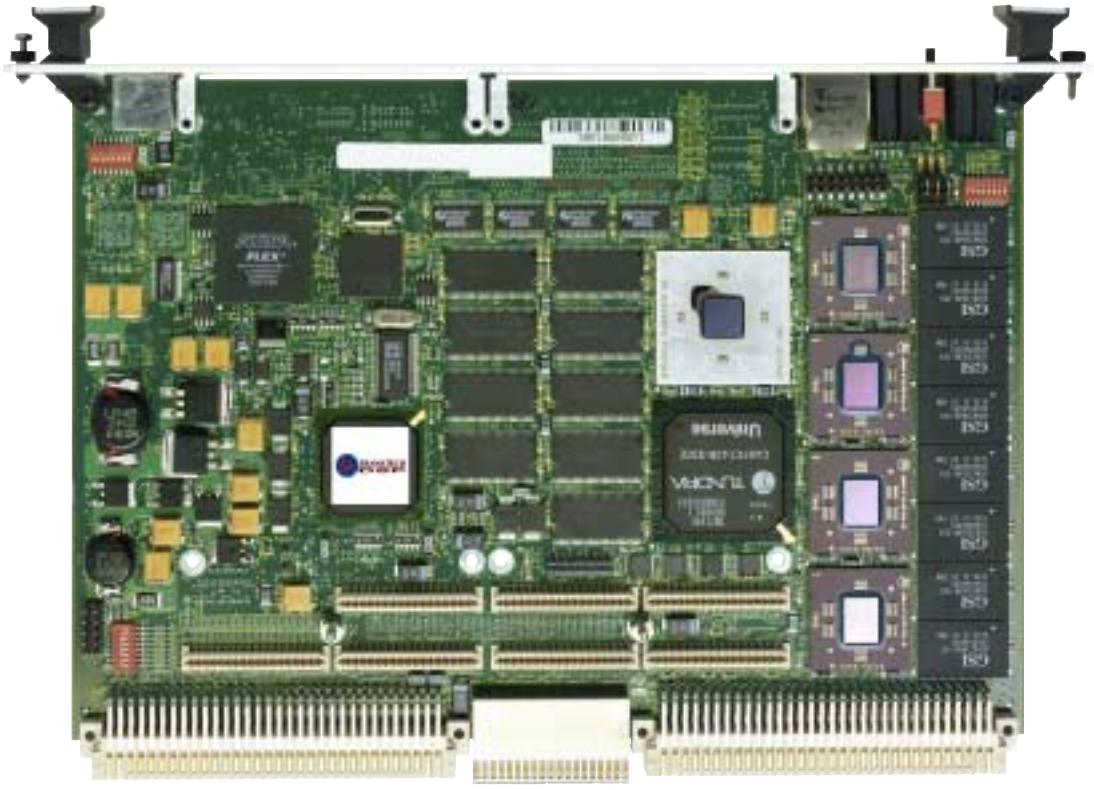
VQG4

Quad PowerPC 7400 with RACE++™ and Dual PMC



Features

1, 2 or 4 PowerPC 7400 CPUs

1 or 2Mbyte L2 cache per CPU

Up to 512Mbytes SDRAM

2 PMC sites (separate PCI buses)

RACE++™ interface

PCI over VME P0

Ethernet (10/100baseT)

32Mbytes Flash

Built-In Test (BIT)

VxWorks™

Linux

Optimized VSIPL DSP libraries

The VQG4 can be configured as a single, dual and quad PowerPC processor based single board computer card and provides high processor density yet retaining traditional I/O facilities such as Ethernet and serial I/O.
A RACE++ interface allows the processor array efficient access to high bandwidth data. This allows the VQG4 to be scaled to include extra processors and is ideal for applications such as radar, sonar and high-speed imaging.

Industry standard operating systems supported and include VxWorks and linux. This is backed up with PowerPC optimized DSP libraries and high-level system design tools.

TRANSTECH™ DSP

www.transtech-dsp.com

Hardware

Overview

The VQG4 is provided with a choice of 1, 2 or 4 PowerPC 7400 CPUs and at different speed grades. The VQG4 will accept future higher speed parts as they become available.

To achieve peak performance, the VQG4 includes 2Mbyte L2 cache per CPU. The L2 cache is connected to the CPU by a dedicated bus running at between 150 and 200MHz, depending on the processor core speed.

The CPUs are connected to the SDRAM by an IBM CPC-710+ bridge for low latency and high bandwidth data streaming.

The VQG4 includes all the necessary resources required by leading real-time operating systems such as a flash BIOS ROM for boot firmware, non-volatile memory, a timer and a serial port. The serial port is available through a 9-way D-type connector, and provides a resource for operating systems to implement diagnostics or configuration support.

"Super-computer like performance on a chip"

PowerPC 7400 Summary Specification

- **Internal CPU Speeds**
  400, 450MHz and 500MHz
- **Bus Interface**
  64-bit bus with MPX/60x protocol
- **Cache**
  L1: 32Kbyte instr/32Kbyte data
  L2: up to 2Mbytes
- **Benchmark (450MHz)**
  21.4 SPECint95
  20.4 SPECfp95
  825 MIPS
- **Execution Units**
  integer (2), floating-point, vector, branch, load/store, system

PowerPC 7400 with AltiVec™

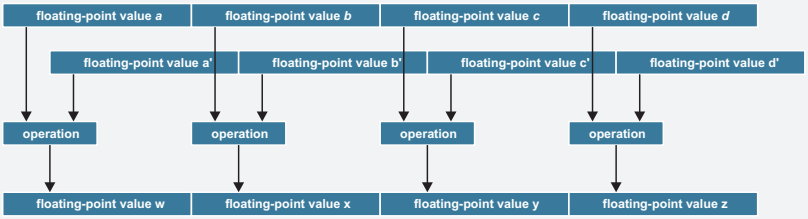
A major of the PowerPC 7400 is a 128-bit vector processing unit. This is the AltiVec extension and operates in a SIMD mode (Single Instruction Multi-Data). With AltiVec, the PowerPC can accelerate many DSP and multimedia type applications by being able to achieve up to 20 operations in a single CPU clock cycle. As an example, the vector unit can process 16x 8-bit data elements in parallel. The vector unit can handle parallel data elements of other lengths including floating-point.

With the vector unit come 162 new instructions. These are similar to those found on the scalar units, but in SIMD format. To implement these instructions at a high level, C language extensions have been made available. Using language extensions, rather than relying on compiler efficiencies, permits maximum performance to be achievable.

Through the Altivec Instruction Set Architecture many application segments are supported including Voice over IP (VoIP), speech recognition, voice/sound processing and communications including multi-channel modems, software modems, data encryption. The parallel nature of Altivec means that PowerPC 7400 can replace modem banks with a single processor whereas many were previously needed. This simplicity makes development easier and reduces cost.

# Parallel Vector Operations	Data Types Supported
16-way 8-way 4-way	8-bit signed & unsigned ints and chars 16-bit signed & unsigned ints 32-bit signed & unsigned ints and IEEE floating-point numbers

Parallelism with AltiVec technology execution unit
Table 1

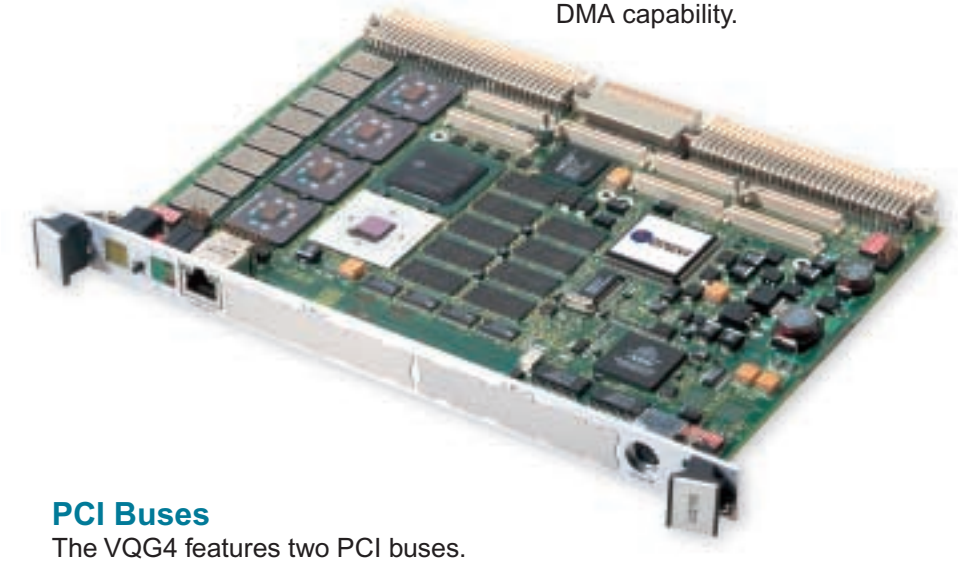


Parallel floating-point operations with AltiVec
Figure 2

VQG4 Benefits

- **PowerPC 7400**
  Maximum performance
  Latest generation devices
  Wide software support
- **Quad Processor**
  High processor density
  Lower cost per processor
- **L2 Cache**
  Maximum performance
  Cost effective
- **RACE++ Interface**
  Fast inter-processor comms
  Scalable systems
- **Dual PMC sites**
  High-speed data I/O
  Fewer VME slots
- **Commercial & Rugged Builds**
  Low cost development
  One architecture suits many applications

Utilities are supplied to test and program the BIOS and NVRAM, as well as programming examples for the timer and serial port.



PCI Buses

The VQG4 features two PCI buses. One segment supports 64-bit/ 66MHz PCI and is made available to one of the PMC mezzanine sites, the other is 32-bit/33MHz only and is used for all other PCI devices including the second PMC slot.

Interrupt Handling

The VQG4 has a large number of interrupt sources. Through the use of an interrupt handler, any interrupt source can be associated with any processor. This provides the flexibility to either share the load of maintaining the peripherals or nominating a single processor to control the system. Each of the four PowerPC processors are also capable of interrupting each other for efficient system synchronization and inter-processor communications.

Peripherals

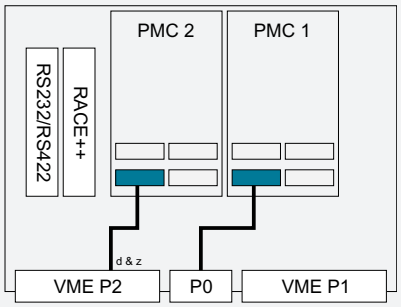
Even though the VQG4 is designed to be a multi-processor card, it also includes most of the resources found on more traditional single board computers. These include networking, serial I/O, mezzanine expansion sites, etc. The VQG4 can also be supplied as a single processor variant: a classic SBC and array processor in one.

- **10/100baseT Ethernet**
  Front panel RJ45 accessible from any CPU with an auto-sensing interface. This device is a full 32-bit PCI interface with DMA capability.
- **Quad RS232**
  Each CPU has its own RS232 channel. All four ports are available through a front panel mini-D connector. A separate breakout cable is available.

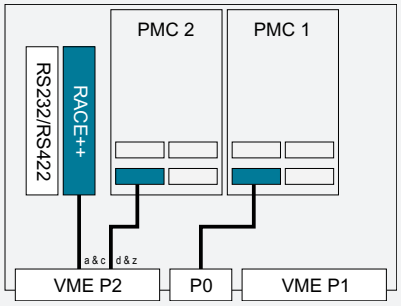
Backplane Data I/O

For some applications backplane/ non-front panel connectivity is the only permitted option - this is often the case for conduction-cooled systems. The VQG4 support VME backplane I/O and has option to route both PMC sites, RACE++ or serial (RS232 or RS422). By using one of the PMC user I/O routing options, PCI through VME P0 is also an option.

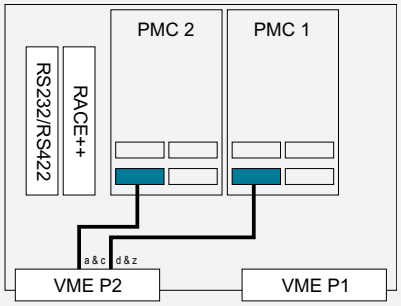
Each PMC module can have a user I/O connector providing up to 64 I/O signals: 128 in total for the VQG4's two sites. These I/O signals can be routed to either the VME P2 DIN connector (rows a&c, d&z) or the VME P0 connector (if fitted). If the RACE++ option is used, VME P2 rows a & c are not available to the PMCs for I/O. See figure 3 for PMC routing schemes depending whether P0 and RACE++ option are fitted.



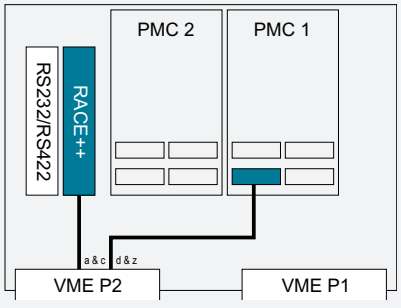
VME P0 without RACE
Figure 3a



VME P0 with RACE++
(RACE++ PMC permits dual RACE++ option)
Figure 3b



No RACE or VME P0
Figure 3c



With RACE and no P0
(RACE++ PMC permits dual RACE++ option)
Figure 3d

Standard VME P0/P2 Options

Option	uses PMC site?	VME P0	VME P2 a&c	VME P2 d&z
Embedded RACE++	-	-	yes	-
Rear 4x RS422	-	-	yes (partial)	-
Rear 4x RS232	-	-	yes (partial)	-
PCI to VME P0	yes	yes	-	-
Dual PMC VME P2	-	-	yes	yes
Dual PMC P2/P0	-	yes	-	yes
Dual RACE++	yes	-	yes	yes
Special	-	optional	optional	optional

VQG4 I/O Backplane Routing Options
Table 2

RACE++™ Interface

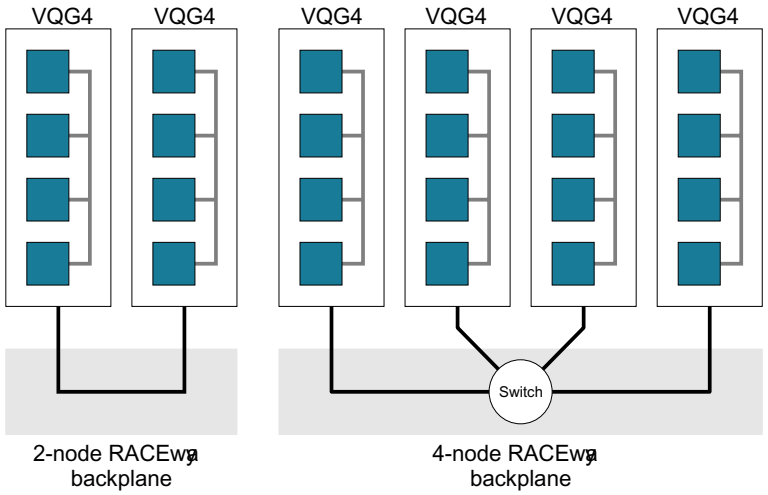
A widely adopted standard for multi-processing systems, RACE++ allows processor arrays to have point-to-point communications with bandwidths of up to 267Mbytes/sec. The VQG4 provides a single RACE++ interface shared between all the onboard PowerPC processors. The RACE++ fabric between boards is implemented using a backplane overlay. For a basic system with just two nodes, this is a simple passive link. Overlays with more than two nodes (slots) use active switches to establish (see figure 4). Each switch dynamically routes data between RACE++ nodes. Provided two or more transactions are not trying to simultaneously access a common node, the data streams will occur concurrently up to the switch's bisection data rate limit.

RACE++ is backwards compatible with RACE: RACE++ can use existing RACE equipment or be used to upgrade existing systems.

Mechanically, the RACE and RACE++ backplanes fit as a mezzanine backplane to VME picking up the VME P2 rows a & c connections and any necessary power.

Dual RACE++™ Option

The VQG4 can be supplied with an onboard RACE++ option and uses the default VME P2 connector on rows a&c. The VQG4 can support a second RACE++ interface using a RACE++ PMC module. This requires the VQG4 configured for one of the PMC sites routed to the VME P2 rows d&z.



2 and 4-slot RACE/RACE++ backplane links
Figure 4

PCI PZERO Backplane

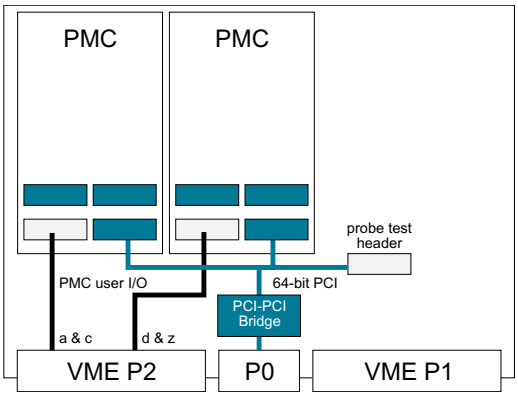
For bussed inter-board connections, the VQG4 can use PCI. This is faster than VME, but limited to fewer slots. In many applications, VME is used for control and alternative datapath are used for the real-time I/O requirements, PCI is ideally suited to this requirement because there are many PCI based peripheral devices.

PCI from the VQG4 uses a PCIzero PMC module. This routes, via a PCI-PCI bridge PCI to the VME P0 connector. PCI P0 backplanes are used to provide the interboard links. The PCI backplanes provides all the necessary arbiters and timings required to support the 64-bit/ 66MHz PCI bus including a unique interrupt for each slot.

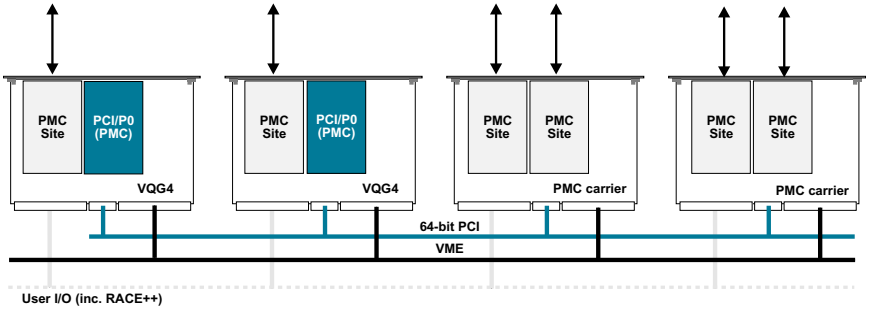
PCI/P0 PMC Carrier

Designed for use with the VQG4 PCI over VME P0 option, a 6U dual PMC carrier (see figure 5) provides one (or more) VQG4s with extra I/O sites. SInce the PCI bus is 64-bit wide and capable of running at 66MHz its bandwidth of up to 528Mbytes/sec provides enough performance for all but the most demanding applications.

The PMC carrier has the option of having a probe test header. These connectors are designed to work with the Hewlett Packard High Density Probe Adaptor, HP5346A and enable the PCI to be monitored as part of a test environment..



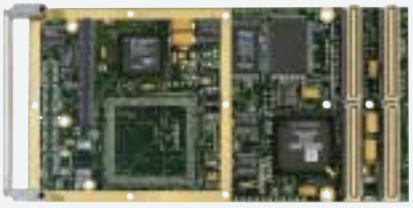
PCI PMC Carrier
Figure 5



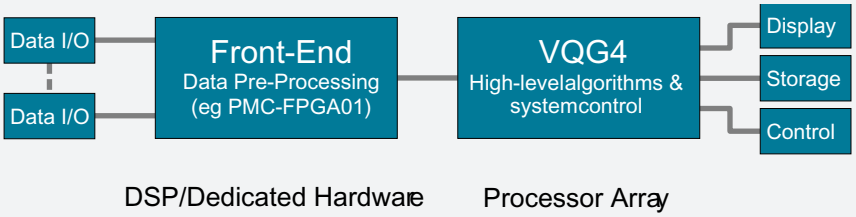
PCI over VME P0
Figure 6

Embedded DSP and FPGAs

Despite the performance of PowerPC processor arrays, some applications benefit from the use of more specialized processing devices such as DSPs which have efficient I/O architectures, optimized libraries and greater levels of code predictability or FPGAs for increased performance and customisable I/O. An example might be a radar application which requires very fast front-end data processing but using an easier to program PowerPC CPU based solution to run more sophisticated algorithms and system control. Transtech's DSP products and tools make this easy to do. For example, the PMC-FPGA01 brings the performance of a Xilinx Virtex FPGA I/O and pre-processing with a high-speed PCI link directly into the PowerPC array. The PMC-FPGA01 has the option of using standard digital interface modules such as LVDS or FPDP or even custom - just by changing the I/O adaptor module.



PMC-FPGA01
example front-end I/O processing
Figure 7



Generic Signal Processing System
Figure 8

Software

Software

- **Real-Time Operating Systems**
  VxWorks
  LynxOS
- **Other Operating Systems**
  Linux (HardHat)
- **Test & Diagnostics**
  Power-On Self Test
  Board level test libraries
- **DSP Libraries**
  VSIPL defined libraries with PowerPC AltiVec optimizations
- **System Communications Libs**
  DSP communications libraries
  RACE APIs
- **PMC Driver Support**
  FPDP
  Analog I/O
  ATM
  Fibre-channel
  Video I/O
  RF Transceiver

Overview

The VQG4 lends itself to different applications and markets. These demands require the VQG4 to be available with different layers of software support. For system critical applications Power-On Self Test (POST), Built-In Test (BIT) and the Board Test Library provides a power up and run-time system diagnostic. In addition, the VQG4 has both real-time and non real-time operating systems targeted ports. To make application development easier I/O board drivers and optimized libraries are also available.

DSP Libraries

Based on the DARPA definitions, the VSIPL library provides a core set of common DSP functions for the VQG4 and optimized for the PowerPC 74xx CPUs. The basis behind the DARPA definition is to provide a processor and operating-system independent set of DSP functions thereby allowing for maximum code portability.

- **VSIPL API**
  Over 500 functions
  DARPA sponsored
  PowerPC G4 (AltiVec) optimizations
- **VSIPL features**
  Portability
  Object-based description
  Opaque objects such as blocks, and views on the blocks (vectors, matrices, and tensors)
  Development and production modes
  Public and private data arrays enabling implementation optimizations
  Explicit memory/algorithm hints
- **Functionality**
  Scalar Functions
  Vector and Matrix (scalar, unary, binary, logical arithmetic, selection and data generation)
  Linear Algebra
  Signal Processing (FFTs, window, filter and convolution routines)
  Image Processing

This library is available separately.

Development Tools

The VQG4 includes a JTAG TAP socket and adapter that is compatible with AMC's PowerTAP™ debugger. This offers power and is functionally similar to a full in-circuit emulator.

Power-On Self Test

The VQG4 can be configured to go into a self-test mode at power-up. The built-in test runs from flash memory prior to any operating system kernel being started. All the major system components checked to see if they are operational with the diagnostic results written to registers. If the board fails, these results can be retrieved using a debugger/emulator tool externally to determine where the failure occurred. Since the VQG4 can be fitted with up to four PowerPC processors, the boot sequence can proceed if any of the CPUs fail: this is done by sequentially releasing each processor from reset should a time-out occur indicating a CPU problem. A front panel LED is used to indicate the diagnostic pass (green), non-catastrophic failure (yellow) or fail (red). Some of these tests include:

- Boot of processor (arbiter, memory address and control lines)
- Memory (inc. ECC force error)
- Flash & NVRAM checksums
- PCI device configuration
- Loopback where possible (Ethernet, RACE++, VME)
- Timer
- Processor to processor mailbox interrupts

The VQG4 can also be tested or run diagnostic functions when running applications under VxWorks. These are accessed using the 'board test library'. Some of these tests are destructive; for example the memory test will destroy the contents of the memory tested.

Vector Signal & Image Processing Library (VSIPL) Functions

Initialization/Finalization Op
vsip_init — Initialize the Library
vsip_finalize — Terminate the Library

Array and Block Object Functions
vsip_dblockadmit_p — Block Admit
vsip_blockbind_p — Memory Block Bind
vsip_cblockbind_p — Complex Memory Block Bind
vsip_dblockcreate_p — Memory Block Create
vsip_dblockdestroy_p — Memory Block Destroy
vsip_blockfind_p — Memory Block Find
vsip_cblockfind_p — Memory Complex Block Find
vsip_blockrebind_p — Block Rebind
vsip_cblockrebind_p — Complex Block Rebind
vsip_blockrelease_p — Block Release
vsip_cblockrelease_p — Complex Block Release
vsip_cstorage — Complex Storage

Vector View Object Functions
vsip_dvalldestroy_p — Destroy Vector and Block
vsip_dvbind_p — Create and Bind a Vector View
vsip_dvcloneview_p — Create Vector View Clone
vsip_dvcreate_p — Create Vector
vsip_dvdestroy_p — Destroy Vector View
vsip_dvget_p — Vector Get Element
vsip_dvgetattrib_p — Vector Get View Attributes
vsip_dvgetblock_p — Vector Get Block
vsip_dvgetlength_p — Vector Get Length
vsip_dvgetoffset_p — Vector Get Offset
vsip_dvgetstride_p — Vector Get Stride
vsip_vimagview_p — Create Imaginary Vector View
vsip_dvput_p — Vector Put
vsip_dvputattrib_p — Put Vector View Attributes
vsip_dvputlength_p — Vector Put Length
vsip_dvputoffset_p — Vector Put Offset
vsip_dvputstride_p — Vector Put Stride
vsip_vrealview_p — Create Real Vector View
vsip_dvsubview_p — Create Subview Vector View

Matrix View Object Functions
vsip_dmalldestroy_p — Destroy Matrix and Block
vsip_dmbind_p — Create and Bind a Matrix View
vsip_dmcloneview_p — Create Matrix View Clone
vsip_dmcolview_p — Create Column-View Matrix View
vsip_dmcreate_p — Create Matrix
vsip_dmdestroy_p — Destroy Matrix View
vsip_dmdiagview_p — Create Matrix Diagonal View
vsip_dmget_p — Matrix Get Element
vsip_dmgetattrib_p — Matrix Get View Attributes
vsip_dmgetblock_p — Matrix Get Block
vsip_dmgetcollength_p — Matrix Get Column Length
vsip_dmgetcolstride_p — Matrix Get Column Stride
vsip_dmgetoffset_p — Matrix Get Offset
vsip_dmgetrowlength_p — Matrix Get Row Length
vsip_dmgetrowstride_p — Matrix Get Row Stride
vsip_dmimagview_p — Create Imaginary Matrix View
vsip_dmput_p — Matrix Put Element
vsip_dmputattrib_p — Matrix Put View Attributes
vsip_dmputcollength_p — Matrix Put Column Length
vsip_dmputcolstride_p — Matrix Put Column Stride
vsip_dmputoffset_p — Matrix Put Offset
vsip_dmputrowlength_p — Matrix Put Row Length
vsip_dmputrowstride_p — Matrix Put Row Stride
vsip_dmrealview_p — Create Real Matrix View
vsip_dmrowview_p — Create Matrix Row View
vsip_dmsubview_p — Create Subview Matrix View
vsip_dmtransview_p — Create Matrix Transposed View

SCALAR FUNCTIONS
vsip_arg_p — Complex Scalar Argument
vsip_cadd_p — Complex Scalar Add
vsip_cdiv_p — Complex Scalar Divide
vsip_cexp_p — Complex Scalar Exponential
vsip_cjmul_p — Complex Conjugate Scalar Multiply
vsip_cmag_p — Complex Scalar Magnitude
vsip_cmagsq_p — Complex Scalar Magnitude Squared
vsip_cmplx_p — Complex Sc
vsip_cmul_p — Complex Scalar Multiply
vsip_cneg_p — Complex Scalar Negate
vsip_conj_p — Complex Scalar Conjugate
vsip_crecip_p — Complex Scalar Reciprocal
vsip_csqrt_p — Complex Scalar Square Root
vsip_csub_p — Complex Scalar Subtract
vsip_imag_p — Complex Scalar Imaginary
vsip_polar_p — Complex Scalar Polar
vsip_real_p — Complex Scalar Real
vsip_rect_p — Complex Scalar Rectangular

Index Scalar Functions
vsip_matindex — Matrix Index
vsip_mcolindex — Matrix Column Index
vsip_mrowindex — Matrix Row Index

RANDOM NUMBER GENERATION
vsip_randcreate — Create Random State
vsip_randdestroy — Destroy Random State
vsip_dsrandu_p — Uniform Random Numbers
vsip_dsrandn_p — Gaussian Random Numbers

VECTOR & ELEMENTWISE OPERATIONS

Elementary Math Functions
vsip_vacos_p — Vector Arccosine
vsip_vasin_p — Vector Arcsin
vsip_vatan_p — Vector Arct
vsip_vatan2_p — Vector Arctangent of Two Arguments
vsip_vcos_p — Vector Cosine
vsip_dvexp_p — Vector Exponential
vsip_vexp10_p — Vector Exponential Base 10
vsip_vlog_p — Vector Log
vsip_vlog10_p — Vector Log Base 10
vsip_vsin_p — Vector Sine
vsip_dvsqrt_p — Vector Square Root
vsip_vtan_p — Vector Tangent

Unary Operations
vsip_cvconj_p — Vector Conjugate
vsip_veuler_p — Vector Euler
vsip_dvmag_p — Vector Magnitude
vsip_cvmagsq_p — Vector Complex Magnitude Squared
vsip_dvmeanval_p — Vector Mean Value
vsip_dvmeansqval_p — Vector Mean Square Value
vsip_dvmodulate_p — Vector Modulate
vsip_dvneg_p — Vector Negate
vsip_dvrecip_p — Vector Reciprocal
vsip_vrsqrt_p — Vector Reciprocal Square Root
vsip_vsq_p — Vector Square
vsip_vsumval_p — Vector Sum Value
vsip_vsumsqval_p — Vector Sum of Squares Value

Binary Operations
vsip_dvadd_p — Vector Add
vsip_dsvadd_p — Scalar Vector Add
vsip_dvdiv_p — Vector Divide
vsip_svdiv_p — Scalar Vector Divide
vsip_dvexpoavg_p — Vector Exponential Average
vsip_vhypot_p — Vector Hypotenuse
vsip_cvjmul_p — Vector Conjugate Multiply (Elementwise)
vsip_dvmul_p — Vector Multiply (Elementwise)
vsip_dsvmul_p — Scalar Vector Multiply
vsip_dvdmmul_p — Vector-Matrix Multiply (Elementwise)
vsip_dvsub_p — Vector Subtract
vsip_dsvsub_p — Scalar Vector Subtract

Ternary Operations
vsip_dvam_p — Vector Add and Multiply
vsip_dvma_p — Vector Multiply and Add
vsip_dvmsa_p — Vector Multiply, Scalar Add
vsip_dvmsb_p — Vector Multiply and Subtract
vsip_dvsam_p — Vector Scalar Add, Vector Multiply
vsip_dvsbm_p — Vector Subtract and Multiply
vsip_dvsma_p — Vector Scalar Multiply, Vector Add
vsip_dvsmsa_p — Vector Scalar Multiply, Scalar Add

Logical Operations
vsip_valltrue_bl — Vector All True
vsip_vanytrue_bl — Vector Any True
vsip_vleq_p — Vector Logical Equal
vsip_vlge_p — Vector Logical Greater Than or Equal
vsip_vlgt_p — Vector Logical Greater Than
vsip_vlle_p — Vector Logical Less Than or Equal
vsip_vllt_p — Vector Logical Less Than
vsip_vlne_p — Vector Logical Not Equal

Selection Operations
vsip_vclip_p — Vector Clip
vsip_vinvclip_p — Vector Inverted Clip
vsip_vindexbool — Vector Index a Boolean
vsip_vmax_p — Vector M
vsip_vmaxmg_p — Vector Maximum Magnitude
vsip_vcmaxmgsq_p — Vector Complex Max Magnitude Squared
vsip_vcmaxmgsqval_p — Vector Complex Max Mag Squared Value
vsip_vmaxmgval_p — Vector Maximum Magnitude Value
vsip_vmaxval_p — Vector Maximum Value
vsip_vmin_p — Vector Mi 214
vsip_vminmg_p — Vector Minimum Magnitude
vsip_vcminmgsq_p — Vector Complex Min Magnitude Squared
vsip_vcminmgsqval_p — Vector Complex Min Mag Squared Value
vsip_vminmgval_p — Vector Minimum Magnitude Value
vsip_vminval_p — Vector Minimum Value

Bitwise and Boolean Logical Operators
vsip_vand_p — Vector AND
vsip_vnot_p — Vector NOT
vsip_vor_p — Vector OR
vsip_vxor_p — Vector Exclusive OR

Element Generation and Copy
vsip_dscopy_p_p — Vector/Matrix Copy
vsip_dvfill_p — Vector Fill
vsip_vramp_p — Vector Ram

Manipulation Operations
vsip_vcmplx_p — Vector
vsip_dvgather_p — Vector Gather
vsip_vimag_p — Vector Im
vsip_vpolar_p — Vector Polar
vsip_vreal_p — Vector Real
vsip_vrect_p — Vector Rectangular
vsip_dvscatter_p — Vector
vsip_dvswap_p — Vector Swap

SIGNAL PROCESSING FUNCTIONS

FFT Functions
vsip_dfftx_create_f — Create 1D FFT Object
vsip_ccfftx_f — FFT Complex to Complex
vsip_crfftop_f — FFT Complex to Real
vsip_rcfftop_f — FFT Real to Complex
vsip_ccfftmx_f — FFT Multiple Complex to Complex
vsip_crfftmop_f — FFT Multiple Complex to Real
vsip_rcfftmop_f — FFT Multiple Real to Complex
vsip_dfftmx_create_f — Create Multiple FFT Object
vsip_fftn_destroy_f — Destroy FFT Object
vsip_fftn_getattr_f — FFT Get Attributes

Convolution/Correlation Functions
vsip_conv1d_create_f — Create 1D Convolution Object
vsip_conv1d_destroy_f — Destroy Conv1D Object
vsip_conv1d_getattr_f — Conv1D Get Attributes
vsip_convolve1d_f — 1D 257
vsip_dcorr1d_create_f — Create 1D Correlation Object
vsip_dcorr1d_destroy_f — Destroy Corr1D Object
vsip_dcorr1d_getattr_f — Corr1D Get Attributes
vsip_dcorrelate1d_f — 1D Correlation

Window Functions
vsip_vcreate_blackman_f — Create Blackman Window
vsip_vcreate_cheby_f — Create Chebyshev Window
vsip_vcreate_hanning_f — Create Hanning Window
vsip_vcreate_kaiser_f — Create Kaiser Window

Filter Functions
vsip_dfir_create_f — Create Decimated FIR Filter
vsip_dfir_destroy_f — Destroy FIR Filter Object
vsip_dfirflt_f — Decimated FIR Filter
vsip_dfir_getattr_f — FIR Get Attributes

Miscellaneous Signal Processing Functions
vsip_shisto_p — Histogram

LINEAR ALGEBRA FUNCTIONS

Matrix and Vector Operations
vsip_cmherm_p — Matrix Hermitian
vsip_cvjdot_p — Complex Vector Conjugate Dot Product
vsip_dgemp_p — General Matrix Product
vsip_dgems_p — General Matrix Sum
vsip_dmprod_p — Matrix Product
vsip_dmprodh_p — Matrix Hermitian Product
vsip_dmprodj_p — Matrix Conjugate Product
vsip_dmprodt_p — Matrix Transpose Product
vsip_dmvprod_p — Matrix Vector Product
vsip_dmtrans_p — Matrix Transpose
vsip_dvdot_p — Vector Dot Product
vsip_dvmprod_p — Vector Matrix Product
vsip_dvouter_p — Vector Outer Product

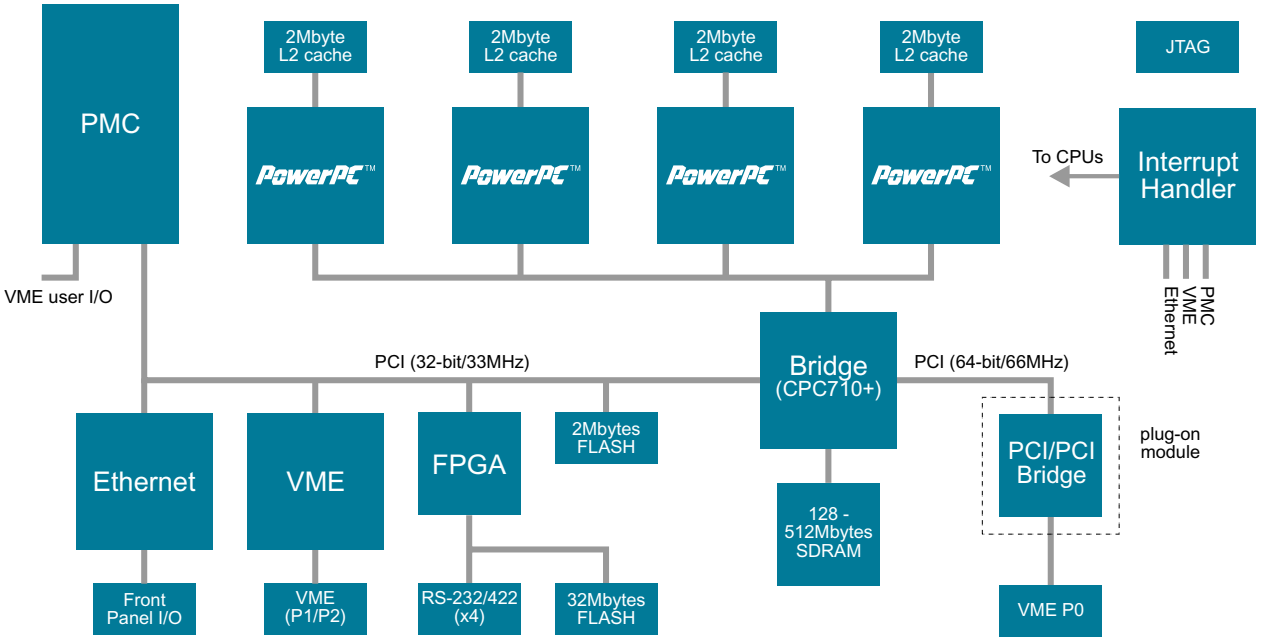
Special Linear System Solvers
vsip_dcovsol_p — Solve Covariance System
vsip_dllsqsol_p — Solve Linear Least Squares Problem
vsip_dtoepsol_p — Solve Toeplitz System

General Square Linear System Solver
vsip_dlud_p — LU Decomposition
vsip_dlud_create_p — Create LU Decomposition Object
vsip_dlud_destroy_p — Destroy LUD Object
vsip_dlud_getattr_p — LUD Get Attributes
vsip_dlusol_p — Solve General Linear System

Symmetric Positive Definite Linear System Solver
vsip_dchold_p — Cholesky Decomposition
vsip_dchold_create_p — Create Cholesky Decomposition Object
vsip_dchold_destroy_p — Destroy CHOLD Object
vsip_dchold_getattr_p — CHOLD Get Attributes
vsip_dcholsol_p — Solve SPD Linear System

Overdetermined Linear System Solver
vsip_dqrd_p — QR Decomp
vsip_dqrd_create_p — Create QR Decomposition Object
vsip_dqrd_destroy_p — Destroy QRD Object
vsip_dqrd_getattr_p — QRD Get Attributes
vsip_dqrdprodq_p — Product with Q from QR Decomposition
vsip_dqrdsolr_p — Solve Linear System Based on R from QR Dec
vsip_dqrsol_p — Solve Covariance or LLSQ System

Block Diagram



Technical Specification

Processor

Processor	PowerPC 7400
Number	1, 2 or 4 (shared bus)
Clock Speed	400MHz
Memory Bridge	CPC710+ (100MHz Memory, 32-bit/ 33MHz, 64-bit/66MHz PCI)

Memory

SDRAM	128, 256, 512Mbytes
L2 cache	2Mbytes
FLASH	2Mbytes (boot) + 32Mbytes (user)

PMC Sites

Number	2 (one shared with RACE++, one shared with VME and Ethernet)
I/O Routing (VME P0/P2)	(see Table 2)

Ethernet

Device	SYM53C885
Type	10baseT, 100baseTX auto sensing (front panel only)

RACE++

Device	PBX Bridge
Bandwidth	Up to 266Mbytes/sec

Serial I/O

Ports	4 (RS232 or RS422)
Connector	6-pin mini DIN (front panel) or VME P2 (a&c)

VME

Device	Tundra Universe II
Compliance	A32/A24/A16 master and slave D64/D32/D16/D08 master and slave, MBLT, BLT, ADOH, RMW, LOCK

PMC Carrier

PMC sites	2
Interface	PCI over P0 (64-bit/66MHz)

Debug

Type	RISCwatch
Connector	16-pin keyed header

Software Support

VxWorks	5.4.x BSP/Wind River Tornado 2.1
Linux	MontaVista HardHat
Signal Processing Library	PowerPC-VSIPL (513 functions optimized for PowerPC 74x0)
Test Software	Built-In Self Test (BIT)
LynxOS	TBA

Build Options

Conformal coating	Contact Transtech for information
Extended Temperature	Contact Transtech for information
PCI Over VME P0	
Dual RACE++	

20 Thornwood Drive, Ithaca, NY 14850-1263, USA
Tel: 607 257 8678 Fax: 607 257 8679
email: sales@transtech-dsp.com

Manor Courtyard, Hughenden Avenue, High Wycombe, HP13 5RE, UK
Tel: +44(0)1494 464432 Fax: +44(0)1494 464472
email: sales@transtech-dsp.com

TRANSTECH DSP™

www.transtech-dsp.com