MVP MICROCODED CPU/16

HISTORY

Glen B. Haydon
Haydon Enterprises
Box 429 Route 2
La Honda, CA 94020

Phil Koopman Jr.
20 Cattail Lane
No. Kingstown, RI 02852

INTRODUCTON

The MVP-MICROCODED CPU/16 design resembles that conceived in the ALCOR project in developing an ALGOL translator utilizing multiple hardware stacks combined with the powerful techniques of a freely microcodable processor implemented in discrete components. In the present form of the CPU/16 design, the user is free to structure the processor according to application requirements for optimal efficiency.

HISTORY

The ALCOR project was led by Samelson and Bauer during the 1950s. Its goal was to provide a direct method for translation of ALGOL. They conceived of a hardware design with two "cellars", one was to hold operational characters and the other to hold numbers. In modern terminology these would be called stacks. They are hardware storage devices based on a last in first out scheme. A block diagram of their concept, Figure 1, has been included in several papers.

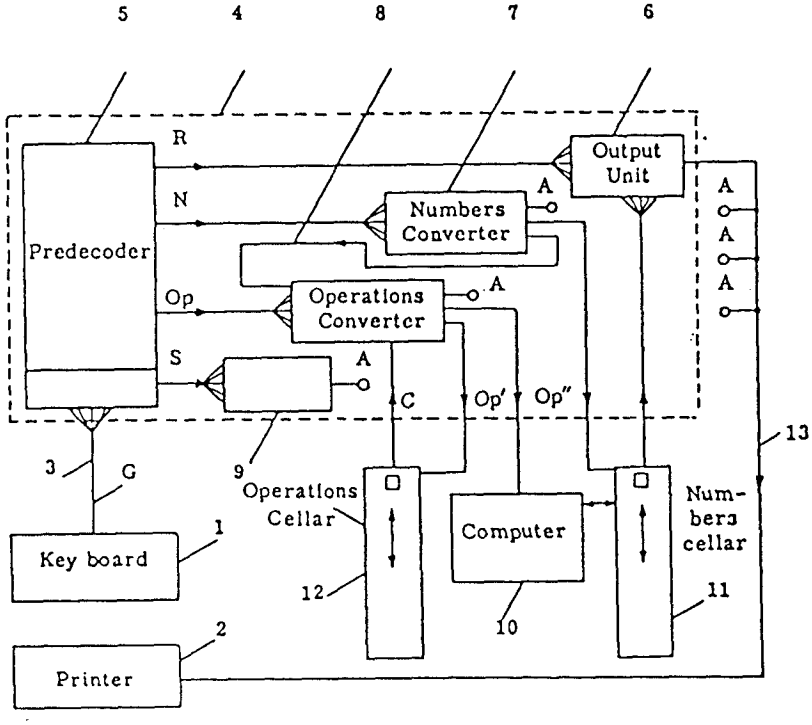

FIG. 1.

It appears that a hardware implementation of the ALCOR design was never completed. Computer processor designs took another direction. A stack operation was often included but with the stack memory mapped into a portion of the system's memory. Such stacks are usually used to store the return location for subroutine calls and sometimes to preserve other values.

In the late 1960s, Charles Moore designed a scheme of programming also using two stacks. One stack contained the return addresses of successive subroutine calls and the other stored interim data values during computation. Unfortunately, he could find no hardware designed to fulfill his needs and resorted to emulating such a processor. Such emulations are available on many systems today. They are known as a FORTH kernel.

A second consideration in the CPU/16 design is similar to that adopted by Seymore Cray. In his Cray computer design, he used discrete components. His claim was that it was the only way to get speed. Of course the Cray design utilized many other features but the basic idea was that faster processors could be implemented utilizing simple components.

The Cray computers used a Data General Eclipse as a host giving access to the outside world. In a similar manner the CPU/16 uses an IBM compatible as a host providing I/O to the outside world. With only minor changes, the CPU/16 could use any common microcomputer.

Finally, the concept of microcoding a simple processor has been utilized in many different ways. Specific microcodable devices have been designed and are commercially available. Examination of these devices suggested that we could design a simple processor with discrete components which could be microcoded and provide even greater versatility and speed.

RESULTS

The end result of these ideas is presently operational and available in kit form. It provides an ideal tool for exploring the potential of the design and as a learning medium. Unfortunately, many people are reluctant to undertake a wire wrapping exercise requiring 30 to 40 hours. However, utilizing the single stepping capabilities from the host, any portion of the processor can be exercised step by step. There is no better way to learn at first hand the capabilities of a multistack microcodable processor.

WORK IN PROGRESS

Now that the CPU/16 design of the kit has stabilized, the next step is to produce that design on printed circuit boards to be placed in the IBM PC compatible. A problem with such a board is that it is no longer simple to change a wire corresponding to a bit in the microcode. The printed circuit board is no longer the experimental tool at the hardware level.

The CPU/16 design is currently being laid out and wire wrapped on a pair of S-100 system boards. It will run with an implementation of MVP-FORTH on an S-100 bus system. There is also interest in implementing the CPU/16 design utilizing the Apple II series of computers as the host.

NEXT GENERATION

Where to from here? The 16-bit bus of the present design is limited to 16-bits of address space. By addressing on word boundaries, the system can address 128K bytes. But without some form of bank switching, virtual memory or some other technique, the size of memory is limited. Intel has overcome this limitation by utilizing several base segments from which addresses can be indexed. This is in essence a form of bank switching although it has been efficiently implemented.

The next bus size to consider is 32-bits wide. Intermediate numbers of address bits can be used but efficiency dictates the next size limitation at twice the size of the 16-bit limit. Using a 32-bit bus in a manner analogous to the current 16-bit design and adding a number of enhancements, a significant further increase in performance is anticipated. Also a billion 32-bit words (4-giga-bytes) of contiguous memory could be addressed without some form of bank switching. Part of the engineering prototype for a CPU/32 based on these considerations is already completed and functional.

The CPU/16 kit is an ideal hardware system with which to study other architectures. For example, the design is clearly not a RISC machine as currently described. However, by addressing items in the dedicated stack memory with optimized microcode, it would be possible to treat stack RAM as an array of registers and emulate a RISC design. In such an implementation, the RISC architecture could be thought of as a subset of the capabilities of the CPU/32 processor.

LANGUAGES

The MVP CPU design lends itself to the efficient implementation of a wide variety of high level languages. For example Smalltalk-80 uses approximately 100 primitives each of which could be implemented in microcode. The design would be ideal for implementation of a p-code machine.

FORTH has been used in the initial phases of this work. FORTH is, after all, an emulation of the hardware design. The language has the advantage of ease of interactive programming and access to all hardware components. The diagnostic suite, micro-assembler and cross-compiler were easily developed with a minimum of effort. The language also provides a versatile facility for programming many applications.

However, with the desirability of making the system compatible with other existing programs, it would be desirable to have a common operating system available. One route to a popular operating system would be to first implement the C language. Already, one group is working to implement Small C. With a full implementation of C, the entire UNIX system could be added.

With the versatility of a microcodable processor, the development of new languages tailored to specific applications becomes more reasonable. The languages of LISP and PROLOG are just a beginning in the field of artificial intelligence and they have been implemented in FORTH. It should be relatively easy to move such implementations to the newly designed processor.

CONCLUSIONS

The MVP CPU design provides flexibility in designing and using hardware to solve many application problems. In addition, many high level languages could be implemented on such a system with excellent efficiency. Initially, FORTH has been chosen as the as the host and processor language. As such, the system complements a variety of other commercially available implementations of FORTH in hardware. The MVP CPU series of products provides flexibility for experimentation and tailoring the processor to specific application and a tool for teaching and testing a variety of hardware processor designs.

The kit would make an ideal starting point for a comprehensive computer science course sequence. Such a sequence might start with the building of the kit as a microcodable processor. That might be followed with the writing of the software for a compiler and an operating system. The series might conclude with a significant application utilizing the tools which were developed.

The fundamental philosophy has been to examine programming requirements of the application at hand, and design the hardware accordingly. It is a shame to have the hardware limitations drive the programming and limit the solution of the application. The present design is a stage in the evolution of hardware to solve problems. Perhaps more than two stacks would be desirable in some applications. Once a design is found for a specific application, the next step would be to cast that design in silicon. But don't get the cart before the horse.

BIBLIOGRAPHY

Bauer. Fredrich l., Between Zuse and Rutishauser- The Early Development of Digital Computing in Central Europe., in A History of Computing in the Twentieth Century, N. Metropolis, J. Howlet, and Gain-Carlo Rota, Editors, Academic Press 1980.

Note: This volume is a treasury of historical ideas which are unknown to many workers in various branches of computer science today.