TRIPUTER

V0.9

User Manual

7.12.2019

Introduction

TRIPUTER V0.9 is a small computer based on the Cyclone V GX Starter Kit. The main purpose is to run NetBSD. The M32632 is the heart of the system running at 50 MHz. The actual version needs around 10,000 of the 29,080 ALMs (34%) in the FPGA. (ALM = adaptive logic module)

First the FPGA has to be configured. This is done by the software Quartus Web Edition. Preferable is version 13.1*) or higher. The configuration needs no project definition. All information is contained in the SOF file or POF file. SOF files configure the SRAM cells in the FPGA directly. This is useful for testing a new hardware. POF is written to a flash device which configures the FPGA at power-up.

TRIPUTER V0.9 uses a terminal for user I/O. The PERI board has a true RS232 interface. Another connection is based on USB. My host is a PC running Windows 7. To enable the USB interface on the FPGA board a driver is needed. The driver is for the FT232R chip and can be downloaded from the website of FTDI (www.ftdichip.com).

I use PuTTY on my PC. The parameters of the transmission are 115,200 baud, 8 data bits, one stop bit and no parity. The software is easy to use. It has a function for downloading a text file without handshaking. This is used to download a program or data to TRIPUTER V0.9 . Upload is currently not available and must be programmed by the user.

*) Quartus 13.1 needs the service pack 4 to function properly.

Hardware

Definitions:
0 : read as "0"
x : read value is unknown
Mem : internal memory of the FPGA
RAM : Read/Write Memory
ROM : Read Only Memory, made of FPGA RAM
Reg : Register , always 4 Bytes wide
R/M : Register/Memory
All addresses are in hexadecimal notation if not otherwise defined.

Reset behavior:
After RESET the CPU starts program execution at address 00000000. To make a defined start the ROM is accessed instead of the DRAM.

This behavior is changed to normal mode by an instruction fetch to an address >= E0000000. The JUMP instruction can be used for this:

```
00000000        JUMP @x'E0000010 ; first opcode in ROM
```

Coding the target address as a displacement is possible.

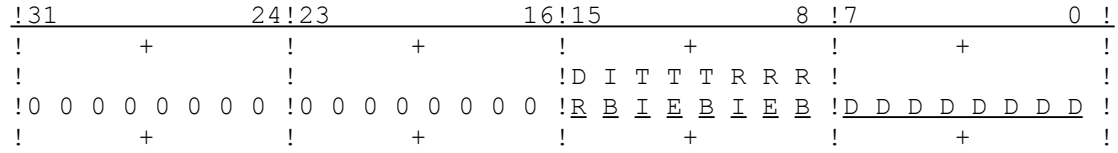
The RESET button is KEY4 = CPU_RESET.

Memory Map:

Address range	Type	Description
00000000..1FFFFFFF	DRAM	512MB external LPDDR2 DRAM, see Reset behavior
E0000000..E0000FFF	ROM	4KB internal MONITOR program
E0001000	Reg	UART : Serial Interface
E0002000:R	Reg	SWK : Slide Switches + Buttons
E0002000:W	Reg	LERG : Red and Green LEDs
E0002004:W	Reg	LESS : Seven Segment LEDs
E0003000:W	Reg	RC_CTRL : Read configuration control register
E0003000..E0003FFF	Mem	RC_MEM : Read Configuration data memory
E0004000	Reg	I2C_C : Control register
E0004800..E0004FFF	Mem	I2C_FIFO : I2C FIFO
E0005000	Reg	COUNT : NMI Counter
E0005004	Reg	NECO : NMIE + NMI Counter
E0006000..E000600F	Reg	Graphic & Terminal control registers
E0006100..E00061FF	Mem	Cursor definition
E0006800..E0006C16	Mem	Color table
E0007000..E0007FFF	Mem	TERM_FIFO : Terminal FIFO
E0008000..E000800F	Reg	SD Card control and status registers
E0008100..E0008107	Reg	SD Card status registers
E0008400..E000847C	R/M	SD Card control register and response memory
E0009000..E000900F	Reg	PERI SD Card control and status registers
E0009100..E0009107	Reg	PERI SD Card status registers
E0009400..E000947C	R/M	PERI SD Card control register and response memory
E000A000..E000AFFF	R/M	Ethernet Interface
E000B000..E000BFFF	Reg	PS2 (Mouse and Keyboard)
E000F000..E000FFFF	Reg	reserved (configuration switch)
E1000000..E107FFFF	SRAM	512KB external 16-bit SRAM
E2000000..E2FFFFFF	Mem	SDC_FIFO : SD Card interface fifo port
E3000000..E3FFFFFF	Mem	PERI SDC_FIFO : SD Card interface fifo port
FFFFFE00..FFFFFE1F	Reg	Interrupt Control Unit : NS32202 ICU

Register UART

Address : x'E000_1000



ID	Name	Bits	Access	Description
DR	DRS	15	R/W	Disable RS232 of PERI: "1" = disable PERI RS232
IB	INB	14	R	Receiver In bit: input of the receiver
TI	TXI	13	R/W	TX Interrupt: set to "1" if transmission finished
TE	TXIE	12	R/W	TX Interrupt Enable: enables interrupt if TXI="1"
TB	TXB	11	R	TX Busy: "1" if transmitter is busy
RI	RXI	10	R/W	RX Interrupt: set to "1" if new data is available
RE	RXIE	9	R/W	RX Interrupt Enable: enables interrupt if RXI="1"
RB	RXB	8	R	RX Busy: "1" if receiver is busy
D	DATA	7:0	R/W	DATA[7:0]: 8-bit data to send or to read. A write starts a transmission and clears TXI. Read data clears the RXI.

Reset
DR = TXI = TXIE = TXB = RXI = RXIE = RXB = 0

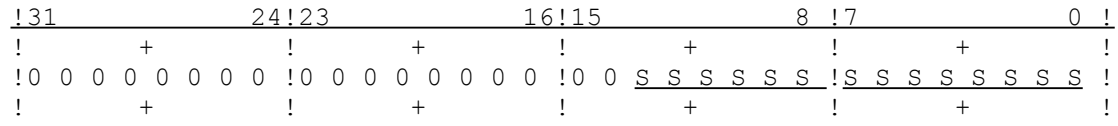
Remark
TXI uses INT 12 of the ICU, RXI uses INT 11 of the ICU. The lower number has the higher priority.

No options are available. The baudrate is fixed at 115,200 baud. The transmission format is fixed to 8-N-1. The BREAK condition can be tested with the help of the IB bit.

Please note that the PERI RS232 port works in parallel to the USB communication. The same information is sent out on both ports and input from both ports is feed into the UART.

Register SWK

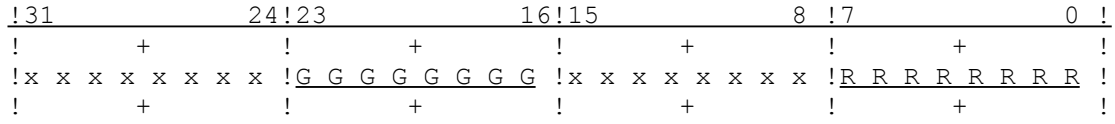
Address : x'E000_2000



ID	Name	Bits	Access	Description
S	SWITCH	13:0	R	SWITCH[13:0]: the 10 slide switches named from SW9 to SW0 = S[9:0] and the 4 buttons named from KEY3 to KEY0 = S[13:10] of the Cyclone V GX GX Starter Kit.

Register LERG

Address : x'E000_2000



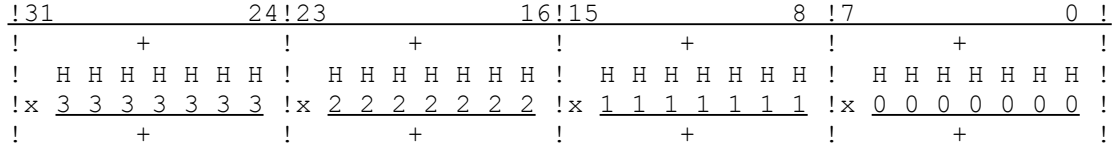
ID	Name	Bits	Access	Description
G	LEDG	23:16	W	LEDG[7:0]: the 8 green LEDs named LEDG0 to LEDG7 of the Cyclone V GX Starter Kit.
R	LEDR	7:0	W	LEDR[7:0]: the 8 red LEDs named LEDR0 to LEDR7 of the Cyclone V GX Starter Kit.

Reset
no action

Remark
The LED named LEDR9 is used by the hardware to show any problems with the DRAM interface. If you ever notice that this LED is blinking please call the service hotline. LEDR8 shows an access of the SD card.

Register LESS

Address : x'E000_2004



ID	Name	Bits	Access	Description
H3	LESS3	30:24	W	LESS3[6:0]: the seven segment LED named HEX3
H2	LESS2	22:16	W	LESS2[6:0]: the seven segment LED named HEX2
H1	LESS1	14:8	W	LESS1[6:0]: the seven segment LED named HEX1
H0	LESS0	6:0	W	LESS0[6:0]: the seven segment LED named HEX0

Reset
no action

Remark
The segment 0 is LESSx[0] at the top clockwise up to segment 6 which is LESSx[6]. DP is not available.

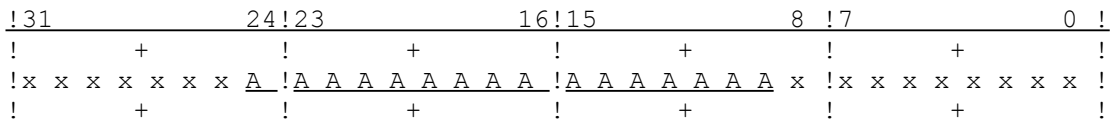
------ Module Read Configuration Memory ------

The starter kit has a flash memory to load the configuration of the FPGA during power up. The flash is much bigger (32 MB) than the FPGA needs. Therefore the rest is free for the user. TRIPUTER V0.9 stores the monitor program of NetBSD (32 KB) at address x'600000.

The module contains a memory block of 1 KB. Program access may be overlapped with reading from the flash. Operation of the module is simple: write the desired start address to a register and the hardware will read 512 byte of data.

Register RC_CTRL

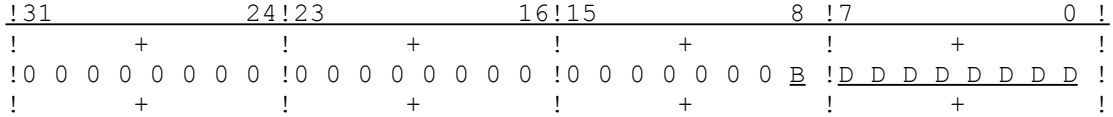
Address : x'E000_3000



ID	Name	Bits	Access	Description
A	ADDRESS	24:9	W	Flash address to access, the lower bits are "0" A[9] selects also the half of the RC_MEM memory where the data read from the flash is stored.

Memory RC_MEM : Read only

Address : x'E000_3000 - x'E000_3FFF



ID	Name	Bits	Access	Description
B	BUSY	8	R	1=Read of flash ongoing, 0=no operation
D	DATA	7:0	R	Flash data, access address must be multiple of 4

The following assembler program is an example for using this module.

```
; This program reads the configuration flash.
; Input parameters:
; R0 contains the number of blocks (each having 512 bytes)
; R1 contains the address to read from Flash
; R2 contains the address where to write the data

rcfg:   save [r3,r4]            ; program entry point
        movd x'E0003000,r4      ; address of RC_CTL = RC_MEM
        movd r1,0(r4)           ; start read
        tbitb 9,r1              ; first access to odd address?
        bfc rcfg1               ; no
        orw x'800,r4            ; yes, change RC_MEM address

rcfg1:  tbitb 8,0(r4)
        bfc rcfg1               ; wait for BUSY to go to "1"

rcfg2:  tbitb 8,0(r4)
        bfs rcfg2               ; wait for BUSY to go to "0"

        cmpqw 1,r0              ; last block read?
        beq rcfg3               ; yes
        addr x'200(r1),r1       ; no, increment pointer
        movd r1,0(r4)           ; start next read
rcfg3:
        movd 512,r3

rcfg4:  movb -4(r4)[r3:d],-1(r2)[r3:b]  ; transfer data
        acbw -1,r3,rcfg4

        addr x'200(r2),r2       ; increment pointer
        xorw x'800,r4
        acbw -1,r0,rcfg2        ; big loop

        restore [r3,r4]
        ret 0
```

------ Module I2C ------

This module controls four external peripherals available on the Starter Kit and PERI. One is the HDMI interface (ADV7513), the second is the Audio interface (SSM2603) and the third is a clock generator (Si5338). On the PERI board is the RTC (DS3232). It is not easy to use today's complex devices and a user has to download and read the documentation. TRIPUTER V0.9 uses the HDMI and program this device at boot time. Its I2C address is x'72.

Register I2C_C

Address : x'E000_4000

```
!31            24!23            16!15             8 !7             0 !
!       +        !       +        !       +        !       +        !
!                !                !    D D C F E E !                !
!x x x x x x x x !x x x x x x x x !x x I O O U M R !D D D D D D D D !
!       +        !       +        !       +        !       +        !
```

ID	Name	Bits	Access	Description
DI	SDA_IN	13	R	SDA pin status
DO	SDA_OUT	12	R	Internal SDA driver status
CO	SCL_OUT	11	R	Internal SCL driver status
FU	FULL	10	R	I2C FIFO full flag
EM	EMPTY	9	R	I2C FIFO empty flag
ER	ERROR	8	R	I2C error flag
D	Data	7:0	R	I2C read data, there is no FIFO for I2C read.

Register I2C_C

Address : x'E000_4000

```
!31            24!23            16!15             8 !7             0 !
!       +        !       +        !       +        !       +        !
!0 0 0 0 0 0 0 0 !0 0 0 0 0 0 0 0 !0 0 0 0 0 0 0 0 !0 0 0 0 0 0 F R !
!       +        !       +        !       +        !       +        !
```

ID	Name	Bits	Access	Description
F	FAST	1	W	Fast bit, if set selects fast operation of I2C: 16 CPU clock cycles per bit, otherwise 64 clock cycles.
R	RUN	0	W	Run bit, if set I2C is active.

Reset
F = R = 0

Memory I2C_FIFO : Write only

Address : x'E000_4800 - x'E000_4FFF

```
!31            24!23            16!15             8 !7             0 !
!       +        !       +        !       +        !       +        !
!                !                !          E B R !                !
!0 0 0 0 0 0 0 0 !0 0 0 0 0 0 0 0 !0 0 0 0 0 N E D !D D D D D D D D !
!       +        !       +        !       +        !       +        !
```

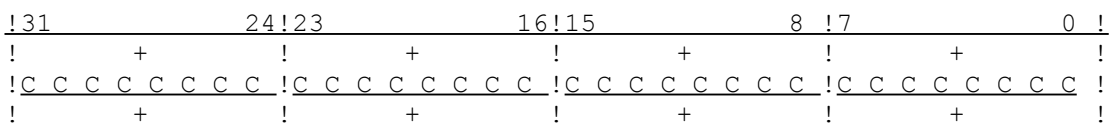
ID	Name	Bits	Access	Description
EN	END	10	W	1=End I2C transmission, STOP condition
BE	BEGIN	9	W	1=Begin I2C transmission, START condition
RD	READ	8	W	1=Read data from device
D	Data	7:0	W	I2C write data

Remark
The FIFO has 128 entries. All bits have to be written in one word. If the FIFO

is full no further writes are possible.

Register COUNT

Address : x'E000_5000

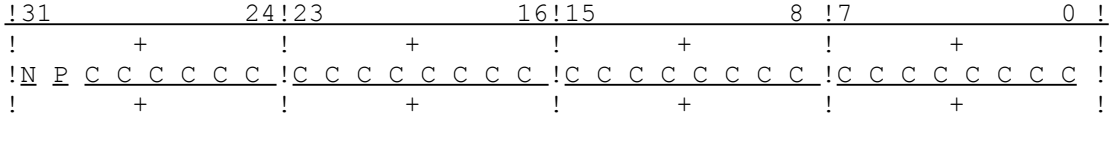


ID	Name	Bits	Access	Description
C	COUNTER	31:0	R	COUNTER[31:0]: counts upward This counter can not be stopped, except in Reset. If NMI is enabled the counter counts up to 49,999,999 and then restarts at 0.

Reset
COUNTER = 32'd0

Register NECO

Address : x'E000_5004



ID	Name	Bits	Access	Description
N	NMIE	31	R/W	NMI Enable: enable NMI if set to "1"
P	PRST	30	R/W	PERI Reset Enable: enable if set to “1”
C	COUNTER	29:0	R	COUNTER[29:0]: same as register COUNT

Reset
NMIE = 0 , PRST = 0 , COUNTER same as in COUNT

------ Module Graphic & Terminal ------

The graphic and terminal interface supports two different operating modes: the graphic mode with a resolution of 1280 by 1024 pixels in four different color modes and a text mode. The text mode shows 64 lines of 128 characters. Control codes are compatible to VT100. These two modes can be used also simultaneously.

The intended monitor is a 19 inch LCD display at 60 Hz frame rate. The pixel clock is 90 MHz. The HDMI output can be connected with the proper cable to a DVI input. For HDTV resolution the required bandwidth is currently too high.

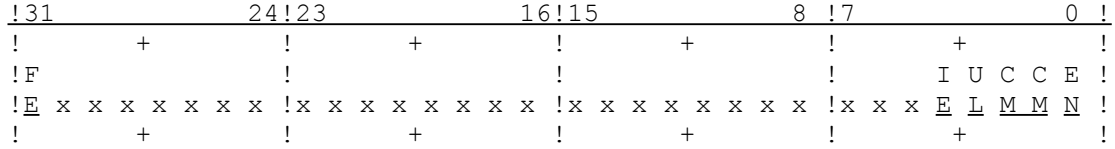
The display data is fetched from the DRAM by DMA. Please note that the 24-bit color mode is using a large part of the available bandwidth of 800 MByte/sec. No user action is required to setup the DMA operation.

The 24-bit color mode uses a compact format in memory. 3 bytes are used for one pixel. A whole line uses 3,840 bytes.

The character set of the terminal is fixed. The character field is 10 pixels wide and has a height of 16 lines.

Register GT_CTRL

Address : x'E000_6000

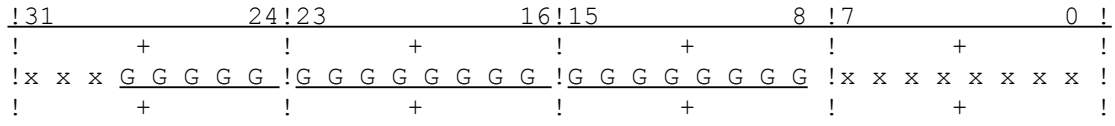


ID	Name	Bits	Access	Description
FE	FRAMEEND	31	R	FE is set at the end of a frame. Software uses this bit to switch to another memory area to display. A read from address x'E000_6004 clears this bit.
IE	INTENA	4	W	Interrupt enable bit for the FE bit (bit 31).
UL	USELUP	3	W	If this bit is set, the color look up table is used.
CM	COLMODE	2:1	W	Color mode: b'00 : 1 bit per pixel b'01 : 8 bit per pixel b'10 : 16 bit per pixel b'11 : 24 bit per pixel
EN	ENABLE	0	R/W	Enable video, this bit must be set if the graphic and terminal unit is used.

Reset
EN = 0

Register GT_GRAF

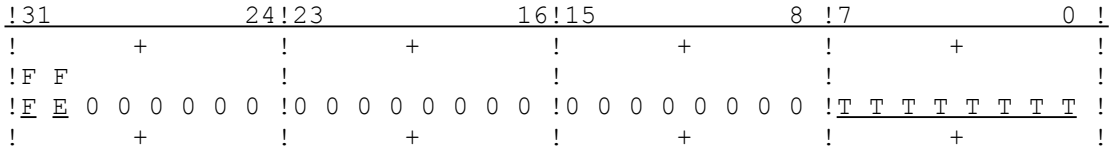
Address : x'E000_6004



ID	Name	Bits	Access	Description
G	GRAFADR	28:8	W	Start address for graphic memory. It can be anywhere in the 512 MB DRAM. The lower 8 bits are always 0.

Register GT_TEXT

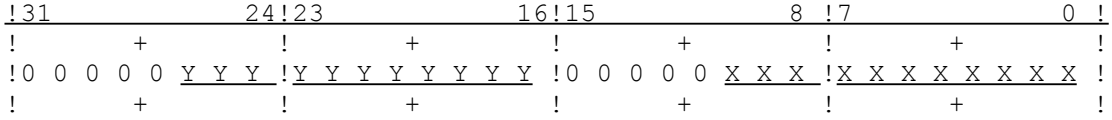
Address : x'E000_6008



ID	Name	Bits	Access	Description
FF	FIFOFUL	31	R	Terminal FIFO is full.
FE	FIFOEMP	30	R	Terminal FIFO is empty.
T	TEXTLIN	7:0	W	Start display text line: valid values are 0 to 63. 0 is at the top of the screen. Any other value will disable the terminal.

Register GT_CURSOR

Address : x'E000_600C

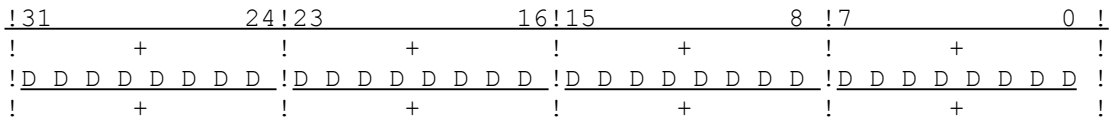


ID	Name	Bits	Access	Description
Y	YPOSI	26:16	W	Y-Position of cursor, 0 is at the top of the screen. Valid values are 0 to 1023, values > 1023 disable the cursor.
X	XPOSI	10:0	W	X-Position of cursor. Valid values are 0 to 1279.

Memory CURSOR : Write only

The cursor is a 32 by 32 pixel wide field which has the highest priority to display if enabled. The upper left pixel is the address in the register GT_CURSOR. Each pixel has 2 bits to use three colors and a transparent mode:
2'b00 : transparent mode
2'b01 : color 1 defined at address x'E000_6C04
2'b10 : color 2 defined at address x'E000_6C08
2'b11 : color 3 defined at address x'E000_6C0C
The 1024 pixels by 2 bits require 256 byte to store. The upper left pixel is defined at the lowest address, bits 1:0 .

Address : x'E000_6100 - x'E000_61FF



ID	Name	Bits	Access	Description
D	DATA	31:0	W	Write data to define cursor.

Memory LOOKUP_TABLE : Write only

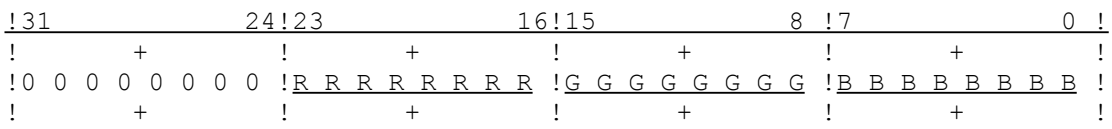
The lookup table is a 24 bits wide memory containing 262 locations. Its use depends on the selected color mode. The 1 bit color mode uses the locations 0 and 1. If no lookup table is used black (0) and white (1) is shown.

8 bit color mode simply selects one of the locations 0 to 255. A 16 bit pixel has three fields used for color definition:
Bits 15:11 used for RED,   lookup location 0 to 31
Bits 10:5 used for GREEN, lookup location 0 to 63
Bits 4:0 used for BLUE, lookup location 0 to 31
24-bit color mode uses three bytes, one for each color. In lookup mode each byte is translated to any other byte.

The locations 256 - 259 are used for the cursor. Locations 260 (background) and 261 (character) are used for the terminal color definitions.

The lookup memory can be written in bytes, words or double-words.

Address : x'E000_6800 - x'E000_6C16



ID	Name	Bits	Access	Description
R	RED	23:16	W	Write data to define RED.
G	GREEN	15:8	W	Write data to define GREEN.
B	BLUE	7:0	W	Write data to define BLUE.

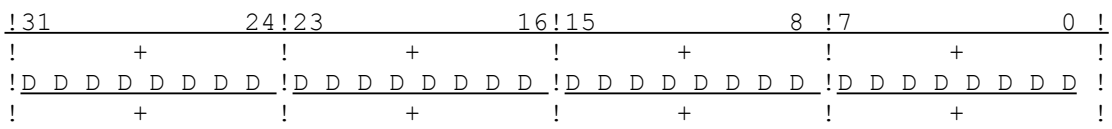
Memory TERM_FIFO : Write only

The terminal FIFO has 1024 byte-wide entries. The speed of the terminal is very high (around 40 Mbaud). Only clear operations will take longer. But it is better to check the FIFO status before writing large amounts of data.

The FIFO can be written with byte, word and double-word in the 4 kbyte address range beginning at x'E000_7000. The byte at the lowest address will be shown first.

The terminal is not able to execute all VT100 command sequences. It is optimized for using NetBSD. If something is missing please inform the author of the user manual.

Address : x'E000_7000 - x'E000_7FFF



ID	Name	Bits	Access	Description
D	DATA	31:0	W	Write data to TERM_FIFO.

------ Module SD Card ------

TRIPUTER V0.9 enables the SD card interface of the starter kit. It uses 4 bit for data transfer. The bus speed is 25 MHz and the transfer rate is 12.5 Mbytes per second. The interface supports at the moment only SDHC cards.

The SDC requires a complex startup procedure. Afterward read and write accesses are more or less easy to implement. TRIPUTER V0.9 has no DMA hardware for data transfer.

Register SDC_STA

Address : x'E000_8000

```
!31             24!23             16!15              8 !7              0 !
!       +        !       +        !       +         !       +         !
!      S S S S S !R C W R C C C C !W R R A           !         A R S   !
!1 1 0 C D D D D !X E D D D D D D !E L S E 0 0 0 0 !0 0 0 0 O F S 0 !
!       +        !       +        !       +         !       +         !
```

ID	Name	Bits	Access	Description
SC	SDCMD	28	R	The status of the SD CMD pin.
SD	SDDAT	27:24	R	The status of the four SD DAT pins.
RX	RXDONE	23	R	If set the response of the device has been received.
CE	CRCERR	22	R	If set the received response has a crc error.
WD	WRDONE	21	R	Write Done: if set data write is done, only set if there is no crc error.
RD	RDDONE	20	R	Read Done: if set data read is done.
CD	CRCDAT	19:16	R	If set a crc error is received during read on the corresponding data pin.
WE	WRERR	15	R	Write Error: if set a crc write error on the data pins was detected in the SD card.
RL	RDLEER	14	R	Read Leer: if set the read buffer is empty.
RS	RESTAT	13	R	Response Status: the response bits 31:19 are not "0"
AE	AUTERR	12	R	Auto Error: if set an error occurred during an APP command
AO	ALLON	3	R(W)	All on: if "1" the SDC interface is active
RF	RUNFL	2	R(W)	Run flag: if "1" the SD clock is running
SS	SLOWSP	1	R(W)	Slow speed: if "1" the slow clock is used (only during initialization)

Remark
The bits ALLON, RUNFL and SLOWSP can be set in the register SDC_CTL.

Registers SDC_COUW & SDC_COUR

The registers count the number of read and written blocks of the SDC interface.

Address : x'E000_8100 = SDC_COUR, number of blocks read
Address : x'E000_8104 = SDC_COUW, number of blocks written

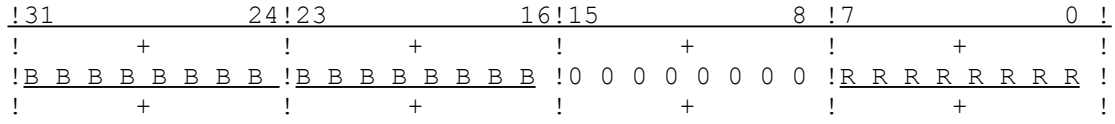
```
!31             24!23             16!15              8 !7              0 !
!       +        !       +        !       +         !       +         !
!0 0 0 0 0 0 C C !C C C C C C C C !C C C C C C C C !C C C C C C C C !
!       +        !       +        !       +         !       +         !
```

ID	Name	Bits	Access	Description
C	COUNT	25:0	R	The 26 bits wide counter value

Remark
The chip reset doesn't reset the registers. Use SDC_CTRL for this purpose.

Register SDC_BM

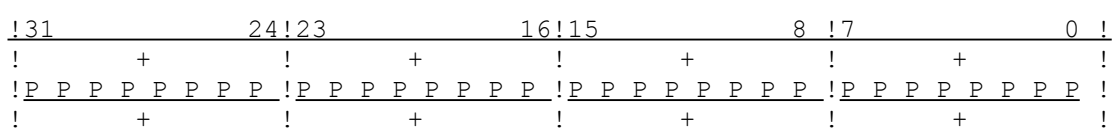
Address : x'E000_8400



ID	Name	Bits	Access	Description
B	BLKCOU	31:16	R	The actual number of blocks to read or write
R	RESDAT	7:0	R	The response data memory: address bits 7:2 select one of 32 bytes received although the maximum of SD response is 136 bits = 18 bytes.

Register SDC_PAR

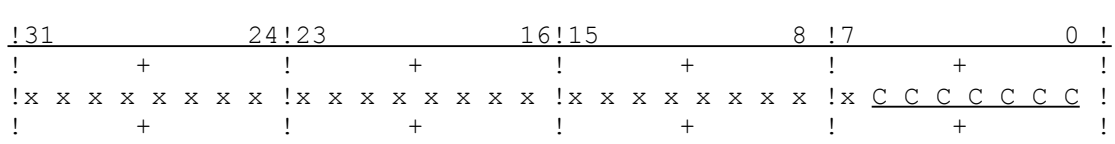
Address : x'E000_8000



ID	Name	Bits	Access	Description
P	PARA	31:0	W	Parameter for a SD card command. This register can be written with bytes, words or double-word.

Register SDC_CMD

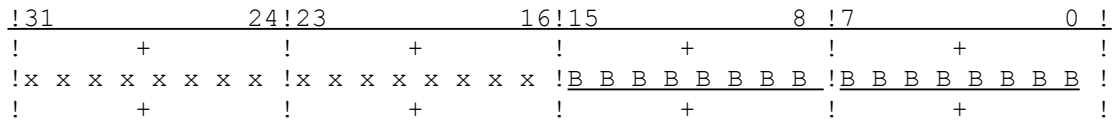
Address : x'E000_8004



ID	Name	Bits	Access	Description
C	CMD	6:0	W	The command number is given in bits 5:0. If bit 6 is set an APP command is executed (two SD commands are executed together).

Register SDC_BLO

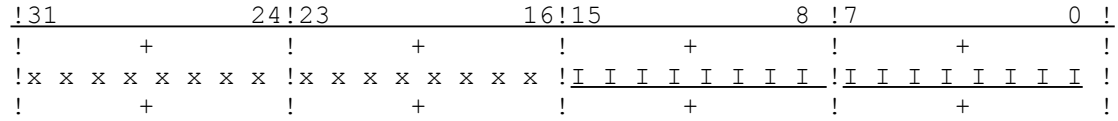
Address : x'E000_8008



ID	Name	Bits	Access	Description
B	BLOCK	15:0	W	The number of blocks to read or write is set in this register: 1 to 65535 is allowed.

Register SDC_ID

Address : x'E000_800C

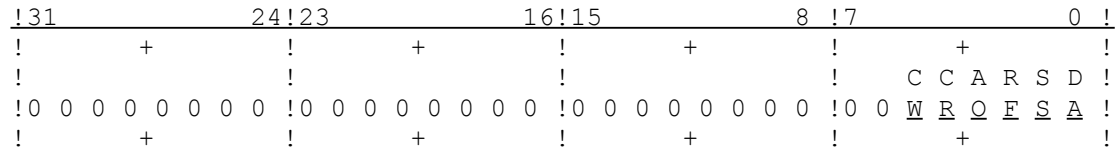


ID	Name	Bits	Access	Description
I	IDNUM	15:0	W	The ID number is set in this register. The ID is determined during initialization and later used by certain commands, for example APP command.

Register SDC_CTRL

The control register has a special write mode: a write access only modifies the bits which are set to "1" during the write access. The bit 0 determines which value is stored in the activated control bit. For example the byte value x'0E set the control bits 3,2 and 1 to "0".

Address : x'E000_8400



ID	Name	Bits	Access	Description
CW	CLEARW	5	W	If "1" set the SDC_COUW register to 0
CR	CLEARR	4	W	If "1" set the SDC_COUR register to 0
AO	ALLON	3	W(R)	All on: if "1" the SDC interface is active
RF	RUNFL	2	W(R)	Run flag: if "1" the SD clock is running
SS	SLOWSP	1	W(R)	Slow speed: if "1" the slow clock is used (only during initialization)
DA	DATA	0	W	Data bit

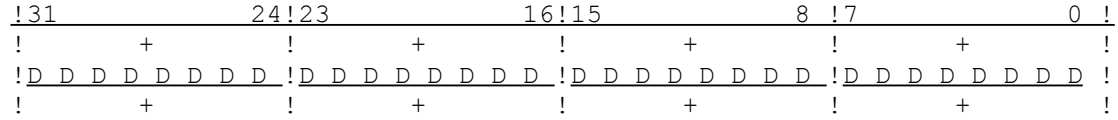
Reset
ALLON = RUNFL = SLOWSP = 0

Remark
The bits ALLON, RUNFL and SLOWSP can be read in register SDC_STA.

Memory SDC_FIFO

The SDC module contains a write FIFO of 2 KB and a read FIFO of 1 KB. The access of the FIFOs is very flexible and can use byte, word or double-word. It is important to note that an access to an empty read FIFO will halt the CPU! The same is true if the write FIFO gets full. The SD card is a serial device by definition and this means that it will send or receive data until the host sends a stop command. Therefore I have reserved a huge address area (16 MB) to cover this behavior with the MOVS opcode of Series 32000.

Address : x'E200_0000 - x'E2FF_FFFF



ID	Name	Bits	Access	Description
D	DATA	31:0	R/W	Read or Write data to SDC FIFOs.

Remark
Inside the address range the address is not used.

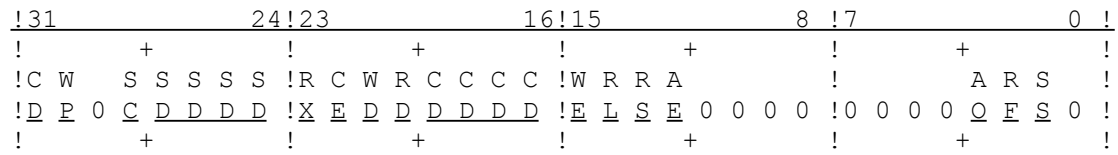
------ Module PERI SD Card ------

The PERI board has an additional SD Card interface. The functionality is identical to the one on the starter kit.

The only difference is that the PERI version reads the pins “card detect” and “write protect”. This information can be found in the register PERI SDC_STA.

Register PERI SDC_STA

Address : x'E000_9000



ID	Name	Bits	Access	Description
CD	CADET	31	R	Card Detect: “1” if a card is inserted
WP	WRPRO	30	R	Write Protect: if this bit is “1” a write should not be done. This is currently managed by software only.

For the other bits see the register SDC_STA.

------ Module PERI Ethernet ------

The Ethernet interface was developed in 2012. Today I have to re-engineer my own verilog code. To use it with NetBSD I have to fully understand it again. Then a description will be written.

------ Module PERI PS2 ------

The PERI board two PS2 ports, one for a keyboard and the other for a mouse. They are used now with old legacy devices. USB is in my view to difficult to implement. Maybe one day in the future I will do it. But this requires also a new PERI board then.

Two registers are provided to read and write to each port. The register at address x'E000_B000 is associated with the keyboard (pink input at the connector). The register at address x'E000_B010 is associated with the mouse (green input at the connector). But the functionality is identical and may be exchanged.

Register PERI PS2_REG

Address : x'E000_B000
Address : x'E000_B010

```
!31            24!23            16!15             8 !7             0 !
!       +        !       +        !       +        !       +        !
!                !                !C D R A N P C D !                !
!0 0 0 0 0 0 0 0 !0 0 0 0 0 0 0 0 !O O U C D A I I !D D D D D D D D !
!       +        !       +        !       +        !       +        !
```

ID	Name	Bits	Access	Description
CO	CLKOUT	15	R	Clock Output: signal of TRIPUTER at the port
DO	DATOUT	14	R	Data Output: signal of TRIPUTER at the port
RU	RUNTX	13	R	Run Transmitter: "1" if the transmitter is running
AC	ACK	12	R	Acknowledge of the external device
ND	NEWDAT	11	R	New data in the receiver, used as interrupt signal, cleared by writing to address+4
PA	PARI	10	R	Parity of the received data
CI	CLKIN	9	R	Clock Input: the input signal at the port
DI	DATIN	8	R	Data Input: the input signal at the port
D	DATA	7:0	R/W	Data: write to send data to the device and read if data is received from the device

Remark
The interrupt output of the keyboard/mouse is connected to INT 13/INT 14 of the ICU.

Please note that the signals CLOCK and DATA at the port are bidirectional by wired-and and therefore both the driving and the sensing status are available in the PS2_REG register.

------ Module NS32202 ICU ------

TRIPUTER V0.9 uses the NS32202 ICU of the PC532. Unfortunately this device is too complex to be described here. Please read the data sheet which is available at www.cpu-ns32k.net . TRIPUTER uses the interrupt 8 for the graphic interrupt All interrupts are active high. Edge triggering on the rising edge or level triggering on the high level is possible. A 5 MHz clock is available for the internal counters. The prescaler of the ICU cannot be used with this clock.

Software

TRIPUTER V0.9 uses a simple monitor program called MONITOR. It provides six commands: LOAD, RUN, DUMP, NETBSD, + and - . Only the first letter of LOAD, RUN, DUMP and NETBSD is necessary. Commands and addresses are not case sensitive. Addresses are entered as hexadecimal numbers.

MONITOR displays a second clock on the seven segment LEDs. If a character is received by the UART the value is shown on the red LEDs and the index in the buffer on the green LEDs. Heavy flashing occurs during a download.

At boot time the MONITOR initializes the ADV7513. Therefore application programs can immediately use the graphic and the terminal.

The content of the ROM can be replaced by any other software on request.

Register Definitions:

```
SP0       Stack Pointer     00000300
INTBASE   Interrupt Base    00000300
SB        Static Base       00000400
All other registers are not used.
```

Memory Map:

```
00000200..000002FF  Stack Area
00000300..0000037F  Interrupt Vector Area
00000380..000004FF  Static Base Area
00000500..000005FF  Serial Interface Input Area
```

Commands	<CR>	Description
L(OAD) address	YES	LOAD downloads bytes from the host system. The format is Intel Hex without any flow control. LOAD shows after each line received the total number of bytes received.
R(UN) address	YES	RUN executes a program. It uses the instruction "BSR addr" to start the program. The instruction "RET 0" is used at the end of the program to return to MONITOR.
D(UMP) address	YES	DUMP displays 16 lines of 16 bytes each. Each line shows the address, the bytes in hexadecimal form and a alphanumerical interpretation.
N(ETBSD) modus	YES	NETBSD starts NetBSD. "modus" can be a two-digit number. The right number (or only one digit) selects the output device: modus 1 : output => hardware-terminal (HDMI) modus 2 : output => host modus 3 : output => host and hardware-terminal (HDMI) The left number selects the base address in GByte for the image to be used on the SDHC card. Only values of 0,2,4,6,8,A,C and E are allowed.
+	NO	performs a DUMP with the last DUMP address + 256.
-	NO	performs a DUMP with the last DUMP address - 256.

The command letters can be also lower-case.

N(ETBSD) requires the monitor program of the PC532. There are two possibilities: either it is already contained in the configuration flash or it has to be loaded manually in the SRAM. For this please enter the download command L(OAD) and use the address E1000000. The pof file for TRIPUTER V0.9 contains the monitor program.

To work with NetBSD it is necessary to load an image on the SD card. Two images are available at www.cpu-ns32K.net/NetBSD.html . The size of one is 1.5 GB and the size of the other is 0.5 GB. Both can be loaded on the SD card. The start address in block numbers is either 0 or a multiple of 0x400000 (= 2GB). Only SDHC cards are supported.

As an example for an Intel hex file the first and the last line is shown:

```
:0800000001020304050607 08D4
:00000001FF
```

NetBSD

TRIPUTER V0.3 has implemented the first guest system: the PC532 running NetBSD. TRIPUTER can be at any time only in one mode of operation: either TRIPUTER or PC532. Some hardware characteristics are different in both systems. For example in PC532 mode the external SRAM became a ROM to store the monitor program of PC532.

It is not the purpose of this manual to describe the features of the guest systems. For all the planned systems this is too much work for one person. I assume that the user knows the basics of the system of interest.

After the start of NetBSD with "N modus" you will see the output of the PC532 monitor program. Enter "boot" to start NetBSD. The next stop requires just <CR>. Then the boot procedure continues. At the end a username is requested. "root" is always available with no password. If you want to stop the system please use "halt".

The PC532 uses a SCSI drive as the mass storage device. TRIPUTER uses an SD card for this purpose. The problem with SD cards is their reliability. NetBSD is writing very often small information packets to the mass storage. Even if you do not enter a program and compile it there will be modifications of the SD content. Therefore it is strongly recommended to make a backup from time to time.

Please send me a note if you see problems with your SD card: info@cpu-ns32k.net Card size and card type and how long it was in use are of interest. I will collect such information and make a statistic which cards are best to use.

There are two choices to see the text output of NetBSD. One is the host system runing a terminal program like putty ("n 2") and the other is the hardwired VT100 terminal inside the FPGA of TRIPUTER ("n 1"). To use this you have to connect a 19 inch LCD monitor to the HDMI port.

With the last one you get a very fast terminal with a large screen of 64 rows by 128 columns. Please use "stty rows 64" and "stty columns 128" to inform NetBSD about the new screen size.

It is possible to see both: select "n 3". This is okay for just looking around in the system. But if you opens vi for example the screen output is not the optimum for one display or the other.

Although TRIPUTER has 512 MB of DRAM NetBSD will only use 256 MB. The reason is the location of the monitor program at x'1000_0000. The monitor can not be shifted to another location without recompilation. For this task a software specialist is needed.

The upper 128 MB of DRAM is available in the range of x'1800_0000 to x'1FFF_FFFF. This range maybe used for graphics memory. Currently the graphic cannot be used due to software restrictions. NetBSD does not allow to write in this memory. Maybe a software driver is needed for this purpose.

TITAN5

TITAN is my own software for my NS32000 systems. The first version TITAN1 was developed on a NS32016 machine in the beginning of the 1990's. Due to the compatibility of the processors, it was easy to port TITAN1 to NS32532. Nevertheless changes in the memory management unit and the larger address space required modifications.

Current versions are TITAN3 on an NS32532 system and TITAN4 for a M32632 system. The main differences come from different hardware.

TRIPUTER V0.9 has a very basic version of TITAN. It uses only 32 Mbyte of the DRAM as main memory. 4 Mbyte can be used for graphic applications. Inside the 32 Mbyte a RAM-Disk is available.

The main purpose for TITAN on TRIPUTER V0.9 is to test the functions of the PERI board. The software is stored in the configuration pof file. To use the software the following steps are necessary:

Important hint: unfortunately TITAN5 uses German language – sorry for that!

Step	Enter	Description
1.	"t 0"	if you are in the MONITOR start TITAN5 with this command.
2.	"set_time"	this command allows you to set the real time clock to the actual time.
3.	"time"	enter this command to see the correct time
4.	"mkdir tools"	this command will create a directory with the name "tools" in the ramdisk.
5.	"rom_read"	this command reads the flash and copies the data in the directory "tools".
6.	"cd","dir"	can be used to look around.

With "set_time" and "time" you can verify that the RTC of PERI is working. If you have inserted a battery in the socket the clock should be running if the Starter Kit is powered off.

To test the mouse you should insert one in the green port of the PS2 connector. This test is only useful if you have connected a monitor to the HDMI port of the Starter Kit. Then you see a mouse symbol on the screen and you can move it around the screen. Use the buttons to see some reactions on the terminal. To finish the program place the mouse symbol in the upper right corner.

Use "type" to look into the text files. All files without an extension are binary files. You can use "dump" to look into them.

"ed", "pas" and "asse" are the programs for program development. "ed" is the editor. If you use PuTTY then select "VT100+" for the function keys. The editor uses a command and insert mode like vi. Use "i" for insert and "x" for replace. To come back to command mode use F5. F6 places the cursor at the beginning of the file and F7 at the end. In the command mode press "q" to see the exit

options: use the first letter to choose one.

“pas” translates a Pascal program to assembler code. The compiler is not intelligent. For example it doesn't understand the concept of pointers. All is done with simple arrays. This program has not been updated since 20 years. Maybe my last project will be to include code optimization – using registers for faster execution.

“pas” generates two text files: one with the extension “.32K” and the other with the extension “.COD”. The former is the header code of the program and the latter is the program code. The assembler “asse” makes the executable out of them.

The Pascal source code file must use the extension “.PAS”. Use “pas filename” without the extension. The same it true for the assembler: use “asse filename”. The output will be an executable named “filename”.

TITAN5 is case sensitive.

Probably the most interesting test is for the Ethernet port. To start it type in “ftp -init” to enable the PHY device on the PERI board. The three LEDs will be lighted: LED1 displays the speed status which is on = 100Mbit/S, LED2 is link status which should be on and LED3 is receive status which is flashing from time to time.

Now you can enter “ftp” to transfer files. To enter the IP address and the MAC address of your computer you use “ftp IP-address MAC-address”. The format of the IP address (32 bit only) is xxx.xxx.xxx.xxx and the MAC address comes in with xx-xx-xx-xx-xx-xx . The user name is “hugo” and the password is “xxx”. You can see this in the “ftp.PAS” source code as constants. You may change it there...

My “ftp” program is not very capable. It does the minimum to work. Five commands are available: “LIST”, “NLST”, “RETR”, “STOR” and “QUIT”. Please enter the commands as written. Only one file is allowed for “STOR” and “RETR”, for example “RETR myfile.dat” . Read in some big file from your PC to see whether the Ethernet is ok.

Use “ftp -stop” to shutdown the Ethernet port.

The program “rdregs” read some registers out of the Ethernet PHY device.

The program “sdc_rw” writes 64 kbytes of data to the SDC and reads them back. Use “sdc_rw w” to write and “sdc_rw r” to read. The program will write the data somewhere near 7.6 Gbyte. The address is coded in the source file. Only HD SDCs are supported.

TITAN5 uses some form of UNIX “history”. Enter “h” to see the last 20 commands. You can execute one with “!” and a number behind.

If you use the monitor at the HDMI output you can redirect the output of TITAN5 to it. Enter “x” to switch forth and back.

“sys” and “stat” show some information about the system.