



nanoEngine Hardware Reference Manual

July 17, 2000 Rev 0.6

Notice

The information in this document is subject to change without notice.

THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, Bright Star Engineering DOES NOT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Restricted Rights Legend

Use, duplication, or disclosure by the United States Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

This document may not, in whole or in part, be copied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from Bright Star Engineering, Inc.

© **Bright Star Engineering, Inc.**

All Rights Reserved

Printed in the USA

June 2000

Contents

CHAPTER 1 NANOENGINE OVERVIEW	1-1
CHAPTER 2 SETUP AND OPERATION.....	2-1
REQUIRED HARDWARE	2-2
UNPACKING THE NANOENGINE	2-3
MOUNTING THE NANOENGINE.....	2-3
CONNECTING A POWER SOURCE	2-3
CONNECTING ETHERNET AND RS-232	2-4
CONFIGURING THE NETWORK CONNECTION	2-4
TESTING NETWORK OPERATION	2-5
WHAT COMES NEXT	2-6
CHAPTER 3 FUNCTIONAL DESCRIPTION – GENERIC BUS CONFIGURATION	3-1
INTEL SA1110 STRONGARM CPU.....	3-1
FLASH AND SDRAM MEMORY.....	3-2
BSE NANOBRIDGE.....	3-2
ON-BOARD 10/100 ETHERNET.....	3-2
NANOENGINE EXTERNAL INTERFACE	3-3
CHAPTER 4 FUNCTION DESCRIPTION – PCI CONFIGURATION	4-1
OVERVIEW	4-1
BSE NANOBRIDGE.....	4-2
NANOENGINE EXTERNAL INTERFACE.....	4-2
CHAPTER 5 NANOENGINE FIRMWARE.....	5-1
OVERVIEW	5-1
R, RB, RS, RW – READ BYTE, SHORT, WORD.....	5-2
W, WB, WS, WW – WRITE BYTE, SHORT, WORD.....	5-3
GET – GET PARAMETER	5-4
SET – SET PARAMETER	5-5
LOAD – TFTP FILE LOAD	5-6
GO – JUMP TO ADDRESS.....	5-7
FERASE – ERASE FLASH MEMORY	5-8
STANDARD PARAMETER VALUES.....	5-9
CHAPTER 6 ELECTRICAL SPECIFICATIONS	6-1
CHAPTER 7 MECHANICAL SPECIFICATIONS	7-1
APPENDIX A RELATED DOCUMENTS	I
APPENDIX B TFTP SERVER SETUP	III

Chapter 1

nanoEngine Overview

Bright Star Engineering's nanoEngine single board computer provides an unprecedented level of power and integration in a 1.4" by 2.4" module. Utilizing the Intel SA1110 StrongARM processor, the nanoEngine is able to provide desktop system class-performance yet consumes well under 2 watts of power. In conjunction with the 200 MHz processor core, the nanoEngine's on-board 10/100 ethernet controller is able to provide in excess of 80 MBits/sec sustained user-level TCP performance. The on-board flash memory provides storage for the operating system as well as OEM application software and data.

The nanoEngine has the following significant features:

- 200 MHz StrongARM CPU
- up to 64 MB DRAM
- up to 16 MB FLASH
- 10/100 Ethernet
- USB Slave Controller
- LCD/Video Controller
- Three RS-232 channels

BSE's nanoEngine provides the OEM with a choice of external interfaces. In one configuration the external interface is configured as a PCI bus interface including a PCI bus arbiter and interrupt controller. This allows glueless connection to common PCI peripheral components.

The second external interface configuration provides a generic bus mode allowing direct connection to common memory and peripheral devices.

The nanoEngine provides a firmware boot loader programmed into the flash memory "bootblock" at the factory. The bootloader is used to initialize the nanoEngine and load the primary operating system from flash memory, from the network or from the serial console. The nanoEngine can readily used with a variety of operating systems like Linux, Windows CE, VxWorks or OS-9.

Chapter 2

Setup and Operation

This chapter shows you how to get your nanoEngine out of the shipping box and begin to use it. The steps you'll follow are:

- Unpack the board.
- Mount the nanoEngine on a motherboard or other chassis (this step is optional).
- Connect power.
- Connect Ethernet and a serial port.
- Set parameters in the flash memory of the nanoEngine to define power-up behavior.
- Verify correct operation.
- Begin downloading software, and start development work and testing.

Required Hardware

You need the following equipment to begin using the nanoEngine:

- The nanoEngine board itself.
- Power source. A regulated source of 3.3 volts DC is required to power the nanoEngine. Development boards supplied with the nanoEngine SDK provide an on-board 3.3 volt regulator. The development boards require a regulated 5 volt input source which is provided by wall-plug power module included with the SDK.
- Ethernet and serial cables, with connectors appropriate for your configuration.
- At least one host computer with an Ethernet connection. You'll use the host computer to set the connection parameters of the nanoEngine, download software, and to debug its operation.
- Software on the host computer. This includes:
 - Terminal emulator. This enables you to configure the nanoEngine via its serial port.
 - TFTP (Trivial File Transfer Protocol) server. (The PumpKIN TFTP server is freely available from BSE.)

Unpacking the nanoEngine

You need to use precautions to avoid electrostatic discharge (ESD) damage to the nanoEngine. You should wear a properly grounded anti-static wrist strap before removing the nanoEngine from its protective anti-static envelope, and whenever you touch it. The nanoEngine should always be kept on a grounded, static-free surface. We recommend that you keep the nanoEngine in an ESD-approved workstation for your development work.

If the shipping carton is damaged when you receive it, the shipper may be liable for damages. To ensure that you can be reimbursed for damage in shipping, make sure that the shipper or their agent is present when you unpack and inspect the equipment.

Using ESD-safe procedures, remove all equipment from the packaging material. Check the packing list to make sure all items have been included.

Mounting the nanoEngine

If you are mounting the nanoEngine to a motherboard (you will have a motherboard if you purchased the Developer's Kit), do it now. Holding the nanoEngine gently by the edges, press firmly until the nanoEngine is completely seated.

Connecting a Power Source

If you purchased the netChassis, just plug the chassis's power module into a 110V AC receptacle, connect the DC power jack to the chassis's, and you're done. You can proceed to the next section.

Connecting Ethernet and RS-232

Plug a standard 10BaseT Ethernet cable into the chassis connector; plug the other end into the network connector of your network hub. Connect a serial cable with RJ45 connector into the evaluation board connector RS232-1; plug the other end into a serial port connector of your host computer. The nanoEngine is configured as DTE.

Configuring the Network Connection

The first time you use the nanoEngine, you must either set network parameters to its flash memory, or have a DHCP/BOOTP server setup to supply the appropriate parameters to the nanoEngine upon bootup. This allows it to initialize and boot correctly from then on. In your product configuration, you'll have several options for downloading. This section shows you only one option, manually configuring an IP address. We recommend you try this, at least the first time you use the nanoEngine, because it allows you to quickly verify that it has a correct Ethernet connection.

In this session, you'll define the following parameters:

- IP address of the nanoEngine.
- Network netmask.
- Host name of the TFTP server.
- Gateway IP address (optional).

Before starting, determine the values of the parameters that you'll use for this session. Your host computer will probably act as the TFTP server, so you'll need its IP address. In the example below the TFTP server has address 192.168.1.150, the gateway address is 192.168.1.1, the netmask is 255.255.255.0, and we'll assign the nanoEngine the address 192.168.1.8.

Start the terminal emulator on your host computer, and connect to your serial port. Press <return> to get a prompt. You can also cycle power to the nanoEngine, or press its manual reset, to get a boot prompt. If you don't receive characters by this point, carefully check all connections. Also make sure that your serial port is correctly configured. The nanoEngine is shipped with these default serial parameters: 9600 baud, no parity, 8 bits, no stop bit. You might also need to swap the TX and RX pins on one side of your serial cable, or get a "gender-changer" connector.

Define the parameters by typing the following:

```
>set myip 192.168.1.8
>set netmask 255.255.255.0
>set serverip 192.168.1.150
>set gateway 192.168.1.1
```

Verify the parameters by displaying them using the `get` command. Here's an example:

```
>get
myip = 192.168.1.8
netmask = 255.255.255.0
serverip = 192.168.1.150
gateway = 192.168.1.1
>
```

Testing Network Operation

You can now test that the nanoEngine is properly hooked up to the network. From your host computer, run a “ping” test. On your have host computer, type:

```
% ping 192.168.1.8
```

If you receive a response without timing out, the nanoEngine is ready to go. If not, make sure that your host computer’s network connection is functioning properly. Make sure no other computer on the network is using the IP address you assigned to the nanoEngine.

If you purchased the Developer’s Kit, you can perform a more complete checkout of the nanoEngine.

What Comes Next

Now you'll begin configuring the nanoEngine for your particular product. You'll also begin developing and testing your software after that. The nanoEngine supports many different options for configuration and development. In particular:

- You can manually configure the IP address of the nanoEngine. This address stays in flash RAM forever, until you change it.
- You can configure the nanoEngine to work with a DHCP server. In this case, the nanoEngine obtains an IP address from the server each time it powers up.

You can load flash memory in a variety of ways:

- As a binary file obtained via TCP/IP from your TFTP server.
- As an S-record file obtained from the TFTP server.
- As an S-record file loaded from a serial port.

Of course, your product may have dynamic functionality, and it can be configured to boot dynamically from the network each time it powers up.

Chapter 3

Functional Description – Generic Bus Configuration

The nanoEngine is factory configured to support one of two types of external bus interfaces. This section provides a description of the nanoEngine configured with a general purpose external bus. Figure 3-1 depicts the internal architecture of the nanoEngine.

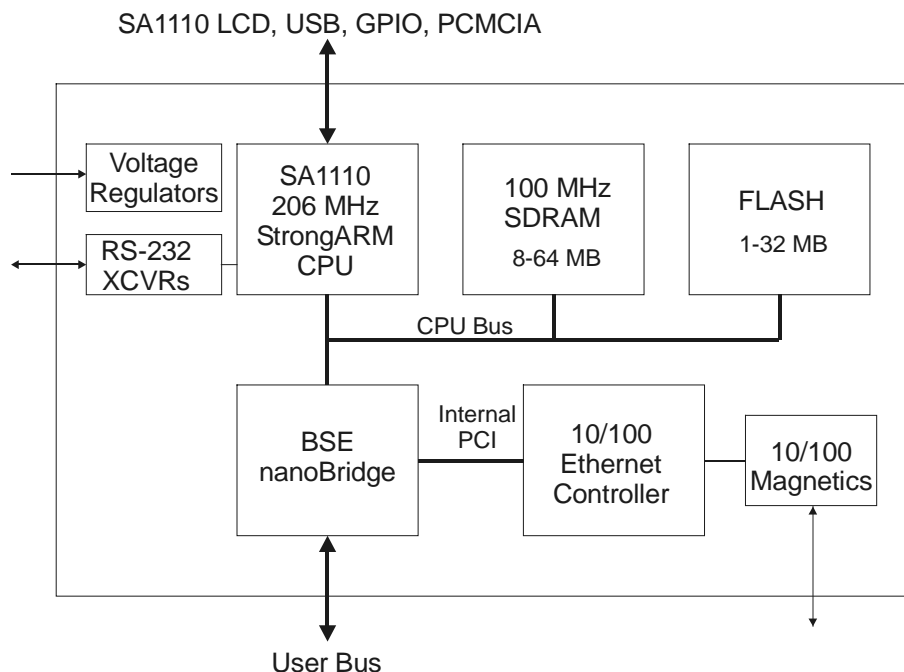


Figure 3-1. nanoEngine Architecture

Intel SA1110 StrongARM CPU

The SA1110 microprocessor, is Intel's high performance implementation of an ARM microprocessor. It combines a high-performance StrongARM core with a variety of on-chip peripherals.

The SA1110 serial ports are used to provide a three RS-232 serial interfaces to the nanoEngine. The serial ports do not support hardware flow control via RTS/CTS.

The SA1110 LCD/Video controller pins are connected directly to the external connector. These may be used as general purpose I/O signals when the LCD interface is not being used. Refer to the SA1110 User's manual for further information on use and programming of the LCD/Video Interface.

FUNCTIONAL DESCRIPTION

Table 3-1 depicts the memory map for the nanoEngine:

Memory Map

Address Range	Description
0000.0000 - 003F.FFFF	4 MB FLASH
C000.0000 - C1FF.FFFF	32 MB SDRAM
1860.0000 - 186F.FFFF	Internal PCI Memory Read/Write
18A1.0000 - 18A1.FFFF	Internal PCI Config Space
4000.0000 - 47FF.FFFF	External Bus I/O – Multiplexed Mode
4800.0000 - 4FFF.FFFF	External Bus I/O – Non-Multiplexed Mode

** remaining memory map locations are as documented in the SA1110 users guide*

Flash and SDRAM Memory

The on-board FLASH memory provides storage for the operating system as well as OEM application and data.

The lowest two memory blocks are the "boot" blocks that are factory programmed with the boot monitor software, board serial number, and Ethernet MAC (hardware address). These blocks are locked and cannot be erased or reprogrammed. This is to ensure there is always some way to reload the board in case all the other flash block is erased or have been loaded with faulty code.

Blocks 2 and 3 are used for parameter storage. These parameters, such as the IP address for the board and others, are used by the boot monitor software. See Chapter 5 for a complete description of the parameters used by the boot monitor.

BSE nanoBridge

The nanoBridge incorporates an internal PCI bus and system control functions including a three-way arbiter, PCI interrupt control and PCI/SDRAM clock generation. The nanoBridge internal PCI interface supports full master/target operation including burst read and write cycles.

On-Board 10/100 Ethernet

nanoEngine 10/100 Ethernet functionality is provided via an Intel 82559ER PCI ethernet controller connected to the nanoEngine internal PCI bus interface. The 82559ER use PCI bus mastering to DMA ethernet frames directly to/from system memory via the nanoBridge. 10/100 ethernet magnetics are included in the nanoEngine allowing the user to connect the ethernet TX/RX pairs directly to an RJ-45 ethernet jack.

nanoEngine External Interface

The external interface to the *nanoEngine* is detailed in the following three tables. A number of the signals are connected directly to the SA1110 CPU. For more information on the function and timing of SA1110 specific signals refer to the “*Intel StrongARM SA-1110 Microprocessor Advanced Developer's Manual.*” Note that SA1110 signals present on the interface connector are not 5 volt tolerant. However, user bus signals are 3.3 volt signals which are 5 volt tolerant.

The external user bus is provides two access modes:

- Multiplexed 32 bit address/data
- Non-multiplexed 24 bit address and 16 bit data

The access mode is determined by the region being accessed by the CPU (see the Table 3-1 memory map). The nanoBridge acts as a bus transceiver/multiplexor for the CPU system bus. External bus timing is determined by the settings of the SA1110 memory control registers. The nanoBridge adds a worst case delay of 12 nanoseconds to signals propagating between the user bus and the internal CPU bus.

SA1110 and SYSTEM SIGNALS

Pin Name	Description
VCC	3.3 volts
GND	GND
TX+, TX-, RX+, RX-	10/100 Ethernet
USB+, USB-	SA1110 Slave USB
TCK, TRST, TDO, TDI, TMS	JTAG
LDD0-LDD7 L_PCLK, L_BIAS, L_LCLK, L_FCLK	SA1110 LCD
GPIO2-GPIO20	SA1110 GPIO
TXD_C, RXD_C, SCLK_C, SFRM_C	SA1110 Serial channel 4
PKTSEL, nPOE, nPWE, nPIOR, nPIOW nPCE_1, nIOIS16, nPWAIT, nPREG, nPCE_2	SA1110 PCMCIA
RSTX1, RSTX2, RSTX3 RSRX1, RSRX2, RSRX3	RS-232 level serial signals
CLOCK	25 MHz Clock
WP	Flash Boot Sector Write Protect
RSVD	Reserved Pin
MRST	Manual Reset In – Active low – can be connected directly to a push button switch
nRST	System Reset Out (Active low)

USER BUS SIGNALS (5 volt tolerant)

Pin Name	Description
AD00-AD15	Multiplexed Address/Data Contains address bits A0-A15 during address phase of multiplexed accesses (when ALE# is low) and contains data bits D0-D15 otherwise.
AD16-AD31	Multiplexed Address/Data Contains address bits A16-A23 during address phase of multiplexed accesses (when ALE# is low) . Contains data bits D15-D31 during data phase (ALE# high) of multiplexed accesses. Contains address bits A8-A23 during non-multiplexed accesses.
A0-A7	Non-multiplexed address
ALE#	Address Latch Enable
RD#	Read Strobe / Output Enable
WE#	Write Strobe
WE0# - WE3#	Byte Lane Enables
RDY	Insert Wait-State
CS0#-CS1#	Chip Select
URST#	User Bus Reset (Active Low)

nanoEngine Pinout – General Purpose Bus Version

Function	Pin	Pin	Function
TX+	2	1	RX+
TX-	4	3	RX-
RSVD	6	5	RSVD
GND	8	7	GND
MRST	10	9	nRST
USB+	12	11	USB-
TCK	14	13	BFAULT
TRST	16	15	VCC
TDO	18	17	VCC
TDI	20	19	VCC
TMS	22	21	RSVD
PKTSEL	24	23	nPOE
GND	26	25	nPWE
nPCE 2	28	27	nPIOR
nPREG	30	29	VCC
nPWAIT	32	31	GND
nIOIS16	34	33	nPIOW
XSDO	36	35	nPCE 1
XSDI	38	37	XSFRAME
GND	40	39	XSCLK
GPIO 20	42	41	GPIO 19
GPIO 18	44	43	GPIO 17
GPIO 16	46	45	GPIO 15
GPIO 14	48	47	GND
GPIO 11	50	49	GPIO 13
VCC	52	51	GPIO 12
LCD D14	54	53	GPIO 10
GND	56	55	LCD D15
LCD D12	58	57	LCD D13
LCD D10	60	59	LCD D11
LCD D8	62	61	LCD D9
L PCLK	64	63	GND
L BIAS	66	65	L FCLK
L LCLK	68	67	L DD7
L DD6	70	69	L DD5
GND	72	71	VCC
L DD4	74	73	L DD3
L DD2	76	75	L DD1
L DD0	78	77	A7
A5	80	79	GND

Function	Pin	Pin	Function
A3	82	81	A6
A1	84	83	A4
GND	86	85	A2
RSVD	88	87	A0
VCC	90	89	URST#
GND	92	91	GND
CLOCK	94	93	CLOCK
GND	96	95	GND
CS0#	98	97	CS1#
AD31	100	99	AD30
GND	102	101	AD28
AD29	104	103	AD26
AD27	106	105	AD24
AD25	108	107	GND
WE3#	110	109	RSVD
AD23	112	111	VCC
AD21	114	113	AD22
AD19	116	115	AD20
GND	118	117	AD18
AD17	120	119	AD16
WE2#	122	121	ALE#
RD#	124	123	RDY#
WE#	126	125	GND
RSVD	128	127	RSVD
RSVD	130	129	RSVD
VCC	132	131	AD15
WE1#	134	133	AD13
GND	136	135	AD11
AD14	138	137	AD9
AD12	140	139	WE0#
AD10	142	141	GND
AD8	144	143	AD6
AD7	146	145	AD4
AD5	148	147	AD2
AD3	150	149	AD0
AD1	152	151	VCC
GND	154	153	WP
RSTX1	156	155	RSRX1
RSTX2	158	157	RSRX2
RSTX3	160	159	RSRX3

FUNCTIONAL DESCRIPTION

Chapter 4

Function Description – PCI Configuration

Overview

The nanoEngine is factory configured to support one of two types of external bus interfaces. This section provides a description of the nanoEngine configured with a PCI Bus external interface. Figure 4-1 depicts the internal architecture of the nanoEngine.

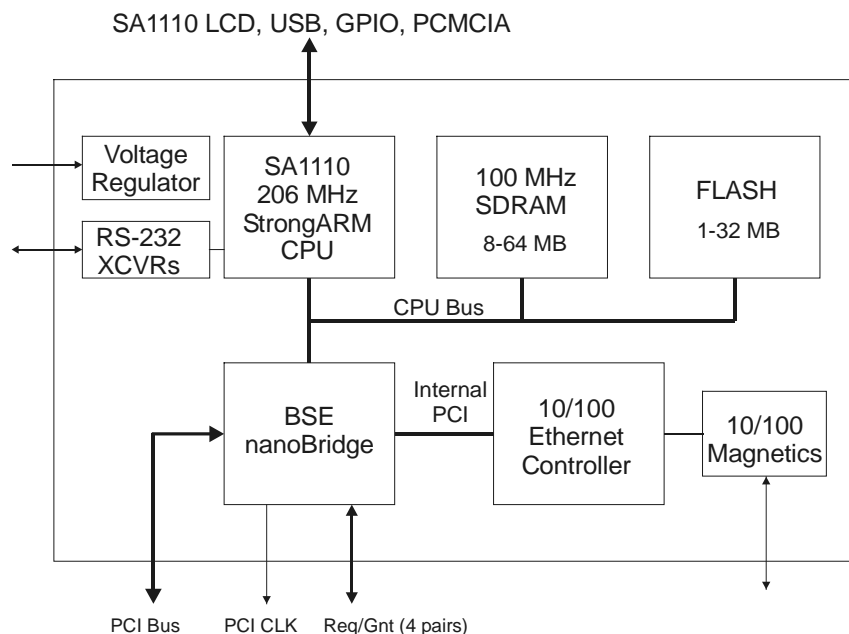


Figure 4-1. nanoEngine Architecture

Table 4-1 depicts the memory map for the nanoEngine PCI configuration:

Memory Map

Address Range	Description
0000.0000 – 003F.FFFF	4 MB FLASH
C000.0000 – C1FF.FFFF	32 MB SDRAM
1860.0000 – 186F.FFFF	Internal PCI Memory Read/Write
18A1.0000 – 18A1.FFFF	Internal PCI Config Space
4000.0000 – 47FF.FFFF	External PCI Config and I/O Space
4800.0000 – 4FFF.FFFF	External PCI Memory Read/Write

* remaining memory map locations are as documented in the SA1110 users guide

BSE nanoBridge

The nanoBridge incorporates all required PCI system control functions including a three-way arbiter, PCI interrupt control and PCI/SDRAM clock generation allowing glueless connection to any device with a PCI interface. Each nanoBridge PCI interface supports full master/target operation including burst read and write cycles. The external; PCI interface provides four REQ/GNT pairs for glueless connection to up to 4 PCI devices.

The external PCI bus interface is not currently designed to operate in a conventional PCI peripheral mode with an external system controller. The nanoBridge does not contain a PCI configuration space. There are no IDSEL inputs or provisions for using an externally provided PCI bus clock or bus arbiter.

Through the use of the on-chip arbiter and the insertion of PCI target disconnect cycles, the nanoBridge is able to control the worst case latency for SA1110 access to the CPU/SDRAM bus. This guarantees SDRAM refresh timing and prevents under-run of the SA1110 LCD controller FIFO.

When bursting from/to the PCI bus the SDRAM is clocked at the same frequency as the PCI bus.

When a target issues a target-retry command, the nanoBridge will immediately retry the bus cycle, other PCI bus transactions will be blocked until the bus cycle completes or is aborted.

PCI-to-PCI bridging is currently not supported

Current silicon uses 3.3v I/O buffers which are PCI compliant and 5 volt tolerant. However, there are no clamp diodes on the I/O buffers hence use in a 3.3 volt signaling environment is not fully compliant with the PCI specification. This is a non-issue for many applications since (1) most PCI devices use or provide for 5 volt signaling and/or have 5 volt tolerant I/O, (2) overshoot can be typically be controlled in an environment where the PCI bus is an on-PCB bus.

nanoEngine External Interface

The external interface to the *nanoEngine* is detailed in the following three tables. A number of the signals are connected directly to the SA1110 CPU. For more information on the function and timing of SA1110 specific signals refer to the “*Intel StrongARM SA-1110 Microprocessor Advanced Developer's Manual.*” Note that SA1110 signals present on the interface connector are not 5 volt tolerant. However, user bus signals are 3.3 volt signals which are 5 volt tolerant.

Table 4-2 depicts SA1110 and System Signals on the nanoEngine External I/O Connector

SA1110 and SYSTEM SIGNALS

Pin Name	Description
VCC	3.3 volts
GND	GND
TX+, TX-, RX+, RX-	10/100 Ethernet
USB+, USB-	SA1110 Slave USB
TCK, TRST, TDO, TDI, TMS	JTAG
LDD0-LDD7 L_PCLK, L_BIAS, L_LCLK, L_FCLK	SA1110 LCD
GPIO2-GPIO20	SA1110 GPIO
TXD_C, RXD_C, SCLK_C, SFRM_C	SA1110 Serial channel 4
PKTSEL, nPOE, nPWE, nPIOR, nPIOW nPCE_1, nIOIS16, nPWAIT, nPREG, nPCE_2	SA1110 PCMCIA
RSTX1, RSTX2, RSTX3 RSRX1, RSRX2, RSRX3	RS-232 level serial signals
CLOCK	25 MHz Clock
WP	Flash Boot Sector Write Protect
RSVD	Reserved Pin
MRST	Manual Reset In
nRST	System Reset Out

Table 4-3 depicts PCI Bus Signals on the nanoEngine External I/O Connector

PCI BUS SIGNALS (5 volt tolerant)

Pin Name	Description
AD00-AD31	Multiplexed Address/Data
CB/E0-3	PCI Command/Byte Enables
FRAME#, IRDY#,	PCI Control
TRDY#, STOP#, DEVSEL#	PCI Control
PAR#	PCI Parity
REQ0#-REQ3#	PCI Bus requests
GNT0#-GNT3#	PCI Bus Grants
PCIRESET#	PCI Bus Reset
PCICLK	25 MHz PCI Clock

nanoEngine Pinout – External PCI Bus Configuration

Function	Pin	Pin	Function
TX+	2	1	RX+
TX-	4	3	RX-
RSVD	6	5	RSVD
GND	8	7	GND
MRST	10	9	nRST
USB+	12	11	USB-
TCK	14	13	BFAULT
TRST	16	15	VCC
TDO	18	17	VCC
TDI	20	19	VCC
TMS	22	21	RSVD
RSVD	24	23	RSVD
GND	26	25	RSVD
RSVD	28	27	RSVD
RSVD	30	29	VCC
RSVD	32	31	GND
RSVD	34	33	RSVD
XSDO	36	35	RSVD
XSDI	38	37	XSFRAME
GND	40	39	XSCLK
GPIO 20	42	41	GPIO 19
GPIO 18	44	43	GPIO 17
GPIO 16	46	45	GPIO 15
GPIO 14	48	47	GND
GPIO 11	50	49	GPIO 13
VCC	52	51	GPIO 12
LCD D14	54	53	GPIO 10
GND	56	55	LCD D15
LCD D12	58	57	LCD D13
LCD D10	60	59	LCD D11
LCD D8	62	61	LCD D9
L PCLK	64	63	GND
L BIAS	66	65	L FCLK
L LCLK	68	67	L DD7
L DD6	70	69	L DD5
GND	72	71	VCC
L DD4	74	73	L DD3
L DD2	76	75	L DD1
L DD0	78	77	RSVD
PCI_GNT3#	80	79	GND

Function	Pin	Pin	Function
PCI_GNT1#	82	81	RSVD
PCI_REQ2	84	83	PCI_GNT2#
GND	86	85	PCI_REQ3
RSVD	88	87	PCI_REQ1
VCC	90	89	PCI_RST#
GND	92	91	GND
PCICLK	94	93	PCICLK
GND	96	95	GND
PCI_GNT0#	98	97	PCI_REQ0
PCI_AD31	100	99	PCI_AD30
GND	102	101	PCI_AD28
PCI_AD29	104	103	PCI_AD26
PCI_AD27	106	105	PCI_AD24
PCI_AD25	108	107	GND
PCI_CBE3#	110	109	RSVD
PCI_AD23	112	111	VCC
PCI_AD21	114	113	PCI_AD22
PCI_AD19	116	115	PCI_AD20
GND	118	117	PCI_AD18
PCI_AD17	120	119	PCI_AD16
PCI_CBE2#	122	121	PCI_FRAME#
PCI_IRDY#	124	123	PCI_TRDY
PCI_DEVSEL#	126	125	GND
RSVD	128	127	PCI_STOP
RSVD	130	129	PCI_PAR
VCC	132	131	PCI_AD15
PCI_CBE1#	134	133	PCI_AD13
GND	136	135	PCI_AD11
PCI_AD14	138	137	PCI_AD9
PCI_AD12	140	139	PCI_CBE0#
PCI_AD10	142	141	GND
PCI_AD8	144	143	PCI_AD6
PCI_AD7	146	145	PCI_AD4
PCI_AD5	148	147	PCI_AD2
PCI_AD3	150	149	PCI_AD0
PCI_AD1	152	151	VCC
GND	154	153	WP
RSTX1	156	155	RSRX1
RSTX2	158	157	RSRX2
RSTX3	160	159	RSRX3

Chapter 5 nanoEngine Firmware

Overview

The firmware boot loader is programmed into the flash memory “bootblock” at the factory. The bootblock is hardware erase protected and cannot be erased by the user. This protection ensures that the system can always be reloaded even if the rest of the flash memory is corrupted through programmer error or by other means. The sole function of the flash memory bootblock is to initialize the system and load the primary operating system. The bootblock also contains the Ethernet MAC address, board serial number, and a number of read/writeable parameter blocks that contain parameter information used by the bootloader.

The bootloader implements a small number of commands that the user can interact with via the serial port. The commands allow the user to control the boot process.

The boot monitor allows loading of S-records. The boot monitor interprets any command beginning with “S” and followed by a digit as an S-record. In this way S-records can be sent from the host system to the console port of the system without entering into a special download mode.

When loading binary or S-record data, if the address of the data to be written is in flash memory the boot monitor will attempt to program the flash data at that address without attempting to erase the block associated with the address. If the address to be written is the beginning of a flash memory block the flash memory block is first erased (erasing the entire contents of the block) and then the memory location is written. This allows binary images to be loaded to flash memory in a straightforward manner without the user having to be concerned with erasing and loading particular flash blocks.

The user can make the system boot from code stored in flash upon power-on reset by setting the bootcmd flash parameter to a “go” command to the entry point in flash memory.

R, RB, RS, RW – Read Byte, Short, Word

Command

R	<ADDR>	<COUNT>	Read
RB	<ADDR>	<COUNT>	Read Byte (8 Bits)
RS	<ADDR>	<COUNT>	Read Short (16 Bits)
RW	<ADDR>	<COUNT>	Read Word (32 Bits)

Description

The read commands allow reading of memory locations in main memory.

Example

```
>rb 0 8
00000000 : 00 00 00 00 80 2B EF 44
>rs 0 8
00000000 : 0000 0000 802B EF44 3821 FFFC 3C00 0000
>rw 0 8
00000000 : 00000000 802BEF44 3821FFFC 3C000000
00000010 : 90010000 9421FFF0 4800018D 4801D5A1
>r
00000000 : 00000000
>rb 0 2
00000000 : 00 00
>r
00000000 : 00
>
```

W, WB, WS, WW – Write Byte, Short, Word

Command

W	<ADDR>	<COUNT>	Write
WB	<ADDR>	<COUNT>	Write Byte (8 Bits)
WS	<ADDR>	<COUNT>	Write Short (16 Bits)
WW	<ADDR>	<COUNT>	Write Word (32 Bits)

Description

The write commands allow writing of memory locations in main memory.

Example

```
>wb 0 a4  
>ws 0 aa55  
>ww 0 12345678  
>
```

GET – Get Parameter

Command

```
get    [<name>]
```

Description

Print parameter from flash memory

Example

```
>get myip  
myip = 192.168.1.9  
>get  
myip = 192.168.1.9  
serverip = 192.168.1.5  
bootcmd = load sim/hello.bin 4000; go 4000
```

SET – Set Parameter

Command

set	<name>	<value>	- set parameter
set	<name>	“<value>”	- set parameter
set	<name>		- remove parameter <name>

Description

Set parameter into flash memory.

Example

```
>get
serverip = 192.168.1.5
myip = 192.168.1.9
bootcmd = load sim/hello 4000; go 4000
>set bootcmd
>get
serverip = 192.168.1.5
myip = 192.168.1.9
>set bootcmd "load sim/hello 4000; go 4000"
>set myip 192.168.1.8
>get
serverip = 192.168.1.5
bootcmd = load sim/hello 4000; go 4000
myip = 192.168.1.8
>
```

LOAD – TFTP File Load

Command

```
load [-s] <filename> [<addr>]
```

Description

Load Binary or S-Record file using the TFTP protocol.

The type of file to be loaded is assumed to be binary unless the “-s” flag is used.

If the file is a binary file the address to load the file to can be specified.

Example

```
>load sim/hello.bin c0008000  
loading ... done  
>
```

Notes

The appropriate network parameters must have been set in flash for the load command to work correctly. If these parameters have not been set, the system firmware will attempt to get these parameters dynamically from a DHCP or BOOTP server on the network.

GO – Jump to Address

Command

go <addr>

Description

Jump to specified address.

Example

```
>load sim/hello.bin 4000
loading ... done
>go 4000
```

FERASE – Erase Flash Memory

Command

```
ferase <addr>
```

Description

Erase the flash memory block associated with <addr>.

Example

```
>ferase fe000000  
>
```

Standard Parameter Values

The *nanoEngine* boot firmware recognizes the following standard firmware parameters:

netmask

Sub-net mask

gateway

Gateway IP address (if any).

eth

On boards with multiple ethernet port specifies which port should be used for boot loading. If not set this value defaults to "0".

serverip

Specifies the server IP address for TFTP load.

myip

Specifies the target board's IP address (e.g. NN.NN.NN.NN)

bootcmd

Specifies the command to be executed at boot time. Multiple commands can be combined by placing quotes around the entire command string and separating commands with semicolons. For example: "load test.bin 80000; go 8000".

When autobooting the user has three seconds to press the "escape" key on the serial console to cancel the autoboot operation.

Chapter 6 Electrical Specifications

Maximum Ratings

DC Power Supply	+3.3V dc \pm 5% 600mA Typical
Operating Temperature	0°C - 70°C
Storage Temperature	-40°C - +85°C
Relative Humidity	10% to 90% (no-condensing)

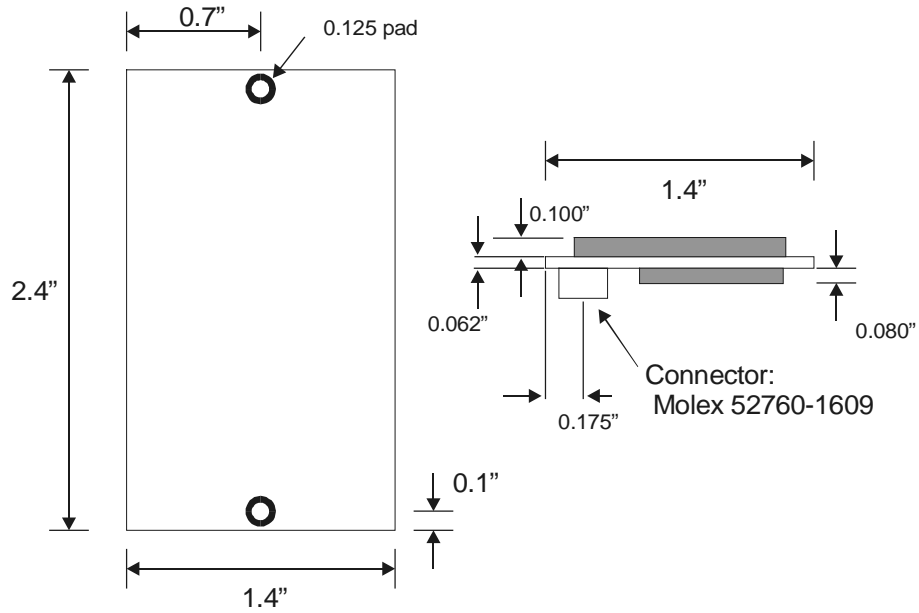
WARNING

Stressing the device beyond the Maximum Ratings may cause permanent damage. These are stress ratings only. Operation beyond the Operating Conditions is not recommended and extended exposure beyond the Operating Conditions may affect reliability.

The integrated circuits used on the nanoEngine have a maximum case temperature rating of 70° C. The user will have to insure that these case temperatures are not violated in a particular application by using appropriate cooling techniques if necessary.

Chapter 7

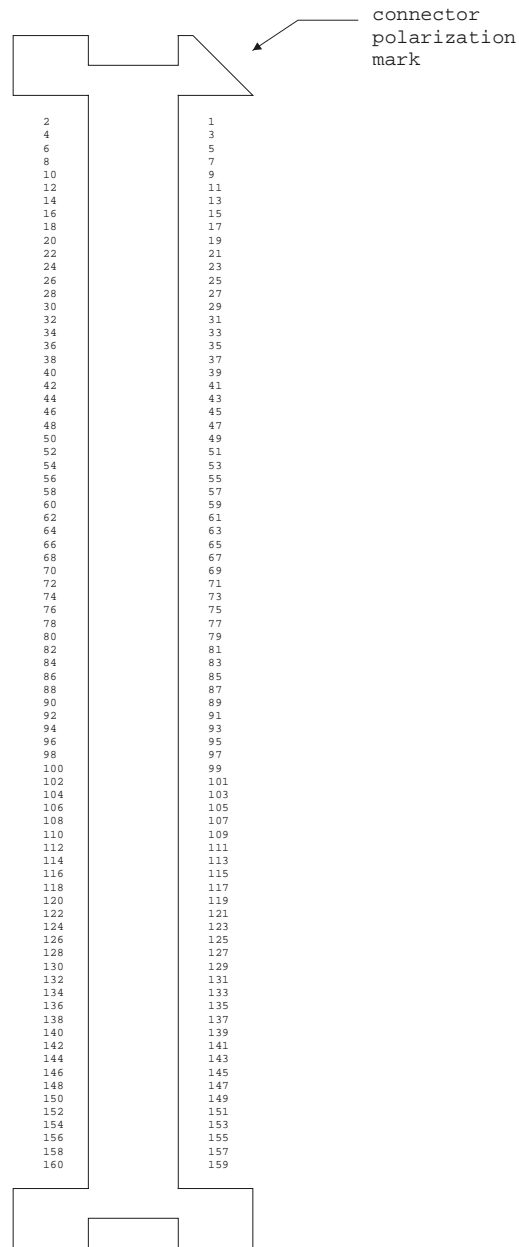
Mechanical Specifications



Mating Molex Connector	Board-to-Board Height
53475-1609	5 mm
53467-1609	6 mm
53481-1609	7 mm

MECHANICAL SPECIFICATIONS

This is how the mating molex connector should look on your board.



Appendix A

Related Documents

SA1110 User's Manual

"Intel StrongARM SA-1110 Microprocessor Advanced Developer's Manual"

<http://www.intel.com/design/strong/manuals/>

RELATED DOCUMENTS

Appendix B

TFTP Server Setup

This Appendix describes how to setup the free PumpKIN TFTP server on a Windows 95/98/NT host. This TFTP server can be used to allow the nanoEngine's boot monitor to load files from our PC. This software is included with the pKernel SDK and is available from BSE web site <http://www.brightstareng.com/> or directly from Klever Co. at <http://www.klever.net/kin/pumpkin.html>.

1. Download and Install the PumpKIN TFTP server if it is not already installed on your system.
2. Start the TFTP server. The first time you start the server you will want to click the "Options" button to set the appropriate parameters for your system. The TFTP filesystem root to the download directory where your nanoEngine binaries are. Set the Read Request Behavior to "Give All Files". In most cases you will want to be sure that "Allow access to subdirectories" is not checked. Set the "Write Request Behavior" to "Deny all requests". Click "Apply" followed by "OK" to accept the changes.

