APPROACHING THE DESKTOP SUPERCOMPUTER

Trevor G Marshall

YARC Systems Corporation

ABSTRACT

Todays desktop Personal Computers, in conjunction with RISC coprocessing technology, can deliver the same computational performance as an IBM 3090 (approx 25% of a CRAY-1) without the delays associated with job submission and execution priorities. This is changing the way researchers view their computing capabilities, and opening up a new era of productivity in scientific computing.

This paper examines the history of 'desktop supercomputing', the current status of the technology, and the problems still remaining to be solved.

INTRODUCTION

The technical computing marketplace has been traditionally dominated by networks of terminals connected to multiuser, multitasking mainframes or supercomputers. Some years ago the terminals were given local intelligence, allowing some functions, such as source editing, to be offloaded from the host, and more recently Personal Computers have allowed some degree of algorithmic testing before submission of the job to the host [1].

By offloading computing tasks to the desktop a scientist gains in productivity in a number of ways. Firstly, desktop computing is invariably less expensive than CPU time at a supercomputing center [1]. Secondly, the time between submission of a job and receipt of the results becomes deterministic, and not subject to time slice availability on the host. Finally, if the PC has some idle time, it costs nothing to run a job with slightly altered parameters, 'on a hunch', just to see what happens. Scientific intuition has often been constrained by the need to stay within a fixed computing budget which was set long before the project began.

The performance of today's advanced RISC PC coprocessor systems is close to that of scientific mainframes, such as the IBM 3090, and the PC will only be clearly outperformed by Supercomputers if the problem is significantly non-scalar.

DESKTOP COMPUTERS

Desktop Computers fall into two major categories. *Workstations* are generally the most expensive desktop computers, often having multiuser capability. *Personal Computers* (in the scientific arena) are typically MSDOS based machines using the INTEL 80386 or 80486 CPUs with 80387 or Weitek floating point chips. The APPLE Macintosh PC has also become very popular with scientists due to its extremely simple user interface.

WORKSTATIONS

Workstations essentially can be viewed as large scale computing systems cut down in size and capability to yield a useful desktop package. They invariably run a variant of the UNIX operating system. UNIX is an operating system that seduces the user with promises of application portability between dissimilar platforms, but which neglects to mention that you cannot get binary portability between microprocessors with different instruction sets. It is an excellent development environment, and is usually emphasised in Computing Science courses in our Universities.

The complexities associated with a UNIX system are often intimidating to the average scientist.

CH2843-1/90/0000/0200$01.00 © 1990 IEEE

Carefully secured access passwords and priorities have little place in a single user, desktop, environment. In addition, a UNIX operating system has significant overhead. Typically as much as 40 megabytes of the disk system and 2 megabytes of RAM storage are dedicated to use by the operating system itself.

Nevertheless workstations are very visible participants in the 'Technical Computing' marketplace, which is currently estimated at around $20 billion, worldwide. Workstations have penetrated about 15% of this total marketplace.

PERSONAL COMPUTERS

Personal computers have traditionally been distinguished from Workstations by two factors: a simplified operating environment and easy extensibility. RAM, disks, and other peripherals can be easily and inexpensively added by a user with virtually no technical expertise.

Additionally, Personal Computers are allegedly 'user friendly'. For instance, if the power to the computer is accidentally removed a Personal Computer will not loose its filesystem. A UNIX system will invariably suffer filesystem damage under the same circumstances, and may require the services of a 'UNIX guru' to recover lost data.

Most operating system commands are less powerful in a Personal Computer. This leads to both advantages and disadvantages. It is not possible to accidentally issue an *'rm * '* command to a PC. The same command in a UNIX system will remove all files from the current directory whether the user really intended that action or not! Conversely that task on a PC takes several additional keystrokes. Typical scientific users prefer the safer PC methodology, whilst programmers prefer the more powerful command structure of UNIX.

Unfortunately even the most advanced Personal computers rarely have the power to perform complex scientific calculations. This is due to limitations in two areas, architectural limitations in the PC, and lack of maturity in the software tools. Both of these can be solved with RISC Coprocessor Technology.

ARCHITECTURAL LIMITATIONS OF PCs

The IBM PC was originally designed with 64 Kilobytes of main memory. This matched the segmented memory architecture of the INTEL 8088 CPU. MSDOS was written to support this segmentation. In order to maintain software compatibility even current versions of DOS (which typically operate with at least 640K of RAM) manipulate data, address and stack spaces in segments of 64K.

This makes it very difficult to support large data structures, such as are common in scientific computing. Extra code has to be included into an application to check which 64K segment a particular array element is stored in before it can be accessed.

Recently a number of compilers have become available using a technology called 'DOS extenders' to use the linear address capabilities of the 80386 processor. These have removed the 64K segmentation limitation, and the performance of 80386 PCs using extender software technology is almost always in excess of that of a VAX 11/780.

The APPLE Macintosh operating system was also designed to manipulate chunks of memory, in this case only 32K large at a time. It is very difficult to port large FORTRAN applications to a Macintosh. Computing Coprocessors are by far the best way to enhance the capabilities of this PC platform.

ARRAY PROCESSORS

One of the earliest approaches to increasing the computing power of a PC was to use Array Processors to perform that portion of the calculations which were enhanced by the vector concurrency or the fast Static RAM available on the array processors.

The early PC AP Systems were derived from the mainframe and minicomputer AP technologies that had become relatively mature before the PC burst onto the scene. Companies such as Sky and Marinco shipped PC array processor products as early as 1984.

Unfortunately, the bandwidth of the PC bus is very much less than that of a mainframe and the computational usefulness of the array processors was limited by the time taken to transfer the data structures from the host PC to and from the AP memory.

In addition, source code has to be rewritten to accomodate array processors, with calls to special subroutines being substituted for appropriate sections of the original code. This is generally not

acceptable to a scientist unless there is just no other way to approach the problem.

COMPUTING COPROCESSORS

In 1984 a new architecture for PC enhancements was described [2]. The 'Trump Card' had a Z8000 16 bit processor running a BASIC language software system as a computing coprocessor.

Whereas an array processor only runs selected portions of a user's code a computing coprocessor takes over the execution of the entire computational task, freeing the host PC's CPU for I/O tasks, such as filesystem maintenance and operator interface.

Even though the Z8000 was executing a high level language it did not execute a copy of any *Operating System*, merely passing parameters to the host MSDOS in such a way as to create and use files that were identical to those created by a program running on the host PC.

This was the first practical demonstration of a user- transparent coprocessing environment (although the 'Baby Blue' coprocessor from Microlog had previously used a Z80 CPU to run CP/M 80 in an MSDOS host).

EARLY 32 BIT COMPUTING COPROCESSORS

In 1985 this author described a computing coprocessor system based on the National Semiconductor 32032 CPU [3].

This was a significant advance over the Trump card in several areas. Multiple languages had been developed, and these languages had been ported from a UNIX environment offering PC portability for applications previously targeted for UNIX. Interaction between the coprocessor software and the PC operating system had been significantly enhanced so that any task that could be performed by software on the host CPU could also be performed by software running on the 32032.

By middle 1986 the technology had advanced to the point where the author's 68020 based coprocessor system [4] consistently provided the same computational performance as a VAX 11/780 over a wide range of scientific applications.

RISC MICROCOMPUTERS

At this point in time scientific micro- computing performance standards are defined by systems based on RISC processors. Although RISC technology in and of itself is not necessarily significantly superior to the CISC, a number of factors have combined to position RISC systems at higher performance levels than thier CISC counterparts.

Firstly, architects of RISC computer chips have started their designs with a clean slate. There were no previous generations of software with which compatibility had to be maintained, no preconceived notions of hardware interface technology to be maintained between generations of processors.

Additionally, it is easier to design a memory system for a modern RISC chip, because the software can maintain closer control of the internal pipeline status than with its CISC counterpart, making it possible to use load scheduling and other optimisation techniques to hide the wait states introduced when interfacing to real-world memory systems.

In terms of performance the two leading microcomputer families are those based on the MIPS R3000 chipset and those based on the AMD 29000. The Motorola 88000 family is also a contender, but only for smaller problem sizes. The reasons for the differences in performance between the RISC families are surprising, and worth further examination.

The 88100 is a very fast CPU, with an integrated FPU that makes it arguably the fastest single precision floating point engine available. It produces its best results on small programs, such as the whetstone benchmark, that fit within its small 16K data cache. It does not perform well with large problems primarily because of a minor design flaw in the 88200 cache controller. When there is a cache miss *the whole 4 word cache line* must be refilled before execution can continue. Thus for programs whose data has good locality of reference the 88100/88200 chipset turns in superlative performance (its whetstone figures are essentially equivalent to the CRAY-1S), but when sparse matrices are being manipulated it is usual that at least 2 words of each cache line are never used, and the overhead associated with having to wait for them to be filled on each cache miss, is crippling. For example, when Guass- Jordan reduction of a 200x200 matrix is attempted (the Argonne Labs LINPACK benchmark suite) its

CRAY- relative performance has dropped to only 10% of what it achieves with Whetstones.

The MIPS R3000 family very closely integrates the Floating Point controller with the CPU, so that even though they are separate chips there is very little overhead passing data between them. In addition, the MIPS computer system is closely designed around its cache memory. Software post optimisers re- arrange the machine language instructions so as to maximise the 'hits' on the cache memory. The combined effect of this optimiser, the CPU/FPU integration, the relatively large cache memories, and the investment in system software has made MIPS based systems the performance leaders for the past few years.

The AMD 29000 achieves its high performance because of an innovative memory architecture: externally Harvard. Although many of the current generation of CPUs have separated data and instruction paths internally, only the 29000 designers chose to support separated paths between the chip and main memory.

This has many ramifications. Firstly, it is possible to design a 29000 memory system specifically intended for instruction access, and one that only has to handle data memory references. Instruction accesses are typically sequential (between branches) and new architectures such as 'Interleaved Burst Mode' can give cache memory performance with lower cost and system complexity [5][6]. Data memory acesses, however, are mostly random in nature, and the design topology of a data RAM subsystem is quite different. Because the code stream continues to flow to a 29000 even when the data pipeline is waiting for a response from slow external memory, it is still possible to obtain very high performance without the cost and complexity of a data cache [6].

29000 PC AND MACINTOSH COPROCESSORS

During 1988 and 1989 YARC Systems released a series of AMD 29000 RISC based computing coprocessors for PCs.

The IBM PC based versions transparently use the host MSDOS operating system to provide a scientific computing platform that, although it is operated identically to any other MSDOS application, actually performs all the computation on the RISC processor.

The Macintosh II based systems also operate transparently to the user. To bring up the FORTRAN compiler, for instance, one merely double-clicks on its ICON, exactly as if operating native Macintosh software. Only the execution speed reveals the fact that the code is actually running on the RISC processor. All files are manipulated by pull-down menus, in the normal manner.

Both these systems use tools (such as FORTRAN and C compilers) that run native on the 29000 and allow total portability with most applications designed for the UNIX environment. The operating system on each coprocessor is designed for optimal single tasking, single user, applications, and takes less than 16K of the 8 megabytes of available RAM on the coprocessor.

Both coprocessors support Bus Master operation, giving direct access to bulk RAM and peripherals on the bus. This can speed video drawing times by an order of magnitude.

MULTITASKING

The DesqView operating system augments an MSDOS PC to allow multi- tasking. Performance is especially good on 80386 based hosts. A task can be dispatched to the computing coprocessor, and switched to the background, freeing the host for tasks such as editing, printing and other tasks.

The Macintosh MultiFinder operating system is quite adequate to allow editing of source files and other host related tasks whilst a coprocessor is crunching as a background task.

Multiple Computing Coprocessors can be operated in the one PC (using DesqView) or in a Macintosh (using Multifinder). Compute intensive jobs can be dispatched to each of these processors, increasing the number of jobs that can be computed at one time and further increasing productivity.

PROBLEM AREAS

Personal computers are designed primarily for single user, single tasking applications. They usually have a single disk subsystem, and MSDOS cannot even perform overlapped seeks. By comparison with the disk units on a mainframe or supercomputer such technology is very crude. Neverthless good results can be obtained provided disk I/O is kept to a minimum.

Many application programs written for Mainframes and Supercomputers have been designed to per-

form a lot of disk I/O so that they will operate with only a few megabytes of RAM memory, as swap space is often a critical factor in determining the priority one's job will obtain when submitted to the supercomputing center. When using a coprocessor it is best to keep all intermediate results in local memory, if possible, and not write intermediate matrices out to files. If the 8 megabytes of high speed memory is inadequate to hold all the data then bulk RAM storage on the bus should be used. This is easier to do in a Macintosh than in a PC. This is fortunate, as the Macintosh disk I/O system is considerably slower than that of the IBM style PC.

VECTORIZABLE APPLICATIONS

Microcomputer software currently has no capability to recognize inherent vectorisable structures in programs and despatch them to the appropriate execution units. Now that more concurrency is available within chips such as the 29050, 88100 and the 80860 it is becoming imperative that software be developed to allow the microcomputers to catch up with the capabilities of the supercomputers in this area.

SUMMARY

The scalar computing performance of a desktop Personal Computer equipped with a computing coprocessor system is essentially the same as that of typical mainframes and approaches that of supercomputers. Professor Stuart Savage [7] reports that the 29000 based PC coprocessor takes an hour to run one of his applications, about the same time as his IBM 3090. The same problem takes 4 hours on a SPARC (non Harvard) based computing system and 23 hours on an 80386/80387 PC without coprocessor assistance. This example is not given for the purpose of 'proving' that any particular technology is better than any other, (obviously the performance increments will be critically dependant on the application) but to illustrate that coprocessing technology has come of age, and is unlikely to disappear as a viable computing solution.

Computing Coprocessors offer the highest levels of performance with simplicity of operation, bringing supercomputer power to the desktop without adding complexity to the well accepted PC software environments.

REFERENCES

1. A.G.W.Cameron, "The Number-Crunching Revolution", *Computers in Science*, Vol.1, No.3, p.4, 1987
2. S.Ciarcia, "Speed up your PC with 16 bit Processing Power", *BYTE*, Vol 9, No 5, May 1984, pp 40-55
3. Trevor G Marshall et al., "The DSI-32 Coprocessor Board", *BYTE*, August 1985, p 120
4. Trevor Marshall et al., "The Definicon 68020 Coprocessor", *BYTE*, July 1986, p 120
5. T.G.Marshall, "Real World RISCs", *BYTE*, May 1988, p 263
6. T.Marshall, "Standard DRAMs get 15 MIPS from RISC", *Electron. Syst. Des. Mag.*, Vol 18, No 12, December 1988, pp 52-56.
7. Professor Stuart B Savage, M^{c}Gill University, Private Communication.