INTEROFFICE MEMORANDUM

NO: RC:69:24 DATE: February 24, 1969

SUBJECT: PDP-11 Instruction and Architecture Compromises

TO: FROM: R. Cady

The attached list is a compilation of comments, objections, and suggestions concerning the PDP-11 architecture. I will not try to answer each one individually, but rather to address myself to the major areas of concern and propose changes to the present system which will try to satisfy outstanding problems.

The over-riding criticism appears to be the lack of simplicity of the PDP-11. Everything appears difficult and lack of uniformity and symmetry confuse the user. This group of criticism includes proponents for elimination of variable length instructions and reduction of addressing modes regardless of lost bit efficiency.

Secondary criticism centers around an appearance of lack of good organization and coordination of the total project, especially at the conceptual level and in the areas of hardware-software inter-relationship. This may have some truth, but I rather feel that the present schedule and my own workload have not permitted me to write volumnes of papers on these subjects. It is true that some suggestions (notably 11, 15, 17, 30, 76) require more study and these we plan in depth despite initial looks that appear reasonable.

The following changes are suggested to satisfy many objections and produce a more ordered and easily understood architecture. References in parenthesis are to comments on attached list.

1. Change nomenclature as follows:

   Status & Conditions Code byte = C
   Stack (push/pop pointer = S
   Index Registers = X, Y
   General Registers = M, N (simplicity symmetry #29, 37, 46)

2. Delete external page addressing mode in favor of 256 byte page 0. This addressing modes are:

   A. Immediate and Immediate defered (full)
   B. Indexed and indexed defered with 8 bit offset through S, X, Y.

C. M, N direct (using M, N as data)
D. M, N deferred (using M, N as auto increment)
E. Relative to P (program counter)
F. Page 0 (where second byte is address on 256 byte page 0.

(Simplicity and #7, 8)

3. Delete ANB and substitute with ORW (or word) instruction (#10 objection to deleting ANB now withdrawn).

4. Delete NOP from operate group and add ATS to Add to Register group (which is subset of operate group). Thus Add to group would have ATS, ATX, ATY, ATM, ATN.

(#2 Symmetry)

5. Since JMP immediate performs a NOP define it as such. Thus CØ is NOP.

6. Since JSR immediate is the same as PUP, it is meaningless. Define JSR immediate to do a JSR to zero, one ? thereby introducing a single byte break point instruction (#15, 28, 31, 48, 55, 56).

7. Rename Push/Pop group to include:

POA	PUA
POS	PUS
POM	PUM
PON	PUN
POX	PUX
POY	PUY
POP	PUP
POC	PUC

Retain POS, PUS as these are convenient to transfer S contents in and out.

PUS		Put S on stack
POA		Put S in Accum.
CPW	Mask	Compare with limit mask
JGT		Jump on result

Transfer to different push lists could be by storing new S and then POS.

(#59, 60, Symmetry)

8. Alter XTR instruction to be 2 bytes with function command = Ø. This allows cleaner implementation of extended arithmetic instructions and a better vehicle for extending instructions for sophisticated options (protect, maping, relocate, multi-processor, etc.)

   (#44, case of implementing bigger systems)

Major comments will now be answered in light of above changes.

1, 2, et al The machine is now more ordered, simpler to use, and a little more straightforward.

4. I/O design is in process and will be a little more expensive than present 8 for electronics, a lot less for cable (data break on 8/L costs $260 for 10' cables). Electronics costs are dropping <u>much</u> faster than cable. Net cost to user with 11 is <u>much</u> less.

5. Not a register-register machine.

6. Not so.

7. 256 bytes. X and Y may be used as base register to extend to 778 bytes.

9. Gorden Bell and others feel firmly sign should be extended.

11. What is largest - who can tell now or wants to put upper limit on it.

12. 32 standard levels seem adequate (how many systems exceed this many devices) and mpx will be available.

13. et al Documentation is being improved.

14. No comment.

16. Assembler documentation available.

17. Assembler will do good job.

18. Not so. 16 bit unit stores extra bytes so only takes extra time if <u>data</u> words cue on odd boundries, and this is easily controllable.

19. Why?

21. John Cohen is generating market plan.

22. Why not?

23. We have never <u>built</u> such before. i.e. by this philosophy we should not have sold any 8, 8/S, 8/I, 8/L until TSS/8 was working.

24. True and this needs to be taken into account in the <u>way</u> in which we present machine to customer.

26. In memory at present. Not firm.

31. We must concentrate on not billing the PDP-8 by default.

35. Complexity reduced, bit efficiency retained.

38. What happens when you do this in 8 and over-run the page. Less likely with relatives than absolute.

41. It will under program option.

43. Good point, but wasteful of memory in every indexed operation, majority of which will reach in 256 locations. <u>This is the most difficult compromise</u>. Hardware implementation considerably easier with 8 bit, and speed economy also.

45. Never intended to be multiple AC machine

46. 2 now.

49. They agree it's the only way to go.

51. Console operation is fully documented.

52. See 16

54. Most agree on reasons for numbering.

57. 65K bytes?

61, 62 The NOVA is not the standard of the world.

64. Not a 10 although we haven't excluded it.

66. Optional.

68. Possible.

70. Multiply not defined yet.

71-73 They were.

79. True.

81. Not so.

82. Not so.

83. Don't use it if you don't want it.

84. Too complex an instruction which would increase cost of 11B. Can be done with LDW, PUA sequence (1 byte time/mem cost)

85. Worth considering.

COMMENTS

Bruce Delagi, 2-10-69

1. Conceptual simplicity
2. Too many exceptions to simmetry (no ATK, M & X deferred work differently)
3. Too many irregualr cares requiring unique gating
4. Cost of I/O to high (cheap date channel doesn't justify expense I/O)
5. Handle byte manipulation on register-register basis
6. Too tailored for I/O controller, desk calc. markets

Teradyne

7. Much too small page Ø
8. Sacrificed much needed addressing

PDP-11 Design Review, 1st session, 2-17-69

9. LDB should clear upper half accum. to zero
10. Add OR to instruction. Do not delete ADB, ANB
11. Should design largest system first

Engineering Committee, 2-13-69

12. 32 Priority levels inadequate
13. Bus dialogue and description not clear
14. Cannot be readily programmed
15. How do you put in breakpoint for DDT
16. Inadequate Assembler documentation and no design review records.
17. No apparent way to assemble to get full list efficiency.
18. Execution time dependent upon word boundries
19. Cannot make recursive patches
20. No centralized soft/hardware control
21. No market plan and strategy document to indicate tangent goals
22. Cannot interface 32-100 TTY as cheap as 680-I
23. Should build examples of most complex structure devices before selling any machines.

Gorden Bell, 2-16-69

24. First impression not too good, but it grows on you

25. Very unique bus structure and concept. System summetry and concept good.

26. How is parity handled? Can (should) processor handle it on the bus.

27. Memory mapping of I/O devices good.

28. Desire a single byte JSR for breakpoint

29. "Push" terminology should be changed to "stack"

30. Consider hos one implements memory mapping for Exec/user mode

31. The 8 will continue to live if we want it to. We should design a cheaper 8/L using new technology.

32. We should study part introductions of new products to be able to build a market model and predict response.

Ad Hoc Architecture Design Review, 2-19-69

33. Too complicated for general purpose users

34. Should be designed more along X structure using subsets for programming which can be easily extended for larger implementations.

35. Motivation is bit efficiency but complexity is not necessary to get 95 percent of efficiency.

36. Dislike 8/16 bit instructions

37. Nomenclature changes would help clarify

38. What if you add to a program and now some relative address doesn't reach

39. 8 bit breakpoint necessary

40. Techniques for organizing large programs must be investigated now.

41. Teletype should clear flag when reading

42. Some arithmetic coding should be done.

43. Index registers should use full 16 bit displacement

44. Make XTR 2 byte instruction.

Uncertainties Over A Certain DCM

Rick Merrill

45. Only 1 AC

46. Only 1 index register, X

47. X cannot be used as a count.

48. No 8-bit break-point provided yet.

49. Applications programmers do not like hexadecimals.

50. No one person is enthusiastic over the operation codes.

51. The control console operation is unexplored. [redacted]

52. A description of the assembler has not yet been forthcoming.

53. Both a new programming language and a new editor language are being created.

54. Bits are numbered backwards.

55. Multi-byte debugging is a nasty business without a symbolic DDT.

56. Debugging tools are not being designed.

57. The computer has too small a memory for good symbolic debugging.

[redacted]

58. No concern has been shown for the OEM or TEACHER who prefers to have only one way of doing an operation, not a prolification of choices.

59. There is no use at all for PUY, an instruction that pushes the push-down-list pointer onto the push-down-list!

60. Any use of POY would be obscure indeed.

61. The memory size, and the relative addressing range is half that of the NOVA.

62. Page zero is one quarter that of the NOVA.

63. A conceptualized or virtual machine has not been formulated. One is certainly needed.

64. There is no provision for ease of programming background/foreground

65. Even if there are responses and perhaps answers to these problems, the fact that there are so many questions should not be overlooked.

66. There is no automatic save of data or exchange of data for multi-mode or multi-user programs.

67. A meaningful interpretive language will require 8K bytes.

68. No mention has been made of a macro-assembler language or of whether such is possible at all.

69. This is not a general purpose instruction set; it is too byte and register oriented.

70. This is not an analytic computer; there is no memory reference multiply (as on the LINC).

71. Training should have been consulted.

72. Salesmen should have been consulted.

73. Teradyne should have been consulted.

74. Only one level of indirection is available. Thus scatter branches are only one instruction shorter than on a PDP-8.

75. There appears to be no learning from, or carry over from past experience on other computers like PDP-6, PDP-5, PDP-7 in any visible terms.

76. No provision or plans have been made for page mapping.

77. Any benefit of "bit-efficiency" over a competing machine must be a surplus greater than the margin for error allowed in estimating the core utilization of the anticipated program. This is most unlikely.

78. The instruction set has evolved, or grown like Topsy. Another approach would be to compare several complete sets and to pick the best. This would ensure unity of the result and yet provide valid, objective systems comparisons.

79. The AC can be shifted only one bit at a time.

80. Arithmetic operations and BIN to BCD conversions will be as difficult as on a PDP-8.

81. Picking up arguments after a subroutine call is too complicated. (78)

82. The link bit has become an abstraction, not a hard bit!?

83. The conditional jump on IOT appears to be completely superfluous.
(73)

R.M.M.

Gorden Bell, 2-23-69

84. Should replace ANB with a Push type memory reference instruction to push any word onto stack list.

85. Should consider this as a stack machine expecially for definition or way multiply and divide is handled.