# IDT79S134
# Evaluation Board Manual

**Version 1.1**
**October 1999**

2975 Stender Way, Santa Clara, California 95054
Telephone: (800) 345-7015 • TWX: 910-338-2070 • FAX: (408) 492-8674
Printed in U.S.A.

# Table of Contents

**Notes**

# List of Tables

**Notes**

# List of Figures

**Notes**

# Description of IDT79S134 Evaluation Board

Notes

## Introduction

The IDT79RC32134 is a high performance system controller chip that supports IDT's RISCore32300 CPU family. The RC32134 provides a direct connection between IDT's RC32364 32-bit embedded microprocessor and contains the system logic for boot memory, main memory, I/O, and PCI. It also includes on-chip peripherals such as DMA channels, reset circuitry, interrupts, timers and UARTs. Together, the RC32364 CPU and the RC32134 system controller form a complete CPU subsystem for embedded designs.

The IDT79S134 Evaluation Board provides an RC32134 evaluation tool as well as a cost effective way to add I/O boards through the PCI interface. The 79S134 is a working example of a typical embedded PCI host/satellite system. This board is highly configurable and contains hardware options for various memory configurations through the RC32364 RISController CPU.

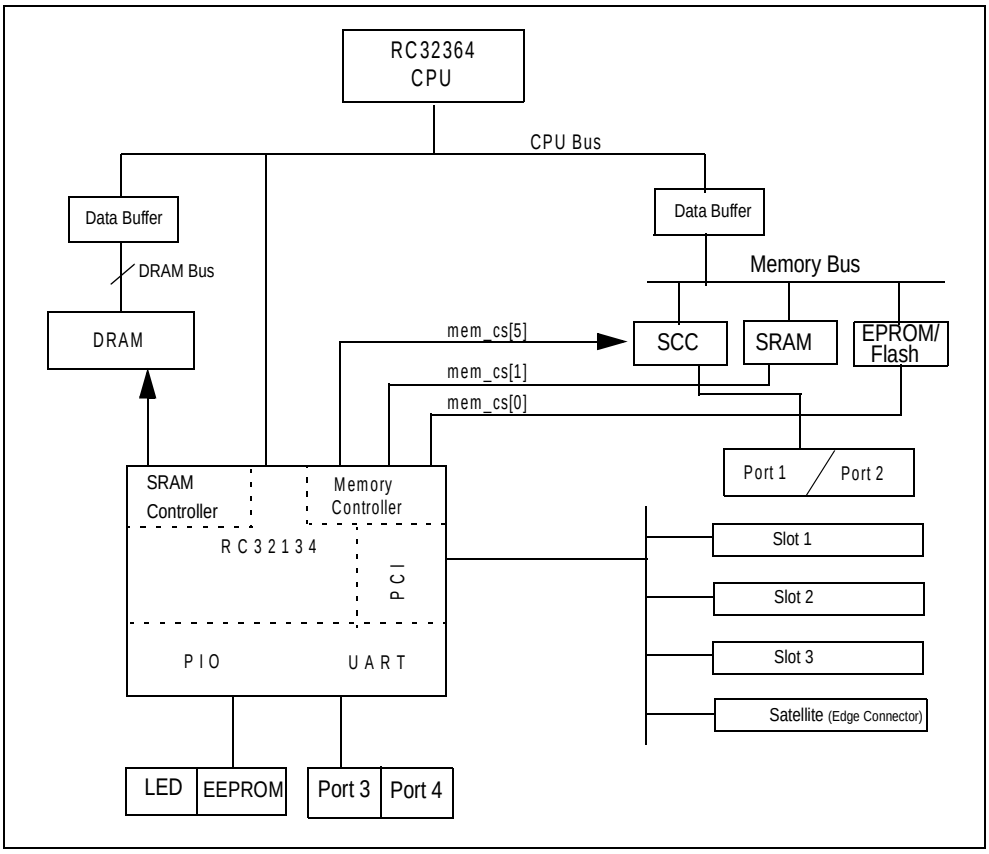



**Figure 1.1 79S134 Evaluation Board Block Diagram**

## Revision History

**March 1999**: Initial publication.

**October 1999**: Version 1.1. Updated schematics in Chapter 4.

# Overview of Features

Major features of the 79S134 Evaluation Board include:

- *Cost effective method of adding I/O boards through a PCI bus interface*
  - *3 PCI slots (Two 5V, one 3.3V) for adding peripheral controllers*
  - *1 PCI Edge connector*
- *EPROM 1Mbyte*
- *SRAM 1 Mbyte*
- *SDRAM 16 Mbytes*
- *85C30 SCC Controller*
- *Serial EEPROM (Mircrowire NM93C46)*
- *Two on-chip serial I/O ports (16550 Compatible UARTs)*
- *External 85C30 Serial I/O Controller*
- *One 4 digit LED display*

# Explanation of Features

IDT's S134 Evaluation Board is a complete working RC32364/RC32134 system and is intended as an evaluation tool and software development platform that uses the high performance RC32364 RISController, which is based on IDT's proprietary RISCore32300 CPU core. The board requires a simple CRT video terminal or emulator and a 5-volt power supply with at least 10 AMP of current. A ±12-volt power supply is also needed to support the PCI bus requirements. The board contains three PCI slots for adding peripheral controllers.

The board contains 4 MB of EPROM implemented with of 1MB each, using 4 sockets. The on-board EPROM memory contains IDT's flexible System Integration Manager (IDT/sim), a debugging monitor that supports code downloading from the host system and I/O. Execution control commands include single stepping and instruction tracing, memory probing, register probing, line-based assembly and disassembly of code. Information on using IDT/sim is provided in a separate document that is available from the IDT website **(www.idt.com)**.

The S134 evaluation board is constructed with both through-hole and surface mount devices on an 11" x 8 7/8" PCB rectangular form factor 8-layer laminated board with standoffs and is intended for use as a stand-alone bench top device.

# Specification Summary

### Part Number

- *IDT79S134 Evaluation Board*

### RISController

- *RC32364 RISController*

### On-Board Memory Capacity

- *RISController CPU on-chip*
  - *Instruction Cache — 8KByte*
  - *Data Cache — 2KByte*
- *As shipped*
  - *DRAM — 32MByte*
  - *EPROM — 1MByte*
- *Maximum*
  - *DRAM — 256MByte*
  - *EPROM — 4MB/Flash — 2 MB*
  - *SRAM — 1Mbyte*

### Debug Monitor Flash

- *1 MByte higher density EPROM 27C080 support, containing IDT/sim*

### Serial Ports

- *Controlled by 85C30 SCC controller*
- Two *RS232 DTE DB9P (9-pin male) connectors*
- *CRT video terminal connects to J1*
- *Software configurable features*
- *Default rate: 9600 Baud, 8 bits, no parity, 1 stop bit*

### Interrupts

- *6 unsynchronized*

### Physical Dimensions

- *Rectangular form factor: 11" x 8 7/8"*

### Operating Temperature

- *0-30°C*

### Relative Humidity

- *5% - 50%*

### Power Supply

- *5.0V ± 5%, 10 Amps typical*
- *± 12.0V ± 5%, required for PCI bus*

### Flash

- *Cached/non-cached, single access*
  - *X8, X16 or X32 support*
  - *Non-interleaved*

### DRAM[1]

- *Basic structure*
  - *168 pin DIMM socketed (sockets on board)*
  - *50 ns DRAM required for 50 MHz bus operation*
- *Configurations allowed*
  - *Only 2 DIMM allowed*
  - *Stays on page between transfers*

### SRAM

- *Basic structure*
  - *64-pin SIMM socket (sockets on board)*
  - *15ns SRAM*
  - *Zero wait-state operation for block read and block write*

### Programmable IO (PIO)

- *Input/Output/Interrupt source*
- *Individually programmable*

### PCI Bus Interface

- *Revision 2.1 compliant*
- *Bus clock frequency up to 66 MHz*
- *Bus speed synchronizer from CPU Bus to PCI 66 MHz*

1. Must be all RAM of one protocol type.

# Installation of IDT79S134 Evaluation Board

Notes

## 79S134 Installation

This chapter discusses the steps required to install and boot the 79S134 Evaluation Board. The primary installation steps are as follows:

1. Connecting a power source
   - *This involves connecting an external power supply to the board through J29.*
2. Connecting a video display terminal
   - *This involves connecting an RS232-C serial cable from a video terminal to the board through connector J1.*
3. Configuring jumper/switch options
   - *This involves altering the CPU reset initialization mode vector and changing the memory configuration. The board is shipped with the jumpers/switches set to their default configurations.*
4. Running Software
   - *No additional software is required.*
5. Booting IDT/sim
   - *When power to the board is turned on, the board's IDT/sim program boots and displays the start-up message.*

## Getting started quickly

The 79S134 board is shipped ready to run. Before the board is shipped, jumpers and switches are configured to the default settings shown in the tables below, and in general, they do not require further modification or setup.

Two basic requirements for the board to run are:

- *+5V power supply with at least 10 Amp of current*
- *±12V power supply to support the PCI bus requirements*

### Video Terminal Requirements

The CRT video terminal can be a typical VT100 type/ANSI terminal or emulator running with 9600 baud, 8 data bits, no parity, and 1 stop bit. Typically, a video terminal will have a male 25-pin DTE connector. On the evaluation board, the RS232-C connector uses a male 9-pin DTE connector (J1) of which only the RX, TX, and GND pins are necessary, as shown in Table 2.15. For the default stand-alone mode, power to the board is provided by using a standard PC/AT power supply, available from a wide variety of computer equipment retailers.

### Power Connector Type

The +5V power supply can be a typical PC compatible power supply. The J29 connector on the board uses the 12-pin power supply connector that mates with the two 6-pin power supply connectors on a standard PC power supply, as shown in Table 2.1.

### RC32134 Power Connector (J29)

| Pin | Definition | Color of Mating Connector Wire |
|---|---|---|
| 1 | — | Orange |
| 2 | +5V | Red |
| 3 | +12V | Yellow |
| 4 | -12V | Blue |
| 5 | Ground | Black |
| 6 | Ground | Black |
| 7 | Ground | Black |
| 8 | Ground | Black |
| 9 | — | White |
| 10 | +5V (vcc) | Red |
| 11 | +5V (vcc) | Red |
| 12 | +5V (vcc) | Red |

**Table 2.1 J29 Power Connectors**

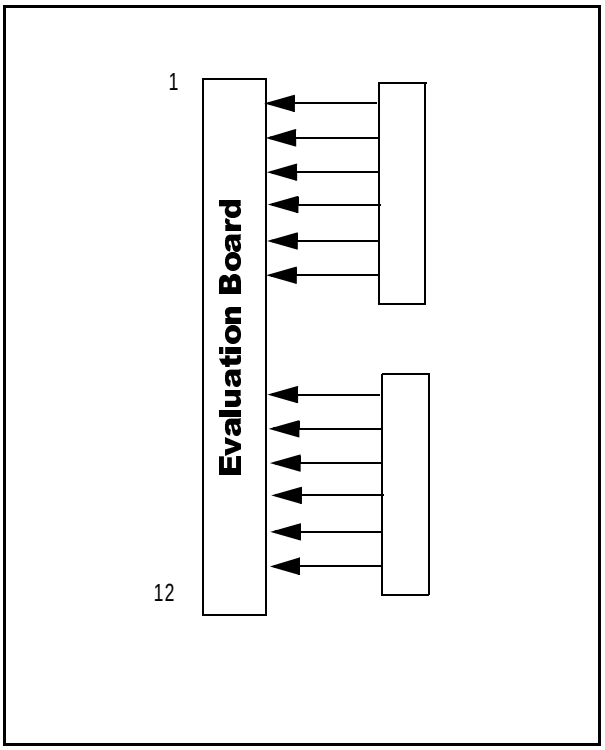



**Figure 2.1 Diagram of 6-Pin Power Supply Connections to S134 Evaluation Board**

## J23 (Debug Boot Connector)

Pin number and signal definitions are provided in Table 2.2.

| Debug Boot Connector[1] | | |
|---|---|---|
| **Pin Number** | **Name** | **Description** |
| 1 | TRST* | The TRST* pin is an active-low signal for asynchronous reset of the debug unit, independent of the processor logic. |
| 3 | TDI/DINT* | On the rising edge of Tclk, serial input data are shifted into either the Instruction or Data register, depending on the TAP controller state. During Real Mode, this input is used as an interrupt line to stop the debug unit from Real Time mode and return the debug unit back to Run Time Mode (standard JTAG). |
| 5 | TDO/TPC | The TDO is serial data shifted out from instruction or data register on the falling edge of Tclk. When no data is shifted out, the TDO is tri-stated. During Real Time Mode, this signal provides a non-sequential program counter at the processor clock or at a division of processor clock. |
| 7 | TMS | The logic signal received at the TMS input is decoded by the TAP controller to control test operation. TMS is sampled on the rising edge of the TCLK. |
| 9 | TCLK | An input test clock, used to shift into or out of the Boundary-Scan register cells. Tclk is independent of the system and the processor clock with nominal 50% duty cycle. |
| 11 | DBugBoot | The Debug Boot input is used during reset and forces the CPU core to take a debug exception at the end of the reset sequence instead of a reset exception. This enables the CPU to boot from the ICE probe without having the external memory working. This input signal is level sensitive and is not latched internally. This signal will also set the JtagBrk bit in the JTAG_Control_Register[12]. |
| 13 | Vcc 3.3 Pullup | |
| 15 | PCST0 | PC Trace Status Information<br>111 (STL) Pipe line Stall<br>110 (JMP) Branch/Jump forms with PC output<br>101 (BRT) Branch/Jump forms with no PC output<br>100 (EXP) Exception generated with an exception vector code output<br>011 (SEQ) Sequential performance<br>010 (TST) Trace is outputted at pipeline stall time<br>001 (TSQ) Trace trigger output at performance time<br>000 (DBM) Run Debug Mode<br>During power-on reset (cold reset), PCST(2:0) serves as ModeBit(2:0). |
| 17 | PCST1 | |
| 19 | PCST2 | |
| 21 | PCST3 | PC Trace Status Information. Reserved Pins for future expansion. During power-on reset, PCST(4:3) serves as ModeBit(4:3). |
| 23 | PCST4 | |
| 25 | DClk | Processor Clock. During Real Time Mode, this signal is used to capture address and data from the TDO signal at the processor clock speed, or any division of the internal pipeline. DCLK will be at 1/3 of the pipeline clock. |

**Table 2.2 J23 Debug Boot Connector**

1. All of the even numbered pins are Ground pins.

## Memory Blocks

The DRAM Controller space is a maximum of 256 Mbytes that can be populated at any time with either EDO or SDRAM DIMM in the connector slots shown in Table 2.3. Each DRAM type has two sockets, both capable of supporting x64 DRAM DIMMs.

| Connector | Memory Type |
|---|---|
| U1 | SRAM SIMM Slot |
| U12 | SDRAM DIMM Slot-1 |
| U16 | SDRAM DIMM Slot-2 |
| U22 | EDO DIMM Slot-1 |
| U23 | EDO DIMM Slot-2 |

**Table 2.3 RC32134 Evaluation Board Memory Block Connector Locations**

## J24 (Jtag Connector)

Table 2.4 lists the JTAG connector pins and signals for the 79RC32134 and PCI slots. For more operational details, refer to the IDT79RC32134 datasheet and the PCI bus specifications.

| Jtag Connector for RC32134 and PCI Slots | |
|---|---|
| **Pin Number** | **Name** |
| 1 | PCI_TRST1* |
| 2 | PCI_TRST* |
| 3 | PCI_TDI1 |
| 4 | PCI_TDI |
| 5 | PCI_TDO1 |
| 6 | PCI_TDO |
| 7 | PCI_TCK1 |
| 8 | PCI_TCK* |
| 9 | PCI_TMS1 |
| 10 | PCI_TMS |
| 11 | PCI_TRST2* |
| 12 | 79RC32134 TRST_N |
| 13 | PCI_TDI2 |
| 14 | 79RC32134 TDI |
| 15 | PCI_TDO2 |
| 16 | 79RC32134 TDO |
| 17 | PCI_TCK2 |
| 18 | 79RC32134 TCK |
| 19 | PCI_TMS2 |
| 20 | 79RC32134 TMS |
| 21 | PCI_TRST3* |
| 22 | N.C |
| 23 | PCI_TDI3 |
| 24 | PCI_TCK3 |
| 25 | PCI_TDO3 |
| 26 | PCI_TMS3 |

**Table 2.4 J24 Jtag Connector for RC32134 and PCI Slots**

# Default Jumper/Switch Settings

J21 and J22 Cache Test/Write Mode

| Jumper or Switch | Default |
|---|---|
| J21 | Closed |
| J22 | Closed |

**Table 2.5 Cache Test/Cache Write Mode Default Settings**

Jumper Strip (25) Programmable I/O Settings

| Jumper Selection | Setting | J25 Function |
|---|---|---|
| W4 | 1-2 | J25_1 = No connect **(Default)** |
| | 2-3 | J25_1 = SPI_DO/SE_MW_DO/PIO8 |
| W6 | 2-3 | J25_2 = No connect **(Default)** |
| | 1-2 | J25_2 = SPI_SK/SE_MW_SK/PIO10 |
| W5 | 1-2 | J25_3 = No connect **(Default)** |
| | 2-3 | J25_3 = SPI_CS/SE_MW_CS/PIO9 |
| W3 | 1-2 | J25_4 = No connect **(Default)** |
| | 2-3 | J25_4 = SPI_DI/SE_MW_DI/PIO11 |
| W19 | 1-2 | J25_5 = No connect **(Default)** |
| | 2-3 | J25_5= UART_TX0/PIO6 |
| W21 | 2-3 | J25_6 = UART_RX0/PIO7 |
| | 1-2 | J25_6 = No connect **(Default)** |
| W18 | 2-3 | J25_7 = UART_TX1/PIO4 |
| | 1-2 | J25_7 = No connect **(Default)** |
| W20 | 2-3 | J25_8 = UART_RX1/PIO5 |
| | 1-2 | J25_8 = No connect **(Default)** |

**Table 2.6 Programmable I/O Signal Special/General Settings**

Jumper Strip (26) Programmable I/O Settings

| Jumper Selection | Setting | J26 Function |
|---|---|---|
| W23 | 1-2 | J26_1 = No connect,<br>J26_7 = TMR_TCO_N/PIO3 |
| | 2-3 | J26_1 = TMR_TC0_N/ LED display RST_N/PIO3 (Default),<br>J26_7 = No connect |
| W22 | 1-2 | J26_2 = No connect,<br>J26_8 = TMR_TC1_N/PIO2 |
| | 2-3 | J26_2 = LED display SCLK/TMR_TC1_N/PIO2 (Default),<br>J26_8 = No connect |
| W2 | 1-2 | J26_3 = SCC DRQA_N/PIO1 |
| | 2-3 | J26_3 = LED display DATA/PIO1 (Default) |

**Table 2.7 Programmable I/O Signals (Page 1 of 2)**

| Jumper Selection | Setting | J26 Function |
|---|---|---|
| W24 | 1-2 | J26_4 = SCC DRQB_N/PIO0 |
| | 2-3 | J26_4 = LED display LOAD_N/PIO0 (Default) |
| | | J26_5 = No connect |
| | | J26_6 = No connect |

**Table 2.7 Programmable I/O Signals (Page 2 of 2)**

## CPU Power (Vcc) Jumper Settings

| Jumper (W8 - W12) | Setting | CPU $V_{cc}$ Selected |
|---|---|---|
| W8 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |
| W9 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |
| W10 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |
| W11 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |
| W12 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |

**Table 2.8 CPU Power Selection Table**

## RC32134 Power (Vcc) Jumper Settings

| Jumper (W13 - W17) | Setting | RC32134 $V_{cc}$ Selected |
|---|---|---|
| W13 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |
| W14 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |
| W15 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |
| W16 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |
| W17 | 1-2 | Reserved |
| | 2-3 | 3.3V (default) |

**Table 2.9 RC32134 Power Selection Table**

## DIP Switch 2 Settings

| S2 | ON | OFF |
|---|---|---|
| 1 | SCC_DTR_REQA is RC32134 DRQ0 | PULL UP **(Default)** |
| 2 | SCC_W_REQA is RC32134 DRQ0 | |
| 3 | SCC_DTR_REQB is RC32134 DRQ1 | |
| 4 | SCC_W_REQB is RC32134 DRQ1 | |
| 5 | | |
| 6 | Clock divide by 1 | Clock divide by 2 **(Default)** |
| 7 | | |
| 8 | | |

**Table 2.10 SCC Switch 2 Settings**

## DIP Switch 3 Settings

| S3 | Setting | Feature |
|---|---|---|
| 1 (BPROMW0) | closed | 8-bit Boot PROM width (Default) |
| 2 (BPROMW1) | closed | |
| 1 (BPROMW0) | open | 16-bit Boot PROM width |
| 2 (BPROMW1) | closed | |
| 1 (BPROMW0) | closed | 32-bit Boot PROM width |
| 2 (BPROMW1) | open | |
| 1 (BPROMW0) | open | Reserved |
| 2 (BPROMW1) | open | |

**Table 2.11 Boot PROM Width Selections**

| S3 | Setting | Feature |
|---|---|---|
| 3 (Reserved) | | |
| 4 (Flash Memory) | Open | Disable FLASH and Enable EPROM. **(Default)** |
| | Closed | Enable FLASH Memory. |
| 5 (Unused) | | |
| 6 (PCI_Host_mode) | Closed | PCI is in host mode. **(Default)** |
| | Open | PCI is in Satellite mode |
| 7 (BRHEA0) | Closed | Boot from RC32134 memory controller. **(Default)** |
| 8 (BRHEA1) | Closed | |
| 7 (BRHEA0) | Open | Boot from PCI. Use PCI EEPROM to pre-load PCI configuration registers. **(Default)** |
| 8 (BRHEA1) | Closed | |

**Table 2.12 PCI and Boot Mode Configuration settings (Page 1 of 2)**

| S3 | Setting | Feature |
|---|---|---|
| 7 (BRHEA0) | Closed | Reserved **(Default)** |
| 8 (BRHEA1) | Open | |
| 7 (BRHEA0) | Open | Idle at reset. RC32134 does not supply boot-code control. **(Default)** |
| 8 (BRHEA1) | Open | |

**Table 2.12 PCI and Boot Mode Configuration settings (Page 2 of 2)**

## DIP Switch 4 Settings

These switch settings control the Mode-bit features and other miscellaneous user selectable features and are implemented during the Power-on or Cold reset stages. Switch 4-1, 2, and 3 implement the clock multiplier used to generate the pipe-line frequency multiplied by the input clock, Mclk.

| S4 | Setting | Feature |
|---|---|---|
| 1 (MCLKX0) | Closed | Clock multiplier X2 **(Default)** |
| 2 (MCLKX1) | Closed | |
| 3 (MCLKX2) | Closed | |
| 1 (MCLKX0) | Open | Clock multiplier X3 |
| 2 (MCLKX1) | Closed | |
| 3 (MCLKX2) | Closed | |
| 1 (MCLKX0) | Closed | Clock multiplier X4 |
| 2 (MCLKX1) | Open | |
| 3 (MCLKX2) | Closed | |
| 1 (MCLKX0) | Open | Clock multiplier X5 |
| 2 (MCLKX1) | Open | |
| 3 (MCLKX2) | Closed | |
| 1 (MCLKX0) | Closed | Clock multiplier X6 |
| 2 (MCLKX1) | Closed | |
| 3 (MCLKX2) | Open | |
| 1 (MCLKX0) | Open | Clock multiplier X7 |
| 2 (MCLKX1) | Closed | |
| 3 (MCLKX2) | Open | |
| 1 (MCLKX0) | Closed | Clock multiplier X8 |
| 2 (MCLKX1) | Open | |
| 3 (MCLKX2) | Open | |
| 1 (MCLKX0) | Open | Reserved |
| 2 (MCLKX1) | Open | |
| 3 (MCLKX2) | Open | |

**Table 2.13 Clock Multiplier Configurations**

| S4 | Setting | Feature |
|---|---|---|
| 4 Endianness | Open | Big Endian **(Default)** |
| | Closes | Little Endian |
| 5 Reserved | Closed | (Default) |
| 6 Reserved | Closed | (Default) |
| 7 Timer Interrupt | Closed | Enabled **(Default)** |
| | Open | Disabled |
| 8 Reserved | Open | (Default) |

**Table 2.14 Endianness, PLL, and Slew Rate Control Configurations**

## System Software - IDT/sim

EPROMs on the 79S134 contain IDT's System Integration Manager (IDT/sim). IDT/sim is a software boot PROM debug monitor that provides functions for downloading software and for integrating hardware with software. Using IDT/sim, software can be downloaded onto the board from a SUN SPARCstation™ or a PC/AT personal computer.

Drivers are added easily by using the IDT Cross Development Software IDT/kit, and the IDT/sim source code can be acquired to support other I/O devices or change I/O addresses to fit their specific application: for example, to change from big-endian to little-endian addressing. The S134 board's default configuration is big-endian addressing. A copy of IDT/sim can be obtained through your local IDT sales representative.

## Serial Port for CRT Video Terminal & Auxiliary Port

The 79S134 system board has four RS232 serial port connectors with pin assignments as shown in Table 2.15 and Table 2.16. The console port for the board is the DB9P connector designated as J1/J3[1] and must be set for a data rate of 9600 baude with 8 bits of data, no parity bit, and one stop bit. The J2/J4 auxiliary port is also a DB9P connector and is used for functions such as down loading software from a PC or SPARCstation™.

| SCC Port - A Console Connector J1 | | SCC Port - B Auxiliary Connector J2 | |
|---|---|---|---|
| Pin | Signal | Pin | Signal |
| 1 | No connection | 1 | No connection |
| 2 | Txd (Output) | 2 | Txd (Output) |
| 3 | Rxd (Input) | 3 | Rxd (Input) |
| 4 | No connection | 4 | No connection |
| 5 | Ground | 5 | Ground |
| 6 | No connection | 6 | No connection |
| 7 | No connection | 7 | No connection |
| 8 | No connection | 8 | No connection |
| 9 | No connection | 9 | No connection |

**Table 2.15 J1/J2 Connector Pins and Signal Descriptions**

1. Based on SIM (for external/internal UART) the console will either be J1 or J3, respectively.

| 79RC32134 UART - A Console Connector J3 | | 79RC32134 UART- B Auxiliary Connector J4 | |
|---|---|---|---|
| Pin | Signal | Pin | Signal |
| 1 | No connection | 1 | No connection |
| 2 | Txd (Output) | 2 | Txd (Output) |
| 3 | Rxd (Input) | 3 | Rxd (Input) |
| 4 | No connection | 4 | No connection |
| 5 | Ground | 5 | Ground |
| 6 | No connection | 6 | No connection |
| 7 | No connection | 7 | No connection |
| 8 | No connection | 8 | No connection |
| 9 | No connection | 9 | No connection |

**Table 2.16 J3/J4 Connector Pins and Signal Descriptions**

# Initialization and System Start-Up

System start-up is performed by turning the power supply on. If power to the board has already been supplied, then pressing the reset button will reinitialize the board. Two of the board's three LED displays indicate that the power has been successfully applied and one indicates the status of reset, as follows:

- ***DS1*** *indicates that the* ***5V*** *power is* ***ON*** *(Green)*
- ***DS2*** *indicates that the* ***Cold Reset*** *is* ***active*** *(Red)*
- ***DS3*** *indicates that the* ***3.3V*** *power supply is* ***ON*** *(Yellow)*

Once started, IDT/ sim automatically boots and sizes the internal cache and main memory. The console is connected via the CRT serial port and a message indicating cache and memory sizes—similar to the one shown in Figure 2.2—will appear along with the first command line prompt. For more information on SIM commands, refer to the IDT/ sim User/Developer's Manual.

**Note:** Future upgrades will be assigned a different version number and date. **The starting address of the free memory space may differ slightly from the example shown** in Figure 2.2.

```
IDT System Integration Manager Ver. 6.5 May, 1995
Copyright 1994, 1995 Integrated Device Technology, Inc.
For help enter '?'
Memory size: 1048576 (0x100000) bytes
Icache size: 4096 (0x1000) bytes
Dcache size: 1024 (0x400) bytes
User Memory Space 0xa00082dc-0xa00ffffc
CPU: RC32364 Default baud rate: 9600 Register: 32-bit
ENDIAN:     Big
<IDT>
```

**Figure 2.2 Initial Screen Display for the IDT/sim Debug Monitor**

# Logic Analyzer Connections

J5,J6,J7,J9,J10,J11,J13,J14,J15, J17, and J18 can be used to connect directly to an HP Logic Analyzer. The pin numbers and signal descriptions for each connector are listed in the following tables.

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C | 11 | CPU_AD7 |
| 2 | N.C | 12 | CPU_AD8 |
| 3 | RHEA_ clk1 | 13 | CPU_AD9 |
| 4 | CPU_AD0 | 14 | CPU_AD10 |
| 5 | CPU_AD1 | 15 | CPU_AD11 |
| 6 | CPU_AD2 | 16 | CPU_AD12 |
| 7 | CPU_AD3 | 17 | CPU_AD13 |
| 8 | CPU_AD4 | 18 | CPU_AD14 |
| 9 | CPU_AD5 | 19 | CPU_AD15 |
| 10 | CPU_AD6 | 20 | GND |

**Table 2.17 Analyzer POD Connector J5**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C. | 11 | CPU_AD23 |
| 2 | N.C. | 12 | CPU_AD24 |
| 3 | RHEA_clk2 | 13 | CPU_AD25 |
| 4 | CPU_AD16 | 14 | CPU_AD26 |
| 5 | CPU_AD17 | 15 | CPU_AD27 |
| 6 | CPU_AD18 | 16 | CPU_AD28 |
| 7 | CPU_AD19 | 17 | CPU_AD29 |
| 8 | CPU_AD20 | 18 | CPU_AD30 |
| 9 | CPU_AD21 | 19 | CPU_AD31 |
| 10 | CPU_AD22 | 20 | GND |

**Table 2.18 Analyzer POD Connector J6**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C. | 11 | CPU_BE0 |
| 2 | N.C. | 12 | CPU_BE1 |
| 3 | RHEA_CLK3 | 13 | CPU_BE2 |
| 4 | CPU_ADDR2 | 14 | CPU_BE3 |
| 5 | CPU_ADDR3 | 15 | CPU_BERR_N |
| 6 | CPU_ALE | 16 | CPU_CRST_N |
| 7 | CPU_WR_N | 17 | RH_RST_N |
| 8 | CPU_CIP_N | 18 | CPU_BREQ_N |
| 9 | CPU_ACK_N | 19 | CPU_BGNT_N |
| 10 | CPU_LAST | 20 | GND |

**Table 2.19 Analyzer POD Connector J7**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C | 11 | R_CAS2 |
| 2 | N.C | 12 | R_CAS3 |
| 3 | N.C | 13 | R_245OE_N |
| 4 | RH_INTR3_N | 14 | R_WE_N |
| 5 | R_RAS0 | 15 | R_CKE |
| 6 | R_RAS1 | 16 | R_SDCAS_N |
| 7 | R_RAS2 | 17 | R_SDRAS_N |
| 8 | R_RAS3 | 18 | RH_TC1_N |
| 9 | R_CAS0 | 19 | RH_TC0_N |
| 10 | R_CAS1 | 20 | GND |

**Table 2.20 Analyzer POD Connector J9**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C | 11 | RH_TDO |
| 2 | N.C | 12 | RH_TDI |
| 3 | N.C | 13 | RH_TMS |
| 4 | RH_DRQ1 | 14 | RH_TCK |
| 5 | RH_DRQ0 | 15 | R_MEM_WE0_N |
| 6 | RH_RX0 | 16 | R_MEM_WE1_N |
| 7 | RH_RX1 | 17 | R_MEM_WE2_N |
| 8 | RH_TX0 | 18 | R_MEM_WE3_N |
| 9 | RH_TX1 | 19 | R_MEM_OE_N |
| 10 | RH_TRST_N | 20 | GND |

**Table 2.21 Analyzer POD Connector J10**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C | 11 | R_MEM_245_OE_N |
| 2 | N.C | 12 | CPU_DT/R_N |
| 3 | CPU_MCLK | 13 | RH_SE_DI |
| 4 | R_MEM_CS0_N | 14 | RH_SE_DO |
| 5 | R_MEM_CS1_N | 15 | RH_SE_CS |
| 6 | R_MEM_CS2_N | 16 | RH_SE_SK |
| 7 | R_MEM_CS3_N | 17 | N.C |
| 8 | R_MEM_CS4_N | 18 | N.C |
| 9 | R_MEM_CS5_N | 19 | N.C |
| 10 | R_MEM_WAIT_N | 20 | GND |

**Table 2.22 Analyzer POD Connector J11**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C | 11 | PCI_AD7 |
| 2 | N.C | 12 | PCI_AD8 |
| 3 | N.C | 13 | PCI_AD9 |
| 4 | PCI_AD0 | 14 | PCI_AD10 |
| 5 | PCI_AD1 | 15 | PCI_AD11 |
| 6 | PCI_AD2 | 16 | PCI_AD12 |
| 7 | PCI_AD3 | 17 | PCI_AD13 |
| 8 | PCI_AD4 | 18 | PCI_AD14 |
| 9 | PCI_AD5 | 19 | PCI_AD15 |
| 10 | PCI_AD6 | 20 | GND |

**Table 2.23 Analyzer POD Connector J13**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C | 11 | PCI_AD23 |
| 2 | N.C | 12 | PCI_AD24 |
| 3 | N.C | 13 | PCI_AD25 |
| 4 | PCI_AD16 | 14 | PCI_AD26 |
| 5 | PCI_AD17 | 15 | PCI_AD27 |
| 6 | PCI_AD18 | 16 | PCI_AD28 |
| 7 | PCI_AD19 | 17 | PCI_AD29 |
| 8 | PCI_AD20 | 18 | PCI_AD30 |
| 9 | PCI_AD21 | 19 | PCI_AD31 |
| 10 | PCI_AD22 | 20 | GND |

**Table 2.24 Analyzer POD Connector J14**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C | 11 | PCI_GNT1# |
| 2 | N.C | 12 | PCI_GNT2# |
| 3 | R_PCI_CLK | 13 | PCI_GNT3# |
| 4 | PCI_C/BE3# | 14 | PCI_LOCK# |
| 5 | PCI_C/BE2# | 15 | PCI_PAR |
| 6 | PCI_C/BE1# | 16 | PCI_RST# |
| 7 | PCI_C/BE0# | 17 | PCI_TRDY# |
| 8 | PCI_REQ1# | 18 | PCI_IRDY# |
| 9 | PCI_REQ2# | 19 | PCI_STOP# |
| 10 | PCI_REQ3# | 20 | GND |

**Table 2.25 Analyzer POD Connector J15**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C | 11 | RH_ADS9 |
| 2 | N.C | 12 | RH_ADS10 |
| 3 | N.C | 13 | RH_ADS11 |
| 4 | RH_ADS2 | 14 | RH_ADS12 |
| 5 | RH_ADS3 | 15 | RH_ADS13 |
| 6 | RH_ADS4 | 16 | RH_ADS14 |
| 7 | RH_ADS5 | 17 | RH_ADS15 |
| 8 | RH_ADS6 | 18 | RH_ADS16 |
| 9 | RH_ADS7 | 19 | RH_ADS17 |
| 10 | RH_ADS8 | 20 | GND |

**Table 2.26 Analyzer POD Connector J17**

| Pin | Signal | Pin | Signal |
|---|---|---|---|
| 1 | N.C | 11 | PCI_SERR# |
| 2 | N.C | 12 | PCI_DEVSEL# |
| 3 | N.C | 13 | N.C |
| 4 | RH_ADS18 | 14 | N.C |
| 5 | RH_ADS19 | 15 | N.C |
| 6 | RH_ADS20 | 16 | N.C |
| 7 | RH_ADS21 | 17 | N.C |
| 8 | RH_ADS22 | 18 | N.C |
| 9 | PCI_FRAME# | 19 | N.C |
| 10 | PCI_PERR# | 20 | GND |

**Table 2.27 Analyzer POD Connector J18**

## ntdatalst1col - Master page style - ntdatalst1col

A **boot-time mode control interface** initializes fundamental processor modes. The boot-time mode control interface is a serial interface that operates at a very low frequency (MasterClock divided by 256).

## About ntdatalst - Master page style - ntdatalst

It has 2 columns and has the same information at the bottom as this page.

> **Note:** Make sure the "***for Tech Support:***" information in the lower right corner of this page is appropriate for the particular data sheet you are creating.

# Theory of Operation and Design Notes

Notes

## Introduction

This chapter provides information on the functional operation of the IDT79S134 evaluation board for the RC32134 system controller chip. For detailed schematics refer to Chapter 4. For detailed PLD equations, refer to Chapter 5.

## Address Space Decoding

The physical addresses of the S134 board's resources are as listed in the tables that follow. The EPROM/FLASH, SRAM and Serial Communication Controller subsystems can be accessed through selection of the Memory chipselects[0,1,5].

The memory_I/O controller includes the EPROM/FLASH subsystem, Serial I/O and SRAM subsystems. The **EPROM/FLASH** module is accessible through **mem_chipselect[0]** (see Table 3.16). Selection between EPROM or FLASH memory space is achieved through Switch 3- 4, as shown in Table 2.9. The **SRAM** memory space is 1 Mbytes and is accessed through **mem_chipselec[1]** (see Table 3.16). The 85C30 **SCC** controller interfaces two RS232 connectors and is located at the fixed address space listed in Table 3.1, which is selected through **mem_chipselect[5]**.

| Description | Physical Address Locations |
|---|---|
| ROM | 1FC0_0000 – 1FFF_FFFF |
| Flash | 1FC0_0000 – 1FDF_FFFF |
| SRAM | 0000_0000 – 000F_FFFF |
| DRAM (EDO/SDRAM) | |
| Bank size is 8MB | 0080_0000 – 00FF_FFFF |
| Bank size is 8MB | 0100_0000 – 017F_FFFF |
| SCC (85C30) | 1600_0000 – 17FF_FFFF |
| 79RC32134 Internal Registers | 1800_0000 – 1BFF_FFFF |

**Table 3.1 Physical Address Mapping of 79S341 Board Resources**

The **DRAM Controller** space is a maximum of 256 Mbytes that can be populated at any time with either EDO or SDRAM DIMM. Each DRAM type has two sockets, both capable of supporting x64 DRAM DIMMs. When the SRAM option is selected, the SRAM and DRAM address locations and ranges change as shown in Table 3.2, Table 3.3. When SRAM is not selected, the address mappings are as shown in Table 3.4 and Table 3.5.

### SRAM Selected, 16Mbyte DRAM DIMM

| SRAM Address Range | | DRAM Address Range | | DRAM Base Addresses | DRAM Mask Addresses |
|---|---|---|---|---|---|
| From | To | From | To | Banks 3:0 | Banks 3:0 |
| 0000_0000 | 000F_FFFF | 0080_0000 | 027F_FFFF | 0080_0000 | FF80_0000 |
| | | | | 0100_0000 | |
| | | | | 0180_0000 | |
| | | | | 0200_0000 | |

**Table 3.2 SRAM/DRAM Address Mapping, 16Mbyte DRAM DIMM**

### SRAM Selected, 64Mbyte DRAM DIMM

| SRAM Address Range | | DRAM Address Range | | DRAM Base Addresses | DRAM Mask Addresses |
|---|---|---|---|---|---|
| From | To | From | To | Banks 3:0 | Banks 3:0 |
| 0000_0000 | 000F_FFFF | 0200_0000 | 09FF_FFFF | 0200_0000 | FE00_0000 |
| | | | | 0400_0000 | |
| | | | | 0600_0000 | |
| | | | | 0800_0000 | |

Table 3.3 SRAM/DRAM Address Range, 64Mbyte DRAM DIMM

### SRAM Not Selected, 16Mbyte DRAM DIMM

| DRAM Address Range | | DRAM Base Addresses | DRAM Mask Addresses |
|---|---|---|---|
| From | To | Banks 3:0 | |
| 0000_0000 | 01FF_FFFF | 0000_0000 | FF80_0000 |
| | | 0080_0000 | |
| | | 0100_0000 | |
| | | 0180_0000 | |

Table 3.4 DRAM Address Range, 16Myte DRAM DIMM & SRAM Not Selected

### SRAM Not Selected, 64Mbyte DRAM DIMM

| DRAM Address Range | | DRAM Base Addresses | DRAM Mask Addresses |
|---|---|---|---|
| From | To | Banks 3:0 | |
| 0000_0000 | 007FF_FFFF | 0000_0000 | FE00_0000 |
| | | 0200_0000 | |
| | | 0400_0000 | |
| | | 0600_0000 | |

Table 3.5 DRAM Address Range, 64Myte DRAM DIMM & SRAM Not Selected

The **PIO Controller** supports either 12-bit general purpose discrete I/O or specific peripheral functions. As general purpose discrete I/O pins, the controller supports I/O functions such as the 4-digit LED display and micro wire serial EEPROM. Specific peripheral I/O functions such as on-chip UART data I/O pins, SPI, timers and DMA are also supported. Each of these functions are implemented through Jumpers 25 and 26 as shown in Table 2.3 and 2.4. The address map of the PIO controller is shown in Table 3.6.

| PIO Controller Address Mapping | |
|---|---|
| From | To |
| 1800_0600 | 1800_0608 |

Table 3.6 PIO Controller Address Mapping

The **PCI bus interface** resources include a control core that provides a master and target controller that uses transmit and receive FIFO sizes of 8 words. The bus interface to the PCI core provides PCI bus arbitration selection, external bus request and bus grant modes, internal RC32134 arbiter mode with fixed and round robin priority selections, mailbox registers, and software programmable endianness (selectable per memory block). The address mapping for these registers is shown in Table 3.8. The internal address map for the PCI interface is as shown in Table 3.7.

| From | To | Allocation |
|---|---|---|
| 1800_2000 | 1800_2FFF | Internal registers (4KB) |
| 1880_0000 | 188F_FFFF | PCI I/O Space (1MB) |
| 18C0_0000 | 18FF_FFFF | Memory space 3 (4MB) (for non-pci boot reset option) |
| 1FC0_0000 | 1FFF_FFFF | Memory space 3 (4MB) (for pci boot reset option) |
| 4000_0000 | 5FFF_FFFF | Memory Space 1 (512MB) |
| 6000_0000 | 7FFF_FFFF | Memory Space 2 (512MB) |

**Table 3.7 PCI Interface Address Ranges and Definitions**

| Address | Registers |
|---|---|
| 1800_05B0 | PCI Controller Interrupt Pending Register 11 |
| 1800_05B4 | PCI Controller Interrupt Mask Register 11 |
| 1800_05B8 | PCI Controller Interrupt Clear Register 11 |
| 1800_05C0 | PCI Satellite Mode Mailbox Interrupt Pending Register 12 |
| 1800_05C4 | PCI Satellite Mode Mailbox Interrupt Mask Register 12 |
| 1800_05C8 | PCI Satellite Mode Mailbox Interrupt Clear Register 12 |
| 1800_05D0 | PCI to CPU Mailbox Interrupt Pending Register 13 |
| 1800_05D4 | PCI to CPU Mode Mailbox Interrupt Mask Register 13 |
| 1800_05D8 | PCI to CPU Mailbox Interrupt Clear Register 13 |
| 1800_20B0 | PCI Memory Space 1 Base Register |
| 1800_20B8 | PCI Memory Space 2 Base Register |
| 1800_20C0 | PCI Memory Space 3 Base Register |
| 1800_20C8 | PCI I/O Space Base Register |
| 1800_20E0 | PCI Arbitration Register |
| 1800_20E8 | PCI Host Memory Space 1 Base Register |
| 1800_2100 | PCI Host IO space Base Register |
| 1800_2CF8 | PCI Configuration Address Register |
| 1800_2CFC | PCI Configuration Data Register |

**Table 3.8 PCI Register Map**

Four general purpose **DMA channels**[1] move data between source and destination resources such as system memory, PCI or external I/O devices (8-,16-,or 32-bit I/O devices are treated as memory-mapped word-aligned devices). Using a flexible, memory-based descriptor structure, any of the four channels efficiently supports "scatter/gather" capability.

The RC32134 DMA supports byte, half-word (16-bit), word, and quad-word burst transfers that cross-over quad-word boundaries and are automatically split into single-word transfers until a quad-word boundary is reached. The DMA controller also automatically prevents burst transfers from crossing page boundaries and supports little- or big-endian data conversions. DMA restrictions include:

- *When the source or destination address is constant (as in I/O devices), it must be word aligned.*
- *DMA is not supported for internal UART.*

And the following transfers are not supported:

- *Source is incremented and destination is decremented*
- *Source is decremented and destination is incremented.*

Additional information on DMA operations is located in the RC32134 hardware user's manual.

## Register Address Maps for DMA Channels 0-3

| Base Address Channel 0 | Register Name | Offset Address | Effective Address Channel 0 |
|---|---|---|---|
| 1800_1400 | Configuration Register | 00 | Base + Offset |
| | Base Descriptor Register | 04 | |
| | Current Address Register | 08 | |
| | Status/Block Size Register | 10 | |
| | Source Address Register | 14 | |
| | Destination Address Register | 18 | |
| | Nest Descriptor Address Register | 1C | |

**Table 3.9 DMA Channel 0 Register Mapping**

| Base Address Channel 1 | Register Name | Offset Address | Effective Address Channel 1 |
|---|---|---|---|
| 1800_1400 | Configuration Register | 40 | Base + Offset |
| | Base Descriptor Register | 44 | |
| | Current Address Register | 48 | |
| | Status/Block Size Register | 50 | |
| | Source Address Register | 54 | |
| | Destination Address Register | 58 | |
| | Nest Descriptor Address Register | 5C | |

**Table 3.10 DMA Channel 1 Register Mapping**

1. DMA channels 3 and 4 do not have the DMA_RDY pins and can not be used to perform DMA transfers with slow I/O devices.

| Base Address Channel 2 | Register Name | Offset Address | Effective Address Channel 2 |
|---|---|---|---|
| 1800_1900 | Configuration Register | 00 | Base + Offset |
| | Base Descriptor Register | 04 | |
| | Current Address Register | 08 | |
| | Status/Block Size Register | 10 | |
| | Source Address Register | 14 | |
| | Destination Address Register | 18 | |
| | Nest Descriptor Address Register | 1C | |

**Table 3.11 DMA Channel 2 Register Mapping**

| Base Address Channel 3 | Register Name | Offset Address | Effective Address Channel 3 |
|---|---|---|---|
| 1800_1900 | Configuration Register | 40 | Base + Offset |
| | Base Descriptor Register | 44 | |
| | Current Address Register | 48 | |
| | Status/Block Size Register | 50 | |
| | Source Address Register | 54 | |
| | Destination Address Register | 58 | |
| | Nest Descriptor Address Register | 5C | |

**Table 3.12 DMA Channel 3 Register Mapping**

The **Expansion Interrupt Controller** extends the CPU's CP0 interrupt control by collating the RC32134 generated interrupts into a single CPU interrupt. When a general purpose interrupt is received, the Interrupt Service Routine (ISR) first saves CPU registers, checks its Cause Register and then checks its Pending Interrupt Register. If the pending interrupt is from the RC32134, then the ISR checks the Expansion Interrupt Controller Pending Interrupt Register. After treating/noting the interrupt condition, the ISR resets the pending interrupt by writing to the corresponding bit in the Expansion Interrupt Clear Register. The ISR can then exit by restoring the CPU register and executing an RFE instruction. The register address mapping for the Expansion Interrupt Controller is shown in Table 3.13.

| Expansion Interrupt Controller Address Mapping | |
|---|---|
| From | To |
| 1800_0500 | 1800_05e8 |

**Table 3.13 Expansion Interrupt Controller Address Mapping**

The RC32134 has eight on-chip **Timers**: Three general purpose timers and five timers that are optionally dedicated to Watchdog, CPU bus time-out, IP bus time-out, DRAM refresh, and WarmReset. Beginning from zero, these eight system timers count on each system clock, timing out after reaching a programmable compare value and resetting to zero automatically. Uses for these timers include real-time clock, cascaded real-time clock and time-slice clock. The register address mapping for the Timer controller is shown in Table 3.14. Additional information on the functional aspects of these timers is located in the RC32134 hardware user's manual.

| Timer Controller Address Mapping | |
|---|---|
| From | To |
| 1800_0700 | 1800_0778 |

**Table 3.14 Timer Controller Address Mapping**

The two 16550 UARTs are an enhanced version of the 16450 UART. Functionally the same as a 16450 at power-up, these UARTs can be put into the 16550 mode, which then relieves the CPU of software overhead. This feature allows execution of 16450 or 16550 compatible software. Two sets of 16-byte buffers are enabled during the 16550 mode: one set in the receive data path and one set in the transmit data path.

The CPU can read the UART status at any time during operation. Status information includes the type and condition of the transfer operation, as well as any error condition (parity, overrun, framing, or break interrupt). A baud rate generator is included that divides down the system clock by 1 to 65K. The baud rate generator provides the 16X clock for driving the transmitter and receiver logic.

The UART controller provides fully programmable serial characteristics such as 5, 6, 7 or 8-bit characters; even, odd or no parity bit generation and detection; and 1, 1-1/2 or 2 stop bit generation. The register address mapping for the UART Controller is shown in Table 3.15.

| UART Controller Address Mapping | |
|---|---|
| From | To |
| 1800_0800 | 1800_083C |

**Table 3.15 UART Controller Address Map**

# Interrupts

Both the on-board and PCI bus interface interrupts are assigned to the CPU as shown in Table 3.16.

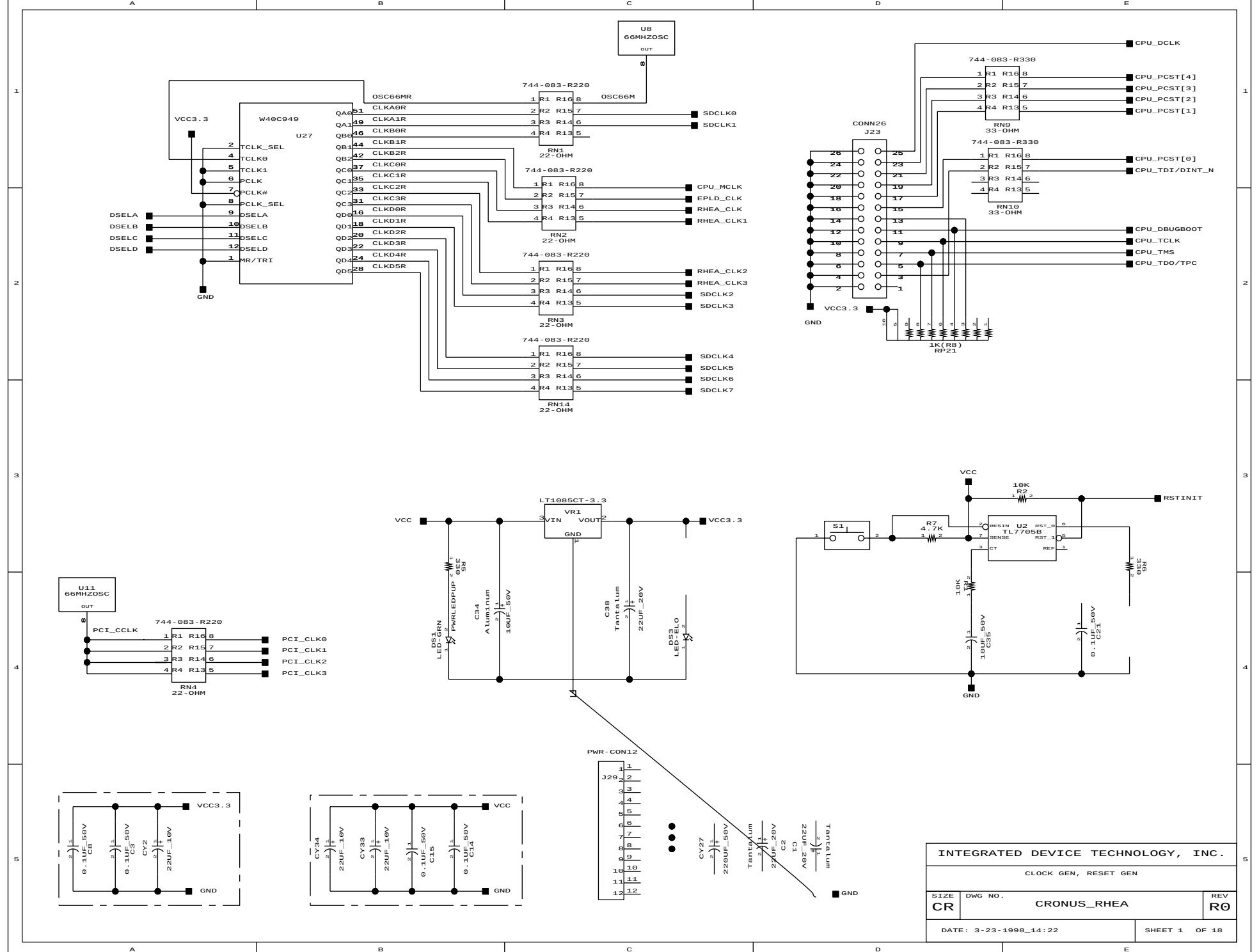
| Interrupts | |
|---|---|
| INT0* | Scc (85C30) INTR. |
| INT1* | PCI Bus INTA# |
| INT2* | PCI Bus INTB# |
| INT3 | 79RC32134 INTR. |
| INT4* | PCI Bus INTC# |
| INT5* | PCI Bus INTD# |
| NMI* | N.C. (Unused) |

**Table 3.16 CPU Interrupt Assignment for S134 Board**

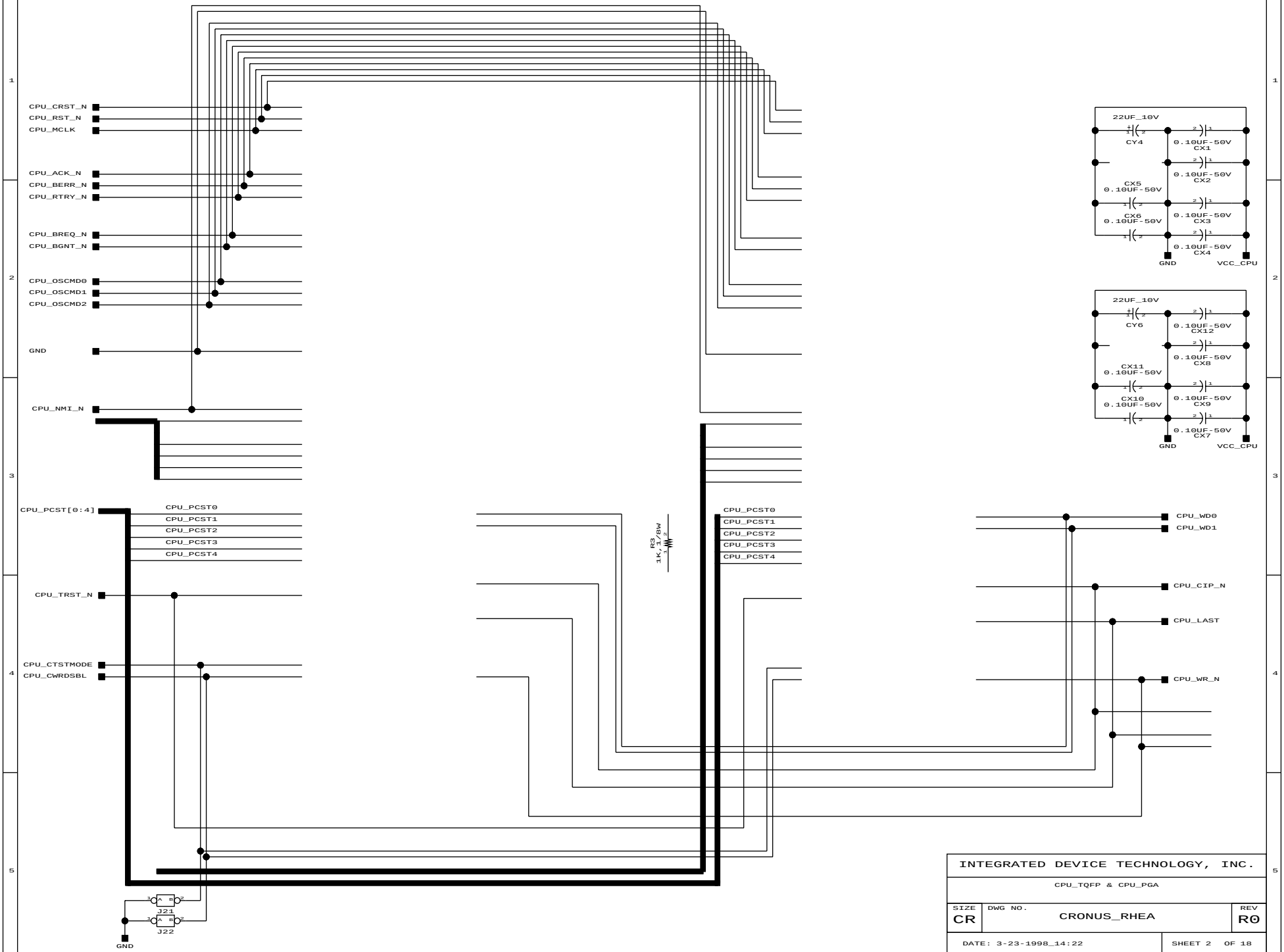
# Schematics

Notes

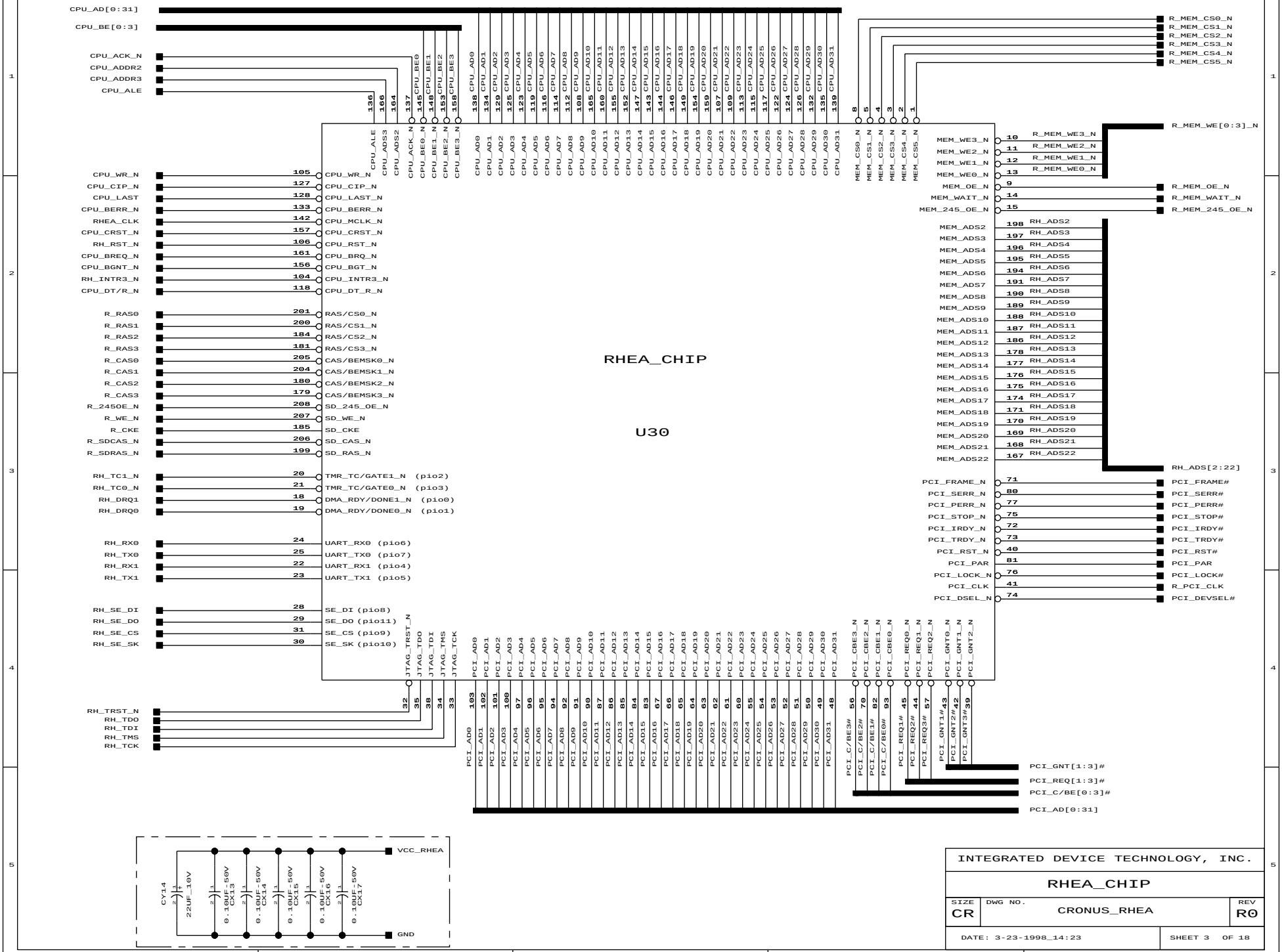
## Schematics



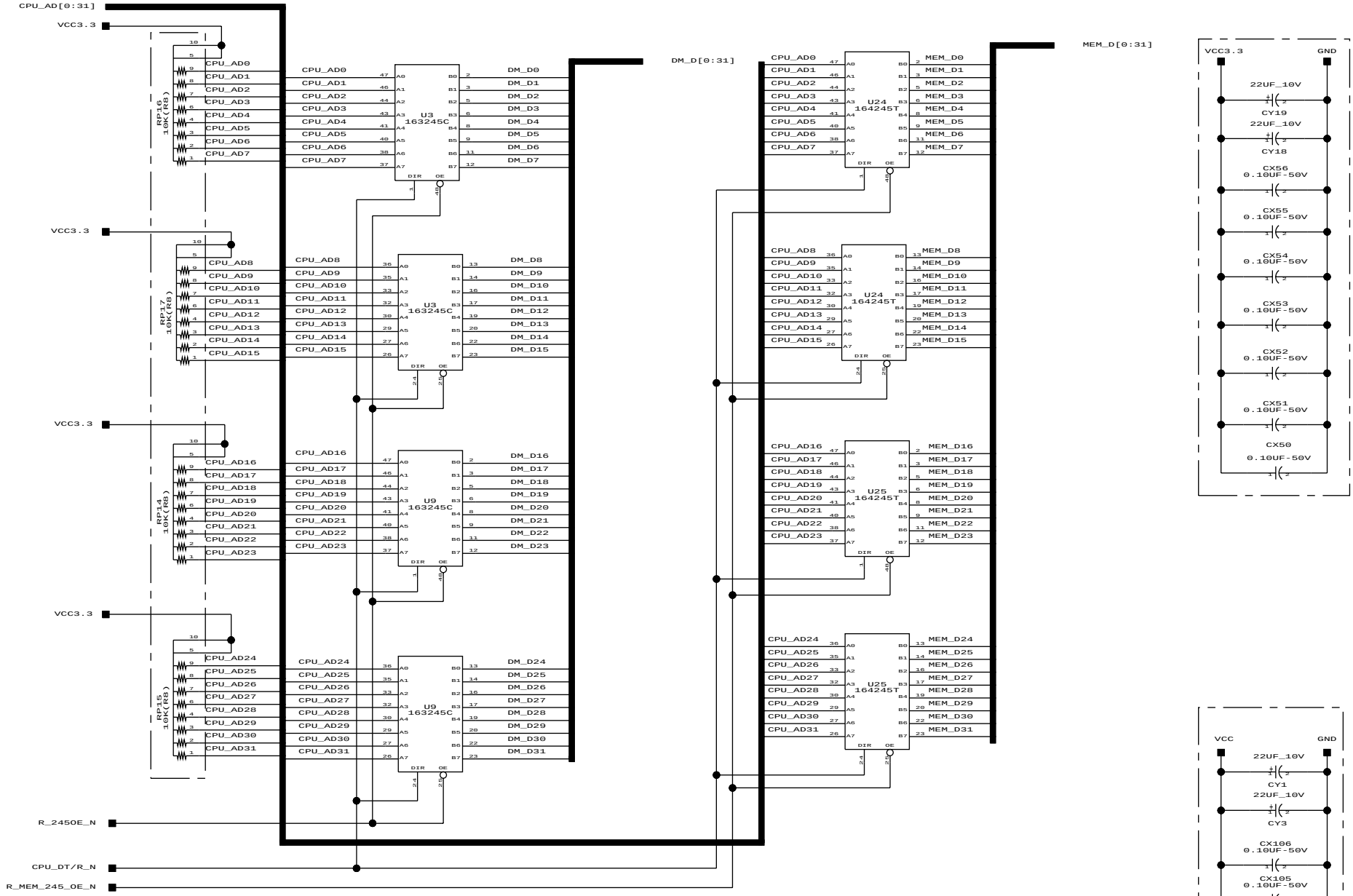

| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| CLOCK GEN, RESET GEN | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:22 | SHEET 1 OF 18 | |




| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| CPU_TQFP & CPU_PGA | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:22 | SHEET 2 OF 18 | |




| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| RHEA_CHIP | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:23 | SHEET 3 OF 18 | |

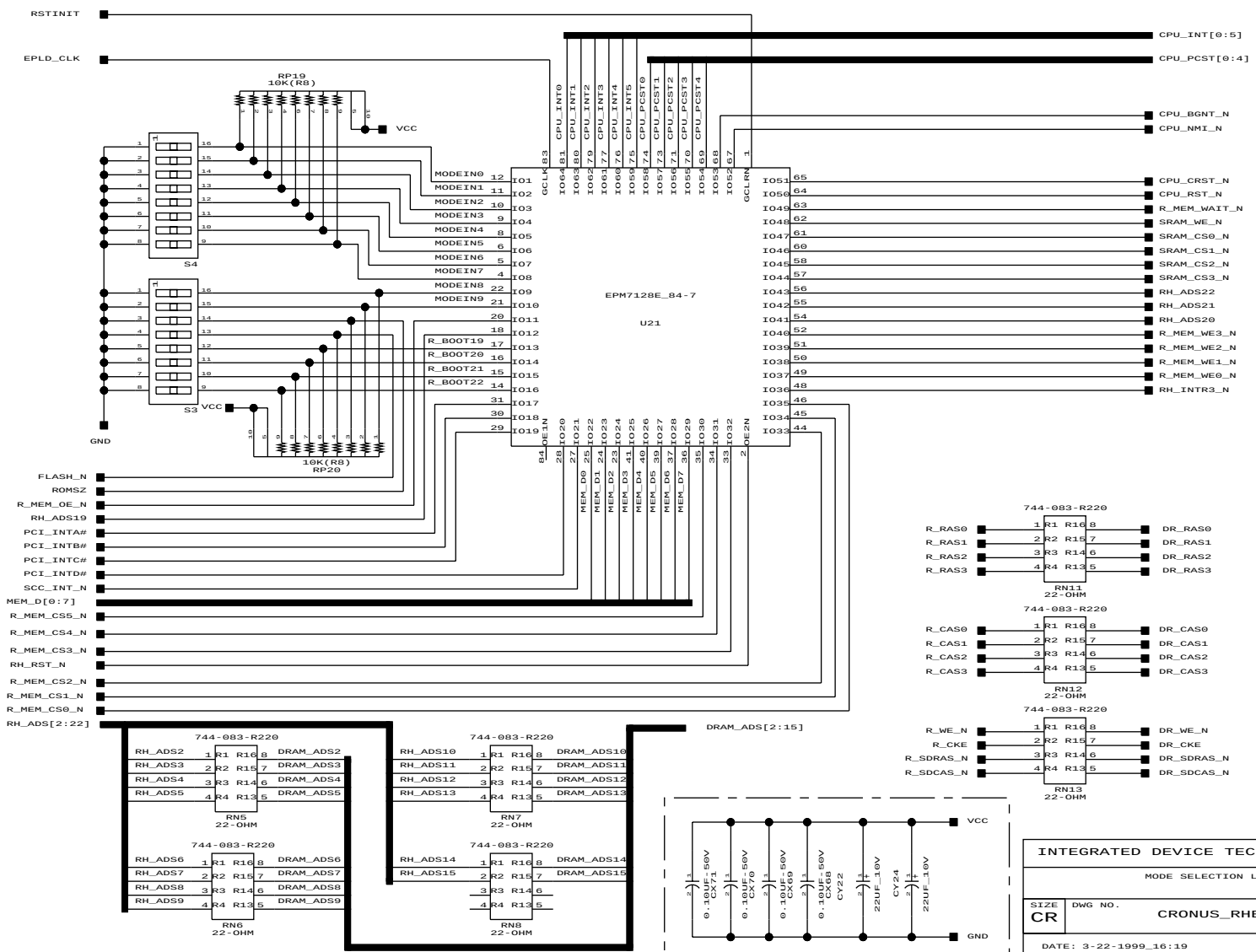



Keep these resistor network nearer to processor

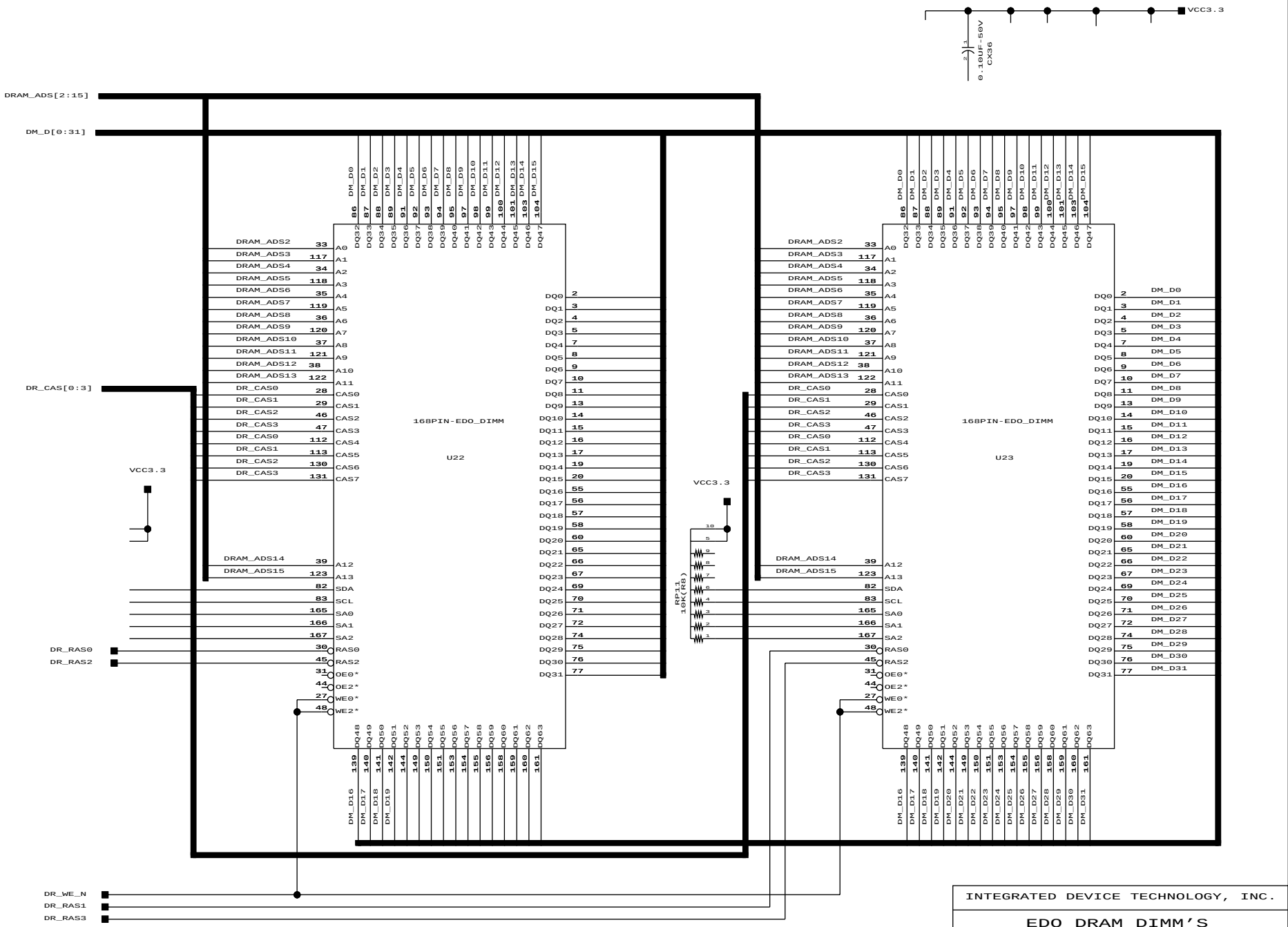
| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| DATA BUFFERS & DAMPING RESISTORS | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:23 | SHEET 4 OF 18 | |

| INIT*3 | INT*2 | INT*1 | INT*0 | BUS GNT* | PCST4 | PCST3 | PCST2 | PCST1 | PCST0 |
|---|---|---|---|---|---|---|---|---|---|
| BOOTPROM WIDTH<br>00 8 BITS<br>01 16 BITS<br>10 32 BITS<br>11 RESERVED | | RESERVED<br>1 | Timer Int En<br>0 Enable<br>1 Disable | RESERVED<br>0 | RESERVED<br>0 | END BIT<br>0 LITTLE<br>1 BIG | CLOCK MULTIPLIER MASTER CLOCK<br>TO GENERATE PCLOCK<br>000 MULTIPLY BY 2 100 MULTIPLY BY 6<br>001 MULTIPLY BY 3 101 MULTIPLY BY 7<br>010 MULTIPLY BY 4 110 MULTIPLY BY 8<br>011 MULTIPLY BY 5 111 RESERVED | | |

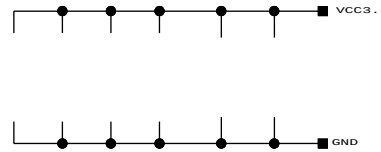
MODE BITS

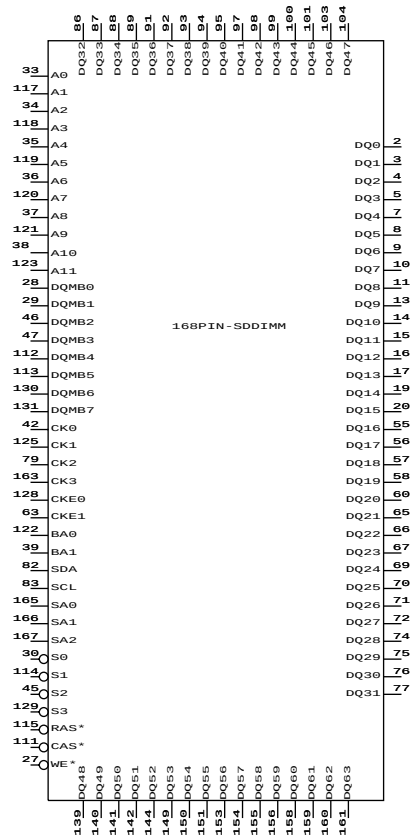


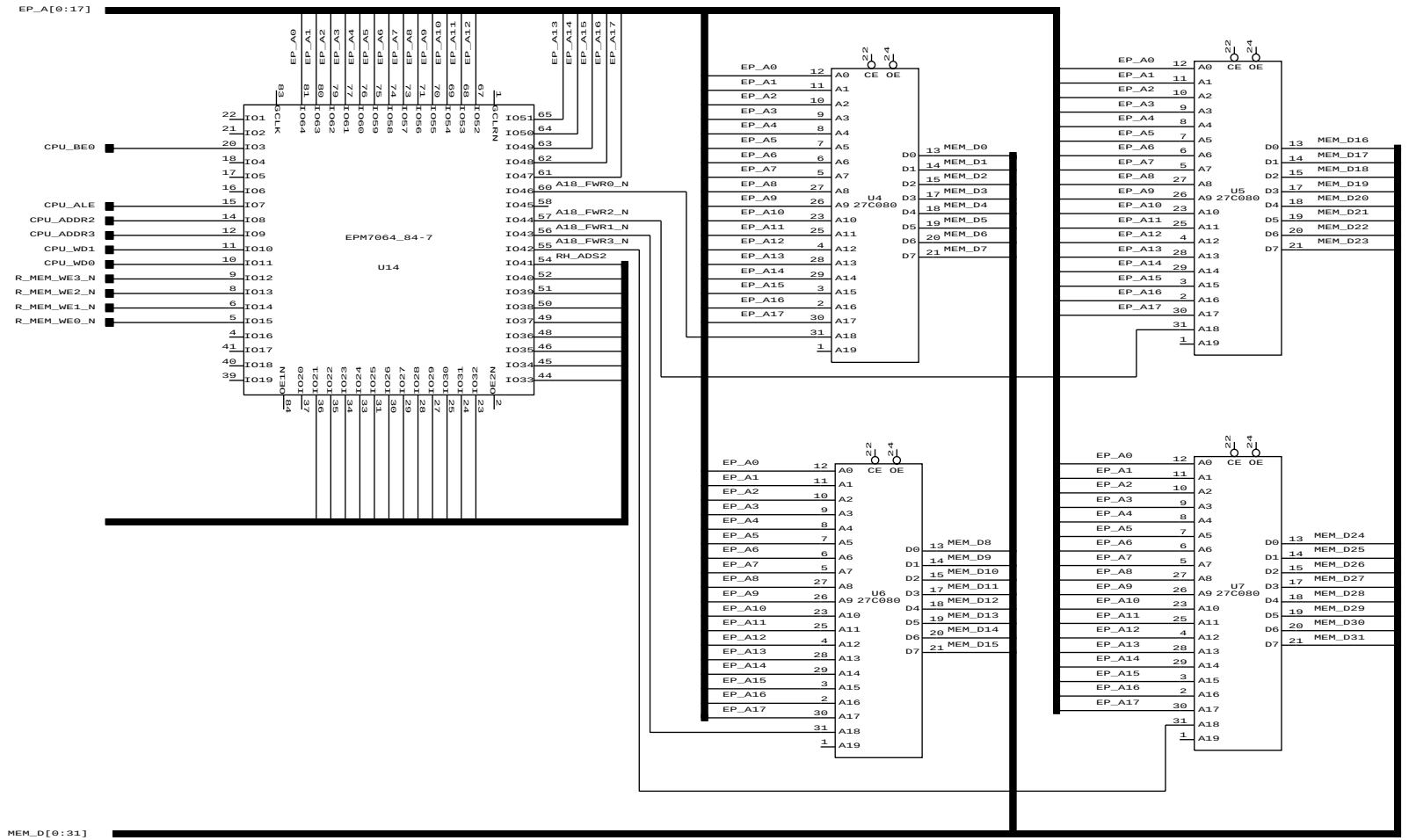

| | | | |
|---|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | | |
| MODE SELECTION LOGIC | | | |
| SIZE<br>CR | DWG NO.<br>CRONUS_RHEA | | REV<br>R0 |
| DATE: 3-22-1999_16:19 | | SHEET 5 OF 18 | |



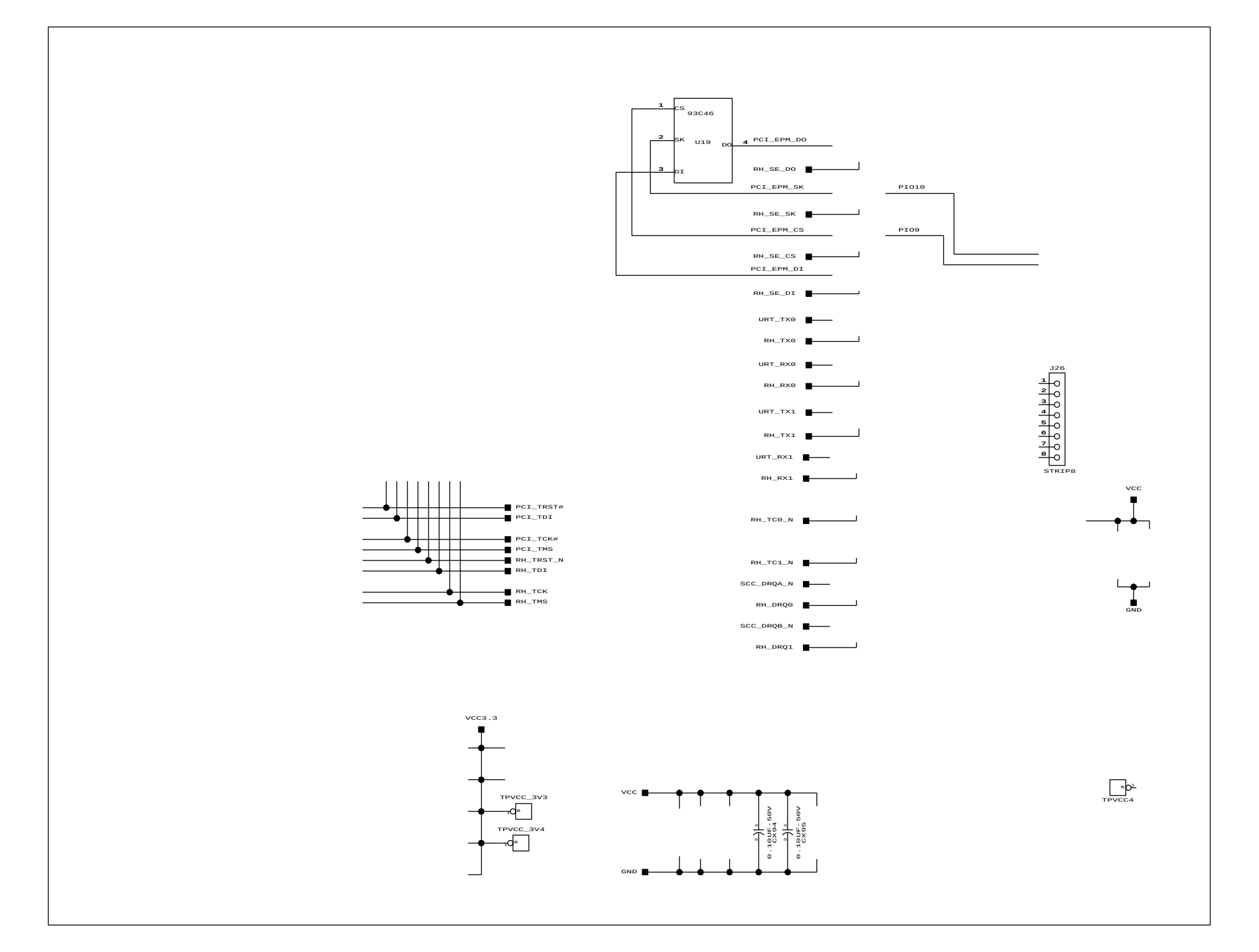

| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| EDO DRAM DIMM'S | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:23 | SHEET 6 OF 18 | |

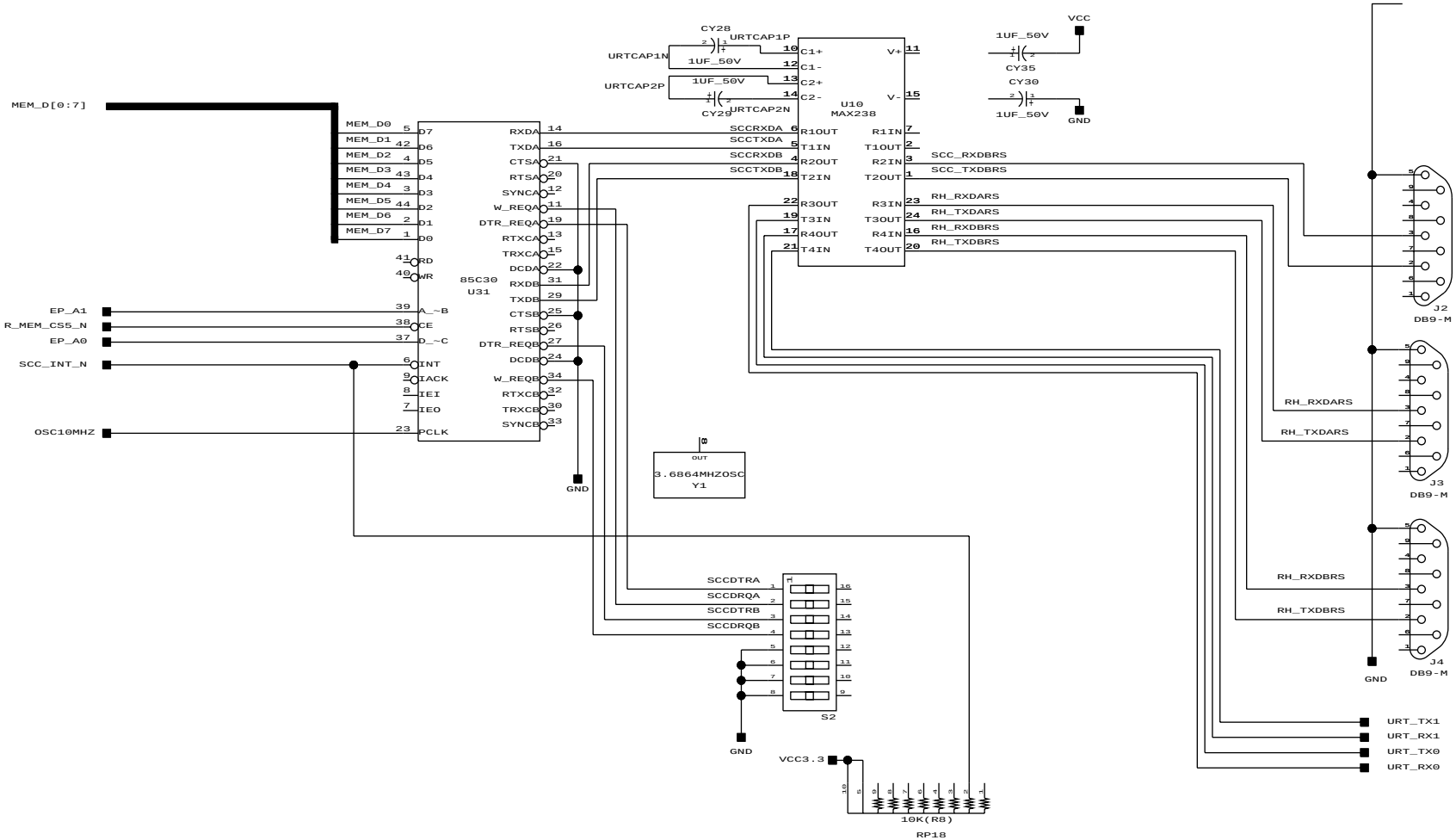

VCC3.3
GND


168PIN-SDDIMM
A0 33
A1 117
A2 34
A3 118
A4 35
A5 119
A6 36
A7 120
A8 37
A9 121
A10 38
A11 123
DQMB0 28
DQMB1 29
DQMB2 46
DQMB3 47
DQMB4 112
DQMB5 113
DQMB6 130
DQMB7 131
CK0 42
CK1 125
CK2 79
CK3 163
CKE0 128
CKE1 63
BA0 122
BA1 39
SDA 82
SCL 83
SA0 165
SA1 166
SA2 167
S0 30
S1 114
S2 45
S3 129
RAS* 115
CAS* 111
WE* 27
DQ0 2
DQ1 3
DQ2 4
DQ3 5
DQ4 7
DQ5 8
DQ6 9
DQ7 10
DQ8 11
DQ9 13
DQ10 14
DQ11 15
DQ12 16
DQ13 17
DQ14 19
DQ15 20
DQ16 55
DQ17 56
DQ18 57
DQ19 58
DQ20 60
DQ21 65
DQ22 66
DQ23 67
DQ24 69
DQ25 70
DQ26 71
DQ27 72
DQ28 74
DQ29 75
DQ30 76
DQ31 77
DQ32 86
DQ33 87
DQ34 88
DQ35 89
DQ36 91
DQ37 92
DQ38 93
DQ39 94
DQ40 95
DQ41 97
DQ42 98
DQ43 99
DQ44 100
DQ45 101
DQ46 103
DQ47 104
DQ48 139
DQ49 140
DQ50 141
DQ51 142
DQ52 144
DQ53 149
DQ54 150
DQ55 151
DQ56 153
DQ57 154
DQ58 155
DQ59 156
DQ60 158
DQ61 159
DQ62 160
DQ63 161

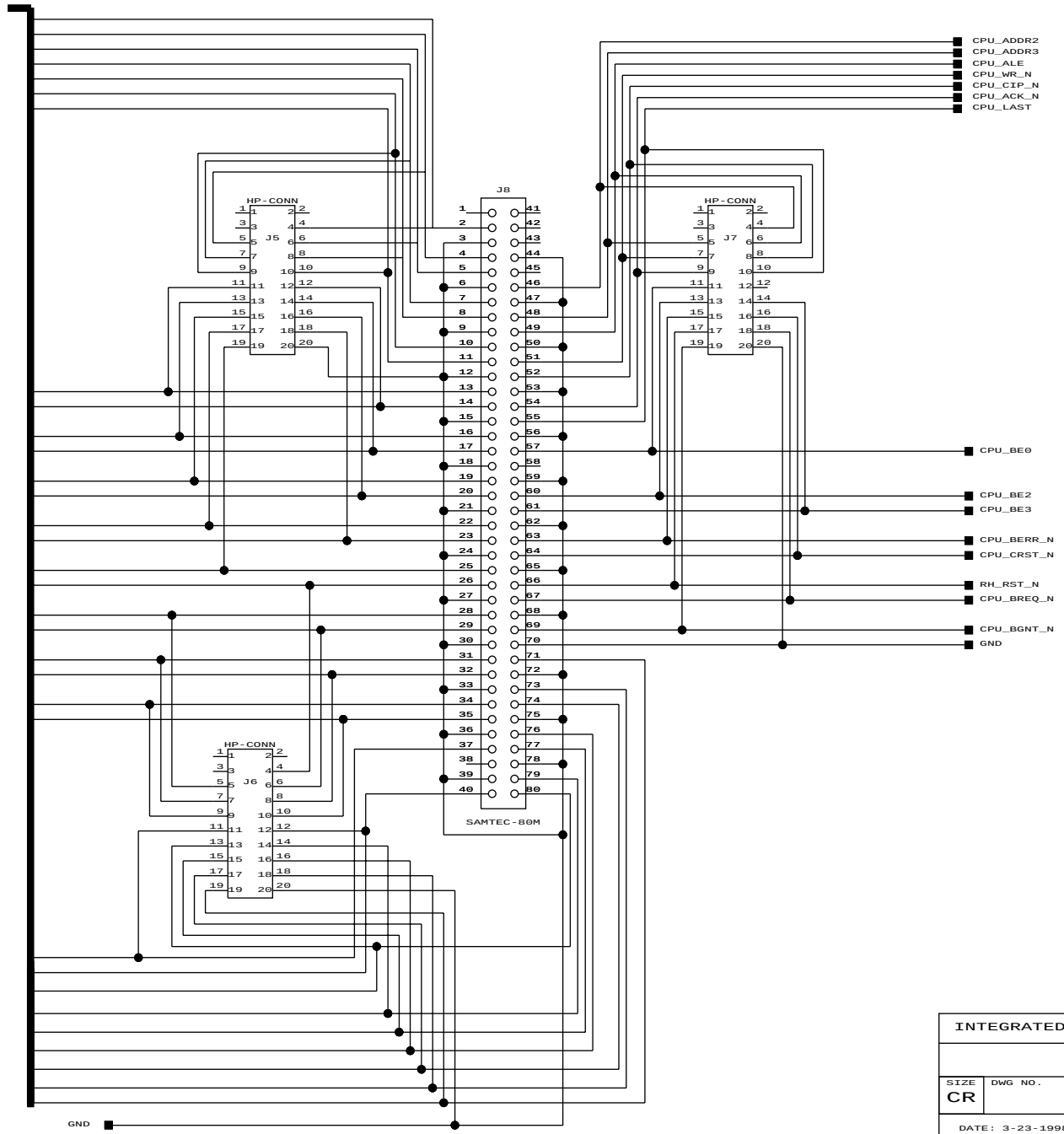



| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| EPROM / FLASH LOGIC | | |
| SIZE CR | DWG NO. CRONUS-RHEA | REV R0 |
| DATE: 11-23-1998_14:05 | SHEET 8 OF 18 | |

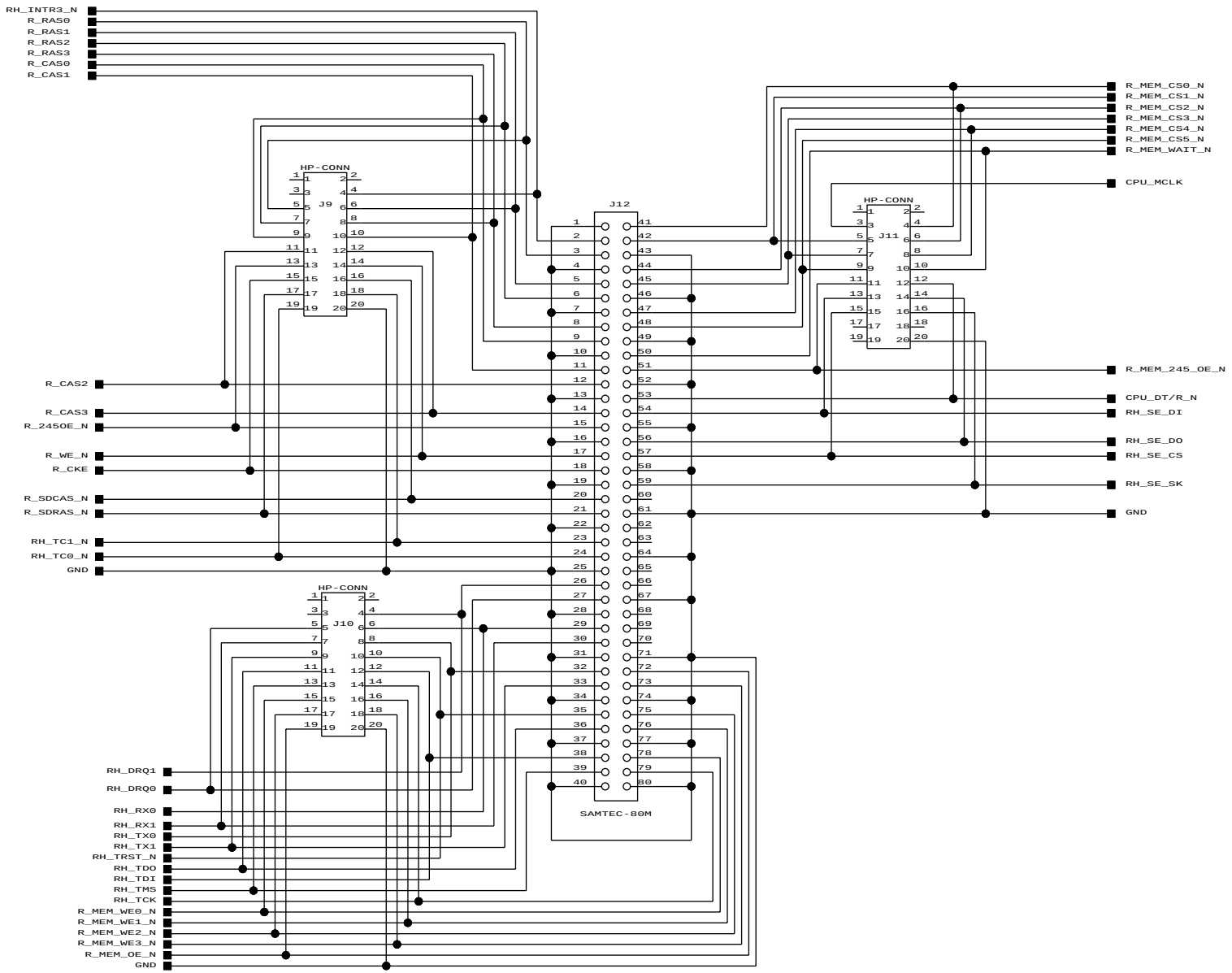

1 CS
93C46
2 SK
U19
DO 4
3 DI
PCI_EPM_DO
RH_SE_DO
PCI_EPM_SK
PIO10
RH_SE_SK
PCI_EPM_CS
PIO9
RH_SE_CS
PCI_EPM_DI
RH_SE_DI
URT_TX0
RH_TX0
URT_RX0
RH_RX0
URT_TX1
RH_TX1
URT_RX1
RH_RX1
RH_TC0_N
RH_TC1_N
SCC_DRQA_N
RH_DRQ0
SCC_DRQB_N
RH_DRQ1
J26
1
2
3
4
5
6
7
8
STRIP8
VCC
GND
PCI_TRST#
PCI_TDI
PCI_TCK#
PCI_TMS
RH_TRST_N
RH_TDI
RH_TCK
RH_TMS
VCC3.3
TPVCC_3V3
TPVCC_3V4
VCC
GND
0.10UF_50V
0.10UF_50V
TPVCC4



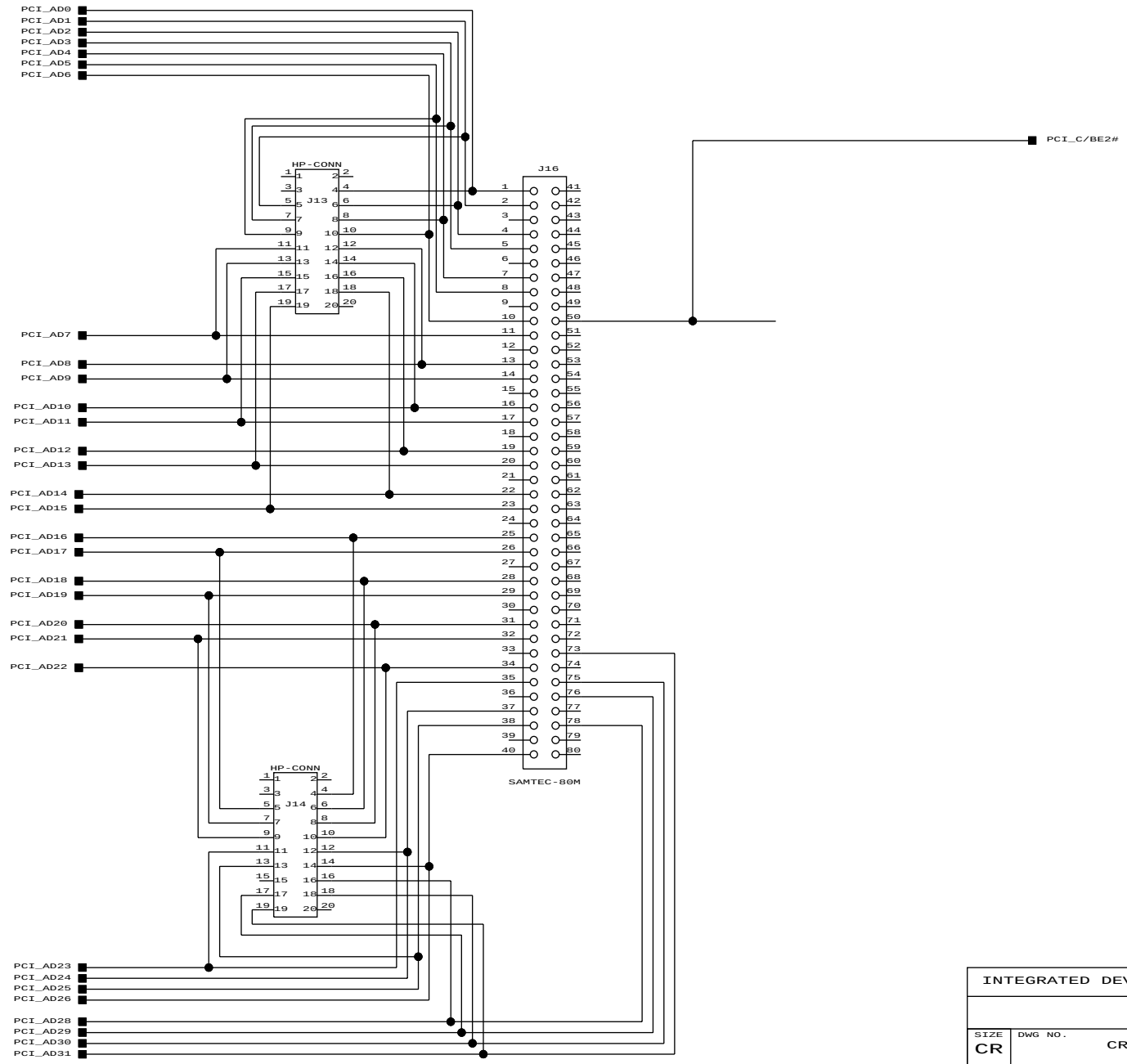

| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| SERIAL COMM. CONTROLLER | | |
| SIZE: CR | DWG NO.: CRONUS-RHEA | REV: R0 |
| DATE: 3-23-1998_14:24 | SHEET 10 OF 18 | |



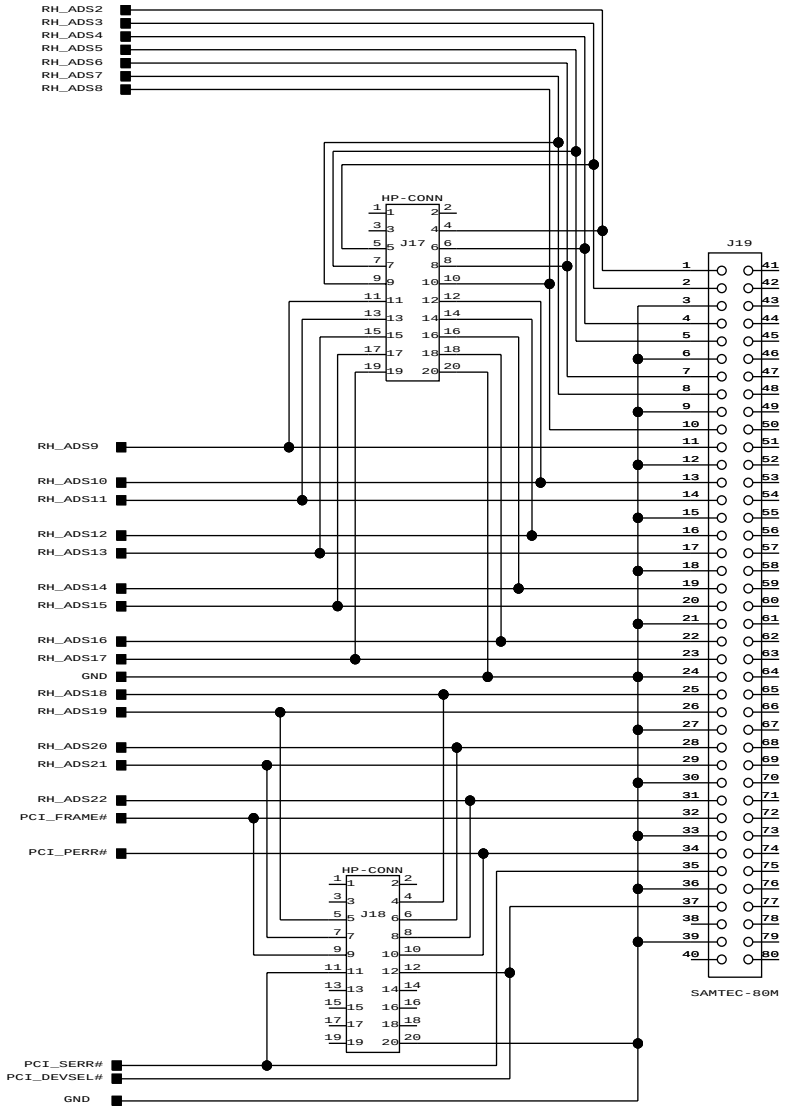

| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| RHEA_POD1 | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:25 | | SHEET 11 OF 18 |



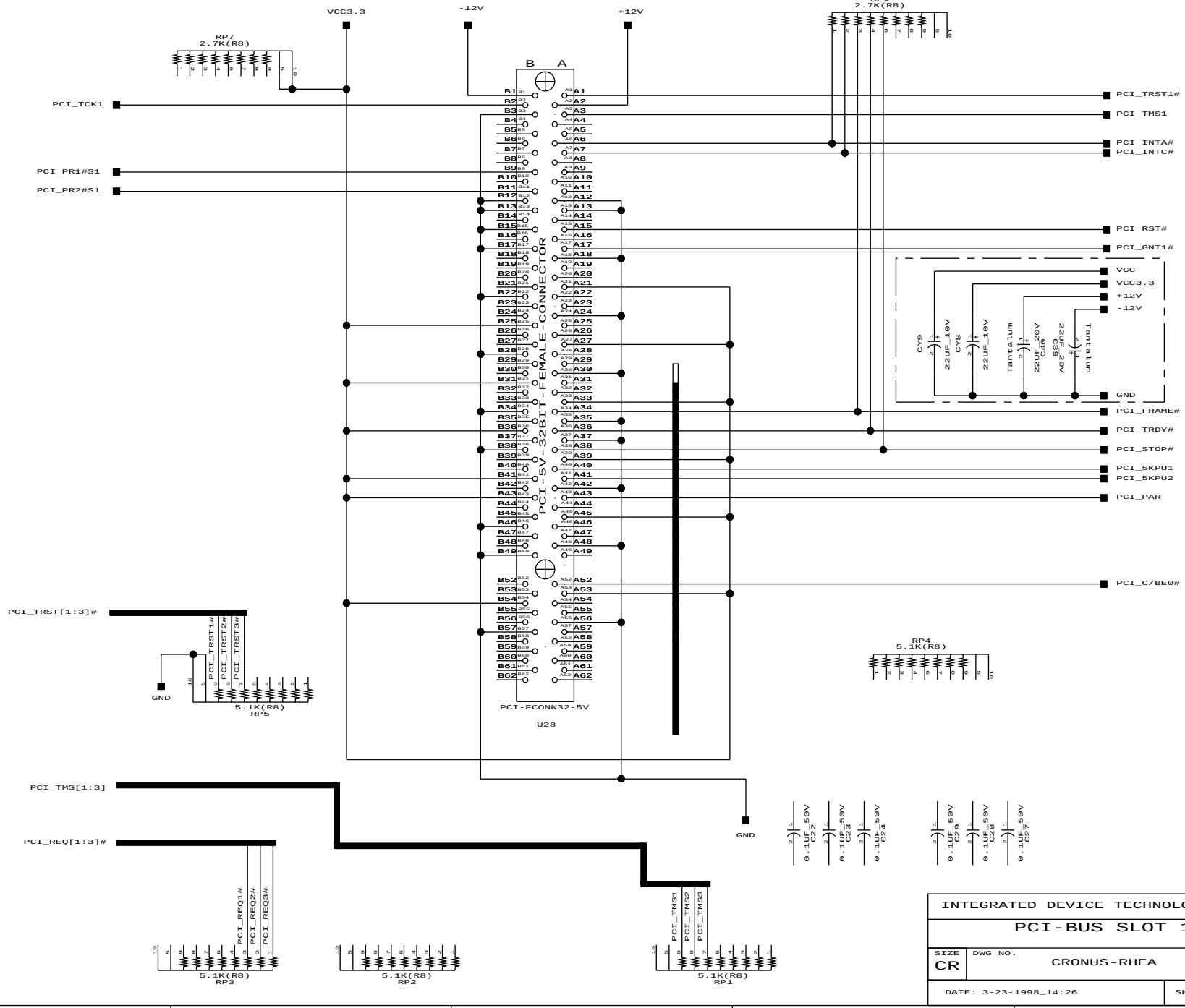

| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| RHEA_POD2 | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:25 | | SHEET 12 OF 18 |



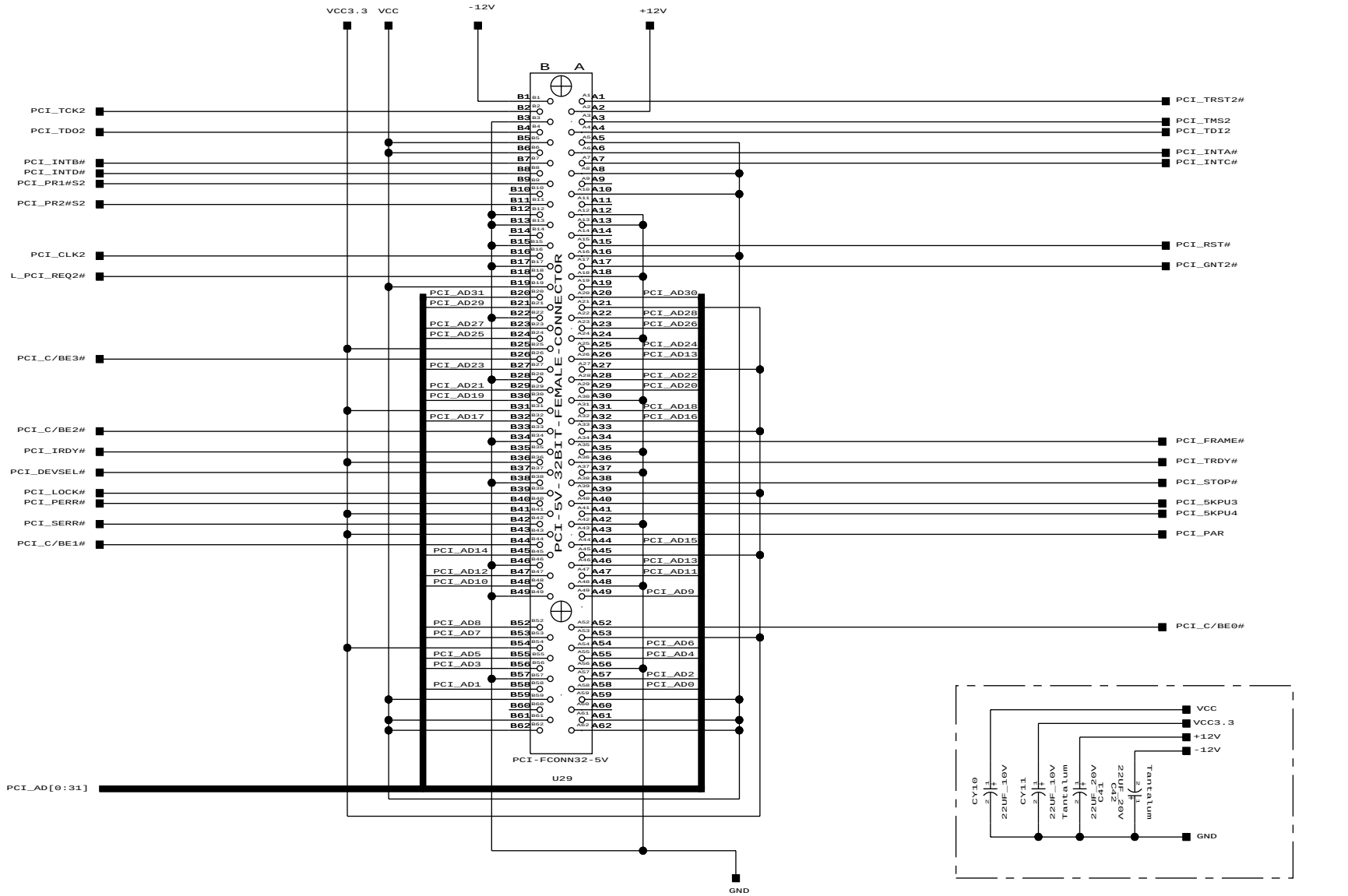

| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| RHEA_POD3 | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:25 | SHEET 13 OF 18 | |



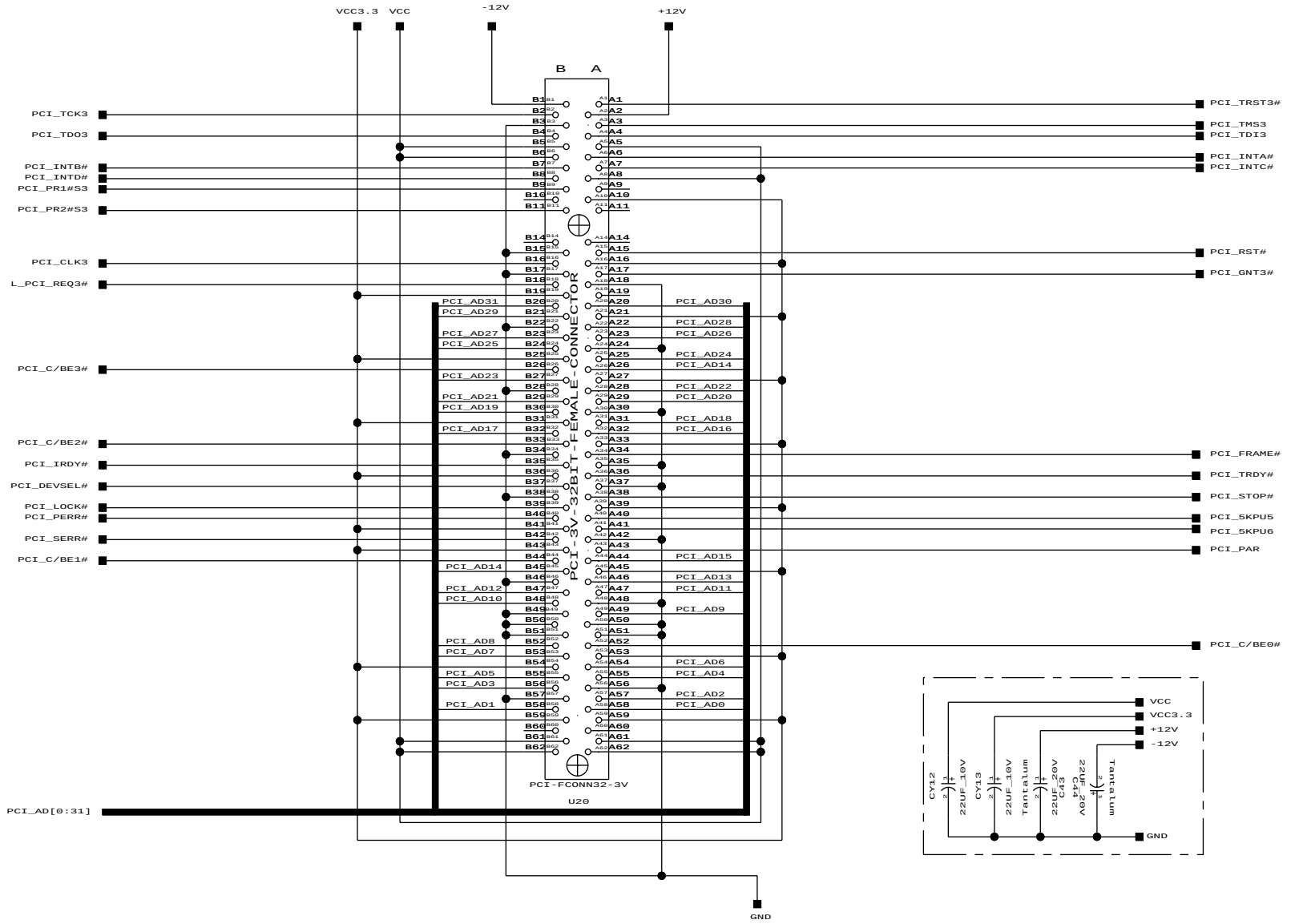

| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| RHEA_POD4 | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:26 | | SHEET 14 OF 18 |



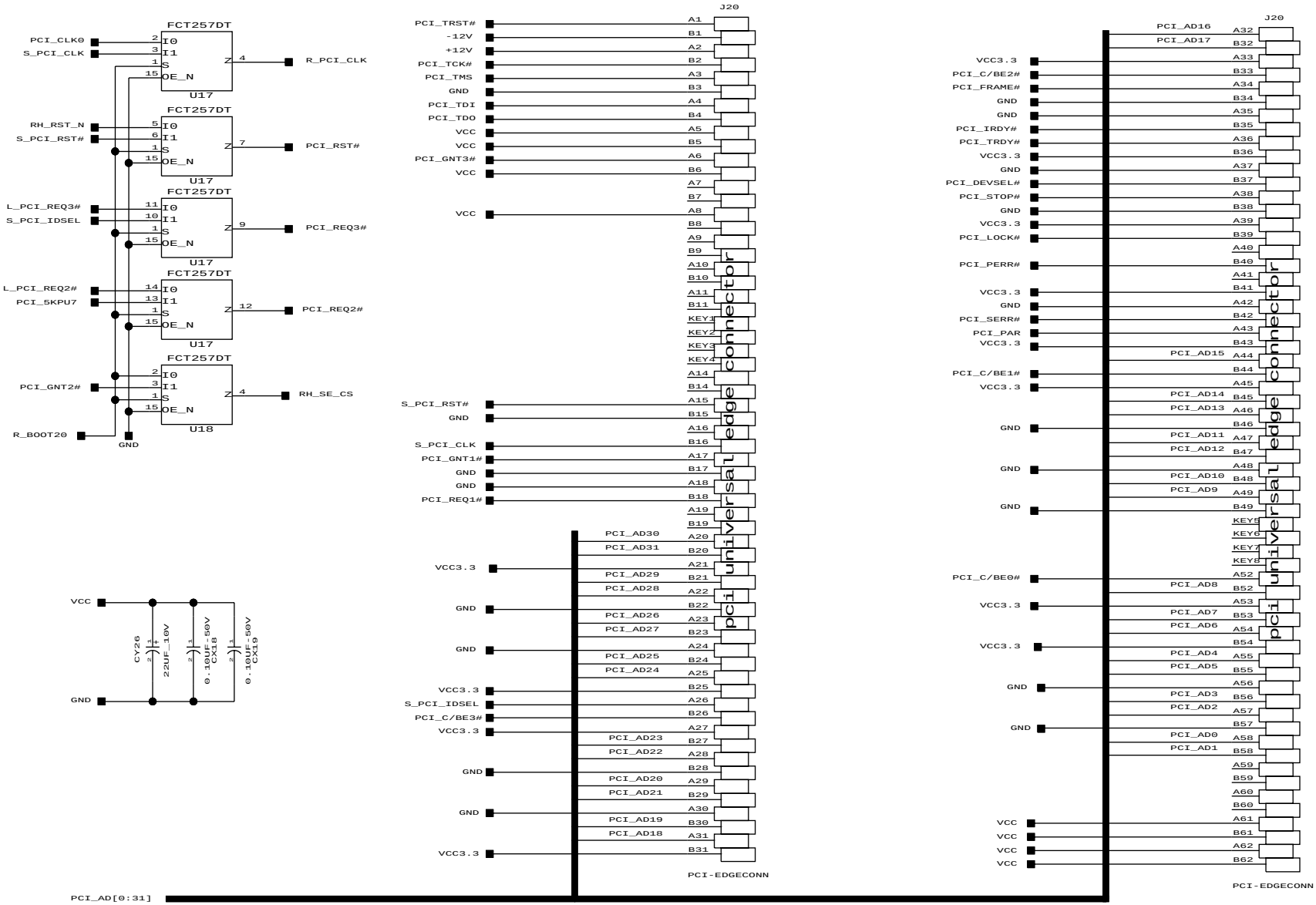

| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| PCI-BUS SLOT 1 | | |
| SIZE CR | DWG NO. CRONUS-RHEA | REV R0 |
| DATE: 3-23-1998_14:26 | | SHEET 15 OF 18 |




| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| PCI-BUS SLOT 2 | | |
| SIZE CR | DWG NO. CRONUS-RHEA | REV R0 |
| DATE: 3-23-1998_14:26 | | SHEET 16 OF 18 |




| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| PCI-BUS SLOT 3 FOR 3.3V | | |
| SIZE CR | DWG NO. CRONUS-RHEA | REV R0 |
| DATE: 3-23-1998_14:27 | SHEET 17 OF 18 | |




| | | |
|---|---|---|
| INTEGRATED DEVICE TECHNOLOGY, INC. | | |
| PCI SATELLITE CONNECTOR | | |
| SIZE CR | DWG NO. CRONUS_RHEA | REV R0 |
| DATE: 3-23-1998_14:27 | | SHEET 18 OF 18 |

# EPLD Equation

## Notes

```
(
--RESET inputs
EPLD_clk: INPUT;
RSTinitN: INPUT;
RHrstN: INPUT;
MODEin[9..0]: INPUT;
Rboot22: INPUT;
Rboot21: INPUT;
Rboot20: INPUT;
Rboot19: INPUT;
--RESET outputs
CPUcrstN: OUTPUT;
CPUrstN: OUTPUT;
CPUintN[5..0]: OUTPUT;
CPUnmiN: OUTPUT;
CPUbgntN: OUTPUT;
CPUpcst[4..0]: OUTPUT;
RHads22: OUTPUT;
RHads21: OUTPUT;
RHads20: OUTPUT;
RHads19: OUTPUT;

--      INTERRUPT inputs
SCCintN: INPUT;
RHintr3N: INPUT;
PCIintAN: INPUT;
PCIintBN: INPUT;
PCIintCN: INPUT;
PCIintDN: INPUT;


--SRAM control inputs
Rmemcs1N: INPUT;
RmemweN[3..0]: INPUT;
RmemoeN: INPUT;
```

```
--SRAM control outputs
SRAMcsN[3..0]: OUTPUT;
SRAMweN: OUTPUT;

--BUS CYCLE CONTROL OUTPUT
RmemwaitN: OUTPUT;

--SOFTWARE PORT
MEMd[7..0]: BIDIR;
Rmemcs3N: INPUT;

--  FUTURE USE INPUTS
Rmemcs0N: INPUT;-- EPROM chip select

Rmemcs2N: INPUT;-- RESERVED chip select

Rmemcs4N: INPUT;-- RESERVED chip select

Rmemcs5N: INPUT;-- SCC chip select



)

--  Variable Section (optional)
VARIABLE

--  Node Declarations
count_rst_dly: NODE;
SOFTrst: NODE;
RmemweNX: NODE;
RmemcsXN: NODE;
PCST[4..0]: TRI;
CPUbgntNtri: TRI;
RHads22tri: TRI;
RHads21tri: TRI;
RHads20tri: TRI;
RHads19tri: TRI;
SWport[7..0]: TRI;
```

```
-- Register Declaration
   CLEAR_RST_DLY      : DFF;
CPUcrstN: DFF;
CPUrstN: DFF;
RmemwaitN: DFF;
rst_dly[6..0]: DFFE;
SOFTport[7..0]: DFFE;

-- State Machine Declaration

-- Reset state_machine

reset_sm : MACHINE
OF BITS (RST1,RST0)
WITH STATES(
RST_IDLE= 0,
RST_CRST= 1,
RST_RST= 2);
-- WAIT state_machine

wait_sm : MACHINE
OF BITS (WST2,WST1,WST0)
WITH STATES(
WAIT_IDLE= 0,
WAIT1= 1,
WAIT2= 2,
WAIT3= 3,
WAIT4= 4,
WAIT5= 5,
WAIT6= 6,
WAIT7= 7);

-- Logic Section

BEGIN

-- Defaults Statement
```

```
DEFAULTS
CPUcrstN= VCC;
CPUrstN= VCC;
RmemwaitN= VCC;
rst_dly[6..0]= GND;
SOFTport[7..0]= GND;
   CLEAR_RST_DLY   = VCC;
END DEFAULTS;
   CLEAR_RST_DLY.clk = GLOBAL(EPLD_clk);
   CLEAR_RST_DLY.clrn = RSTinitN;

CPUcrstN.clk= GLOBAL(EPLD_clk);
CPUcrstN.clrn= RSTinitN;

CPUrstN.clk= GLOBAL(EPLD_clk);
CPUrstN.clrn= RSTinitN;

RmemwaitN.clk= GLOBAL(EPLD_clk);
RmemwaitN.clrn= RSTinitN;

rst_dly[].clk= GLOBAL(EPLD_clk);
rst_dly[].clrn= CLEAR_RST_DLY;
rst_dly[].ena= count_rst_dly;


--  State Machine Defaults

reset_sm.clk= GLOBAL(EPLD_clk);
reset_sm.reset= !RSTinitN;

wait_sm.clk= GLOBAL(EPLD_CLK);
wait_sm.reset= CPUrstN;

--RESET LOGIC

rst_dly[]= rst_dly[] + 1;


--MODE INITIALIZATION
```

```
PCST[].oe= !CPUrstN;

PCST[0].in= MODEin[0];
CPUpcst[0]= PCST[0].out;

PCST[1].in= MODEin[1];
CPUpcst[1]= PCST[1].out;

PCST[2].in= MODEin[2];
CPUpcst[2]= PCST[2].out;

PCST[3].in= MODEin[3];
CPUpcst[3]= PCST[3].out;

PCST[4].in= MODEin[4];
CPUpcst[4]= PCST[4].out;

CPUbgntNtri.oe= !CPUrstN;
CPUbgntNtri.in= MODEin[5];
CPUbgntN= CPUbgntNtri.out;

RHads22tri.oe= !CPUrstN;
RHads22tri.in= Rboot22;
RHads22= RHads22tri.out;

RHads21tri.oe= !CPUrstN;
RHads21tri.in= Rboot21;
RHads21= RHads21tri.out;

RHads20tri.oe= !CPUrstN;
RHads20tri.in= Rboot20;
RHads20= RHads20tri.out;

RHads19tri.oe= !CPUrstN;
RHads19tri.in= Rboot19;
RHads19= RHads19tri.out;

--INTERRUPT LOGIC
```

```
CPUintN[0]=!(!CPUrstN & !MODEin[6]
  # CPUrstN & !SCCintN );

CPUintN[1]=!(!CPUrstN & !MODEin[7]
  # CPUrstN & !PCIintaN);

CPUintN[2]=!(!CPUrstN & !MODEin[8]
  # CPUrstN & !PCIintbN );

CPUintN[3]=!(!CPUrstN & !MODEin[9]
 # CPUrstN & !RHintr3N );

CPUintN[4]=!( CPUrstN & !PCIintcN );

CPUintN[5]=!( CPUrstN & !PCIintdN );

CPUnmiN=VCC;

--SRAM CONTROL LOGIC
RmemweNX= ( RmemweN0 & RmemweN1 & RmemweN2 & RmemweN3 );

SRAMcsN0= !(!Rmemcs1N & !RmemweN0
   #!Rmemcs1N & !RmemoeN );

SRAMcsN1= !(!Rmemcs1N & !RmemweN1
   #!Rmemcs1N & !RmemoeN );

SRAMcsN2= !(!Rmemcs1N & !RmemweN2
   #!Rmemcs1N & !RmemoeN );

SRAMcsN3= !(!Rmemcs1N & !RmemweN3
   #!Rmemcs1N & !RmemoeN );

SRAMweN= !(!Rmemcs1N & !RmemweNX );

--BUS CYCLE CONTROL LOGIC

RmemcsXN= ( Rmemcs0N & Rmemcs1N & Rmemcs2N & Rmemcs3N & Rmemcs4N & Rmemcs5N );
```

```
CASE wait_sm IS

WHEN WAIT_IDLE =>
RmemwaitN= VCC;
wait_sm= WAIT1;

WHEN WAIT1=>
IF (RmemcsXN) THEN
RmemwaitN= GND;
wait_sm= WAIT2;
ELSE
RmemwaitN= VCC;
wait_sm= WAIT_IDLE;
END IF;
WHEN WAIT2=>
IF (RmemcsXN) THEN
RmemwaitN= GND;
wait_sm= WAIT3;
ELSE
RmemwaitN= VCC;
wait_sm= WAIT_IDLE;
END IF;

WHEN WAIT3=>
IF (RmemcsXN) THEN
RmemwaitN= GND;
wait_sm= WAIT4;
ELSE
RmemwaitN= VCC;
wait_sm= WAIT_IDLE;
END IF;

WHEN WAIT4=>
IF (RmemcsXN) THEN
RmemwaitN= GND;
wait_sm= WAIT5;
ELSE
```

```
RmemwaitN= VCC;
wait_sm= WAIT_IDLE;
END IF;

WHEN WAIT5=>
IF (RmemcsXN) THEN
RmemwaitN= GND;
wait_sm= WAIT6;
ELSE
RmemwaitN= VCC;
wait_sm= WAIT_IDLE;
END IF;

WHEN WAIT6=>
IF (RmemcsXN) THEN
RmemwaitN= GND;
wait_sm= WAIT7;
ELSE
RmemwaitN= VCC;
wait_sm= WAIT_IDLE;
END IF;

WHEN WAIT7=>
IF (RmemcsXN) THEN
RmemwaitN= VCC;
wait_sm= WAIT_IDLE;
ELSE
RmemwaitN= VCC;
wait_sm= WAIT_IDLE;
END IF;




END CASE;

RmemwaitN= VCC;

--S/W PROGRAMMABLE PORT LOGIC
```

```
SWport[].oe= (!Rmemcs3N & !RmemoeN );
SWport[].in   = SOFTport[];
MEMd[]= SWport[].out;
SOFTport[].d= MEMd[];
SOFTport[].clrn= CPUrstN;
SOFTport[].clk= GLOBAL(EPLD_clk);
SOFTport[].ena= !(!Rmemcs3N & !RmemweN0 );
SOFTrst= SOFTport[0];

--RESET LOGIC

CASE reset_sm IS

WHEN RST_IDLE =>
CPUrstN= GND;
reset_sm= RST_CRST;

WHEN RST_CRST =>
count_rst_dly= VCC;
IF (rst_dly[6]) THEN
        CPUrstN     = GND;
reset_sm= RST_RST;
ELSE
CPUrstN= GND;
reset_sm= RST_CRST;
END IF;

WHEN RST_RST  =>
        CLEAR_RST_DLY = GND;
--IF (SOFTrst # !RHrstN) THEN
        IF (!RHrstN) THEN
CPUrstN= GND;
reset_sm= RST_IDLE;
ELSE
reset_sm= RST_RST;
END IF;

END CASE;
```

```
END;

(
%
Address gen. for 8/16/32 bit devices and
Flash write generation.
%

EPLD_clk: INPUT;
CPU_rstN: INPUT;
Rmem_cs0_N: INPUT;
Rmem_cs5_N: INPUT;

FlashN: INPUT;
CPUaddrs2: INPUT;
CPUaddrs3: INPUT;
RHads[22..2]: INPUT;
CPU_beN[3..0]: INPUT;
CPU_aleN: INPUT;
CPU_wdN[1..0]: INPUT;
RmemWrN[3..0]: INPUT;
   RmemoeN: INPUT;

EP_a[17..0]: OUTPUT;
A18_FwrN[3..0]: OUTPUT;
A18_A19: OUTPUT;

--  ~10MHz Clock generation.

OSC10Mhz: OUTPUT;
Brh_clk: OUTPUT;

-- SCC control signals

SCC_WR_N: OUTPUT;
SCC_RD_N: OUTPUT;

)
```

```
-- Variable Section (optional)
VARIABLE

-- Node Declaration
count_dly: NODE;




-- Register Declaration

Counter[4..0]: DFF;
OSC10Mhz: DFF;



-- State Machine Declaration









-- Logic Section

BEGIN

-- Defaults Statement

DEFAULTS
Counter[4..0]= GND;
OSC10Mhz= GND;

-- State Machine Defaults

--

END DEFAULTS;
```

```
Counter[4..0].clk= GLOBAL(EPLD_clk);
Counter[4..0].clrn= CPU_rstN;

OSC10Mhz.clk= Counter[4];
OSC10Mhz.clrn= CPU_rstN;



--ADDRESS LOGIC

IF CPU_wdN[1..0] == 0 THEN
EP_a[0] = CPU_beN[0];
EP_a[1] = CPU_beN[1];
EP_a[2] = CPUaddrs2;
EP_a[3] = CPUaddrs3;
EP_a[17..4] = RHads[17..4];
IF FlashN THEN
A18_FwrN[3..0] = RHads[18];
A18_A19= RHads[19];

ELSE
A18_FwrN[3]= RmemWrN[3];
A18_FwrN[2]= RmemWrN[2];
A18_FwrN[1]= RmemWrN[1];
A18_FwrN[0]= RmemWrN[0];
A18_A19= Rhads[18];
END IF;

ELSIF CPU_wdN[1..0] == 1 THEN
EP_a[0] = CPU_beN[1];
EP_a[1] = CPUaddrs2;
EP_a[2] = CPUaddrs3;
EP_a[17..3] = RHads[18..4];
IF FlashN THEN
A18_FwrN[3..0]= RHads[19];
A18_A19= RHads[20];

ELSE
A18_FwrN[3]= RmemWrN[3];
```

```
A18_FwrN[2]= RmemWrN[2];
A18_FwrN[1]= RmemWrN[1];
A18_FwrN[0]= RmemWrN[0];
A18_A19= Rhads[19];
END IF;

ELSIF CPU_wdN[1..0] == 2 THEN
EP_a[0] = CPUaddrs2;
EP_a[1] = CPUaddrs3;
EP_a[17..2] = RHads[19..4];
IF FlashN THEN
A18_FwrN[3..0] = RHads[20];
A18_A19= RHads[21];

ELSE
A18_FwrN[3]= RmemWrN[3];
A18_FwrN[2]= RmemWrN[2];
A18_FwrN[1]= RmemWrN[1];
A18_FwrN[0]= RmemWrN[0];
A18_A19= Rhads[20];
END IF;

END IF;

--OSCILLATOR LOGIC

OSC10Mhz.d= !OSC10Mhz.q;
IF Counter[].q == 16 THEN
Counter[].d = 0;
ELSE
Counter[].d = Counter[].q + 1;
END IF;

Brh_clk= GLOBAL(EPLD_clk);

-- SCC LOGIC

SCC_WR_N= CPU_rstN & ( RmemWrN0 # Rmem_cs5_N );
SCC_RD_N= CPU_rstN & ( RmemoeN # Rmem_cs5_N );
```

END;