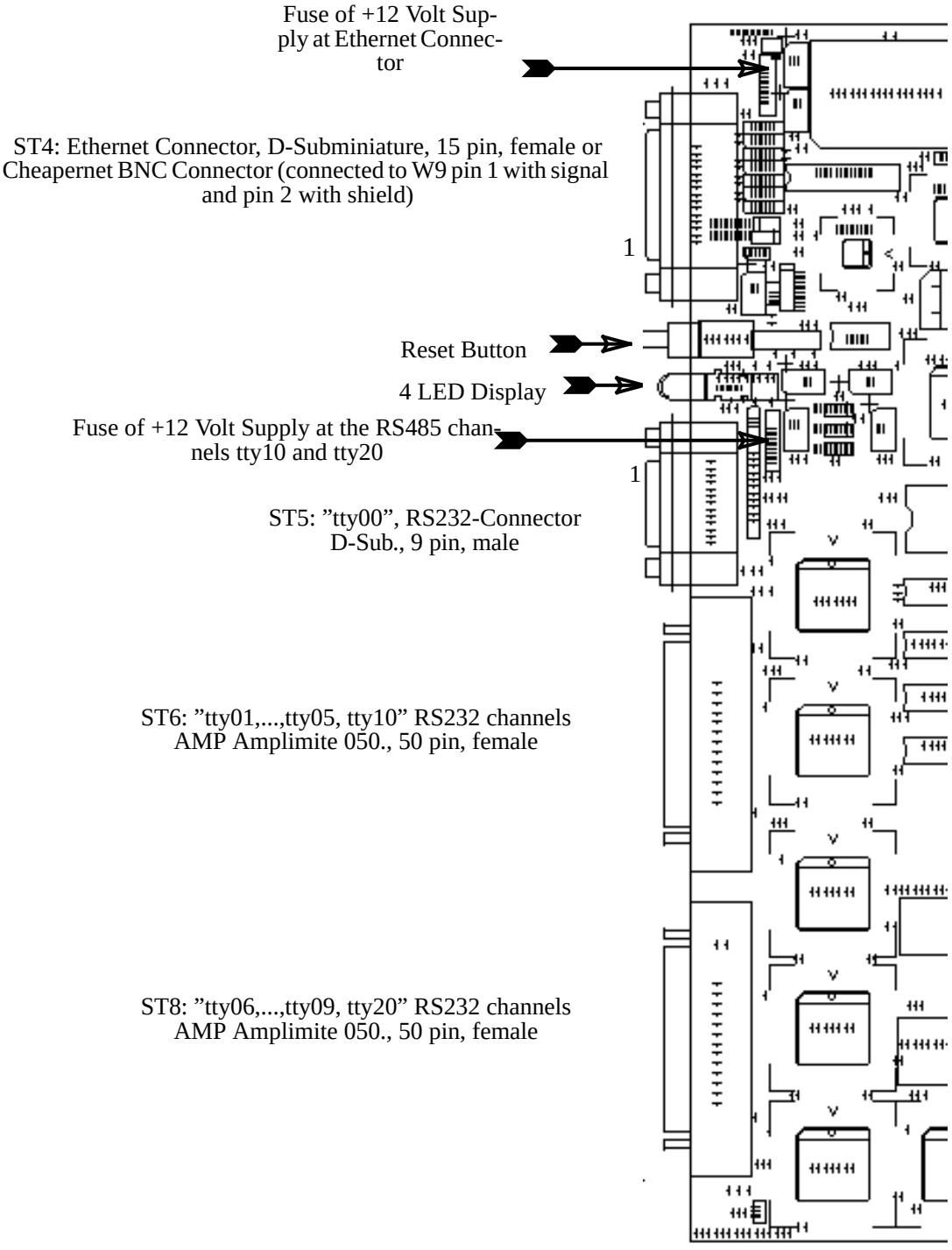
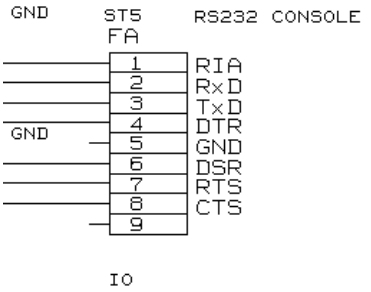


Figure 6: Location of Connectors and Operating Elements

1. 8. 3. Connectors and Signal Allocations

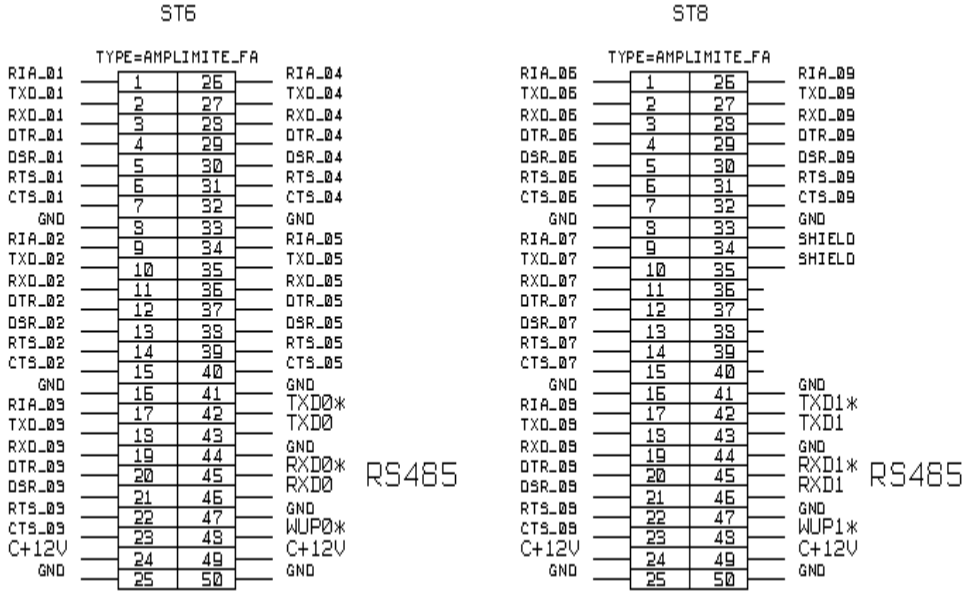
ST5 ”Console” (tty00) RS232-Connector

- D-Subminiature, 9 pin, male



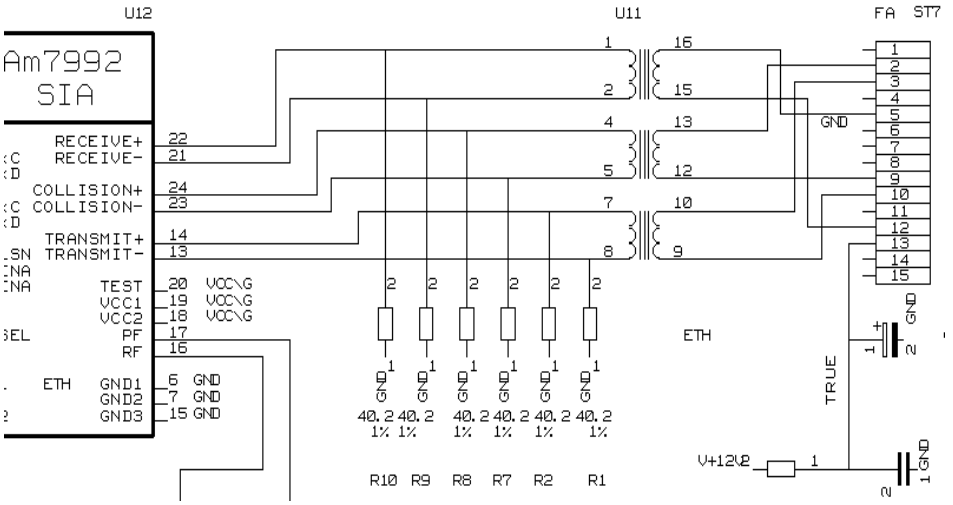
ST6/ST8 ”tty01,...,tty09,tty10,tty20” RS232-Connector

- AMP Amplimite 050, 50 pin, female



ST4 Ethernet Connector (AUI)

- D-Subminiature, 15 pin, female



There is the choice to use the Ethernet AUI interface or the Thinwire (Cheapernet) interface. The Cheapernet solution needs a special front panel with a BNC connector wired to W9pin1 (signal) and pin2 (shield) and the jumpers W2,...,W8 set in the right position (see chapter 1. 7. 1.).

1. 8. 4. Power Requirements

The CU08 requires power supply of the following voltages:

	Part-No.	+5 V	+12 V	-12 V	+5 V analog J3: C8	-5 V analog J3: C1,...,C5
CCU: CU08	H2570	6,7 A	0,1 A	0,1 A	0	0

It supplies fused current at the following connectors:

ST6/8 RS485+12 Volt 0.2 Ampere

ST4 Ethernet+12 Volt 0.2 Ampere

- The resistance of the fuses will be suddenly growing with temperature caused by high current. They recover when cooled down and need not be replaced.

2. Manufacturing Informations

2. 1. Manufacturing Data

Table 4: Table of Assembly Groups

Amount	Title	Function	Part-Nr.
1	AQX_CCU	Assembled PCB	H2579
1		Layout	H3P2130A
1	PCB	Plain PCB	H2580
1	CU08 PAL set		H
1	Front-Panel Assembly Set	Front panel with Ethernet AUI connector	Hz03496
1	Front-Panel Assembly Set	Front panel with Cheapernet BNC connector	Hz03632
1	Front-Panel-Ident	”CCU/8”	Hz3571

2. 2. Introduction Status

2. 2. 1. Configuration

Normal setting of installed jumpers:

W1	A+B on CU08 (16 Mbyte DRAM)
W2,...,W7	B (Cheapernet)
W8	B (SQE enabled)
W10	not set
W11	set
W12	set

2. 2. 1. 1. NVRAM setting

The way how you can set the NVRAM parameters to the right value depends on the value of its parameter ”**magic**”.
If magic=RISCPROM and the bauderate or console are incorrect, you have only the chance to insert this device into a CPU running in monitor without power off and set ”magic” to an other value than RISCPROM.

The monitor sets after power up the NVRAM parameters to the right value if it detects ”magic”as not equal to ”RISCPROM”.

There are only the following parameters which have to be corrected.

console

”0” : - The monitor uses tty0 as console. This is the default value if magic ≠ RISCPROM and the monitor initializes the NVRAM

”L” : - the monitor tries to use the graphic channel as console or tries using tty0, tty1, if the graphic channel can’t be opened

”C” : - only the graphic channel will be used

bootmode

”m”: - the monitor runs all the diagnostic tests after power up and sets boot mode to ”e”

”e” : - the monitor clears the memory after reset and boots without tests

”d” : - the memory will not be cleared and no tests will be carried out during boot procedure

eth_phys_addr

This parameter is the physical ethernet address. It can only be changed if the programmable device ”CU08Sx02” is inserted. This device may only be available to the service staff. It must be replaced by ”CU08Mx02” before delivering.

It is absolutely necessary to keep book of all customers and their physical ethernet addresses. A customer must keep his ethernet address even if the CPU or the NVRAM have to be replaced.

2. 2. 1. 2. Firmware

Version CU08 of the CCU needs a firmware different from that on the R3000 CCU’s CU04 through CU07.

The firmware prom’s of CU08 should be labeled with

- 46Hyymmdd and 46Lyymmdd

where ”46” means R4600, ”H” or ”L” mean high or low bytes,and the rest represents the version as date.

This firmware can not used on former CCU versions.

The firmware prom’s are listed in the PAL part list.

2. 2. 1. 3. Boot Prom

The processor R4600 reads some internal operational settings out of a serial boot prom at power up.
The boot prom volume is 256 bits. The first 15 bits are valid. The rest is reserved and have to be clamped to zero.

Serial Bit	Meaning
0	Reserved and zero
1 to 4	Block write data mode and ratio
5 to 7	Clock ratio between internal and external clock
8	Endian ordering
9 to 10	Non-block write handling
11	Enable of timer interrupt
12	Reserved and zero
13 to 14	Output driving slew rate
15 to 255	Reserved and zero

The firmware boot prom is listed in the PAL part list.

2. 2. 1. 4. Assembling

Network connection

Connecting to a network can be carried out by the Ethernet AUI connector ST4 or a Cheapernet BNC connector. Both solutions need there proper jumper setting (W2,...,W8), there special front panel and assembling:

- ST4 has got its own foot print on the PCB
- The Cheaprnet BNC connector has to be connected to the foot print of W9 with signal at pin 1 and shield at pin 2

2. 2. 2. Modifications of the introduced layout

2. 2. 3. Service Informations

- The ethernet address starts with:
  0x0000b5(0010)08
  counting upward the part between the brackets.
- CU08 needs the Unix version for R4600 available on ”Boot Tape” not befor version ”950701”

2. 2. 3. 1. How to replace CCU’s by CU08

- All VME bus memories should be removed or their addresses have to be shifted to the range above the on board DRAM.
- Any other device used with CU04 through CU07 need no modifications
- Install Unix version for R4600 available on ”Boot Tape” not befor version ”950701”

2. 3. History of Modifications

EC No.	Date	Part Number	Description of Bugs, Changes and Modifications	Ser.No.	New EC-Level
2141	13.4.95	H2570	Introduction of Layout H3P2130A	901	30
2213	4.10.95	H2570	Replace Mach 210-15 to 210-7 and inatall Latch feedback	935	31
2221	8.11.95	H2570	Inserting missed terminator on BCLK6	971	32
2241	23.1.96	H2570	Allowing use of SIMM with quad CAS chips	1063	33

3. Testing

3. 1. Basic CCU Test

The basic test procedure which is equivalent to that of the former CCU versions is available in the prom devices

- T46H950412
- T46L950412

This test tries writes and reads to the onboard devices and to memory address 0x0 and writes after each successful device test a new value to the LED display. An unsuccessful action will be repeated in a loop.The test sequence and the LED values are as follows:

LED	Action
1	Prom read and LED write O.K.
2	Ethernet read/write O.K.
3	Timer read/write O.K.
4	NVRAM read/write O.K.
5	Memory read/write at address 0x00, wordwise O.K.
6	RS232 device read/write O.K.

The test now does a complete memory test from 0 to 16 megabyte and prints messages on tty0.

7	Memory failure during test or Exeption happened
8	Test finished successfully

You can now use the monitor proms with its own test procedures.

3. 2. Testprograms of AQX devices

3. 2. 1. Usage

3. 2. 1. 1. Where to use the testprograms

On AMX spectrometers (`amx, arx, asx`**)**

`aqtest`	Tests the AQI interface to Aspect3000, Aspect 30001
`gctest`	Tests the Gradient Controller
`gcutest`	Tests the Gradient Controller

On spectrometers of the DMX series (`dmx, drx, dsx`**)**

`fcutest`	FCU test (frequency control unit)
`tcutest`	TCU test (timing control unit)
`gcutest`	GCU test (gradient control unit)
`rcutest`	RCU test (receiver control unit)

On all spectrometers

`memtest`	Memory test. This test runs only stand alone
`siotest`	Tests the serial interfaces on the CCU and the SIO board. This test runs only under UNIX

Note: If the board to be tested is not present, the test will print an error message and exit.
The `gcutest` decides by itself which hardware is available and has to be tested.

3. 2. 1. 2. How to start a test program

`device` has to be specified as a choice out of the following device names `fcu, tcu, gcu, rcu, gc, aq, mem, sio`
The test programs have to be started on the AQX CCU of the spectrometer. Otherwise it warns you and exits. To log in at the spectrometer enter

```
telnet spect
root
```

Start a test using UNIX with

```
cd /u/systest/device
./devicetest
```

During execution the *`device`*`.firm` is loaded to the board or device under test and executed by the local processor. To run a test stand alone (without UNIX) shutdown the CCU with

```
/etc/init 5
```

On the console which is connected to the CCU enter

```
boot -f bfs()/usr/diskless/clients/spect/root
                    \/u/systest/device/devicetestsa
```

Exception: `memtest` is started standalone without the extension `sa`
`siotest` cannot be started standalone

Normally you should enter `auto`, when the testprogram prompts you for an input

3. 2. 1. 3. Special files used by the test programs

To use the driver and the full functionality of the test programs it is necessary that the following special files of each device had been created and are avalable:

File name	major#
/dev/AQI	55
/dev/gc	56
/dev/rcu	59
/dev/fcu	60
/dev/tcu	51

Such a special file is created with `mknod`, for example:

```
mknod /dev/AQI c 55 0
```

The major number can be checked with :

```
ls -l /dev/aq
```

```
crw -rw -rw 1 root bin 55 0 Jun14 1993 /dev/aq
```

3. 2. 1. 4. Main features of the test programs

1. Get program version

Start the test program with:

```
devicetest -v
```

The test will print its version number and exits.

Note: This paper applies to program version 950901.1 and the newer ones

2. auto-command

Start the test and enter the command `auto`. All tests are executed automatically. Errors found are printed on your terminal and listed in the file

```
/u/systest/device/errorfile
```

This error file is rewritten each time you exit and restart the test program.

3. help-command

When you enter `h`, you will get a list of all available commands with a short description.

4. protocol

When you enter the command `prot` for the first time, all subsequent input and output is written into a protocol file until you enter `prot` for the second time. You can write several protocol files while the test is running. The name is to your choice.

5. command file

Instead of entering commands directly to the test, you can put them into a file, then start with:

```
devicetest -c cmd
```

where `cmd` is the name of that file. The test program will execute the commands and if the last command is not `quit` or `q` it will continue with reading more commands from the keyboard.

6. shell

With the command `sh` you get a shell without leaving the test program. You can exit that shell and return to the test program by entering `exit` or `crtl-d`. This feature does not work in the stand alone tests.

7. terminate the test program

If you leave the test program by the commands `q` or `quit` , the program resets the i960 on this board (if there is one) and restores registers that may have been modified during the test. If you leave with the command `l` , nothing is changed or reset.

8. loops

The most tests can be started in a loop. See the section titled ‘‘parameter setting’’.

9. registers

The names of on-board registers can be found with the command `rname`. An information for each register is given with the command `rinfo`.

10. debug print's

The accesses by the CPU or i960 to memory can be made visible by the command `sw` (switch). The second time `sw` is used, it makes the accesses invisible.

11. DELETE

Any command can be interrupted with `DELETE`. This feature may be delayed in stand alone programs.

Note: If the i960 is just executing a command, only the program running on the main CPU notices your `DELETE`. Before the i960 can execute a new command, you must reset it.

12. execution of a command

At first the processor will be started, if the command has to be executed on the i960. Then the user is asked for the necessary parameters. If necessary, they are transferred into i960-memory. During an execution of a command by the i960, the CPU polls the i960-memory to check for completion. All communication is done via the mail box located at offset 0x3600 in the i960-memory.

For RCU and AQ, the physical page addresses for the VME-memory to be used with a DMA start at offset 0x4000 in i960-memory.

13. Load (and execute) another program

Use the command `load`, then enter the name of the program to be loaded to the i960-memory. All subsequent commands for the i960 will load and use this program. You can directly start it with the command `run`.

14. List these manual pages

Enter the command

```
man
```

to the test and select amanual page.
It will be listed on the screen and can be saved in a file.

3. 2. 1. 5. Parameter setting

Defaults

Each value or string of the console print out written in brackets [] is a default setting. If you enter `RETURN`, this default value is kept and not modified. Use `gpar` to get the values of all available paramters. Use `spar` to set them (or part of them).

`start/lstart` If there is an i960 on the board, `start` is the VME-address used by the CPU. `lstart` is the corresponding local start address used by the i960. If you enter the test start address by `spar`, you must always use the local address.

`number of loops` Affects memory tests, register tests, read and write memory, read and write registers, and board specific tests.

Test `mode` `mode` is for memory tests started on the i960.
`f`: read/write words forward
`r`: read/write words in reverse order

q: read/write quad words forward
s: read/write quad words in reverse order

`continue on error` If this parameter is set and an error is found, the tests prints out the error message and continues. The total error count is printed out when the whole test finished. If this parameter is not set, the test terminates after the first error has been found.

`print on mem-access` Use the command `sw` to switch on/off printing on memory access.

To switch on for CPU-memory accesses enter:

```
sw
c
```

To switch on for i960-memory accesses enter:

```
sw
l
```

3. 2. 1. 6. Overview of tests

Device memory test executed by the CCU

(a) `tim`	test instruction memory
(b) `tdm`	test data memory (if present)
(c) `tcm`	test combox memory (if present)
(d) `tms`	test memory and set param's
(e) `tmv`	test memory with value
(f) `tmiv`	test memory with incr. value

(a), (b) and (c) test the whole memory region present. (d), (e) and (f) use the parameters for start address and size which have to be set before with the command "`spar`".

(e) tests with one constant value set by `spar`,

(f) increments this value during the test.

(a) - (d) test in subsequent passes with the following values :

1.Pass: 0
2.Pass: 1, 2, 4, 0x10, ..., 0x80000000
3.Pass: value == address
4.Pass: value incremented by 0x10001
5.Pass: -1
6.Pass: 0
7.Pass: 0xaaaaaaaa and 0x55555555 alternativly

Device memory test executed by the local processor (i960)

These tests are not applicable on the FCU's.

(a) `timl`	test instruction memory local
(b) `tdml`	test data memory local

(c) `tcml` test combox memory local

(d) `tmsl` test mem, set param's local

(e) `tmvl` test memory with value local

These commands operate in the same manner, except that the i960 instead of the CCU accesses the device memory.

Register tests

(a) `tr` The registers are accessed by the CPU

(b) `trl` The registers are accessed by the i960.

Parameters:

`Name`: Select a register name or enter `all`.
If you enter `all`, all registers are tested for which this is possible.

`Value`: Select a number in hexadecimal, ‘‘`bits`’’ or ‘‘`all`’’.
If you enter ‘‘`bits`’’, the register will be tested with the values 1, 2, 4, 0x10, ...
If you enter ‘‘`all`’’, the register will be tested with the values 0, 1, 2, 3, 4, ...

Interrupt tests

(a) `int` Interrupt from i960 to CPU

(b) `intl` Interrupt(s) from CPU to i960

Basic tests for the i960

If the `auto` command in any test running on the local i960 does not work properly check the following basic functions:

1. Reset the i960

   `res`

2. Test if the device memory is accessible

   `tim`

3. Load the test program

   `load` *`device`*`test`

4. Start the i960 without any command

   `run`

5. Run a command on the i960

   `hello` (prints ”hello” on the screen)

3. 2. 1. 7. Special RCU test features

1. Test of the DMA-channel 1 (DMA’s to the VME memory):

`rdma`	The CPU fills a buffer in the VME memory, the RCU reads this buffer with DMA and the i960 checks the data
`wdma`	The i960 fills a local buffer, writes the data to the VME memory and the CPU checks the data
`tdma`	the i960 fills a first buffer, writes the data to the VME memory, reads them back to a second buffer and compares their contents

Seven different types of DMA transfers will be carried out in each loop of these commands

1.pass: offset 0, increment 0
2.pass: offset 0, increment 1
3. pass: offset 0x1234, increment 0
4. pass: offset 0xf3c, increment 1
5. pass: offset 0xa50, increment 0
6. pass: offset 0x5a0, increment 2
7. pass: offset 0, increment 3 and splitted regions

`Offset` refers to the offset in a page of VME-memory `Increment` refers to the pattern written into the memory. Splitted regions means that the i960-memory is not consecutive. Each type of DMA-transfer is performed 16 times in one loop. These parameters can be changed with the command `dpar`.

2. Parallel testing of the DMA-channels 1 and 2:

   `pdma`

   The first buffer has a size of 0x8000, start value 0x10203040, increment 1, starts at program end + 0x10000 and the DMA uses the first 8 page addresses from VME-memory.
   The second buffer has a size of 0x1330, start value 0x76543210, increment 1, starts at program end + 0x20000 and the DMA uses page 9 and 10 from VME-memory.

3. Test of the FIFO and the DSP's:

`hpci`	All parameters concern the DSP's. The DSP's are loaded with them. The RCU-GO pulse must be given externally, then the FIFO starts to copy data into SRAM at the memory behind program end.
`file`	The data copied from FIFO to SRAM are transferred into a file to be used as FID.

Parameters for the `hpci`-command (set with `spar`):

`hpci-frequency`:	transformed and loaded into hpci-divider
`hpci-word-count`:	loaded into pulse-counter and p.c.-preload
`qmode`:	load selected quad mode (one of quad on, quad off, simultane) into gp_out and set strobe in HPC-control respectively
`stop`:	if set, set stop bit in pulse-counter and divider-preload-sec

4. Test how many wait states are required for Region 9 (SRAM)

   `wstate` (For 0 wait states, the new PAL RCU0AB03ZM is required.)

5. Read from memory to file

   `rf`

3. 2. 1. 8. Special TCU test features

1. Wait operation test

   `wait` The CPU fills 4-PORT RAM with the instructions `WAIT`, `CLEAR WAIT, NMI` and tests them.

2. Duration test

   `dur`

3. Loop counter test

   `lpcnt` The i960 checks loop counter, decrement counter and unconditional loop back.

4. Address generator test

   `tagen`
   1. Interrupt INT0 Test
   2. Pre-register Test
   3. Address generator Bit Test, value = 0,1,2,4,8...0x100
   3. Address generator value Test, 0 <= value <=0x1ff
   4. Address generator Test with 'Astep' register

5. Blanking register test

   `nmr`

6. Create RCU GO pulse

   `rcugo`

ACQ bus test between TCU-FCU

7. ACQ bus data line test

   `acq` The CPU writes 4-PORT RAM and reads ACQ Bus from FCU register

8. ACQ - FCU Pointer test

   The i960 fills 4-Port Ram with FCU board number 0-7, FCU Pointer 0-255 and data. The CPU reads FCU-Memory pointer and compares them.

   The commands `aqincr`, `aqstep` and `aqreload` consist of the two components: command-name and pointer-number.

`aqfcu`	Increment, reload and step fcu instructions test. (from acq bus)
`aqincr`*nnn*	Fcu increment test (from acq bus)
`aqstep`*nnn*	Fcu step test (from acq bus)
`aqreload`*nnn*	Fcu reload test (from acq bus)
Note:	*nnn* is the FCU pointer number

   Examples:

```
aqincr0
aqstep1
aqreload255
```

ACQ bus test between TCU-GCU

1. Test of some GCU functions initiated by the TCU via the AQ bus

`gcu`	The CPU strts each of these tests below
`ng`	NG pulse test
`rtf`	gcu real-time AQ data test
`aqctl`	AQY Bus - Data and Control flags test
`aqdat`	AQY Bus Data flags test
`aqst`	AQY Bus Interrupt and Status register test
`aqf0`	AQY Bus Control Flag0 test
`aqf1`	AQY Bus Control Flag1 test
`aqint`	AQY Interrupt Flag test

3. 2. 1. 9. Special FCU test features

`conf`	Fcu board and memory configuration test
`fcun`	Set fcu board number
`durg`	duration test
`dds`	Test of DDS interface

   The commands `lddp`, `ldp`, `incr`, `step` and `reload` consist of the two components: command-name and pointer-number

`lddp`*nnn*	load data via pointer test
`ldp` *nnn*	load pointer test

`incr`*nnn*	Fcu increment test
`step`*nnn*	Fcu step test
`reload` *nnn*	Fcu reload test

Examples:

```
lddp0
ldp12
incr255
```

3. 2. 1. 10. Special GC/GCU test features

1. Test of the D/A-Converter:

`dac`	Data are written into `xd`, `yd` or `zd`, then decremented and a Next-Gradient-Pulse is given. This is done in a loop until a lower bound is reached. The start value for data is `0xffff`.

2. Test of the `xd`,... registers

   If these registers are physically not present, a special test mode can be enabled that writes a value to the specified register and reads it back from `rbg` (read back gradient). This is done by entering ”1” to the question ”`read back Gr data from rbg?`”.

   Example:

```
spar
read back Gr data from rgb? (1=yes, 0=no) 1
```

   At program start this parameter is set.

3. 2. 1. 11. Special AQI test features

1. DMA tests between AQI and VME memory of AQX rack:

`rdma`	CPU fills the buffer, i960 reads and checks it.
`wdma`	i960 fills the buffer and writes it, CPU checks it.
`tdma`	i960 fills the first buffer, writes it to VME-memory, reads it to a second buffer in local memory and compares the two buffers.

   The commandos `rdmas`, `wdmas` and `tdmas` operate in the same manner, but use the ”single” mode instead of the ”nibble” mode. That is, each access refers to a 4-byte word. In the ”nibble” mode (`rdma`, `wdma` and `tdma`), each access refers to a 16-byte word.

2. a3000 memory specification for DMA tests between AQI and a3000 memory

   Instead of local memory, a3000-memory can be used in the DMA tests. This is done by setting the parameter ”`start`” to an address in the a3000 address space.

The `auto` test will include DMA tests for the a3000-memory, if the necessary parameter is set to indicate that the a3000 exists and should be tested.
This is done with the command `spar` as follows:

Example:

```
spar
a3000 present? (1=yes, 0=no) 1
```

At program start, this parameter is NOT set.

3. 2. 1. 12. Special SIO test features

1. Start options of the `siotest`

   The siotest can be started with the following options:

   `./siotest [-h] [-q] [-c] [-f filename] [-s]`

`-h`	(help) prints the usage
`-q`	(quiet) supresses some print messages
`-c`	(command) enters a command loop for special tests
`-f`	(file) read the interfaces to be tested from a file named *`filename`*
`-s`	(short_circuit) allows to test with a short circuit instead of a terminal. (The DCD-pin is not tested in this case.)

2. Selecting the present `tty`'s to be tested

   You can only select the following groups of `tty`'s to be tested:

S `tty00` on all CPU versions and ASPECT station

S `tty01 - tty07, tty10, tty20` on CCU version CU05

S `tty01 - tty09, tty10, tty20` on CCU version CU06/ 7/ 8/ 9

S `tty01 - tty03` on the CPU/4,

S `tty01 - tty03` on the ASPECT station

S `tty11 - tty18` on the second ethernet subsystem of the ASPECT station

S `tty11 - tty16` on the SIO-board 1

S `tty21 - tty26` on the SIO-board 2

S `tty31 - tty36` on the SIO-board 3

S `tty41 - tty46` on the SIO-board 4.

Only in the groups tty00-tty09 and tty00-tty03 can each file separately be selected.
The SIO boards can be selected in groups of consecutive bord numbers beginning with bord #1 (for example, if you select SIO-board 1 but not number 2, you cannot select number 3 and 4).

3. The file of interface names under test

   The first line of this file consits of the key word

S `station` (for ASPECT station) or

S `x32` (for CPU/4 and CPU/5)

   The following lines consist of the interface names as

S `/dev/tty`*`ii`* or

S `/dev/ttyn`*`ii`*

   where *`ii`* means the two digits of the interface number. and `n` ”don’t check the DCD pin on open”
   If you use

```
/dev/ttyn ii
```

   then the DCD pin is not checked when the device will be opened.

   For example:

```
x32
/dev/tty03 /dev/tty05 /dev/ttyn11 /dev/tty16
/dev/tty21 /dev/tty22 /dev/tty23
```

4. Loop back connection between interfaces

   To test the interfaces it is necessary to send and receive test data streams.
   Always two neighboured interfaces have to be connected via an ordinary RS232 cable.

   That means `tty00` and `tty01`,
   `tty02` and `tty03`,
   ...
   `tty11` and `tty12`,
   `tty13` and `tty14`,
   ...
   and so on have to be connected.

3. 2. 1. 13. Special MEM test features

1. The memory configuration

   Use `conf` to check the memory configuration

   `conf` prints out the actual memory configuration, the program `start address` and `size`, and the stack start address.
   Actually, the program is loaded to 0x200000 (= 2 Mega).
   On CPU/4 and CCU/5, the stack is at 0x800000 (= 8 Mega).

2. Test commands

   All following tests (except `auto`) use the current parameters `start`, `size` and `value` as set by the user (default is `start=value=0, size=0x40`).

`auto` uses the total memory region found.
In `auto`, the regions are split into blocks of at most 4 Mega,
that are tested one after the other.
`auto` calls the `value` test for 8 different values!

`bit`	Test the region with the `values` 1, 2, 4, 8, 0x10, ... 0x80000000
`value`	Test the region with the parameter `value`.
`incr`	Increment the parameter `value` during filling the buffer.
`pat`	Fill the buffer with the patterns 0, 80001, ... i*80001, ...
`comp`	Fill the buffer with `values` 0xaaaaaaaa and 0x55555555.
`addrw`	Set `value=address` for each `address` and write it to the address as a 32-bit-word.
`addrb`	Set `value=address` for each address and write it to the address as a byte. Do this for Bytes 0, 1 and 2 of the address.
`copy`	Copy memory regions byte per byte and compare the two regions.
`cache`	Check whether the memory contents of address 0 changes while the cache address 0 is used for writing.
`refr`	Fill the region with 0x5a, then wait 30 seconds and check it. Then fill the region with 0xa5, wait 30 seconds and check it.
`clear`	Write a pattern to one word in a region and clear the rest. Then check each word.
`tm`	Call all these tests with the current parameters.