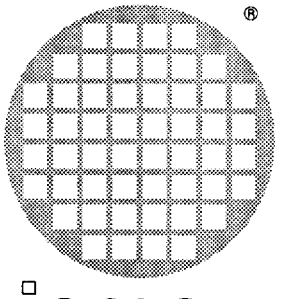
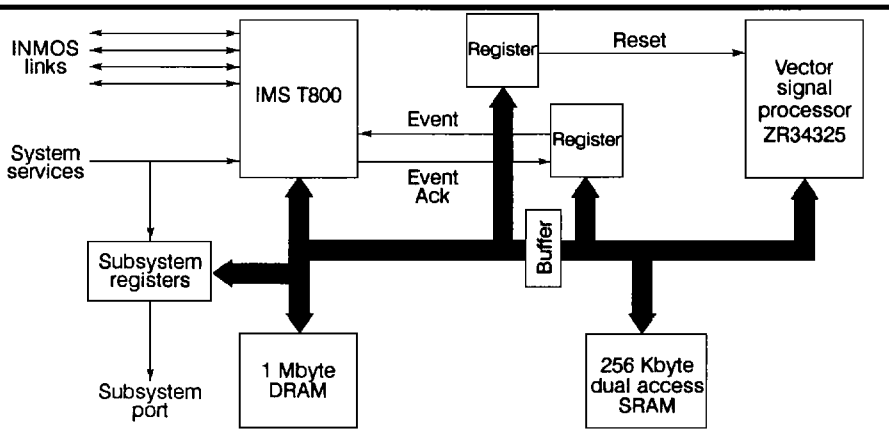


IMS B420
Vector processing TRAM Size 4

inmos®

Engineering Data



FEATURES

- IMS T800 -25 or T800 -20 floating point transputer
- High performance vector/signal processing co-processor (ZR34325)
  – e.g. 1K complex FFT < 2ms for 25MHz co-processor
- Both processors support IEEE 754-1985 floating point
- 4 INMOS serial communication links allowing connection of multiple VecTRAMs
- 1 Mbyte DRAM for IMS T800
- 256 Kbyte, dual access SRAM for full speed co-processor operation
- Size 4 TRAM
- Sub-system port
- Supplied with IMS F000 C and occam libraries
- IMS F007 DSP libraries available
- Designed to a published specification (*INMOS Technical Note 29*)

GENERAL DESCRIPTION

The IMS B420 VecTRAM is a transputer module combining the communications ability and scalar floating point performance of the IMS T800 with a high performance vector/signal processing co-processor (ZR34325). The two processors can operate concurrently, using separate dynamic and static memory blocks. The vector/signal processor is normally operated as a slave to the IMS T800 which can read and write to the SRAM, to set up vector/DSP routines as well as to load data for processing. The two processors can handshake via interrupts, thus allowing the transputer to initiate vector routines and the co-processor to signal the termination of the requested task.

Examples at 25MHz operation of the coprocessor's capabilities are: 1K complex FFT in 1.8 ms, 10×10 by 10×10 matrix multiplication in approximately 135μs and 64-tap FIR in 6 μs.

Application areas include speech and image processing, graphics and numerical processing, radar, sonar and seismology.

SGS-THOMSON MICROELECTRONICS
INMOS is a member of the SGS-THOMSON Microelectronics group

42 1467 00 July 1990

31.1 Introduction

The IMS B420 (VecTRAM) has been developed to offer both scalar and vector processing in a single 'TRAM' module. This module combines the scalar floating/integer performance and general purpose capabilities of the transputer; with a high performance vector/signal processing co-processor. A set of software libraries is available which allows programs written in high level languages, running on the transputer, to call vector routines executed on the co-processor.

The IMS T800 transputer has sole access to 1 Mbyte of dynamic memory. 256 Kbyte of the static memory is accessible by both the transputer and the co-processor, under the control of an arbiter. The two processors can operate concurrently, using the separate dynamic and static memory blocks. The vector /signal processor is operated as a slave to the IMS T800 which can read and write to the static RAM, to set up vector/signal processing routines as well as to transfer data for processing.

The operation of the module is better understood with a description of the two processors.

The IMS T800 Processor

The IMS T800 transputer integrates a 12.5 MIPS (25MHz) 32-bit central processing unit(CPU), a 2 MFLOP scalar floating point unit (FPU) supporting IEEE arithmetic, communication links for parallel processing, and 4 Kbytes of on-chip memory. Each communication link is capable of providing around 1.2 Mbyte/s data rate in each direction. These allow easy connection to other TRAMs and transputers.

The IMS T800 also has one- and two-dimensional block-move capabilities. This allows blocks of data to be moved speedily from one memory segment to another. This is particularly important in the support for a vector co-processor where data blocks must be transferred for processing. The two-dimensional block-move, in particular, allows operations such as corner turning to be carried out while the data is being transferred.

The ZR34325 Vector/Signal Processor

The ZR34325 is a vector/signal processor which supports vector floating point arithmetic with a peak performance of 37.5 MFLOP. The floating-point arithmetic is 32-bit and conforms to the IEEE 754-1985 standard, making it compatible with the single-precision floating point format on the transputer. The device is optimised to execute efficiently a wide variety of signal/vector processing functions. These include multidimensional FFT's, digital filters, vector-scalar, vector-vector, and matrix operations. The block diagram of the processor, showing various functional blocks, is given in figure 31.1.

The co-processor integrates six main functional units plus on-chip memory and registers. The main functional units are:

- **Execution Unit:** This unit is configured to efficiently execute complex multiply/accumulate operations, while conforming to the IEEE standard for binary floating-point arithmetic. All four rounding modes, as well as $\pm\infty$, NaNs and denormalised numbers are supported. This unit can operate on both internal and external memories. The results can be written in the on-chip RAM or registers. The intermediate or final results which are written to on-chip registers can be automatically copied to off-chip memory. The execution unit consists of three major computational blocks; a 32-bit floating-point multiplier, a 32-bit floating-point adder, and a second adder/subtractor which can perform accumulation in extended 44-bit precision. The execution unit also contains address-generation hardware for accessing internal RAM, and coefficient ROM. It also includes three registers. Among operations supported by the execution unit are vector multiply, vector addition, vector subtraction, comparisons, as well as direct support for higher level functions such as FFT's, matrix operations, and filtering.
- **Bus-Interface Unit:** This unit coordinates the activities of the external memory interface. It supports a 32-bit bidirectional data bus, and a 24-bit address bus allowing a 16 Mword address space. The interface also allows master or slave operations. In master mode the co-processor

has control of the external memory and it fetches its own instruction. In slave mode, an external host can read to/write from the memory-mapped internal registers and memory.

- **Move Unit:** This unit transfers data between internal and external memories. It supports a variety of vector-oriented addressing including: 1-D/2-D transfers, complex and real data addressing, circular buffers, bit-reversal. It also transfers data between the external memory and the execution unit via the vector unit.
- **Vector Unit:** This unit acts as a vector buffer (FIFO) between the move unit and the execution unit. It decouples the execution unit from the external memory behaviour thus improving performance.
- **Control Unit:** This unit supports external memory address computations, and also performs 24-bit integer arithmetic and logical operations on the integer registers. It also carries out all load and store operations on all integer registers.
- **Fetch Unit:** This unit manages an instruction queue that is implemented as a FIFO with four 64-bit words. Every instruction fetched goes through this buffer except program flow control instructions such as LOOP. This allows loop code to be kept in the instruction buffer during the first pass and used subsequently.

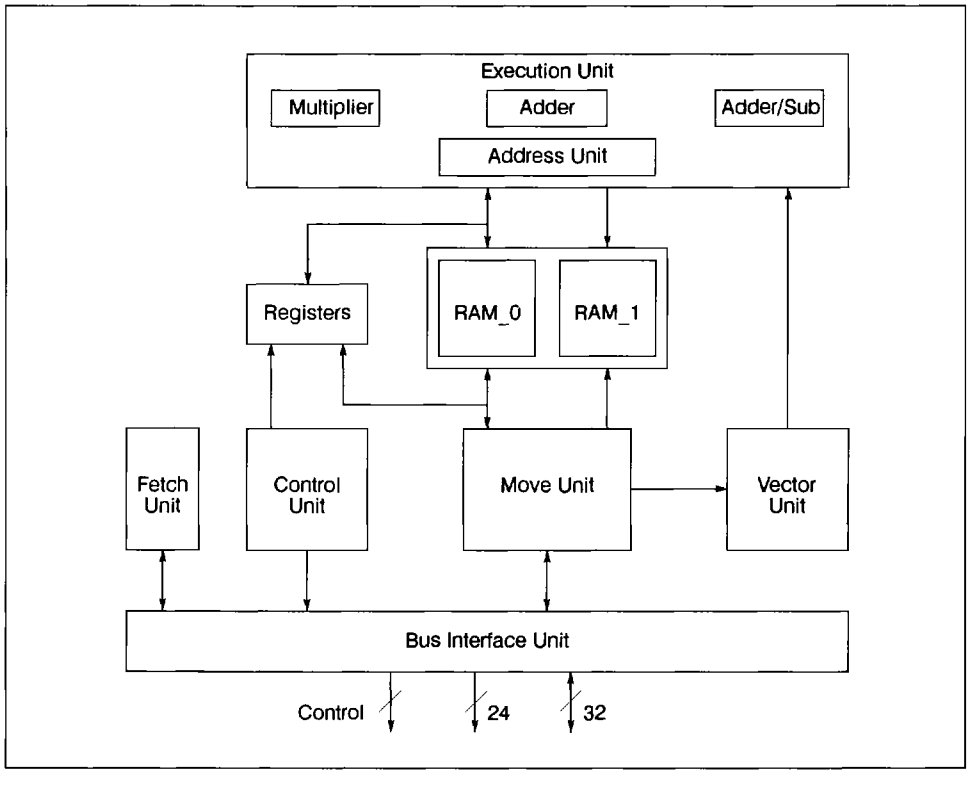


Figure 31.1 The signal/vector processor block diagram

Other major elements within the co-processor are the internal RAM and registers. The internal RAM consists of 512 bytes of memory organised as an array of 64 complex words. The RAM can also be configured as two separate blocks, each of 32 complex words. Operands in the internal RAM can be either complex or real vectors. To increase internal RAM utilisation, double length real vectors can use both real and imaginary parts of the RAMs.

31.2 Transputer memory map

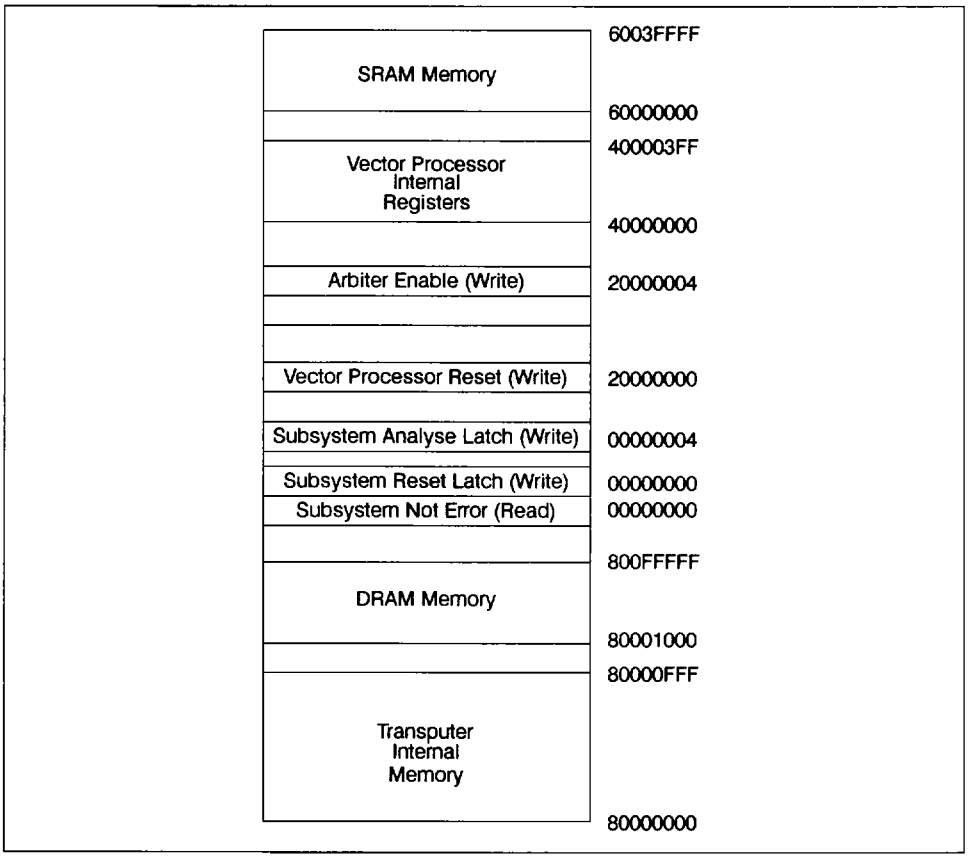


Figure 31.2 Transputer memory map (byte addresses)

- **Vector Processor Reset:** To reset the vector processor, write 0 to this location and wait for at least 1μs. Then write 1 to deassert the reset signal.

- **Arbiter Enable:** Write 1 to enable the SRAM access arbiter. In this mode, both the IMS T800 and the ZR34325 can access the SRAM. Cycle time from the IMS T800 is typically 6 clock cycles (300ns at 20MHz). Write 0 to disable the SRAM access arbiter. In this mode, only the IMS T800 may access the SRAM. This mode provides faster access for block copying of data and programs, while the ZR34325 is not running. Cycle time from the IMS T800 is 4 clock cycles (200ns at 20MHz).The IMS T800 always accesses the DRAM in 4 clock cycles (200ns at 20MHz).

31.3 ZR34325 memory map

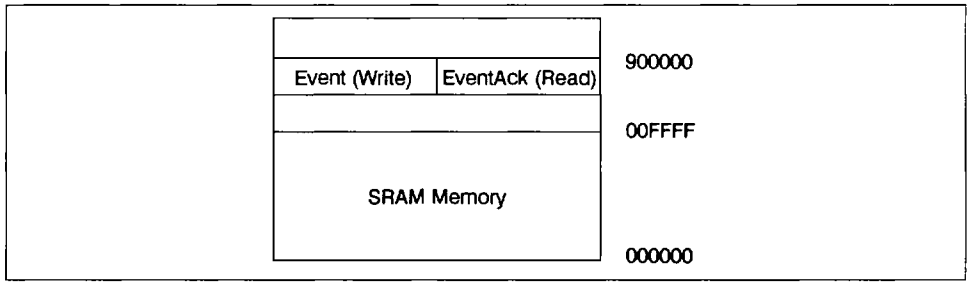


Figure 31.3 ZR34325 memory map (word addresses)

- **Event:** Write 1 to assert the **Event** pin of the IMS T800; write 0 to deassert it. Reading this location returns the state of the the **EventAck** pin of the IMS T800.

31.4 Mechanical details

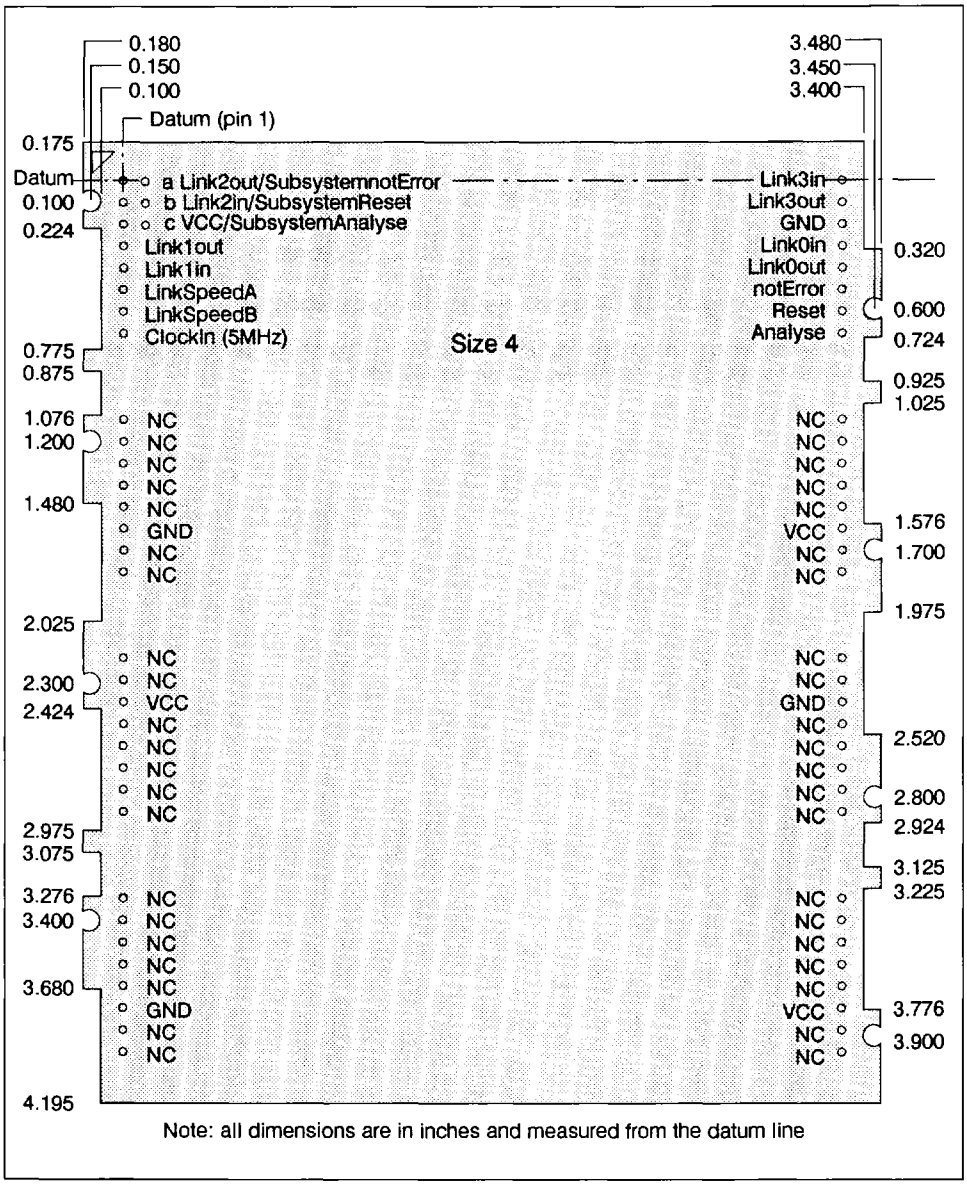


Figure 31.4 IMS B420 outline drawing (All dimensions in inches)

31.5 Specification

TRAM feature		Unit	Notes
IMS T800 transputer	1		
ZR34325 vector processor	1		
Fast dual-port RAM	256	Kbyte	
DRAM	1	Mbyte	
TRAM size	4		
Length	3.66	inch	
Width	4.35	inch	
Pitch between pins	3.30	inch	
Component height above PCB	9.2	mm	
Component height below PCB	3.7	mm	1
Weight (approx.)	113	g	
Storage temperature	0–70	°C	
Operating temperature	0–50	°C	
Power supply voltage (Vcc)	4.75–5.25	Volt	
Power consumption (Max)	9.5	W	2

Table 31.1 IMS B420 specification

NOTES

1 This dimension includes the thickness of the PCB.

2 Measured at Vcc = 5.25V.

31.6 IMS F000 software library

31.6.1 Software support

Currently INMOS provides a C signal/vector processing library (IMS F000A) which allows the co-processor to be used from a high level language running on the transputer. This library is normally supplied in binary and compatible with the parallel C compiler family, IMS x11. Future releases of this software will extend to other compiler families, in particular, the new generation of INMOS C and occam compilers and will support additional functions.

The IMS F000 consists of a library of C functions which implement common vector/signal processing tasks. The functions are callable from a C program running on an IMS B420 TRAM, and can be used to dramatically speed up parallel applications and system performance involving vector/signal processing computation. The use of the libraries is most easily shown by a simple example:

```
#include"decc11.h"      /*Function declarations */
#include"memc11.h"      /*Workspace constants */

main()
{
float a[100], b[100], c[100];
int i, flag;

/* Initialise the co-processor and set up workspaces*/
VT_INIT(co-processor_MEM_BASE, LIB_WORKSPACE_BASE,
USER_WORKSPACE_BASE, co-processor_MEM_TOP);

/* Initialise test data arrays */
for(i=0; i<100; i++){
   a[i] = 10.0;
   b[i] = 20.0;
}

flag =0;

/* Call vector multiply function */
VT_MULT_F32R(a, 1, b, 1, c, 1, 100, flag);
}
```

During program execution, when a vector library function is called, it first checks the addresses of its operands. If the data to be processed is in the transputer local memory space, it is automatically copied (using the blockmove capability of the T800) to a predetermined area in the co-processor space. The co-processor is then activated to execute the required function.

If the destination vector operand address, specified in the call, is in the transputer space, the processed data is automatically copied back to the specified area in the transputer memory space. This built-in copying means that the co-processor operation can be totally transparent to the programmer, and programs can be accelerated without the need for detailed knowledge about the operation of the IMS B420 TRAM.

The overhead associated with data copying between the transputer and co-processor (or visa versa) is avoided if the source and destination operands for the specified function are already in the co-processor local memory space. The library functions automatically check the operand addresses and take appropriate action. In general, if the address of an input or an output operand, in a function call, is in the co-processor space, no data copying will take place for that operand. This is particularly important if operands are to undergo several vector/signal processing operations. For optimal performance the user can specify desti-

nation (and/or source) addresses which are local to the co-processor address space. In this way data copying, between the transputer and the shared memory area can be minimised.

Apart from vector and arithmetic functions, the IMS F000 includes efficient vector move functions which allow optimisation at the application level. The library also supports co-processor control calls which are used to set rounding modes and error handling.

IMS F000A function calls include:

VT_ABS_F32R
Function: Vector Absolute Value – Real

VT_ADD_F32R
Function: Vector Addition – Real

VT_CMP_F32R
Function: Vector Compare – Real

VT_DISERROR
Function: Disable co-processor Error Flags

VT_DOT_F32C
Function: Vector Dot Product – Complex

VT_DOT_F32R
Function: Vector Dot Product

VT_ENERROR
Function: Enable co-processor Error Interrupts

VT_FFT_F32C
Function: Fast Fourier Transform – Complex

VT_F32TOI16_R
Function: Vector Floating point to Integer (16-bit) Conversion

VT_IFFT_F32C
Function: Inverse Fast Fourier Transform – Complex

VT_I16TOF32_R
Function: Vector Integer (16-bit) to 32-bit floating point Conversion

VT_LOG10_F32R
Function: Vector Log to the base 10 – Real

VT_MAG_F32C
Function: Vector Magnitude – Complex

VT_MAGSQ_F32C
Function: Vector Magnitude Square – Complex

VT_MAX_F32R
Function: Find the Element with the Maximum Value and its position – Real

VT_MEAN_F32R
Function: Vector Mean – Real

VT_MIN_F32R
Function: Find the Element with the Minimum Value and its position – Real

VT_MOV_BYTE
Function: Vector Move – Bytes

VT_MOV_WORD
Function: Vector Move – Words (32-bit)

VT_MULT_F32C
Function: Vector Multiply – Complex

VT_MULT_F32R
Function: Vector Multiply – Real

VT_POWER_F32R
Function: Vector Power – Real

VT_ROUNDMODE
Function: Modify co-processor Rounding mode

VT_SCALE_F32R
Function: Vector Scale – Real

VT_SQRT_F32R
Function: Vector Square Root – Real

VT_SUB_F32R
Function: Vector Subtract – Real

31.7 Ordering Information

Description	Order number
IMS B420 VecTRAM 20 MHz operation	IMS B420-3*
IMS B420 VecTRAM 25 MHz operation	IMS B420-5*
IMS F000 VecTRAM library software – supplied on IBM PC format 5 1/4" and 3 1/2" discs.	IMS F000A-1
IMS F007 DSP software library – supplied on IBM PC format 5 1/4" and 3 1/2" discs.	IMS F007A-1

Table 31.2 Ordering information

* Includes IMS F000A-1 VecTRAM library software