KFKI-1978-100

G. AMBRÓZY
J. MISKOLCZI
F. VAJDA

SMALL COMPUTER BUILT WITH INTEL 3000 BIT-SLICED MICROPROCESSOR SYSTEM

Hungarian Academy of Sciences

CENTRAL
RESEARCH
INSTITUTE FOR
PHYSICS

BUDAPEST

KFKI-1978-100

SMALL COMPUTER BUILT WITH INTEL 3000 BIT-SLICED MICROPROCESSOR SYSTEM

G. Ambrózy, J. Miskolczi, F. Vajda

Central Research Institute for Physics
H-1525. Budapest, P.O.B.49. Hungary

The paper was delivered in the Conference MIMI '78 in Zürich, June, 1978.

HU ISSN 0368 5330
ISBN 963 371 492 3

ABSTRACT

A microprogrammable microprocessor system enabling the emulation of central processing units of small computers is described. A processor with a word length of 16 bits was developed whose instruction set can be wholly defined by the controlling microprogram. As a first step the emulation of the PDP-11/40 /Digital Equipment Corporation/ was realized. The CPU with an interface generating the signals of the UNIBUS forms an operable small computer. Here the hardware structure and the controlling microprogram of the system are presented as well as the evaluation of the Intel 3000 microprocessor family in implementation of an emulator.

АННОТАЦИЯ

В данной работе описывается система микропрограммируемого микропроцессора, позволяющего осуществлять эмуляцию центральных блоков малых ЭВМ. Система команд 16-битного центрального блока полностью определяется управляющей микропрограммой. В качестве первого шага нами проведено эмулирование малой ЭВМ типа Digital Equipment Corporation PDP-11/40. В результате нами получена работоспособная ЭВМ, состоящая из центрального блока и адаптера, вырабатывающего сигналы UNIBUS. Описывается hardwer построение системы и управляющая микропрограмма, а также проводится оценка семейства микропроцессоров Intel 3000, используемого в качестве эмулятора.

KIVONAT

A közlemény olyan mikroprogramozható mikroprocesszoros rendszert ismertet, amely lehetővé teszi kisszámitógépek központi egységeinek az emulálását. A 16 bites központi egység utasitásrendszerét teljes mértékben a vezérlő mikroprogram határozza meg. Első lépésként a Digital Equipment Corporation PDP-11/40 kisszámitógépének az emulálását végeztük el. A központi egység az UNIBUS jeleit előállitó illesztővel kiegészitve müködőképes számitógépet alkot. Bemutatjuk a rendszer hardware felépitését és a vezérlő mikroprogramot, valamint értékeljük az Intel 3000 mikroprocesszor családot emulátor alkalmazásban.

ABSTRACT

A microprogrammable microprocessor system enabling the emulation of central processing units (CPUs) of small computers is described. A processor with a word length of 16 bits was developed whose instruction set can be wholly defined by the controlling microprogram.

As a first step the emulation of the PDP 11/40 (Digital Equipment Corporation) was realized. The CPU was completed by an interface generating the signals of the UNIBUS+ and by a memory of 16 Kwords; in this way an operable small computer came into being.

The CPU consists of 8 Intel 3002 CPEs (Central Processing Elements) with the 3003 Look Ahead Carry Generator, the Microprogram Control Unit (Intel 3001 MCU), the control memory, a scratch-pad register block containing the general purpose registers of the PDP-11/40 and two Field-Programmable Logical Arrays (FPLA) performing instruction- and addressing mode de-

+ UNIBUS is the registered trade mark of DEC

coding.

The beginning addresses of the microprogram routines executing the macroinstructions are determined by the instruction decoding FPLA-1. The addressing mode decoding of the instructions with single or double operands is performed by the FPLA-2 followed by the actual execution of the instructions. The microprogram realizing the basic instruction set is stored in a 512 word control memory module; the optional extended instruction set and floating point instructions are to be found in a second 512 word module of the 1 Kword capacity control memory.

For such an emulation task the applicability of the microprocessor family chosen can be examined. A study of Intel's 3001 MCU and 3002 CPE shows their disadvantages as well as their advantages.

Emulation was greatly facilitated by a PDP-8 based development system enabling hardware tests and symbolic writing and assembling of the microprograms as well as their verification.

1. INTRODUCTION

Although the concept of microprogramming is well over twenty years old the techniques have become widely used only in the last few years. Semiconductor memory technology made the first major impact on the development of microprogramming; this was followed by the LSI bit-sliced microprocessor chip sets [1],

[2], [3] - the second important step. Thus the state of semiconductor technology and the large number of applications involved in microprogramming greatly affected emulation too. It is for this reason that it might be better to change the original definition of Tucker [4] i.e. emulation is "a combined software and hardware approach to the problem of making a computer run programs alien instruction code" to another one [5] stressing the role of microprogramming. Nowadays, the view that microprogramming should be recognized as an approach primarily applied to the task of emulation is also becoming much more widespread.

With this definition an emulator could be defined as a special interpreter which dynamically maps each statement (instruction) at a point to be executed into an execution sequence of statements in another environment. In a microprogrammable machine this environment would be microcode instructions. A research period of some years in the field of emulation [6], [7] and some succesful host processor designs oriented for emulation purposes [8],[9] were just the beginning. Recently, supported by the background of these research and design works a new category of computers became technically feasible and started to appear on the market as the general microprogrammable machine, the so called universal (general purpose) host [10] , [11]. The members of this machine category are usually medium and large systems having large architecture (with flexible bus structure and often two levels of microprogramming) and can run several emulators at the same time.

We had quite a different goal when the main points of our project were settled. As target machines, only a limited set of small processors with a word length of 16 bits were selected. At the same time we wanted to confine our host machine to the microprocessor range with regard both to system size and cost. Obviously this consideration resulted in a number of limitations. Only a part of the host could really be universal and the host would need to be supported by a target bus (input-output); this specific part was eventually realized as a so called "personality board".

The following hardware design viewpoints were considered:

- the hardware system should be general enough to enable the emulation of various central processing units (CPUs),
- the execution of the tasks requiring many microsteps should be facilitated by appropriate hardware arrangement,
- the microprogram word should not be too long,
- further extension of the system should be possible,
- control memory extension with a RAM memory of arbitrary access time (e.g. minicomputer based development system) should be possible.

To provide a truly universal "core" for this micro-host machine a generalized flexible decoding structure (realized by independent FPLAs decoding instructions and addressing modes of single or double operands), a sufficient number of registers (en-

abling efficient mapping for "visible" and microregisters), a flexible microinstruction structure (with predefined compulsory fields for the micro-host and free form empty part for the personality board control) and an efficient interface (for the different external bus boards) were defined. The system contains extra hardware for microcode development.

As a test of our "soft" micromachine's capability of emulating a range of small and mini computers, initially the emulation of PDP-11/40 was realized.

2. HARDWARE STRUCTURE OF THE SYSTEM

Our goal was to build a general purpose microprogrammable architecture of contemporary integrated circuits suitable for emulation of the CPUs of several small somputers with similar performance; this would then give the opportunity for developing hardware-software tools for microprogramming.

Fig. 1 shows the two main functional parts of the emulator machine, i.e. the CPU (EM) and the input/output unit (EMIO-11).

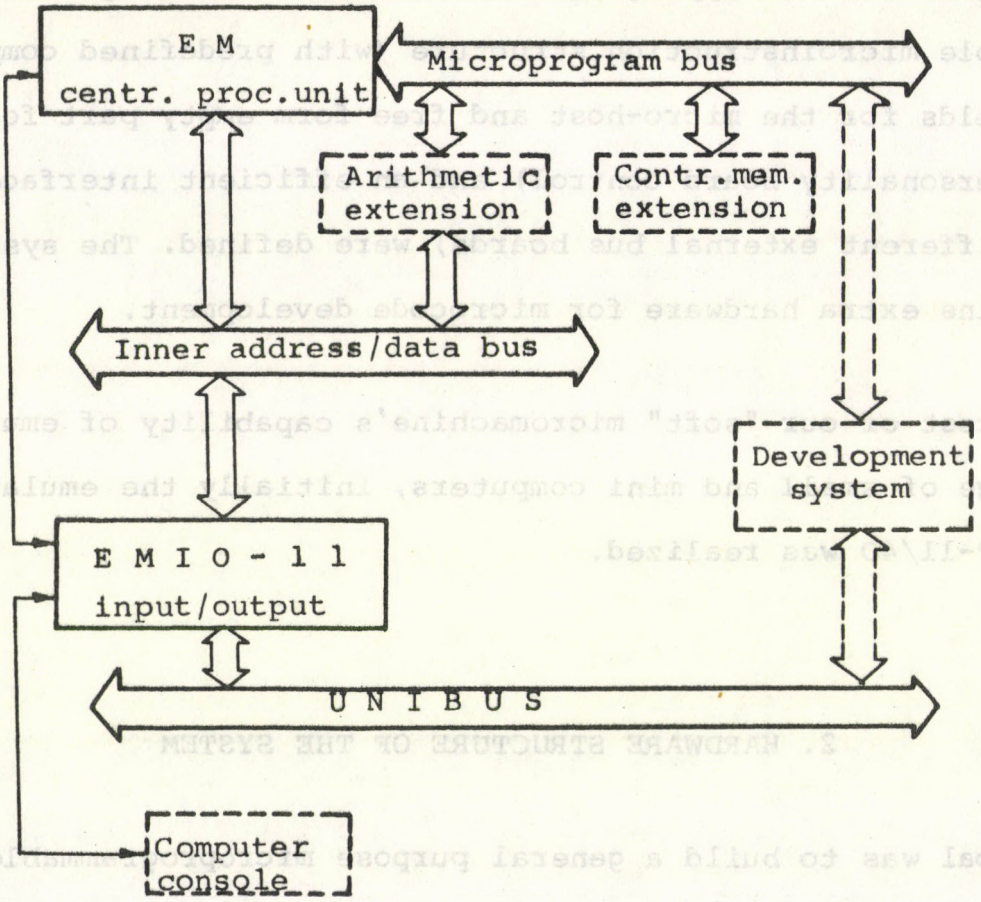


Figure 1: Functional blockdiagram of the system

The CPU shown in more detail in Fig. 2 was built with micro-programmable microprocessor elements of Intel's Series 3000. It consists of 8 Central Processing Elements (CPEs), a Micro-program Control Unit (MCU) and a Look Ahead Carry Generator (LCG). In addition it contains

- a Register Block of 16 registers each with a length of 16 bits,
- an Instruction Register,
- FPLAs for decoding purposes,

- a Condition Register facilitating the microprogram branches,
- a PROM Control Memory Block containing 2x512 words of max. 64-bit capacity.

The CPE, the Register Block, the Instruction Register and the Condition Register are connected to a microprogram controlled inner data bus. The Register Block functions as the general purpose registers of the emulated small computer, the inner registers of the CPE are working registers.

The address output of the MCU and the output of the microprogram memory constitute a microprogram bus (supplied by a few control signals) which enables the connection of a control memory of arbitrary access time. This control memory may be

- RAM memory of a minicomputer based development system [12] facilitating the verification of the microprogram routines of single macroinstructions.
- EPROM memory containing the microprogram of the complete instruction set. This memory helps the examination of the system by test programs.
- Fast PROM or RAM memory as the extension of the control memory of 2x512 words.

This architecture of the microprogram bus and the inner data bus permits the extension of the system (e.g. with further processing circuits such as fast floating point processor);

the data transfer between the EM and the extension unit takes place on the inner data bus, and is controlled by the microprogram (see Fig. 1).

The instruction set and handling of the input/output peripherals (i.e. the I/O bus) can be considered the main parts of a small computer. In this emulation (as shown in Fig. 1), the hardware of the system is divided into two parts: the EM unit and the EMIO-11 unit. This latter is also microprogram controlled and contains the drivers-receivers and timing circuits of the UNIBUS only. In this way it is possible to emulate several small computers with similar instruction sets by means of the EM unit.

3. MICROPROGRAM REALIZING THE INSTRUCTION SET OF THE TARGET SMALL COMPUTER

3.1. Instruction set of PDP-11/40

We have no intention of detailing the instructions of the PDP-11 small computer family but for a better understanding of the structure of the microprogram it is useful to have some knowledge of the instruction set of the machine to be emulated[13].

The basic instructions can be classified into three main types:

- arithmetic instructions,
- program control instructions,

- miscellaneous instructions.

Within these classifications, they can be grouped as follows:

Arithmetic instructions

- single operand instructions (e.g. INC A)
- double operand instructions (e.g. BIC A,B)

Program control instructions

- branches (e.g. BEQ .-4)
- jump and subroutine calls (e.g. JSR R5,A)
- traps and interrupt returns (e.g. IOT)

Miscellaneous instructions

- condition code operators (e.g. CLN)
- HALT, RESET and WAIT instructions.

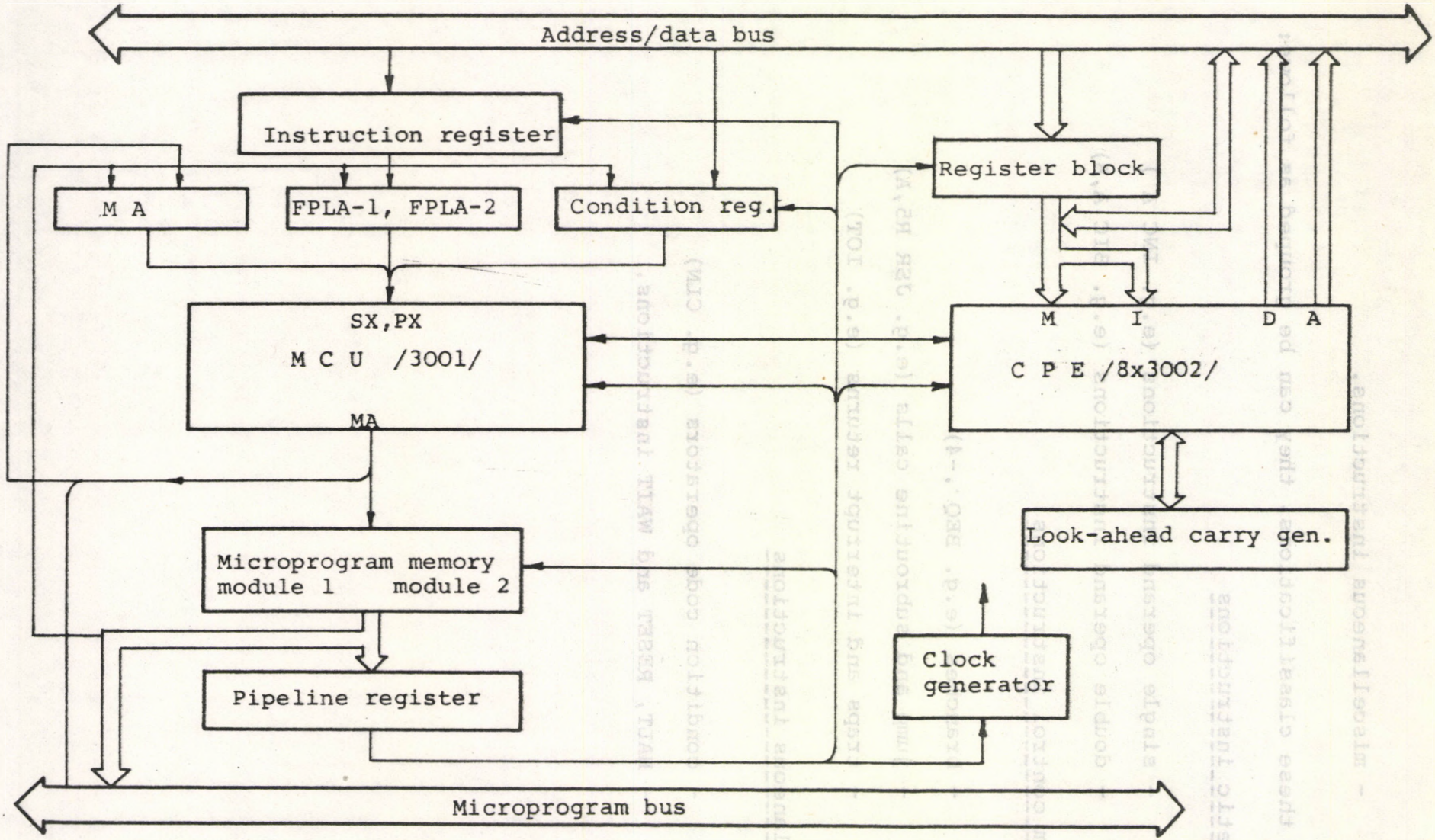


Figure 2: Detailed blockdiagram of the emulator

The PDP-11/40 also has optional instructions which are arithmetic extension instructions and two special instructions for the Memory Management option. In our system these options were realized in a wholly microprogrammed manner.

3.2. Organization of the microprogram

The flowchart of the microprogram is given in Fig. 3. The events shown above the dotted line can be considered special in a certain sense, viz. they are exceptional with regard to the "normal" cases. These events are the following tests of the possible states (according to the PDP-11/40):

- does trace-trap exist?
- does an interrupt request exist?
- is there a console HALT?
- is there an odd PC error?
- is there a stack limit violation?

If any of the above states exists a particular microroutine starts working and executes the necessary steps. Since these tests have to be done in each macroinstruction, this task increases the average time of the instruction execution. In order to avoid this slowdown we applied a special "switch" to "short-circuit" some of these tests (e.g. stack limit violation). In most cases of verified and operable program running it is certain that neither odd PC nor stack limit violation occur so the testing microroutines can be suppressed and in this way the actual speed of the machine increases. It can be said that the

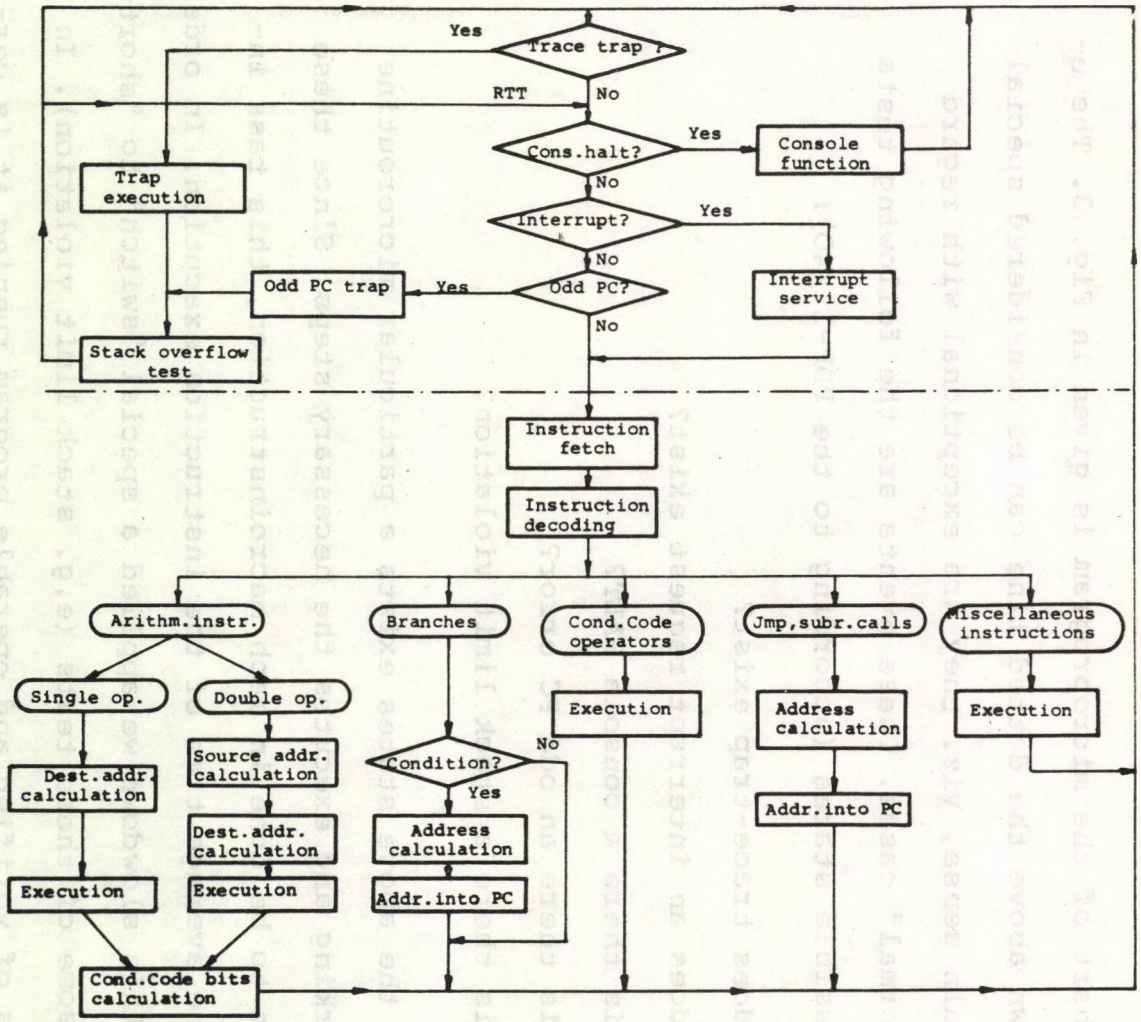


Figure 3: Organization of the microprogram

"switch" changes the state of the emulator from class A to class B [14].

The lower part of the flowchart shows the fetch, decoding and execution of the instructions. Instead of a microprogrammed solution the instruction decoding is performed by an FPLA circuit. This circuit determines the proper microroutine addresses of the different macroinstructions; this solution saves the generality and flexibility of the emulator since any microroutine address arrangement can be obtained simply by changing the FPLA circuit.

The addressing mode decoding of the arithmetic instruction operands is also performed by means of another FPLA circuit. Having fetched the operand or operands the actual execution of the instruction follows, and after the calculation and setting the condition code bits (according to the result) the microprogram gets to the beginning of the cycle again.

In the case of instructions without operands the instruction decoding is immediately followed by the actual execution.

As an example, in Fig. 4 the flowchart of a double operand instruction is shown in detail. In the execution phase the possibility of the Program Status Word (PSW) as destination represents an interesting problem. The T bit (Trace Trap) of the PSW is a non-writable, non-erasable bit, hence the new destination value must be masked before writing into the

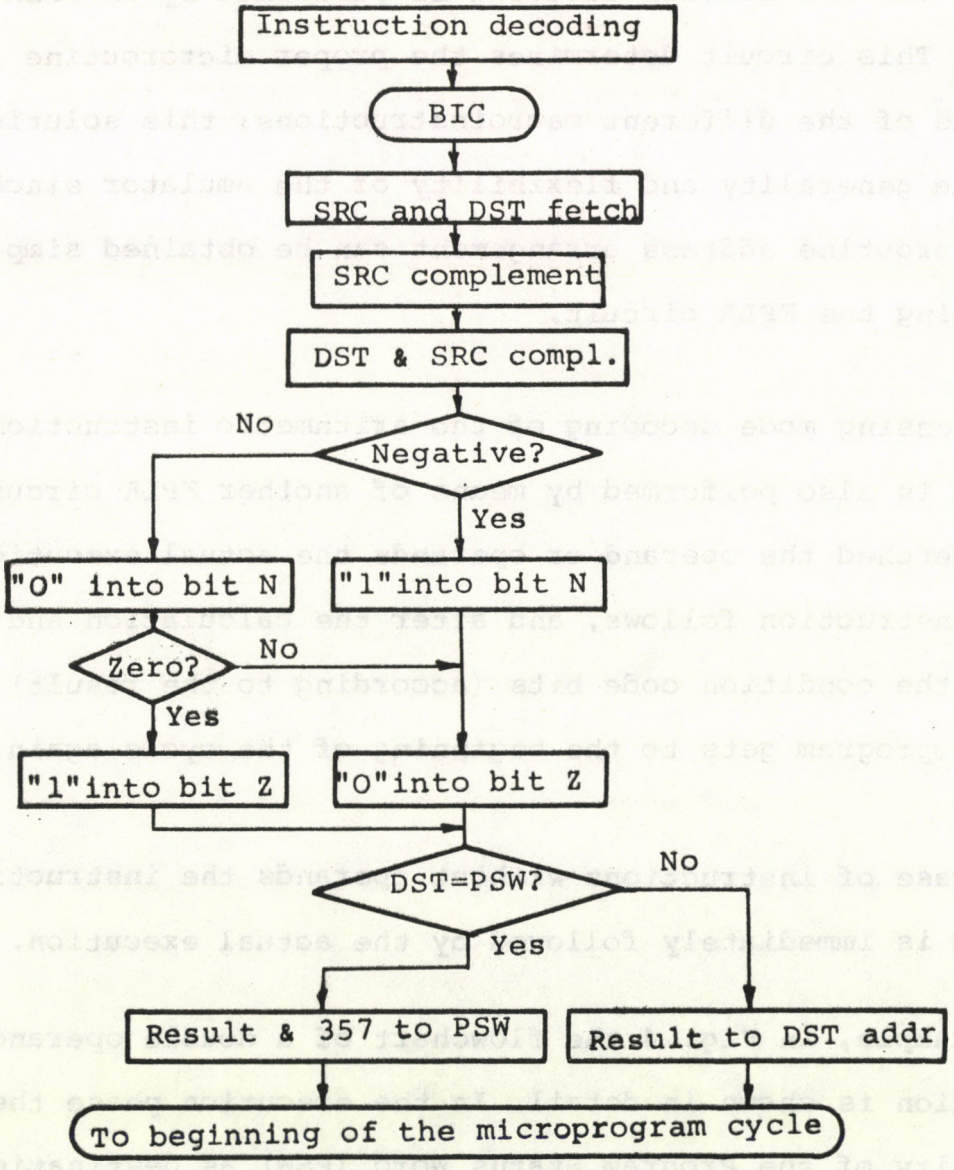


Figure 4: Double op. arithm. instr. example

register dedicated to the PSW. This possibility may occur in the case of every instruction with destination operand (except CMP, CMPB, BIT, BITB, TST and TSTB) so its test increases the execution time of this kind of instruction. Here however, the short-circuiting of this test is not possible.

In the group of program control instructions (jump and subroutine calls), the adressing mode decoding and address calculation is performed by the same microroutine as in case of the arithmetic instructions. The difference between the two interrupt exit instructions (RTI and RTT) can be seen on the microprogram flowhart: after the execution of RTT in order to disable the trace trap the microprogram does not get to the beginning of the cycle.

The options are realized in another 512 word microprogram module; changing between the modules is controlled by special bits of the microinstructions.

4. EVALUATION OF INTEL 3000 BIT-SLICED SYSTEM IN IMPLEMENTATION OF AN EMULATOR

Bearing in mind how bit-sliced microprocessors will be used in the design of an emulator where certain performance criteria must be met, any comparison of the properties of the different microprocessor systems available on the market is extremely difficult [15]. We selected the Intel 3000 family at an early stage and having developed its well defined application environment [12] found it necessary to keep to this; thus our only choice is to accept and later on access this family's inherent disadvantages as well as its advantages based on our experiences in the design of the emulator. Here, we restrict ourselves to discussing the two major components of the family, the 3001 MCU and the 3002 CPE, not forgetting the importance of related members performing such functions as carry-look ahead, interrupt control, bus interfacing, etc. in system realization. The three basic features to be evaluated are

- micro cycle time,
- architecture,
- microinstruction set,

though this obviously does not exclude the considerations of cost and ease of programming.

4.1. Advantages and disadvantages of the microsequence controller (3001 MCU)

The 9 bit address width of the MCU was sufficient for micro-

programming the basic emulator. The main drawback resulted from the restricted address modification (jump) capability defined by the column-row address arrangement principle. It led to particular difficulties in mapping microinstructions with conditional jump where the destination address selection is even smaller. In that there were not enough flags to store the target condition code bits, many extra microintructions were required in the microprogram. The use of the register known as PR latch leads to difficulties because it can be loaded by the JPX instructions only. At the same time the multiway branch capability proved to be very effective for the direct interpretation of all decoded functions (e.g. console switch state, addressing mode).

Other major drawbacks were the lack of built-in supporting hardware for loop organization and subroutine handling; these were overcome by extra microinstructions and added hardware, respectively.

4.2. Advantages and disadvantages of the arithmetic-logic unit (3002 CPE)

An unquestionable advantage is the relatively large number of buses available but even so this number is insufficient if one considers their uses by the instructions linked with them. The unit contains sufficient registers for mapping the target machine registers but their application is strictly AC centred.

In other words, since no register-register type instruction exists, it is necessary to save the AC content all the time by many extra microinstructions. For this reason we had no alternative but to use an external register block for target machine general purpose registers.

The separately available macro address (MAR) and data register (AC) are very useful because they can be applied in the same cycle. The sophisticated masking capability providing register bit set, clear and complement is also very effective as well as the nondestructive test realized by conditional clocking. Although the instruction set is rich, there are several important elementary functions missing (e.g. loading external constant into a given register, subtraction, left shift). The result of the operation (e.g. at addition) destroys both the original operands (because it is loaded both in Rn and AC). There are definitely not enough condition bits so the missing ones must be programmed by long microprogram sequences.

The manual is of little help in facilitating one's understanding of the system's disadvantages.

CONCLUSIONS

The general purpose central processing unit built in our institute has proved to be adequate to its first emulation task. Our aim was not to make a small computer which is better than the target machine in every respect but rather to examine the new possibilities of a system realized by means of such

an emulation. Comparison between our microprogrammable microprocessor based CPU and the target computer is possible and is most certainly worthwhile. Two viewpoints of this comparison are of special interest.

- Physical dimensions

The physical dimensions of the microprocessor based CPU provide a considerable advantage compared with the dimensions of the DEC small computer. The number of necessary PC boards is two as opposed to nine of the target machine; the ICs are in a similar ratio.

- Speed relations

This is an important characteristic of the CPU. The speed of the instruction execution of the microprogrammed CPU strongly depends on the number of necessary microsteps, i.e. the complexity of the instruction. The operations which are very fast in hardware solutions may require the greatest number of microsteps in the microprogrammed version. Hence, any comparison between the execution time of each instruction is not too profitable; a better purpose would be served by considering the average execution time 16 . The microprocessor based CPU is slower than the target machine by a factor of 1.5 (some instructions are faster but most of them are slower). For example, the slowdown factor of the emulation with QM-1 presented in [10] is slightly worse (it is 2 compared with our 1.5).

The general purpose CPU has other interesting possibilities besides the emulation of the instruction sets of small computers. Our system can be enlarged with control memory (up to 4 Kword); in this way user-defined instructions can be built in the instruction set, user program modules can be applied, or even interpreters can be written in microcode.

In the development phase of the system (concerning both the hardware and the software - i.e. the microprogram) our work was greatly facilitated by the development system described in [12]. The testing of the microprogram routines was helped by a monitor program; it was accomplished in three consecutive phases: testing of a part of the macroinstructions, testing single macroinstructions and verifying complete macroprograms. For verification of the entire system, original DEC system programs (programs of the papertape system [17]) were applied successfully.

REFERENCES

1. Intel Series 3000 Reference Manual, Intel Corp. 1975.

2. Am 2903 Four-Bit Bipolar Microprocessor Slice, Am 2910 Microprogram Controller. Technical Data, Advanced Micro Devices, Inc.

3. Motorola Microsystem. MC 10800 Series, Motorola Inc. 1977.

4. S.G.Tucker, "Emulation of Large Systems", Comm. ACM., 8, (1965), pp. 753-761.

5. S.M.Fuller et al., "The Effect of Emerging Technology and Emulation Requirements on Microprogramming", IEEE Trans. on Computers C-25 (1976) pp. 1000.1009.

6. R.M.McClure, "A Low Cost Host Processor for Emulation Research",SIGMICRO 74, pp. 34-40.

7. E.A.Torrero, "Bipolar Bit-Slice uPs Shrink the Size and Cost of Minis and Controllers", Electronic Design, May 10, 1976, pp. 34-40.

8. R.George, Jr.Trimble, "Emulation of the IBM System/360 on a Microprogrammable Computer", SIGMICRO 74, pp. 141-150.

9. G.T.Almes et al., "Emulating the NOVA on the PDP-11/40: A Case Study", COMPCON '75 Fall, Digital Papers IEEE New Yersey, USA, 1975. pp. 53-56.

10. J.C. Demco, T.A. Marsland, "An Insight into PDP-11 Emulation", 9th Annual Workshop of Microprogramming (New Orleans, 27-29 Sept. 1976) IEEE New York, 1976. pp. 20-26.

11. R.E.Pollman, "Control Data 480 Series Microprogrammable Computer Family", Computer, Oct, 1977. pp. 45-53.

12. G.Ambrozy, J. Miskolczi, "Development of Microprogrammable Microprocessor System by means of a PDP-8E Minicomputer", Euromicro Journal, Vol.4. No 2., March 1978. pp. 92-103.

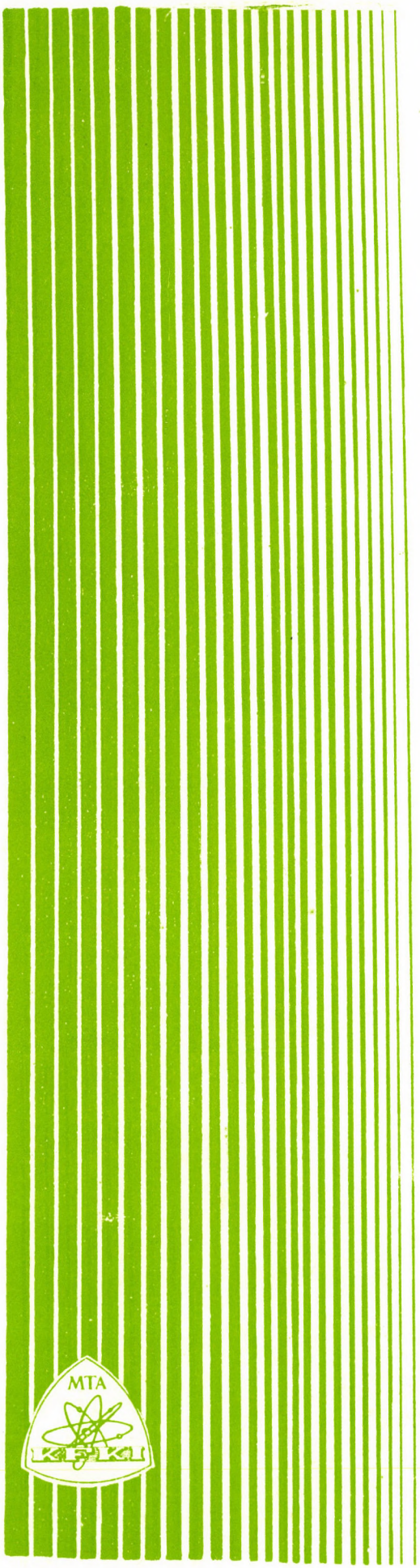
13. PDP-11/04,05,10,35,40,45 Programming Handbook, Digital Equipment Corporation 1975.

14. M.J. Flynn, "Classes of Emulators", Sigmicro Newsletter 8 (1977), pp. 34-35.

15. G.Ambrozy, J.Miskolczi, F.Vajda, "Evaluation of Bit-Slice Microprocessors in Implementation of an Emulator" (to be published).

16.J.N.Frisina, "Metries for Microprogrammed Instruction Sets", Sigmicro Newsletter 8 (1977), pp. 21-23.

17. PDP-11 Paper Tape Software Programming Handbook, June 1975.



Kiadja a Központi Fizikai Kutató Intézet
Felelős kiadó: Sándory Mihály
Szakmai lektor: Szalay Miklós
Nyelvi lektor: Harvey Shenker
Példányszám: 315 Törzsszám: 78-1155
Készült a KFKI sokszorosító üzemében
Budapest, 1978. december hó